# Справочник API Яндекс.Карт

Версия 2.1.42

13.07.2016

Яндекс

Справочник API Яндекс.Карт. Версия 2.1.42. Версия 2.1
Дата сборки документа: 13.07.2016.
Этот документ является составной частью технической документации Яндекса.
Сайт справки к сервисам Яндекса: http://help.yandex.ru
© 2008—2016 ООО «ЯНДЕКС». Все права защищены.

## Предупреждение об исключительных правах

Яндексу (а также указанному им правообладателю) принадлежат исключительные права на все результаты интеллектуальной деятельности и приравненные к ним средства индивидуализации, используемые при разработке, поддержке и эксплуатации сервиса Справочник API Яндекс.Карт. К таким результатам могут относиться, но не ограничиваясь указанными, программы для ЭВМ, базы данных, изображения, тексты, другие произведения, а также изобретения, полезные модели, товарные знаки, знаки обслуживания, коммерческие обозначения и фирменные наименования. Эти права охраняются в соответствии с Гражданским кодексом РФ и международным правом.

Вы можете использовать сервис Справочник API Яндекс.Карт или его составные части только в рамках полномочий, предоставленных вам Пользовательским соглашением сервиса Справочник API Яндекс.Карт или специального соглашения.

Нарушение требований по защите исключительных прав правообладателя влечет за собой дисциплинарную, гражданско-правовую, административную или уголовную ответственность в соответствии с российским законодательством.

## Контактная информация

ООО «ЯНДЕКС»
http://www.yandex.ru
Тел.: +7 495 739 7000
Email: pr@yandex-team.ru
Главный офис: 119021, Россия, г. Москва, ул. Льва Толстого, д. 16

# Содержание


О справочнике ... 12
Balloon ... 12

**behavior** ... 17
- behavior.DblClickZoom ... 17
- behavior.Drag ... 18
- behavior.LeftMouseButtonMagnifier ... 20
- behavior.MultiTouch ... 22
- behavior.RightMouseButtonMagnifier ... 23
- behavior.RouteEditor ... 25
- behavior.Ruler ... 27
- behavior.ScrollZoom ... 30
- behavior.storage ... 31

Circle ... 33

**clusterer** ... 43

- **clusterer.addon** ... 43
  - clusterer.addon.balloon ... 43
  - clusterer.addon.hint ... 44
- clusterer.Balloon ... 45
- clusterer.Hint ... 47

Clusterer ... 50
ClusterPlacemark ... 59

**collection** ... 70
- collection.Item ... 70

Collection ... 74

**control** ... 79
- control.Button ... 79
- control.FullscreenControl ... 86
- control.GeolocationControl ... 91
- control.ListBox ... 96
- control.ListBoxItem ... 103
- control.Manager ... 107
- control.RouteEditor ... 113
- control.RulerControl ... 118
- control.SearchControl ... 121
- control.storage ... 133
- control.TrafficControl ... 133
- control.TypeSelector ... 140
- control.ZoomControl ... 146

**coordSystem** ... 150
- coordSystem.cartesian ... 150
- coordSystem.geo ... 150

**data** ... 152
- data.Manager ... 152

DomEvent ... 156

**domEvent** ... 158
- domEvent.manager ... 158
- domEvent.MultiPointer ... 161
- domEvent.MultiTouch ... 162
- domEvent.Pointer ... 163
- domEvent.Touch ... 164

**event** ... 166

event.Group ... 166
event.Manager ... 167
event.Mapper ... 170

Event ... 172
formatter ... 175
geocode ... 176
GeocodeResult ... 178
geolocation ... 183

**geometry** ... 186

**geometry.base** ... 186
geometry.base.Circle ... 186

**geometry.base.LinearRing** ... 188
geometry.base.LinearRing.fromEncodedCoordinates ... 191
geometry.base.LinearRing.toEncodedCoordinates ... 192

**geometry.base.LineString** ... 193
geometry.base.LineString.fromEncodedCoordinates ... 195
geometry.base.LineString.toEncodedCoordinates ... 195

geometry.base.Point ... 197

**geometry.base.Polygon** ... 198
geometry.base.Polygon.fromEncodedCoordinates ... 202
geometry.base.Polygon.toEncodedCoordinates ... 202

geometry.base.Rectangle ... 203

geometry.Circle ... 205

**geometry.json** ... 208
geometry.json.circle ... 208
geometry.json.lineString ... 208
geometry.json.Point ... 209
geometry.json.polygon ... 210
geometry.json.rectangle ... 211

**geometry.LineString** ... 212
geometry.LineString.fromEncodedCoordinates ... 216
geometry.LineString.toEncodedCoordinates ... 217

**geometry.pixel** ... 218
geometry.pixel.Circle ... 218
geometry.pixel.LineString ... 219
geometry.pixel.MultiPolygon ... 220
geometry.pixel.Point ... 224
geometry.pixel.Polygon ... 225
geometry.pixel.Rectangle ... 228

geometry.Point ... 230

**geometry.Polygon** ... 232
geometry.Polygon.fromEncodedCoordinates ... 237
geometry.Polygon.toEncodedCoordinates ... 237

geometry.Rectangle ... 239

**geometryEditor** ... 243
geometryEditor.LineString ... 243

**geometryEditor.model** ... 254
geometryEditor.model.ChildLinearRing ... 254
geometryEditor.model.ChildLineString ... 256
geometryEditor.model.ChildVertex ... 258
geometryEditor.model.Edge ... 261

geometryEditor.model.EdgeGeometry ... 263
geometryEditor.model.RootLineString ... 264
geometryEditor.model.RootPolygon ... 266

geometryEditor.Point ... 268
geometryEditor.Polygon ... 270

**geometryEditor.view** ... 282
geometryEditor.view.Edge ... 282
geometryEditor.view.MultiPath ... 283
geometryEditor.view.Path ... 283
geometryEditor.view.Vertex ... 284

**geoObject** ... 286

**geoObject.addon** ... 286
geoObject.addon.balloon ... 286
geoObject.addon.editor ... 287
geoObject.addon.hint ... 288

geoObject.Balloon ... 290
geoObject.Hint ... 292
geoObject.Sequence ... 294

GeoObject ... 300
GeoObjectCollection ... 316
geoQuery ... 324
GeoQueryResult ... 327

**geoXml** ... 363
geoXml.load ... 363

getZoomRange ... 365

**graphics** ... 366

**graphics.style** ... 366
graphics.style.color ... 366
graphics.style.stroke ... 366

Hint ... 369
Hotspot ... 371

**hotspot** ... 374

**hotspot.layer** ... 374

**hotspot.layer.addon** ... 374
hotspot.layer.addon.balloon ... 374
hotspot.layer.addon.hint ... 375

hotspot.layer.Balloon ... 376
hotspot.layer.Hint ... 378
hotspot.layer.Object ... 380

hotspot.Layer ... 384
hotspot.ObjectSource ... 396

**interactivityModel** ... 404
interactivityModel.storage ... 404

**layer** ... 405
layer.storage ... 405

**layer.tile** ... 405
layer.tile.CanvasTile ... 405
layer.tile.DomTile ... 406

**layer.tileContainer** ... 409

layer.tileContainer.CanvasContainer ... 409
layer.tileContainer.DomContainer ... 411

Layer ... 414
LayerCollection ... 420

**layout** ... 426
layout.Image ... 426
layout.ImageWithContent ... 429
layout.storage ... 432

**layout.templateBased** ... 433
layout.templateBased.Base ... 433

LoadingObjectManager ... 438
Map ... 446

**map** ... 469

**map.action** ... 469
map.action.Continuous ... 469
map.action.Manager ... 471
map.action.Single ... 475

**map.addon** ... 479
map.addon.balloon ... 479
map.addon.hint ... 479

map.Balloon ... 481

**map.behavior** ... 483
map.behavior.Manager ... 483

map.Container ... 488
map.Converter ... 492
map.Copyrights ... 493
map.GeoObjects ... 497
map.Hint ... 502

**map.layer** ... 504
map.layer.Manager ... 504

**map.margin** ... 508
map.margin.Accessor ... 508
map.margin.Manager ... 510

**map.pane** ... 513
map.pane.Manager ... 513

map.ZoomRange ... 517

MapEvent ... 519

**mapType** ... 521
mapType.storage ... 521

MapType ... 522
meta ... 523

**modules** ... 526
modules.define ... 526
modules.isDefined ... 529
modules.require ... 529

Monitor ... 531

**multiRouter** ... 535

**multiRouter.driving** ... 535

multiRouter.driving.Path ... 535
multiRouter.driving.PathModel ... 539
multiRouter.driving.Route ... 542
multiRouter.driving.RouteModel ... 546
multiRouter.driving.Segment ... 549
multiRouter.driving.SegmentModel ... 553

multiRouter.Editor ... 557
multiRouter.EditorAddon ... 567

**multiRouter.masstransit** ... 576
multiRouter.masstransit.Path ... 576
multiRouter.masstransit.PathModel ... 581
multiRouter.masstransit.Route ... 583
multiRouter.masstransit.RouteModel ... 587
multiRouter.masstransit.StopModel ... 590
multiRouter.masstransit.TransferSegment ... 591
multiRouter.masstransit.TransferSegmentModel ... 596
multiRouter.masstransit.TransportSegment ... 598
multiRouter.masstransit.TransportSegmentModel ... 602
multiRouter.masstransit.WalkSegment ... 605
multiRouter.masstransit.WalkSegmentModel ... 609

multiRouter.MultiRoute ... 612
multiRouter.MultiRouteModel ... 620

**multiRouter.pedestrian** ... 626
multiRouter.pedestrian.Path ... 626
multiRouter.pedestrian.PathModel ... 630
multiRouter.pedestrian.Route ... 633
multiRouter.pedestrian.RouteModel ... 637
multiRouter.pedestrian.Segment ... 639
multiRouter.pedestrian.SegmentModel ... 644

multiRouter.ViaPoint ... 646
multiRouter.ViaPointModel ... 649
multiRouter.WayPoint ... 652
multiRouter.WayPointModel ... 656

ObjectManager ... 660

**objectManager** ... 672

**objectManager.addon** ... 672
objectManager.addon.clustersBalloon ... 672
objectManager.addon.clustersHint ... 672
objectManager.addon.objectsBalloon ... 673
objectManager.addon.objectsHint ... 674

objectManager.Balloon ... 676
objectManager.ClusterCollection ... 680
objectManager.Hint ... 686
objectManager.ObjectCollection ... 689
objectManager.OverlayCollection ... 695

**option** ... 707
option.Manager ... 707
option.presetStorage ... 711

**overlay** ... 725
overlay.Circle ... 725

**overlay.hotspot** ... 729
overlay.hotspot.Circle ... 729
overlay.hotspot.Placemark ... 730
overlay.hotspot.Polygon ... 730

overlay.hotspot.Polyline ... 731
overlay.hotspot.Rectangle ... 731

**overlay.html** ... 733
overlay.html.Balloon ... 733
overlay.html.Hint ... 738
overlay.html.Placemark ... 743
overlay.html.Rectangle ... 748

overlay.Pin ... 754
overlay.Placemark ... 758
overlay.Polygon ... 764
overlay.Polyline ... 768
overlay.Rectangle ... 772
overlay.storage ... 777

**pane** ... 779
pane.EventsPane ... 779
pane.MovablePane ... 782
pane.StaticPane ... 784

Placemark ... 787
Polygon ... 796
Polyline ... 806
Popup ... 815

**projection** ... 818
projection.Cartesian ... 818
projection.sphericalMercator ... 819
projection.wgs84Mercator ... 819

ready ... 821
Rectangle ... 822

**regions** ... 833
regions.load ... 833

RemoteObjectManager ... 835
route ... 844

**router** ... 848

**router.addon** ... 848

**router.addon.editor** ... 848
router.addon.editor.get ... 848

router.Editor ... 849
router.Path ... 852
router.Route ... 858
router.Segment ... 864
router.ViaPoint ... 867
router.WayPoint ... 872

**shape** ... 878
shape.Circle ... 878
shape.LineString ... 879
shape.MultiPolygon ... 880
shape.Polygon ... 881
shape.Rectangle ... 883
shape.storage ... 884

suggest ... 885
SuggestView ... 885

**template** ... 890
template.filtersStorage ... 890

Template ... 892

templateLayoutFactory ... 894

**traffic** ... 897

**traffic.provider** ... 897
traffic.provider.Actual ... 897
traffic.provider.Archive ... 899
traffic.provider.Forecast ... 904
traffic.provider.storage ... 906

**util** ... 908
util.AsyncStorage ... 908
util.augment ... 911
util.bind ... 912
util.bounds ... 913

**util.cursor** ... 919
util.cursor.Accessor ... 919
util.cursor.Manager ... 920

util.defineClass ... 923
util.Dragger ... 924
util.extend ... 928
util.hd ... 929

**util.math** ... 930
util.math.areEqual ... 930
util.math.cycleRestrict ... 931
util.math.restrict ... 931

util.pixelBounds ... 933
util.requireCenterAndZoom ... 936
util.Storage ... 937

**vow** ... 939
vow.Deferred ... 941
vow.Promise ... 942

**Интерфейсы** ... 947
IBalloon ... 947
IBalloonLayout ... 948
IBalloonManager ... 951
IBalloonOwner ... 954
IBaseCircleGeometry ... 954
IBaseGeometry ... 956
IBaseLinearRingGeometry ... 957
IBaseLineStringGeometry ... 959
IBasePointGeometry ... 961
IBasePolygonGeometry ... 962
IBaseRectangleGeometry ... 965
IBehavior ... 966
ICanvasTile ... 968
IChild ... 970
IChildOnMap ... 971
ICircleGeometry ... 972
ICircleGeometryAccess ... 975
ICollection ... 977
IContainerPane ... 979
IControl ... 981
IControlParent ... 983
ICoordSystem ... 984
ICopyrightsAccessor ... 988
ICopyrightsProvider ... 989
ICustomizable ... 991
IDataManager ... 991
IDomEvent ... 992

IDomEventEmitter ... 995
IDomTile ... 998
IEvent ... 1000
IEventController ... 1002
IEventEmitter ... 1003
IEventGroup ... 1004
IEventManager ... 1005
IEventPane ... 1008
IEventTrigger ... 1010
IEventWorkflowController ... 1011
IExpandableControlLayout ... 1013
IFreezable ... 1016
IGeocodeProvider ... 1017
IGeometry ... 1019
IGeometryEditor ... 1022
IGeometryEditorChildModel ... 1023
IGeometryEditorModel ... 1025
IGeometryEditorRootModel ... 1026
IGeometryJson ... 1026
IGeoObject ... 1027
IGeoObjectCollection ... 1031
IGeoObjectPopupData ... 1036
IGeoObjectSequence ... 1037
IGroupControlLayout ... 1040
IHint ... 1043
IHintManager ... 1045
IHintOwner ... 1046
IHotspot ... 1047
IHotspotLayerObject ... 1050
IHotspotObjectSource ... 1053
IHotspotShape ... 1055
IIterator ... 1059
ILayer ... 1060
ILayout ... 1062
ILinearRingGeometryAccess ... 1066
ILineStringGeometry ... 1072
ILineStringGeometryAccess ... 1074
IMapAction ... 1079
IMapObjectCollection ... 1080
IMapState ... 1081
IMultiRouteModelJson ... 1082
IMultiRouteParams ... 1083
IMultiRouteReferencePoint ... 1084
IOptionManager ... 1085
IOverlay ... 1088
IPane ... 1093
IPanorama ... 1095
IPanoramaConnection ... 1097
IPanoramaGraph ... 1098
IPanoramaGraphEdge ... 1099
IPanoramaGraphNode ... 1099
IPanoramaMarker ... 1100
IPanoramaMarkerIcon ... 1101
IPanoramaMarkerIconSet ... 1102
IPanoramaThoroughfare ... 1103
IPanoramaTileLevel ... 1104
IPanoramaTransition ... 1104
IParentOnMap ... 1105
IPixelCircleGeometry ... 1106
IPixelGeometry ... 1107
IPixelLineStringGeometry ... 1109
IPixelMultiPolygonGeometry ... 1111

IPixelPointGeometry ... 1114
IPixelPolygonGeometry ... 1115
IPixelRectangleGeometry ... 1118
IPointGeometry ... 1120
IPointGeometryAccess ... 1121
IPolygonGeometry ... 1122
IPolygonGeometryAccess ... 1125
IPopup ... 1131
IPopupManager ... 1134
IPositioningContext ... 1137
IProjection ... 1139
IPromiseProvider ... 1140
IRatioMap ... 1141
IRectangleGeometry ... 1141
IRectangleGeometryAccess ... 1143
ISearchControlLayout ... 1145
ISearchProvider ... 1149
ISelectableControl ... 1151
ISelectableControlLayout ... 1153
IShape ... 1156
ISuggestProvider ... 1158
ISuggestViewLayout ... 1160
ITrafficControlLayout ... 1160
ITrafficProvider ... 1163
ITransportProperties ... 1164
IZoomControlLayout ... 1165

# О справочнике

Справочник содержит описание программного интерфейса **JavaScript API** версии 2.1.42.

Целевой аудиторией справочника являются разработчики сайтов, которые хотят использовать интерактивные Яндекс.Карты на своих веб-страницах.

Справочник содержит описания открытых классов и методов API и рассчитан на разработчиков, знакомых с JavaScript и сервисом Яндекс.Карты. Классы перечислены в алфавитном порядке.

# Balloon

Расширяет **IBalloon**, **Popup**.

Балун — всплывающее облако, которое может отображать любое HTML-содержимое. Обычно балун существует на карте в единственном экземпляре, управление им происходит через специальные менеджеры (карты, геообъектов, слоёв активных областей и т.д.). Не создавайте их самостоятельно, если в этом нет необходимости.

Конструктор | Поля | События | Методы

## Конструктор

```
Balloon(map[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Ссылка на объект карты. |
| options | — | Тип: Object<br><br>Опции. |
| options.autoPan | true | Тип: Boolean<br><br>Сдвигать карту, чтобы отобразить открывшийся балун. |
| options.autoPanCheckZoomRange | false | Тип: Boolean<br><br>Включает автомасштабирование при невозможности отобразить карту после перемещения на том же масштабе. |
| options.autoPanDuration | 500 | Тип: Number<br><br>Длительность перемещения к точке балуна (в мс). |
| options.autoPanMargin | 34 | Тип: Number \| Number[] |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Отступ или отступы от краев видимой области карты при выполнении действия autoPan. Значение может быть задано в виде одного числа (равный отступ для всех сторон), в виде двух чисел (для вертикальные и горизонтальные отступы) и в виде четырех чисел (в порядке: верхний, правый, нижний и левый отступ). Следует иметь в виду, данное значение будет суммироваться со значением рассчитыванным в менеджере отступов **map.margin.Manager**. |
| options.autoPanUseMapMargin | true | Тип: Boolean<br><br>Нужно ли учитывать отступы карты **map.margin.Manager** при выполнении действия autoPan. |
| options.closeButton | true | Тип: Boolean<br><br>Флаг наличия кнопки закрытия. |
| options.contentLayout | — | Тип: Function \| String<br><br>Макет для содержимого балуна. (Тип: конструктор объекта с интерфейсом **ILayout** или ключ макета). |
| options.layout | islands#balloon | Тип: Function \| String<br><br>Внешний макет балуна. (Тип: конструктор объекта с интерфейсом **ILayout** или ключ макета). |
| options.maxHeight | — | Тип: Number<br><br>Максимальная высота, в пикселах. |
| options.maxWidth | — | Тип: Number<br><br>Максимальная ширина, в пикселах. |
| options.minHeight | — | Тип: Number<br><br>Минимальная высота, в пикселах. |
| options.minWidth | — | Тип: Number<br><br>Минимальная ширина, в пикселах. |
| options.offset | — | Тип: Number[]<br><br>Дополнительное смещение позиции относительно точки привязки. |
| options.pane | 'balloon' | Тип: String<br><br>Ключ пейна, в который помещается оверлей балуна. |
| options.panelContentLayout | null | Тип: Function \| String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Макет для содержимого балуна в режиме панели. Если опция не задана, то используется значение опции contentLayout. (Тип: конструктор объекта с интерфейсом **ILayout** или ключ макета). |
| options.panelMaxHeightRatio | — | Тип: Number<br><br>Максимальная высота балуна-панели. Задаётся как коэфициент отношения к высоте карты: числом от 0 до 1. |
| options.panelMaxMapArea | — | Тип: Number<br><br>Максимальная площадь карты, при которой балун будет отображаться в виде панели. Можно отключить режим панели, задав значение *0*, и наоборот, всегда отображать балун в виде панели, задав значение *Infinity*. |
| options.shadow | true | Тип: Boolean<br><br>Флаг наличия тени. |
| options.shadowLayout | — | Тип: Function \| String<br><br>Макет тени. (Тип: конструктор объекта с интерфейсом **ILayout** или ключ макета). |
| options.shadowOffset | — | Тип: Number[]<br><br>Дополнительное смещение позиции тени относительно точки привязки. |

* Обязательный параметр/опция.

**Пример:**

```
// Создание независимого экземпляра балуна и отображение его в центре карты.
var balloon = new ymaps.Balloon(myMap);
// Здесь родительскими устанавливаются опции карты,
// где содержатся значения по умолчанию для обязательных опций.
balloon.options.setParent(myMap.options);
// Открываем балун в центре карты:
balloon.open(myMap.getCenter());
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **autopanbegin** | Начало автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. |
| **autopanend** | Окончание автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. |
| **beforeuserclose** | Событие, предшествующее Balloon.userclose. Позволяет отменить пользовательское действие посредством вызова метода preventDefault. Экземпляр класса **Event**. |
| **close** | Закрытие инфо-объекта.<br>Унаследовано от **IPopup**. |
| **open** | Открытие инфо-объекта.<br>Унаследовано от **IPopup**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **userclose** | Закрытие балуна пользователем. Экземпляр класса **Event**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **autoPan**() | **vow.Promise** | Перемещает карту таким образом, чтобы балун оказался видимым.<br>Унаследован от **IBalloon**. |
| **close**([force]) | **vow.Promise** | Закрывает инфо-объект.<br>Унаследован от **IPopup**. |
| **getData**() | | Возвращает данные инфо-объекта.<br>Унаследован от **IPopup**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от **IPopup**. |
| **getOverlaySync**() | **IOverlay** | Возвращает оверлей, если тот существует.<br>Унаследован от **IPopup**. |
| **getPosition**() | | Возвращает координаты инфо-объекта.<br>Унаследован от **IPopup**. |
| **isOpen**() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от **IPopup**. |
| **open**([position[, data]]) | **vow.Promise** | Открывает инфо-объект в указанной позиции. Если инфо-объект уже открыт, перемещает его в указанную точку. Формат и суть координат определяется проекцией **IProjection**, которая содержится в опциях. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IPopup**. |
| **setData**(data) | **vow.Promise** | Задаёт инфо-объекту новые данные.<br>Унаследован от **IPopup**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopup**. |

## Описание событий

### autopanbegin

Начало автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**.

### autopanend

Окончание автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**.

### beforeuserclose

Событие, предшествующее Balloon.userclose. Позволяет отменить пользовательское действие посредством вызова метода preventDefault. Экземпляр класса **Event**.

### userclose

Закрытие балуна пользователем. Экземпляр класса **Event**.

# behavior

## behavior.DblClickZoom

Расширяет **IBehavior**.

Поведение "масштабирование карты двойным щелчком кнопки мыши".

Конструктор | Поля | События | Методы

### Конструктор

```
behavior.DblClickZoom([options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции. |
| options.centering | true | Тип: Boolean<br><br>true — при двойном щелчке мышью карта масштабируется так, что точка под курсором мыши становится новым центром карты; false — точка под курсором мыши сохраняет свое положение при масштабировании двойным щелчком. |
| options.duration | 200 | Тип: Number<br><br>Время анимации масштабирования двойным щелком кнопки мыши (0 — анимации нет). |
| options.useMapMargin | true | Тип: Boolean<br><br>Нужно ли при центрировании учитывать отступы, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

### События

| Имя | Описание |
|---|---|
| **disable** | Выключение поведения.<br><br>Унаследовано от **IBehavior**. |

| Имя | Описание |
|---|---|
| **enable** | Включение поведения.<br>Унаследовано от **IBehavior**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **disable**() | | Выключает поведение.<br>Унаследован от **IBehavior**. |
| **enable**() | | Включает поведение.<br>Унаследован от **IBehavior**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **isEnabled**() | Boolean | Проверяет, включено ли поведение.<br>Унаследован от **IBehavior**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

# behavior.Drag

Расширяет **IBehavior**.

Поведение "перетаскивание карты с помощью мыши либо одиночного касания".

Конструктор | Поля | События | Методы

## Конструктор

```
behavior.Drag([options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br>Опции. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.actionCursor | 'grabbing' | Тип: String<br><br>Курсор для поведения **behavior.Drag** при перетаскивании карты. |
| options.cursor | 'grab' | Тип: String<br><br>Курсор для поведения **behavior.Drag** при наведении на карту. |
| options.inertia | true | Тип: Boolean<br><br>Включает кинетическую инерцию движения в конце перетаскивания. |
| options.inertiaDuration | 400 | Тип: Number \| String<br><br>Продолжительность инерции в мс. Строковое значение 'auto' устанавливает продолжительность инерции пропорциональным расстоянию. |
| options.tremor | 2 | Тип: Integer<br><br>Минимальное смещение курсора после нажатия кнопки мыши, после которого начинает двигаться карта. |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **disable** | Выключение поведения.<br>Унаследовано от **IBehavior**. |
| **enable** | Включение поведения.<br>Унаследовано от **IBehavior**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **disable**() | | Выключает поведение.<br>Унаследован от **IBehavior**. |
| **enable**() | | Включает поведение.<br>Унаследован от **IBehavior**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **isEnabled**() | Boolean | Проверяет, включено ли поведение.<br>Унаследован от **IBehavior**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

# behavior.LeftMouseButtonMagnifier

Расширяет **IBehavior**.

Поведение "масштабирование карты при выделении области левой кнопкой мыши".

Конструктор | Поля | События | Методы

## Конструктор

```
behavior.LeftMouseButtonMagnifier([options])
```

Создает поведение "масштабирование карты при выделении области левой кнопкой мыши".

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции. |
| options.actionCursor | 'crosshair' | Тип: String<br><br>Курсор во время выбора области для увеличения при включенном поведении **behavior.LeftMouseButtonMagnifier**. Список всех доступных курсоров описан в Менеджере курсоров. |
| options.cursor | 'zoom' | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Курсор при включенном поведении **behavior.LeftMouseButtonMagnifier**. Список всех доступных курсоров описан в Менеджере курсоров. |
| `options.duration` | 300 | Тип: Number<br><br>Продолжительность анимации увеличения области при использовании поведения **behavior.LeftMouseButtonMagnifier**, мс. |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **disable** | Выключение поведения.<br>Унаследовано от **IBehavior**. |
| **enable** | Включение поведения.<br>Унаследовано от **IBehavior**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **disable**() | | Выключает поведение.<br>Унаследован от **IBehavior**. |
| **enable**() | | Включает поведение.<br>Унаследован от **IBehavior**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **isEnabled**() | Boolean | Проверяет, включено ли поведение.<br>Унаследован от **IBehavior**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

# behavior.MultiTouch

Расширяет **IBehavior**.

Поведение "масштабирование карты мультисенсорным касанием".

Конструктор | Поля | События | Методы

## Конструктор

```
behavior.MultiTouch([options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br>Опции. |
| options.tremor | 2 | Тип: Number<br>Минимальное движение по экрану устройства (в пикселях), при котором сработает поведение **behavior.MultiTouch**. |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **disable** | Выключение поведения.<br>Унаследовано от **IBehavior**. |
| **enable** | Включение поведения.<br>Унаследовано от **IBehavior**. |

| Имя | Описание |
|---|---|
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **disable**() | | Выключает поведение.<br>Унаследован от **IBehavior**. |
| **enable**() | | Включает поведение.<br>Унаследован от **IBehavior**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **isEnabled**() | Boolean | Проверяет, включено ли поведение.<br>Унаследован от **IBehavior**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

# behavior.RightMouseButtonMagnifier

Расширяет **IBehavior**.

Поведение "масштабирование карты при выделении области правой кнопкой мыши".

Конструктор | Поля | События | Методы

## Конструктор

```
behavior.RightMouseButtonMagnifier([options])
```

Создает поведение "масштабирование карты при выделении области правой кнопкой мыши".

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br>Опции. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.actionCursor | 'crosshair' | Тип: String<br><br>Курсор во время выбора области для увеличения при включенном поведении **behavior.RightMouseButtonMagnifier**. Список всех доступных курсоров описан в Менеджере курсоров. |
| options.duration | 300 | Тип: Number<br><br>Продолжительность анимации при выделении области при помощи поведения **behavior.RightMouseButtonMagnifier**, мс. |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **disable** | Выключение поведения.<br>Унаследовано от **IBehavior**. |
| **enable** | Включение поведения.<br>Унаследовано от **IBehavior**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **disable**() | | Выключает поведение.<br>Унаследован от **IBehavior**. |
| **enable**() | | Включает поведение.<br>Унаследован от **IBehavior**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **isEnabled**() | Boolean | Проверяет, включено ли поведение.<br>Унаследован от **IBehavior**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

# behavior.RouteEditor

Расширяет **IBehavior**.

Поведение "редактор маршрутов".

Конструктор | Поля | События | Методы

## Конструктор

```
behavior.RouteEditor()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **disable** | Выключение поведения.<br>Унаследовано от **IBehavior**. |
| **enable** | Включение поведения.<br>Унаследовано от **IBehavior**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель. |

| Имя | Описание |
|---|---|
| | Унаследовано от **IChild**. |
| **routechange** | Событие смены маршрута вследствие вызова метода setState или включения/выключения поведения. Старый и новый маршрут можно получить через свойства события 'oldRoute' и 'newRoute' соответственно. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **disable**() | | Выключает поведение.<br>Унаследован от **IBehavior**. |
| **enable**() | | Включает поведение.<br>Унаследован от **IBehavior**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getRoute**() | **router.Route** | Возвращает маршрут. |
| **getState**() | String | Возвращает текущее состояние редактора маршрутов в закодированном виде. |
| **isEnabled**() | Boolean | Проверяет, включено ли поведение.<br>Унаследован от **IBehavior**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |
| **setState**(state) | | Восстанавливает состояние редактора маршрутов из закодированной строки. При установлении нового маршрута будет вызвано событие routechange. |

## Описание событий

### routechange

Событие смены маршрута вследствие вызова метода setState или включения/выключения поведения. Старый и новый маршрут можно получить через свойства события 'oldRoute' и 'newRoute' соответственно.

## Описание методов

### getRoute

```
{router.Route} getRoute()
```

**Возвращает** маршрут.

## getState

```
{String} getState()
```

**Возвращает** текущее состояние редактора маршрутов в закодированном виде.

## setState

```
setState(state)
```

Восстанавливает состояние редактора маршрутов из закодированной строки. При установлении нового маршрута будет вызвано событие routechange.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| state* | — | Тип: String \| null<br><br>Закодированное состояние редактора маршрутов. Состояние задается в следующем формате: rt=точка1~точка2~точка3~точка4...&via=индексы-via-points Например, если закодированная строка выглядит так: "rt=50,30~42,35~45,32~40,30&via=1,2", это значит что путевыми точками маршрута должны быть "50,30" и "40,30", а промежуточными "42,35" и "45,32". Если параметр via не указывать, то маршрут будет состоять только из путевых точек. |

* Обязательный параметр/опция.

**Пример:**

```
var behavior = map.behaviors.get('routeEditor');
behavior.events.add('routechange', function (e) {
   var newRoute = e.get('newRoute');
   alert(newRoute.getLength()); // длина нового маршрута
   setTimeout(function () {
       // удалить маршрут
       behavior.setState(null);
   })
});
behavior.setState('rt=55.874872,37.562677~55.92517867214157,37.62725433916199~55
.920011490602526,37.6629269628905&via=1');
```

# behavior.Ruler

Расширяет **IBehavior**.

Поведение "Линейка". Позволяет отмечать на карте точки и отображать расстояния между ними.

Конструктор | Поля | События | Методы

## Конструктор

```
behavior.Ruler()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| geometry | geometry.LineString | Геометрия поведения "Линейка". |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| disable | Выключение поведения.<br>Унаследовано от IBehavior. |
| enable | Включение поведения.<br>Унаследовано от IBehavior. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| close() | Boolean | Удаляет все точки линейки. Если текущее количество точек больше двух, то будет выведен запрос на подтверждение действия. |
| disable() | | Выключает поведение.<br>Унаследован от IBehavior. |
| enable() | | Включает поведение.<br>Унаследован от IBehavior. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |

| Имя | Возвращает | Описание |
|---|---|---|
| **getState**() | String | Состояние линейки описывается строкой, состоящей из последовательностей, разделяемых символом "~". Каждая последовательность представляет собой подстроку вида "долгота,широта" и описывает приращение координат относительно предыдущей точки линейки. |
| **isEnabled**() | Boolean | Проверяет, включено ли поведение.<br>Унаследован от **IBehavior**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |
| **setState**(state) | | Восстанавливает состояние линейки из закодированной строки. Смотрите behavior.Ruler.getState. |

## Описание полей

### geometry

```
{geometry.LineString} geometry
```

Геометрия поведения "Линейка".

**Пример:**

```
// Задаем координаты первой точки линейки.
myMap.behaviors.get('ruler').geometry.set(0, [0, 0]);
```

## Описание методов

### close

```
{Boolean} close()
```

Удаляет все точки линейки. Если текущее количество точек больше двух, то будет выведен запрос на подтверждение действия.

**Возвращает** true, если действие произведено успешно.

### getState

```
{String} getState()
```

Состояние линейки описывается строкой, состоящей из последовательностей, разделяемых символом "~". Каждая последовательность представляет собой подстроку вида "долгота,широта" и описывает приращение координат относительно предыдущей точки линейки.

**Возвращает** текущее состояние линейки в закодированном виде.

### setState

```
setState(state)
```

Восстанавливает состояние линейки из закодированной строки. Смотрите behavior.Ruler.getState.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| state* | — | Тип: String<br><br>Закодированное состояние линейки. |

* Обязательный параметр/опция.

## behavior.ScrollZoom

Расширяет **IBehavior**.

Поведение "масштабирование карты колесом мыши".

Конструктор | Поля | События | Методы

### Конструктор

```
behavior.ScrollZoom([options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции. |
| options.maximumDelta | 5 | Тип: Number<br><br>Максимальное возможное изменение, на которое можно изменить зум карты при одном событии скролла (непрерывным вращением колеса мыши). При достижении этого лимита карта перерисовывается (показываются тайлы для достигнутого уровня масштаба). |
| options.speed | 5 | Тип: Number<br><br>Скорость увеличения/уменьшения масштаба карты при включенном поведении **behavior.ScrollZoom**, уровней масштаба в секунду. |

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **disable** | Выключение поведения.<br>Унаследовано от **IBehavior**. |
| **enable** | Включение поведения.<br>Унаследовано от **IBehavior**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **disable**() | | Выключает поведение.<br>Унаследован от **IBehavior**. |
| **enable**() | | Включает поведение.<br>Унаследован от **IBehavior**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **isEnabled**() | Boolean | Проверяет, включено ли поведение.<br>Унаследован от **IBehavior**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

# behavior.storage

Статический объект.

Экземпляр класса **util.Storage**

Хранилище классов поведений карты. Добавление нового поведения на карту производится через это хранилище.

По умолчанию в хранилище добавлены следующие поведения:

- "drag" — перемещение карты при нажатой левой кнопке мыши либо одиночным касанием **behavior.Drag**;
- "scrollZoom" — изменение масштаба колесом мыши **behavior.ScrollZoom** (только для десктопных браузеров);
- "dblClickZoom" — масштабирование карты двойным щелчком кнопки мыши **behavior.DblClickZoom**;
- "multiTouch" — масштабирование карты двойным касанием (например, пальцами на сенсорном экране) **behavior.MultiTouch** (только для мобильных браузеров);
- "rightMouseButtonMagnifier" — увеличение области, выделенной правой кнопкой мыши (только для настольных браузеров), **behavior.RightMouseButtonMagnifier** (только для десктопных браузеров);
- "leftMouseButtonMagnifier" — увеличение области, выделенной левой кнопкой мыши либо одиночным касанием, **behavior.LeftMouseButtonMagnifier**;
- "ruler" — измерение расстояния **behavior.Ruler**.
- "routeEditor" — редактор маршрутов **behavior.RouteEditor**;

Методы

```
behavior.storage()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |

# Circle

Расширяет **GeoObject**.

Круг. Представляет собой геообъект с геометрией **geometry.Circle**.

**См.: GeoObject geometry.Circle**

Конструктор | Поля | События | Методы

## Конструктор

```
Circle(geometry[, properties[, options]])
```

Создает экземпляр круга.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **ICircleGeometry** \| Number[][] \| Object<br><br>Ссылка на объект геометрии точки или массив, первый элемент которого координаты центра круга, второй — радиус в метрах, или объект с параметрами геометрии. |
| properties | — | Тип: Object \| **IDataManager**<br><br>Данные круга. Соответствуют данным класса **GeoObject**. Могут задаваться как экземпляр класса, реализующего интерфейс **IDataManager**, либо в виде хэша. При значениях опций по умолчанию кругу трактуются следующие поля данных:<br>• hintContent — содержимое всплывающей подсказки круга;<br>• balloonContent — содержимое балуна круга;<br>• balloonContentHeader — содержимое заголовка балуна круга;<br>• balloonContentBody — содержимое основной части балуна круга;<br>• balloonContentFooter — содержимое нижней части балуна круга.<br><br>Поле balloonContent является кратким обозначением для поля balloonContentBody, но при одновременном задании balloonContentBody более приоритетен. Также вы можете дополнить данные круга своими собственными полями и использовать их везде, где это возможно. Например в макете круга или макете балуна. |
| options | — | Тип: Object |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опции круга. Используя этот параметр можно задавать как опции самого круга, так и опции его составных частей:<br>• Опции балуна круга с префиксом `balloon`.<br>• Опции всплывающей подсказки круга с префиксом `hint`.<br>• Опции геометрии могут задаваться без префикса. Смотрите описание класса **IGeometry** геометрии **geometry.Circle**. |
| `options.circleOverlay` | "default#circle" | Тип: String \| Function<br><br>Ключ-идентификатор из хранилища **overlay.storage** или класс оверлея. Функция-генератор принимает три параметра:<br>• geometry: **IPixelCircleGeometry** — непосредственно пиксельная геометрия;<br>• data: Object — данные оверлея;<br>• options: Object — опции оверлея.<br><br>И возвращает **vow.Promise**. |
| `options.cursor` | "pointer" | Тип: String<br><br>Вид курсора над кругом. |
| `options.draggable` | false | Тип: Boolean<br><br>Определяет возможность перетаскивания круга. |
| `options.fill` | true | Тип: Boolean<br><br>Наличие заливки фигуры. |
| `options.fillColor` | "0066ff99" | Тип: String<br><br>Цвет заливки. |
| `options.fillImageHref` | — | Тип: String<br><br>Фоновое изображение. При включении данной опции в режиме заливки **stretch** значение fillColor игнорируется. |
| `options.fillMethod` | 'stretch' | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Тип заливки фоном. Может принимать одно из двух значений:<br>• stretch — Фоновое изображение растягивается по размеру оверлея.<br>• tile — Фоновое изображение повторяется без изменений размера. Аналог background-repeat в css. Можно использовать для заливки фигуры неким шаблоном. |
| options.fillOpacity | 1 | Тип: Number<br><br>Прозрачность заливки. |
| options.hasBalloon | true | Тип: Boolean<br><br>Определяет наличие поля balloon у круга. |
| options.hasHint | true | Тип: Boolean<br><br>Определяет наличие поля hint у круга. |
| options.hideIconOnBalloonOpen | true | Тип: Boolean<br><br>Скрывать иконку при открытии балуна. |
| options.interactiveZIndex | false | Тип: Boolean<br><br>Включает режим автоматического изменения z-index круга в зависимости от его состояния. |
| options.interactivityModel | "default#geoObject" | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.opacity | 1 | Тип: Number<br><br>Прозрачность. |
| options.openBalloonOnClick | true | Тип: Boolean<br><br>Определяет, показывать ли балун при щелчке на круге. |
| options.openEmptyBalloon | false | Тип: Boolean<br><br>Определяет, показывать ли пустой балун при щелчке на круге. |
| options.openEmptyHint | false | Тип: Boolean<br><br>Определяет, показывать ли пустой хинт при наведении указателя мыши на круг. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.openHintOnHover | true | Тип: Boolean<br><br>Определяет, показывать ли хинт при наведении указателя мыши на круг. |
| options.outline | true | Тип: Boolean<br><br>Наличие обводки круга. |
| options.pane | "areas" | Тип: String<br><br>Ключ пейна, в который помещается оверлей круга. |
| options.strokeColor | "0066ffff" | Тип: String \| String[]<br><br>Цвет линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeOpacity | 1 | Тип: Number \| Number[]<br><br>Прозрачность линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeStyle | — | Тип: String \| Object \| String[] \| Object[]<br><br>Стиль линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeWidth | 1 | Тип: Number \| Number[]<br><br>Толщина линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.syncOverlayInit | false | Тип: Boolean<br><br>Включает синхронное добавление оверлея на карту. По умолчанию добавление оверлея осуществляется асинхронно, что позволяет предотвратить зависания браузера при добавлении на карту большого числа геообъектов. Однако, асинхронное добавление не позволяет получать доступ к оверлею сразу после добавления круга на карту. |
| options.useMapMarginInDragging | true | Тип: Boolean<br><br>При перетаскивании объекта к краю карты происходит автоматическое изменение центра карты. Нужно ли учитывать отступы карты при автоматическом смещении центра карты **map.margin.Manager**. |
| options.visible | true | Тип: Boolean<br><br>Определяет видимость круга. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.zIndex | — | Тип: Number<br><br>z-index круга в обычном состоянии. Наименее приоритетный. |
| options.zIndexActive | — | Тип: Number<br><br>z-index круга с открытым балуном. Наиболее приоритетный. |
| options.zIndexDrag | — | Тип: Number<br><br>z-index круга при перетаскивании. |
| options.zIndexHover | — | Тип: Number<br><br>z-index круга при наведении на него указателя мыши. |

* Обязательный параметр/опция.

**Пример:**

```
// Создаем геодезический круг радиусом 1000 километров.
var circle = new ymaps.Circle([[50, 75], 1000000], {}, {
    geodesic: true
});
// Добавляем круг на карту.
myMap.geoObjects.add(circle);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **balloon** | **geoObject.Balloon** | Балун геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **editor** | Null | Редактор геометрии "Круг" на данный момент не реализован. |
| **events** | **event.Manager** | Менеджер событий.<br><br>Унаследовано от **GeoObject**. |
| **geometry** | **geometry.Circle** | Геометрия типа "Круг". |
| **hint** | **geoObject.Hint** | Хинт геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **options** | **option.Manager** | Менеджер опций геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **properties** | **data.Manager** | Менеджер данных геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **state** | **data.Manager** | Состояние геообъекта. Определяется следующими полями:<br><br>• active: Boolean — признак того, что на геообъекте открыт балун;<br>• hover: Boolean — признак того, что в данный момент на геообъект наведен указатель мыши;<br>• drag: Boolean — признак того, что геообъект перетаскивается.<br><br>Унаследовано от **GeoObject**. |

## События

| Имя | Описание |
|---|---|
| **balloonclose** | Закрытие балуна. Экземпляр класса **Event**.<br><br>Унаследовано от **GeoObject**. |
| **balloonopen** | Открытие балуна на геообъекте. Экземпляр класса **Event**.<br><br>Унаследовано от **GeoObject**. |
| **beforedrag** | Событие, предшествующее событию drag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br><br>Имена методов, доступных через метод Event.callMethod:<br><br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br><br>Если у данного события вызвать метод Event.preventDefault, то последующее событие drag будет отменено.<br><br>Унаследовано от **GeoObject**. |
| **beforedragstart** | Событие, предшествующее событию dragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br><br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие dragstart, будет отменено.<br><br>Унаследовано от **GeoObject**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |

| Имя | Описание |
|---|---|
| **drag** | Перетаскивание геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragend** | Завершение перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragstart** | Начало перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **editorstatechange** | Изменение состояния редактора геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent — оригинальное событие редактора геометрии.<br>Унаследовано от **GeoObject**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **hintopen** | Открытие хинта на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

| Имя | Описание |
| --- | --- |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |

| Имя | Описание |
|---|---|
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание полей

### editor

```
{Null} editor
```

Редактор геометрии "Круг" на данный момент не реализован.

### geometry

```
{geometry.Circle} geometry
```

Геометрия типа "Круг".

# clusterer

## clusterer.addon

### clusterer.addon.balloon

**Примечание:**
Конструктор класса clusterer.addon.balloon скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Статический объект.

Методы

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **get**(clusterer) | **IPopupManager** | Возвращает менеджер балуна кластеризатора. |

### Описание методов

#### get

```
{IPopupManager} get(clusterer)
```

**Возвращает** менеджер балуна кластеризатора.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| clusterer* | — | Тип: **Clusterer**<br><br>Кластеризатор |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.clusterer.addon.balloon.get(clusterer)
```

# clusterer.addon.hint

**Примечание:**
Конструктор класса clusterer.addon.hint скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Статический объект.

Методы

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **get**(clusterer) | **IPopupManager** | Возвращает менеджер хинта кластеризатора. |

## Описание методов

### get

```
{IPopupManager} get(clusterer)
```

**Возвращает** менеджер хинта кластеризатора.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| clusterer* | — | Тип: **Clusterer**<br><br>Кластеризатор |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.clusterer.addon.hint.get(clusterer)
```

# clusterer.Balloon

Расширяет **IBalloonManager**.

Менеджер балуна кластеризатора. Позволяет управлять балуном кластера, открывать и скрывать его. Использует внутри себя менеджер балуна карты **map.Balloon**. Кластеризаторы содержат экземпляр этого класса, доступный как myClusterer.balloon. Не создавайте экземпляры этого класса без необходимости.

**См.: Balloon**

Конструктор | Поля | События | Методы

## Конструктор

```
clusterer.Balloon(clusterer)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| clusterer * | — | Тип: **Clusterer**<br><br>Кластеризатор. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **autopanbegin** | Начало автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **autopanend** | Окончание автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **beforeuserclose** | Событие, предшествующее Balloon.userclose. Позволяет отменить пользовательское действие посредством вызова метода preventDefault. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |

| Имя | Описание |
|---|---|
| close | Закрытие балуна.<br>• target — ссылка на менеджер балуна кластеризатора;<br>• cluster — ссылка на объект-кластер.<br>Экземпляр класса Event. |
| open | Открытие балуна на кластере. Имена полей, доступных через метод Event.get:<br>• target — ссылка на менеджер балуна кластеризатора;<br>• cluster — ссылка на объект-кластер.<br>Экземпляр класса Event. |
| userclose | Закрытие балуна пользователем. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• target — ссылка на объект IBalloonOwner.<br>Унаследовано от IBalloonManager. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| autoPan() | vow.Promise | Перемещает карту таким образом, чтобы балун оказался видимым.<br>Унаследован от IBalloonManager. |
| close([force]) | vow.Promise | Закрывает инфо-объект.<br>Унаследован от IPopupManager. |
| destroy() | | Отключает менеджер инфо-объекта.<br>Унаследован от IPopupManager. |
| getData() | Object \| null | Возвращает данные инфо-объекта или `null`.<br>Унаследован от IPopupManager. |
| getOptions() | IOptionManager \| null | Возвращает менеджер опций или `null`.<br>Унаследован от IPopupManager. |
| getOverlay() | vow.Promise | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от IPopupManager. |
| getOverlaySync() | IOverlay \| null | Возвращает оверлей, если тот существует.<br>Унаследован от IPopupManager. |
| getPosition() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от IPopupManager. |
| isOpen() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от IPopupManager. |

| Имя | Возвращает | Описание |
|---|---|---|
| **open**([position[, data[, options]]]) | **vow.Promise** | Открывает инфо-объект в указанной позиции.<br>Унаследован от **IPopupManager**. |
| **setData**(data) | **vow.Promise** | Задаёт инфо-объекту новые данные.<br>Унаследован от **IPopupManager**. |
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopupManager**. |

## Описание событий

### close

Закрытие балуна.

- target — ссылка на менеджер балуна кластеризатора;
- cluster — ссылка на объект-кластер.

Экземпляр класса **Event**.

### open

Открытие балуна на кластере. Имена полей, доступных через метод Event.get:

- target — ссылка на менеджер балуна кластеризатора;
- cluster — ссылка на объект-кластер.

Экземпляр класса **Event**.

**Пример:**

```
clusterer.balloon.events.add('open', function (e) {
    var clusterPlacemark = e.get('cluster');
});
```

# clusterer.Hint

Расширяет **IHintManager**.

Менеджер хинта кластеризатора. Позволяет управлять хинтом кластара, открывать и скрывать его. Использует внутри себя менеджер хинта карты **map.Hint**. Кластеризаторы содержат экземпляр этого класса, доступный как myClusterer.hint. Не создавайте экземпляры этого класса без необходимости.

**См.: Hint**

Конструктор | Поля | Методы

## Конструктор

```
clusterer.Hint(clusterer)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| clusterer * | — | Тип: **Clusterer**<br><br>Кластеризатор. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **close**([force]) | **vow.Promise** | Закрывает инфо-объект.<br>Унаследован от **IPopupManager**. |
| **destroy**() | | Отключает менеджер инфо-объекта.<br>Унаследован от **IPopupManager**. |
| **getData**() | Object \| null | Возвращает данные инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **getOptions**() | **IOptionManager** \| null | Возвращает менеджер опций или `null`.<br>Унаследован от **IPopupManager**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от **IPopupManager**. |
| **getOverlaySync**() | **IOverlay** \| null | Возвращает оверлей, если тот существует.<br>Унаследован от **IPopupManager**. |
| **getPosition**() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **isOpen**() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от **IPopupManager**. |
| **open**([position[, data[, options]]]) | **vow.Promise** | Открывает инфо-объект в указанной позиции.<br>Унаследован от **IPopupManager**. |
| **setData**(data) | **vow.Promise** | Задаёт инфо-объекту новые данные.<br>Унаследован от **IPopupManager**. |
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |

| Имя | Возвращает | Описание |
| --- | --- | --- |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopupManager**. |

# Clusterer

Расширяет **IChildOnMap**, **ICustomizable**, **IEventEmitter**, **IParentOnMap**.

Кластеризатор геообъектов. Кластеризует объекты в видимой области карты. Если объект не попадает в видимую область карты, он не будет добавлен на карту. Обратите внимание, что кластеризатор не поддерживает реакцию на изменение координат объектов (программно, или в результате драга). Если у объекта в кластеризаторе необходимо изменить координаты, следует удалить объект из кластеризатора, поменять его геометрию, а затем добавить обратно.

**См.: ClusterPlacemark**

Конструктор | Поля | События | Методы

## Конструктор

```
Clusterer([options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции. Опции для дочерних объектов-кластеров задаются с префиксом "cluster". См. **ClusterPlacemark**. |
| options.gridSize | 64 | Тип: Number<br><br>Размер ячейки кластеризации в пикселях. Значение должно быть равно 2^n (в область 256 на 256 пикселей должно умещаться ровное количество ячеек). |
| options.groupByCoordinates | false | Тип: Boolean<br><br>Специальный режим работы кластеризатора при котором кластеры образуются только из геобъектов с одинаковыми координатами. |
| options.hasBalloon | true | Тип: Boolean<br><br>Флаг наличия у кластеризатора поля .balloon. Если при клике на кластер не нужно открывать балун, рекомендуется установить эту опцию в значение false, чтобы избежать лишних инициализаций. |
| options.hasHint | true | Тип: Boolean<br><br>Флаг наличия у кластеризатора поля .hint. Если при наведении на кластер не нужно показывать всплывающую подсказку, рекомендуется установить эту опцию в значение false, чтобы избежать лишних инициализаций. |
| options.margin | 10 | Тип: Number \| Number[] |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Число или массив чисел, задающие отступ для центра кластера относительно ячеек кластеризации. Если задано одно число — оно применяется ко всем сторонам. Если задано два — то это вертикальные и горизонтальные отступы соответственно. Если задан массив из 4х чисел, то это отступы top, right, bottom, left. |
| options.maxZoom | Infinity | Тип: Number[]<br><br>Максимальный коэффициент масштабирования карты, на котором происходит кластеризация объектов. Даже если кластеризация отключена, будут показаны только объекты в видимой области карты. |
| options.minClusterSize | 2 | Тип: Number<br><br>Минимальное количество объектов, образующих кластер. |
| options.preset | — | Тип: String<br><br>Ключ предустановленных опций кластеризатора. Список ключей, доступных в пакете package.clusters, содержится в описании **option.presetStorage**. |
| options.showInAlphabeticalOrder | false | Тип: Boolean<br><br>Показывать метки в балуне в алфавитном порядке при нажатии на кластер. Геообъекты кластера сортируются по специальным полям в данных этих геообъектов — clusterCaption (или balloonContentHeader, если предыдущее поле не определено). По умолчанию геообъекты показываются в порядке добавления в кластеризатор. |
| options.useMapMargin | true | Тип: Boolean<br><br>Нужно ли учитывать отступы карты **map.margin.Manager** при приближении карты после клика на кластере. |
| options.viewportMargin | 128 | Тип: Number \| Number[]<br><br>Отступ для области, в которой производится кластеризация. Кластеризация проводится для видимой области карты. С помощью данной опции область кластеризации расширяется по отношению к видимой области карты. |
| options.zoomMargin | 0 | Тип: Number \| Number[] |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Отступы от границ видимой области карты, которые соблюдаются при приближении карты после клика на кластере. Рекомендуется устанавливать значение опции в соответствии с размером иконок кластеров и меток. Например, если метка попадает в видимую область карты только нижним концом ножки, стоит выставить ненулевой отступ top, чтобы метка оставалась полностью видна после того, как кластер распался. Если задано одно число — оно применяется ко всем сторонам. Если задано два — то это горизонтальные и вертикальные отступы соответственно. Если задан массив из 4х чисел, то это отступы top, right, bottom, left. |

**Пример:**

```
// Создание кластеризатора.
// Создадим карту, на которой необходимо кластеризовать геообъекты.
var map = new ymaps.Map('mapsID', { center: [56.034, 36.992], zoom: 8 }),
// Создадим массив геообъектов.
myGeoObjects = [];
myGeoObjects[0] = new ymaps.GeoObject({
    geometry: { type: "Point", coordinates: [56.034, 36.992] },
    properties: {
        clusterCaption: 'Геообъект №1',
        balloonContentBody: 'Содержимое балуна геообъекта №1.'
    }
});
myGeoObjects[1] = new ymaps.GeoObject({
    geometry: { type: "Point", coordinates: [56.021, 36.983] },
    properties: {
        clusterCaption: 'Геообъект №2',
        balloonContentBody: 'Содержимое балуна геообъекта №2.'
    }
});

// Создадим кластеризатор и запретим приближать карту при клике на кластеры.
var clusterer = new ymaps.Clusterer({ clusterDisableClickZoom: true });
clusterer.add(myGeoObjects);
map.geoObjects.add(clusterer);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **balloon** | **clusterer.Balloon** | Балун кластеризатора. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **hint** | clusterer.hint | Хинт кластеризатора. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **balloonclose** | Закрытие балуна.<br>• target — ссылка на менеджер балуна кластеризатора;<br>• cluster — ссылка на объект-кластер.<br>Экземпляр класса **Event**. |
| **balloonopen** | Открытие балуна на кластере. Имена полей, доступных через метод Event.get:<br>• target — ссылка на менеджер балуна кластеризатора;<br>• cluster — ссылка на объект-кластер.<br>Экземпляр класса **Event**. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**. |
| **hintopen** | Открытие хинта на кластере. Экземпляр класса **Event**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(objects) | **Clusterer** | Добавляет геообъект или массив геообъектов в кластеризатор. |
| **createCluster**(center, geoObjects) | **IGeoObject** | Функция для создания кластера кластеризатором. Вызывается кластеризатором непосредственно в процессе кластеризации. Принимает на вход центр кластера и массив геообъектов, попадающих в данный кластер. Возвращает кластер, который потом будет добавлен на карту. Если требуется, чтобы кластеризатор создавал пользовательские объекты-кластеры, следует переопределить этот метод у кластеризатора. |
| **getBounds**() | Number[][] \| null | Возвращает географические координаты прямоугольной области, охватывающей все элементы кластеризатора. |

| Имя | Возвращает | Описание |
|---|---|---|
| **getClusters**() | **IGeoObject**[] | Метод для получения текущего массива объектов-кластеров. Обратите внимание, что объекты-кластеры меняются при изменении масштаба карты или при сдвиге центра карты. Если необходимо выполнять операции над всеми кластерами, корректнее переопределить метод createCluster и добавлять необходимые операции в его вызов. |
| **getGeoObjects**() | **IGeoObject**[] | Возвращает массив геообъектов, добавленных в кластеризатор. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getObjectState**(geoObject) | Object | Функция для получения информации о текущем состоянии объекта, добавленного в кластеризатор. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **remove**(objects) | **Clusterer** | Удаляет геообъекты из кластеризатора. |
| **removeAll**() | **Clusterer** | Удаляет все геообъекты из кластеризатора. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание полей

### balloon

```
{clusterer.Balloon} balloon
```

Балун кластеризатора.

### hint

```
{clusterer.hint} hint
```

Хинт кластеризатора.

## Описание событий

### balloonclose

Закрытие балуна.

- target — ссылка на менеджер балуна кластеризатора;
- cluster — ссылка на объект-кластер.

Экземпляр класса **Event**.

### balloonopen

Открытие балуна на кластере. Имена полей, доступных через метод Event.get:

- target — ссылка на менеджер балуна кластеризатора;
- cluster — ссылка на объект-кластер.

Экземпляр класса **Event**.

**Пример:**

```
clusterer.events.add('balloonopen', function (e) {
    var clusterPlacemark = e.get('cluster');
});
```

### hintclose

Закрытие хинта. Экземпляр класса **Event**.

### hintopen

Открытие хинта на кластере. Экземпляр класса **Event**.

## Описание методов

### add

```
{Clusterer} add(objects)
```

Добавляет геообъект или массив геообъектов в кластеризатор.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objects* | — | Тип: **IGeoObject** \| **IGeoObject**[]<br><br>Массив геообъектов или единичный геообъект. |

* Обязательный параметр/опция.

**Пример:**

```
var myPlacemark = new ymaps.Placemark([35, 21]);
clusterer.add(myPlacemark);
var placemarks = [
    new ymaps.Placemark([44, 55]),
    new ymaps.Placemark([34, 45])
];
clusterer.add(placemarks);
```

### createCluster

```
{IGeoObject} createCluster(center, geoObjects)
```

Функция для создания кластера кластеризатором. Вызывается кластеризатором непосредственно в процессе кластеризации. Принимает на вход центр кластера и массив геообъектов, попадающих в данный кластер. Возвращает кластер, который потом будет добавлен на карту. Если требуется, чтобы кластеризатор создавал пользовательские объекты-кластеры, следует переопределить этот метод у кластеризатора.

**Возвращает** объект-кластер. По умолчанию создает экземпляры класса **ClusterPlacemark**.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| center* | — | Тип: Number[]<br><br>Центр кластера в геокоординатах. |
| geoObjects* | — | Тип: **IGeoObject**[]<br><br>Массив плейсмарков в кластере. |

* Обязательный параметр/опция.

**Пример:**

```
// Зададим хинт кластерам в зависимости от состава.
var clusterer = new ymaps.Clusterer();
clusterer.createCluster = function (center, geoObjects) {
    // Создаем метку-кластер с помощью стандартной реализации метода.
    var clusterPlacemark = ymaps.Clusterer.prototype.createCluster.call(this,
center, geoObjects),
        geoObjectsLength = clusterPlacemark.getGeoObjects().length,
        hintContent;
    if (geoObjectsLength < 10) {
        hintContent = 'Мало меток';
    } else if (geoObjectsLength < 100) {
        hintContent = 'Нормально так меток';
    } else {
        hintContent = 'Меток навалом';
    }
    clusterPlacemark.properties.set('hintContent', hintContent);
    return clusterPlacemark;
};
```

## getBounds

```
{Number[][] | null} getBounds()
```

**Возвращает** географические координаты прямоугольной области, охватывающей все элементы кластеризатора.

**Пример:**

```
var clusterer = new ymaps.Clusterer();
clusterer.add(myPlacemarks);
map.setBounds(clusterer.getBounds());
map.geoObjects.add(clusterer);
```

## getClusters

```
{IGeoObject[]} getClusters()
```

Метод для получения текущего массива объектов-кластеров. Обратите внимание, что объекты-кластеры меняются при изменении масштаба карты или при сдвиге центра карты. Если необходимо выполнять операции над всеми кластерами, корректнее переопределить метод createCluster и добавлять необходимые операции в его вызов.

**Возвращает** массив кластеров, добавленных на карту в данный момент.

**Пример:**

```
map.events.add('boundschange', function () {
    var clusters = clusterer.getClusters();
    alert('Карта подвинулась и сейчас показано ' + clusters.length + '
кластеров.');
});
```

## getGeoObjects

```
{IGeoObject[]} getGeoObjects()
```

**Возвращает** массив геообъектов, добавленных в кластеризатор.

**Пример:**

```
// Посчитаем количество объектов, попавших в видимую область карты.
var geoObjects = clusterer.getGeoObjects(),
    shownObjectsCounter = 0;
for (var i = 0, l = geoObjects.length; i < l; i++) {
    if (clusterer.getObjectState(geoObjects[i]).isShown) {
        shownObjectsCounter++;
    }
}
alert('Сейчас на карте показаны ' + shownObjectsCounter + ' меток из ' +
geoObjects.length + '.');
```

## getObjectState

```
{Object} getObjectState(geoObject)
```

Функция для получения информации о текущем состоянии объекта, добавленного в кластеризатор.

**Возвращает** объект с полями:

- isShown — признак, указывающий, находится ли объект в видимой области карты;
- cluster — ссылка на кластер, в который добавлен объект;
- isClustered — признак, указывающий, попал ли объект в состав кластера.

.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geoObject* | — | Тип: **IGeoObject**<br><br>Геообъект, для которого нужно получить состояние. |

* Обязательный параметр/опция.

**Пример:**

```
// Открытие балуна кластера с выбранным объектом.
// Получим данные о состоянии объекта внутри кластера.
var geoObjectState = clusterer.getObjectState(myGeoObjects[1]);
// Проверяем, находится ли объект в видимой области карты.
if (geoObjectState.isShown) {

    // Если объект попадает в кластер, открываем балун кластера с нужным выбранным
объектом.
    if (geoObjectState.isClustered) {
        geoObjectState.cluster.state.set('activeObject', myGeoObjects[1]);
        clusterer.balloon.open(geoObjectState.cluster);

    } else {
        // Если объект не попал в кластер, открываем его собственный балун.
        myGeoObjects[1].balloon.open();
    }
}
```

## remove

```
{Clusterer} remove(objects)
```

Удаляет геообъекты из кластеризатора.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objects* | — | Тип: **IGeoObject** \| **IGeoObject**[]<br><br>Массив геообъектов. |

* Обязательный параметр/опция.

**Пример:**

```
var myPlacemark = new ymaps.Placemark([35, 21]),
    placemarks = [
        new ymaps.Placemark([44, 55]),
        new ymaps.Placemark([34, 45]),
        ...
    ];
clusterer.add(myPlacemark).add(placemarks);
...
// Удалим первые 10 объектов массива из кластеризатора.
clusterer.remove(placemarks.slice(0, 10));
```

## removeAll

```
{Clusterer} removeAll()
```

Удаляет все геообъекты из кластеризатора.

**Возвращает** ссылку на себя.

**Пример:**

```
clusterer.add(placemark1);
clusterer.add(placemark2);
clusterer.removeAll();
```

# ClusterPlacemark

Расширяет **IGeoObject**, **collection.Item**.

Кластер геообъектов. Используется по умолчанию в **Clusterer**.

Конструктор | Поля | События | Методы

## Конструктор

```
ClusterPlacemark(geometry, properties[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: Number[] \| Object \| **IPointGeometry**<br><br>Координаты метки или хэш с описанием геометрии или ссылка на объект точечной геометрии. |
| properties * | — | Тип: **IDataManager**<br><br>Данные кластера. |
| properties.geoObjects * | — | Тип: **IGeoObject**[]<br><br>Массив геообъектов, которые находятся в данном кластере. |
| options | — | Тип: Object<br><br>Опции кластера. Помимо частных опций, балун кластера поддерживает те же опции, что и **Balloon**. Опции для балуна кластера указываются с префиксом 'balloon'. |
| options.balloonContentLayout | 'cluster#balloonTwoColumns' | Тип: Function \| String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Макет балуна кластера в обычном режиме. Можно передать конструктор объекта с интерфейсом **ILayout** или ключ одного из стандартных макетов. Каждый стандартный макет имеет некоторые собственные опции. В стандартных макетах по умолчанию используются поля данных геообъектов: clusterCaption, balloonContentHeader, balloonContent и balloonContentFooter (см. **Placemark**).<br>• 'cluster#balloonTwoColumns' — Макет с двумя колонками. В левой колонке находится список названий меток (поле данных метки clusterCaption или balloonContentHeader, если предыдущее поле не определено). В правой колонке располагается вся информация о геообъекте. **Опции макета**<br>  • balloonLeftColumnWidth Ширина поля со списком объектов в балуне кластера в пикселях. Значение по умолчанию: 125<br>• 'cluster#balloonCarousel' — Информация о геообъекте располагается в центре. По бокам размещаются кнопки перехода к предыдущему и следующему геообъекту. В нижней части балуна находится меню быстрой навигации. **Опции макета**<br>  • balloonCycling Цикличность списка при навигации боковыми стрелками. Значение по умолчанию: true<br>  • balloonPagerSize Количество элементов навигации в нижней панели. Значение по умолчанию зависит от типа устройства. Для мобильных телефонов — это 4, а для планшетов и персональных компьютеров — 9.<br>  • balloonPagerType Тип нижней панели навигации. Может принимать значения 'numeric' и 'marker'.<br>  • numeric — отображение номера геообъекта в списке;<br>  • marker — отображение маркеров без номеров; Рекомендуется использовать при количестве элементов в кластере |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | меньшем или равном значению options.balloonPagerSize.<br><br>Значение по умолчанию: numeric<br>• balloonPagerVisible Отображать ли панель навигации. Значение по умолчанию: true<br>• 'cluster#balloonAccordion' — Информация о геообъектах отображается в виде списка. Каждый элемент списка — это маленькая иконка и название метки (поле данных метки clusterCaption или balloonContentHeader, если предыдущее поле не определено). Цвет иконок соответствует цвету метки. Цвет метки определяется опцией "iconColor" **Placemark**. В браузере Internet Explorer ниже 9й версии иконки не отображаются всегда. После клика по элементу списка под ним разворачивается информация о геообъекте. **Опции макета**<br>• balloonAccordionShowIcons Опция, которая определяет отображается ли иконка геообъекта в макете 'cluster#balloonAccordionItemTitle'. В браузере Internet Explorer ниже 9й версии опция не используется. Значение по умолчанию: true |
| options.balloonContentLayoutHeight | — | Тип: Number<br><br>Высота макета контента балуна кластера. По умолчанию опция балуна balloonMaxHeight для кластера не установлена, так как все стандартные макеты балунов кластеров имеют определенные размеры. Стандартное значение зависит от макета.<br>• 'cluster#balloonTwoColumns' — 210 пикселей<br>• 'cluster#balloonCarousel' — 177 пикселей<br>• 'cluster#balloonAccordion' — 283 пикселей |
| options.balloonContentLayoutWidth | — | Тип: Number |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Ширина макета контента балуна кластера. По умолчанию опция балуна balloonMaxWidth для кластера не установлена, так как все стандартные макеты балунов кластеров имеют определенные размеры. Стандартное значение зависит от макета.<br>• 'cluster#balloonTwoColumns' — 475 пикселей<br>• 'cluster#balloonCarousel' — 308 пикселей<br>• 'cluster#balloonAccordion' — 305 пикселей<br>Опция не используется в режиме панели. |
| options.balloonItemContentLayout | — | Тип: **ILayout** \| String<br>Макет с информацией о геообъекте. Стандартное значение зависит от макета.<br>• В макете 'cluster#balloonTwoColumns' отображается справа от списка. Стандартно значение 'cluster#balloonTwoColumnsItemContent'.<br>• В макете 'cluster#balloonCarousel' отображается в центре. Стандартно значение 'cluster#balloonCarouselItemContent'.<br>• В макете 'cluster#balloonAccordion' отображает после клика по элементу списка. Стандартно значение 'cluster#balloonAccordionItemContent'.<br>**Набор полей, которые поступают в данный макет, отличаются от родительского и соответствует полям, которые поступают в обычный макет балуна геообъекта. Еще были добавлены поля ownerProperties, ownerOptions и ownerState для доступа к данным кластера.** |
| options.balloonPanelContentLayout | null | Тип: Function \| String<br>Макет балуна кластера в режиме "панель". Можно передать конструктор объекта с интерфейсом **ILayout**. Доступные значения такие же, как и у опции 'balloonContentLayout'. Если значение равно null, то применяется значение опции 'balloonContentLayout'. |
| options.cursor | 'pointer' | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Курсор над меткой кластера. |
| options.disableClickZoom | false | Тип: Boolean<br><br>Флаг, запрещающий увеличение коэффициента масштабирования карты при клике на кластер. |
| options.hideIconOnBalloonOpen | true | Тип: Boolean<br><br>Скрывать иконку при открытии балуна. |
| options.iconColor | — | Тип: String<br><br>Цвет иконки кластера. Эта опция применяется для стандартных иконок в браузерах, поддерживающих SVG. |
| options.iconContentLayout | 'cluster#iconContent' | Тип: Function \| String<br><br>Макет содержимого метки кластера. (Тип: конструктор объекта с интерфейсом **ILayout** или ключ макета). Если у метки не нужно отображать содержимое, значение опции нужно выставить в null. |
| options.iconLayout | 'cluster#icon' | Тип: Function \| String<br><br>Макет метки кластера (Тип: конструктор объекта с интерфейсом **ILayout** или ключ макета). |
| options.icons | — | Тип: Object[]<br><br>Массив, описывающий иконки для стандартной реализации кластера. Описание иконки представляет из себя объект с полями<br>• href — ссылка на картинку;<br>• size — массив из двух чисел — размер иконки в пикселях;<br>• offset — смещение иконки относительно точки привязки объекта;<br>• shape — необязательное поле. Объект, реализующий интерфейс **IShape** или JSON-описание геометрии. Позволяет задавать описание геометрии иконки. Если параметр отсутствует, то активной для событий (наведение мыши, клик) будет считаться прямоугольная область вокруг иконки. |
| options.iconShape | — | Тип: **IGeometryJson** \| null |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Фигура активной области метки кластера. Задается в виде JSON-описания пиксельной геометрии иконки. Эту опцию нужно использовать при создании своих HTML макетов. Координаты геометрии фигуры отсчитываются от точки привязки. |
| options.interactivityModel | 'default#geoObject' | Тип: String<br><br>Модель интерактивности кластера. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.numbers | [10, 100] | Тип: Number[]<br><br>Массив, описывающий граничные значения для размеров кластеров. Количество иконок, описанных в опции "icons", должно быть на 1 больше, чем чисел в данном массиве. |
| options.openBalloonOnClick | true | Тип: Boolean<br><br>Опция, позволяющая запретить открытие балуна при клике на кластере. По умолчанию открытие балуна разрешено. |
| options.openEmptyHint | false | Тип: Boolean<br><br>Опция, включающая показ пустых всплывающих подсказок. По умолчанию пустые хинты кластера не показываются. |
| options.openHintOnHover | true | Тип: Boolean<br><br>Опция, позволяющая запретить показ всплывающей подсказки при наведении на кластер. По умолчанию показ хинтов разрешен. |
| options.zIndexHover | — | Тип: Number<br><br>Значение zIndex, которое выставляется метке кластера при наведении. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Переменная с описанием двух видов иконок кластеров.
var clusterIcons = [
    {
        href: 'small.png',
        size: [40, 40],
        // Отступ, чтобы центр картинки совпадал с центром кластера.
        offset: [-20, -20]
    },
    {
        href: 'big.png',
        size: [60, 60],
        offset: [-30, -30],
```

```
         // Можно задать геометрию активной области метки.
         // Если геометрия не задана, активной областью будет
         // прямоугольник.
         shape: {
            type: 'Circle',
            coordinates: [0, 0],
            radius: 30
         }
     }],
     // При размере кластера до 100 будет использована картинка 'small.jpg'.
     // При размере кластера больше 100 будет использована 'big.png'.
     clusterNumbers = [100],
     // Сделаем макет содержимого иконки кластера,
     // в котором цифры будут раскрашены в белый цвет.
     MyIconContentLayout = ymaps.templateLayoutFactory.createClass(
     '<div style="color: #FFFFFF; font-weight:
bold;">{{ properties.geoObjects.length }}</div>'),
     var clusterer = new ymaps.Clusterer({
        // Если опции для кластеров задаются через кластеризатор,
        // необходимо указывать их с префиксами "cluster".
        clusterIcons: clusterIcons,
        clusterNumbers: clusterNumbers,
        clusterIconContentLayout: MyIconContentLayout
    });
```

**2.**

```
// Создание кластеризатора с макетом-каруселью
     var clusterer = new ymaps.Clusterer({
        clusterDisableClickZoom: true,
        // Используем макет "карусель"
        clusterBalloonContentLayout: "cluster#balloonCarousel",
        // Запрещаем зацикливание списка при постраничной навигации
        clusterBalloonCycling: false,
        // Настройка внешнего вида панели навигации. Элементами панели навигации
будут маркеры.
        clusterBalloonPagerType: "marker",
        // Количество элементов в панели навигации
        clusterBalloonPagerSize: 6
    });
```

**3.**

```
// Создание кластеризатора с макетом-аккордеоном
var clusterer = new ymaps.Clusterer({
        clusterDisableClickZoom: true,
        // Используем макет "аккордеон"
        clusterBalloonContentLayout: "cluster#balloonAccordion"
    });
```

**4.**

```
// Создание кластеризатора с произвольным HTML-макетом иконки кластера.
var clusterer = new ymaps.Clusterer({
        // Зададим произвольный макет метки кластера.
        clusterIconLayout: ymaps.templateLayoutFactory.createClass('<div
class="clusterIcon">{{ properties.geoObjects.length }}</div>'),
        // Чтобы метка стала кликабельной, нужно переопределить ее активную
область.
        clusterIconShape: {
            type: 'Rectangle',
            coordinates: [[0, 0], [20, 20]]
        }
    });
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |
| geometry | IGeometry \| null | Геометрия геообъекта.<br>Унаследовано от IGeoObject. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |
| properties | IDataManager | Данные геообъекта.<br>Унаследовано от IGeoObject. |
| state | data.Manager | Состояние кластера. Определяется следующими полями:<br>• activeObject — Ссылка на активный объект кластера. Активным объектом является тот, который в данный момент выбран в балуне кластера. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| geometrychange | Изменение геометрии геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие геометрии.<br>Унаследовано от IGeoObject. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

| Имя | Описание |
| --- | --- |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getBounds() | Number[][] \| null | Возвращает географические координаты прямоугольной области, охватывающей все элементы кластера. |
| getGeoObjects() | IGeoObject[] | Метод является упрощенным вызовом cluster.properties.get('geoObjects'); |
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |

| Имя | Возвращает | Описание |
|---|---|---|
| **onAddToMap**(map) | | Функция, которая вызывается при добавлении элемента на карту. Для выполнения дополнительных действий при добавлении объекта на карту переопределите эту функцию.<br><br>Унаследован от **collection.Item**. |
| **onRemoveFromMap**(oldMap) | | Функция, которая вызывается при удалении элемента с карты. Для выполнения дополнительных действий при удалении объекта с карты переопределите эту функцию.<br><br>Унаследован от **collection.Item**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br><br>Унаследован от **IChildOnMap**. |

## Описание полей

### state

```
{data.Manager} state
```

Состояние кластера. Определяется следующими полями:

- activeObject — Ссылка на активный объект кластера. Активным объектом является тот, который в данный момент выбран в балуне кластера.

**Пример:**

```
var geoObjects = cluster.properties.get('geoObjects');
// При открытии балуна кластера будет выбран третий объект в списке.
cluster.state.set('activeObject', geoObjects[2]);
```

## Описание методов

### getBounds

```
{Number[][] | null} getBounds()
```

**Возвращает** географические координаты прямоугольной области, охватывающей все элементы кластера.

### getGeoObjects

```
{IGeoObject[]} getGeoObjects()
```

Метод является упрощенным вызовом cluster.properties.get('geoObjects');

**Возвращает** массив геообъектов, образующих кластер.

# collection

## collection.Item

Расширяет **IChildOnMap**, **ICustomizable**, **IEventEmitter**, **IParentOnMap**.

Базовый класс элемента коллекции объектов карты.

Конструктор | Поля | События | Методы

### Конструктор

```
collection.Item([options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции обьекта. |

**Пример:**

```
// Пример реализации собственного элемента управления на основе наследования от
collection.Item.
// Элемент управления отображает название объекта, который находится в центре
карты.
var map = new ymaps.Map('map', {
        center: [55.819543, 37.611619],
        zoom: 6
    }),
// Создаем собственный класс.
    CustomControl = function (options) {
        CustomControl.superclass.constructor.call(this, options);
    };
// И наследуем его от collection.Item.
ymaps.util.defineClass(CustomControl, ymaps.collection.Item, {
    onAddToMap: function (map) {
        CustomControl.superclass.onAddToMap.call(this, map);
        // Создаем HTML-элемент с текстом.
        this.getParent().getChildElement(this).then(this._onChildElementGet,
this);
    },

    onRemoveFromMap: function (oldMap) {
        CustomControl.superclass.onRemoveFromMap.call(this, oldMap);
    },

    _onChildElementGet: function(parentElementContainer) {
        // Здесь можно создать dom-отображение для элемента управления
        // и добавить его дочерним элементом в parentElementContaner.
        // ...
    }
});

var customControl = new CustomControl();
map.controls.add(customControl, {top: 10, left: 10});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает карту, которой принадлежит элемент коллекции. |
| getParent() | IParentOnMap | Возвращает родительский объект. |
| onAddToMap(map) | | Функция, которая вызывается при добавлении элемента на карту. Для выполнения дополнительных действий при добавлении объекта на карту переопределите эту функцию. |
| onRemoveFromMap(oldMap) | | Функция, которая вызывается при удалении элемента с карты. Для выполнения дополнительных действий при удалении объекта с карты переопределите эту функцию. |
| setParent(parent) | collection.Item | Устанавливает родителя для заданного элемента коллекции. |

## Описание методов

### getMap

```
{Map} getMap()
```

**Возвращает** карту, которой принадлежит элемент коллекции.

### getParent

```
{IParentOnMap} getParent()
```

**Возвращает** родительский объект.

### onAddToMap

```
onAddToMap(map)
```

Функция, которая вызывается при добавлении элемента на карту. Для выполнения дополнительных действий при добавлении объекта на карту переопределите эту функцию.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map**<br><br>Карта, на которую добавлен объект. |

* Обязательный параметр/опция.

### onRemoveFromMap

```
onRemoveFromMap(oldMap)
```

Функция, которая вызывается при удалении элемента с карты. Для выполнения дополнительных действий при удалении объекта с карты переопределите эту функцию.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| oldMap* | — | Тип: **Map**<br><br>Карта, с которой удален объект. |

* Обязательный параметр/опция.

### setParent

```
{collection.Item} setParent(parent)
```

Устанавливает родителя для заданного элемента коллекции.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parent* | — | Тип: **IParentOnMap**<br><br>Родительский объект. |

* Обязательный параметр/опция.

# Collection

Расширяет **ICollection**, **collection.Item**.

Базовая реализация коллекции объектов на карте.

Конструктор | Поля | События | Методы

## Конструктор

```
Collection([options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции коллекции. |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **add** | Произошло добавление дочернего объекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• child — добавленный дочерний элемент. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **remove** | Произошло удаление дочернего объекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• child — удаленный дочерний элемент. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(child) | **Collection** | Добавляет элемент в коллекцию. |
| **each**(callback, context) | **Collection** | Вызывает функцию-обработчик для всех элементов коллекции. |
| **filter**(filterFunction) | Object[] | Вызывает функцию-фильтр для всех элементов коллекции. При возвращении фильтром ненулевого значения элемент коллекции попадает в конечный массив. |
| **get**(index) | Object | Возвращает элемент коллекции или null, если номер находится вне диапозона номеров коллекции. |
| **getAll**() | Object[] | Возвращает массив со всеми элементами коллекции. |
| **getIterator**() | **IIterator** | Возвращает итератор для перебора элементов коллекции. |
| **getLength**() | Number | Возвращает количество элементов коллекции. |
| **getMap**() | **Map** | Возвращает карту, которой принадлежит элемент коллекции.<br>Унаследован от **collection.Item**. |
| **getParent**() | **IParentOnMap** | Возвращает родительский объект.<br>Унаследован от **collection.Item**. |
| **indexOf**(childToFind) | Number | Возвращает порядковый номер объекта в коллекции либо -1, если объект не найден. |
| **onAddToMap**(map) | | Функция, которая вызывается при добавлении элемента на карту. Для выполнения дополнительных действий при добавлении объекта на карту переопределите эту функцию.<br>Унаследован от **collection.Item**. |
| **onRemoveFromMap**(oldMap) | | Функция, которая вызывается при удалении элемента с карты. Для выполнения дополнительных действий при удалении объекта с карты переопределите эту функцию.<br>Унаследован от **collection.Item**. |
| **remove**(child) | **Collection** | Удаляет элемент из коллекции. |
| **removeAll**() | **Collection** | Удаляет все элементы из коллекции. |
| **setParent**(parent) | **collection.Item** | Устанавливает родителя для заданного элемента коллекции.<br>Унаследован от **collection.Item**. |

## Описание событий

### add

Произошло добавление дочернего объекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- child — добавленный дочерний элемент.

### remove

Произошло удаление дочернего объекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- child — удаленный дочерний элемент.

## Описание методов

### add

```
{Collection} add(child)
```

Добавляет элемент в коллекцию.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **collection.Item**<br><br>Элемент, который необходимо добавить. |

* Обязательный параметр/опция.

### each

```
{Collection} each(callback, context)
```

Вызывает функцию-обработчик для всех элементов коллекции.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик. На вход получает элемент коллекции. При возвращении функцией значения false перебор останавливается. |
| context* | — | Тип: Object<br><br>Контекст вызываемой функции. |

* Обязательный параметр/опция.

## filter

```
{Object[]} filter(filterFunction)
```

Вызывает функцию-фильтр для всех элементов коллекции. При возвращении фильтром ненулевого значения элемент коллекции попадает в конечный массив.

**Возвращает** массив из элементов, которые попали в выборку.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| filterFunction* | — | Тип: Function<br><br>Функция, которая является фильтром для объектов коллекции. Первым параметром принимает элемент из коллекции. Должна возвращать булевое значение. |

* Обязательный параметр/опция.

## get

```
{Object} get(index)
```

**Возвращает** элемент коллекции или null, если номер находится вне диапозона номеров коллекции.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Number<br><br>Порядковый номер элемента в коллекции. |

* Обязательный параметр/опция.

## getAll

```
{Object[]} getAll()
```

**Возвращает** массив со всеми элементами коллекции.

## getIterator

```
{IIterator} getIterator()
```

**Возвращает** итератор для перебора элементов коллекции.

## getLength

```
{Number} getLength()
```

**Возвращает** количество элементов коллекции.

## indexOf

```
{Number} indexOf(childToFind)
```

**Возвращает** порядковый номер объекта в коллекции либо -1, если объект не найден.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| childToFind* | — | Тип: Object<br><br>Искомый объект. |

* Обязательный параметр/опция.

## remove

```
{Collection} remove(child)
```

Удаляет элемент из коллекции.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **collection.Item**<br><br>Элемент, который необходимо удалить. |

* Обязательный параметр/опция.

## removeAll

```
{Collection} removeAll()
```

Удаляет все элементы из коллекции.

**Возвращает** ссылку на себя.

# control

## control.Button

Расширяет **ICustomizable**, **ISelectableControl**.

Элемент управления "Кнопка". Стандартный макет кнопки изменяет свой внешний вид в зависимости от размеров карты. Если карта имеет большую ширину, то кнопка принимает вид "картинка + текст". Если карта среднего размера, кнопка принимает вид "текст". Если карта небольшого размера, то в макете кнопки отображается только иконка. Если для кнопки не задана иконка, то во всех состояниях будет отображаться текст и наоборот.

Конструктор | Поля | События | Методы

### Конструктор

```
control.Button([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: Object \| String<br><br>Параметры кнопки или строка — содержимое кнопки в виде HTML. |
| parameters.data | — | Тип: Object<br><br>Данные кнопки. |
| parameters.data.content | — | Тип: String<br><br>Содержимое кнопки в виде HTML. |
| parameters.data.image | — | Тип: String<br><br>URL иконки кнопки. Стандартный макет кнопки рассчитан на иконку размером 16x16 пикселей. |
| parameters.data.title | — | Тип: String<br><br>Текст всплывающей подсказки, которая появляется при наведении на кнопку курсора мыши. |
| parameters.options | — | Тип: Object<br><br>Опции кнопки. |
| parameters.options.adjustMapMargin | false | Тип: Boolean<br><br>Регистрирует ли кнопка свои размеры в менеджере отступов карты **map.margin.Manager**. |
| parameters.options.float | "right" | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Сторона, по которой нужно выравнивать элемент управления. Может принимать три значения: "left", "right" или "none". При значении "left" или "right" элементы управления выстраиваются друг за другом, начиная от левого или правого края карты соответственно. При значении "none" элементы управления позиционируется только по значениям опций left, right, bottom, top относительно границ карты. Также смотрите описание опции position. |
| parameters.options.floatIndex | 0 | Тип: Number<br><br>Приоритет расположения элемента управления. Элемент с максимальным приоритетом находится ближе к указанному в свойстве float краю карты. Не работает при float = "none". |
| parameters.options.layout | — | Тип: **ISelectableControlLayout** \| String<br><br>Конструктор макета элемента управления, реализующий интерфейс **ISelectableControlLayout** или ключ макета в хранилище **layout.storage**. В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.Button.options;<br>• data — менеджер данных элемента управления **control.Button.data**;<br>• state — менеджер состояния элемента управления **control.Button.state**.<br><br>Макет меняет свой внешний вид на основе данных, состояния и опций элемента управления. Элемент управления, в свою очередь, реагирует на интерфейсные события макета и меняет значения полей **control.Button.state** в зависимости от полученных команд. |
| parameters.options.maxWidth | 90 | Тип: Number \| Number[] |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Максимальная ширина кнопки в различных состояниях. Если задано число, то считается, что кнопка имеет одинаковые максимальные размеры во всех состояниях. Если задан массив, то он будет трактоваться как максимальная ширина кнопки в различных состояниях — от меньшего к большему. Количество состояний задается в экземпляре класса **control.Manager**, который обычно является полем **Map.controls**, через опцию states. По умолчанию у элементов управления есть три состояния — ['small', 'medium', 'large']. |
| parameters.options.position | — | Тип: Object<br><br>Объект, описывающий позицию элемента управления. При указании опции position значение опции float автоматически трактуется как "none". |
| parameters.options.position.bottom | 'auto' | Тип: Number \| String<br><br>Положение относительно нижнего края карты. |
| parameters.options.position.left | 'auto' | Тип: Number \| String<br><br>Положение относительно левого края карты. |
| parameters.options.position.right | 'auto' | Тип: Number \| String<br><br>Положение относительно правого края карты. |
| parameters.options.position.top | 'auto' | Тип: Number \| String<br><br>Положение относительно верхнего края карты. |
| parameters.options.selectOnClick | true | Тип: Boolean<br><br>Опция, описывающая поведение кнопки.<br><br>• true — кнопка становится "нажатой" после клика — меняется ее внешний вид и значение поля **control.Button.state** 'selected';<br>• false — кнопка не меняет свой внешний вид и состояние после клика на нее. |
| parameters.options.size | 'auto' | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Параметр, отвечающий за внешний вид стандартного макета кнопки. Может принимать значения:<br>• 'auto' — стандартный макет кнопки автоматически изменяет свой внешний вид в зависимости от размеров карты и от количества добавленных элементов управления;<br>• 'small' — в макете кнопки отображается иконка, независимо от размера карты;<br>• 'medium' — в макете кнопки отображается только текст, независимо от размера карты;<br>• 'large' — в макете кнопки всегда отображается иконка и текст, независимо от размера карты. |
| parameters.options.visible | true | Тип: Boolean<br><br>Признак того, что элемент управления отображается. |
| parameters.state | — | Тип: Object<br><br>Объект, описывающий состояние кнопки. |
| parameters.state.enabled | true | Тип: Boolean<br><br>Признак, активна ли кнопка. |
| parameters.state.selected | false | Тип: Boolean<br><br>Признак, нажата ли кнопка. |

**Примеры:**

**1.**

```
// Пример 1.
// Создание кнопки и добавление ее на карту.
var button = new ymaps.control.Button({
         data: {
             // Зададим иконку для кнопки
             image: 'images/button.jpg',
             // Текст на кнопке.
             content: 'Сохранить',
             // Текст всплывающей подсказки.
             title: 'Нажмите для сохранения маршрута'
         },
         options: {
             // Зададим опции для кнопки.
             selectOnClick: false,
             // Кнопка будет иметь три состояния - иконка, текст и иконка+текст.
             // Поэтому зададим три значения ширины кнопки для всех состояний.
             maxWidth: [30, 100, 150]
         }});
map.controls.add(button, { float: 'right', floatIndex: 100 });
```

**2.**

```
// Пример 2.
// Создание кнопки с пользовательским макетом
var button = new ymaps.control.Button({
        data: {
            content: 'Красная кнопка',
            title: 'Нажмите на кнопку'
        },
        options: {
            layout: ymaps.templateLayoutFactory.createClass(
                // Если кнопка не нажата, к ней применится css-стиль 'myButton'
                // Если кнопка нажата, к ней применятся css-стили 'myButton' и
'myButtonSelected'.
                "<div class='myButton {% if state.selected %}myButtonSelected{%
endif %}' title='{{ data.title }}'>" +
                "{{ data.content }}" +
                "</div>"
            ),
            // Чтобы другие элементы управления корректно позиционировались по
горизонтали,
            // нужно обязательно задать максимальную ширину для макета.
            maxWidth: 150
        }});
map.controls.add(button, { float: 'left', floatIndex: 0 });

// Можно задать позиционирование относительно краев карты. В этом случае
// значение опции maxWidth не влияет на позиционирование
// элементов управления.
map.controls.add(button, { float: 'none', position: {left: '5px', top: '5px'} });
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **data** | **data.Manager** | Данные кнопки. Имена полей, доступных через метод data.Manager.get:<br>• image — иконка кнопки, если есть;<br>• content — содержимое кнопки в виде HTML;<br>• title — текст всплывающей подсказки, которая появляется при наведении на кнопку курсора мыши. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **IControl**. |
| **state** | **data.Manager** | Состояние кнопки. Имена полей, доступных через метод data.Manager.get:<br>• selected — признак, нажата ли кнопка;<br>• enabled — признак, активна ли кнопка;<br>• size — размер, который в данный момент выставлен кнопке; |

## События

| Имя | Описание |
|---|---|
| **click** | Нажатие на кнопку. Экземпляр класса **Event**. |
| **deselect** | Элемент управления не выбран.<br>Унаследовано от **ISelectableControl**. |

| Имя | Описание |
|---|---|
| **disable** | Элемент управления стал недоступным.<br>Унаследовано от **ISelectableControl**. |
| **enable** | Элемент управления стал доступным.<br>Унаследовано от **ISelectableControl**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **press** | Событие, сигнализирующее о том, что кнопка была нажата. В отличии от click генерируется только если состояние isEnabled == true. Экземпляр класса **Event**. |
| **select** | Элемент управления выбран.<br>Унаследовано от **ISelectableControl**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **deselect**() | | Отменяет выбор элемента управления (выключает).<br>Унаследован от **ISelectableControl**. |
| **disable**() | | Делает элемент управления недоступным (запрещены пользовательские действия).<br>Унаследован от **ISelectableControl**. |
| **enable**() | | Делает элемент управления доступным (разрешены пользовательские действия).<br>Унаследован от **ISelectableControl**. |
| **getMap**() | **Map** | Возвращает ссылку на карту. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IControl**. |
| **isEnabled**() | Boolean | Возвращает true — элемент управления доступен, false — недоступен.<br>Унаследован от **ISelectableControl**. |
| **isSelected**() | Boolean | Возвращает true — элемент управления выбран, false — не выбран.<br>Унаследован от **ISelectableControl**. |
| **select**() | | Выбирает (включает) элемент управления. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **ISelectableControl**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IControl**. |

## Описание полей

### data

```
{data.Manager} data
```

Данные кнопки. Имена полей, доступных через метод data.Manager.get:

- image — иконка кнопки, если есть;
- content — содержимое кнопки в виде HTML;
- title — текст всплывающей подсказки, которая появляется при наведении на кнопку курсора мыши.

### state

```
{data.Manager} state
```

Состояние кнопки. Имена полей, доступных через метод data.Manager.get:

- selected — признак, нажата ли кнопка;
- enabled — признак, активна ли кнопка;
- size — размер, который в данный момент выставлен кнопке;

**Пример:**

```
var button = new ymaps.control.Button('Редактировать');
// Зададим состояние кнопки "нажата" -
// аналогично вызову метода button.select();
button.state.set('selected', true);
```

## Описание событий

### click

Нажатие на кнопку. Экземпляр класса **Event**.

### press

Событие, сигнализирующее о том, что кнопка была нажата. В отличии от click генерируется только если состояние isEnabled == true. Экземпляр класса **Event**.

## Описание методов

### getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

# control.FullscreenControl

Расширяет **control.Button**.

Элемент управления "Полноэкранный режим". Задать свойство z-index контейнеру карты, находящейся в "полноэкранном режиме", можно с помощью опции Map.options.fullscreenZIndex. Ключ элемента управления в хранилище **control.storage** — "fullscreenControl".

Конструктор | Поля | События | Методы

## Конструктор

```
control.FullscreenControl([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: Object<br><br>Параметры элемента управления. |
| parameters.data | — | Тип: Object<br><br>Объект, описывающий данные элемента управления. |
| parameters.data.title | — | Тип: String<br><br>Текст всплывающей подсказки, которая появляется при наведении на кнопку курсора мыши. |
| parameters.options | — | Тип: Object<br><br>Опции элемента управления. |
| parameters.options.adjustMapMargin | false | Тип: Boolean<br><br>Регистрирует ли элемент управления свои размеры в менеджере отступов карты **map.margin.Manager**. |
| parameters.options.float | "right" | Тип: String<br><br>Сторона, по которой нужно выравнивать элемент управления. Может принимать три значения: "left", "right" или "none". При значении "left" или "right" элементы управления выстраиваются друг за другом, начиная от левого или правого края карты соответственно. При значении "none" элементы управления позиционируется только по значениям опций left, right, bottom, top относительно границ карты. |
| parameters.options.floatIndex | 300 | Тип: Number |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Приоритет расположения элемента управления. Элемент с максимальным приоритетом находится ближе к указанному в свойстве float краю карты. Не работает при float = "none". Также смотрите описание опции position. |
| parameters.options.layout | — | Тип: **ISelectableControlLayout** \| String<br><br>Конструктор макета элемента управления, реализующий интерфейс **ISelectableControlLayout** или ключ макета в хранилище **layout.storage**. В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.FullscreenControl.options;<br>• data — менеджер данных элемента управления control.FullscreenControl.data;<br>• state — менеджер состояния элемента управления **control.FullscreenControl.state**.<br><br>Макет меняет свой внешний вид на основе данных, состояния и опций элемента управления. Элемент управления, в свою очередь, реагирует на интерфейсные события макета. |
| parameters.options.maxWidth | 28 | Тип: Number \| Number[]<br><br>Максимальная ширина элемента управления в различных состояниях. Если задано число, то считается, что элемент управления имеет одинаковые максимальные размеры во всех состояниях. Если задан массив, то он будет трактоваться как максимальная ширина в различных состояниях — от меньшего к большему. Количество состояний задается в экземпляре класса **control.Manager**, который обычно является полем **Map.controls**, через опцию states. По умолчанию у элементов управления есть три состояния — ['small', 'medium', 'large']. По умолчанию элемент управления не изменяет своего размера, всегда имея вид кнопки с иконкой. |
| parameters.options.position | — | Тип: Object<br><br>Объект, описывающий позицию элемента управления. При указании опции position значение опции float автоматически трактуется как "none". |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters.options.position.bottom | 'auto' | Тип: Number \| String<br><br>Положение относительно нижнего края карты. |
| parameters.options.position.left | 'auto' | Тип: Number \| String<br><br>Положение относительно левого края карты. |
| parameters.options.position.right | 'auto' | Тип: Number \| String<br><br>Положение относительно правого края карты. |
| parameters.options.position.top | 'auto' | Тип: Number \| String<br><br>Положение относительно верхнего края карты. |
| parameters.options.visible | true | Тип: Boolean<br><br>Признак того, что элемент управления отображается. |
| parameters.state | — | Тип: Object<br><br>Объект, описывающий состояние элемента управления. |
| parameters.state.enabled | true | Тип: Boolean<br><br>Признак, активна ли кнопка. |
| parameters.state.selected | false | Тип: Boolean<br><br>Признак, нажата ли кнопка. |

**Пример:**

```
// Добавим элемент управления на карту и сразу переведем
// её в «полноэкранный режим».
var fullscreenControl = new ymaps.control.FullscreenControl();
myMap.controls.add(fullscreenControl);
fullscreenControl.enterFullscreen();
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **data** | **data.Manager** | Данные кнопки. Имена полей, доступных через метод data.Manager.get:<br>• image — иконка кнопки, если есть;<br>• content — содержимое кнопки в виде HTML;<br>• title — текст всплывающей подсказки, которая появляется при наведении на кнопку курсора мыши.<br><br>Унаследовано от **control.Button**. |
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

| Имя | Тип | Описание |
|---|---|---|
| options | IOptionManager | Менеджер опций.<br>Унаследовано от IControl. |
| state | data.Manager | Состояние элемента управления. Имена полей, доступных через метод data.Manager.get:<br>• fullscreen — признак, находится ля карта в полноэкранном режиме. |

## События

| Имя | Описание |
|---|---|
| click | Нажатие на кнопку. Экземпляр класса Event.<br>Унаследовано от control.Button. |
| deselect | Элемент управления не выбран.<br>Унаследовано от ISelectableControl. |
| disable | Элемент управления стал недоступным.<br>Унаследовано от ISelectableControl. |
| enable | Элемент управления стал доступным.<br>Унаследовано от ISelectableControl. |
| fullscreenenter | Карта перешла в полноэкранный режим. Экземпляр класса Event. |
| fullscreenexit | Карта вышла из полноэкранного режима. Экземпляр класса Event. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| press | Событие, сигнализирующее о том, что кнопка была нажата. В отличии от click генерируется только если состояние isEnabled == true. Экземпляр класса Event.<br>Унаследовано от control.Button. |
| select | Элемент управления выбран.<br>Унаследовано от ISelectableControl. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| deselect() | | Отменяет выбор элемента управления (выключает).<br>Унаследован от ISelectableControl. |
| disable() | | Делает элемент управления недоступным (запрещены пользовательские действия). |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **ISelectableControl**. |
| **enable**() | | Делает элемент управления доступным (разрешены пользовательские действия).<br>Унаследован от **ISelectableControl**. |
| **enterFullscreen**() | | Позволяет переводить карту в полноэкранный режим. |
| **exitFullscreen**() | | Позволяет вывести карту из полноэкранного режима. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **control.Button**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IControl**. |
| **isEnabled**() | Boolean | Возвращает true — элемент управления доступен, false — недоступен.<br>Унаследован от **ISelectableControl**. |
| **isSelected**() | Boolean | Возвращает true — элемент управления выбран, false — не выбран.<br>Унаследован от **ISelectableControl**. |
| **select**() | | Выбирает (включает) элемент управления.<br>Унаследован от **ISelectableControl**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IControl**. |

## Описание полей

### state

```
{data.Manager} state
```

Состояние элемента управления. Имена полей, доступных через метод data.Manager.get:

- fullscreen — признак, находится ля карта в полноэкранном режиме.

## Описание событий

### fullscreenenter

Карта перешла в полноэкранный режим. Экземпляр класса **Event**.

### fullscreenexit

Карта вышла из полноэкранного режима. Экземпляр класса **Event**.

## Описание методов

### enterFullscreen

```
enterFullscreen()
```

Позволяет переводить карту в полноэкранный режим.

### exitFullscreen

```
exitFullscreen()
```

Позволяет вывести карту из полноэкранного режима.

# control.GeolocationControl

Расширяет **control.Button**.

Элемент управления "геолокация". Позволяет отразить положение пользователя на карте. Ключ элемента управления в хранилище **control.storage** — "geolocationControl".

**См.: geolocation**

Конструктор | Поля | События | Методы

## Конструктор

```
control.GeolocationControl([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: Object<br><br>Параметры элемента управления. |
| parameters.data | — | Тип: Object<br><br>Объект, описывающий данные элемента управления. |
| parameters.data.image | 'geolocation' | Тип: String<br><br>URL иконки кнопки. |
| parameters.data.title | — | Тип: String<br><br>Текст всплывающей подсказки, которая появляется при наведении на кнопку курсора мыши. |
| parameters.options | — | Тип: Object<br><br>Опции элемента управления. |
| parameters.options.adjustMapMargin | false | Тип: Boolean<br><br>Регистрирует ли элемент управления свои размеры в менеджере отступов карты **map.margin.Manager**. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters.options.float | "right" | Тип: String<br><br>Сторона, по которой нужно выравнивать элемент управления. Может принимать три значения: "left", "right" или "none". При значении "left" или "right" элементы управления выстраиваются друг за другом, начиная от левого или правого края карты соответственно. При значении "none" элементы управления позиционируется только по значениям опций left, right, bottom, top относительно границ карты. Также смотрите описание опции position. |
| parameters.options.floatIndex | 300 | Тип: Number<br><br>Приоритет расположения элемента управления. Элемент с максимальным приоритетом находится ближе к указанному в свойстве float краю карты. Не работает при float = "none". |
| parameters.options.maxWidth | 28 | Тип: Number \| Number[]<br><br>Максимальная ширина элемента управления в различных состояниях. Если задано число, то считается, что элемент управления имеет одинаковые максимальные размеры во всех состояниях. Если задан массив, то он будет трактоваться как максимальная ширина в различных состояниях — от меньшего к большему. Количество состояний задается в экземпляре класса **control.Manager**, который обычно является полем **Map.controls**, через опцию states. По умолчанию элемент управления не изменяет своего размера, всегда имея вид кнопки с иконкой. |
| parameters.options.noPlacemark | false | Тип: Boolean<br><br>При значении true метка-местоположение не будет показана на карте, а так же не произойдет автоматической смены центра и масштабирования карты. |
| parameters.options.position | — | Тип: Object<br><br>Объект, описывающий позицию элемента управления. При указании опции position значение опции float автоматически трактуется как "none". |
| parameters.options.position.bottom | 'auto' | Тип: Number \| String<br><br>Положение относительно нижнего края карты. |
| parameters.options.position.left | 'auto' | Тип: Number \| String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Положение относительно левого края карты. |
| parameters.options.position.right | 'auto' | Тип: Number \| String<br><br>Положение относительно правого края карты. |
| parameters.options.position.top | 'auto' | Тип: Number \| String<br><br>Положение относительно верхнего края карты. |
| parameters.options.visible | true | Тип: Boolean<br><br>Признак того, что элемент управления отображается. |
| parameters.state | — | Тип: Object<br><br>Объект, описывающий состояние элемента управления. |
| options.useMapMargin | true | Тип: Boolean<br><br>Нужно ли учитывать отступы карты **map.margin.Manager** при центрировании карты. |

**Пример:**

```
// Добавим элемент управления с собственной меткой геолокации на карте.
var geolocationControl = new ymaps.control.GeolocationControl({
    options: {noPlacemark: true}
});
geolocationControl.events.add('locationchange', function (event) {
    var position = event.get('position'),
    // При создании метки можно задать ей любой внешний вид.
        locationPlacemark = new ymaps.Placemark(position);

    myMap.geoObject.add(locationPlacemark);
    // Установим новый центр карты в текущее местоположение пользователя.
    myMap.panTo(position);
});
myMap.controls.add(geolocationControl);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **data** | **data.Manager** | Данные кнопки. Имена полей, доступных через метод data.Manager.get:<br>• image — иконка кнопки, если есть;<br>• content — содержимое кнопки в виде HTML;<br>• title — текст всплывающей подсказки, которая появляется при наведении на кнопку курсора мыши.<br><br>Унаследовано от **control.Button**. |
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

| Имя | Тип | Описание |
|---|---|---|
| options | IOptionManager | Менеджер опций.<br>Унаследовано от IControl. |
| state | data.Manager | Состояние кнопки. Имена полей, доступных через метод data.Manager.get:<br>• selected — признак, нажата ли кнопка;<br>• enabled — признак, активна ли кнопка;<br>• size — размер, который в данный момент выставлен кнопке;<br>Унаследовано от control.Button. |

## События

| Имя | Описание |
|---|---|
| click | Нажатие на кнопку. Экземпляр класса Event.<br>Унаследовано от control.Button. |
| deselect | Элемент управления не выбран.<br>Унаследовано от ISelectableControl. |
| disable | Элемент управления стал недоступным.<br>Унаследовано от ISelectableControl. |
| enable | Элемент управления стал доступным.<br>Унаследовано от ISelectableControl. |
| locationchange | Событие определения положения пользователя. Список полей события, доступных через метод Event.get:<br>• position — положение пользователя в географических координатах.<br>• geoObjects — экземпляр класса GeoObjectCollection, в котором находится объект, обозначающий текущее местоположение пользователя. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| press | Событие, сигнализирующее о том, что кнопка была нажата. В отличии от click генерируется только если состояние isEnabled == true. Экземпляр класса Event.<br>Унаследовано от control.Button. |
| select | Элемент управления выбран.<br>Унаследовано от ISelectableControl. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| deselect() | | Отменяет выбор элемента управления (выключает).<br>Унаследован от ISelectableControl. |
| disable() | | Делает элемент управления недоступным (запрещены пользовательские действия).<br>Унаследован от ISelectableControl. |
| enable() | | Делает элемент управления доступным (разрешены пользовательские действия).<br>Унаследован от ISelectableControl. |
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от control.Button. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IControl. |
| isEnabled() | Boolean | Возвращает true — элемент управления доступен, false — недоступен.<br>Унаследован от ISelectableControl. |
| isSelected() | Boolean | Возвращает true — элемент управления выбран, false — не выбран.<br>Унаследован от ISelectableControl. |
| select() | | Выбирает (включает) элемент управления.<br>Унаследован от ISelectableControl. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IControl. |

## Описание событий

### locationchange

Событие определения положения пользователя. Список полей события, доступных через метод Event.get:

- position — положение пользователя в географических координатах.
- geoObjects — экземпляр класса GeoObjectCollection, в котором находится объект, обозначающий текущее местоположение пользователя.

# control.ListBox

Расширяет **ICollection**, **IControl**, **ICustomizable**.

Класс для создания элемента управления в виде раскрывающегося списка. Стандартный макет выпадающего списка изменяет свой внешний вид в зависимости от размеров карты. Если карта имеет большую ширину, то в заголовке выпадающего списка отображается текст. Если карта небольшого размера, то в заголовке списка отображается только иконка. Если для кнопки не задана иконка, то во всех состояниях будет отображаться текст и наоборот.

Конструктор | Поля | События | Методы

## Конструктор

```
control.ListBox([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: Object<br><br>Параметры выпадающего списка. |
| parameters.data | — | Тип: Object<br><br>Данные. |
| parameters.data.content | — | Тип: String<br><br>Заголовок списка. |
| parameters.data.image | — | Тип: String<br><br>URL иконки кнопки. Стандартный макет кнопки рассчитан на иконку размером 16x16 пикселей. |
| parameters.data.title | — | Тип: String<br><br>Текст всплывающей подсказки, которая появляется при наведении на список курсора мыши. |
| parameters.items | — | Тип: **IControl**[]<br><br>Массив дочерних элементов списка. |
| parameters.options | — | Тип: Object<br><br>Опции элемента управления. |
| parameters.options.adjustMapMargin | false | Тип: Boolean<br><br>Регистрирует ли элемент управления свои размеры в менеджере отступов карты **map.margin.Manager**. |
| parameters.options.collapseOnBlur | true | Тип: Boolean<br><br>Флаг, разрешающий сворачиваение списка при потере фокуса на кнопке. Например при клике на документ. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters.options.expandOnClick | true | Тип: Boolean<br><br>Флаг, разрешающий автоматическое разворачивание/сворачивание списка по клику. |
| parameters.options.float | "right" | Тип: String<br><br>Сторона, по которой нужно выравнивать элемент управления. Может принимать три значения: "left", "right" или "none". При значении "left" или "right" элементы управления выстраиваются друг за другом, начиная от левого или правого края карты соответственно. При значении "none" элементы управления позиционируется только по значениям опций left, right, bottom, top относительно границ карты. Также смотрите описание опции position. |
| parameters.options.floatIndex | 0 | Тип: Number<br><br>Приоритет расположения элемента управления. Элемент с максимальным приоритетом находится ближе к указанному в свойстве float краю карты. Не работает при float = "none". |
| parameters.options.layout | — | Тип: Function \| String<br><br>Конструктор макета элемента управления, реализующий интерфейсы **ISelectableControlLayout** и **IGroupControlLayout** или ключ макета в хранилище **layout.storage**. В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.ListBox.options;<br>• data — менеджер данных элемента управления **control.ListBox.data**;<br>• state — менеджер состояния элемента управления **control.ListBox.state**.<br><br>Макет меняет свой внешний вид на основе данных, состояния и опций элемента управления. Элемент управления, в свою очередь, реагирует на интерфейсные события макета и меняет значения полей **control.ListBox.state** в зависимости от полученных команд. |
| parameters.options.maxWidth | 90 | Тип: Number \| Number[] |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Максимальная ширина кнопки списка в различных состояниях. Если задано число, то считается, что кнопка имеет одинаковые максимальные размеры во всех состояниях. Если задан массив, то он будет трактоваться как максимальная ширина кнопки в различных состояниях — от меньшего к большему. Количество состояний задается в экземпляре класса **control.Manager**, который обычно является полем **Map.controls**, через опцию states. По умолчанию у элементов управления есть три состояния — ['small', 'medium', 'large']. |
| parameters.options.popupFloat | — | Тип: String<br><br>Сторона, по которой нужно выравнивать попап элемента управления. Может принимать два значения: "left" и "right". По умолчанию определяется согласно опции "float". |
| parameters.options.position | — | Тип: Object<br><br>Объект, описывающий позицию элемента управления. При указании опции position значение опции float автоматически трактуется как "none". |
| parameters.options.position.bottom | 'auto' | Тип: Number \| String<br><br>Положение относительно нижнего края карты. |
| parameters.options.position.left | 'auto' | Тип: Number \| String<br><br>Положение относительно левого края карты. |
| parameters.options.position.right | 'auto' | Тип: Number \| String<br><br>Положение относительно правого края карты. |
| parameters.options.position.top | 'auto' | Тип: Number \| String<br><br>Положение относительно верхнего края карты. |
| parameters.options.visible | true | Тип: Boolean<br><br>Признак того, что элемент управления отображается. |
| parameters.state | — | Тип: Object<br><br>Состояние выпадающего списка. |
| parameters.state.expanded | false | Тип: Boolean<br><br>Признак, развернут ли список. |

**Примеры:**

**1.**

```
// Пример 1.
// Обработка клика на элементах списка.
var cityList = new ymaps.control.ListBox({
        data: {
            content: 'Выбрать город'
        },
        items: [
            new ymaps.control.ListBoxItem('Москва'),
            new ymaps.control.ListBoxItem('Новосибирск'),
            new ymaps.control.ListBoxItem({options: {type: 'separator'}}),
            new ymaps.control.ListBoxItem('Нью-Йорк'),
        ]
    });
cityList.get(0).events.add('click', function () {
    map.setCenter([55.752736, 37.606815]);
});
cityList.get(1).events.add('click', function () {
    map.setCenter([55.026366, 82.907803]);
});
cityList.get(3).events.add('click', function () {
    map.setCenter([40.695537, -73.97552]);
});
map.controls.add(cityList, { floatIndex: 0 });
```

**2.**

```
// Пример 2.
// Создание пользовательского списка.
// В примере используется jQuery, загруженный с http://yandex.st/jquery/1.6.4/
jquery.min.js

// Выпадающий список по умолчанию реагирует на события 'click' и автоматически
// меняет состояние на развернутое или свернутое.
var MyListBoxLayout = ymaps.templateLayoutFactory.createClass(
        '<div id="my-listbox-header" >{{ data.title }}</div >' +
        // Этот элемент будет служить контейнером для дочерних элементов списка.
       '<div id="my-list-box" style="display: {% if state.expanded %}block{% else
%}none{% endif %};" >' +
        '</div >', {

        build: function() {
            MyListBoxLayout.superclass.build.call(this);
            this.childContainerElement = $('#my-list-box').get(0);
            // Каждый раз при перестроении будем генерировать событие,
            // сигнализирующее о смене контейнера для дочерних элементов.
            // Формат события описан в интерфейсе IGroupControlLayout.
            this.events.fire('childcontainerchange', {
                newChildContainerElement: this.childContainerElement,
                oldChildContainerElement: null
            });
        },

        // Переопределим метод, который требует интерфейс IGroupControlLayout.
        getChildContainerElement: function () {
            return this.childContainerElement;
        }
    }),
    // Создадим список и выставим созданный макет через опции.
    listBox = new ymaps.control.ListBox({options: {layout: MyListBoxLayout}});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| data | data.Manager | Данные выпадающего списка. Имена полей, доступных через метод data.Manager.get:<br>• content — заголовок выпадающего списка;<br>• title — текст всплывающей подсказки, которая появляется при наведении на список курсора мыши. |
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от IControl. |
| state | data.Manager | Состояние выпадающего списка. Имена полей, доступных через метод data.Manager.get:<br>• expanded — признак, раскрыт ли список;<br>• size — размер, который в данный момент выставлен списку; |

## События

| Имя | Описание |
|---|---|
| add | Произошло добавление дочернего объекта.<br>Унаследовано от ICollection. |
| click | Щелчок по заголовку списка. Экземпляр класса Event. |
| collapse | Список закрыт. Экземпляр класса Event. |
| expand | Список открыт. Экземпляр класса Event. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| press | Событие, сигнализирующее о том, что кнопка была нажата. В отличии от click генерируется только если состояние isEnabled == true. Экземпляр класса Event. |
| remove | Произошло удаление дочернего объекта.<br>Унаследовано от ICollection. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| add(object) | ICollection | Добавляет в коллекцию дочерний объект.<br>Унаследован от ICollection. |

| Имя | Возвращает | Описание |
|---|---|---|
| collapse() | control.ListBox | Сворачивает список. |
| expand() | control.ListBox | Раскрывает список. |
| getIterator() | IIterator | Возвращает итератор по коллекции.<br>Унаследован от ICollection. |
| getMap() | Map | Возвращает ссылку на карту. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IControl. |
| isExpanded() | Boolean | Возвращает признак того, находится ли контрол в развернутом состоянии. |
| remove(object) | ICollection | Удаляет из коллекции дочерний объект.<br>Унаследован от ICollection. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IControl. |

## Описание полей

### data

```
{data.Manager} data
```

Данные выпадающего списка. Имена полей, доступных через метод data.Manager.get:

- content — заголовок выпадающего списка;
- title — текст всплывающей подсказки, которая появляется при наведении на список курсора мыши.

**Пример:**

```
// Добавим на карту выпадающий список и будем менять его подсказку
// в зависимости от того, свернут список или развернут.

// Создадим группу слушателей событий.
var listBoxListener = listBox.events.group()
    .add('expand', function () {
        listBox.data.set('title', 'Список развернут');
    })
    .add('collapse', function () {
        listBox.data.set('title', 'Список свернут');
    });
map.controls.add(listBox, {float: 'none', top: 10, left: 10});
// ...
map.controls.remove(listBox);
// После удаления элемента с карты удалим слушателей.
listBoxListener.removeAll();
```

### state

```
{data.Manager} state
```

Состояние выпадающего списка. Имена полей, доступных через метод data.Manager.get:

- expanded — признак, раскрыт ли список;
- size — размер, который в данный момент выставлен списку;

**Пример:**

```
// Создадим и добавим на карту список, который изначально открыт.
var listBox = new ymaps.control.ListBox();
listBox.state.set('expanded', true);
map.controls.add(listBox);
```

## Описание событий

### click

Щелчок по заголовку списка. Экземпляр класса **Event**.

### collapse

Список закрыт. Экземпляр класса **Event**.

### expand

Список открыт. Экземпляр класса **Event**.

### press

Событие, сигнализирующее о том, что кнопка была нажата. В отличии от click генерируется только если состояние isEnabled == true. Экземпляр класса **Event**.

## Описание методов

### collapse

```
{control.ListBox} collapse()
```

Сворачивает список.

**Возвращает** ссылку на себя.

### expand

```
{control.ListBox} expand()
```

Раскрывает список.

**Возвращает** ссылку на себя.

### getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

### isExpanded

```
{Boolean} isExpanded()
```

**Возвращает** признак того, находится ли контрол в развернутом состоянии.

# control.ListBoxItem

Расширяет **ICustomizable**, **ISelectableControl**.

Элемент раскрывающегося списка.

Конструктор | Поля | События | Методы

## Конструктор

```
control.ListBoxItem([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: Object \| String<br><br>Параметры элемента либо строка — HTML-содержимое элемента. |
| parameters.data | — | Тип: Object<br><br>Данные элемента. |
| parameters.data.content | — | Тип: String<br><br>Содержимое элемента. |
| parameters.options | — | Тип: Object<br><br>Опции элемента управления. |
| parameters.options.layout | 'islands#listBoxItemLayout' | Тип: Function \| String<br><br>Конструктор макета элемента управления, реализующий интерфейс **ISelectableControlLayout** или ключ макета в хранилище **layout.storage**. Это базовый макет, который в зависимости от значения опции type выбирает, какой из подмакетов отобразить — options.selectableLayout или options.separatorLayout. В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.ListBoxItem.options;<br>• data — менеджер данных элемента управления **control.ListBoxItem.data**;<br>• state — менеджер состояния элемента управления **control.ListBoxItem.state**. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Макет меняет свой внешний вид на основе данных, состояния и опций элемента управления. Элемент управления, в свою очередь, реагирует на интерфейсные события макета и меняет значения полей **control.ListBoxItem.state** в зависимости от полученных команд. |
| parameters.options.selectableLayout | 'islands#listBoxItemSelectableLayout' | Тип: Function \| String<br><br>Конструктор макета элемента списка, реализующий интерфейс **ISelectableControlLayout** или ключ макета в хранилище **layout.storage**. Применяется к элементам с опцией type='item'. Опция стандартной реализации макета элемента списка. |
| parameters.options.selectOnClick | true | Тип: Boolean<br><br>Флаг, разрешающий автоматическое выделение элемента списка по клику.<br>• true — элемент списка выделяется после клика и меняет значение поля **control.ListBoxItem.state** 'selected';<br>• false — элемент списка не меняет свой внешний вид и состояние после клика. |
| parameters.options.separatorLayout | 'islands#listBoxItemSeparatorLayout' | Тип: Function \| String<br><br>Конструктор макета разделителя элементов списка, реализующий интерфейс IControlLayout или ключ макета в хранилище **layout.storage**. Применяется к элементам с опцией type='separator'. Опция стандартной реализации макета элемента списка. |
| parameters.options.type | 'selectable' | Тип: String<br><br>Тип пункта меню. В зависимости от значения этой опции макет элемента списка инстанцирует один из подмакетов — options.selectableLayout или options.separatorLayout. Возможные значения:<br>• 'selectable' — элемент списка, выделяется галочкой справа от содержимого;<br>• 'separator' — разделитель. |
| parameters.options.visible | true | Тип: Boolean<br><br>Признак того, что элемент управления отображается. |
| parameters.state | — | Тип: Object |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Объект, описывающий состояние пункта меню. |
| parameters.state.selected | false | Тип: Boolean<br><br>Признак, выбран ли пункт. |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **data** | **data.Manager** | Данные элемента списка. Имена полей, доступных через метод data.Manager.get:<br>• content — содержимое элемента списка в виде HTML;<br>• title — текст всплывающей подсказки. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **state** | **data.Manager** | Состояние элемента списка. Имена полей, доступных через метод data.Manager.get:<br>• selected — признак, выбран ли элемент списка;<br>• enabled — признак, активен ли элемент списка. |

## События

| Имя | Описание |
|---|---|
| **click** | Выбор элемента списка. |
| **deselect** | Элемент управления не выбран.<br>Унаследовано от **ISelectableControl**. |
| **disable** | Элемент управления стал недоступным.<br>Унаследовано от **ISelectableControl**. |
| **enable** | Элемент управления стал доступным.<br>Унаследовано от **ISelectableControl**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **select** | Элемент управления выбран.<br>Унаследовано от **ISelectableControl**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| deselect() | | Отменяет выбор элемента управления (выключает).<br>Унаследован от ISelectableControl. |
| disable() | | Делает элемент управления недоступным (запрещены пользовательские действия).<br>Унаследован от ISelectableControl. |
| enable() | | Делает элемент управления доступным (разрешены пользовательские действия).<br>Унаследован от ISelectableControl. |
| getMap() | Map | Возвращает ссылку на карту. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IControl. |
| isEnabled() | Boolean | Возвращает true — элемент управления доступен, false — недоступен.<br>Унаследован от ISelectableControl. |
| isSelected() | Boolean | Возвращает true — элемент управления выбран, false — не выбран.<br>Унаследован от ISelectableControl. |
| select() | | Выбирает (включает) элемент управления.<br>Унаследован от ISelectableControl. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IControl. |

## Описание полей

### data

```
{data.Manager} data
```

Данные элемента списка. Имена полей, доступных через метод data.Manager.get:

- content — содержимое элемента списка в виде HTML;
- title — текст всплывающей подсказки.

### state

```
{data.Manager} state
```

Состояние элемента списка. Имена полей, доступных через метод data.Manager.get:

- selected — признак, выбран ли элемент списка;
- enabled — признак, активен ли элемент списка.

## Описание событий

### click

Выбор элемента списка.

## Описание методов

### getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

# control.Manager

Менеджер элементов управления.

Конструктор | Поля | Методы

## Конструктор

```
control.Manager(map[, controls[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Экземпляр карты. |
| controls | — | Тип: String[] \| **IControl**[]<br><br>Элементы управления, которые требуется разместить на карте изначально. |
| options | — | Тип: Object<br><br>Опции менеджера. |
| options.margin | 10 | Тип: Number<br><br>Отступ элементов управления друг от друга и от краёв карты. Задаётся в пикселях. |
| options.pane | — | Тип: **IPane**<br><br>Контейнер для всех элементов управления. |
| options.states | ['small', 'medium', 'large'] | Тип: String[]<br><br>Массив размеров, начиная от наименьшего к наибольшему. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **event.Manager** | Менеджер событий. |
| **options** | **option.Manager** | Опции менеджера. |
| **state** | **data.Manager** | Состояние менеджера. Имена полей, доступных через метод data.Manager.get:<br>• size — состояние в котором находятся элементы управления. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(control[, options]) | **control.Manager** | Добавляет элемент управления в менеджер. |
| **each**(callback, context) | **control.Manager** | Вызывает функцию-обработчик для всех элементов управления. |
| **get**(index) | **IControl** \| null | Возвращает элемент управления или null, если элемент не найден. |
| **getChildElement**(control) | **vow.Promise** | Возвращает объект-обещание, который подтверждается HTML-элементом, в который должен разместиться дочерний элемент. |
| **getContainer**() | HTMLElement | Возвращает контейнер, в который добавляются элементы управления. |
| **getMap**() | **Map** | Возвращает ссылку на карту. |
| **indexOf**(childToFind) | Integer | Возвращает -1, если элемент управления не найден, либо индекс элемента в менеджере. |
| **remove**(control) | **control.Manager** | Удаление элемента управления из менеджера. |

# Описание полей

## events

```
{event.Manager} events
```

Менеджер событий.

## options

```
{option.Manager} options
```

Опции менеджера.

## state

```
{data.Manager} state
```

Состояние менеджера. Имена полей, доступных через метод data.Manager.get:

- size — состояние в котором находятся элементы управления.

## Описание методов

### add

```
{control.Manager} add(control[, options])
```

Добавляет элемент управления в менеджер.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| control* | — | Тип: **IControl** \| String<br><br>Элементы управления, заданные экземплярами классов, реализующих интерфейс **IControl** либо ключами.<br><br>Возможные значения ключей:<br>• "fullscreenControl" — кнопка разворачивания карты на весь экран **control.FullscreenControl**;<br>• "geolocationControl" — кнопка определения местоположения пользователя **control.GeolocationControl**;<br>• "routeEditor" — кнопка включения и отключения поведения "редактор маршрута" **control.RouteEditor**;<br>• "rulerControl" — кнопка включения и отключения поведения "линейка" **control.RulerControl**;<br>• "searchControl" — панель поиска **control.SearchControl**;<br>• "trafficControl" — панель пробок **control.TrafficControl**;<br>• "typeSelector" — панель переключения типа карты **control.TypeSelector**;<br>• "zoomControl" — ползунок масштаба **control.ZoomControl**; |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Кроме того, вы можете задавать один из предустановленных наборов элементов управления с помощью специальных ключей:<br>• "smallMapDefaultSet" — базовый набор элементов управления, оптимизированный для карт небольшого размера и экранов телефонов. Состоит из элементов управления "zoomControl", "searchControl", "typeSelector", "geolocationControl" и "fullscreenControl". В этом наборе все элементы управления минимизированы до кнопок с пиктограммами.<br>• "mediumMapDefaultSet" — набор элементов управления, оптимизированный для карт среднего размера и экранов планшетов. Помимо базового набора элементов управления (см. выше), добавляет дополнительно элементы управления "rulerControl" и "trafficControl".<br>• "largeMapDefaultSet" — набор элементов управления, оптимизированный для карт большого размера и экранов десктопных компьютеров. Отличается от набора "mediumMapDefaultSet" наличием элемента управления "routeEditor" и отсутствием элемента управления "fullScreen";<br>• "default" — набор элементов управления по умолчанию; синоним для набора "mediumMapDefaultSet". |
| options | — | Тип: Object<br><br>Опции элемента управления. |
| options.float | "right" | Тип: String<br><br>Сторона, по которой нужно выравнивать элемент управления. Может принимать три значения: "left", "right" или "none". При значении "left" или "right" элементы управления выстраиваются друг за другом, начиная от левого или правого края карты, соответственно. При значении "none" элементы управления позиционируется только по значениям опций left, right, bottom, top относительно границ карты. |
| options.floatIndex | 0 | Тип: Number |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Приоритет расположения элемента управления. Элемент с максимальным приоритетом находится ближе к указанному в свойстве float краю карты. Не работает при float = "none". |
| options.position | — | Тип: Object<br><br>Объект, описывающий позицию элемента управления. При указании опции position значение опции float автоматически трактуется как "none". |
| options.position.bottom | 'auto' | Тип: Number \| String<br><br>Положение относительно нижнего края карты. Работает только при float = none. |
| options.position.left | 'auto' | Тип: Number \| String<br><br>Положение относительно левого края карты. Работает только при float = none. |
| options.position.right | 'auto' | Тип: Number \| String<br><br>Положение относительно правого края карты. Работает только при float = none. |
| options.position.top | 'auto' | Тип: Number \| String<br><br>Положение относительно верхнего края карты. Работает только при float = none. |

* Обязательный параметр/опция.

**Пример:**

```
map.controls
    .add('zoomControl')
    .add('typeSelector');
```

## each

```
{control.Manager} each(callback, context)
```

Вызывает функцию-обработчик для всех элементов управления.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик. На вход получает элемент коллекции. При возвращении функцией значения false перебор останавливается. |
| context* | — | Тип: Object |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Контекст вызываемой функции. |

* Обязательный параметр/опция.

## get

```
{IControl | null} get(index)
```

**Возвращает** элемент управления или null, если элемент не найден.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Number \| String<br><br>Индекс искомого элемента или его ключ. |

* Обязательный параметр/опция.

## getChildElement

```
{vow.Promise} getChildElement(control)
```

**Возвращает** объект-обещание, который подтверждается HTML-элементом, в который должен разместиться дочерний элемент.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| control* | — | Тип: **IControl**<br><br>Элемент управления. |

* Обязательный параметр/опция.

## getContainer

```
{HTMLElement} getContainer()
```

**Возвращает** контейнер, в который добавляются элементы управления.

## getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

## indexOf

```
{Integer} indexOf(childToFind)
```

**Возвращает** -1, если элемент управления не найден, либо индекс элемента в менеджере.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| childToFind* | — | Тип: String \| **IControl** |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Элемент управления или его ключ. |

* Обязательный параметр/опция.

### remove

```
{control.Manager} remove(control)
```

Удаление элемента управления из менеджера.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| control* | — | Тип: **IControl** \| String<br><br>Удаляемый элемент управления, либо его ключ. |

* Обязательный параметр/опция.

## control.RouteEditor

Расширяет **control.Button**.

Элемент управления "Редактор маршрутов". Ключ элемента управления в хранилище **control.storage** — "routeEditor".

Конструктор | Поля | События | Методы

### Конструктор

```
control.RouteEditor([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: Object<br><br>Параметры элемента управления. |
| parameters.data | — | Тип: Object<br><br>Объект, описывающий данные элемента управления. |
| parameters.data.image | 'routes' | Тип: String<br><br>URL иконки кнопки. |
| parameters.data.title | — | Тип: String<br><br>Текст всплывающей подсказки, которая появляется при наведении на кнопку курсора мыши. |
| parameters.options | — | Тип: Object<br><br>Опции элемента управления. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters.options.adjustMapMargin | false | Тип: Boolean<br><br>Регистрирует ли элемент управления свои размеры в менеджере отступов карты **map.margin.Manager**. |
| parameters.options.float | "right" | Тип: String<br><br>Сторона, по которой нужно выравнивать элемент управления. Может принимать три значения: "left", "right" или "none". При значении "left" или "right" элементы управления выстраиваются друг за другом, начиная от левого или правого края карты соответственно. При значении "none" элементы управления позиционируется только по значениям опций left, right, bottom, top относительно границ карты. Также смотрите описание опции position. |
| parameters.options.floatIndex | 100 | Тип: Number<br><br>Приоритет расположения элемента управления. Элемент с максимальным приоритетом. |
| parameters.options.layout | — | Тип: **ISelectableControlLayout** \| String<br><br>Конструктор макета элемента управления, реализующий интерфейс **ISelectableControlLayout** или ключ макета в хранилище **layout.storage**. В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.FullscreenControl.options;<br>• data — менеджер данных элемента управления control.FullscreenControl.data;<br>• state — менеджер состояния элемента управления **control.FullscreenControl.state**.<br><br>Макет меняет свой внешний вид на основе данных, состояния и опций элемента управления. Элемент управления, в свою очередь, реагирует на интерфейсные события макета |
| parameters.options.maxWidth | 28 | Тип: Number \| Number[] |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Максимальная ширина элемента управления в различных состояниях. Если задано число, то считается, что элемент управления имеет одинаковые максимальные размеры во всех состояниях. Если задан массив, то он будет трактоваться как максимальная ширина в различных состояниях — от меньшего к большему. Количество состояний задается в экземпляре класса **control.Manager**, который обычно является полем **Map.controls**, через опцию states. По умолчанию у элементов управления есть три состояния — ['small', 'medium', 'large']. По умолчанию элемент управления не изменяет своего размера, всегда имея вид кнопки с иконкой. |
| parameters.options.position | — | Тип: Object<br><br>Объект, описывающий позицию элемента управления. При указании опции position значение опции float автоматически трактуется как "none". |
| parameters.options.position.bottom | 'auto' | Тип: Number \| String<br><br>Положение относительно нижнего края карты. |
| parameters.options.position.left | 'auto' | Тип: Number \| String<br><br>Положение относительно левого края карты. |
| parameters.options.position.right | 'auto' | Тип: Number \| String<br><br>Положение относительно правого края карты. |
| parameters.options.position.top | 'auto' | Тип: Number \| String<br><br>Положение относительно верхнего края карты. |
| parameters.options.visible | true | Тип: Boolean<br><br>Признак того, что элемент управления отображается. |
| parameters.state | — | Тип: Object<br><br>Объект, описывающий состояние элемента управления. |

**Пример:**

```
// Добавляем контрол на карту.
map.controls.add('routeEditor');
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **data** | **data.Manager** | Данные кнопки. Имена полей, доступных через метод data.Manager.get:<br>• image — иконка кнопки, если есть;<br>• content — содержимое кнопки в виде HTML;<br>• title — текст всплывающей подсказки, которая появляется при наведении на кнопку курсора мыши.<br>Унаследовано от **control.Button**. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **IControl**. |
| **state** | **data.Manager** | Состояние кнопки. Имена полей, доступных через метод data.Manager.get:<br>• selected — признак, нажата ли кнопка;<br>• enabled — признак, активна ли кнопка;<br>• size — размер, который в данный момент выставлен кнопке;<br>Унаследовано от **control.Button**. |

## События

| Имя | Описание |
|---|---|
| **click** | Нажатие на кнопку. Экземпляр класса **Event**.<br>Унаследовано от **control.Button**. |
| **deselect** | Элемент управления не выбран.<br>Унаследовано от **ISelectableControl**. |
| **disable** | Элемент управления стал недоступным.<br>Унаследовано от **ISelectableControl**. |
| **enable** | Элемент управления стал доступным.<br>Унаследовано от **ISelectableControl**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **press** | Событие, сигнализирующее о том, что кнопка была нажата. В отличии от click генерируется только если состояние isEnabled == true. Экземпляр класса **Event**.<br>Унаследовано от **control.Button**. |

| Имя | Описание |
|---|---|
| select | Элемент управления выбран.<br>Унаследовано от ISelectableControl. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| deselect() | | Отменяет выбор элемента управления (выключает).<br>Унаследован от ISelectableControl. |
| disable() | | Делает элемент управления недоступным (запрещены пользовательские действия).<br>Унаследован от ISelectableControl. |
| enable() | | Делает элемент управления доступным (разрешены пользовательские действия).<br>Унаследован от ISelectableControl. |
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от control.Button. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IControl. |
| getRoute() | router.Route | Возвращает маршрут. |
| isEnabled() | Boolean | Возвращает true — элемент управления доступен, false — недоступен.<br>Унаследован от ISelectableControl. |
| isSelected() | Boolean | Возвращает true — элемент управления выбран, false — не выбран.<br>Унаследован от ISelectableControl. |
| select() | | Выбирает (включает) элемент управления.<br>Унаследован от ISelectableControl. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IControl. |

## Описание методов

### getRoute

```
{router.Route} getRoute()
```

**Возвращает** маршрут.

# control.RulerControl

Расширяет **control.Button**.

Элемент управления "Линейка". Ключ элемента управления в хранилище **control.storage** — "rulerControl".

Конструктор | Поля | События | Методы

## Конструктор

```
control.RulerControl([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: Object<br><br>Параметры элемента управления. |
| parameters.data | — | Тип: Object<br><br>Объект, описывающий данные элемента управления. |
| parameters.options | — | Тип: Object<br><br>Опции элемента управления. |
| parameters.options.adjustMapMargin | false | Тип: Boolean<br><br>Регистрирует ли элемент управления свои размеры в менеджере отступов карты **map.margin.Manager**. |
| parameters.options.position | — | Тип: Object<br><br>Объект, описывающий позицию элемента управления. При указании опции position значение опции float автоматически трактуется как "none". |
| parameters.options.position.bottom | 30 | Тип: Number \| String<br><br>Положение относительно нижнего края карты. |
| parameters.options.position.left | 'auto' | Тип: Number \| String<br><br>Положение относительно левого края карты. |
| parameters.options.position.right | 10 | Тип: Number \| String<br><br>Положение относительно правого края карты. |
| parameters.options.position.top | 'auto' | Тип: Number \| String<br><br>Положение относительно верхнего края карты. |
| parameters.options.scaleLine | true | Тип: Boolean |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Флаг, указывающий, отображать или не отображать масштабную линейку справа от кнопки линейки. |
| parameters.options.visible | true | Тип: Boolean<br><br>Признак того, что элемент управления отображается. |
| parameters.state | — | Тип: Object<br><br>Объект, описывающий состояние элемента управления. |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **data** | **data.Manager** | Данные кнопки. Имена полей, доступных через метод data.Manager.get:<br>• image — иконка кнопки, если есть;<br>• content — содержимое кнопки в виде HTML;<br>• title — текст всплывающей подсказки, которая появляется при наведении на кнопку курсора мыши.<br>Унаследовано от **control.Button**. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **IControl**. |
| **state** | **data.Manager** | Состояние кнопки. Имена полей, доступных через метод data.Manager.get:<br>• selected — признак, нажата ли кнопка;<br>• enabled — признак, активна ли кнопка;<br>• size — размер, который в данный момент выставлен кнопке;<br>Унаследовано от **control.Button**. |

## События

| Имя | Описание |
|---|---|
| **click** | Нажатие на кнопку. Экземпляр класса **Event**.<br>Унаследовано от **control.Button**. |
| **deselect** | Элемент управления не выбран.<br>Унаследовано от **ISelectableControl**. |
| **disable** | Элемент управления стал недоступным.<br>Унаследовано от **ISelectableControl**. |
| **enable** | Элемент управления стал доступным.<br>Унаследовано от **ISelectableControl**. |
| **optionschange** | Изменение в опциях объекта. |

| Имя | Описание |
|---|---|
| | Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **press** | Событие, сигнализирующее о том, что кнопка была нажата. В отличии от click генерируется только если состояние isEnabled == true. Экземпляр класса **Event**.<br>Унаследовано от **control.Button**. |
| **select** | Элемент управления выбран.<br>Унаследовано от **ISelectableControl**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **deselect**() | | Отменяет выбор элемента управления (выключает).<br>Унаследован от **ISelectableControl**. |
| **disable**() | | Делает элемент управления недоступным (запрещены пользовательские действия).<br>Унаследован от **ISelectableControl**. |
| **enable**() | | Делает элемент управления доступным (разрешены пользовательские действия).<br>Унаследован от **ISelectableControl**. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **control.Button**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IControl**. |
| **isEnabled**() | Boolean | Возвращает true — элемент управления доступен, false — недоступен.<br>Унаследован от **ISelectableControl**. |
| **isSelected**() | Boolean | Возвращает true — элемент управления выбран, false — не выбран.<br>Унаследован от **ISelectableControl**. |
| **select**() | | Выбирает (включает) элемент управления.<br>Унаследован от **ISelectableControl**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br><br>Унаследован от **IControl**. |

## control.SearchControl

Расширяет **IControl**, **ICustomizable**.

Элемент управления "Поиск по карте". Позволяет обрабатывать поисковый запрос пользователя и отображать результат в панели и на карте.

Каждый результат поиска представляется в панели элемента управления в виде двухстрочного блока. Для формирования блока используются поля name и description объекта-результата геокодирования.

Ключ элемента управления в хранилище **control.storage** — "searchControl".
Конструктор | Поля | События | Методы

### Конструктор

```
control.SearchControl([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: Object<br><br>Параметры элемента управления. |
| parameters.data | — | Тип: Object<br><br>Объект, описывающий данные элемента управления. |
| parameters.options | — | Тип: Object<br><br>Опции элемента управления. |
| parameters.options.adjustMapMargin | false | Тип: Boolean<br><br>Регистрирует ли элемент управления свои размеры в менеджере отступов карты **map.margin.Manager**. |
| parameters.options.boundedBy | — | Тип: Number[][]<br><br>Прямоугольная область на карте, где предположительно находится искомый объект. При ранжировании объекты, находящиеся внутри указанной области, будут получать больший приоритет. |
| parameters.options.float | "right" | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Сторона, по которой нужно выравнивать элемент управления. Может принимать три значения: "left", "right" или "none". При значении "left" или "right" элементы управления выстраиваются друг за другом, начиная от левого или правого края карты соответственно. При значении "none" элементы управления позиционируется только по значениям опций left, right, bottom, top относительно границ карты. Также смотрите описание опции position. |
| parameters.options.floatIndex | 200 | Тип: Number<br><br>Приоритет расположения элемента управления. Элемент с максимальным приоритетом находится ближе к указанному в свойстве float краю карты. Не работает при float = "none". |
| parameters.options.formLayout | 'islands#searchControlFormLayout' | Тип: **ILayout** \| String<br><br>Конструктор макета формы для реализации поиска в макете элемента управления по умолчанию или ключ в хранилище **layout.storage**.<br><br>В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.SearchControl.options;<br>• data — менеджер данных элемента управления control.SearchControl.data;<br>• state — менеджер состояния элемента управления **control.SearchControl.state**. |
| parameters.options.kind | 'house' | Тип: String<br><br>Вид топонима (только для обратного геокодирования). Список возможных значений:<br>• house — дом;<br>• street — улица;<br>• metro — станция метро;<br>• district — район города;<br>• locality — населенный пункт (город/поселок/деревня/село/...). |
| parameters.options.layout | 'islands#searchControlLayout' | Тип: **ISearchControlLayout** \| String<br><br>Макет элемента управления. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.SearchControl.options;<br>• data — менеджер данных элемента управления control.SearchControl.data;<br>• state — менеджер состояния элемента управления **control.SearchControl.state**.<br>Макет меняет свой внешний вид на основе данных, состояния и опций элемента управления. Элемент управления, в свою очередь, реагирует на интерфейсные события макета и меняет значения полей **control.SearchControl.state** в зависимости от полученных команд. |
| parameters.options.maxWidth | [30, 72, 290] | Тип: Number \| Number[]<br>Максимальная ширина элемента управления в различных состояниях. Если задано число, то считается, что элемент управления имеет одинаковые максимальные размеры во всех состояниях. Если задан массив, то он будет трактоваться как максимальная ширина в различных состояниях — от меньшего к большему. Количество состояний задается в экземпляре класса **control.Manager**, который обычно является полем **Map.controls**, через опцию states. По умолчанию у элементов управления есть три состояния — ['small', 'medium', 'large']. |
| parameters.options.noCentering | false | Тип: Boolean<br>false — автоматически располагать центр карты так, чтобы объект был виден целиком. Значение true — не изменять центр карты при показе найденного объекта. Если указать noCentering = true и noPlacemark = true, то при щелчке по результату поиска никаких видимых изменений на карте не произойдет. |
| parameters.options.noPlacemark | false | Тип: Boolean |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | false — автоматически добавлять в центр найденного объекта метку с открытым балуном, true — не добавлять. Если указать noCentering = true и noPlacemark = true, то при щелчке по результату поиска никаких видимых изменений на карте не произойдет. Опция не работает с провайдером yandex#search. |
| parameters.options.noPopup | false | Тип: Boolean<br><br>true — не показывать выпадающий список результатов, false — показывать. |
| parameters.options.noSelect | false | Тип: Boolean<br><br>При значении false будет автоматически показываться результат поиска, если он является единственным найденным объектом, true — результат не будет выбираться. |
| parameters.options.noSuggestPanel | false | Тип: Boolean<br><br>true — не показывать панель с поисковыми подсказками, false — показывать. |
| parameters.options.placeholderContent | — | Тип: String<br><br>Текст-подсказка, показываемый в поле ввода элемента управления. |
| parameters.options.popupItemLayout | 'islands#searchControlPopupItemLayout' | Тип: **ILayout** \| String<br><br>Конструктор макета результата поиска в выпадающем списке в макете элемента управления по умолчанию или ключ в хранилище **layout.storage**.<br><br>В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.SearchControl.options;<br>• data — менеджер данных элемента управления control.SearchControl.data;<br>• state — менеджер состояния элемента управления **control.SearchControl.state**. |
| parameters.options.popupLayout | 'islands#searchControlPopupLayout' | Тип: **ILayout** \| String<br><br>Конструктор макета выпадающего списка с результатами поиска в макете элемента управления по умолчанию или ключ в хранилище **layout.storage**. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.SearchControl.options;<br>• data — менеджер данных элемента управления control.SearchControl.data;<br>• state — менеджер состояния элемента управления **control.SearchControl.state**. |
| parameters.options.position | — | Тип: Object<br><br>Объект, описывающий позицию элемента управления. При указании опции position значение опции float автоматически трактуется как "none". |
| parameters.options.position.bottom | 'auto' | Тип: Number \| String<br><br>Положение относительно нижнего края карты. |
| parameters.options.position.left | 'auto' | Тип: Number \| String<br><br>Положение относительно левого края карты. |
| parameters.options.position.right | 'auto' | Тип: Number \| String<br><br>Положение относительно правого края карты. |
| parameters.options.position.top | 'auto' | Тип: Number \| String<br><br>Положение относительно верхнего края карты. |
| parameters.options.provider | 'yandex#map' | Тип: **IGeocodeProvider** \| String<br><br>Провайдер геокодирования. Можно воспользоваться одним из стандартных провайдеров:<br>• 'yandex#map' — поиск топонимов на карте карте;<br>• 'yandex#search' — поиск по топонимам и организациям. Для поиска по организациям не работают следующие опции: «noPlacemark», «noCentering», «noSelect», «strictBounds», «kind». Кроме того, по умолчанию на карте показываются 20 объектов, если не указана опция «results» для элемента управления. |
| parameters.options.searchCoordOrder | 'latlong' | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Определяет каким образом нужно интрепретировать координаты в запросе. По умолчанию координаты будут обрабатываться как широта-долгота. Опция не работает с провайдером yandex#search. |
| parameters.options.size | 'auto' | Тип: String<br><br>Параметр, отвечающий за внешний вид элемента управления. Может принимать значения:<br>• 'small' — всегда показывать кнопку с иконкой;<br>• 'medium' — всегда показывать кнопку с текстом;<br>• 'large' — всегда показывать полную поисковую форму;<br>• 'auto' — производить автоматический выбор размера элемента управления в зависимости от наличия свободного пространства в тулбаре. |
| parameters.options.strictBounds | — | Тип: Boolean<br><br>Искать только внутри области, заданной опцией boundedBy. Объекты вне указанной области попадать в выдачу не будут. |
| parameters.options.suppressYandexSearch | false | Тип: Boolean<br><br>Скрывать ли сообщение с предложением произвести поиск на портале Яндекса, если результаты не были найдены. |
| parameters.options.useMapBounds | — | Тип: Boolean<br><br>Флаг, позволяющий учитывать при поиске границы видимой области карты. При значении true рассчитанная область видимости имеет больший приоритет, чем заданная через boundedBy. |
| parameters.options.zoomMargin | 0 | Тип: Number \| Number[] |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Отступы от границ видимой области карты при показе найденных результатов. Для того чтобы опция применилась, значение noCentering должно быть false. Если задано одно число — оно применяется ко всем сторонам. Если задано два, то это горизонтальные и вертикальные отступы соответственно. Если задан массив из 4х чисел, то это отступы top, right, bottom, left. При включенной опции "useMapMargin" значение "zoomMargin" складывается со значениями, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |
| parameters.state | — | Тип: Object<br><br>Объект, описывающий состояние элемента управления. |
| options.useMapMargin | true | Тип: Boolean<br><br>Нужно ли учитывать отступы карты **map.margin.Manager** при показе результатов поиска на карте. |

**Примеры:**

**1.**

```
// Пример 1.
// Создание элемента управления и добавление его на карту.

var searchControl = new ymaps.control.SearchControl({
     options: {
         float: 'right',
         floatIndex: 100,
         noPlacemark: true
     }
});
myMap.controls.add(searchControl);
```

**2.**

```
// Пример 2.
// Если элемент управления уже добавлен на карту по ключу из хранилища
control.storage,
// то его экземпляр можно получить с помощью метода control.Manager.get
// менеджера элементов управления control.Manager.

var searchControl = myMap.controls.get('searchControl');
searchControl.events.add('submit', function () {
    console.log('request: ' + searchControl.getRequestString());
}, this);
```

**3.**

```
// Пример 3.
// Включим поиск по организациям в элементе управления, уже добавленном на
карту.
// Если элемент управления уже добавлен на карту по ключу из хранилища
control.storage,
// то его экземпляр можно получить с помощью метода control.Manager.get

var searchControl = myMap.controls.get('searchControl');
searchControl.options.set('provider', 'yandex#search');
```

**4.**

```
// Пример 4.
// Создадим элемент управления «поиск по карте» с включенным поиском по
организациям.

var searchControl = new ymaps.control.SearchControl({
     options: {
         float: 'left',
         provider: 'yandex#search'
     }
});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от IControl. |
| state | data.Manager | Состояние элемента управления. Имена полей, доступных через метод data.Manager.get:<br>• size — текущий размер элемента управления;<br>• results — массив, содержащий результаты поиска;<br>• currentIndex — индекс текущего выбранного элемента;<br>• found — общее количество найденных результатов;<br>• request — текущий активный запрос;<br>• correction — исправленный запрос;<br>• noSuggestPanel — флаг, скрывать ли панель поисковых подсказок. |

## События

| Имя | Описание |
|---|---|
| clear | Событие очистки результатов поиска. Экземпляр класса Event. |
| error | Событие ошибки получения результатов поиска с сервера. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• error — информация об ошибке |
| load | Событие получения результатов поиска с сервера. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• skip — сколько элементов было пропущено;<br>• count — количество загруженных элементов. |
| optionschange | Изменение в опциях объекта. |

| Имя | Описание |
|---|---|
| | Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **resultselect** | Событие выбора результата поиска. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• index — индекс выбранного результата. |
| **resultshow** | Событие показа результата поиска. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• index — индекс выбранного результата. |
| **submit** | Событие отправления поискового запроса на сервер Экземпляр класса **Event**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **clear**() | | Очищает результаты поиска, а также содержимое поисковой строки элемента управления. |
| **getMap**() | **Map** | Возвращает ссылку на карту. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IControl**. |
| **getRequestString**() | String | Возвращает поисковый запрос. |
| **getResponseMetaData**() | Object | возвращает метаданные геопоиска. |
| **getResult**(index) | **vow.Promise** | Получение результатов поиска. |
| **getResultsArray**() | Object[] | Возвращает массив, содержащий все текущие результаты. |
| **getResultsCount**() | Integer | Возвращает количество найденых результатов. |
| **getSelectedIndex**() | Integer | Возвращает индекс выбранного элемента. |
| **hideResult**() | | Скрывает результат, показанный на карте. |
| **search**(request) | **vow.Promise** | Осуществляет поиск. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IControl**. |

| Имя | Возвращает | Описание |
|---|---|---|
| showResult(index) | control.SearchControl | Отображает результат с заданным индексом. |

## Описание полей

### state

```
{data.Manager} state
```

Состояние элемента управления. Имена полей, доступных через метод data.Manager.get:

- size — текущий размер элемента управления;
- results — массив, содержащий результаты поиска;
- currentIndex — индекс текущего выбранного элемента;
- found — общее количество найденных результатов;
- request — текущий активный запрос;
- correction — исправленный запрос;
- noSuggestPanel — флаг, скрывать ли панель поисковых подсказок.

## Описание событий

### clear

Событие очистки результатов поиска. Экземпляр класса **Event**.

### error

Событие ошибки получения результатов поиска с сервера. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- error — информация об ошибке

### load

Событие получения результатов поиска с сервера. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- skip — сколько элементов было пропущено;
- count — количество загруженных элементов.

### resultselect

Событие выбора результата поиска. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- index — индекс выбранного результата.

### resultshow

Событие показа результата поиска. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- index — индекс выбранного результата.

### submit

Событие отправления поискового запроса на сервер Экземпляр класса **Event**.

## Описание методов

### clear

```
clear()
```

Очищает результаты поиска, а также содержимое поисковой строки элемента управления.

### getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

### getRequestString

```
{String} getRequestString()
```

**Возвращает** поисковый запрос.

### getResponseMetaData

```
{Object} getResponseMetaData()
```

возвращает метаданные геопоиска.

### getResult

```
{vow.Promise} getResult(index)
```

Получение результатов поиска.

**Возвращает** объект типа **vow.Promise**.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс результата (начиная с 0). |

* Обязательный параметр/опция.

### getResultsArray

```
{Object[]} getResultsArray()
```

**Возвращает** массив, содержащий все текущие результаты.

### getResultsCount

```
{Integer} getResultsCount()
```

**Возвращает** количество найденых результатов.

### getSelectedIndex

```
{Integer} getSelectedIndex()
```

**Возвращает** индекс выбранного элемента.

## hideResult

```
hideResult()
```

Скрывает результат, показанный на карте.

**Пример:**

```
// Если мы показали какой-то результат на карте с помощью
control.SearchControl.showResult,
// или он был показан автоматически при поиске, мы можем скрыть его, например,
// по клику на кнопку.
var myButton = new ymaps.control.Button("Скрыть результаты");
myButton.events.add('click', function () {
    searchControl.hideResult();
}, this);
myMap.controls.add(myButton, { selectOnClick: false });
```

## search

```
{vow.Promise} search(request)
```

Осуществляет поиск.

**Возвращает** объект типа **vow.Promise**.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| request* | — | Тип: String<br><br>Запрос. |

* Обязательный параметр/опция.

**Пример:**

```
// Попробуем найти Москву и вывести поле name
// первого найденного результата в консоль.
searchControl.search('Москва').then(function () {
    var geoObjectsArray = searchControl.getResultsArray();
    if (geoObjectsArray.length) {
        // Выводит свойство name первого геообъекта из результатов запроса.
        console.log(geoObjectsArray[0].properties.get('name'));
    }
});
```

## showResult

```
{control.SearchControl} showResult(index)
```

Отображает результат с заданным индексом.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс результата, начинается с 0. |

* Обязательный параметр/опция.

**Пример:**

```
// Мы хотим показывать всегда первый результат,
// вне зависимости от количества найденных
// объектов на карте (1 или более).
var searchControl = new ymaps.control.SearchControl({
    // Опция отключает автоматический выбор результата поиска.
    options: { noSelect: true }
});
searchControl.events.add('load', function (event) {
    // Проверяем, что это событие не "дозагрузки" результатов и
    // по запросу найден хотя бы один результат.
    if (!event.get('skip') && searchControl.getResultsCount()) {
        searchControl.showResult(0);
    }
});
```

# control.storage

Статический объект.

Экземпляр класса **util.Storage**

Хранилище элементов управления карты. Содержит соответствие ключей элементов управления и их конструкторов.

По умолчанию в хранилище добавлены следующие элементы управления:

- "rulerControl" — линейка и масштабный отрезок **control.RulerControl**;
- "searchControl" — панель поиска **control.SearchControl**;
- "trafficControl" — панель пробок **control.TrafficControl**;
- "typeSelector" — панель переключения типа карты **control.TypeSelector**;
- "zoomControl" — ползунок масштаба **control.ZoomControl**;
- "geolocationControl" — элемент управления геолокацией **control.GeolocationControl**;
- "routeEditor" — редактор маршрутов **control.RouteEditor**.
- "fullscreenControl" — элемент управления «полноэкранным режимом» **control.FullscreenControl**;

Методы

```
control.storage()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |

# control.TrafficControl

Расширяет **IControl**, **ICustomizable**.

Панель управления пробками на карте.

Конструктор | Поля | События | Методы

## Конструктор

```
control.TrafficControl([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: Object<br><br>Параметры элемента управления. |
| parameters.options | — | Тип: Object<br><br>Опции элемента управления. |
| parameters.options.adjustMapMargin | false | Тип: Boolean<br><br>Регистрирует ли элемент управления свои размеры в менеджере отступов карты **map.margin.Manager**. |
| parameters.options.collapseOnBlur | true | Тип: Boolean<br><br>Флаг, разрешающий сворачивание панели при потере фокуса на элементе управления. Например при клике на документ. |
| parameters.options.float | "right" | Тип: String<br><br>Сторона, по которой нужно выравнивать элемент управления. Может принимать три значения: "left", "right" или "none". При значении "left" или "right" элементы управления выстраиваются друг за другом, начиная от левого или правого края карты соответственно. При значении "none" элементы управления позиционируется только по значениям опций left, right, bottom, top относительно границ карты. Также смотрите описание опции position. |
| parameters.options.floatIndex | 100 | Тип: Number<br><br>Приоритет расположения элемента управления. Элемент с максимальным приоритетом находится ближе к указанному в свойстве float краю карты. Не работает при float = "none". |
| parameters.options.layout | — | Тип: Function \| String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Макет элемента управления. В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.TrafficControl.options;<br>• data — менеджер данных элемента управления **control.TrafficControl.data**;<br>• state — менеджер состояния элемента управления **control.TrafficControl.state**.<br>Макет меняет свой внешний вид на основе данных, состояния и опций элемента управления. Элемент управления, в свою очередь, реагирует на интерфейсные события макета и меняет значения полей **control.TrafficControl.state** в зависимости от полученных команд. (Тип: конструктор объекта с интерфейсом **ITrafficControlLayout** или ключ макета). |
| parameters.options.maxWidth | [26, 195, 195] | Тип: Number \| Number[]<br>Максимальная ширина кнопки в различных состояниях. Если задано число, то считается, что элемент управления имеет одинаковые максимальные размеры во всех состояниях. Если задан массив, то он будет трактоваться как максимальная ширина в различных состояниях — от меньшего к большему. Количество состояний задается в экземпляре класса **control.Manager**, который обычно является полем **Map.controls**, через опцию states. По умолчанию у элементов управления есть три состояния — ['small', 'medium', 'large']. |
| parameters.options.position | — | Тип: Object<br>Объект, описывающий позицию элемента управления. При указании опции position значение опции float автоматически трактуется как "none". |
| parameters.options.position.bottom | 'auto' | Тип: Number \| String<br>Положение относительно нижнего края карты. |
| parameters.options.position.left | 'auto' | Тип: Number \| String<br>Положение относительно левого края карты. |
| parameters.options.position.right | 'auto' | Тип: Number \| String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Положение относительно правого края карты. |
| parameters.options.position.top | 'auto' | Тип: Number \| String<br><br>Положение относительно верхнего края карты. |
| parameters.options.size | 'auto' | Тип: String<br><br>Параметр, отвечающий за внешний вид стандартного макета элемента управления пробками. Может принимать значения:<br>• 'auto' — макет автоматически изменяет свой внешний вид в зависимости от размеров карты и от количества добавленных элементов управления;<br>• 'small' — в макете кнопки отображается только иконка светофора, независимо от размера карты;<br>• 'large' — в макете кнопки всегда отображается иконка сфетофора и текст, независимо от размера карты. |
| parameters.options.visible | true | Тип: Boolean<br><br>Признак того, что элемент управления отображается. |
| parameters.state | — | Тип: Object<br><br>Состояние элемента управления. |
| parameters.state.providerKey | 'traffic#actual' | Тип: String<br><br>Ключ провайдера пробок, показываемых на карте.<br>• 'traffic#actual' — пробки "сейчас"<br>• 'traffic#archive' — пробки "обычно" |
| parameters.state.trafficShown | false | Тип: Boolean<br><br>Показаны ли данные по пробкам на карте. |

**Пример:**

```
// Добавление на карту элемента управления пробками
// с включенными пробками "сейчас".
var trafficControl = new ymaps.control.TrafficControl({state: {trafficShown:
true}});
map.controls.add(trafficControl, {top: 10, left: 10});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| data | data.Manager | Данные панели. |
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от IControl. |
| state | data.Manager | Состояние панели. Имена полей, доступных через метод data.Manager.get:<br>• trafficShown — признак, показан ли на карте провайдер пробок;<br>• providerKey — ключ провайдера, который показывает панель. Может принимать значения 'traffic#actual' или 'traffic#archive';<br>• expanded — признак, развернута ли панель. |

## События

| Имя | Описание |
|---|---|
| collapse | Панель пробок свернута. Экземпляр класса Event. |
| expand | Панель пробок развернута. Экземпляр класса Event. |
| hidetraffic | Пробки скрыты. Экземпляр класса Event. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| providerkeychange | Сменился ключ провайдера. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• newProviderKey — новое значение ключа провайдера;<br>• oldProviderKey — старое значение ключа. |
| showtraffic | Показаны пробки. Экземпляр класса Event. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| collapse() | | Свернуть панель пробок. |
| expand() | | Развернуть панель пробок. |
| getMap() | Map | Возвращает ссылку на карту. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IControl**. |
| **getProvider**([key]) | **ITrafficProvider** | Возвращает экземпляр провайдера пробок. |
| **hideTraffic**() | | Скрыть провайдер пробок с карты. |
| **isExpanded**() | Boolean | Возвращает признак, развернута ли панель. |
| **isTrafficShown**() | Boolean | Возвращает признак, показан ли провайдер пробок на карте. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IControl**. |
| **showTraffic**() | | Показать провайдер пробок на карте. |

## Описание полей

### data

```
{data.Manager} data
```

Данные панели.

### state

```
{data.Manager} state
```

Состояние панели. Имена полей, доступных через метод data.Manager.get:

- trafficShown — признак, показан ли на карте провайдер пробок;
- providerKey — ключ провайдера, который показывает панель. Может принимать значения 'traffic#actual' или 'traffic#archive';
- expanded — признак, развернута ли панель.

## Описание событий

### collapse

Панель пробок свернута. Экземпляр класса **Event**.

### expand

Панель пробок развернута. Экземпляр класса **Event**.

### hidetraffic

Пробки скрыты. Экземпляр класса **Event**.

### providerkeychange

Сменился ключ провайдера. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- newProviderKey — новое значение ключа провайдера;
- oldProviderKey — старое значение ключа.

### showtraffic

Показаны пробки. Экземпляр класса **Event**.

## Описание методов

### collapse

```
collapse()
```

Свернуть панель пробок.

### expand

```
expand()
```

Развернуть панель пробок.

### getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

### getProvider

```
{ITrafficProvider} getProvider([key])
```

**Возвращает** экземпляр провайдера пробок.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key | — | Тип: String<br><br>Ключ провайдера информации о пробках. Список доступных ключей:<br>• 'traffic#actual' — провайдер пробок "сейчас";<br>• 'traffic#archive' — провайдер пробок "обычно".<br><br>Если параметр не передан, возвращается текущий провайдер. |

**Пример:**

```
// Добавим на карту контрол пробок.
map.controls.add('trafficControl');
// Провайдер пробок "сейчас" при открытии будет показывать слой дорожных
событий.
map.controls.get('trafficControl').getProvider('traffic#actual').state.set('info
LayerShown', true);
```

### hideTraffic

```
hideTraffic()
```

Скрыть провайдер пробок с карты.

### isExpanded

```
{Boolean} isExpanded()
```

**Возвращает** признак, развернута ли панель.

### isTrafficShown

```
{Boolean} isTrafficShown()
```

**Возвращает** признак, показан ли провайдер пробок на карте.

### showTraffic

```
showTraffic()
```

Показать провайдер пробок на карте.

## control.TypeSelector

Расширяет **control.ListBox**.

Элемент управления "Типы карты". В данный элемент управления можно добавлять элементы списка, которые описывают типы карты, а также дополнительные элементы. Ключ элемента управления в хранилище **control.storage** — "typeSelector".

Конструктор | Поля | События | Методы

### Конструктор

```
control.TypeSelector([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: String[] \| **MapType**[] \| Object<br><br>Объект с описаниями параметров элемента управления. Если передан массив, то он трактуется как массив типов карты. |
| parameters.mapTypes | — | Тип: String[] \| **MapType**[] |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Массив конструкторов типов карты или ключей. Если параметр отсутствует, в элемент будет добавлен стандартный набор типов карты. Список доступных типов карты:<br>• 'yandex#map' — тип карты "схема";<br>• 'yandex#satellite' — тип карты "спутник";<br>• 'yandex#hybrid' — тип карты "гибрид". |
| parameters.options | — | Тип: Object<br><br>Опции элемента управления. |
| parameters.options.adjustMapMargin | false | Тип: Boolean<br><br>Регистрирует ли элемент управления свои размеры в менеджере отступов карты **map.margin.Manager**. |
| parameters.options.collapseOnBlur | true | Тип: Boolean<br><br>Флаг, разрешающий сворачивание списка при потере фокуса на кнопке. Например при клике на документ. |
| parameters.options.collapseTimeout | 3000 | Тип: Number<br><br>Временной интервал, по истечении которого раскрытый список автоматически закрывается. |
| parameters.options.expandOnClick | true | Тип: Boolean<br><br>Флаг, разрешающий автоматическое разворачивание/сворачивание списка по клику. |
| parameters.options.float | "right" | Тип: String<br><br>Сторона, по которой нужно выравнивать элемент управления. Может принимать три значения: "left", "right" или "none". При значении "left" или "right" элементы управления выстраиваются друг за другом, начиная от левого или правого края карты соответственно. При значении "none" элементы управления позиционируется только по значениям опций left, right, bottom, top относительно границ карты. Также смотрите описание опции position. |
| parameters.options.floatIndex | 200 | Тип: Number<br><br>Приоритет расположения элемента управления. Элемент с максимальным приоритетом находится ближе к указанному в свойстве float краю карты. Не работает при float = "none". |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters.options.layout | — | Тип: Function \| String<br><br>Конструктор макета элемента управления, реализующий интерфейсы **ISelectableControlLayout** и **IGroupControlLayout** или ключ макета в хранилище **layout.storage**. В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.ListBox.options;<br>• data — менеджер данных элемента управления **control.ListBox.data**;<br>• state — менеджер состояния элемента управления **control.ListBox.state**.<br><br>Макет меняет свой внешний вид на основе данных, состояния и опций элемента управления. Элемент управления, в свою очередь, реагирует на интерфейсные события макета и меняет значения полей **control.ListBox.state** в зависимости от полученных команд. |
| parameters.options.maxWidth | [30, 65, 85] | Тип: Number \| Number[]<br><br>Максимальная ширина кнопки списка при различных состояниях. Если задано число, то считается, что кнопка имеет одинаковые максимальные размеры во всех состояниях. Если задан массив, то он будет трактоваться как максимальная ширина кнопки в различных состояниях — от меньшего к большему. |
| parameters.options.position | — | Тип: Object<br><br>Объект, описывающий позицию элемента управления. При указании опции position значение опции float автоматически трактуется как "none". |
| parameters.options.position.bottom | 'auto' | Тип: Number \| String<br><br>Положение относительно нижнего края карты. |
| parameters.options.position.left | 'auto' | Тип: Number \| String<br><br>Положение относительно левого края карты. |
| parameters.options.position.right | 'auto' | Тип: Number \| String<br><br>Положение относительно правого края карты. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters.options.position.top | 'auto' | Тип: Number \| String<br><br>Положение относительно верхнего края карты. |
| parameters.options.visible | true | Тип: Boolean<br><br>Признак того, что элемент управления отображается. |
| parameters.state | — | Тип: Object<br><br>Состояние элемента управления. |
| parameters.state.expanded | false | Тип: Boolean<br><br>Признак, развернут ли список. |

**Пример:**

```
map.controls.add(new ymaps.control.TypeSelector(['yandex#map',
'yandex#hybrid']));
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| data | data.Manager | Данные элемента управления. |
| events | IEventManager | Менеджер событий.<br><br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br><br>Унаследовано от IControl. |
| state | data.Manager | Состояние выпадающего списка. Имена полей, доступных через метод data.Manager.get:<br><br>• expanded — признак, раскрыт ли список;<br>• size — размер, который в данный момент выставлен списку;<br><br>Унаследовано от control.ListBox. |

## События

| Имя | Описание |
|---|---|
| add | Произошло добавление дочернего объекта.<br><br>Унаследовано от ICollection. |
| click | Щелчок по заголовку списка. Экземпляр класса Event.<br><br>Унаследовано от control.ListBox. |
| collapse | Список закрыт. Экземпляр класса Event.<br><br>Унаследовано от control.ListBox. |
| expand | Список открыт. Экземпляр класса Event.<br><br>Унаследовано от control.ListBox. |
| optionschange | Изменение в опциях объекта. |

| Имя | Описание |
|---|---|
| | Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **press** | Событие, сигнализирующее о том, что кнопка была нажата. В отличии от click генерируется только если состояние isEnabled == true. Экземпляр класса **Event**.<br>Унаследовано от **control.ListBox**. |
| **remove** | Произошло удаление дочернего объекта.<br>Унаследовано от **ICollection**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(object) | **ICollection** | Добавляет в коллекцию дочерний объект.<br>Унаследован от **ICollection**. |
| **addMapType**(mapType[, positionIndex]) | **control.TypeSelector** | Добавляет тип карты в список. |
| **collapse**() | **control.ListBox** | Сворачивает список.<br>Унаследован от **control.ListBox**. |
| **expand**() | **control.ListBox** | Раскрывает список.<br>Унаследован от **control.ListBox**. |
| **getIterator**() | **IIterator** | Возвращает итератор по коллекции.<br>Унаследован от **ICollection**. |
| **getMap**() | **Map** | Возвращает ссылку на карту. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IControl**. |
| **isExpanded**() | Boolean | Возвращает признак того, находится ли контрол в развернутом состоянии.<br>Унаследован от **control.ListBox**. |
| **remove**(object) | **ICollection** | Удаляет из коллекции дочерний объект.<br>Унаследован от **ICollection**. |
| **removeAllMapTypes**() | **control.TypeSelector** | Удаляет все типы карты из элемента управления. |
| **removeMapType**(mapType) | **control.TypeSelector** | Удаляет тип карты. |

| Имя | Возвращает | Описание |
|---|---|---|
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br><br>Унаследован от **IControl**. |

## Описание полей

### data

```
{data.Manager} data
```

Данные элемента управления.

## Описание методов

### addMapType

```
{control.TypeSelector} addMapType(mapType[, positionIndex])
```

Добавляет тип карты в список.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| mapType* | — | Тип: String \| **MapType**<br><br>Тип карты. |
| positionIndex | — | Тип: Integer<br><br>Позиция в списке (если не указана, новый тип карты добавляется в конец списка). Список значений positionIndex по умолчанию для стандартных типов карты:<br>• 'yandex#map' — 5;<br>• 'yandex#satellite' — 10;<br>• 'yandex#hybrid' — 15. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
var typeSelector = new ymaps.control.TypeSelector([]);
typeSelector.addMapType('yandex#map', 1);
typeSelector.addMapType('yandex#hybrid', 0);
```

**2.**

```
// Если используется стандартный набор типов карты, и мы
// хотим добавить свой из хранилища mapType.storage между
// «спутник» и «схема».
var typeSelector = myMap.controls.get('typeSelector');
typeSelector.addMapType('my#mapType', 6);
```

### getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

### removeAllMapTypes

```
{control.TypeSelector} removeAllMapTypes()
```

Удаляет все типы карты из элемента управления.

**Возвращает** ссылку на себя.

### removeMapType

```
{control.TypeSelector} removeMapType(mapType)
```

Удаляет тип карты.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| mapType* | — | Тип: String \| **MapType**<br><br>Тип карты. |

* Обязательный параметр/опция.

## control.ZoomControl

Расширяет **IControl**, **ICustomizable**.

Элемент управления масштабом карты. Ключ элемента управления в хранилище **control.storage** — "zoomControl".

Конструктор | Поля | События | Методы

### Конструктор

```
control.ZoomControl([parameters])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters | — | Тип: Object<br><br>Параметры элемента управления. |
| parameters.data | — | Тип: Object<br><br>Данные элемента управления. |
| parameters.options | — | Тип: Object<br><br>Опции элемента управления. |
| parameters.options.adjustMapMargin | false | Тип: Boolean |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Регистрирует ли элемент управления свои размеры в менеджере отступов карты **map.margin.Manager**. |
| parameters.options.layout | — | Тип: **IZoomControlLayout** \| String<br><br>Конструктор макета элемента управления или его ключ в хранилище **layout.storage**. В конструктор макета передается объект, содержащий поля:<br>• control — ссылка на элемент управления;<br>• options — менеджер опций элемента управления control.ZoomControl.options;<br>• data — менеджер данных элемента управления control.ZoomControl.data;<br>• state — менеджер состояния элемента управления control.ZoomControl.state. |
| parameters.options.position | — | Тип: Object<br><br>Объект, описывающий позицию элемента управления. |
| parameters.options.position.bottom | 'auto' | Тип: Number \| String<br><br>Положение относительно нижнего края карты. |
| parameters.options.position.left | 10 | Тип: Number \| String<br><br>Положение относительно левого края карты. |
| parameters.options.position.right | 'auto' | Тип: Number \| String<br><br>Положение относительно правого края карты. |
| parameters.options.position.top | 108 | Тип: Number \| String<br><br>Положение относительно верхнего края карты. |
| parameters.options.size | 'auto' | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Параметр, отвечающий за внешний вид элемента управления. Может принимать значения:<br>• 'small' — всегда показывать маленький элемент управления масштабом карты;<br>• 'large' — всегда показывать большой элемент управления масштабом карты;<br>• 'auto' — производить автоматический выбор размера элемента управления в зависимости от высоты контейнера карты. |
| parameters.options.visible | true | Тип: Boolean<br><br>Признак того, что элемент управления отображается. |
| parameters.options.zoomDuration | 200 | Тип: Number<br><br>Время анимации при смене уровня масштабирования. |
| parameters.options.zoomStep | 1 | Тип: Number<br><br>Шаг изменения уровня масштабирования карты. |
| parameters.state | — | Тип: Object<br><br>Объект, описывающий состояние элемента управления. |

**Примеры:**

**1.**

```
// Пример 1.
// Создание маленького элемента управления масштабом карты и добавление его на
карту.
var zoomControl = new ymaps.control.ZoomControl({
    options: {
        size: "small"
    }
});
myMap.controls.add(zoomControl);
```

**2.**

```
// Пример 2.
// Добавление большого элемента управления масштабом карты по ключу из
control.storage.
myMap.controls.add('zoomControl', {
    size: "large"
});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br><br>Унаследовано от IEventEmitter. |

| Имя | Тип | Описание |
|---|---|---|
| options | IOptionManager | Менеджер опций.<br>Унаследовано от IControl. |

## События

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IControl. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IControl. |

# Описание методов

## getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

# coordSystem

## coordSystem.cartesian

Статический объект.

Экземпляр класса **ICoordSystem**

Класс, описывающий геометрию декартовой плоскости. Используется как координатная система по умолчанию при построении геодезических линий в нестандартных проекциях.

Методы

```
coordSystem.cartesian([scale])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| scale | 1 | Тип: Number \| Number[]<br><br>Масштаб одного деления на оси. Может быть числом или парой чисел для каждой из осей. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getDistance**(point1, point2) | Number | Возвращает кратчайшее (вдоль геодезической линии) расстояние между двумя заданными точками (в метрах). |
| **solveDirectProblem**(startPoint, direction, distance) | Object | Решает так называемую первую (прямую) геодезическую задачу: где мы окажемся, если выйдем из указанной точки в указанном направлении и пройдём, не сворачивая, указанное расстояние. |
| **solveInverseProblem**(startPoint, endPoint[, reverseDirection]) | Object | Решает так называемую вторую (обратную) геодезическую задачу: построить кратчайший маршрут между двумя точками на картографируемой поверхности и определелить расстояние и направление движения. |

## coordSystem.geo

Статический объект.

Экземпляр класса **ICoordSystem**

Объект, описывающий геометрию земной поверхности. Позволяет строить кратчайшие маршруты (геодезические линии) между точками земной поверхности и находить расстояния.

Методы

```
coordSystem.geo()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getDistance**(point1, point2) | Number | Возвращает кратчайшее (вдоль геодезической линии) расстояние между двумя заданными точками (в метрах). |
| **solveDirectProblem**(startPoint, direction, distance) | Object | Решает так называемую первую (прямую) геодезическую задачу: где мы окажемся, если выйдем из указанной точки в указанном направлении и пройдём, не сворачивая, указанное расстояние. |
| **solveInverseProblem**(startPoint, endPoint[, reverseDirection]) | Object | Решает так называемую вторую (обратную) геодезическую задачу: построить кратчайший маршрут между двумя точками на картографируемой поверхности и определелить расстояние и направление движения. |

# data

## data.Manager

Расширяет **IDataManager**, **IFreezable**.

Менеджер произвольных данных.

Конструктор | Поля | События | Методы

### Конструктор

```
data.Manager(data)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data * | — | Тип: Object<br><br>Данные. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий объекта.<br>Унаследовано от **IFreezable**. |

### События

| Имя | Описание |
|---|---|
| **change** | Изменение внутреннего состояния объекта.<br>Унаследовано от **IFreezable**. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **get**(path[, defaultValue]) | Object | Возвращает значение поля данных с заданным именем. |
| **getAll**() | Object | Возвращает объект, содержащий все поля данных. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **set**(path[, value]) | **data.Manager** | Устанавливает значения для указанных полей. Поддерживаются две сигнатуры:<br>• один аргумент, представляющий собой объект вида {"имя": "значение"};<br>• два аргумента, первый из которых — имя поля, второй — значение.<br>Имя может адресовать вложенные поля, т.е. содержать '.'. |
| **setAll**() | **data.Manager** | Полностью перезаписывает все поля данных. Равносильно последовательному вызову методов unsetAll и set, но обеспечивает большую производительность. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |
| **unset**(path) | **data.Manager** | Очищает заданные поля данных. |
| **unsetAll**() | **data.Manager** | Очищает все поля данных. |

## Описание методов

### get

```
{Object} get(path[, defaultValue])
```

**Возвращает** значение поля данных с заданным именем.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| path* | — | Тип: String<br>Строка с именем поля данных. Имя может адресовать вложенные поля, т.е. содержать '.'. |
| defaultValue | — | Тип: Object<br>Значение по умолчанию. |

* Обязательный параметр/опция.

### getAll

```
{Object} getAll()
```

**Возвращает** объект, содержащий все поля данных.

### set

```
{data.Manager} set(path[, value])
```

Устанавливает значения для указанных полей. Поддерживаются две сигнатуры:

- один аргумент, представляющий собой объект вида {"имя": "значение"};
- два аргумента, первый из которых — имя поля, второй — значение.

Имя может адресовать вложенные поля, т.е. содержать '.'.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| path* | — | Тип: Object \| String<br><br>Строка, содержащая имя поля данных, либо объект вида {"имя поля": "значение"}. |
| value | — | Тип: Object<br><br>Значение, если в качестве первого аргумента передается строка, содержащая имя поля. |

* Обязательный параметр/опция.

**Пример:**

```
var balloonData = new ymaps.data.Manager({
     balloonContentHeader: 'Заголовок балуна'
});
balloonData.set({
    balloonContentBody: 'Содержимое балуна',
    balloonContentFooter: '<a href="//ya.ru">test</a>'
});
```

## setAll

```
{data.Manager} setAll()
```

Полностью перезаписывает все поля данных. Равносильно последовательному вызову методов unsetAll и set, но обеспечивает большую производительность.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data* | — | Тип: |

* Обязательный параметр/опция.

## unset

```
{data.Manager} unset(path)
```

Очищает заданные поля данных.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| path* | — | Тип: String \| String[]<br><br>Имя поля, либо массив имён. |

* Обязательный параметр/опция.

## unsetAll

```
{data.Manager} unsetAll()
```

Очищает все поля данных.

**Возвращает** ссылку на себя.

# DomEvent

Расширяет **IDomEvent**.

DOM-событие в системе API Яндекс.Карт. Предоставляет прокси-методы для доступа к полям и методам исходного DOM-события. Возвращаемые значения нормализованы с учётом особенностей различных браузеров. Также переопределено свойство position, возвращается массив вида [pageX, pageY].

Конструктор | Методы

## Конструктор

```
DomEvent(originalEvent[, type])
```

Создаёт DOM-событие в системе API Яндекс.Карт.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| originalEvent * | — | Тип: **DomEvent**<br><br>DOM-событие. |
| type | — | Тип: Object<br><br>Тип события. Если не указан, берётся из originalEvent.type. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **allowMapEvent**() | | Разрешает распространение данного события на карту.<br><br>Унаследован от **IEvent**. |
| **callMethod**(name) | | Вызывает указанный метод у исходного события. Второй и последующие аргументы передаются методу при вызове.<br><br>Унаследован от **IEvent**. |
| **get**(name) | Object | Возвращает значение некоторого свойства. Сначала проверяется, не было ли свойство задано через set, затем проверяется наличие свойства в domEvent.overrideStorage, если не найдено — то в originalEvent. Значения свойств кэшируются. |
| **getSourceEvent**() | **IDomEvent** | Возвращает исходное DOM-событие.<br><br>Унаследован от **IDomEvent**. |
| **isDefaultPrevented**() | Boolean | Возвращает true, если реакция по умолчанию на событие отменена.<br><br>Унаследован от **IEvent**. |
| **isImmediatePropagationStopped**() | Boolean | Возвращает true, если обработка события прервана. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IEvent**. |
| **isMapEventAllowed**() | Boolean | Возвращает true, если событие карты разрешено.<br><br>Унаследован от **IEvent**. |
| **isPropagationStopped**() | Boolean | Возвращает true, если распространение события прервано,.<br><br>Унаследован от **IEvent**. |
| **preventDefault**() | | Отменяет реакцию по умолчанию на событие в рамках системы событий API Яндекс.Карт. Вызов этого метода не влияет на обработку браузером действия по умолчанию исходного DOM-события.<br><br>Унаследован от **IDomEvent**. |
| **stopImmediatePropagation**() | | Прекращает обработку события в системе событий API Яндекс.Карт. Т.е. после вызова данного метода ни один обработчик данного события не будет вызван. Вызов этого метода не влияет на обработку исходного DOM-события на уровне браузера.<br><br>Унаследован от **IDomEvent**. |
| **stopPropagation**() | | Прекращает распространение DOM-события в системе событий API Яндекс.Карт. Вызов этого метода не влияет на распространение исходного DOM-события по DOM-дереву.<br><br>Унаследован от **IDomEvent**. |

## Описание методов

### get

```
{Object} get(name)
```

Возвращает значение некоторого свойства. Сначала проверяется, не было ли свойство задано через set, затем проверяется наличие свойства в domEvent.overrideStorage, если не найдено — то в originalEvent. Значения свойств кэшируются.

**Возвращает** значение.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя свойства. |

* Обязательный параметр/опция.

# domEvent

## domEvent.manager

Статический объект.

Предоставляет единый интерфейс для работы с событиями DOM-элементов во всех браузерах и на всех устройствах. Для устройств, которые не поддерживают мышь, события будут транслироваться.

- Событие начала касания (touchstart/pointerdown) с одной точкой прикосновения транслируется в последовательность событий mouseenter, mousemove и mousedown;
- Событие перемещения касания (touchmove/pointermove) с одной точкой прикосновения транслируется в событие mousemove;
- События завершения касания (touchend/pointerup) или отмены (touchcancel/pointercancel) транслируются в последовательность событий mouseup, mousemove и mouseleave, если ранее произошло событие начала касания с одной точкой прикосновения;
- Быстрое поступление событий начала и завершения с одной точкой прикосновения без перемещения транслируется в событие click;
- Быстрое поступление двух событий click транслируется в событие dblclick;
- Если между событиями начала и завершения с одной точкой прикосновения без перемещения была продолжительная задержка, то это транслируется в событие contextmenu.

Также поддерживаются специальные события для обработки нескольких одновременных касаний.

- multitouchstart отправляется при поступлении события начала касания с двумя и более точками прикосновений;
- multitouchmove отправляется при поступлении события перемещения касания с двумя и более точками прикосновений;
- multitouchend отправляется при поступлении события завершения касания, если ранее было отправлено событие multitouchstart;
- При добавлении/удалении точки прикосновения будет отправлено событие multitouchend и событие multitouchstart, если оставшееся количество точек больше или равно двум.

Менеджер работы с событиями DOM-элементов.

Методы

```
domEvent.manager()
```

**Примеры:**

**1.**

```
// Прослушивание событий одного DOM-элемента.
var block = document.getElementById('block');
ymaps.domEvent.manager
    .add(block, 'click', function (event) {
        console.log(event.get('type')); // click
    })
    .add(block, 'mouseleave', function (event) {
        console.log(event.get('type')); // mouseleave
    })
    // Задание одного слушателя на несколько событий.
    .add(block, ['mousedown', 'mouseup'], function (event) {
        console.log(event.get('type')); // mousedown / mouseup
    });
```

**2.**

```
// Использование контейнера событий.
var block = document.getElementById('block'),
    domEventsGroup = ymaps.domEvent.manager.group(block);
domEventsGroup.add('click', function (event) {
    console.log(event.get('type')); // click
    // Удаляем все прослушивания событий.
    domEventsGroup.removeAll();
});
```

**3.**

```
// Исполнение слушателя в контексте определенного объекта.
var block = document.getElementById('block'),
    // Определяем класс.
    someClass = function () {
        this.property = 'value';
    },
    // Создаем реализацию класса.
    someObj = new someClass();
ymaps.domEvent.manager.add(block, 'click', function (event) {
    console.log(this.property + ' ' + event.get('type')); // value click
}, someObj);
```

**4.**

```
// На устройствах, которые поддерживают сенсорный ввод, можно прослушивать
специальные события multitouch*
var block = document.getElementById('block');
ymaps.domEvent.manager
    .add(block, ['multitouchstart', 'multitouchmove', 'multitouchend'], function
(event) {
        console.log(event.get('type')); // multitouchstart / multitouchmove /
multitouchend
        // Запрещаем сдвиг и масштабирование страницы пальцами.
        event.callMethod('preventDefault');
    });
```

## Методы

| Имя | Статический | Возвращает | Описание |
|---|---|---|---|
| domEvent.manager.**add**(htmlElement, types, callback[, context[, capture]]) | ✔ | **domEvent.manager** | Добавляет слушателя DOM-событий объекта. |
| domEvent.manager.**group**(htmlElement[, capture]) | ✔ | **event.Group** | Возвращает группу слушателей событий для заданного DOM-элемента. |
| domEvent.manager.**remove**(htmlElement, types, callback[, context[, capture]]) | ✔ | **domEvent.manager** | Удаляет слушателя DOM-событий объекта. |

## Описание методов

### add

```
{domEvent.manager} <static> domEvent.manager.add(htmlElement, types, callback[,
context[, capture]])
```

Добавляет слушателя DOM-событий объекта.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| htmlElement* | — | Тип: HTMLElement<br><br>DOM-элемент, события которого необходимо прослушивать. |
| types* | — | Тип: String \| String[]<br><br>Тип или типы событий. |
| callback* | — | Тип: Function<br><br>Функция-обработчик события. |
| context | — | Тип: Object<br><br>Контекст исполнения функции-обработчика. |
| capture | — | Тип: Boolean<br><br>Признак того, что событие необходимо отслеживать на фазе захвата. |

* Обязательный параметр/опция.

## group

```
{event.Group}  <static> domEvent.manager.group(htmlElement[, capture])
```

**Возвращает** группу слушателей событий для заданного DOM-элемента.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| htmlElement* | — | Тип: HTMLElement<br><br>DOM-элемент. |
| capture | — | Тип: Boolean<br><br>Признак того, что событие необходимо отслеживать на фазе захвата. |

* Обязательный параметр/опция.

## remove

```
{domEvent.manager}  <static>  domEvent.manager.remove(htmlElement,  types,
callback[, context[, capture]])
```

Удаляет слушателя DOM-событий объекта.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| htmlElement* | — | Тип: HTMLElement |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | DOM-элемент, события которого прослушиваются. |
| types* | — | Тип: String \| String[]<br><br>Тип или типы событий. |
| callback* | — | Тип: Function \| String<br><br>Функция-обработчик события либо уникальный id пары callback-context. |
| context | — | Тип: Object<br><br>Контекст исполнения функции-обработчика. |
| capture | — | Тип: Boolean<br><br>Признак того, что событие необходимо отслеживать на фазе захвата. |

* Обязательный параметр/опция.

# domEvent.MultiPointer

Расширяет IMultiTouchEvent.

Объект, описывающий multitouch-событие, которое было вызвано несколькими событиями PointerEvent. Предоставляет прокси-методы для доступа к полям и методам DOM-события.

Конструктор | Методы

## Конструктор

```
domEvent.MultiPointer(originalEvent[, type])
```

Создаёт объект, описывающий multitouch-событие.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| originalEvent * | — | Тип: Object<br><br>pointer-событие. |
| type | — | Тип: String<br><br>Тип события. Если не указан, считается равным 'multi' + originalEvent.type. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **get**(name) | Object | Возвращает значение свойства оригинального события. Значения свойств кэшируются. |

## Описание методов

### get

```
{Object} get(name)
```

Возвращает значение свойства оригинального события. Значения свойств кэшируются.

**Возвращает** значение.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя свойства. |

* Обязательный параметр/опция.

# domEvent.MultiTouch

Расширяет IMultiTouchEvent.

Объект-событие. Предоставляет прокси-методы для доступа к полям и методам DOM-события. Возвращаемые значения нормализованы с учётом особенностей различных браузеров.

Конструктор | Методы

## Конструктор

```
domEvent.MultiTouch(originalEvent[, type])
```

Создаёт объект-событие, описывающий multitouch-событие.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| originalEvent * | — | Тип: Object<br><br>multitouch-событие. |
| type | — | Тип: String<br><br>Тип события. Если не указан, считается равным 'multi' + originalEvent.type. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **get**(name) | Object | Возвращает значение свойства оригинального события. Значения свойств кэшируются. |

## Описание методов

### get

```
{Object} get(name)
```

Возвращает значение свойства оригинального события. Значения свойств кэшируются.

**Возвращает** значение.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя свойства. |

* Обязательный параметр/опция.

# domEvent.Pointer

Расширяет IMultiTouchEvent.

Объект-событие. Предоставляет прокси-методы для доступа к полям и методам DOM-события (одиночное касание экрана).

Конструктор | Методы

## Конструктор

```
domEvent.Pointer(originalEvent[, type])
```

Создаёт объект, описывающий pointer-событие.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| originalEvent * | — | Тип: Object<br><br>DOM-событие. |
| type | — | Тип: String<br><br>Тип события. Если не указан, берётся из originalEvent.type. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get(name) | Object | Возвращает значение свойства оригинального события. Значения свойств кэшируются. |

## Описание методов

### get

```
{Object} get(name)
```

Возвращает значение свойства оригинального события. Значения свойств кэшируются.

**Возвращает** значение.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя свойства. |

* Обязательный параметр/опция.

# domEvent.Touch

Расширяет IMultiTouchEvent.

Объект-событие. Предоставляет прокси-методы для доступа к полям и методам DOM-события. Возвращаемые значения нормализованы с учётом особенностей различных браузеров.

Конструктор | Методы

## Конструктор

```
domEvent.Touch(originalEvent[, type])
```

Создаёт объект-событие, описывающий touch-событие (одиночное касание экрана).

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| originalEvent * | — | Тип: Object<br><br>DOM-событие. |
| type | — | Тип: String<br><br>Тип события. Если не указан, берётся из originalEvent.type. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get(name) | Object | Возвращает значение свойства оригинального события. Значения свойств кэшируются. |

## Описание методов

### get

```
{Object} get(name)
```

Возвращает значение свойства оригинального события. Значения свойств кэшируются.

**Возвращает** значение.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя свойства. |

* Обязательный параметр/опция.

# event

## event.Group

Расширяет **IEventGroup**.

Группа слушателей событий.

Конструктор | Поля | Методы

### Конструктор

```
event.Group(events)
```

Создает группу слушателей событий.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| events * | — | Тип: **IEventManager**<br><br>Менеджер событий, для которого создается группа. |

* Обязательный параметр/опция.

**Пример:**

```
// Создадим группу слушателей.
var listeners = events.group()
   .add('click', function () {
       alert('click');
   })
   .add('dblclick', function () {
       alert('dblclick');
   });
// ...
// Когда хранящиеся в контейнере обработчики событий
// нам больше не нужны, просто очистим группу.
listeners.removeAll();
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий, для которого создана группа. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(types, callback[, context[, priority]]) | **IEventGroup** | Добавляет слушатель события.<br><br>Унаследован от **IEventGroup**. |
| **getLength**() | Integer | Возвращает текущее количество подпсчиков в группе. |
| **remove**(types, callback[, context[, priority]]) | **IEventGroup** | Удаляет слушатель события из группы.<br><br>Унаследован от **IEventGroup**. |

| Имя | Возвращает | Описание |
|---|---|---|
| removeAll() | IEventGroup | Удаляет всех слушателей событий из группы.<br>Унаследован от IEventGroup. |

## Описание полей

### events

```
{IEventManager} events
```

Менеджер событий, для которого создана группа.

## Описание методов

### getLength

```
{Integer} getLength()
```

**Возвращает** текущее количество подпсчиков в группе.

# event.Manager

Расширяет **IEventManager**.

Менеджер событий. При помощи менеджера событий можно подписываться на события и отписываться от них, а также инициировать сами события. Менеджер реализует возможность построения иерархии распространения событий с помощью метода event.Manager.setParent.

Распространение события происходит в три фазы:

- 1. Событие получают непосредственные подписчики;
- 2. Событие получают вышестоящие в иерархии объекты посредством перебрасывания события на родительском менеджере событий;
- 3. Событие получают обработчики действия по умолчанию посредством служебного события type + 'defaultaction'; действие по умолчанию выполняется, только если поле target события совпадает с контекстом менеджера событий.

Также менеджер позволяет добавлять обработчики событий с указанием приоритета. При бросании событий обработчики будут вызваны в порядке убывания приоритета.

Подписки с одинаковыми параметрами callback, context, но с разными приоритетами считаются разными. При удалении подписки необходимо указывать тот же приоритет, что и при добавлении.

Конструктор | Методы

## Конструктор

```
event.Manager([params])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| params | — | Тип: Object<br>Параметры менеджера событий. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| params.context | — | Тип: Object<br><br>Контекстный объект менеджера событий. |
| params.controllers | — | Тип: **IEventWorkflowController**[]<br><br>Контроллер или контроллеры менеджера событий. |
| params.parent | — | Тип: **IEventManager**<br><br>Родительский менеджер событий. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(types, callback[, context[, priority]]) | **IEventManager** | Добавляет новую подписку.<br><br>Унаследован от **IEventManager**. |
| **createEventObject**(type, event, target) | **Event** | Функция, которая создает объект-событие. Вызывается в методе fire, когда переданный объект не является экземпляром класса Event или его наследником. |
| **fire**(type[, event]) | **event.Manager** | Бросает событие. |
| **getParent**() | **IEventManager** \| null | Возвращает ссылку на родительский менеджер событий.<br><br>Унаследован от **IEventManager**. |
| **group**() | **IEventGroup** | Возвращает группу слушателей событий, ассоциированную с данным менеджером событий.<br><br>Унаследован от **IEventManager**. |
| **once**(types, callback[, context[, priority]]) | **IEventManager** | Добавляет слушателя, который вызовет функцию-обработчик только 1 раз. |
| **remove**(types, callback[, context[, priority]]) | **IEventManager** | Удаляет существующую подписку.<br><br>Унаследован от **IEventManager**. |
| **setParent**(parent) | **event.Manager** | Устанавливает родительский менеджер событий. |

## Описание методов

### createEventObject

```
{Event} createEventObject(type, event, target)
```

Функция, которая создает объект-событие. Вызывается в методе fire, когда переданный объект не является экземпляром класса Event или его наследником.

**Возвращает** объект-событие.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| type* | — | Тип: String<br><br>Тип события. |
| event* | — | Тип: Object<br><br>Объект, описывающий событие. |
| target* | — | Тип: Object<br><br>Объект, на котором произошло событие. |

* Обязательный параметр/опция.

## fire

```
{event.Manager} fire(type[, event])
```

Бросает событие.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| type* | — | Тип: String<br><br>Тип события. |
| event | — | Тип: Object \| **Event**<br><br>Событие. Если передан хэш с данными, то для него будет вызван метод createEventObject, и дальнейшие действия будут производиться с новым созданным событием. |

* Обязательный параметр/опция.

## once

```
{IEventManager} once(types, callback[, context[, priority]])
```

Добавляет слушателя, который вызовет функцию-обработчик только 1 раз.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| types* | — | Тип: String \| String[]<br><br>Тип события или массив типов. |
| callback* | — | Тип: Function<br><br>Функция-обработчик события. В качестве параметра в функцию передается объект события **IEvent**. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| context | — | Тип: Object<br><br>Контекст исполнения обработчика. |
| priority | 0 | Тип: Integer<br><br>Приоритет подписки. |

* Обязательный параметр/опция.

### setParent

```
{event.Manager} setParent(parent)
```

Устанавливает родительский менеджер событий.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parent* | — | Тип: **IEventManager** \| null<br><br>Родительский менеджер событий. |

* Обязательный параметр/опция.

## event.Mapper

Расширяет **IEventTrigger**.

Маппер событий. Позволяет управлять распространением событий по иерархии менеджеров событий.

Конструктор | Методы

### Конструктор

```
event.Mapper(targetEventManager, mappingTable)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| targetEventManager * | — | Тип: **IEventManager**<br><br>Менеджер событий, на который маппер распространяет инициируемые на нем события. |
| mappingTable * | — | Тип: Object |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Таблица правил маппинга. Представляет собой хэш, ключами которого являются типы событий, а значениями — соответствующие функции маппинга, либо Boolean значения. Функция маппинга для конкретного типа событий получает на вход экземпляр события, инициированного на маппере, а возвращать должна экземпляр события для распространения далее по иерархии, либо null, если распространение необходимо запретить. Boolean значения трактуются следующим образом:<br>• true — события данного типа распространяются по иерархии в неизменном виде;<br>• false — события данного типа по иерархии не распространяются.<br>Дополнительно в таблице доступен ключ "*" для правила обработки по умолчанию. |

* Обязательный параметр/опция.

**Пример:**

```
// Создаем и устанавливаем маппер событий, который преобразует событие "click"
корневой
// коллекции геообъектов в событие "geoobjectclick" самой карты.
var mapper = new ymaps.event.Mapper(myMap.events, {
    "*": false,
    "click": function (event) {
        return new ymaps.Event({
             type: "geoobjectclick",
             target: map,
             originalTarget: event.get("target")
        }, event);
    }
});

myMap.geoObjects.events.setParent(mapper);
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **fire**(type[, eventObject]) | **IEventTrigger** | Инициирует событие.<br>Унаследован от **IEventTrigger**. |

# Event

Расширяет **IEvent**.

Событие. Предоставляет методы для доступа к полям и методам объекта originalEvent с возможностью их переопределения.

Конструктор | Методы

## Конструктор

```
Event(originalEvent[, sourceEvent])
```

Создаёт событие.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| originalEvent * | — | Тип: Object<br><br>Исходные данные. |
| sourceEvent | — | Тип: **IEvent**<br><br>Исходное событие. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **allowMapEvent**() | | Разрешает распространение данного события на карту.<br><br>Унаследован от **IEvent**. |
| **callMethod**(name) | Object | Вызывает указанный метод. Операция эквивалентна поиску поля через get и вызову с передачей в качестве контекста originalEvent. Все аргументы после первого передаются в качестве параметров вызываемому методу. |
| **get**(name) | Object | Возвращает значение поля из originalEvent. originalEvent всегда имеет следующие поля:<br><br>• type — строковый тип события;<br>• target — ссылка на объект, породивший событие. |
| **getSourceEvent**() | **IEvent** \| null | Возвращает исходное событие.<br><br>Унаследован от **IEvent**. |
| **isDefaultPrevented**() | Boolean | Проверяет, отменена ли реакция по умолчанию на событие в рамках системы событий API Яндекс.Карт. |
| **isImmediatePropagationStopped**() | Boolean | Определяет, прекращено ли распространение события в системе событий API Яндекс.Карт. |

| Имя | Возвращает | Описание |
|---|---|---|
| **isMapEventAllowed**() | Boolean | Возвращает true, если событие карты разрешено.<br><br>Унаследован от **IEvent**. |
| **isPropagationStopped**() | Boolean | Проверяет, остановлено ли распространение события по иерархии объектов и коллекций в системе событий API Яндекс.Карт. |
| **preventDefault**() | | Отменяет реакцию по умолчанию на событие в рамках системы событий API Яндекс.Карт. Вызов этого метода никак не влияет на распространение исходного DOM-события (если оно есть) по DOM-дереву. |
| **stopImmediatePropagation**() | | Прекращает распространение события в системе событий API Яндекс.Карт. Вызов этого метода никак не влияет на распространение исходного DOM-события (если оно есть) по DOM-дереву. |
| **stopPropagation**() | | Прекращает распространение события по иерархии объектов и коллекций в системе событий API Яндекс.Карт. Вызов этого метода никак не влияет на распространение исходного DOM-события (если оно есть) по DOM-дереву. |

## Описание методов

### callMethod

```
{Object} callMethod(name)
```

Вызывает указанный метод. Операция эквивалентна поиску поля через get и вызову с передачей в качестве контекста originalEvent. Все аргументы после первого передаются в качестве параметров вызываемому методу.

**Возвращает** значение.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя метода. |

* Обязательный параметр/опция.

### get

```
{Object} get(name)
```

Возвращает значение поля из originalEvent. originalEvent всегда имеет следующие поля:

- type — строковый тип события;
- target — ссылка на объект, породивший событие.

**Возвращает** значение.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя свойства. |

* Обязательный параметр/опция.

**Пример:**

```
// Синхронизируем два объекта между собой.
object1.events.add(["add", "remove"], function (event) {
    object2[event.get("type")](event.get("child"));
});
```

## isDefaultPrevented

```
{Boolean} isDefaultPrevented()
```

Проверяет, отменена ли реакция по умолчанию на событие в рамках системы событий API Яндекс.Карт.

**Возвращает** true, если реакция на событие по умолчанию отменена, false — в противном случае.

## isImmediatePropagationStopped

```
{Boolean} isImmediatePropagationStopped()
```

Определяет, прекращено ли распространение события в системе событий API Яндекс.Карт.

**Возвращает** true — распространение прекращено, false — нет.

## isPropagationStopped

```
{Boolean} isPropagationStopped()
```

Проверяет, остановлено ли распространение события по иерархии объектов и коллекций в системе событий API Яндекс.Карт.

**Возвращает** true — распространение по иерархии отменено, false — нет.

## preventDefault

```
preventDefault()
```

Отменяет реакцию по умолчанию на событие в рамках системы событий API Яндекс.Карт. Вызов этого метода никак не влияет на распространение исходного DOM-события (если оно есть) по DOM-дереву.

## stopImmediatePropagation

```
stopImmediatePropagation()
```

Прекращает распространение события в системе событий API Яндекс.Карт. Вызов этого метода никак не влияет на распространение исходного DOM-события (если оно есть) по DOM-дереву.

## stopPropagation

```
stopPropagation()
```

Прекращает распространение события по иерархии объектов и коллекций в системе событий API Яндекс.Карт. Вызов этого метода никак не влияет на распространение исходного DOM-события (если оно есть) по DOM-дереву.

# formatter

Статический объект.

Статический класс, содержащий методы форматирования единиц измерения с учётом текущего языка.

Методы

```
formatter()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **distance**(value[, significantDigits]) | String | Возвращает отформатированное согласно локальным правилам и переведённое в нужную систему единиц измерения строковое представление расстояния. |
| **duration**(value[, significantDigits]) | String | Возвращает строковое представление интервала времени. |

## Описание методов

### distance

```
{String} distance(value[, significantDigits])
```

**Возвращает** отформатированное согласно локальным правилам и переведённое в нужную систему единиц измерения строковое представление расстояния.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| value* | — | Тип: Number<br><br>Длина в метрах. |
| significantDigits | 2 | Тип: Integer<br><br>Число значащих цифр в ответе. |

* Обязательный параметр/опция.

### duration

```
{String} duration(value[, significantDigits])
```

**Возвращает** строковое представление интервала времени.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| value* | — | Тип: Number<br><br>Интервал времени в секундах. |
| significantDigits | 2 | Тип: Integer<br><br>Число значащих цифр в ответе. |

* Обязательный параметр/опция.

## geocode

Статическая функция.

Обрабатывает запросы геокодирования. Результат запроса может быть представлен в формате JSON или в виде объекта **GeoObjectCollection**. Формат ответа геокодера описан в разделе Геокодирование.

---

**Примечание:**
Для платной версии API поиск доступен только по следующим странам: Россия, Украина, Беларусь, Казахстан, Грузия, Абхазия, Южная Осетия, Армения, Азербайджан, Молдова, Туркмения, Таджикистан, Узбекистан, Киргизия и Турция.

---

**См.: GeocodeResult**

**Возвращает:** Объект-promise. (тип **vow.Promise**).

```
geocode(request[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| request* | — | Тип: String \| Number[]<br><br>Адрес, для которого необходимо определить координаты (прямое геокодирование) или координаты, для которых необходимо определить адрес (обратное геокодирование). |
| options | — | Тип: Object<br><br>Опции. |
| options.boundedBy | — | Тип: Number[][]<br><br>Прямоугольная область на карте, в которой предположительно находится искомый объект. |
| options.json | false | Тип: Boolean<br><br>Если true, то в функцию-обработчик передается JSON. В противном случае в функцию обработчик передается объект, содержащий в поле geoObjects результаты геокодирования в виде коллекции **GeoObjectCollection**. При геокодировании посредством геокодера 'yandex#map' в коллекции будут находиться объекты **GeocodeResult**. |
| options.kind | 'house' | Тип: String<br><br>Вид топонима (только для обратного геокодирования). |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Список возможных значений:<br>• house — дом;<br>• street — улица;<br>• metro — станция метро;<br>• district — район города;<br>• locality — населенный пункт (город/поселок/деревня/село/...). |
| options.provider | 'yandex#map' | Тип: **IGeocodeProvider** \| String<br><br>Провайдер геокодирования. Можно воспользоваться одним из стандартных провайдеров:<br>• 'yandex#map' — поиск по карте. |
| options.results | 10 | Тип: Integer<br><br>Максимальное количество возвращаемых результатов. |
| options.searchCoordOrder | — | Тип: String<br><br>Определяет каким образом нужно интрепретировать координаты в запросе. |
| options.skip | 0 | Тип: Integer<br><br>Число результатов, которое необходимо пропустить. |
| options.strictBounds | false | Тип: Boolean<br><br>Искать только внутри области, заданной опцией boundedBy. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Осуществляет поиск объекта с именем "Москва".
// Полученный результат сразу отображается на карте.
var myGeocoder = ymaps.geocode("Москва");
myGeocoder.then(
    function (res) {
        map.geoObjects.add(res.geoObjects);
        // Выведем в консоль данные, полученные в результате геокодирования
объекта.
        console.log(res.geoObjects.get(0).properties.get('metaDataProperty'));
    },
    function (err) {
        // обработка ошибки
    }
);
```

**2.**

```
// реализует интерфейс IGeocodeProvider
var randomPointProvider = {
     geocode: function (request, options) {
         var deferred = ymaps.vow.defer(),
             geoObjects = new ymaps.GeoObjectCollection(),
             results = options.results || 10;

         for (var i = 0; i < results; i++) {
             geoObjects.add(new ymaps.GeoObject({
                 geometry: {
                     type: "Point",
                     coordinates: [(Math.random() - 0.5) * 180, (Math.random() -
0.5) * 360]
                 },
                 properties: {
                     name: request + ' ' + i,
                     description: request + '\'s description ' + i,
                     balloonContentBody: '<p>' + request + ' ' + i + '</p>'
                 }
             }));
         }

         var response = {
             geoObjects: geoObjects, // геообъекты поисковой выдачи
             // метаинформация ответа
             metaData: {
                 geocoder: {
                     request: request, // строка обработанного запроса
                     found: results, // количество найденных результатов
                     results: results, // количество возвращенных результатов
                     skip: options.skip || 0 // количество пропущенных
результатов
                 }
             }
         };

         // выполняем асинхронную обработку
         setTimeout(function () {
             deferred.resolve(response);
         }, 0);

         return deferred.promise();
     }
},
    myGeocoder = ymaps.geocode("Москва", { provider: randomPointProvider });

myGeocoder.then(
    function (res) {
        map.geoObjects.add(res.geoObjects);
    },
    function (err) {
        // обработка ошибки
    }
);
```

## GeocodeResult

Расширяет **IGeoObject**.

Результат геокодирования.

**См.: Placemark geocode geolocation control.SearchControl**

Конструктор | Поля | События | Методы

## Конструктор

```
GeocodeResult()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |
| geometry | IGeometry \| null | Геометрия геообъекта.<br>Унаследовано от IGeoObject. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |
| properties | IDataManager | Данные геообъекта.<br>Унаследовано от IGeoObject. |
| state | IDataManager | Состояние геообъекта.<br>Унаследовано от IGeoObject. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| geometrychange | Изменение геометрии геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие геометрии.<br>Унаследовано от IGeoObject. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

| Имя | Описание |
| --- | --- |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getAddressLine() | String | Возвращает строку с адресом объекта. |
| getAdministrativeAreas() | String[] | Возвращает административно-территориальные образования, в которые входит объект (федеральный округ, регион, район), не больше двух. |
| getCountry() | String \| null | Возвращает государство, которому принадлежит топоним (если применимо). |
| getCountryCode() | String \| null | Возвращает код государства, которому принадлежит топоним (если применимо), в виде двухбуквенного кода ISO 3166. |
| getLocalities() | String[] | Возвращает населённый пункт и, опционально, образование внутри населённого пункта, которым принадлежит топоним. |
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |

| Имя | Возвращает | Описание |
|---|---|---|
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| getPremise() | String \| null | Возвращает наименование здания, если оно есть (например, название терминала аэропорта). |
| getPremiseNumber() | String \| null | Возвращает номер здания (включая корпус, владение и прочие дополнительные признаки). |
| getThoroughfare() | String \| null | Возвращает путь сообщения (улица, шоссе, проезд и т.д.), которому принадлежит топоним (если применимо). |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание методов

### getAddressLine

```
{String} getAddressLine()
```

**Возвращает** строку с адресом объекта.

### getAdministrativeAreas

```
{String[]} getAdministrativeAreas()
```

**Возвращает** административно-территориальные образования, в которые входит объект (федеральный округ, регион, район), не больше двух.

### getCountry

```
{String | null} getCountry()
```

**Возвращает** государство, которому принадлежит топоним (если применимо).

### getCountryCode

```
{String | null} getCountryCode()
```

**Возвращает** код государства, которому принадлежит топоним (если применимо), в виде двухбуквенного кода ISO 3166.

### getLocalities

```
{String[]} getLocalities()
```

**Возвращает** населённый пункт и, опционально, образование внутри населённого пункта, которым принадлежит топоним.

### getPremise

```
{String | null} getPremise()
```

**Возвращает** наименование здания, если оно есть (например, название терминала аэропорта).

### getPremiseNumber

```
{String | null} getPremiseNumber()
```

**Возвращает** номер здания (включая корпус, владение и прочие дополнительные признаки).

### getThoroughfare

```
{String | null} getThoroughfare()
```

**Возвращает** путь сообщения (улица, шоссе, проезд и т.д.), которому принадлежит топоним (если применимо).

## geolocation

Статический объект.

Предоставляет информацию о местоположении пользователя.

Методы

```
geolocation()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get([options]) | vow.Promise | Пытается определить местоположение пользователя. Возвращает объект-promise, который будет либо подтверждён объектом с полем geoObjects, либо отклонён с сообщением об ошибке. Поле geoObjects является экземпляром GeoObjectCollection. В коллекцию будет добавлен объект, обозначающий текущее местоположение пользователя. |

### Описание методов

#### get

```
{vow.Promise} get([options])
```

Пытается определить местоположение пользователя. Возвращает объект-promise, который будет либо подтверждён объектом с полем geoObjects, либо отклонён с сообщением об ошибке. Поле geoObjects является экземпляром **GeoObjectCollection**. В коллекцию будет добавлен объект, обозначающий текущее местоположение пользователя.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: |
| options.autoReverseGeocode | true | Тип:<br><br>true — автоматически прогеокодировать положение пользователя, false — вернуть как есть. В случае автоматического геокодирования объект, обозначающий текущее местоположение пользователя, будет иметь структуру, аналогичную результату выполнения **geocode**. |
| options.mapStateAutoApply | false | Тип:<br><br>true — при добавлении на карту автоматически выставить центр и уровень масштабирования карты так, чтобы показать текущее местоположение пользователя, false — ничего не делать. |
| options.provider | 'auto' | Тип:<br><br>Провайдер геолокации. Допустимые значения: 'yandex' — геолокация по данным Яндекса на основе ip пользователя, 'browser' — встроенная браузерная геолокация, 'auto' — провести геолокацию всеми доступными способами и выбрать лучшее значение. |
| options.timeout | 30000 | Тип:<br><br>Время ожидания ответа в мс. |
| options.useMapMargin | true | Тип: Boolean<br><br>Нужно ли учитывать отступы карты **map.margin.Manager** при автоматическом выставлении центра и уровня масштаба карте. |

**Примеры:**

**1.**

```
ymaps.geolocation.get({
    // Выставляем опцию для определения положения по ip
    provider: 'yandex',
    // Карта автоматически отцентрируется по положению пользователя.
    mapStateAutoApply: true
}).then(function (result) {
    myMap.geoObjects.add(result.geoObjects);
});
```

**2.**

```
ymaps.geolocation.get({
    // Выставляем опцию для определения положения по ip
    provider: 'yandex',
    // Автоматически геокодируем полученный результат.
    autoReverseGeocode: true
}).then(function (result) {
    // Выведем в консоль данные, полученные в результате геокодирования объекта.
    console.log(result.geoObjects.get(0).properties.get('metaDataProperty'));
});
```

# geometry

## geometry.base

### geometry.base.Circle

Расширяет **IBaseCircleGeometry**.

Базовая геометрия "Круг".

Конструктор | Поля | События | Методы

#### Конструктор

```
geometry.base.Circle([coordinates[, radius]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | null | Тип: Number[] \| null<br><br>Координаты центра круга. |
| radius | 0 | Тип: Number<br><br>Радиус круга. |

**Пример:**

```
var myCircle = new ymaps.geometry.base.Circle([0, 0], 10);
myCircle.events.add('change', function () {
    alert('Геометрия изменилась');
});
myCircle.freeze();
myCircle.setCoordinates([10, 10]);
myCircle.setRadius(20);
// В этот момент будет сгенерировано единственное событие, и будет выведено
сообщение.
myCircle.unfreeze();
```

#### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **event.Manager** | Менеджер событий геометрии. |

#### События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты центра;<br>• newCoordinates — новые координаты центра;<br>• oldRadius — старый радиус;<br>• newRadius — новый радиус. |

| Имя | Описание |
| --- | --- |
| | Унаследовано от **ICircleGeometryAccess**. |

## Методы

| Имя | Возвращает | Описание |
| --- | --- | --- |
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри круга.<br><br>Унаследован от **ICircleGeometryAccess**. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br><br>Унаследован от **IFreezable**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IBaseGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на окружности точку, ближайшую к anchorPosition.<br><br>Унаследован от **ICircleGeometryAccess**. |
| **getCoordinates**() | Number[] \| null | Возвращает координаты центра круга.<br><br>Унаследован от **ICircleGeometryAccess**. |
| **getRadius**() | Number | Возвращает радиус круга.<br><br>Унаследован от **ICircleGeometryAccess**. |
| **getType**() | String | Возвращает строку "Circle".<br><br>Унаследован от **IBaseCircleGeometry**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br><br>Унаследован от **IFreezable**. |
| **setCoordinates**(coordinates) | **ICircleGeometryAccess** | Задает координаты центра круга.<br><br>Унаследован от **ICircleGeometryAccess**. |
| **setRadius**(radius) | **ICircleGeometryAccess** | Задает радиус круга.<br><br>Унаследован от **ICircleGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br><br>Унаследован от **IFreezable**. |

### Описание полей

#### events

```
{event.Manager} events
```

Менеджер событий геометрии.

## geometry.base.LinearRing

Расширяет **IBaseLinearRingGeometry**.

Базовая геометрия "Замкнутый контур".

Конструктор | Поля | События | Методы

### Конструктор

```
geometry.base.LinearRing([coordinates[, fillRule]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | [] | Тип: Number[][]<br><br>Координаты геометрии. |
| fillRule | "evenOdd" | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Строковый идентификатор, определяющий алгоритм заливки многоугольника. Может принимать одно из двух значений:<br>• evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.<br>• nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри. |

**Пример:**

```
var linearRing = new ymaps.geometry.base.LinearRing([
    [0, 0], [0, 10], [10, 10], [10, 0], [0, 0]
]);
//...
linearRing.set(1, [5, 10]);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | event.Manager | Менеджер событий геометрии. |

## События

| Имя | Описание |
|---|---|
| change | Изменение координат. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты;<br>• oldFillRule — старый алгоритм заливки;<br>• newFillRule — новые алгоритм заливки.<br>Унаследовано от ILinearRingGeometryAccess. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри контура.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **get**(index) | Number[] | Возвращает координаты точки с заданным индексом.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getChildGeometry**(index) | **IPointGeometryAccess** | Создает и возвращает объект **IPointGeometryAccess** для заданной вершины контура.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре точку, ближайшую к anchorPosition.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getCoordinates**() | Number[][] | Возвращает массив координат геометрии.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getFillRule**() | String | Возвращает идентификатор алгоритма заливки.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getLength**() | Integer | Возвращает количество точек в геометрии.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getType**() | String | Возвращает строку "LinearRing".<br>Унаследован от **IBaseLinearRingGeometry**. |
| **insert**(index, coordinates) | **ILinearRingGeometryAccess** | Добавляет новую точку с заданным индексом. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **ILinearRingGeometryAccess**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | Number[] | Удаляет точку с заданным индексом.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **set**(index, coordinates) | **ILinearRingGeometryAccess** | Задает координаты точки с заданным индексом.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **setCoordinates**(coordinates) | **ILinearRingGeometryAccess** | Задает массив координат геометрии.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **setFillRule**(fillRule) | **ILinearRingGeometryAccess** | Задает алгоритм заливки контура.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **splice**(index, number) | Number[][] | Удаляет определенное число точек начиная с заданного индекса. При этом на место удаляемых точек могут быть добавлены новые. Координаты новых точек можно передавать дополнительными аргументами после параметра number.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание полей

### events

```
{event.Manager} events
```

Менеджер событий геометрии.

# geometry.base.LinearRing.fromEncodedCoordinates

Статическая функция.

Создает геометрию **geometry.base.LinearRing** на основе строки закодированных в Base64 координат.

**Возвращает:** Геометрию. (тип **geometry.base.LinearRing**).

```
geometry.base.LinearRing.fromEncodedCoordinates(encodedCoordinates)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| encodedCoordinates* | — | Тип: String<br><br>Закодированные в Base64 координаты вершин контура. |

* Обязательный параметр/опция.

## geometry.base.LinearRing.toEncodedCoordinates

Статическая функция.

**Возвращает:** Строку закодированных в Base64 координат для заданного объекта геометрии. (тип String).

```
geometry.base.LinearRing.toEncodedCoordinates(geometry)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry* | — | Тип: **geometry.base.LinearRing**<br><br>Геометрия. |

* Обязательный параметр/опция.

# geometry.base.LineString

Расширяет **IBaseLineStringGeometry**.

Базовая геометрия "Ломаная линия".

Конструктор | Поля | События | Методы

## Конструктор

```
geometry.base.LineString([coordinates])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | [] | Тип: Number[][]<br><br>Координаты геометрии. |

**Пример:**

```
var lineString = new ymaps.geometry.base.LineString([
    [30, 50], [31, 51], [32, 52]
]);
//...
lineString.set(1, [20, 40]).remove(2);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **event.Manager** | Менеджер событий геометрии. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты.<br>Унаследовано от **ILineStringGeometryAccess**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **get**(index) | Number[] | Возвращает координаты точки с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getChildGeometry**(index) | **IPointGeometryAccess** | Создает и возвращает объект **IPointGeometryAccess** для заданной вершины ломаной линии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getClosest**(anchorPosition) | Object | Ищет на ломаной линии точку, ближайшую к anchorPosition.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getCoordinates**() | Number[][] | Возвращает массив координат геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getLength**() | Integer | Возвращает количество точек в геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getType**() | String | Возвращает строку "LineString".<br>Унаследован от **IBaseLineStringGeometry**. |
| **insert**(index, coordinates) | **ILineStringGeometryAccess** | Добавляет новую точку с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | Number[] | Удаляет точку с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **set**(index, coordinates) | **ILineStringGeometryAccess** | Задает координаты точки с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **setCoordinates**(coordinates) | **ILineStringGeometryAccess** | Задает массив координат геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **splice**(index, number) | Number[][] | Удаляет определенное число точек начиная с заданного индекса. При этом на место удаляемых точек могут быть добавлены новые. Координаты новых точек можно передавать дополнительными аргументами после параметра number.<br>Унаследован от **ILineStringGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание полей

### events

```
{event.Manager} events
```

Менеджер событий геометрии.

# geometry.base.LineString.fromEncodedCoordinates

Статическая функция.

Создает геометрию **geometry.base.LineString** на основе строки закодированных в Base64 координат.

**Возвращает:** Геометрию. (тип **geometry.base.LineString**).

```
geometry.base.LineString.fromEncodedCoordinates(encodedCoordinates)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| encodedCoordinates* | — | Тип: String<br>Закодированные в Base64 координаты вершин ломаной линии. |

* Обязательный параметр/опция.

# geometry.base.LineString.toEncodedCoordinates

Статическая функция.

**Возвращает:** Строку закодированных в Base64 координат для заданного объекта геометрии. (тип String).

```
geometry.base.LineString.toEncodedCoordinates(geometry)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry* | — | Тип: **geometry.base.LineString**<br>Геометрия. |

* Обязательный параметр/опция.

# geometry.base.Point

Расширяет **IBasePointGeometry**.

Базовая геометрия "Точка".

Конструктор | Поля | События | Методы

## Конструктор

```
geometry.base.Point([coordinates])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | null | Тип: Number[] \| null<br><br>Координаты точки. |

**Пример:**

```
var point = new ymaps.geometry.base.Point([30, 50]);

// Точка всегда будет соответствовать центру карты.
map.events.add('boundschange', function (e) {
    if (e.get('newCenter') != e.get('oldCenter')) {
        point.setCoordinates(e.get('newCenter'));
    }
});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **event.Manager** | Менеджер событий геометрии. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты.<br><br>Унаследовано от **IPointGeometryAccess**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IBaseGeometry**. |

| Имя | Возвращает | Описание |
|---|---|---|
| getCoordinates() | Number[] \| null | Возвращает координаты точки.<br>Унаследован от IPointGeometryAccess. |
| getType() | String | Возвращает строку "Point".<br>Унаследован от IBasePointGeometry. |
| setCoordinates(coordinates) | IPointGeometryAccess | Задает координаты точки.<br>Унаследован от IPointGeometryAccess. |

### Описание полей

#### events

```
{event.Manager} events
```

Менеджер событий геометрии.

## geometry.base.Polygon

Расширяет IBasePolygonGeometry.

Базовая геометрия "Многоугольник".

Конструктор | Поля | События | Методы

### Конструктор

```
geometry.base.Polygon([coordinates[, fillRule]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | [] | Тип: Number[][][]<br>Координаты геометрии. |
| fillRule | "evenOdd" | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Строковый идентификатор, определяющий алгоритм заливки многоугольника. Может принимать одно из двух значений:<br>• evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.<br>• nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри. |

**Пример:**

```
var polygon = new ymaps.geometry.base.Polygon([
    // Внешний контур.
    [
        [0, 0], [0, 5], [5, 5], [5, 0], [0, 0]
    ],
    // Внутренний контур.
    [
        [1, 1], [1, 2], [2, 2], [2, 1], [1, 1]
    ]
]);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | event.Manager | Менеджер событий геометрии. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты;<br>• oldFillRule — старый алгоритм заливки;<br>• newFillRule — новые алгоритм заливки.<br>Унаследовано от **IPolygonGeometryAccess**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри многоугольника.<br>Унаследован от **IPolygonGeometryAccess**. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **get**(index) | Number[][] | Возвращает координаты контура с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getChildGeometry**(index) | **ILinearRingGeometryAccess** | Создает и возвращает объект **ILinearRingGeometryAccess** для заданного контура.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре многоугольника точку, ближайшую к anchorPosition.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getCoordinates**() | Number[][][] | Возвращает массив координат геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getFillRule**() | String | Возвращает идентификатор алгоритма заливки. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IPolygonGeometryAccess**. |
| **getLength**() | Integer | Возвращает количество контуров в геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getType**() | String | Возвращает строку "Polygon".<br>Унаследован от **IBasePolygonGeometry**. |
| **insert**(index, path) | **IPolygonGeometryAccess** | Добавляет новый контур с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | **ILinearRingGeometryAccess** | Удаляет контур с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **set**(index, path) | **IPolygonGeometryAccess** | Задает координаты контура с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **setCoordinates**(coordinates) | **IPolygonGeometryAccess** | Задает массив координат геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **setFillRule**(fillRule) | **IPolygonGeometryAccess** | Задает алгоритм заливки многоугольника.<br>Унаследован от **IPolygonGeometryAccess**. |
| **splice**(index, number) | **ILinearRingGeometryAccess**[] | Удаляет определенное число контуров начиная с заданного индекса. При этом на место удаляемых контуров могут быть добавлены новые. Координаты новых контуров можно передавать дополнительными аргументами после параметра number.<br>Унаследован от **IPolygonGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание полей

### events

```
{event.Manager} events
```

Менеджер событий геометрии.

## geometry.base.Polygon.fromEncodedCoordinates

Статическая функция.

Создает геометрию **geometry.base.Polygon** на основе строки закодированных в Base64 координат.

**Возвращает:** Геометрию. (тип **geometry.base.Polygon**).

```
geometry.base.Polygon.fromEncodedCoordinates(encodedCoordinates)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| encodedCoordinates* | — | Тип: String<br><br>Закодированные в Base64 координаты вершин многоугольника. |

* Обязательный параметр/опция.

## geometry.base.Polygon.toEncodedCoordinates

Статическая функция.

**Возвращает:** Строку закодированных в Base64 координат для заданного объекта геометрии. (тип String).

```
geometry.base.Polygon.toEncodedCoordinates(geometry)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry* | — | Тип: **geometry.base.Polygon**<br><br>Геометрия. |

* Обязательный параметр/опция.

# geometry.base.Rectangle

Расширяет **IBaseRectangleGeometry**.

Базовая геометрия "Прямоугольник".

Конструктор | Поля | События | Методы

## Конструктор

```
geometry.base.Rectangle([coordinates])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | null | Тип: Number[][] \| null<br><br>Массив, содержащий координаты двух противоположных углов прямоугольника. |

**Пример:**

```
var rectangle = new ymaps.geometry.base.Rectangle([
    [30, 50], [31, 51]
]);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **event.Manager** | Менеджер событий геометрии. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат углов. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты углов;<br>• newCoordinates — новые координаты углов.<br>Унаследовано от **IRectangleGeometryAccess**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри прямоугольника.<br>Унаследован от **IRectangleGeometryAccess**. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IBaseGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре прямоугольника точку, ближайшую к anchorPosition.<br><br>Унаследован от **IRectangleGeometryAccess**. |
| **getCoordinates**() | Number[][] | Возвращает координаты двух противоположных углов прямоугольника.<br><br>Унаследован от **IRectangleGeometryAccess**. |
| **getType**() | String | Возвращает строку "Rectangle".<br><br>Унаследован от **IBaseRectangleGeometry**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br><br>Унаследован от **IFreezable**. |
| **setCoordinates**(coordinates) | **IRectangleGeometryAccess** | Задает координаты двух противоположных углов прямоугольника.<br><br>Унаследован от **IRectangleGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br><br>Унаследован от **IFreezable**. |

## Описание полей

### events

```
{event.Manager} events
```

Менеджер событий геометрии.

# geometry.Circle

Расширяет **ICircleGeometry**.

Геометрия "Круг".

**См.: Circle**

Конструктор | Поля | События | Методы

## Конструктор

```
geometry.Circle([coordinates[, radius[, options]]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | null | Тип: Number[] \| null<br><br>Координаты центра круга. |
| radius | 0 | Тип: Number<br><br>Радиус круга в метрах. |
| options | — | Тип: Object<br><br>Опции геометрии. |
| options.geodesic | false | Тип: Boolean<br><br>Включает отображение с использованием геодезических линий. |
| options.pixelRendering | "jumpy" | Тип: String<br><br>Метод расчета пиксельных координат отображения в зацикленных проекциях. Опция может принимать одно из следующих значений:<br>• jumpy — отображение располагается как можно ближе к центру области показа карты и может скачкообразно перемещается во время движения карты;<br>• static — отображение всегда располагается в начальном мире и не перемещается при движении карты. |
| options.projection | — | Тип: **IProjection**<br><br>Проекция. |

**Пример:**

```
// Создаем экземпляр класса геометрии круга (указываем координаты и радиус в
метрах).
var circleGeometry = new ymaps.geometry.Circle([30, 50], 10),
// Создаем экземпляр класса геообъекта и передаем нашу геометрию в конструктор.
    circleGeoObject = new ymaps.GeoObject({ geometry: circleGeometry });

// Изменяем радиус геометрии через свойство geometry геообъекта.
circleGeoObject.geometry.setRadius(5)
// Или напрямую.
circleGeometry.setRadius(5);
// Также доступ к circleGeometry можно получить через circleGeoObject.geometry.
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты центра;<br>• newCoordinates — новые координаты центра;<br>• oldRadius — старый радиус;<br>• newRadius — новый радиус.<br>Унаследовано от **ICircleGeometryAccess**. |
| **mapchange** | Сменилась карта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IGeometry**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **pixelgeometrychange** | Изменилась пиксельная геометрия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия **IPixelGeometry**.<br>Унаследовано от **IGeometry**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри круга.<br>Унаследован от **ICircleGeometryAccess**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.<br>Унаследован от **IGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на окружности точку, ближайшую к anchorPosition.<br>Унаследован от **ICircleGeometryAccess**. |
| **getCoordinates**() | Number[] \| null | Возвращает координаты центра круга.<br>Унаследован от **ICircleGeometryAccess**. |
| **getMap**() | **Map** \| null | Возвращает текущую карту.<br>Унаследован от **IGeometry**. |
| **getPixelGeometry**([options]) | **IPixelGeometry** | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.<br>Унаследован от **IGeometry**. |
| **getRadius**() | Number | Возвращает радиус круга.<br>Унаследован от **ICircleGeometryAccess**. |
| **getType**() | String | Возвращает строку "Circle".<br>Унаследован от **ICircleGeometry**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **setCoordinates**(coordinates) | **ICircleGeometryAccess** | Задает координаты центра круга.<br>Унаследован от **ICircleGeometryAccess**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |
| **setRadius**(radius) | **ICircleGeometryAccess** | Задает радиус круга.<br>Унаследован от **ICircleGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

# geometry.json

## geometry.json.circle

Расширяет **IGeometryJson**.

Объект, описывающий JSON-представление геометрии "Круг".

Конструктор | Поля

### Конструктор

```
geometry.json.circle()
```

**Пример:**

```
var geometryJson = {
    type: "Circle",
    coordinates: [1, 2],
    radius: 100
};
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **coordinates** | Number[] \| null | Координаты центра круга. |
| **radius** | Number | Радиус круга. |
| **type** | String | Идентификатор типа геометрии "Круг". Всегда должен принимать значение "Circle". |

### Описание полей

#### coordinates

```
{Number[] | null} coordinates
```

Координаты центра круга.

#### radius

```
{Number} radius
```

Радиус круга.

#### type

```
{String} type
```

Идентификатор типа геометрии "Круг". Всегда должен принимать значение "Circle".

## geometry.json.lineString

Расширяет **IGeometryJson**.

Объект, описывающий JSON-представление геометрии "Ломаная линия".

Конструктор | Поля

### Конструктор

```
geometry.json.lineString()
```

**Пример:**

```
var geometryJson = {
    type: "LineString",
    coordinates: [[1, 2], [3, 5], [7, 11]]
};
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| coordinates | Number[][] | Координаты ломаной линии. |
| type | String | Идентификатор типа геометрии "Ломаная линия". Всегда должен принимать значение "LineString". |

## Описание полей

### coordinates

```
{Number[][]} coordinates
```

Координаты ломаной линии.

### type

```
{String} type
```

Идентификатор типа геометрии "Ломаная линия". Всегда должен принимать значение "LineString".

# geometry.json.Point

Расширяет **IGeometryJson**.

Объект, описывающий JSON-представление геометрии "Точка".

Конструктор | Поля

### Конструктор

```
geometry.json.Point()
```

**Пример:**

```
var geometryJson = {
    type: "Point",
    coordinates: [1, 2]
};
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| type | String | Идентификатор типа геометрии.<br><br>Унаследовано от **IGeometryJson**. |

# geometry.json.polygon

Расширяет **IGeometryJson**.

Объект, описывающий JSON-представление геометрии "Многоугольник".

Конструктор | Поля

## Конструктор

```
geometry.json.polygon()
```

**Пример:**

```
var geometryJson = {
    type: "Polygon",
    coordinates: [
        [[0, 0], [7, 11]],
        [[1, 2], [3, 5]]
    ]
};
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **coordinates** | Number[][][] | Координаты многоугольника. |
| **fillRule** | String | Идентификатор алгоритма заливки многоугольника. Может принимать одно из двух значений:<br>• evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.<br>• nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри. |
| **type** | String | Идентификатор типа геометрии.<br>Унаследовано от **IGeometryJson**. |

## Описание полей

### coordinates

```
{Number[][][]} coordinates
```

Координаты многоугольника.

### fillRule

```
{String} fillRule
```

Идентификатор алгоритма заливки многоугольника. Может принимать одно из двух значений:

- evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.
- nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри.

# geometry.json.rectangle

Расширяет **IGeometryJson**.

Объект, описывающий JSON-представление геометрии "Прямоугольник".

Конструктор | Поля

## Конструктор

```
geometry.json.rectangle()
```

**Пример:**

```
var geometryJson = {
    type: "Rectangle",
    coordinates: [[1, 2], [3, 5]]
};
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **coordinates** | Number[][] \| null | Координаты двух противоположных углов прямоугольника. |
| **type** | String | Идентификатор типа геометрии "Прямоугольник". Всегда должен принимать значение "Rectangle". |

## Описание полей

### coordinates

```
{Number[][] | null} coordinates
```

Координаты двух противоположных углов прямоугольника.

### type

```
{String} type
```

Идентификатор типа геометрии "Прямоугольник". Всегда должен принимать значение "Rectangle".

# geometry.LineString

Расширяет **ILineStringGeometry**.

Геометрия "Ломаная линия".

**См.:** **Polyline**

Конструктор | Поля | События | Методы

## Конструктор

```
geometry.LineString([coordinates[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | [] | Тип: Number[][]<br><br>Координаты геометрии. |
| options | — | Тип: Object<br><br>Опции геометрии. |
| options.coordRendering | — | Тип: String<br><br>Строковый идентификатор, определяющий алгоритм пересчета координат геометрии в пиксельные координаты. Для геометрии "Ломаная линия" может принимать одно из двух значений:<br>• shortestPath — алгоритм, учитывающий зацикленность проекции по осям, и формирующий пиксельные координаты так, чтобы расстояние между двумя соседними точками было минимальным;<br>• straightPath — алгоритм, не учитывающий зацикленность проекции. |
| options.geodesic | false | Тип: Boolean<br><br>Включает отображение с использованием геодезических линий. |
| options.pixelRendering | "jumpy" | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Метод расчета пиксельных координат отображения в зацикленных проекциях. Опция может принимать одно из следующих значений:<br>• jumpy — отображение располагается как можно ближе к центру области показа карты и может скачкообразно перемещается во время движения карты;<br>• static — отображение всегда располагается в начальном мире и не перемещается при движении карты. |
| options.projection | — | Тип: **IProjection**<br><br>Проекция. |
| options.simplification | true | Тип: Boolean<br><br>Включает симплификацию при рендеринге пиксельной геометрии. |

**Пример:**

```
// Создаем инстанцию геометрии линии (указываем координаты вершин).
var lineStringGeometry = new ymaps.geometry.LineString([
    [30, 50], [31, 51], [32, 52]
]),
// Создаем инстанцию геообъекта и передаем нашу геометрию в конструктор.
    lineStringGeoObject = new ymaps.GeoObject({ geometry: lineStringGeometry });

lineStringGeometry.events.add('change', function (e) {
   alert([e.get('newCoordinates'), e.get('oldCoordinates')]);
});

// Изменяем вершины через свойство geometry геообъекта (задаем новые координаты
второй точке в линии).
lineStringGeoObject.geometry
    .set(1, [20, 40])
    .remove(2);
// Или напрямую.
lineStringGeometry
    .set(1, [20, 40])
    .remove(2);
// Также доступ к lineStringGeometry можно получить через
lineStringGeoObject.geometry.
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| change | Изменение координат. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты.<br>Унаследовано от ILineStringGeometryAccess. |
| mapchange | Сменилась карта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IGeometry. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| pixelgeometrychange | Изменилась пиксельная геометрия. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия IPixelGeometry.<br>Унаследовано от IGeometry. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| freeze() | IFreezable | Переводит объект в "замороженный" режим.<br>Унаследован от IFreezable. |
| get(index) | Number[] | Возвращает координаты точки с заданным индексом.<br>Унаследован от ILineStringGeometryAccess. |
| getBounds() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.<br>Унаследован от IGeometry. |
| getChildGeometry(index) | IPointGeometryAccess | Создает и возвращает объект IPointGeometryAccess для заданной вершины ломаной линии.<br>Унаследован от ILineStringGeometryAccess. |
| getClosest(anchorPosition) | Object | Ищет на ломаной линии точку, ближайшую к anchorPosition.<br>Унаследован от ILineStringGeometryAccess. |

| Имя | Возвращает | Описание |
|---|---|---|
| **getCoordinates**() | Number[][] | Возвращает массив координат геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getDistance**([from[, to]]) | Number | Возвращает длину указанного отрезка, либо всей линии, если ограничители не заданы. |
| **getLength**() | Integer | Возвращает количество точек в геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getMap**() | **Map** \| null | Возвращает текущую карту.<br>Унаследован от **IGeometry**. |
| **getPixelGeometry**([options]) | **IPixelGeometry** | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.<br>Унаследован от **IGeometry**. |
| **getType**() | String | Возвращает строку "LineString".<br>Унаследован от **ILineStringGeometry**. |
| **insert**(index, coordinates) | **ILineStringGeometryAccess** | Добавляет новую точку с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | Number[] | Удаляет точку с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **set**(index, coordinates) | **ILineStringGeometryAccess** | Задает координаты точки с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **setCoordinates**(coordinates) | **ILineStringGeometryAccess** | Задает массив координат геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |
| **splice**(index, number) | Number[][] | Удаляет определенное число точек начиная с заданного индекса. При этом на место удаляемых точек могут быть добавлены новые. Координаты новых точек можно передавать дополнительными аргументами после параметра number. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от ILineStringGeometryAccess. |
| unfreeze() | IFreezable | Переводит объект в активный режим.<br>Унаследован от IFreezable. |

## Описание методов

### getDistance

```
{Number} getDistance([from[, to]])
```

**Возвращает** длину указанного отрезка, либо всей линии, если ограничители не заданы.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| from | 0 | Тип: Number<br>Начиная с какой точки считать длину. |
| to | — | Тип: Number<br>До какой точки считать длину. Если не указано, берётся последняя точка. |

**Пример:**

```
var lineStringGeometry = new ymaps.geometry.LineString([[30, 50], [31, 51], [32, 52]]),
    geoObject = new ymaps.GeoObject({ geometry: lineStringGeometry });
myMap.geoObjects.add(geoObject);
// Длина всей линии.
console.log(geoObject.geometry.getDistance());
// Длина отрезка от первой до второй точки.
console.log(geoObject.geometry.getDistance(0, 1));
```

## geometry.LineString.fromEncodedCoordinates

Статическая функция.

Создает геометрию geometry.LineString на основе строки закодированных в Base64 координат.

**Возвращает:** Геометрию. (тип geometry.LineString).

```
geometry.LineString.fromEncodedCoordinates(encodedCoordinates)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| encodedCoordinates* | — | Тип: String<br>Закодированные в Base64 координаты вершин ломаной линии. |

* Обязательный параметр/опция.

**Пример:**

```
var base64Coords = "gMPJAYDw-gJAQg8AQEIPAEBCDwBAQg8A",
    geometry = ymaps.geometry.LineString.fromEncodedCoordinates(base64Coords);
    polyline = new ymaps.Polyline(geometry);
```

## geometry.LineString.toEncodedCoordinates

Статическая функция.

**Возвращает:** Строку закодированных в Base64 координат для заданного объекта геометрии. (тип String).

```
geometry.LineString.toEncodedCoordinates(geometry)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry* | — | Тип: **geometry.LineString**<br><br>Геометрия. |

* Обязательный параметр/опция.

# geometry.pixel

## geometry.pixel.Circle

Расширяет **IPixelCircleGeometry**.

Пиксельная геометрия "Круг".

Конструктор | Поля | Методы

### Конструктор

```
geometry.pixel.Circle(coordinates, radius[, metaData])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates * | — | Тип: Number[] \| null<br><br>Координаты центра круга. |
| radius * | — | Тип: Number \| null<br><br>Радиус круга. |
| metaData | — | Тип: Object<br><br>Метаданные. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br><br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IBaseGeometry**. |
| **getCoordinates**() | Number[] | Возвращает координаты центра круга.<br><br>Унаследован от **IPixelCircleGeometry**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br>Унаследован от **IPixelGeometry**. |
| **getRadius**() | Number | Возвращает радиус круга.<br>Унаследован от **IPixelCircleGeometry**. |
| **getType**() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от **IBaseGeometry**. |
| **scale**(factor) | **IPixelGeometry** | Создает отмасштабированную копию геометрии.<br>Унаследован от **IPixelGeometry**. |
| **shift**(offset) | **IPixelGeometry** | Создает копию геометрии сдвинутую на заданную величину.<br>Унаследован от **IPixelGeometry**. |

# geometry.pixel.LineString

Расширяет **IPixelLineStringGeometry**.

Пиксельная геометрия "Ломаная линия".

Конструктор | Поля | Методы

## Конструктор

```
geometry.pixel.LineString(coordinates[, metaData])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates * | — | Тип: Number[][]<br>Координаты линии. |
| metaData | — | Тип: Object<br>Метаданные. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на ломаной линии точку, ближайшую к anchorPosition.<br>Унаследован от **IPixelLineStringGeometry**. |
| **getCoordinates**() | Number[][] | Возвращает координаты линии.<br>Унаследован от **IPixelLineStringGeometry**. |
| **getLength**() | Integer | Возвращает количество точек в геометрии.<br>Унаследован от **IPixelLineStringGeometry**. |
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br>Унаследован от **IPixelGeometry**. |
| **getType**() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от **IBaseGeometry**. |
| **scale**(factor) | **IPixelGeometry** | Создает отмасштабированную копию геометрии.<br>Унаследован от **IPixelGeometry**. |
| **shift**(offset) | **IPixelGeometry** | Создает копию геометрии сдвинутую на заданную величину.<br>Унаследован от **IPixelGeometry**. |

# geometry.pixel.MultiPolygon

Расширяет **IPixelMultiPolygonGeometry**.

Пиксельная геометрия "Многоугольник из нескольких фигур".

Конструктор | Поля | Методы

## Конструктор

```
geometry.pixel.MultiPolygon(coordinates, fillRule[, metaData])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates * | — | Тип: Number[][][][]<br><br>Координаты многоугольников. |
| fillRule * | — | Тип: String<br><br>Строковый идентификатор, определяющий алгоритм заливки многоугольников. Может принимать одно из двух значений:<br>• evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.<br>• nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри. |
| metaData | — | Тип: Object<br><br>Метаданные. |
| metaData.convex | false | Тип: Boolean<br><br>Признак выпуклости многоугольника. true — выпуклый, false — невыпуклый. Для выпуклых многоугольников быстрее рассчитывается попадание точки в многоугольник. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br><br>Унаследовано от IEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет лежит ли переданная точка внутри мультимногоугольника.<br><br>Унаследован от **IPixelMultiPolygonGeometry**. |
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br><br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IBaseGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре мультимногоугольника точку, ближайшую к anchorPosition.<br><br>Унаследован от **IPixelMultiPolygonGeometry**. |
| **getCoordinates**() | Number[][][][] | Возвращает координаты мультимногоугольника.<br><br>Унаследован от **IPixelMultiPolygonGeometry**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **getFillRule**() | String | Возвращает строковый идентификатор, определяющий алгоритм заливки мультимногоугольника. Идентификатор может иметь одно из двух значений:<br>• evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.<br>• nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри.<br>.<br>Унаследован от **IPixelMultiPolygonGeometry**. |
| **getLength**() | Integer | Возвращает количество многоугольников в мультимногоугольнике.<br>Унаследован от **IPixelMultiPolygonGeometry**. |
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br>Унаследован от **IPixelGeometry**. |
| **getType**() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от **IBaseGeometry**. |
| **scale**(factor) | **IPixelGeometry** | Создает отмасштабированную копию геометрии.<br>Унаследован от **IPixelGeometry**. |
| **shift**(offset) | **IPixelGeometry** | Создает копию геометрии сдвинутую на заданную величину.<br>Унаследован от **IPixelGeometry**. |

# geometry.pixel.Point

Расширяет **IPixelPointGeometry**.

Пиксельная геометрия "Точка".

Конструктор | Поля | Методы

## Конструктор

```
geometry.pixel.Point(position[, metaData])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position * | — | Тип: Number[] \| null<br><br>Координаты точки. |
| metaData | — | Тип: Object<br><br>Метаданные. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br><br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IBaseGeometry**. |
| **getCoordinates**() | Number[] | Возвращает координаты точки.<br><br>Унаследован от **IPixelPointGeometry**. |
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br><br>Унаследован от **IPixelGeometry**. |
| **getType**() | String | Возвращает идентификатор типа геометрии. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IBaseGeometry**. |
| **scale**(factor) | **IPixelGeometry** | Создает отмасштабированную копию геометрии.<br>Унаследован от **IPixelGeometry**. |
| **shift**(offset) | **IPixelGeometry** | Создает копию геометрии сдвинутую на заданную величину.<br>Унаследован от **IPixelGeometry**. |

# geometry.pixel.Polygon

Расширяет **IPixelPolygonGeometry**.

Пиксельная геометрия "Многоугольник".

Конструктор | Поля | Методы

## Конструктор

```
geometry.pixel.Polygon(coordinates, fillRule[, metaData])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| `coordinates` * | — | Тип: Number[][][]<br>Координаты многоугольника. |
| `fillRule` * | — | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Строковый идентификатор, определяющий алгоритм заливки многоугольника. Может принимать одно из двух значений:<br>• evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.<br>• nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри. |
| metaData | — | Тип: Object<br><br>Метаданные. |
| metaData.convex | false | Тип: Boolean<br><br>Признак выпуклости многоугольника. true — выпуклый, false — невыпуклый. Для выпуклых многоугольников быстрее рассчитывается попадание точки в многоугольник. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет лежит ли переданная точка внутри многоугольника. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IPixelPolygonGeometry**. |
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br><br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IBaseGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре многоугольника точку, ближайшую к anchorPosition.<br><br>Унаследован от **IPixelPolygonGeometry**. |
| **getCoordinates**() | Number[][][] | Возвращает координаты многоугольника.<br><br>Унаследован от **IPixelPolygonGeometry**. |
| **getFillRule**() | String | Возвращает строковый идентификатор, определяющий алгоритм заливки многоугольника. Идентификатор может принимать одно из двух значений:<br><br>• evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.<br>• nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | .<br><br>Унаследован от **IPixelPolygonGeometry**. |
| **getLength**() | Integer | Возвращает количество контуров в многоугольнике.<br><br>Унаследован от **IPixelPolygonGeometry**. |
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br><br>Унаследован от **IPixelGeometry**. |
| **getType**() | String | Возвращает идентификатор типа геометрии.<br><br>Унаследован от **IBaseGeometry**. |
| **scale**(factor) | **IPixelGeometry** | Создает отмасштабированную копию геометрии.<br><br>Унаследован от **IPixelGeometry**. |
| **shift**(offset) | **IPixelGeometry** | Создает копию геометрии сдвинутую на заданную величину.<br><br>Унаследован от **IPixelGeometry**. |

# geometry.pixel.Rectangle

Расширяет **IPixelRectangleGeometry**.

Пиксельная геометрия "Прямоугольник".

Конструктор | Поля | Методы

## Конструктор

```
geometry.pixel.Rectangle([coordinates[, metaData]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | null | Тип: Number[][] \| null<br><br>Координаты двух противоположных углов прямоугольника. |
| metaData | — | Тип: Object<br><br>Метаданные. |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br><br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IBaseGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на прямоугольнике точку, ближайшую к anchorPosition.<br><br>Унаследован от **IPixelRectangleGeometry**. |
| **getCoordinates**() | Number[][] | Возвращает координаты двух противоположных углов прямоугольника.<br><br>Унаследован от **IPixelRectangleGeometry**. |
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br><br>Унаследован от **IPixelGeometry**. |
| **getType**() | String | Возвращает идентификатор типа геометрии.<br><br>Унаследован от **IBaseGeometry**. |
| **scale**(factor) | **IPixelGeometry** | Создает отмасштабированную копию геометрии.<br><br>Унаследован от **IPixelGeometry**. |
| **shift**(offset) | **IPixelGeometry** | Создает копию геометрии сдвинутую на заданную величину.<br><br>Унаследован от **IPixelGeometry**. |

# geometry.Point

Расширяет **IPointGeometry**.

Геометрия "Точка".

**См.: Placemark**

Конструктор | Поля | События | Методы

## Конструктор

```
geometry.Point([position[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position | null | Тип: Number[]<br><br>Координаты точки. |
| options | — | Тип: Object<br><br>Опции геометрии. |
| options.pixelRendering | "jumpy" | Тип: String<br><br>Метод расчета пиксельных координат отображения в зацикленных проекциях. Опция может принимать одно из следующих значений:<br>• jumpy — отображение располагается как можно ближе к центру области показа карты и может скачкообразно перемещается во время движения карты;<br>• static — отображение всегда располагается в начальном мире и не перемещается при движении карты. |
| options.projection | — | Тип: **IProjection**<br><br>Проекция. |

**Пример:**

```
// Создаем инстанцию геометрии точки (указываем координаты).
var pointGeometry = new ymaps.geometry.Point([30, 50]),
// Создаем инстанцию геообъекта и передаем нашу геометрию в конструктор.
    placemark = new ymaps.GeoObject({ geometry: pointGeometry });

// Изменяем вершины через свойство geometry геообъекта.
placemark.geometry.setCoordinates([20, 40]);
// Или напрямую.
pointGeometry.setCoordinates([20, 40]);
// Также доступ к pointGeometry можно получить через placemark.geometry
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| change | Изменение координат. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты.<br>Унаследовано от IPointGeometryAccess. |
| mapchange | Сменилась карта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IGeometry. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| pixelgeometrychange | Изменилась пиксельная геометрия. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия IPixelGeometry.<br>Унаследовано от IGeometry. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getBounds() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.<br>Унаследован от IGeometry. |
| getCoordinates() | Number[] \| null | Возвращает координаты точки.<br>Унаследован от IPointGeometryAccess. |
| getMap() | Map \| null | Возвращает текущую карту.<br>Унаследован от IGeometry. |
| getPixelGeometry([options]) | IPixelGeometry | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IGeometry**. |
| **getType**() | String | Возвращает строку "Point".<br>Унаследован от **IPointGeometry**. |
| **setCoordinates**(coordinates) | **IPointGeometryAccess** | Задает координаты точки.<br>Унаследован от **IPointGeometryAccess**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |

# geometry.Polygon

Расширяет **IPolygonGeometry**.

Геометрия "Многоугольник".

**См.: Polygon**

Конструктор | Поля | События | Методы

## Конструктор

```
geometry.Polygon([coordinates[, fillRule[, options]]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | [] | Тип: Number[][][]<br><br>Координаты геометрии. Трехмерный массив, элементами которого являются двумерные массивы координат контуров многоугольника. Первый элемент описывает внешний контур, остальные — внутренние. |
| fillRule | "evenOdd" | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Строковый идентификатор, определяющий алгоритм заливки многоугольника. Может принимать одно из двух значений:<br>• evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.<br>• nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри. |
| options | — | Тип: Object<br><br>Опции геометрии. |
| options.coordRendering | "shortestPath" | Тип: String<br><br>Строковый идентификатор, определяющий алгоритм пересчета координат геометрии в пиксельные координаты. Может принимать одно из двух значений:<br>• shortestPath — алгоритм, учитывающий зацикленность проекции по осям, и формирующий пиксельные координаты так, чтобы расстояние между двумя соседними точками было минимальным;<br>• straightPath — алгоритм, не учитывающий зацикленность проекции. |
| options.geodesic | false | Тип: Boolean<br><br>Включает отображение с использованием геодезических линий. |
| options.pixelRendering | "jumpy" | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Метод расчета пиксельных координат отображения в зацикленных проекциях. Опция может принимать одно из следующих значений:<br>• jumpy — отображение располагается как можно ближе к центру области показа карты и может скачкообразно перемещается во время движения карты;<br>• static — отображение всегда располагается в начальном мире и не перемещается при движении карты. |
| options.projection | — | Тип: **IProjection**<br><br>Проекция. |
| options.simplification | true | Тип: Boolean<br><br>Включает симплификацию при рендеринге пиксельной геометрии. |

**Пример:**

```
// Создаем инстанцию геометрии многоугольника (указываем координаты вершин
контуров).
var polygonGeometry = new ymaps.geometry.Polygon([
    // Внешний контур.
    [
        [0, 0], [0, 5], [5, 5], [5, 0], [0, 0]
    ],
    // Внутренний контур.
    [
        [1, 1], [1, 2], [2, 2], [2, 1], [1, 1]
    ]
]),
// Создаем инстанцию геообъекта и передаем нашу геометрию в конструктор.
    polygonGeoObject = new ymaps.GeoObject({ geometry: polygonGeometry });

// Также доступ к polygonGeometry можно получить через polygonGeoObject.geometry.
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты;<br>• oldFillRule — старый алгоритм заливки;<br>• newFillRule — новые алгоритм заливки.<br>Унаследовано от **IPolygonGeometryAccess**. |
| **mapchange** | Сменилась карта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IGeometry**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **pixelgeometrychange** | Изменилась пиксельная геометрия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия **IPixelGeometry**.<br>Унаследовано от **IGeometry**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри многоугольника.<br>Унаследован от **IPolygonGeometryAccess**. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **get**(index) | Number[][] | Возвращает координаты контура с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.<br>Унаследован от **IGeometry**. |
| **getChildGeometry**(index) | **ILinearRingGeometryAccess** | Создает и возвращает объект **ILinearRingGeometryAccess** для заданного контура. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IPolygonGeometryAccess**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре многоугольника точку, ближайшую к anchorPosition.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getCoordinates**() | Number[][][] | Возвращает массив координат геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getFillRule**() | String | Возвращает идентификатор алгоритма заливки.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getLength**() | Integer | Возвращает количество контуров в геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getMap**() | **Map** \| null | Возвращает текущую карту.<br>Унаследован от **IGeometry**. |
| **getPixelGeometry**([options]) | **IPixelGeometry** | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.<br>Унаследован от **IGeometry**. |
| **getType**() | String | Возвращает строку "Polygon".<br>Унаследован от **IPolygonGeometry**. |
| **insert**(index, path) | **IPolygonGeometryAccess** | Добавляет новый контур с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | **ILinearRingGeometryAccess** | Удаляет контур с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **set**(index, path) | **IPolygonGeometryAccess** | Задает координаты контура с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **setCoordinates**(coordinates) | **IPolygonGeometryAccess** | Задает массив координат геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **setFillRule**(fillRule) | **IPolygonGeometryAccess** | Задает алгоритм заливки многоугольника. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IPolygonGeometryAccess**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |
| **splice**(index, number) | **ILinearRingGeometryAccess**[] | Удаляет определенное число контуров начиная с заданного индекса. При этом на место удаляемых контуров могут быть добавлены новые. Координаты новых контуров можно передавать дополнительными аргументами после параметра number.<br>Унаследован от **IPolygonGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## geometry.Polygon.fromEncodedCoordinates

Статическая функция.

Создает геометрию **geometry.Polygon** на основе строки закодированных в Base64 координат.

**Возвращает:** Геометрию. (тип **geometry.Polygon**).

```
geometry.Polygon.fromEncodedCoordinates(encodedCoordinates)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| encodedCoordinates* | — | Тип: String<br><br>Закодированные в Base64 координаты вершин ломаной линии. |

* Обязательный параметр/опция.

**Пример:**

```
var base64Coords =
"AAAAAAAAAAAAAAAAQEtMAEBLTAAAAAAAAAAAAMC0s__AtLP_AAAAAA==;QEIPAEBCDwAAAAAQEIPAE
BCDwAAAAAAAAAAAMC98P_AvfD_AAAAAA==",
    geometry = ymaps.geometry.Polygon.fromEncodedCoordinates(base64Coords);
    polygon = new ymaps.Polygon(geometry);
```

## geometry.Polygon.toEncodedCoordinates

Статическая функция.

**Возвращает:** Строку закодированных в Base64 координат для заданного объекта геометрии. (тип String).

```
geometry.Polygon.toEncodedCoordinates(geometry)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry* | — | Тип: **geometry.Polygon**<br><br>Геометрия. |

* Обязательный параметр/опция.

# geometry.Rectangle

Расширяет **IRectangleGeometry**.

Геометрия "Прямоугольник".

**См.:** **Rectangle**

Конструктор | Поля | События | Методы

## Конструктор

```
geometry.Rectangle([coordinates[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates | null | Тип: Number[][] \| null<br><br>Массив, содержащий координаты двух противоположных углов прямоугольника. |
| options | — | Тип: Object<br><br>Опции геометрии. |
| options.coordRendering | — | Тип: String<br><br>Строковый идентификатор, определяющий алгоритм пересчета координат геометрии в пиксельные координаты. Для геометрии "Прямоугольник" может принимать одно из трех значений:<br>• shortestPath — алгоритм, учитывающий зацикленность проекции по осям, и формирующий пиксельные координаты так, чтобы расстояние между противоположными углами было минимальным;<br>• straightPath — алгоритм, не учитывающий зацикленность проекции;<br>• boundsPath — алгоритм, трактующий координаты углов прямоугольника, как координаты соответственно нижнего и верхнего углов ограничивающей области. При расчете диагонали по зацикленным осям проекции всегда выбирается направление обхода против часовой стрелки. |
| options.geodesic | false | Тип: Boolean<br><br>Включает отображение с использованием геодезических линий. |
| options.pixelRendering | "jumpy" | Тип: String |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Метод расчета пиксельных координат отображения в зацикленных проекциях. Опция может принимать одно из следующих значений:<br>• jumpy — отображение располагается как можно ближе к центру области показа карты и может скачкообразно перемещается во время движения карты;<br>• static — отображение всегда располагается в начальном мире и не перемещается при движении карты. |
| options.projection | — | Тип: **IProjection**<br><br>Проекция. |

**Пример:**

```
// Создаем инстанцию геометрии прямоугольника (указываем координаты
противоположных углов).
var rectangleGeometry = new ymaps.geometry.Rectangle([[30, 50], [31, 51]]),
// Создаем инстанцию геообъекта и передаем нашу геометрию в конструктор.
    rectangleGeoObject = new ymaps.GeoObject({ geometry: rectangleGeometry });

// Изменяем координаты через свойство geometry геообъекта.
rectangleGeoObject.geometry.setCoordinates([[10, 20], [51, 71]]);
// Или напрямую.
rectangleGeometry.setCoordinates([[10, 20], [51, 71]]);
// Также доступ к rectangleGeometry можно получить через
rectangleGeoObject.geometry.
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат углов. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты углов;<br>• newCoordinates — новые координаты углов.<br><br>Унаследовано от **IRectangleGeometryAccess**. |
| **mapchange** | Сменилась карта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IGeometry**. |

| Имя | Описание |
|---|---|
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **pixelgeometrychange** | Изменилась пиксельная геометрия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия **IPixelGeometry**.<br>Унаследовано от **IGeometry**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри прямоугольника.<br>Унаследован от **IRectangleGeometryAccess**. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.<br>Унаследован от **IGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре прямоугольника точку, ближайшую к anchorPosition.<br>Унаследован от **IRectangleGeometryAccess**. |
| **getCoordinates**() | Number[][] | Возвращает координаты двух противоположных углов прямоугольника.<br>Унаследован от **IRectangleGeometryAccess**. |
| **getMap**() | **Map** \| null | Возвращает текущую карту.<br>Унаследован от **IGeometry**. |
| **getPixelGeometry**([options]) | **IPixelGeometry** | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.<br>Унаследован от **IGeometry**. |
| **getType**() | String | Возвращает строку "Rectangle".<br>Унаследован от **IRectangleGeometry**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **setCoordinates**(coordinates) | **IRectangleGeometryAccess** | Задает координаты двух противоположных углов прямоугольника. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IRectangleGeometryAccess**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

# geometryEditor

## geometryEditor.LineString

Расширяет **IGeometryEditor**.

Редактор геометрии "Ломаная линия".

Конструктор | Поля | События | Методы

### Конструктор

```
geometryEditor.LineString(geometry[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **ILineStringGeometry**<br><br>Геометрия "Ломаная линия". |
| options | — | Тип: Object<br><br>Опции редактора геометрии. |
| options.dblClickHandler | — | Тип: Function<br><br>Обработчик двойного щелчка по вершине. Принимает ссылку на модель редактируемой вершины. По умолчанию, обработчик определен функцией, удаляющей соответствующую вершину. |
| options.drawingCursor | "arrow" | Тип: Boolean<br><br>Курсор мыши в режиме добавления новых вершин. |
| options.drawOver | true | Тип: Boolean<br><br>Разрешает ставить точки поверх объектов карты в режиме добавления новых вершин. |
| options.edgeLayout | — | Тип: Function<br><br>Класс макета для промежуточных меток. |
| options.maxPoints | Infinity | Тип: Number<br><br>Максимально допустимое число вершин в ломаной линии. |
| options.menuManager | — | Тип: Function |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Диспетчер контекстного меню, открывающегося при щелчке по вершине. Принимает два аргумента:<br>• массив объектов, описывающих пункты контекстного меню для этой вершины;<br>• ссылку на модель редактируемой вершины.<br>Диспетчер может изменять данные в переданном массиве. Должен возвращать массив объектов, описывающих пункты контекстного меню. |
| options.useMapMarginInDrawing | true | Тип: Boolean<br>Нужно ли учитывать отступы карты в режиме рисования. |
| options.vertexLayout | — | Тип: Function<br>Класс макета для меток на вершинах ломаной линии. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **geometry** | **IGeometry** | Редактируемая геометрия.<br>Унаследовано от **IGeometryEditor**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **state** | **IDataManager** | Менеджер состояния редактора геометрии.<br>Поля данных, доступные посредством методов get и set:<br>• editing — определяет включен ли редактор. Тип — Boolean. Значение по умолчанию — false.<br>• drawing — определяет включен ли режим добавления новых вершин. Тип — Boolean. Значение по умолчанию — false.<br>• drawingFrom — определяет то, как добавляются новые точки в режиме добавления. Может принимать одно из двух строковых значений: "begin" — точки добавляются в начало ломаной линии, "end"- точки добавляются в конец. Значение по умолчанию — "end". |

## События

| Имя | Описание |
|---|---|
| beforeedgedrag | Событие, предшествующее событию edgedrag. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• edgeModel — ссылка на модель перетаскиваемой метки;<br>• globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие edgedrag будет отменено. |
| beforeedgedragstart | Событие, предшествующее событию edgedragstart. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется;<br>• edgeModel — ссылка на модель перетаскиваемой метки;<br>• globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие edgedragstart, будет отменено. |
| beforevertexadd | Событие, предшествующее событию vertexadd. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;<br>• vertexIndex — индекс добавляемой вершины;<br>• globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах.<br>Имена методов, доступных через метод Event.callMethod:<br>• setGlobalPixels — метод позволяет скорректировать значения координат добавляемой вершины. В качестве аргумента принимает новые глобальные пиксельные координаты вершины в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexadd будет отменено. |
| beforevertexdrag | Событие, предшествующее событию vertexdrag. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• vertexModel — ссылка на модель перетаскиваемой вершины;<br>• globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах. |

| Имя | Описание |
|---|---|
| | Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexdrag будет отменено. |
| **beforevertexdragstart** | Событие, предшествующее событию vertexdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;<br>• vertexModel — ссылка на модель перетаскиваемой вершины;<br>• globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие vertexdragstart, будет отменено. |
| **beforevertexdraw** | Событие, предшествующее событию vertexdraw. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;<br>• vertexIndex — индекс добавляемой вершины;<br>• globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах.<br>Имена методов, доступных через метод Event.callMethod:<br>• setGlobalPixels — метод позволяет скорректировать значения координат добавляемой вершины. В качестве аргумента принимает новые глобальные пиксельные координаты вершины в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexdraw будет отменено. |
| **drawingstart** | Включение режима добавления новых вершин. Экземпляр класса **Event**. |
| **drawingstop** | Отключение режима добавления новых вершин. Экземпляр класса **Event**. |
| **edgedrag** | Перетаскивание промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• edgeModel — ссылка на модель перетаскиваемой метки;<br>• globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах. |
| **edgedragend** | Завершение перетаскивания промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• edgeModel — ссылка на модель перетаскиваемой метки;<br>• globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах. |

| Имя | Описание |
|---|---|
| **edgedragstart** | Начало перетаскивания промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;<br>• edgeModel — ссылка на модель перетаскиваемой метки;<br>• globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах. |
| **editingstart** | Включение режима редактирования вершин. Экземпляр класса **Event**. |
| **editingstop** | Отключение режима редактирования вершин. Экземпляр класса **Event**. |
| **framingstart** | Включение режима масштабирования. Экземпляр класса **Event**. |
| **framingstop** | Отключение режима масштабирования. Экземпляр класса **Event**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **statechange** | Изменение состояния редактора геометрии. Экземпляр класса **Event**.<br>Унаследовано от **IGeometryEditor**. |
| **vertexadd** | Добавление новой вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• parentModel — ссылка на родительскую, для добавленной вершины, модель данных;<br>• vertexIndex — индекс добавленной вершины;<br>• globalPixels — координаты добавленной вершины в глобальных пиксельных координатах. |
| **vertexdrag** | Перетаскивание вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• vertexModel — ссылка на модель перетаскиваемой вершины;<br>• globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах. |
| **vertexdragend** | Завершение перетаскивания вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• vertexModel — ссылка на модель перетаскиваемой вершины;<br>• globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах. |
| **vertexdragstart** | Начало перетаскивания вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;<br>• vertexModel — ссылка на модель перетаскиваемой вершины;<br>• globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах. |

| Имя | Описание |
| --- | --- |
| vertexdraw | Рисование новой вершины. Данное событие, как правило, предшествует событию добавления вершины и возникает в момент движения мышью при включенном режиме добавления новых вершин. На основе данных, передаваемых в данном событии, отображаются направляющие линии в режиме добавления новых вершин. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;<br>• vertexIndex — индекс добавляемой вершины;<br>• globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах. |

## Методы

| Имя | Возвращает | Описание |
| --- | --- | --- |
| getModel() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом модели в момент ее фактического создания, либо отклоняется с одним из следующих кодов ошибки:<br>• Canceled — режим редактирования отключен до момента фактического создания модели;<br>• Editor wasn't started — режим редактирования не включен.<br>. |
| getModelSync() | geometryEditor.model.RootLineString \| null | Возвращает модель данных редактора или null, если на момент вызова она отсутствует. |
| getView() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом отображения в момент его фактического создания, либо отклоняется с одним из следующих кодов ошибки:<br>• Canceled — режим редактирования отключен до момента фактического создания отображения;<br>• Editor wasn't started — режим редактирования не включен.<br>. |
| getViewSync() | geometryEditor.view.Path \| null | Возвращает отображение редактора или null, если на момент вызова оно отсутствует. |
| startDrawing() | vow.Promise | Включает режим добавления новых вершин в ломаную линию. Включение происходит асинхронно. |
| startEditing() | vow.Promise | Включает режим редактирования. Включение происходит асинхронно. |
| startFraming() | vow.Promise | Включает режим масштабирования для ломаной линии. Включение происходит асинхронно. |

| Имя | Возвращает | Описание |
|---|---|---|
| stopDrawing() | | Отключает режим добавления новых вершин в ломаную линию. |
| stopEditing() | | Отключает режим редактирования. |
| stopFraming() | | Отключает режим масштабирования. |

## Описание полей

### state

```
{IDataManager} state
```

Менеджер состояния редактора геометрии.

Поля данных, доступные посредством методов get и set:

- editing — определяет включен ли редактор. Тип — Boolean. Значение по умолчанию — false.
- drawing — определяет включен ли режим добавления новых вершин. Тип — Boolean. Значение по умолчанию — false.
- drawingFrom — определяет то, как добавляются новые точки в режиме добавления. Может принимать одно из двух строковых значений: "begin" — точки добавляются в начало ломаной линии, "end"- точки добавляются в конец. Значение по умолчанию — "end".

## Описание событий

### beforeedgedrag

Событие, предшествующее событию edgedrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;
- edgeModel — ссылка на модель перетаскиваемой метки;
- globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие edgedrag будет отменено.

### beforeedgedragstart

Событие, предшествующее событию edgedragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;
- edgeModel — ссылка на модель перетаскиваемой метки;
- globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.

Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие edgedragstart, будет отменено.

## beforevertexadd

Событие, предшествующее событию vertexadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;
- vertexIndex — индекс добавляемой вершины;
- globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах.

Имена методов, доступных через метод Event.callMethod:

- setGlobalPixels — метод позволяет скорректировать значения координат добавляемой вершины. В качестве аргумента принимает новые глобальные пиксельные координаты вершины в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexadd будет отменено.

**Пример:**

```
// Корректируем координаты событий добавления вершин так, чтобы они не выходили
за границы квадрата со стороной
// 100 пикселей и центром, совпадающим с центром карты.
polyline.editor.events.add(["beforevertexdraw", "beforevertexadd"], function
(event) {
    var mapGlobalPixelCenter = geoMap.getGlobalPixelCenter(),
        globalPixels = event.get("globalPixels"),
        pixelBounds = [
            [mapGlobalPixelCenter[0] - 100, mapGlobalPixelCenter[1] - 100],
            [mapGlobalPixelCenter[0] + 100, mapGlobalPixelCenter[1] + 100]
        ];
    event.callMethod("setGlobalPixels", [
        Math.max(Math.min(globalPixels[0], pixelBounds[1][0]), pixelBounds[0]
[0]),
        Math.max(Math.min(globalPixels[1], pixelBounds[1][1]), pixelBounds[0]
[1])
    ]);
});
```

## beforevertexdrag

Событие, предшествующее событию vertexdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;
- vertexModel — ссылка на модель перетаскиваемой вершины;
- globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexdrag будет отменено.

**Пример:**

```
// Инвертируем сдвиг при перетаскивании вершины.
polyline.editor.events.add("beforevertexdrag", function (event) {
    var pixelOffset = event.get("pixelOffset");
    event.callMethod("setPixelOffset", [-pixelOffset[0], -pixelOffset[1]]);
});
```

## beforevertexdragstart

Событие, предшествующее событию vertexdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;
- vertexModel — ссылка на модель перетаскиваемой вершины;
- globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.

Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие vertexdragstart, будет отменено.

## beforevertexdraw

Событие, предшествующее событию vertexdraw. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;
- vertexIndex — индекс добавляемой вершины;
- globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах.

Имена методов, доступных через метод Event.callMethod:

- setGlobalPixels — метод позволяет скорректировать значения координат добавляемой вершины. В качестве аргумента принимает новые глобальные пиксельные координаты вершины в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexdraw будет отменено.

## drawingstart

Включение режима добавления новых вершин. Экземпляр класса **Event**.

## drawingstop

Отключение режима добавления новых вершин. Экземпляр класса **Event**.

## edgedrag

Перетаскивание промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;
- edgeModel — ссылка на модель перетаскиваемой метки;
- globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.

## edgedragend

Завершение перетаскивания промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- edgeModel — ссылка на модель перетаскиваемой метки;
- globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.

## edgedragstart

Начало перетаскивания промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;
- edgeModel — ссылка на модель перетаскиваемой метки;
- globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.

## editingstart

Включение режима редактирования вершин. Экземпляр класса **Event**.

## editingstop

Отключение режима редактирования вершин. Экземпляр класса **Event**.

## framingstart

Включение режима масштабирования. Экземпляр класса **Event**.

## framingstop

Отключение режима масштабирования. Экземпляр класса **Event**.

## vertexadd

Добавление новой вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- parentModel — ссылка на родительскую, для добавленной вершины, модель данных;
- vertexIndex — индекс добавленной вершины;
- globalPixels — координаты добавленной вершины в глобальных пиксельных координатах.

## vertexdrag

Перетаскивание вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;
- vertexModel — ссылка на модель перетаскиваемой вершины;
- globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.

## vertexdragend

Завершение перетаскивания вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- vertexModel — ссылка на модель перетаскиваемой вершины;
- globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.

## vertexdragstart

Начало перетаскивания вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;
- vertexModel — ссылка на модель перетаскиваемой вершины;
- globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.

### vertexdraw

Рисование новой вершины. Данное событие, как правило, предшествует событию добавления вершины и возникает в момент движения мышью при включенном режиме добавления новых вершин. На основе данных, передаваемых в данном событии, отображаются направляющие линии в режиме добавления новых вершин. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;
- vertexIndex — индекс добавляемой вершины;
- globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах.

## Описание методов

### getModel

```
{vow.Promise} getModel()
```

**Возвращает** объект-обещание, который подтверждается объектом модели в момент ее фактического создания, либо отклоняется с одним из следующих кодов ошибки:

- Canceled — режим редактирования отключен до момента фактического создания модели;
- Editor wasn't started — режим редактирования не включен.

.

### getModelSync

```
{geometryEditor.model.RootLineString | null} getModelSync()
```

**Возвращает** модель данных редактора или null, если на момент вызова она отсутствует.

### getView

```
{vow.Promise} getView()
```

**Возвращает** объект-обещание, который подтверждается объектом отображения в момент его фактического создания, либо отклоняется с одним из следующих кодов ошибки:

- Canceled — режим редактирования отключен до момента фактического создания отображения;
- Editor wasn't started — режим редактирования не включен.

.

### getViewSync

```
{geometryEditor.view.Path | null} getViewSync()
```

**Возвращает** отображение редактора или null, если на момент вызова оно отсутствует.

### startDrawing

```
{vow.Promise} startDrawing()
```

Включает режим добавления новых вершин в ломаную линию. Включение происходит асинхронно.

**Возвращает** объект-обещание, который подтверждается в момент фактического старта режима добавления новых вершин, либо отклоняется с одним из следующих кодов ошибки:

- Canceled — режим добавления новых вершин отключен до момента его фактического старта.

.

### startEditing

```
{vow.Promise} startEditing()
```

Включает режим редактирования. Включение происходит асинхронно.

**Возвращает** объект-обещание, который подтверждается в момент фактического старта режима редактирования, либо отклоняется с одним из следующих кодов ошибки:

- Canceled — режим редактирования отключен до момента его фактического старта.

.

### startFraming

```
{vow.Promise} startFraming()
```

Включает режим масштабирования для ломаной линии. Включение происходит асинхронно.

**Возвращает** объект-обещание, который подтверждается в момент фактического старта режима масштабирования.

### stopDrawing

```
stopDrawing()
```

Отключает режим добавления новых вершин в ломаную линию.

### stopEditing

```
stopEditing()
```

Отключает режим редактирования.

### stopFraming

```
stopFraming()
```

Отключает режим масштабирования.

# geometryEditor.model

## geometryEditor.model.ChildLinearRing

---

**Примечание:**
Конструктор класса geometryEditor.model.ChildLinearRing скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет **geometryEditor.model.ChildLineString**.

Модель дочернего замкнутого контура. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Поля | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| editor | IGeometryEditor | Редактор геометрии.<br>Унаследовано от IGeometryEditorChildModel. |
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| geometry | IBaseGeometry | Геометрия модели.<br>Унаследовано от IGeometryEditorChildModel. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Деструктор.<br>Унаследован от IGeometryEditorModel. |
| getAllVerticesNumber() | Integer | Возвращает суммарное количество вершин редактируемой геометрии.<br>Унаследован от geometryEditor.model.ChildLineString. |
| getEdgeModels() | geometryEditor.model.Edge[] | Возвращает массив моделей промежуточных меток.<br>Унаследован от geometryEditor.model.ChildLineString. |
| getIndex() | Integer | Возвращает индекс дочерней ломаной в родительской модели.<br>Унаследован от geometryEditor.model.ChildLineString. |
| getParent() | IGeometryEditorModel | Возвращает родительскую модель данных.<br>Унаследован от IGeometryEditorChildModel. |
| getPixels() | Number[] | Возвращает пиксельные данные модели.<br>Унаследован от IGeometryEditorModel. |
| getVertexModels() | geometryEditor.model.ChildVertex[] | Возвращает массив моделей дочерних вершин.<br>Унаследован от geometryEditor.model.ChildLineString. |
| setIndex(index) | | Задает индекс дочерней ломаной в родительской модели.<br>Унаследован от geometryEditor.model.ChildLineString. |

| Имя | Возвращает | Описание |
|---|---|---|
| **setPixels**(pixels) | | Задает пиксельные данные модели.<br>Унаследован от **IGeometryEditorChildModel**. |
| **spliceVertices**(start, deleteCount) | Number[][] | Удаляет определенное число вершин ломаной начиная с заданного индекса. При этом на место удаляемых вершин могут быть добавлены новые. Координаты новых вершин в глобальных пикселях можно передавать дополнительными аргументами после параметра deleteCount.<br>Унаследован от **geometryEditor.model.ChildLineString**. |

# geometryEditor.model.ChildLineString

**Примечание:**
Конструктор класса geometryEditor.model.ChildLineString скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeometryEditorChildModel**.

Модель дочерней ломаной линии. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Поля | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **editor** | **IGeometryEditor** | Редактор геометрии.<br>Унаследовано от **IGeometryEditorChildModel**. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **geometry** | **IBaseGeometry** | Геометрия модели.<br>Унаследовано от **IGeometryEditorChildModel**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Деструктор.<br>Унаследован от **IGeometryEditorModel**. |
| **getAllVerticesNumber**() | Integer | Возвращает суммарное количество вершин редактируемой геометрии. |

| Имя | Возвращает | Описание |
|---|---|---|
| getEdgeModels() | geometryEditor.model.Edge[] | Возвращает массив моделей промежуточных меток. |
| getIndex() | Integer | Возвращает индекс дочерней ломаной в родительской модели. |
| getParent() | IGeometryEditorModel | Возвращает родительскую модель данных.<br>Унаследован от IGeometryEditorChildModel. |
| getPixels() | Number[] | Возвращает пиксельные данные модели.<br>Унаследован от IGeometryEditorModel. |
| getVertexModels() | geometryEditor.model.ChildVertex[] | Возвращает массив моделей дочерних вершин. |
| setIndex(index) | | Задает индекс дочерней ломаной в родительской модели. |
| setPixels(pixels) | | Задает пиксельные данные модели.<br>Унаследован от IGeometryEditorChildModel. |
| spliceVertices(start, deleteCount) | Number[][] | Удаляет определенное число вершин ломаной начиная с заданного индекса. При этом на место удаляемых вершин могут быть добавлены новые. Координаты новых вершин в глобальных пикселях можно передавать дополнительными аргументами после параметра deleteCount. |

## Описание методов

### getAllVerticesNumber

```
{Integer} getAllVerticesNumber()
```

**Возвращает** суммарное количество вершин редактируемой геометрии.

### getEdgeModels

```
{geometryEditor.model.Edge[]} getEdgeModels()
```

**Возвращает** массив моделей промежуточных меток.

### getIndex

```
{Integer} getIndex()
```

**Возвращает** индекс дочерней ломаной в родительской модели.

### getVertexModels

```
{geometryEditor.model.ChildVertex[]} getVertexModels()
```

**Возвращает** массив моделей дочерних вершин.

## setIndex

```
setIndex(index)
```

Задает индекс дочерней ломаной в родительской модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс дочерней вершины. |

* Обязательный параметр/опция.

## spliceVertices

```
{Number[][]} spliceVertices(start, deleteCount)
```

Удаляет определенное число вершин ломаной начиная с заданного индекса. При этом на место удаляемых вершин могут быть добавлены новые. Координаты новых вершин в глобальных пикселях можно передавать дополнительными аргументами после параметра deleteCount.

**Возвращает** массив координат удаленных вершин.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| start* | — | Тип: Integer<br><br>Индекс, начиная с которого производится удаление и добавление вершин. |
| deleteCount* | — | Тип: Integer<br><br>Количество удаляемых вершин. |

* Обязательный параметр/опция.

# geometryEditor.model.ChildVertex

**Примечание:**
Конструктор класса geometryEditor.model.ChildVertex скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeometryEditorChildModel**.

Модель дочерней вершины. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Поля | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| editor | IGeometryEditor | Редактор геометрии.<br>Унаследовано от IGeometryEditorChildModel. |
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| geometry | IBaseGeometry | Геометрия модели.<br>Унаследовано от IGeometryEditorChildModel. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Деструктор.<br>Унаследован от IGeometryEditorModel. |
| getAllVerticesNumber() | Integer | Возвращает суммарное количество вершин редактируемой геометрии. |
| getIndex() | Integer | Возвращает индекс дочерней вершины в родительской модели. |
| getNextVertex() | geometryEditor.model.ChildVertex \| null | Возвращает ссылку на модель следующей вершины. |
| getParent() | IGeometryEditorModel | Возвращает родительскую модель данных.<br>Унаследован от IGeometryEditorChildModel. |
| getPixels() | Number[] | Возвращает пиксельные данные модели.<br>Унаследован от IGeometryEditorModel. |
| getPrevVertex() | geometryEditor.model.ChildVertex \| null | Возвращает ссылку на модель предыдущей вершины. |
| setGlobalPixels(pixels) | | Задает глобальные пиксельные координаты вершины. |
| setIndex(index) | | Задает индекс дочерней вершины в родительской модели. |
| setNextVertex(nextVertex) | | Устанавливает ссылку на модель следующей вершины. |
| setPixels(pixels) | | Задает пиксельные данные модели.<br>Унаследован от IGeometryEditorChildModel. |
| setPrevVertex(prevVertex) | | Устанавливает ссылку на модель предыдущей вершины. |

## Описание методов

### getAllVerticesNumber

```
{Integer} getAllVerticesNumber()
```

**Возвращает** суммарное количество вершин редактируемой геометрии.

### getIndex

```
{Integer} getIndex()
```

**Возвращает** индекс дочерней вершины в родительской модели.

### getNextVertex

```
{geometryEditor.model.ChildVertex | null} getNextVertex()
```

**Возвращает** ссылку на модель следующей вершины.

### getPrevVertex

```
{geometryEditor.model.ChildVertex | null} getPrevVertex()
```

**Возвращает** ссылку на модель предыдущей вершины.

### setGlobalPixels

```
setGlobalPixels(pixels)
```

Задает глобальные пиксельные координаты вершины.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pixels* | — | Тип: Number[]<br><br>Глобальные пиксельные координаты вершины. |

* Обязательный параметр/опция.

### setIndex

```
setIndex(index)
```

Задает индекс дочерней вершины в родительской модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс дочерней вершины. |

* Обязательный параметр/опция.

### setNextVertex

```
setNextVertex(nextVertex)
```

Устанавливает ссылку на модель следующей вершины.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| nextVertex* | — | Тип: geometryEditor.model.ChildVertex \| null<br><br>Модель следующей вершины. |

* Обязательный параметр/опция.

### setPrevVertex

```
setPrevVertex(prevVertex)
```

Устанавливает ссылку на модель предыдущей вершины.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| prevVertex* | — | Тип: geometryEditor.model.ChildVertex \| null<br><br>Модель предыдущей вершины. |

* Обязательный параметр/опция.

## geometryEditor.model.Edge

**Примечание:**
Конструктор класса geometryEditor.model.Edge скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeometryEditorRootModel**.

Модель промежуточной метки. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Поля | Методы

### Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br><br>Унаследовано от IEventEmitter. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Деструктор. |

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IGeometryEditorModel**. |
| **getNextVertex**() | **geometryEditor.model.ChildVertex** \| null | Возвращает ссылку на модель следующей вершины. |
| **getPixels**() | Number[] | Возвращает пиксельные данные модели.<br>Унаследован от **IGeometryEditorModel**. |
| **getPrevVertex**() | **geometryEditor.model.ChildVertex** \| null | Возвращает ссылку на модель предыдущей вершины. |
| **setNextVertex**(nextVertex) | | Устанавливает ссылку на модель следующей вершины. |
| **setPrevVertex**(prevVertex) | | Устанавливает ссылку на модель предыдущей вершины. |

## Описание методов

### getNextVertex

```
{geometryEditor.model.ChildVertex | null} getNextVertex()
```

**Возвращает** ссылку на модель следующей вершины.

### getPrevVertex

```
{geometryEditor.model.ChildVertex | null} getPrevVertex()
```

**Возвращает** ссылку на модель предыдущей вершины.

### setNextVertex

```
setNextVertex(nextVertex)
```

Устанавливает ссылку на модель следующей вершины.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| nextVertex* | — | Тип: **geometryEditor.model.ChildVertex** \| null<br>Модель следующей вершины. |

* Обязательный параметр/опция.

### setPrevVertex

```
setPrevVertex(prevVertex)
```

Устанавливает ссылку на модель предыдущей вершины.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| prevVertex* | — | Тип: **geometryEditor.model.ChildVertex** \| null<br><br>Модель предыдущей вершины. |

* Обязательный параметр/опция.

# geometryEditor.model.EdgeGeometry

**Примечание:**
Конструктор класса geometryEditor.model.EdgeGeometry скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeometry**.

Геометрия промежуточной метки. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Поля | События | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **mapchange** | Сменилась карта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IGeometry**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **pixelgeometrychange** | Изменилась пиксельная геометрия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• pixelGeometry — новая пиксельная геометрия **IPixelGeometry**.<br><br>Унаследовано от **IGeometry**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.<br>Унаследован от **IGeometry**. |
| **getMap**() | **Map** \| null | Возвращает текущую карту.<br>Унаследован от **IGeometry**. |
| **getPixelGeometry**([options]) | **IPixelGeometry** | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.<br>Унаследован от **IGeometry**. |
| **getType**() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от **IBaseGeometry**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |

# geometryEditor.model.RootLineString

**Примечание:**
Конструктор класса geometryEditor.model.RootLineString скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeometryEditorRootModel**.

Модель корневой ломаной линии. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Поля | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Деструктор.<br>Унаследован от **IGeometryEditorModel**. |

| Имя | Возвращает | Описание |
|---|---|---|
| getAllVerticesNumber() | Integer | Возвращает суммарное количество вершин редактируемой геометрии. |
| getPixels() | Number[] | Возвращает пиксельные данные модели.<br><br>Унаследован от IGeometryEditorModel. |
| getVertexModels() | geometryEditor.model.ChildVertex[] | Возвращает массив моделей дочерних вершин. |
| spliceVertices(start, deleteCount) | Number[][] | Удаляет определенное число вершин ломаной начиная с заданного индекса. При этом на место удаляемых вершин могут быть добавлены новые. Координаты новых вершин в глобальных пикселях можно передавать дополнительными аргументами после параметра deleteCount. |

## Описание методов

### getAllVerticesNumber

```
{Integer} getAllVerticesNumber()
```

**Возвращает** суммарное количество вершин редактируемой геометрии.

### getVertexModels

```
{geometryEditor.model.ChildVertex[]} getVertexModels()
```

**Возвращает** массив моделей дочерних вершин.

### spliceVertices

```
{Number[][]} spliceVertices(start, deleteCount)
```

Удаляет определенное число вершин ломаной начиная с заданного индекса. При этом на место удаляемых вершин могут быть добавлены новые. Координаты новых вершин в глобальных пикселях можно передавать дополнительными аргументами после параметра deleteCount.

**Возвращает** массив координат удаленных вершин.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| start* | — | Тип: Integer<br><br>Индекс, начиная с которого производится удаление и добавление вершин. |
| deleteCount* | — | Тип: Integer<br><br>Количество удаляемых вершин. |

* Обязательный параметр/опция.

# geometryEditor.model.RootPolygon

**Примечание:**
Конструктор класса geometryEditor.model.RootPolygon скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeometryEditorRootModel**.

Модель корневого полигона. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Поля | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Деструктор.<br>Унаследован от **IGeometryEditorModel**. |
| **getAllVerticesNumber**() | Integer | Возвращает суммарное количество вершин редактируемой геометрии. |
| **getPathModels**() | **geometryEditor.model.ChildLinearRing**[] | Возвращает массив моделей дочерних контуров. |
| **getPixels**() | Number[] | Возвращает пиксельные данные модели.<br>Унаследован от **IGeometryEditorModel**. |
| **splicePaths**(start, deleteCount) | Number[][] | Удаляет определенное число контуров многоугольника начиная с заданного индекса. При этом на место удаляемых контуров могут быть добавлены новые. Координаты новых контуров в глобальных пикселях можно передавать дополнительными аргументами после параметра deleteCount. |

## Описание методов

### getAllVerticesNumber

```
{Integer} getAllVerticesNumber()
```

**Возвращает** суммарное количество вершин редактируемой геометрии.

## getPathModels

```
{geometryEditor.model.ChildLinearRing[]} getPathModels()
```

**Возвращает** массив моделей дочерних контуров.

## splicePaths

```
{Number[][]} splicePaths(start, deleteCount)
```

Удаляет определенное число контуров многоугольника начиная с заданного индекса. При этом на место удаляемых контуров могут быть добавлены новые. Координаты новых контуров в глобальных пикселях можно передавать дополнительными аргументами после параметра deleteCount.

**Возвращает** массив координат удаленных контуров.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| start* | — | Тип: Integer<br><br>Индекс, начиная с которого производится удаление и добавление контуров. |
| deleteCount* | — | Тип: Integer<br><br>Количество удаляемых контуров. |

* Обязательный параметр/опция.

# geometryEditor.Point

Расширяет **IGeometryEditor**.

Редактор геометрии "Точка".

Конструктор | Поля | События | Методы

## Конструктор

```
geometryEditor.Point(geometry[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPointGeometry**<br><br>Геометрия "Точка". |
| options | — | Тип: Object<br><br>Опции редактора геометрии. |
| options.dblClickHandler | — | Тип: Function<br><br>Обработчик двойного щелчка по вершине. Принимает ссылку на модель редактируемой вершины. По умолчанию, обработчик определен функцией, удаляющей соответствующую вершину. |
| options.drawingCursor | "arrow" | Тип: Boolean<br><br>Курсор мыши в режиме рисования. |
| options.drawOver | true | Тип: Boolean<br><br>Разрешает ставить точки поверх объектов карты в режиме рисования. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **geometry** | **IGeometry** | Редактируемая геометрия.<br><br>Унаследовано от **IGeometryEditor**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |
| **state** | **IDataManager** | Менеджер состояния редактора геометрии. |

| Имя | Тип | Описание |
|---|---|---|
| | | Поля данных, доступные посредством методов get и set:<br>• editing — определяет включен ли редактор. Тип — Boolean. Значение по умолчанию — false.<br>• drawing — определяет включен ли режим рисования. Тип — Boolean. Значение по умолчанию — false. |

## События

| Имя | Описание |
|---|---|
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **statechange** | Изменение состояния редактора геометрии. Экземпляр класса **Event**.<br>Унаследовано от **IGeometryEditor**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **startDrawing**() | **vow.Promise** | Включает режим рисования точки. |
| **startEditing**() | | Включает режим редактирования.<br>Унаследован от **IGeometryEditor**. |
| **stopDrawing**() | **vow.Promise** | Отключает режим рисования точки. |
| **stopEditing**() | | Отключает режим редактирования.<br>Унаследован от **IGeometryEditor**. |

## Описание полей

### state

```
{IDataManager} state
```

Менеджер состояния редактора геометрии.

Поля данных, доступные посредством методов get и set:

- editing — определяет включен ли редактор. Тип — Boolean. Значение по умолчанию — false.
- drawing — определяет включен ли режим рисования. Тип — Boolean. Значение по умолчанию — false.

## Описание методов

### startDrawing

```
{vow.Promise} startDrawing()
```

Включает режим рисования точки.

**Возвращает** объект-обещание.

### stopDrawing

```
{vow.Promise} stopDrawing()
```

Отключает режим рисования точки.

**Возвращает** объект-обещание.

# geometryEditor.Polygon

Расширяет **IGeometryEditor**.

Редактор геометрии "Многоугольник".

Конструктор | Поля | События | Методы

## Конструктор

```
geometryEditor.Polygon(geometry[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPolygonGeometry**<br><br>Геометрия "Многоугольник". |
| options | — | Тип: Object<br><br>Опции редактора геометрии. |
| options.dblClickHandler | — | Тип: Function<br><br>Обработчик двойного щелчка по вершине. Принимает ссылку на модель редактируемой вершины. По умолчанию, обработчик определен функцией, удаляющей соответствующую вершину. |
| options.drawingCursor | "arrow" | Тип: Boolean<br><br>Курсор мыши в режиме добавления новых вершин. |
| options.drawOver | true | Тип: Boolean<br><br>Разрешает ставить точки поверх объектов карты в режиме добавления новых вершин. |
| options.edgeLayout | — | Тип: Function<br><br>Класс макета для промежуточных меток. |
| options.maxPoints | Infinity | Тип: Number<br><br>Максимально допустимое число вершин в многоугольнике. |
| options.menuManager | — | Тип: Function |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Диспетчер контекстного меню, открывающегося при щелчке по вершине. Принимает два аргумента:<br>• массив объектов, описывающих пункты контекстного меню для этой вершины;<br>• ссылку на модель редактируемой вершины.<br>Диспетчер может изменять данные в переданном массиве. Должен возвращать массив объектов, описывающих пункты контекстного меню. |
| options.useMapMarginInDrawing | true | Тип: Boolean<br>Нужно ли учитывать отступы карты в режиме рисования. |
| options.vertexLayout | — | Тип: Function<br>Класс макета для меток на вершинах многоугольника. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **geometry** | **IGeometry** | Редактируемая геометрия.<br>Унаследовано от **IGeometryEditor**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **state** | **IDataManager** | Менеджер состояния редактора геометрии.<br>Поля данных, доступные посредством методов get и set:<br>• editing — определяет включен ли редактор. Тип — Boolean. Значение по умолчанию — false.<br>• drawing — определяет включен ли режим добавления новых вершин. Тип — Boolean. Значение по умолчанию — false.<br>• drawingFrom — определяет то, как добавляются новые точки в режиме добавления. Может принимать одно из двух строковых значений: "begin" — точки добавляются в начало многоугольника, "end"- точки добавляются в конец.<br>• drawingFromIndex — индекс вершины многоугольника, после которой будут добавляться новые точки в режиме добавления. Данное поле доступно только на запись, т.к. факт записи приводит к изменению порядка вершин в многоугольнике, так чтобы точка с заданным индексом стала последней по счету. Тип — целое число.<br>• drawingPath — индекс контура многоугольника, в который добавляются новые точки в режиме добавления. Тип — целое число. Значение по умолчанию — 0. |

## События

| Имя | Описание |
|---|---|
| **beforeedgedrag** | Событие, предшествующее событию edgedrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• edgeModel — ссылка на модель перетаскиваемой метки;<br>• globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие edgedrag будет отменено. |
| **beforeedgedragstart** | Событие, предшествующее событию edgedragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;<br>• edgeModel — ссылка на модель перетаскиваемой метки;<br>• globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие edgedragstart, будет отменено. |
| **beforevertexadd** | Событие, предшествующее событию vertexadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;<br>• vertexIndex — индекс добавляемой вершины;<br>• globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах.<br>Имена методов, доступных через метод Event.callMethod:<br>• setGlobalPixels — метод позволяет скорректировать значения координат добавляемой вершины. В качестве аргумента принимает новые глобальные пиксельные координаты вершины в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexadd будет отменено. |
| **beforevertexdrag** | Событие, предшествующее событию vertexdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• vertexModel — ссылка на модель перетаскиваемой вершины;<br>• globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах. |

| Имя | Описание |
|---|---|
| | Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexdrag будет отменено. |
| **beforevertexdragstart** | Событие, предшествующее событию vertexdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;<br>• vertexModel — ссылка на модель перетаскиваемой вершины;<br>• globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие vertexdragstart, будет отменено. |
| **beforevertexdraw** | Событие, предшествующее событию vertexdraw. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;<br>• vertexIndex — индекс добавляемой вершины;<br>• globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах.<br>Имена методов, доступных через метод Event.callMethod:<br>• setGlobalPixels — метод позволяет скорректировать значения координат добавляемой вершины. В качестве аргумента принимает новые глобальные пиксельные координаты вершины в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexdraw будет отменено. |
| **drawingstart** | Включение режима добавления новых вершин. Экземпляр класса **Event**. |
| **drawingstop** | Отключение режима добавления новых вершин. Экземпляр класса **Event**. |
| **edgedrag** | Перетаскивание промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• edgeModel — ссылка на модель перетаскиваемой метки;<br>• globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах. |
| **edgedragend** | Завершение перетаскивания промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• edgeModel — ссылка на модель перетаскиваемой метки;<br>• globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах. |

| Имя | Описание |
|---|---|
| **edgedragstart** | Начало перетаскивания промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;<br>• edgeModel — ссылка на модель перетаскиваемой метки;<br>• globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах. |
| **editingstart** | Включение режима редактирования вершин. Экземпляр класса **Event**. |
| **editingstop** | Отключение режима редактирования вершин. Экземпляр класса **Event**. |
| **framingstart** | Включение режима масштабирования. Экземпляр класса **Event**. |
| **framingstop** | Отключение режима масштабирования. Экземпляр класса **Event**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **statechange** | Изменение состояния редактора геометрии. Экземпляр класса **Event**.<br>Унаследовано от **IGeometryEditor**. |
| **vertexadd** | Добавление новой вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• parentModel — ссылка на родительскую, для добавленной вершины, модель данных;<br>• vertexIndex — индекс добавленной вершины;<br>• globalPixels — координаты добавленной вершины в глобальных пиксельных координатах. |
| **vertexdrag** | Перетаскивание вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• vertexModel — ссылка на модель перетаскиваемой вершины;<br>• globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах. |
| **vertexdragend** | Завершение перетаскивания вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• vertexModel — ссылка на модель перетаскиваемой вершины;<br>• globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах. |
| **vertexdragstart** | Начало перетаскивания вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;<br>• vertexModel — ссылка на модель перетаскиваемой вершины;<br>• globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах. |

| Имя | Описание |
| --- | --- |
| vertexdraw | Рисование новой вершины. Данное событие, как правило, предшествует событию добавления вершины и возникает в момент движения мышью при включенном режиме добавления новых вершин. На основе данных, передаваемых в данном событии, отображаются направляющие линии в режиме добавления новых вершин. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;<br>• vertexIndex — индекс добавляемой вершины;<br>• globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах. |

## Методы

| Имя | Возвращает | Описание |
| --- | --- | --- |
| getModel() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом модели в момент ее фактического создания, либо отклоняется с одним из следующих кодов ошибки:<br>• Canceled — режим редактирования отключен до момента фактического создания модели;<br>• Editor wasn't started — режим редактирования не включен.<br>. |
| getModelSync() | geometryEditor.model.RootPolygon \| null | Возвращает модель редактора или null, если на момент вызова она отсутствует. |
| getView() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом отображения в момент его фактического создания, либо отклоняется с одним из следующих кодов ошибки:<br>• Canceled — режим редактирования отключен до момента фактического создания отображения;<br>• Editor wasn't started — режим редактирования не включен.<br>. |
| getViewSync() | geometryEditor.view.MultiPath \| null | Возвращает отображение редактора или null, если на момент вызова оно отсутствует. |
| startDrawing() | vow.Promise | Включает режим добавления новых вершин в многоугольник. Включение происходит асинхронно. |
| startEditing() | vow.Promise | Включает режим добавления новых вершин в многоугольник. Включение происходит асинхронно. |
| startFraming() | vow.Promise | Включает режим масштабирования для многоугольника. Включение происходит асинхронно. |

| Имя | Возвращает | Описание |
|---|---|---|
| stopDrawing() | | Отключает режим добавления новых вершин в многоугольник. |
| stopEditing() | | Отключает режим добавления новых вершин в многоугольник. |
| stopFraming() | | Отключает режим масштабирования. |

## Описание полей

### state

```
{IDataManager} state
```

Менеджер состояния редактора геометрии.

Поля данных, доступные посредством методов get и set:

- editing — определяет включен ли редактор. Тип — Boolean. Значение по умолчанию — false.
- drawing — определяет включен ли режим добавления новых вершин. Тип — Boolean. Значение по умолчанию — false.
- drawingFrom — определяет то, как добавляются новые точки в режиме добавления. Может принимать одно из двух строковых значений: "begin" — точки добавляются в начало многоугольника, "end"- точки добавляются в конец.
- drawingFromIndex — индекс вершины многоугольника, после которой будут добавляться новые точки в режиме добавления. Данное поле доступно только на запись, т.к. факт записи приводит к изменению порядка вершин в многоугольнике, так чтобы точка с заданным индексом стала последней по счету. Тип — целое число.
- drawingPath — индекс контура многоугольника, в который добавляются новые точки в режиме добавления. Тип — целое число. Значение по умолчанию — 0.

## Описание событий

### beforeedgedrag

Событие, предшествующее событию edgedrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;
- edgeModel — ссылка на модель перетаскиваемой метки;
- globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие edgedrag будет отменено.

## beforeedgedragstart

Событие, предшествующее событию edgedragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;
- edgeModel — ссылка на модель перетаскиваемой метки;
- globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.

Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие edgedragstart, будет отменено.

## beforevertexadd

Событие, предшествующее событию vertexadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;
- vertexIndex — индекс добавляемой вершины;
- globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах.

Имена методов, доступных через метод Event.callMethod:

- setGlobalPixels — метод позволяет скорректировать значения координат добавляемой вершины. В качестве аргумента принимает новые глобальные пиксельные координаты вершины в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexadd будет отменено.

**Примеры:**

**1.**

```
// Запрещаем добавлять новые вершины на расстоянии более 100 пикселей от центра
карты.
polygon.editor.events.add("beforevertexadd", function (event) {
    var mapGlobalPixelCenter = geoMap.getGlobalPixelCenter(),
        globalPixels = event.get("globalPixels"),
        vector = [mapGlobalPixelCenter[0] - globalPixels[0],
mapGlobalPixelCenter[1] - globalPixels[1]],
        vectorLength = Math.sqrt(vector[0] * vector[0] + vector[1] * vector[1]);
    if (dist > 100) {
        event.preventDefault();
    }
});
```

**2.**

```
// Корректируем координаты событий добавления вершин так, чтобы они не выходили
за границы квадрата со стороной
// 100 пикселей и центром, совпадающим с центром карты.
polygon.editor.events.add(["beforevertexdraw", "beforevertexadd"], function
(event) {
    var mapGlobalPixelCenter = geoMap.getGlobalPixelCenter(),
        globalPixels = event.get("globalPixels"),
        pixelBounds = [
            [mapGlobalPixelCenter[0] - 100, mapGlobalPixelCenter[1] - 100],
            [mapGlobalPixelCenter[0] + 100, mapGlobalPixelCenter[1] + 100]
        ];
    event.callMethod("setGlobalPixels", [
        Math.max(Math.min(globalPixels[0], pixelBounds[1][0]), pixelBounds[0]
[0]),
        Math.max(Math.min(globalPixels[1], pixelBounds[1][1]), pixelBounds[0]
[1])
    ]);
});
```

## beforevertexdrag

Событие, предшествующее событию vertexdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;
- vertexModel — ссылка на модель перетаскиваемой вершины;
- globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexdrag будет отменено.

## beforevertexdragstart

Событие, предшествующее событию vertexdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;
- vertexModel — ссылка на модель перетаскиваемой вершины;
- globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.

Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие vertexdragstart, будет отменено.

**Пример:**

```
// Запрещаем перетаскивание вершины с индексом 0 в контуре с тем же индексом.
polygon.editor.events.add("beforevertexdragstart", function (event) {
    var vertexModel = event.get("target").properties.get("vertexModel"),
        pathModel = vertexModel.getParent();
    if (pathModel.getIndex() == 0 && vertexModel.getIndex() == 0) {
        event.preventDefault();
    }
});
```

## beforevertexdraw

Событие, предшествующее событию vertexdraw. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;
- vertexIndex — индекс добавляемой вершины;
- globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах.

Имена методов, доступных через метод Event.callMethod:

- setGlobalPixels — метод позволяет скорректировать значения координат добавляемой вершины. В качестве аргумента принимает новые глобальные пиксельные координаты вершины в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие vertexdraw будет отменено.

## drawingstart

Включение режима добавления новых вершин. Экземпляр класса **Event**.

## drawingstop

Отключение режима добавления новых вершин. Экземпляр класса **Event**.

## edgedrag

Перетаскивание промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;
- edgeModel — ссылка на модель перетаскиваемой метки;
- globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.

## edgedragend

Завершение перетаскивания промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- edgeModel — ссылка на модель перетаскиваемой метки;
- globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.

## edgedragstart

Начало перетаскивания промежуточной метки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;
- edgeModel — ссылка на модель перетаскиваемой метки;
- globalPixels — координаты перетаскиваемой метки в глобальных пиксельных координатах.

## editingstart

Включение режима редактирования вершин. Экземпляр класса **Event**.

## editingstop

Отключение режима редактирования вершин. Экземпляр класса **Event**.

### framingstart

Включение режима масштабирования. Экземпляр класса **Event**.

### framingstop

Отключение режима масштабирования. Экземпляр класса **Event**.

### vertexadd

Добавление новой вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- parentModel — ссылка на родительскую, для добавленной вершины, модель данных;
- vertexIndex — индекс добавленной вершины;
- globalPixels — координаты добавленной вершины в глобальных пиксельных координатах.

### vertexdrag

Перетаскивание вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;
- vertexModel — ссылка на модель перетаскиваемой вершины;
- globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.

### vertexdragend

Завершение перетаскивания вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- vertexModel — ссылка на модель перетаскиваемой вершины;
- globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.

### vertexdragstart

Начало перетаскивания вершины. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется;
- vertexModel — ссылка на модель перетаскиваемой вершины;
- globalPixels — координаты перетаскиваемой вершины в глобальных пиксельных координатах.

### vertexdraw

Рисование новой вершины. Данное событие, как правило, предшествует событию добавления вершины и возникает в момент движения мышью при включенном режиме добавления новых вершин. На основе данных, передаваемых в данном событии, отображаются направляющие линии в режиме добавления новых вершин. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- parentModel — ссылка на родительскую, для добавляемой вершины, модель данных;
- vertexIndex — индекс добавляемой вершины;
- globalPixels — координаты добавляемой вершины в глобальных пиксельных координатах.

## Описание методов

### getModel

```
{vow.Promise} getModel()
```

**Возвращает** объект-обещание, который подтверждается объектом модели в момент ее фактического создания, либо отклоняется с одним из следующих кодов ошибки:

- Canceled — режим редактирования отключен до момента фактического создания модели;
- Editor wasn't started — режим редактирования не включен.

.

## getModelSync

```
{geometryEditor.model.RootPolygon | null} getModelSync()
```

**Возвращает** модель редактора или null, если на момент вызова она отсутствует.

## getView

```
{vow.Promise} getView()
```

**Возвращает** объект-обещание, который подтверждается объектом отображения в момент его фактического создания, либо отклоняется с одним из следующих кодов ошибки:

- Canceled — режим редактирования отключен до момента фактического создания отображения;
- Editor wasn't started — режим редактирования не включен.

.

## getViewSync

```
{geometryEditor.view.MultiPath | null} getViewSync()
```

**Возвращает** отображение редактора или null, если на момент вызова оно отсутствует.

## startDrawing

```
{vow.Promise} startDrawing()
```

Включает режим добавления новых вершин в многоугольник. Включение происходит асинхронно.

**Возвращает** объект-обещание, который подтверждается в момент фактического старта режима добавления новых вершин, либо отклоняется с одним из следующих кодов ошибки:

- Canceled — режим добавления новых вершин отключен до момента его фактического старта.

.

## startEditing

```
{vow.Promise} startEditing()
```

Включает режим добавления новых вершин в многоугольник. Включение происходит асинхронно.

**Возвращает** объект-обещание, который подтверждается в момент фактического старта режима редактирования, либо отклоняется с одним из следующих кодов ошибки:

- Canceled — режим редактирования отключен до момента его фактического старта.

.

## startFraming

```
{vow.Promise} startFraming()
```

Включает режим масштабирования для многоугольника. Включение происходит асинхронно.

**Возвращает** объект-обещание, который подтверждается в момент фактического старта режима масштабирования.

### stopDrawing

```
stopDrawing()
```

Отключает режим добавления новых вершин в многоугольник.

### stopEditing

```
stopEditing()
```

Отключает режим добавления новых вершин в многоугольник.

### stopFraming

```
stopFraming()
```

Отключает режим масштабирования.

# geometryEditor.view

## geometryEditor.view.Edge

---

**Примечание:**
Конструктор класса geometryEditor.view.Edge скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Отображение промежуточной метки. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Методы

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getPlacemark() | GeoObject | Возвращает ссылку на геообъект промежуточной метки. В данных метки, под именем "model", доступна ссылка на модель данных промежуточной метки. |

### Описание методов

### getPlacemark

```
{GeoObject} getPlacemark()
```

Возвращает ссылку на геообъект промежуточной метки. В данных метки, под именем "model", доступна ссылка на модель данных промежуточной метки.

**Возвращает** геообъект промежуточной метки.

# geometryEditor.view.MultiPath

**Примечание:**
Конструктор класса geometryEditor.view.MultiPath скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Отображение множества контуров. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Методы

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getEdgePlacemarks() | GeoObjectCollection | Возвращает коллекцию промежуточных меток дочерних отображений контуров. |
| getPathViews() | geometryEditor.view.Path[] | Возвращает массив отображений дочерних контуров. |
| getVertexPlacemarks() | GeoObjectCollection | Возвращает коллекцию меток вершин дочерних отображений контуров. |

## Описание методов

### getEdgePlacemarks

```
{GeoObjectCollection} getEdgePlacemarks()
```

**Возвращает** коллекцию промежуточных меток дочерних отображений контуров.

### getPathViews

```
{geometryEditor.view.Path[]} getPathViews()
```

**Возвращает** массив отображений дочерних контуров.

### getVertexPlacemarks

```
{GeoObjectCollection} getVertexPlacemarks()
```

**Возвращает** коллекцию меток вершин дочерних отображений контуров.

# geometryEditor.view.Path

**Примечание:**
Конструктор класса geometryEditor.view.Path скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Отображение контура. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Методы

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getEdgePlacemarks() | GeoObjectCollection | Возвращает коллекцию промежуточных меток, входящих в контур. |
| getEdgeViews() | geometryEditor.view.Edge[] | Возвращает массив отображений промежуточных меток. |
| getVertexPlacemarks() | GeoObjectCollection | Возвращает коллекцию меток вершин, входящих в контур. |
| getVertexViews() | geometryEditor.view.Vertex[] | Возвращает массив отображений дочерних вершин. |

## Описание методов

### getEdgePlacemarks

```
{GeoObjectCollection} getEdgePlacemarks()
```

**Возвращает** коллекцию промежуточных меток, входящих в контур.

### getEdgeViews

```
{geometryEditor.view.Edge[]} getEdgeViews()
```

**Возвращает** массив отображений промежуточных меток.

### getVertexPlacemarks

```
{GeoObjectCollection} getVertexPlacemarks()
```

**Возвращает** коллекцию меток вершин, входящих в контур.

### getVertexViews

```
{geometryEditor.view.Vertex[]} getVertexViews()
```

**Возвращает** массив отображений дочерних вершин.

# geometryEditor.view.Vertex

---

**Примечание:**
Конструктор класса geometryEditor.view.Vertex скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Отображение вершины. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Методы

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getPlacemark() | GeoObject | Возвращает ссылку на геообъект метки, обозначающей вершину. В данных метки, под именем "model", доступна ссылка на модель данных вершины. |

## Описание методов

### getPlacemark

```
{GeoObject} getPlacemark()
```

Возвращает ссылку на геообъект метки, обозначающей вершину. В данных метки, под именем "model", доступна ссылка на модель данных вершины.

**Возвращает** геообъект метки, обозначающей вершину.

# geoObject

## geoObject.addon

### geoObject.addon.balloon

---

**Примечание:**
Конструктор класса geoObject.addon.balloon скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Статический объект.

Модуль, который предоставляет возможность использовать балун у геообъекта. Добавляет геообъекту (**GeoObject**) интерфейс **IBalloonOwner**. При подключении package.full (стандартный набор модулей) доступен по умолчанию. Если **GeoObject** подключается отдельно, то необходимо явно указать этот модуль в загрузчике. Если **geoObject.addon.balloon** подключается отдельно после создания **GeoObject**, то интерфейс **IBalloonOwner** не будет добавлен. Тогда для инициализации менеджера балуна нужно будет использовать метод geoObject.addon.balloon#get.

Методы

---

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

---

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **get**(geoObject) | **IPopupManager** | Возвращает менеджер балуна геообъекта. |

## Описание методов

### get

```
{IPopupManager} get(geoObject)
```

**Возвращает** менеджер балуна геообъекта.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geoObject* | — | Тип: **IGeoObject**<br><br>Геообъект |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.geoObject.addon.balloon.get(geoObject)
```

# geoObject.addon.editor

**Примечание:**
Конструктор класса geoObject.addon.editor скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Статический объект.

Модуль, который предоставляет возможность использовать редактор у геообъекта. Добавляет поле GeoObject#editor в **GeoObject**. При подключении package.full (стандартного набора модулей) доступен по умолчанию. Если **GeoObject** подключается отдельно, то необходимо явно указать этот модуль в загрузчике. Если **geoObject.addon.editor** подключается отдельно после создания **GeoObject**, то поле GeoObject#editor не будет добавлено. Тогда для редактора нужно будет использовать метод geoObject.addon.editor#get.

Методы

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **create**(geoObject) | | Функция, создающая у переданного геообъекта поле editor. |
| **get**(geoObject) | **IGeometryEditor** | Возвращает редактор геообъекта. |

## Описание методов

### create

```
create(geoObject)
```

Функция, создающая у переданного геообъекта поле editor.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geoObject* | — | Тип: **IGeoObject**<br><br>Геообъект. |

* Обязательный параметр/опция.

### get

```
{IGeometryEditor} get(geoObject)
```

**Возвращает** редактор геообъекта.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geoObject* | — | Тип: **IGeoObject**<br><br>Геообъект |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.geoObject.addon.editor.get(geoObject)
```

# geoObject.addon.hint

**Примечание:**
Конструктор класса geoObject.addon.hint скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Статический объект.

Модуль, который предоставляет возможность использовать хинт у геообъекта. Добавляет геообъекту (**GeoObject**) интерфейс **IHintOwner**. При подключении package.full (стандартный набор модулей) доступен по умолчанию. Если **GeoObject** подключается отдельно, то необходимо явно указать этот модуль в загрузчике. Если **geoObject.addon.hint** подключается отдельно после создания **GeoObject**, то интерфейс **IHintOwner** не будет добавлен. Тогда для инициализации менеджера балуна нужно будет использовать метод geoObject.addon.hint#get.

Методы

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get(geoObject) | **IPopupManager** | Возвращает менеджер хинта геообъекта. |

## Описание методов

### get

```
{IPopupManager} get(geoObject)
```

**Возвращает** менеджер хинта геообъекта.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geoObject* | — | Тип: **IGeoObject**<br><br>Геообъект |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.geoObject.addon.hint.get(geoObject)
```

# geoObject.Balloon

Расширяет **IBalloonManager**.

Менеджер балуна геообъекта. Позволяет управлять балуном геообъекта, открывать и скрывать его. Передает балуну данные в формате **IGeoObjectPopupData**. Использует внутри себя менеджер балуна карты **map.Balloon**. Геообъекты содержат экземпляр этого класса, доступный как myGeoObject.balloon. Не создавайте экземпляры этого класса без необходимости.

**См.: Balloon GeoObject.balloon**

Конструктор | Поля | События | Методы

## Конструктор

```
geoObject.Balloon(geoObject)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geoObject * | — | Тип: Object<br><br>Геообъект. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **autopanbegin** | Начало автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **autopanend** | Окончание автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **beforeuserclose** | Событие, предшествующее Balloon.userclose. Позволяет отменить пользовательское действие посредством вызова метода preventDefault. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |

| Имя | Описание |
|---|---|
| **userclose** | Закрытие балуна пользователем. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• target — ссылка на объект **IBalloonOwner**.<br>Унаследовано от **IBalloonManager**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **autoPan**() | **vow.Promise** | Перемещает карту таким образом, чтобы балун оказался видимым.<br>Унаследован от **IBalloonManager**. |
| **close**([force]) | **vow.Promise** | Закрывает инфо-объект.<br>Унаследован от **IPopupManager**. |
| **destroy**() | | Отключает менеджер инфо-объекта.<br>Унаследован от **IPopupManager**. |
| **getData**() | Object \| null | Возвращает данные инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **getOptions**() | **IOptionManager** \| null | Возвращает менеджер опций или `null`.<br>Унаследован от **IPopupManager**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от **IPopupManager**. |
| **getOverlaySync**() | **IOverlay** \| null | Возвращает оверлей, если тот существует.<br>Унаследован от **IPopupManager**. |
| **getPosition**() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **isOpen**() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от **IPopupManager**. |
| **open**([position[, data[, options]]]) | **vow.Promise** | Открывает балун геообъекта. |
| **setData**([data]) | **vow.Promise** | Устанавливает новые пользовательские данные. |
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopupManager**. |

## Описание методов

### open

```
{vow.Promise} open([position[, data[, options]]])
```

Открывает балун геообъекта.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position | — | Тип: Number[]<br><br>Координаты открытия. По умолчанию: точка на геообъекте, ближайшая к текущему центру карты. Проекцию координат можно указать в опциях, иначе используется проекция геообъекта. |
| data | — | Тип: Object<br><br>Данные, добавляемые в поле *userData* объекта данных, передаваемых балуну. |
| options | — | Тип: Object<br><br>Опции. |

### setData

```
{vow.Promise} setData([data])
```

Устанавливает новые пользовательские данные.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data | — | Тип: Object<br><br>Данные, добавляемые в поле *userData* объекта данных, передаваемых балуну. |

# geoObject.Hint

Расширяет **IHintManager**.

Менеджер хинта геообъекта. Позволяет управлять хинтом геообъекта, открывать и скрывать его. Передает хинту данные в формате **IGeoObjectPopupData**. Использует внутри себя менеджер хинтов карты **map.Hint**. Геообъекты содержат экземпляр этого класса, доступный как myGeoObject.hint. Не создавайте экземпляры этого класса без необходимости.

**См.: Hint GeoObject.hint**

Конструктор | Поля | Методы

## Конструктор

```
geoObject.Hint(geoObject)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geoObject * | — | Тип: Object<br><br>Геообъект. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **close**([force]) | **vow.Promise** | Закрывает инфо-объект.<br><br>Унаследован от **IPopupManager**. |
| **destroy**() | | Отключает менеджер инфо-объекта.<br><br>Унаследован от **IPopupManager**. |
| **getData**() | Object \| null | Возвращает данные инфо-объекта или `null`.<br><br>Унаследован от **IPopupManager**. |
| **getOptions**() | **IOptionManager** \| null | Возвращает менеджер опций или `null`.<br><br>Унаследован от **IPopupManager**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br><br>Унаследован от **IPopupManager**. |
| **getOverlaySync**() | **IOverlay** \| null | Возвращает оверлей, если тот существует.<br><br>Унаследован от **IPopupManager**. |
| **getPosition**() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br><br>Унаследован от **IPopupManager**. |
| **isOpen**() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br><br>Унаследован от **IPopupManager**. |
| **open**([position[, data[, options]]]) | **vow.Promise** | Открывает хинт геообъекта. |
| **setData**([data]) | **vow.Promise** | Устанавливает новые пользовательские данные. |

| Имя | Возвращает | Описание |
|---|---|---|
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopupManager**. |

## Описание методов

### open

```
{vow.Promise} open([position[, data[, options]]])
```

Открывает хинт геообъекта.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position | — | Тип: Number[]<br><br>Координаты открытия. По умолчанию: геометрический центр тяжести геообъекта. Проекцию координат можно указать в опциях, иначе используется проекция геообъекта. |
| data | — | Тип: Object<br><br>Данные, добавляемые в поле *userData* объекта данных, передаваемых хинту. |
| options | — | Тип: Object<br><br>Опции. |

### setData

```
{vow.Promise} setData([data])
```

Устанавливает новые пользовательские данные.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data | — | Тип: Object<br><br>Данные, добавляемые в поле *userData* объекта данных, передаваемых хинту. |

# geoObject.Sequence

Расширяет **IGeoObject**, **IGeoObjectSequence**.

Неизменяемая коллекция геообъектов. Позволяет группировать геообъекты для добавления на карту, задания опций и т.д. Коллекция также является геообъектом.

Конструктор | Поля | События | Методы

## Конструктор

```
geoObject.Sequence([feature[, options]])
```

Создает неизменяемую коллекцию геообъектов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| feature | — | Тип: Object<br><br>Описание коллекции. Функционально соответствует описанию геообъекта. Смотрите **GeoObject**. |
| feature.children | — | Тип: **IGeoObject**[]<br><br>Массив дочерних геообъектов. |
| feature.geometry | — | Тип: **IGeometry** \| Object<br><br>Геометрия коллекции. |
| feature.properties | — | Тип: **IDataManager** \| Object<br><br>Данные коллекции. |
| options | — | Тип: Object<br><br>Опции коллекции. Здесь можно задавать все опции описанные в классе **GeoObject**. Значения опций будут применяться как к самой коллекции, так и к ее дочерним объектам, если для них данные опции не определены. |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |
| **properties** | **IDataManager** | Данные геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **state** | **IDataManager** | Состояние геообъекта.<br><br>Унаследовано от **IGeoObject**. |

## События

| Имя | Описание |
|---|---|
| **boundschange** | Изменение координат географической области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса **Event**.<br><br>Унаследовано от **IGeoObjectSequence**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br><br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |

| Имя | Описание |
| --- | --- |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br><br>Поля данных:<br><br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br><br>Унаследовано от **IChild**. |
| **pixelboundschange** | Изменение пиксельных координат области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса **Event**. |

| Имя | Описание |
|---|---|
| | Унаследовано от **IGeoObjectSequence**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **each**(callback[, context]) | | Для каждого дочернего геообъекта, вызывает функцию-обработчик.<br>Унаследован от **IGeoObjectSequence**. |
| **get**(index) | **IGeoObject** | Возвращает дочерний геообъект с заданным индексом.<br>Унаследован от **IGeoObjectSequence**. |
| **getBounds**() | Number[][] \| null | Возвращает географические координаты области, охватывающей коллекции и ее дочерние геообъекты.<br>Унаследован от **IGeoObjectSequence**. |
| **getIterator**() | **IIterator** | Возвращает итератор по дочерним геообъектам коллекции.<br>Унаследован от **IGeoObjectSequence**. |
| **getLength**() | Integer | Возвращает длину коллекции.<br>Унаследован от **IGeoObjectSequence**. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |

| Имя | Возвращает | Описание |
|---|---|---|
| **getPixelBounds**() | Number[][] \| null | Возвращает глобальные пиксельные координаты области, охватывающей коллекцию и ее дочерние геообъекты.<br><br>Унаследован от **IGeoObjectSequence**. |
| **indexOf**(object) | Integer | Возвращает индекс дочернего геообъекта. Если геообъект в коллекции не найден, возвращается -1.<br><br>Унаследован от **IGeoObjectSequence**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br><br>Унаследован от **IChildOnMap**. |

# GeoObject

Расширяет **IGeoObject**.

Геообъект. В зависимости от типа геометрии может отображаться как метка, линия, полигон и т.д. Вы также можете использовать вспомогательные классы для упрощенного создания геообъектов с определенным типом геометрии.

**См.: Placemark Polyline Polygon Circle Rectangle**

Конструктор | Поля | События | Методы

## Конструктор

```
GeoObject([feature[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| feature | — | Тип: Object<br><br>Описание геообъекта. |
| feature.geometry | — | Тип: **IGeometry** \| **IGeometryJson**<br><br>Геометрия геообъекта. Может задаваться непосредственно объектом, либо в виде объекта JSON-представления геометрии. Примеры классов и объектов, которые могут представлять геометрию — **geometry.Point**, geometry.json.point. |
| feature.properties | — | Тип: **IDataManager** \| Object<br><br>Данные геообъекта. Могут задаваться как экземпляр класса, реализующего интерфейс **IDataManager**, либо в виде хэша. При значениях опций по умолчанию геообъектом трактуются следующие поля данных:<br>• iconContent — содержимое иконки геообъекта;<br>• iconCaption — подпись иконки геообъекта;<br>• hintContent — содержимое всплывающей подсказки геообъекта;<br>• balloonContent — содержимое балуна геообъекта;<br>• balloonContentHeader — содержимое заголовка балуна геообъекта;<br>• balloonContentBody — содержимое основной части балуна геообъекта;<br>• balloonContentFooter — содержимое нижней части балуна геообъекта. |

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Поле balloonContent является кратким обозначением для поля balloonContentBody, но при одновременном задании balloonContentBody более приоритетен. Также вы можете дополнить данные геообъекта своими собственными полями и использовать их везде, где это возможно. Например в макете метки или макете балуна. |
| options | — | Тип: Object<br><br>Опции геообъекта. Используя этот параметр можно задавать как опции самого геообъекта, так и опции его составных частей:<br>• Опции балуна геообъекта с префиксом balloon.<br>• Опции всплывающей подсказки геообъекта с префиксом hint.<br>• Опции редактора геометрии геообъекта с префиксом editor. Тип редактора и перечень доступных опций зависит от типа геометрии геообъекта. Смотрите описание классов **geometryEditor.LineString**, **geometryEditor.Polygon**, **geometryEditor.Point**.<br>• Опции геометрии могут задаваться без префикса. Смотрите описание классов **IGeometry** геометрий **geometry.Point**, **geometry.Polygon** и др. |
| options.circleOverlay | "default#circle" | Тип: String \| Function<br><br>Ключ-идентификатор из хранилища **overlay.storage** или класс оверлея. Функция-генератор принимает три параметра:<br>• geometry: **IPixelCircleGeometry** — непосредственно пиксельная геометрия;<br>• data: Object — данные оверлея;<br>• options: Object — опции оверлея.<br><br>И возвращает **vow.Promise**. |
| options.cursor | "pointer" | Тип: String<br><br>Вид курсора над геообъектом. |
| options.draggable | false | Тип: Boolean<br><br>Определяет возможность перетаскивания геообъекта. |
| options.fill | true | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Наличие заливки фигуры. |
| options.fillColor | "0066ff99" | Тип: String<br><br>Цвет заливки. |
| options.fillImageHref | — | Тип: String<br><br>Фоновое изображение. При включении данной опции в режиме заливки **stretch** значение fillColor игнорируется. |
| options.fillMethod | 'stretch' | Тип: String<br><br>Тип заливки фоном. Может принимать одно из двух значений:<br>• stretch — Фоновое изображение растягивается по размеру оверлея.<br>• tile — Фоновое изображение повторяется без изменений размера. Аналог background-repeat в css. Можно использовать для заливки фигуры неким шаблоном. |
| options.fillOpacity | 1 | Тип: Number<br><br>Прозрачность заливки. |
| options.hasBalloon | true | Тип: Boolean<br><br>Определяет наличие поля balloon у геообъекта. |
| options.hasHint | true | Тип: Boolean<br><br>Определяет наличие поля hint у геообъекта. |
| options.hideIconOnBalloonOpen | true | Тип: Boolean<br><br>Скрывать иконку при открытии балуна. |
| options.iconCaptionMaxWidth | 188 | Тип: Number<br><br>Максимальная ширина подписи метки. |
| options.iconColor | — | Тип: String<br><br>Цвет иконки. Эта опция применяется для стандартных иконок в браузерах, поддерживающих SVG. |
| options.iconContentLayout | — | Тип: Function \| String<br><br>Макет содержимого иконки. (Тип: конструктор объекта с интерфейсом ILayout или его ключ в хранилище). |
| options.iconContentOffset | — | Тип: Number[] |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Пиксельный сдвиг содержимого иконки относительно родительского элемента. Используется в макете default#imageWithContent. |
| options.iconContentPadding | — | Тип: Number[]<br><br>Отступ для содержимого в иконке. |
| options.iconContentSize | — | Тип: Number[]<br><br>Размер содержимого. Используется в макете default#imageWithContent. |
| options.iconImageClipRect | [[0, 0], [{imageWidth}, {imageHeight}]] | Тип: Number[][]<br><br>Прямоугольная область (указывается левый верхний и правый нижний углы), которая будет вырезана из исходного графического файла и смасштабирована под размеры иконки (например для использования спрайтов). По умолчанию исходная картинка подставляется полностью. |
| options.iconImageHref | — | Тип: String<br><br>URL графического файла иконки. Применяется только в комбинации с макетами (iconLayout) 'default#image' и 'default#imageWithContent'. |
| options.iconImageOffset | — | Тип: Number[]<br><br>Пиксельный сдвиг изображения иконки внутри родительского элемента. |
| options.iconImageShape | — | Тип: **IShape** \| null<br><br>Фигура активной области. Если не задана, то автоматически будет рассчитана прямоугольная фигура на основе размера и смещения картинки. Координаты геометрии фигуры отсчитываются от точки привязки. Применяется только в комбинации с макетами (iconLayout) 'default#image' и 'default#imageWithContent'. |
| options.iconImageSize | — | Тип: Number[]<br><br>Размер иконки в пикселях. |
| options.iconLayout | — | Тип: Function \| String<br><br>Макет иконки. (Тип: конструктор объекта с интерфейсом ILayout или его ключ в хранилище). |
| options.iconMaxHeight | — | Тип: Number |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Максимальная высота иконки с содержимым. |
| options.iconMaxWidth | — | Тип: Number<br><br>Максимальная ширина иконки с содержимым. |
| options.iconOffset | [0, 0] | Тип: Number[]<br><br>Пиксельное смещение левого верхнего угла иконки относительно географической точки привязки метки. |
| options.iconShadow | false | Тип: Boolean<br><br>Флаг наличия тени у иконки. |
| options.iconShadowImageClipRect | [[0, 0], [{imageWidth}, {imageHeight}]] | Тип: Number[][]<br><br>Прямоугольная область (указывается левый верхний и правый нижний углы), которая будет вырезана из исходного графического файла и смасштабирована под размеры тени (например для использования спрайтов). По умолчанию исходная картинка подставляется полностью. |
| options.iconShadowImageHref | — | Тип: String<br><br>URL графического файла тени иконки. Применяется только в комбинации с макетами (iconShadowLayout) 'default#image' и 'default#imageWithContent'. |
| options.iconShadowImageOffset | — | Тип: Number[]<br><br>Пиксельный сдвиг изображения тени иконки внутри родительского элемента. |
| options.iconShadowImageSize | — | Тип: Number[]<br><br>Размер тени иконки. |
| options.iconShadowLayout | — | Тип: Function \| String<br><br>Макет тени иконки. (Тип: конструктор объекта с интерфейсом ILayout или его ключ в хранилище). |
| options.iconShadowOffset | — | Тип: Number[]<br><br>Пиксельное смещение тени иконки относительно заданной позиции. |
| options.interactiveZIndex | — | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Включает режим автоматического изменения z-index геообъекта в зависимости от его состояния. По умолчанию принимает значение true для геометрии **geometry.Point** и false для других геометрий. |
| options.interactivityModel | "default#geoObject" | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.lineStringOverlay | "default#polyline" | Тип: String \| Function<br><br>Ключ-идентификатор из хранилища **overlay.storage** или класс оверлея. Функция-генератор принимает три параметра:<br>• geometry: **IPixelLineStringGeometry** — непосредственно пиксельная геометрия;<br>• data: **IDataManager** или Object — данные оверлея;<br>• options: Object — опции оверлея.<br><br>И возвращает **vow.Promise**. |
| options.opacity | 1 | Тип: Number<br><br>Прозрачность. |
| options.openBalloonOnClick | true | Тип: Boolean<br><br>Определяет показывать ли балун при щелчке на геообъекте. |
| options.openEmptyBalloon | false | Тип: Boolean<br><br>Определяет показывать ли пустой балун при щелчке на геообъекте. |
| options.openEmptyHint | false | Тип: Boolean<br><br>Определяет показывать ли пустой хинт при наведении указателя мыши на геообъект. |
| options.openHintOnHover | true | Тип: Boolean<br><br>Определяет показывать ли хинт при наведении указателя мыши на геообъект. |
| options.outline | true | Тип: Boolean<br><br>Наличие обводки фигуры. |
| options.pane | — | Тип: String |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Ключ пейна, в который помещается оверлей геообъекта. Значение по умолчанию определяется текущим оверлеем. |
| options.pointOverlay | "default#placemark" | Тип: String \| Function<br><br>Ключ-идентификатор из хранилища **overlay.storage** или класс оверлея. Функция-генератор принимает три параметра:<br>• geometry: **IPixelPointGeometry** — непосредственно пиксельная геометрия;<br>• data: **IDataManager** или Object — данные оверлея;<br>• options: Object — опции оверлея.<br><br>И возвращает **vow.Promise**. |
| options.polygonOverlay | "default#polygon" | Тип: String \| Function<br><br>Ключ-идентификатор из хранилища **overlay.storage** или класс оверлея. Функция-генератор принимает три параметра:<br>• geometry: **IPixelPolygonGeometry** — непосредственно пиксельная геометрия;<br>• data: **IDataManager** или Object — данные оверлея;<br>• options: Object — опции оверлея.<br><br>И возвращает **vow.Promise**. |
| options.preset | — | Тип: String<br><br>Ключ предустановленных опций геообъекта. Список ключей содержится в описании **option.presetStorage**. |
| options.rectangleOverlay | "default#rectangle" | Тип: String \| Function<br><br>Ключ-идентификатор из хранилища **overlay.storage** или класс оверлея. Функция-генератор принимает три параметра:<br>• geometry: **IPixelRectangleGeometry** — непосредственно пиксельная геометрия;<br>• data: **IDataManager** или Object — данные оверлея;<br>• options: Object — опции оверлея.<br><br>И возвращает **vow.Promise**. |
| options.strokeColor | "0066ffff" | Тип: String \| String[] |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Цвет линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeOpacity | 1 | Тип: Number \| Number[]<br><br>Прозрачность линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeStyle | — | Тип: String \| Object \| String[] \| Object[]<br><br>Стиль линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeWidth | 1 | Тип: Number \| Number[]<br><br>Толщина линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.syncOverlayInit | false | Тип: Boolean<br><br>Включает синхронное добавление оверлея на карту. По умолчанию добавление оверлея осуществляется асинхронно, что позволяет предотвратить зависания браузера при добавлении на карту большого числа геообъектов. Однако, асинхронное добавление не позволяет получать доступ к оверлею сразу после добавления геообъекта на карту. |
| options.useMapMarginInDragging | true | Тип: Boolean<br><br>При перетаскивании объекта к краю карты происходит автоматическое изменение центра карты. Нужно ли учитывать отступы карты при автоматическом смещении центра карты **map.margin.Manager**. |
| options.visible | true | Тип: Boolean<br><br>Определяет видимость геообъекта. |
| options.zIndex | — | Тип: Number<br><br>z-index геообъекта в обычном состоянии. Наименее приоритетный. |
| options.zIndexActive | — | Тип: Number<br><br>z-index геообъекта с открытым балуном. Наиболее приоритетный. |
| options.zIndexDrag | — | Тип: Number<br><br>z-index геообъекта при перетаскивании. |
| options.zIndexHover | — | Тип: Number |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | z-index геообъекта при наведении на него указателя мыши. |

**Примеры:**

**1.**

```
var myGeoObject = new ymaps.GeoObject({
        // Описываем геометрию типа "Ломаная линия".
        geometry: {
            type: "LineString",
            coordinates: [
                [37.61, 55.75], [13.38, 52.51]
            ]
        },
        // Описываем данные геообъекта.
        properties: {
            hintContent: "Москва-Берлин"
        }
    }, {
        // Включаем отображение в форме геодезических кривых.
        geodesic: true,
        // Задаем ширину в 5 пикселей.
        strokeWidth: 5,
        // Задаем цвет линии.
        strokeColor: "#F008"
    });
// Добавляем геообъект на карту.
geoMap.geoObjects.add(myGeoObject);
```

**2.**

```
var myGeoObject = new ymaps.GeoObject({
        // Описываем геометрию типа "Точка".
        geometry: {
            type: "Point",
            coordinates: [37.61, 55.75]
        },
        // Описываем данные геообъекта.
        properties: {
            hintContent: "Москва",
            balloonContentHeader: "Москва",
            balloonContentBody: "Столица России",
            population: 11848762
        }
    }, {
        // Задаем пресет метки с точкой без содержимого.
        preset: "islands#redDotIcon",
        // Включаем возможность перетаскивания.
        draggable: true,
        // Переопределяем макет содержимого нижней части балуна.
        balloonContentFooterLayout: ymaps.templateLayoutFactory.createClass(
            'население: {{ properties.population }}, координаты:
{{ geometry.coordinates }}'
        ),
        // Отключаем задержку закрытия всплывающей подсказки.
        hintCloseTimeout: null
    });
// Добавляем геообъект на карту.
geoMap.geoObjects.add(myGeoObject);
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| balloon | geoObject.Balloon | Балун геообъекта. |
| editor | IGeometryEditor | Редактор геометрии геообъекта. |
| events | event.Manager | Менеджер событий. |
| geometry | IGeometry \| null | Геометрия геообъекта. |
| hint | geoObject.Hint | Хинт геообъекта. |
| options | option.Manager | Менеджер опций геообъекта. |
| properties | data.Manager | Менеджер данных геообъекта. |
| state | data.Manager | Состояние геообъекта. Определяется следующими полями:<br>• active: Boolean — признак того, что на геообъекте открыт балун;<br>• hover: Boolean — признак того, что в данный момент на геообъект наведен указатель мыши;<br>• drag: Boolean — признак того, что геообъект перетаскивается. |

## События

| Имя | Описание |
|---|---|
| balloonclose | Закрытие балуна. Экземпляр класса Event. |
| balloonopen | Открытие балуна на геообъекте. Экземпляр класса Event. |
| beforedrag | Событие, предшествующее событию drag. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие drag будет отменено. |
| beforedragstart | Событие, предшествующее событию dragstart. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие dragstart, будет отменено. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **drag** | Перетаскивание геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **dragend** | Завершение перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **dragstart** | Начало перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **editorstatechange** | Изменение состояния редактора геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• originalEvent — оригинальное событие редактора геометрии. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br><br>Унаследовано от **IGeoObject**. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**. |
| **hintopen** | Открытие хинта на геообъекте. Экземпляр класса **Event**. |
| **mapchange** | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IParentOnMap**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание полей

### balloon

```
{geoObject.Balloon} balloon
```

Балун геообъекта.

### editor

```
{IGeometryEditor} editor
```

Редактор геометрии геообъекта.

**Пример:**

```
// Начало редактирования геометрии геообъекта.
geoObject.editor.startEditing();
// ...
// Завершение редактирования.
geoObject.editor.stopEditing();
```

### events

```
{event.Manager} events
```

Менеджер событий.

### geometry

```
{IGeometry | null} geometry
```

Геометрия геообъекта.

**Пример:**

```
// При изменении координат геометрии геообъекта устанавливаем карте границы так,
// чтобы охватить геообъект целиком.
myMap.geoObjects.add(myGeoObject);
myGeoObject.geometry.events.add("change", function () {
    myMap.setBounds(myGeoObject.geometry.getBounds());
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### hint

```
{geoObject.Hint} hint
```

Хинт геообъекта.

### options

```
{option.Manager} options
```

Менеджер опций геообъекта.

### properties

```
{data.Manager} properties
```

Менеджер данных геообъекта.

**Пример:**

```
// При изменении данных выводим пользовательские идентификатор,
// если объект у объекта поле synchronized ложно.
myGeoObject.properties.events.add("change", function () {
    if (!myGeoObject.properties.get("synchronized")) {
        console.log(myGeoObject.properties.get("myID"));
    }
});
```

### state

```
{data.Manager} state
```

Состояние геообъекта. Определяется следующими полями:

- active: Boolean — признак того, что на геообъекте открыт балун;
- hover: Boolean — признак того, что в данный момент на геообъект наведен указатель мыши;
- drag: Boolean — признак того, что геообъект перетаскивается.

## Описание событий

### balloonclose

Закрытие балуна. Экземпляр класса **Event**.

### balloonopen

Открытие балуна на геообъекте. Экземпляр класса **Event**.

### beforedrag

Событие, предшествующее событию drag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- position — координаты относительно документа. Массив вида [pageX, pageY];
- pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие drag будет отменено.

**Пример:**

```
// Разрешаем перетаскивание геообъекта только вдоль горизонтальной оси.
geoObject.events.add("beforedrag", function (event) {
    var originalOffset = event.get("pixelOffset");
    event.callMethod("setPixelOffset", [originalOffset[0], 0]);
});
```

## beforedragstart

Событие, предшествующее событию dragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- position — координаты относительно документа. Массив вида [pageX, pageY];
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие dragstart, будет отменено.

## drag

Перетаскивание геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- position — координаты относительно документа. Массив вида [pageX, pageY];
- pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## dragend

Завершение перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- position — координаты относительно документа. Массив вида [pageX, pageY];
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## dragstart

Начало перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- position — координаты относительно документа. Массив вида [pageX, pageY];
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## editorstatechange

Изменение состояния редактора геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- originalEvent — оригинальное событие редактора геометрии.

## hintclose

Закрытие хинта. Экземпляр класса **Event**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### hintopen

Открытие хинта на геообъекте. Экземпляр класса **Event**.

## GeoObjectCollection

Расширяет **IGeoObject**, **IGeoObjectCollection**.

Коллекция геообъектов. Позволяет группировать геообъекты для добавления на карту, задания опций и т.д. Коллекция также является геообъектом.

Конструктор | Поля | События | Методы

### Конструктор

```
GeoObjectCollection([feature[, options]])
```

Создает коллекцию геообъектов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| feature | — | Тип: Object<br><br>Описание коллекции. Функционально соответствует описанию геообъекта. Смотрите **GeoObject**. |
| feature.children | — | Тип: **IGeoObject**[]<br><br>Массив дочерних геообъектов. |
| feature.geometry | — | Тип: **IGeometry** \| Object<br><br>Геометрия коллекции. |
| feature.properties | — | Тип: **IDataManager** \| Object<br><br>Данные коллекции. |
| options | — | Тип: Object<br><br>Опции коллекции. Здесь можно задавать все опции описанные в классе **GeoObject**. Значения опций будут применяться как к самой коллекции, так и к ее дочерним объектам, если для них данные опции не определены. |

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Создаем коллекцию геообъектов и задаем опции.
var myGeoObjects = new ymaps.GeoObjectCollection({}, {
    preset: "islands#redCircleIcon",
    strokeWidth: 4,
    geodesic: true
});

// Добавляем в коллекцию метки и линию.
myGeoObjects.add(new ymaps.Placemark([13.38, 52.51]));
myGeoObjects.add(new ymaps.Placemark([30.30, 50.27]));
myGeoObjects.add(new ymaps.Polyline([[13.38, 52.51], [30.30, 50.27]]));

// Добавляем коллекцию на карту.
myMap.geoObjects.add(myGeoObjects);
// Устанавливаем карте центр и масштаб так, чтобы охватить коллекцию целиком.
myMap.setBounds(myGeoObjects.getBounds());
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **properties** | **IDataManager** | Данные геообъекта.<br>Унаследовано от **IGeoObject**. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |

## События

| Имя | Описание |
|---|---|
| **add** | Добавлен (вставлен) дочерний геообъект. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• index: Integer — индекс добавленного геообъекта;<br>• child: **IGeoObject** — ссылка на добавленный геообъект.<br>Унаследовано от **IGeoObjectCollection**. |
| **boundschange** | Изменение координат географической области, охватывающей коллекцию и все ее дочерние геообъекты. Экземпляр класса **Event**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br><br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **pixelboundschange** | Изменение пиксельных координат области, охватывающей коллекцию и все ее дочерние геообъекты. Экземпляр класса **Event**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **remove** | Удален дочерний геообъект. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• index: Integer — индекс удаленного геообъекта;<br>• child: **IGeoObject** — ссылка на удаленный геообъект.<br>Унаследовано от **IGeoObjectCollection**. |
| **set** | В коллекцию записан новый дочерний геообъект. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• index: Integer — индекс геообъекта;<br>• child: **IGeoObject** — ссылка на новый геообъект;<br>• prevChild: **IGeoObject** — ссылка на предыдущее значение для данного индекса.<br>Унаследовано от **IGeoObjectCollection**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(child) | **GeoObjectCollection** | Добавляет геообъект в коллекцию. |
| **each**(callback[, context]) | | Перебирает все элементы коллекции, вызывая для каждого функцию-обработчик. |
| **get**(index) | **IGeoObject** | Возвращает дочерний геообъект с заданным индексом.<br>Унаследован от **IGeoObjectCollection**. |
| **getBounds**() | Number[][] \| null | Возвращает географические координаты области, охватывающей коллекцию и все ее дочерние геообъекты. |
| **getIterator**() | **IIterator** | Возвращает итератор по коллекции. |
| **getLength**() | Integer | Возвращает количество геообъектов в коллекции. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getPixelBounds**() | Number[][] \| null | Возвращает глобальные пиксельные координаты области, охватывающей коллекцию и все ее дочерние геообъекты. |
| **indexOf**(object) | Integer | Возвращает индекс дочернего геообъекта. Если геообъект в коллекции не найден, возвращается -1.<br>Унаследован от **IGeoObjectCollection**. |
| **remove**(child) | **GeoObjectCollection** | Удаляет геообъект из коллекции. |
| **removeAll**() | **GeoObjectCollection** | Удаляет из коллекции все геообъекты. |
| **set**(index, child) | **GeoObjectCollection** | Записывает в коллекцию новый дочерний геообъект. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |
| **splice**(index, number) | **GeoObjectCollection** | Удаляет геообъекты из коллекции. Если необходимо, вставляет на их место другие. Объекты, которые будут вставлены вместо удаляемых, передаются как дополнительные параметры (после параметра number). |
| **toArray**() | **IGeoObject**[] | Возвращает массив, содержащий все геообъекты коллекции на момент вызова. |

## Описание событий

### boundschange

Изменение координат географической области, охватывающей коллекцию и все ее дочерние геообъекты. Экземпляр класса **Event**.

### pixelboundschange

Изменение пиксельных координат области, охватывающей коллекцию и все ее дочерние геообъекты. Экземпляр класса **Event**.

## Описание методов

### add

```
{GeoObjectCollection} add(child)
```

Добавляет геообъект в коллекцию.

**Возвращает** ссылку на коллекцию.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **IGeoObject**<br><br>Дочерний геообъект. |

* Обязательный параметр/опция.

## each

```
each(callback[, context])
```

Перебирает все элементы коллекции, вызывая для каждого функцию-обработчик.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик. |
| context | — | Тип: Object<br><br>Контекст выполнения функции. |

* Обязательный параметр/опция.

## getBounds

```
{Number[][] | null} getBounds()
```

**Возвращает** географические координаты области, охватывающей коллекцию и все ее дочерние геообъекты.

**Пример:**

```
// Устанавливаем центр и масштаб карты так, чтобы отобразить всю коллекцию
целиком.
myMap.setBounds(myCollection.getBounds());
```

## getIterator

```
{IIterator} getIterator()
```

**Возвращает** итератор по коллекции.

**Пример:**

```
// Ищем в коллекции геообъект с геометрией "Ломаная линия".
var iterator = myGroup.getIterator(),
    object;
while ((object = iterator.getNext()) != iterator.STOP_ITERATION) {
    if (object.geometry.getType() == "LineString") {
        break;
    }
}
```

## getLength

```
{Integer} getLength()
```

**Возвращает** количество геообъектов в коллекции.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getPixelBounds

```
{Number[][] | null} getPixelBounds()
```

**Возвращает** глобальные пиксельные координаты области, охватывающей коллекцию и все ее дочерние геообъекты.

## remove

```
{GeoObjectCollection} remove(child)
```

Удаляет геообъект из коллекции.

**Возвращает** ссылку на коллекцию.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **IGeoObject**<br><br>Дочерний геообъект. |

* Обязательный параметр/опция.

## removeAll

```
{GeoObjectCollection} removeAll()
```

Удаляет из коллекции все геообъекты.

**Возвращает** ссылку на коллекцию.

## set

```
{GeoObjectCollection} set(index, child)
```

Записывает в коллекцию новый дочерний геообъект.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс. |
| child* | — | Тип: **IGeoObject**<br><br>Дочерний геообъект. |

* Обязательный параметр/опция.

## splice

```
{GeoObjectCollection} splice(index, number)
```

Удаляет геообъекты из коллекции. Если необходимо, вставляет на их место другие. Объекты, которые будут вставлены вместо удаляемых, передаются как дополнительные параметры (после параметра number).

**Возвращает** коллекция удаленных геообъектов.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс геообъекта, с которого начинается удаление. |
| number* | — | Тип: Integer<br><br>Количество удаляемых геообъектов. |

* Обязательный параметр/опция.

**Пример:**

```
// Удаляет второй объект.
myGeoObjects.splice(1, 1);
// Вставляет новый объект obj во вторую позицию.
myGeoObjects.splice(1, 0, obj);
// Заменяет второй объект на новый объект obj.
myGeoObjects.splice(1, 1, obj);
```

## toArray

```
{IGeoObject[]} toArray()
```

**Возвращает** массив, содержащий все геообъекты коллекции на момент вызова.

# geoQuery

Статическая функция.

Формирует набор данных из указанного источника и создает на его основе экземпляр GeoQueryResult.

**Возвращает:** Результат, содержащий данные из источника. (тип **GeoQueryResult**).

```
geoQuery(source)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| source* | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Источник геообъектов:<br>• IGeoObject — объект, реализующий соответствующий интерфейс;<br>• IGeoObject[] — массив объектов, реализующих соответствующий интерфейс;<br>• ICollection — коллекция объектов, реализующих интерфейс IGeoObject;<br>• ICollection[] — массив коллекций объектов, реализующих интерфейс IGeoObject;<br>• vow.Promise — объект-промис, передающий в функцию-обработчик источник данных для geoQuery; Также в функцию-обработчик может быть передан объект с полем geoObjects, содержащий источник данных для geoQuery;<br>• GeoQueryResult — объект класса GeoQueryResult;<br>• String\|Object — строка или объект с JSON-описанием объектов.<br>JSON-описание объектов строится по следующему принципу (см. пример ниже). Объектом может быть сущность или коллекция сущностей. Коллекция сущностей представляет из себя объект с полями:<br>• type — тип объекта. Значение поля должно быть равно "FeatureCollection";<br>• features — массив дочерних сущностей коллекции. Дочерние объекты могут быть сущностями или вложенными коллекциями сущностей.<br>Сущность представляет из себя объект с полями:<br>• type — тип объекта. Значение поля должно быть равно "Feature";<br>• geometry — геометрия объекта. Содержит поля type и coordinates. Соответствует параметру, передаваемому в конструктор объекта ymaps.GeoObject;<br>• options — опции геообъекта;<br>• properties — данные геообъекта. |

* Обязательный параметр/опция.

**Примеры:**

**1.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Создание GeoQueryResult из одиночного геообъекта.
var placemark = new ymaps.Placemark([34, 56]);
ymaps.geoQuery(placemark).addToMap(myMap);
```

**2.**

```
// Создание GeoQueryResult из массива геообъектов.
var objects = [
        new ymaps.Placemark([34, 56]),
        new ymaps.Rectangle([[34, 56], [36, 57]])
    ];
ymaps.geoQuery(objects).addToMap(myMap);
```

**3.**

```
// Создание GeoQueryResult из коллекции геообъектов.
var result = ymaps.geoQuery(myMap.geoObjects).searchIntersect(myMap);
alert("Количество геообъектов в видимой области карты: " + result.getLength());
```

**4.**

```
// Создание GeoQueryResult из vow.Deferred.
var result = ymaps.geoQuery(ymaps.geocode('Сыромятнический
переулок')).searchInside(myGeoBounds);
// Поскольку источник данных асинхронный, нужно дождаться обработки результата.
result.then(function () {
    alert('Количество объектов, лежащих внутри заданной области: ' +
result.getLength());
});
```

**5.**

```
// Создание GeoQueryResult из JSON.
var result = ymaps.geoQuery({
        type: 'FeatureCollection',
            features: [
                {
                    type: 'Feature',
                    geometry: {
                        type: 'Circle',
                        coordinates: [15, 15],
                        radius: 100
                    }
                },
                {
                    type: 'Feature',
                    geometry: {
                        type: 'LineString',
                        coordinates: [[15, 16], [66, 23]]
                    }
                },
                {
                    type: 'FeatureCollection',
                    features: [
                        {
                            type: 'Feature',
                            geometry: {
                                type: 'Point',
                                coordinates: [12, 41]
                            },
                            properties: {
                              name: 'point'
                            },
                            options: {
                                preset: 'islands#yellowIcon'
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
                                }
                            }
                        ]
                    }
                ]
        });
// Неточечные объекты добавим на карту как есть.
result.search('geometry.type != "Point"').addToMap(myMap);
// Точечные объекты добавим на карту через кластеризатор.
myMap.geoObjects.add(result.search('geometry.type == "Point"').clusterize());
```

# GeoQueryResult

Расширяет **IPromiseProvider**.

Результат геозапроса.

Конструктор | Методы

## Конструктор

```
GeoQueryResult(source)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| source * | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Источник геообъектов:<br>• IGeoObject — объект, реализующий соответствующий интерфейс;<br>• IGeoObject[] — массив объектов, реализующих соответствующий интерфейс;<br>• ICollection — коллекция объектов, реализующих интерфейс IGeoObject;<br>• ICollection[] — массив коллекций объектов, реализующих интерфейс IGeoObject;<br>• IPromiseProvider — объект-промис, передающий в функцию-обработчик источник данных для geoQuery; Также в функцию-обработчик может быть передан объект с полем geoObjects, содержащий источник данных для geoQuery;<br>• GeoQueryResult — объект класса GeoQueryResult;<br>• String\|Object — строка или объект с JSON-описанием объектов;<br>• Object[] — массив с JSON-описаниями геометрий. Элемент массива представляет из себя объект с полями type и coordinates. При описании круга также необходимо поле radius. При описании многоугольника можно указывать необязательное поле fillRule.<br>JSON-описание объектов строится по следующему принципу (см. пример ниже). Объектом может быть сущность или коллекция сущностей. Коллекция сущностей представляет из себя объект с полями:<br>• type — тип объекта. Значение поля должно быть равно "FeatureCollection";<br>• features — массив дочерних сущностей коллекции. Дочерние объекты могут быть сущностями или вложенными коллекциями сущностей. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Сущность представляет из себя объект с полями:<br>• type — тип объекта. Значение поля должно быть равно "Feature";<br>• geometry — геометрия объекта. Содержит поля type и coordinates. Соответствует параметру, передаваемому в конструктор объекта ymaps.GeoObject;<br>• options — опции геообъекта;<br>• properties — данные геообъекта. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Создание GeoQueryResult из одиночного геообъекта.
var placemark = new ymaps.Placemark([34, 56]);
ymaps.geoQuery(placemark).addToMap(myMap);
```

**2.**

```
// Создание GeoQueryResult из массива геообъектов.
var objects = [
        new ymaps.Placemark([34, 56]),
        new ymaps.Rectangle([[34, 56], [36, 57]])
    ];
ymaps.geoQuery(objects).addToMap(myMap);
```

**3.**

```
// Создание GeoQueryResult из коллекции геообъектов.
var result = ymaps.geoQuery(myMap.geoObjects).searchIntersect(myMap);
alert("Количество геообъектов в видимой области карты: " + result.getLength());
```

**4.**

```
// Создание GeoQueryResult из vow.Deferred.
var result = ymaps.geoQuery(ymaps.geocode('Сыромятнический
переулок')).searchInside(myGeoBounds);
// Поскольку источник данных асинхронный, нужно дождаться обработки результата.
result.then(function () {
    alert('Количество объектов, лежащих внутри заданной области: ' +
result.getLength());
});
```

**5.**

```
// Создание GeoQueryResult из JSON.
var result = ymaps.geoQuery({
        type: 'FeatureCollection',
            features: [
                {
                    type: 'Feature',
                    geometry: {
                        type: 'Circle',
                        coordinates: [15, 15],
                        radius: 100
                    }
                },
                {
                    type: 'Feature',
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
                    geometry: {
                        type: 'LineString',
                        coordinates: [[15, 16], [66, 23]]
                    }
                },
                // Коллекции объектов могут быть вложенными.
                {
                    type: 'FeatureCollection',
                    features: [
                        {
                            type: 'Feature',
                            geometry: {
                                type: 'Point',
                                coordinates: [12, 41]
                            },
                            properties: {
                              name: 'point'
                            },
                            options: {
                                preset: 'islands#yellowIcon'
                            }
                        }
                    ]
                }
            ]
        });
// Неточечные объекты добавим на карту как есть.
result.search('geometry.type != "Point"').addToMap(myMap);
// Точечные объекты добавим на карту через кластеризатор.
myMap.geoObjects.add(result.search('geometry.type == "Point"').clusterize());
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(source) | **GeoQueryResult** | Добавляет объекты из источника к объектам результата. Не изменяет исходный объект, а создает новый, содержащий объединенный набор геообъектов. |
| **addEvents**(events, callback, context) | **GeoQueryResult** | Навешивает обработчики событий на элементы выборки. |
| **addTo**(collection) | **GeoQueryResult** | Метод для добавления объектов в коллекцию геообъектов. |
| **addToMap**(map) | **GeoQueryResult** | Метод для добавления объектов на карту. |
| **applyBoundsToMap**(map[, options]) | **GeoQueryResult** | Метод, позволяющий установить видимую область карты так, чтобы видны были все объекты из выборки. |
| **clusterize**([options]) | **Clusterer** | Метод создает кластеризатор и добавляет в него объекты из выборки. В случае, если данные выборки еще не готовы, они будут добавлены в кластеризатор сразу после обработки, а возвращенный кластеризатор изначально будет пуст. В кластеризатор будут добавлены только объекты с геометрией "Point". |
| **each**(callback, context) | **GeoQueryResult** | Возвращает ссылку на себя. |
| **get**(index) | **IGeoObject** | Метод возвращает элемент выборки по индексу. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Возвращает географические координаты области, охватывающей объекты результата. |
| **getCenter**(map) | Number[] | Метод, возвращающий центр области, охватывающей объекты результата, в географических координатах. |
| **getCentralObject**(map) | **IGeoObject** \| null | Метод для получения объекта, наиболее близкого к центру видимой области карты. |
| **getClosestTo**(object) | **IGeoObject** \| null | Метод, возвращающий объект выборки, ближайший к указанному. Если на вход подается объект, уже находящийся в выборке, то вернется другой объект выборки, ближайший к указанному. Обратите внимание, что многие геообъекты требуют добавления на карту для корректных расчетов. |
| **getExtreme**(key) | Number | Метод, возвращающий максимальные и минимальные значения координат среди координат объектов выборки. |
| **getExtremeObject**(key) | **IGeoObject** | Метод, возвращающий объект с минимальной или максимальной координатой среди координат объектов выборки. |
| **getGlobalPixelBounds**(map) | Number[][] \| null | Метод, возвращающий глобальные пиксельные координаты области для текущего зума карты. |
| **getGlobalPixelCenter**(map) | Number[] | Метод, возвращающий центр области, охватывающей объекты результата, в глобальных пиксельных координатах для текущего зума карты. |
| **getIterator**() | **IIterator** | Возвращает итератор по объектам результата. |
| **getLength**() | Number | Возвращает количество элементов в результате. |
| **getMaxZoom**(map[, options]) | Number | Метод, вычисляющий максимальный коэффициент масштабирования, при котором в видимую область карты попадают все объекты. |
| **getParent**() | **GeoQueryResult** \| null | Возвращает ссылку на родительскую выборку, если текущая выборка была создана в результате изменения другого объекта GeoQueryResult. |
| **indexOf**(item) | Number | Возвращает индекс элемента в выборке. Если элемент не найден, возвращается -1. |
| **intersect**(result) | **GeoQueryResult** | Метод создает новую выборку, содержащую общие элементы для двух других выборок. |
| **isReady**() | Boolean | Метод, возвращающий признак готовности выборки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **map**(callback[, context]) | **GeoQueryResult** | Метод, вызывающий метод callback для всех элементов выборки и формирующий новую выборку на основе полученных результатов. |
| **remove**(objects) | **GeoQueryResult** | Удаляет объекты из результата. Не изменяет исходный объект, а создает новый, содержащий результирующий набор геообъектов. |
| **removeEvents**(events, callback, context) | | Удаляет подписку на событие с объекта. Обратите внимание, что для корректной отписки передаваемые аргументы должны быть точно такие же, как при подписке через метод addEvents. |
| **removeFrom**(collection) | **GeoQueryResult** | Метод для удаления объектов из коллекции. |
| **removeFromMap**(map) | **GeoQueryResult** | Метод для удаления объектов с карты. |
| **reverse**() | **GeoQueryResult** | Переставляет элементы выборки в обратном порядке и возвращает новую выборку. |
| **search**(condition) | **GeoQueryResult** | Метод для поиска объектов выбоки, удовлетворяющих условиям. |
| **searchContaining**(object) | **GeoQueryResult** | Метод, создающий новую выборку из объектов, содержащих указанный объект. Обратите внимание, что многие геообъекты требуют добавления на карту для корректных расчетов. |
| **searchInside**(object) | **GeoQueryResult** | Метод, создающий новую выборку из объектов, целиком входящих в указанный объект. Обратите внимание, что многие геообъекты требуют добавления на карту для корректных расчетов. |
| **searchIntersect**(object[, options]) | **GeoQueryResult** | Метод, создающий новую выборку из объектов, пересекающих указанный объект. Обратите внимание, что многие геообъекты требуют добавления на карту для корректных расчетов. |
| **setOptions**(key[, value]) | **GeoQueryResult** | Метод, позволяющий задать значение опций всем элементам выборки. |
| **setProperties**(path, value) | **GeoQueryResult** | Метод, позволяющий задать значение поля properties всем элементам выборки. |
| **slice**(begin[, end]) | **GeoQueryResult** | Метод, возвращающий срез выборки. |
| **sort**(comparator) | | Метод для сортировки объектов выборки. Не изменяет исходную выборку, а создает новую, содержащую упорядоченные объекты. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **sortByDistance**(object) | **GeoQueryResult** | Метод, позволяющий получить выборку, содержащую объекты, отсортированные по расстоянию до указанного объекта. Обратите внимание, что многие геообъекты требуют добавления на карту для корректных расчетов. Не изменяет исходную выборку. |
| **then**([onFulfill[, onReject, context) | **GeoQueryResult** | Подписка на "обещание". |
| **unsetOptions**(keys) | **GeoQueryResult** | Метод, позволяющий обнулять значение опций всем элементам выборки. |
| **unsetProperties**(path) | **GeoQueryResult** | Метод позволяющий обнулить значение поля properties у всех элементов выборки. |

## Описание методов

### add

```
{GeoQueryResult} add(source)
```

Добавляет объекты из источника к объектам результата. Не изменяет исходный объект, а создает новый, содержащий объединенный набор геообъектов.

**Возвращает** новый объект с объединенным набором геообъектов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| source* | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Источник геообъектов:<br>• IGeoObject — объект, реализующий соответствующий интерфейс;<br>• IGeoObject[] — массив объектов, реализующих соответствующий интерфейс;<br>• ICollection — коллекция объектов, реализующих интерфейс IGeoObject;<br>• ICollection[] — массив коллекций объектов, реализующих интерфейс IGeoObject;<br>• vow.Promise — объект-промис, передающий в функцию-обработчик источник данных для geoQuery; Также в функцию-обработчик может быть передан объект с полем geoObjects, содержащий источник данных для geoQuery;<br>• GeoQueryResult — объект класса GeoQueryResult;<br>• String\|Object — строка или объект с JSON-описанием объектов;<br>• Object[] — массив с JSON-описаниями геометрий. Элемент массива представляет из себя объект с полями type и coordinates. При описании круга также необходимо поле radius. При описании многоугольника можно указывать необязательное поле fillRule.<br>JSON-описание объектов строится по следующему принципу (см. пример ниже). Объектом может быть сущность или коллекция сущностей. Коллекция сущностей представляет из себя объект с полями:<br>• type — тип объекта. Значение поля должно быть равно "FeatureCollection";<br>• features — массив дочерних сущностей коллекции. Дочерние объекты могут быть сущностями или вложенными коллекциями сущностей. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Сущность представляет из себя объект с полями:<br>• type — тип объекта. Значение поля должно быть равно "Feature";<br>• geometry — геометрия объекта. Содержит поля type и coordinates. Соответствует параметру, передаваемому в конструктор объекта ymaps.GeoObject;<br>• options — опции геообъекта;<br>• properties — данные геообъекта. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Добавление в GeoQueryResult одиночного геообъекта.
var placemark = new ymaps.Placemark([34, 56]);
myGeoQueryResult.add(placemark);
```

**2.**

```
// Добавление в GeoQueryResult массива геообъектов.
var objects = [
        new ymaps.Placemark([34, 56]),
        new ymaps.Rectangle([[34, 56], [36, 57]])
    ];
// Обратите внимание, что в результате будет получен другой объект
GeoQueryResult,
// а старый останется без изменений.
var newResult = myGeoQueryResult.add(objects);
```

**3.**

```
// Добавление в GeoQueryResult коллекции геообъектов.
// Обратите внимание, что в результате будет получен другой объект
GeoQueryResult,
// а старый останется без изменений.
var newResult =
myGeoQueryResult.add(myMap.geoObjects).searchIntersect(myBounds);
alert("Количество геообъектов в указанной области: " +
myGeoQueryResult.getLength());
```

**4.**

```
// Добавление в GeoQueryResult результатов геокодирования.
// Обратите внимание, что в результате будет получен другой объект
GeoQueryResult,
// а старый останется без изменений.
var newResult = myGeoQueryResult.add(ymaps.geocode('Сыромятнический переулок'));
newResult.searchInside(myGeoBounds);
// Поскольку источник данных асинхронный, нужно дождаться обработки результата.
result.then(function () {
    // Исходный объект все еще существует.
    alert('Было объектов до добавления:' + myGeoQueryResult.getLength());
    // В новом объекте уже и старые геообъекты, и добавленные.
    alert('Стало объектов после добавления: ' + result.getLength());
});
```

**5.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Добавление объектов из JSON-строки.
var result = ymaps.geoQuery(myMap.geoObjects),
    extendedResult = result.add('{"type": "Feature", "geometry": { "type":
"Point", "coordinates": [15, 64] }}');
extendedResult.addToMap(myMap);
```

## addEvents

```
{GeoQueryResult} addEvents(events, callback, context)
```

Навешивает обработчики событий на элементы выборки.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| events* | — | Тип: String \| String[]<br><br>Тип или массив типов событий, на которые совершается подписка. |
| callback* | — | Тип: Function<br><br>Функция-обработчик. |
| context* | — | Тип: Object<br><br>Контекст выполнения функции-обработчика. |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.geoQuery(map.geoObjects).search('geometry.type="Circle"').addEvents('click
', function () {
    alert('Вы кликнули по кругу!');
});
```

## addTo

```
{GeoQueryResult} addTo(collection)
```

Метод для добавления объектов в коллекцию геообъектов.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| collection* | — | Тип: **ICollection**<br><br>Коллекция, в которую добавляются объекты. |

* Обязательный параметр/опция.

**Пример:**

```
// Покажем на карте объекты северного полушария.
var result1 = ymaps.geoQuery(placemarks).search('lat >
0').addTo(myMap.geoObjects);
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## addToMap

```
{GeoQueryResult} addToMap(map)
```

Метод для добавления объектов на карту.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map**<br><br>Карта, на которую добавляются объекты. |

* Обязательный параметр/опция.

**Пример:**

```
// Покажем на карте объекты северного полушария.
var result1 = ymaps.geoQuery(placemarks).search('lat > 0').addToMap(myMap);
```

## applyBoundsToMap

```
{GeoQueryResult} applyBoundsToMap(map[, options])
```

Метод, позволяющий установить видимую область карты так, чтобы видны были все объекты из выборки.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map**<br><br>Карта. |
| options | — | Тип: Object<br><br>Опции. |
| options.checkZoomRange | false | Тип: Boolean<br><br>Проверка возможности установить указанный коэффициент масштабирования. Если значение опции равно true, метод вызывается асинхронно. Производится обращение к серверу, который возвращает интервал допустимых для данного центра значений коэффициентов масштабирования. После этого выставляется указанный центр и корректный зум. |
| options.duration | 0 | Тип: Number<br><br>Время анимации в миллисекундах. |
| options.preciseZoom | false | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Возможность использования дробных коэффициентов масштабирования. |
| options.timingFunction | 'linear' | Тип: String<br><br>Тайминг-функция. Аналогично значению CSS свойства transition-timing-function. Полный список значений: http://www.w3.org/TR/css3-transitions/#transition-timing-function_tag |
| options.useMapMargin | true | Тип: Boolean<br><br>Нужно ли учитывать отступы карты **map.margin.Manager**. |
| options.zoomMargin | 0 | Тип: Number \| Number[]<br><br>Отступы от границ видимой области карты. Если задано одно число — оно применяется ко всем сторонам. Если задано два — то это горизонтальные и вертикальные отступы соответственно. Если задан массив из 4х чисел, то это отступы top, right, bottom, left. При включенной опции "useMapMargin" значение "zoomMargin" складывается со значениями, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Сделаем запрос на геокодирование и выставим карте видимую область так,
// чтобы на ней были видны сразу все результаты.
var result = ymaps.geoQuery(ymaps.geocode('улица
Ленина')).applyBoundsToMap(myMap);
// Обратите внимание, что запрос на сервер асинхронный, нужно дождаться
результата.
result.then(function () {
    alert("Результаты получены и видимая область карты скорректирована.");
}, function () {
    alert("Произошла ошибка.");
});
```

**2.**

```
// Для синхронных запросов выставление новой области карты происходит сразу.
var result = ymaps.geoQuery(objects).applyBoundsToMap(myMap);
alert('Видимая область карты изменена.');
```

## clusterize

```
{Clusterer} clusterize([options])
```

Метод создает кластеризатор и добавляет в него объекты из выборки. В случае, если данные выборки еще не готовы, они будут добавлены в кластеризатор сразу после обработки, а возвращенный кластеризатор изначально будет пуст. В кластеризатор будут добавлены только объекты с геометрией "Point".


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** кластеризатор.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции, передаваемые в конструктор объекта **Clusterer**. |

**Пример:**

```
// Выберем только кафе и добавим их в кластеризатор.
var clusterer =
ymaps.geoQuery(objects).search('properties.type="Cafe"').clusterize();
myMap.geoObjects.add(clusterer);
```

## each

```
{GeoQueryResult} each(callback, context)
```

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик. Вызывается для каждого элемента выборки и получает его на вход. |
| context* | — | Тип: Object<br><br>Контекст исполения функции-обработчика. |

* Обязательный параметр/опция.

**Пример:**

```
// Скроем красные метки в видимой области карты.
ymaps.geoQuery(placemarks).searchIntersect(myMap).each(function(pm) {
    if (pm.options.get('preset') == 'islands#redIcon') {
        myMap.geoObjects.remove(pm);
    }
});
```

## get

```
{IGeoObject} get(index)
```

Метод возвращает элемент выборки по индексу.

**Возвращает** объект.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Number<br><br>Индекс элементов. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Примеры:**

**1.**

```
// Пример при синхронной обработке.
var result = ymaps.geoQuery(placemarks).sort('lat'),
    // Самый южный объект.
    southObject = result.get(0),
    // Самый северный объект.
    northObject = result.get(result.getLength() - 1);
```

**2.**

```
// Пример при асинхронной обработке.
var result = ymaps.geoQuery(ymaps.geocode('кафе Москва')).sort('lat');
// Нужно дождаться готовности результата для дальнейшей обработки.
result.then(function () {
    // Самый южный объект.
    var southObject = result.get(0),
    // Самый северный объект.
        northObject = result.get(result.getLength() - 1);
});
```

## getBounds

```
{Number[][] | null} getBounds()
```

**Возвращает** географические координаты области, охватывающей объекты результата.

**Примеры:**

**1.**

```
// Пример работы при синхронном добавлении объектов.
// Устанавливаем центр и масштаб карты так, чтобы были видны все объекты,
добавленные на карту.
var myResult = ymaps.geoQuery(myMap.geoObjects);
myMap.setBounds(myResult.getBounds(), { checkZoomRange: true });
```

**2.**

```
// Пример работы при асинхронном добавлении объектов.
// Нужно дождаться готовности данных и только потом вызывать метод getBounds.
var myResult = ymaps.geoQuery(ymaps.geocode('p. Звероножка')).then(function () {
     myMap.setBounds(myResult.getBounds(), { checkZoomRange: true });
});
```

## getCenter

```
{Number[]} getCenter(map)
```

Метод, возвращающий центр области, охватывающей объекты результата, в географических координатах.

**Возвращает** координаты центра области в географических координатах.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map**<br><br>Карта, для которой нужно произвести расчеты. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Сместим центр карты в центр области, охватывающей объекты.
myMap.setCenter(ymaps.geoQuery(objects).getCenter());
```

## getCentralObject

```
{IGeoObject | null} getCentralObject(map)
```

Метод для получения объекта, наиболее близкого к центру видимой области карты.

**Возвращает** ссылка на геообъект или null в случае пустой выборки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map**<br><br>Карта. |

* Обязательный параметр/опция.

**Пример:**

```
// Пример работы при синхронных операциях.
// Откроем балун у объекта, ближайшего к центру видимой области карты.
ymaps.geoQuery(objects).getCentralObject(myMap).balloon.open();
```

## getClosestTo

```
{IGeoObject | null} getClosestTo(object)
```

Метод, возвращающий объект выборки, ближайший к указанному. Если на вход подается объект, уже находящийся в выборке, то вернется другой объект выборки, ближайший к указанному. Обратите внимание, что многие геообъекты требуют добавления на карту для корректных расчетов.

**Возвращает** объект выборки, ближайший к указанному или null, если объект невозможно найти.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| object* | — | Тип: Object<br><br>Объект, относительно которого будет производится поиск. Может принимать следующие значения:<br>• IGeoObject — объект, реализующий интефейс **IGeoObject**;<br>• IGeometry — объект, реализующий интерфейс **IGeometry**;<br>• Map — карта. В этом случае опорным объектом будет прямоугольная граница карты;<br>• Number[] — координаты точки;<br>• Object — Json-описание геометрии. Содержит поля type и coordinates. При описании круга также необходимо поле radius. При описании многоугольника можно указывать необязательное поле fillRule. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Примеры:**

**1.**

```
// Примеры использования метода при различных входных данных.
var result = ymaps.geoQuery(objects).addToMap(myMap);

// 1. IGeoObject.
// Можно вести поиск относительно стороннего объекта.
var polyline = new ymaps.Polyline([[35, 65], [35, 66], [34, 62], [34, 63]]);
myMap.geoObjects.add(polyline);
var closestObject = result.getClosestTo(polyline);
// Можно искать по объекту из этой же выборки.
var closestToFirst = result.getClosestTo(result.get(0));

// 2. IGeometry.
var closestToGeometry = result.getClosestTo(placemark.geometry);

// 3. Map.
// Найдем объект, ближайший к границе видимой области карты.
var edgeObject = result.getClosestTo(myMap);

// 4. Объект, ближайший к точке.
var closestObject = result.getClosestTo([34, 53]);
```

**2.**

```
// Пример использования при асинхронных операциях.
var result = ymaps.geoQuery(ymaps.geocode('Париж')).addToMap(myMap);
// Дождемся ответа от сервера и получим объект, ближайший к точке.
result.then(function () {
    var closestObject = result.getClosestTo([34, 65]);
    // Если ответ пуст, то ближайший объект не найдется.
    if (closestObject) {
        closestObject.balloon.open();
    }
});
```

## getExtreme

```
{Number} getExtreme(key)
```

Метод, возвращающий максимальные и минимальные значения координат среди координат объектов выборки.

**Возвращает** соответствующее значение координаты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Ключ для получения данных. Может принимать следующие значения:<br>• top;<br>• right;<br>• bottom;<br>• left. |

* Обязательный параметр/опция.

**Примеры:**

**1.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Пример работы при синхронных операциях.
alert('Самая северная координата: ',
ymaps.geoQuery(myMap.geoObjects).getExtreme('top'));
```

**2.**

```
// Пример работы при асинхронных операциях.
var result = ymaps.geoQuery(ymaps.geocode('Новгород'));
// Дождемся ответа от сервера и получим самую северную координату.
result.then(function () {
    alert('Самая северная координата в ответе: ' + result.getExtreme('top'));
});
```

## getExtremeObject

```
{IGeoObject} getExtremeObject(key)
```

Метод, возвращающий объект с минимальной или максимальной координатой среди координат объектов выборки.

**Возвращает** объект с требуемой координатой.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Ключ для поиска объекта. Может принимать следующие значения:<br>• top;<br>• right;<br>• bottom;<br>• left. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Пример работы при синхронных операциях.
// Откроем балун на самом северном объекте.
var topObject = ymaps.geoQuery(myMap.geoObjects).getExtremeObject('top');
topObject.balloon.open();
```

**2.**

```
// Пример работы при асинхронных операциях.
var result = ymaps.geoQuery(ymaps.geocode('море Лаптевых'));
// Нужно дождаться ответа от сервера, чтобы работать с данными.
result.then(function () {
    var topObject = result.getExtremeObject('top');
    // Если ответ пуст, то объект не найдется.
    if (topObject) {
        topObject.balloon.open();
    }
});
```

## getGlobalPixelBounds

```
{Number[][] | null} getGlobalPixelBounds(map)
```

Метод, возвращающий глобальные пиксельные координаты области для текущего зума карты.

**Возвращает** глобальные пиксельные координаты области, охватывающей объекты результата.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map**<br><br>Карта, для которой ведутся расчеты. |

* Обязательный параметр/опция.

**Пример:**

```
var result =
ymaps.geoQuery(placemarks).search('properties.type="shop"').getGlobalPixelBounds
(myMap);
if (Math.abs(result[0][0] - result[1][0]) > myMap.container.getSize()[0]) {
   alert('Объекты не поместятся на карту по ширине!');
}
```

## getGlobalPixelCenter

```
{Number[]} getGlobalPixelCenter(map)
```

Метод, возвращающий центр области, охватывающей объекты результата, в глобальных пиксельных координатах для текущего зума карты.

**Возвращает** координаты центра области в глобальных пиксельных координатах.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map**<br><br>Карта, для которой нужно произвести расчеты. |

* Обязательный параметр/опция.

**Пример:**

```
// Посчитаем номер тайла, на который приходится центр области, охватывающей
результат.
var globalPixelCenter = ymaps.geoQuery(objects).getGlobalPixelCenter(myMap),
    tileNumber = [
        Math.floor(globalPixelCenter[0] / 256),
        Math.floor(globalPixelCenter[1] / 256)
    ];
alert('Номер центрального тайла: ' + tileNumber[0] + ' ' + tileNumber[1]);
```

## getIterator

```
{IIterator} getIterator()
```

**Возвращает** итератор по объектам результата.

**Примеры:**

**1.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Использование итератора при синхронных операциях.
// Поиск элементов, на которые попадают координаты клика.
myMap.events.add('click', function (event) {
    var iterator = ymaps.geoQuery(myMap.geoObjects)
            .searchContaining(event.getCoordinates())
            .getIterator(),
        obj;
   while ((obj = iterator.getNext()) != iterator.STOP_ITERATION) {
       // Совершаем необходимые действия над геообъектом.
   }
});
```

**2.**

```
// Использование итератора при асинхронных операциях.
// Создадим результат из запроса к геокодеру.
var result = ymaps.geoQuery(ymaps.geocode("Староколпакский
переулок")).search("lat > 20");
// Обратите внимание, что мы только отправили запрос к серверу, ответ придет чуть
позже.
// Поскольку запрос асинхронный, нужно дождаться его готовности перед получением
результата.
result.then(function () {
    var iterator = result.getIterator(),
        obj;
    while ((obj = iterator.getNext()) != iterator.STOP_ITERATION) {
        // Совершаем необходимые действия над геообъектом.
    }

});
```

## getLength

```
{Number} getLength()
```

**Возвращает** количество элементов в результате.

**Примеры:**

**1.**

```
var result = ymaps.geoQuery(myMap.geoObject).searchIntersect(myPolygon);
alert('Количество геообъектов, пересекающих многоугольник: ' +
result.getLength());
```

**2.**

```
var result = ymaps.geoQuery(ymaps.geocode('Иваново')).searchInside(myMap);
// Поскольку мы отправили запрос на сервер, сразу посчитать количество элементов
// в результате нельзя. Нужно дождаться ответа.
result.then(function () {
    alert('Количество объектов в видимой области карты: ' + result.getLength());
});
```

## getMaxZoom

```
{Number} getMaxZoom(map[, options])
```

Метод, вычисляющий максимальный коэффициент масштабирования, при котором в видимую область карты попадают все объекты.

**Возвращает** максимальный коэффициент масштабирования карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map** |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Карта, для которой ведется расчет. |
| options | — | Тип: Object<br><br>Опции. |
| options.useMapMargin | true | Тип: Boolean<br><br>Нужно ли учитывать отступы карты **map.margin.Manager**. |

* Обязательный параметр/опция.

**Пример:**

```
// Вычислим максимальный коэффициент масштабирования,
// при котором видны все объекты выборки
// и выставим ограничение для карты.
var maxZoom = ymaps.geoQuery(objects).getMaxZoom();
myMap.options.set('maxZoom', maxZoom);
```

## getParent

```
{GeoQueryResult | null} getParent()
```

**Возвращает** ссылку на родительскую выборку, если текущая выборка была создана в результате изменения другого объекта GeoQueryResult.

**Пример:**

```
ymaps.geoQuery(objectsArray)
    // Сначала выберем объекты с типом "кафе" и зададим для них стили.
    .search('properties.type == "cafe"')
    .setOptions('preset', 'islands#yellowDotIcon')
    // А затем, не прерывая цепочку вызовов,
    // вернемся к исходной выборке и зададим стили для другой группы объектов.
    .getParent()
    .search('properties.type == "shop"')
    .setOptions('preset', 'islands#greenDotIcon');
```

## indexOf

```
{Number} indexOf(item)
```

**Возвращает** индекс элемента в выборке. Если элемент не найден, возвращается -1.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| item* | — | Тип: **IGeoObject**<br><br>Искомый объект. |

* Обязательный параметр/опция.

**Пример:**

```
// Отсортируем выборку по полю name.
var result = ymaps.geoQuery(polygons).sort('properties.name');
alert('Новая позиция первого элемента: ' + result.indexOf(polygons[0]));
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## intersect

```
{GeoQueryResult} intersect(result)
```

Метод создает новую выборку, содержащую общие элементы для двух других выборок.

**Возвращает** новую выборку, содержащую результат пересечения.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| result* | — | Тип: **GeoQueryResult**<br><br>Выборка, с которой пересекают исходную. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Пример пересечения при синхронных операциях.
var result = ymaps.geoQuery(placemarks),
    greenObjects = result.search('properties.color="green"'),
    roundObjects = result.search('properties.shape="round"'),
    greenRoundObjects = greenObjects.intersect(roundObjects);
alert('Количество круглых зеленых объектов: ' + greenRoundObjects.getLength());
```

**2.**

```
// Пример пересечения при асинхронных операциях.
var result = ymaps.geoQuery(ymaps.geocode('Ивановка')),
    filteredByLat = result.search('lat > 56'),
    filteredByLong = result.search('long > 36'),
    intersectedResult = filteredByLat.intersect(filteredByLong);
// Поскольку исходный запрос асинхронный, нужно дождаться готовности данных.
intersectedResult.then(function () {
    alert('Количество объектов с именем "Ивановка"' +
        'с координатами более [56, 36]: ' +
        intersectedResult.getLength());
});
```

## isReady

```
{Boolean} isReady()
```

Метод, возвращающий признак готовности выборки.

**Возвращает** признак, готовы результаты выборки или еще обрабатываются.

**Пример:**

```
var result = ymaps.geoQuery(ymaps.geocode('Иваново'));
if (!result.isReady()) {
    result.then(function () {
        // Обработка данных.
    });
} else {
    // Обработка данных.
}
```

## map

```
{GeoQueryResult} map(callback[, context])
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Метод, вызывающий метод callback для всех элементов выборки и формирующий новую выборку на основе полученных результатов.

**Возвращает** новую выборку.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик. Принимает на вход элемент выборки. Возвращает экземпляр **IGeoObject**. |
| context | — | Тип: Object<br><br>Контекст выполнения функции-обработчика. |

* Обязательный параметр/опция.

**Пример:**

```
// Добавим на карту только объекты-окружности.
var circlesResult =
ymaps.geoQuery(objects).search('geometry.type="Circle"').addToMap(myMap),
// Также добавим на карту метки, обозначающие центры окружностей.
    centers = circlesResult.map(function (object) {
        return new ymaps.Placemark(object.geometry.getCenter());
    }).addToMap(myMap);
```

## remove

```
{GeoQueryResult} remove(objects)
```

Удаляет объекты из результата. Не изменяет исходный объект, а создает новый, содержащий результирующий набор геообъектов.

**Возвращает** новый объект с результирующим набором геообъектов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objects* | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Объекты могут быть представлены в разном виде:<br>• IGeoObject — объект, реализующий соответствующий интерфейс;<br>• IGeoObject[] — массив объектов, реализующих соответствующий интерфейс;<br>• ICollection — коллекция объектов, реализующих интерфейс IGeoObject;<br>• ICollection[] — массив коллекций объектов, реализующих интерфейс IGeoObject;<br>• GeoQueryResult — объект класса GeoQueryResult. Обратите внимание, что при асинхронных операциях нужно дождаться готовности результата для его корректного удаления;<br>• vow.Promise — объект класса vow.Deferred. Должен быть зарезовлен массивом геообъектов или объектом с полем geoObjects. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
var objects = [
        new ymaps.Placemark([34, 56]),
        new ymaps.Rectangle([[34, 56], [36, 57]])
    ],
    result = ymaps.geoQuery(objects);
// Обратите внимание, что в результате будет получен другой объект
GeoQueryResult,
// а старый останется без изменений.
var newResult = result.remove(objects[1]);
```

**2.**

```
// Удаление из GeoQueryResult массива геообъектов.
var objects = [
        new ymaps.Placemark([34, 56]),
        new ymaps.Rectangle([[34, 56], [36, 57]]),
        new ymaps.Placemark([35, 64])
    ],
    result = ymaps.geoQuery(objects);
// Обратите внимание, что в результате будет получен другой объект
GeoQueryResult,
// а старый останется без изменений.
var newResult = result.remove([objects[1], objects[2]]);
```

**3.**

```
// Удаление из GeoQueryResult коллекции геообъектов.
// Добавим на карту объекты, которых там еще нет.
myGeoQueryResult.remove(myMap.geoObjects).addToMap(myMap);
```

**4.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Удаление из GeoQueryResult данных другого GeoQueryResult.
var result1 = ymaps.geoQuery(placemarks).search('properties.color="green"'),
    result2 = ymaps.geoQuery(placemarks).search('properties.shape="circle"'),
    result3 = result1.remove(result2);
alert('Количество зеленых некруглых фигур: ' + result3.getLength());
```

## removeEvents

```
removeEvents(events, callback, context)
```

Удаляет подписку на событие с объекта. Обратите внимание, что для корректной отписки передаваемые аргументы должны быть точно такие же, как при подписке через метод addEvents.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| events* | — | Тип: String \| String[]<br><br>Тип или массив типов событий, на которые была сделана подписка. |
| callback* | — | Тип: Function<br><br>Функция-обработчик, которую указали при подписке. |
| context* | — | Тип: Object<br><br>Контекст, который был указан при подписке. |

* Обязательный параметр/опция.

**Пример:**

```
var callback = function () {
        alert('Вы кликнули по кругу!');
    };
ymaps.geoQuery(map.geoObjects).search('geometry.type="Circle"').addEvents('click
', callback);
// ...
ymaps.geoQuery(map.geoObjects).search('geometry.type="Circle"').removeEvents('cl
ick', callback);
```

## removeFrom

```
{GeoQueryResult} removeFrom(collection)
```

Метод для удаления объектов из коллекции.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| collection* | — | Тип: ICollection<br><br>Коллекция, из которой нужно удалить объекты. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Покажем на карте все объекты.
var result1 = ymaps.geoQuery(placemarks).addTo(myMap.geoObjects),
    // А затем скроем объекты из северного полушария.
    result2 = result1.search('lat > 0').removeFrom(myMap.geoObjects);
```

## removeFromMap

```
{GeoQueryResult} removeFromMap(map)
```

Метод для удаления объектов с карты.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map**<br><br>Карта, с которой нужно удалить объекты. |

* Обязательный параметр/опция.

**Пример:**

```
// Покажем на карте все объекты.
var result1 = ymaps.geoQuery(placemarks).addToMap(myMap),
// А затем скроем объекты из северного полушария.
    result2 = result1.search('lat > 0').removeFromMap(myMap);
```

## reverse

```
{GeoQueryResult} reverse()
```

Переставляет элементы выборки в обратном порядке и возвращает новую выборку.

**Возвращает** новую выборку с элементами в обратном порядке.

**Примеры:**

**1.**

```
// Использование при синхронных запросах.
var result = ymaps.geoQuery(myMap.geoObjects).sort('x'),
    invertedResult = result.reverse();
```

**2.**

```
// Использование при асинхронных запросах.
var result = ymaps.geoQuery(ymaps.geocode('село Добрые пчелы')).sort('x'),
    invertedResult = result.reverse();
invertedResult.then(function () {
    // Результат получен, можно начинать пользоваться.
});
```

## search

```
{GeoQueryResult} search(condition)
```

Метод для поиска объектов выбоки, удовлетворяющих условиям.

**Возвращает** новую выборку, содержащую результаты поиска.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| condition* | — | Тип: String \| Function<br><br>Строковый шаблон для поиска или функция-фильтр. Строковый шаблон имеет структуру "<fieldname> <condition> <expression>". Возможные значения <fieldname>:<br>• 'lat' — широта. Поддерживается только для точечных объектов;<br>• 'lng', 'long' — долгота. Поддерживается только для точечных объектов;<br>• 'x' — глобальные пиксельные координаты по оси x. Поддерживается только для точечных объектов;<br>• 'y' — глобальные пиксельные координаты по оси y. Поддерживается только для точечных объектов;<br>• 'geometry.type' — тип геометрии, результат выполнения метода geometry.getType();<br>• 'geometry.coordinates.<index>' — координаты, результат выполнения метода geometry.getCoordinates(). Для получения доступа к координате следует указывать ее индексы через точку — 'geometry.coordinates.0.1';<br>• 'properties.<path>' — значение поля данных;<br>• 'options.<key>' — значение опции.<br><br>Возможные значения <condition>:<br>• '>';<br>• '>=';<br>• '==', '=';<br>• '!=';<br>• '<=';<br>• '<';<br>• 'rlike', 'regexp' — соответствие значения <fieldname> некоторому регулярному выражению. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Возможные значения <expression>:<br>• Число;<br>• Строка — обязательно указывается в кавычках;<br>• true;<br>• false;<br>• null;<br>• undefined.<br>Функция-фильтр принимает на вход объект из выборки и возвращает true/false. True — объект попадет в результирующую выборку, false — не попадет. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Пример поиска объектов при синхронных операциях.
var result = ymaps.geoQuery(myMap.geoObjects);

        // Поиск объектов с определенным типом геометрии. Обратите внимание,
что
// значение поля указано в кавычках - это строка.
result.search('geometry.type = "Circle"')
// Поиск по координате.
    .search('geometry.coordinates.0 > 100')
// Поиск по широте.
    .search('lat < 0')
// Поиск по пиксельным координатам.
    .search('x >= 100')
// Поиск по значению поля в properties.
    .search('properties.name != null').search('properties.name rlike "(.) \\""')
    .search('properties.author.name = "Степан"')
// Поиск по значению опции.
    .search('options.visible = true');
```

**2.**

```
// Использование метода при асинхронных операциях.
var result = ymaps.geoQuery(ymaps.geocode('Лось'));
// Для работы с результатом нужно дождаться его готовности.
result.then(function () {
    result.search('properties.description regExp "*поселок*"')
        .addToMap(myMap)
        .applyBoundsToMap(myMap);
});
```

## searchContaining

```
{GeoQueryResult} searchContaining(object)
```

Метод, создающий новую выборку из объектов, содержащих указанный объект. Обратите внимание, что многие геообъекты требуют добавления на карту для корректных расчетов.

**Возвращает** новую выборку, содержащую искомые объекты.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| object* | — | Тип: Object<br><br>Объект, относительно которого будет производится поиск. Может принимать следующие значения:<br>• IGeoObject — объект, реализующий интефейс **IGeoObject**;<br>• IGeometry — объект, реализующий интерфейс **IGeometry**;<br>• Map — карта. В этом случае опорным объектом будет прямоугольная граница карты;<br>• Number[] — координаты точки. Если передан один параметр, трактуется как точка. Если передано два аргумента, трактуется как центр окружности;<br>• Number[][] — координаты прямоугольной области;<br>• Object — Json-описания геометрии. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Примеры использования метода при различных входных данных.
var result = ymaps.geoQuery(objects).addToMap(myMap);

// 1. IGeoObject.
var polygon = new ymaps.Polygon([[[35, 65], [35, 66], [34, 62], [34, 63], [35,
65]]]);
myMap.geoObjects.add(polygon);
var objectsContainingPolygon = result.searchContaining(polygon);

// 2. IGeometry.
var objectsContainingGeometry = result.searchContaining(polygon.geometry);

// 3. Map.
var objectsContainingMapBounds = result.searchContaining(myMap);
```

**2.**

```
// Пример использования при асинхронных операциях.
var result = ymaps.geoQuery(ymaps.geocode('кафе'))
        .addToMap(myMap);
// Дождемся ответа от сервера перед обработкой результатов.
result.then(function () {
    var areas = result.map(function (object) {
                return new ymaps.Circle(object.geometry.getCoordinates(), 100);
            })
            .setOptions('visible', false)
            .addToMap(myMap);
    myMap.events.add('click', function (event) {
        if (areas.searchContaining(event.getCoordinates()).getLength()) {
            alert('Вы кликнули около кафе.');
        }
    });
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## searchInside

```
{GeoQueryResult} searchInside(object)
```

Метод, создающий новую выборку из объектов, целиком входящих в указанный объект. Обратите внимание, что многие геообъекты требуют добавления на карту для корректных расчетов.

**Возвращает** новую выборку, содержащую искомые объекты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| object* | — | Тип: Object<br><br>Объект, относительно которого будет производится поиск. Может принимать следующие значения:<br>• IGeoObject — объект, реализующий интефейс **IGeoObject**;<br>• IGeometry — объект, реализующий интерфейс **IGeometry**;<br>• Map — карта. В этом случае опорным объектом будет прямоугольная граница карты. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Примеры использования метода при различных входных данных.
var result = ymaps.geoQuery(objects).addToMap(myMap);

// 1. IGeoObject.
var polygon = new ymaps.Polygon([[[35, 65], [35, 66], [34, 62], [34, 63], [35,
65]]]);
myMap.geoObjects.add(polygon);
var objectsInsidePolygon = result.searchInside(polygon);

// 2. IGeometry.
var objectsInsideGeometry = result.searchInside(polygon.geometry);

// 3. Map.
// Найдем объекты, целиком лежащие в видимой области карты.
var visibleObject = result.searchInside(myMap);
```

**2.**

```
// Пример использования при асинхронных операциях.
var result = ymaps.geoQuery(ymaps.geocode('Иваново'))
        .setOptions('visible', false)
        .addToMap(myMap);
// Дождемся ответа от сервера и покажем объекты, попадающие в видимую область
карты.
result.then(function () {
    result.searchInside(myMap).setOptions('visible', true);
});
```

## searchIntersect

```
{GeoQueryResult} searchIntersect(object[, options])
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Метод, создающий новую выборку из объектов, пересекающих указанный объект. Обратите внимание, что многие геообъекты требуют добавления на карту для корректных расчетов.

**Возвращает** новую выборку, содержащую искомые объекты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| object* | — | Тип: Object<br><br>Объект, относительно которого будет производится поиск. Может принимать следующие значения:<br>• IGeoObject — объект, реализующий интефейс **IGeoObject**;<br>• IGeometry — объект, реализующий интерфейс **IGeometry**;<br>• Map — карта. В этом случае опорным объектом будет прямоугольная граница карты. |
| options | — | Тип: Object<br><br>Опции. |
| options.considerOccurance | true | Тип: Object<br><br>Флаг, показывающий, считаем ли мы пересечением, когда одна фигура полностью содержится в другой. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Примеры использования метода при различных входных данных.
var result = ymaps.geoQuery(objects).addToMap(myMap);

// 1. IGeoObject.
var polygon = new ymaps.Polygon([[[35, 65], [35, 66], [34, 62], [34, 63], [35,
65]]]);
myMap.geoObjects.add(polygon);
var objectsIntersectPolygon = result.searchIntersect(polygon);

// 2. IGeometry.
var objectsIntersectGeometry = result.searchIntersect(polygon.geometry);

// 3. Map.
// При этом будем искать только объекты, пересекающие непосредственно
// границы карты. То есть объекты, попадающие целиком в видимую область карты,
// не попадут в финальную выборку.
var objectsIntersectMapBounds = result.searchIntersect(myMap,
{considerOccurance: false});
```

**2.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Пример использования при асинхронных операциях.
var result = ymaps.geoQuery(ymaps.geocode('кафе'))
        .setOptions('visible', false)
        .addToMap(myMap);

// Дождемся ответа от сервера и покажем объекты, попадающие в видимую область
карты.
result.then(function () {
    result.searchIntersect(myMap).setOptions('visible', true);
});
```

## setOptions

```
{GeoQueryResult} setOptions(key[, value])
```

Метод, позволяющий задать значение опций всем элементам выборки.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String \| Object<br><br>Название опции или хэш с опциями и их значениями. |
| value | — | Тип: Object<br><br>Значение опции. |

* Обязательный параметр/опция.

**Пример:**

```
var result = ymaps.geoQuery(placemarks);
// Сделаем видимыми элементы, попадающие в прямоугольную область.
result.searchIntersect(myBounds).setOptions('visible', true);

// Зададим опции с помощью хэша.
result.setOptions({zIndex: 10, fillColor: '#ff0005'});
```

## setProperties

```
{GeoQueryResult} setProperties(path, value)
```

Метод, позволяющий задать значение поля properties всем элементам выборки.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| path* | — | Тип: String<br><br>Имя поля, которому присваивается значение. Может содержать ".". |
| value* | — | Тип: Object<br><br>Значение поля. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
var result = ymaps.geoQuery(objects);
// Пометим элементы, попадающие в области.
result.searchIntersect(myBounds1).setProperties('intersectBounds', true);
result.searchIntersect(myBounds2).setProperties('intersectBounds', true);
// ...
result.search('properties.intersectBounds = true').addToMap(myMap);
```

## slice

```
{GeoQueryResult} slice(begin[, end])
```

Метод, возвращающий срез выборки.

**Возвращает** новую выборку, содержащую элементы среза.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| begin* | — | Тип: Number<br><br>Индекс начального элемента в срезе. |
| end | — | Тип: Number<br><br>Индекс элемента в выборке, на котором срез завершится. При этом последним в новом срезе будет элемент с индексом end-1. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Выполним срез при синхронной обработке.
var result = ymaps.geoQuery(map.geoObjects).slice(0, 10);
alert('Количество элементов в новой выборке:' + result.getLength());
```

**2.**

```
// Взятие среза при асинхронной обработке.
var result = ymaps.geoQuery(ymaps.geocode('кафе Москва')).slice(0, 10);
// Сразу после запроса результат еще не готов.
alert('Пока выборка пуста. Количество элементов в выборке: ' +
result.getLength());
// Дождемся готовности результата и посмотрим, сколько элементов будет в
выборке.
result.then(function () {
    alert('Ответ получен. Количество данных в выборке: ' + result.getLength());
});
```

## sort

```
sort(comparator)
```

Метод для сортировки объектов выборки. Не изменяет исходную выборку, а создает новую, содержащую упорядоченные объекты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| comparator* | — | Тип: String \| Function |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Строка с шаблоном сортировки или функция-компаратор. Строка-шаблон может быть представлена в виде '<fieldname> [<order>=asc]'. Возможные значения <fieldname>:<br>• 'lat' — широта. Поддерживается только для точечных объектов;<br>• 'lng', 'long' — долгота. Поддерживается только для точечных объектов;<br>• 'x' — глобальные пиксельные координаты по оси x. Поддерживается только для точечных объектов;<br>• 'y' — глобальные пиксельные координаты по оси y. Поддерживается только для точечных объектов;<br>• 'geometry.type' — тип геометрии, результат выполнения метода geometry.getType();<br>• 'geometry.coordinates.<index>' — координаты, результат выполнения метода geometry.getCoordinates(). Для получения доступа к координате следует указывать ее индексы через точку — 'geometry.coordinates.0.1';<br>• 'properties.<path>' — значение поля данных;<br>• 'options.<key>' — значение опции.<br>Возможные значения необязательного параметра <order>:<br>• 'asc' — сортировка по возрастанию;<br>• 'desc' — сортировка по убыванию.<br>Функция-компаратор принимает на вход два элемента выборки. Возвращаемые значения: Если первый объект больше второго, функция возвращает значение > 0; Если первый объект равен второму, функция возвращает 0; Если первый объект меньше второго, функция возвращает < 0. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Пример использования при синхронных операциях.
var result = ymaps.geoQuery(myMap.geoObjects);
result.sort('lat').sort('x')
    .sort('properties.name desc')
    .sort('options.preset')
    .sort(function (a, b) {
        if (a.properties.get('name') == b.properties.get('name')) {
            return a.geometry.getCoordinates()[0] - b.geometry.getCoordinates()
[0];
        } else {
            return (a.properties.get('name') > b.properties.get('name')) ? 1 :
-1;
        }
    });
```

## sortByDistance

```
{GeoQueryResult} sortByDistance(object)
```

Метод, позволяющий получить выборку, содержащую объекты, отсортированные по расстоянию до указанного объекта. Обратите внимание, что многие геообъекты требуют добавления на карту для корректных расчетов. Не изменяет исходную выборку.

**Возвращает** новую упорядоченную выборку.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| object* | — | Тип: Object<br><br>Объект, до которого будет считаться расстояние. Может принимать следующие значения:<br>• IGeoObject — объект, реализующий интефейс **IGeoObject**;<br>• IGeometry — объект, реализующий интерфейс **IGeometry**;<br>• Map — карта. В этом случае опорным объектом будет прямоугольная граница карты;<br>• Number[] — координаты точки;<br>• Number[][] — координаты прямоугольной области;<br>• Object — Json-описание геометрии. Содержит поля type и coordinates. При описании круга также необходимо поле radius. При описании многоугольника можно указывать необязательное поле fillRule. |

* Обязательный параметр/опция.

**Примеры:**

**1.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Примеры использования метода при различных входных данных.
var result = ymaps.geoQuery(objects).addToMap(myMap);

// 1. IGeoObject.
var polyline = new ymaps.Polyline([[35, 65], [35, 66], [34, 62], [34, 63]]);
myMap.geoObjects.add(polyline);
var sortedByPolyline = result.sortByDistance(polyline);

// 2. IGeometry.
var sortedByGeometry = result.sortByDistance(placemark.geometry);

// 3. Map.
var sortedByMapBounds = result.sortByDistance(myMap);

// 4. Выборка, отсортированная по точке.
var sortedByPoint = result.sortByDistance([34, 53]);
```

**2.**

```
// Пример использования при асинхронных операциях.
var result =
ymaps.geoQuery(ymaps.geocode('Париж')).addToMap(myMap).sortByDistance([45, 64]);
// Дождемся ответа от сервера и получим ближайший и наиболее удаленный
// объект по отношению к точке.
result.then(function () {
    alert('Ближайший объект имеет координаты ' +
result.get(0).geometry.getCoordinates());
    alert('Наиболее удаленный объект имеет координаты ' +
result.get(result.getLength() - 1).geometry.getCoordinates());
});
```

## then

```
{GeoQueryResult} then([onFulfill[, onReject, context)
```

Подписка на "обещание".

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| onFulfill | — | Тип: Function<br><br>Функция-обработчик, которая вызывается, если обещание выполнено. |
| onReject | — | Тип: Function<br><br>Функция-обработчик, которая вызывается, если обещание не выполнено (произошла ошибка). |
| context* | — | Тип: Object<br><br>Контекст функций-обработчиков. |

* Обязательный параметр/опция.

**Пример:**

```
var result = ymaps.geoQuery(ymaps.geocode('река Лена'));
result.then(function () {
    alert('Количество найденных объектов: ' + result.getLength());
}, function () {
    alert('Произошла ошибка.');
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## unsetOptions

```
{GeoQueryResult} unsetOptions(keys)
```

Метод, позволяющий обнулять значение опций всем элементам выборки.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| keys* | — | Тип: String \| String<br><br>Название или массив названий опций, которые должны быть аннулированы. |

* Обязательный параметр/опция.

**Пример:**

```
result.unsetOptions('visible');
```

## unsetProperties

```
{GeoQueryResult} unsetProperties(path)
```

Метод позволяющий обнулить значение поля properties у всех элементов выборки.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| path* | — | Тип: String<br><br>Имя поля, значение которого нужно обнулить. Может содержать ".". |

* Обязательный параметр/опция.

**Пример:**

```
var result = ymaps.geoQuery(objects);
// Пометим элементы, попадающие в первую область, но не попадающие во вторую.
result.searchIntersect(myBounds1).setProperties('intersectBounds', true);
result.searchIntersect(myBounds2).unsetProperties('intersectBounds', true);
// ...
result.search('properties.intersectBounds = true').addToMap(myMap);
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# geoXml

## geoXml.load

Статическая функция.

Загружает XML-файл с географическими данными и преобразует его в коллекцию **GeoObjectCollection**. Сформированная коллекция может быть передана заданной функции для последующей обработки. Поддерживаемые форматы XML-данных: YMapsML, KML, GPX. Для верхней коллекции геообъектов GPX-файла доступны следующие пресеты:

- 'gpx#interactive' — обеспечивает вывод информации о точке маршрута при клике. Также при использовании этого пресета в макете балуна становятся доступными следующие свойства геообъекта: time, velocity, trackName, trackDescription, pointName, pointDescription, lon, lat, sym. Используется по умолчанию;
- 'gpx#plain' — элементы gpx-коллекции ведут себя как обычные геообъекты;

**Возвращает:** Объект-promise. Если xml-файл по указанному урлу будет успешно загружен, promise будет подтверждён и получит в качестве параметров объект со следующими полями:

- geoObjects — коллекция геообъектов **GeoObjectCollection**;
- mapState — описание состояния карты **IMapState** (только для YMapsML);

(тип **vow.Promise**).

```
geoXml.load(url)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| url* | — | Тип: String<br><br>URL файла с географическими данными. |

* Обязательный параметр/опция.

**Пример:**

```
// Создание и инициализация карты map...

// Загрузка и отображение ymapsml-файла с сервиса "Мои карты"
ymaps.geoXml.load('http://maps.yandex.ru/export/usermaps/
HNQ5uTUgbjy6L0dW2uReUjSoXb1Ad7jw/')
     .then(function (res) {
         // добавление элементов ymapsml-файла на карте
         map.geoObjects.add(res.geoObjects);
         // Выставление границ и типа к карты.
         // границы res.mapState.getBounds() применяются к карте асинхронно,
         // т.к. необходимо получить информацию о доступных масштабах для данных
границ
         // тип карты res.mapState.getType() применяется к карте синхронно
         if (res.mapState) {
             res.mapState.applyToMap(map).then(function () {
                 alert('К карте применены границы ' +
res.mapState.getBounds().toString());
             });
         }
         // если информация о границах не предоставлена в repr:View YMapsML
файла,
         // можно применить применить gml:boundedBy верхнего элемента
ymaps:GeoObjectCollection
         else if (res.geoObjects.properties.get('boundedBy')) {
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
            map.setBounds(res.geoObjects.properties.get('boundedBy'), {
                checkZoomRange: true
            });
        }
    });

// Загрузка и отображение kml-файла
ymaps.geoXml.load('http://api.yandex.ru/maps/doc/jsapi/1.x/examples/kml/
demonstration.xml')
    .then(function (res) {
        map.geoObjects.add(res.geoObjects);
    });

// Загрузка и отображение gpx-файла
ymaps.geoXml.load('http://karmatsky.narod2.ru/MskChel2.xml')
    .then(function (res) {
        res.geoObjects.options.set({
            balloonContentBodyLayout: ymaps.templateLayoutFactory.createClass(
            // будем показывать в балуне только свойство name геообъекта и
скорость
            '{{ properties.name }} {{ properties.velocity }}'
            )
        });

        map.geoObjects.add(res.geoObjects);
        // метаданные о границах из gpx-файла хранятся в свойствах коллекции
res.geoObjects
        // применим эти границы к карте
        if (res.geoObjects.properties.get('boundedBy')) {
            map.setBounds(res.geoObjects.properties.get('boundedBy'), {
                checkZoomRange: true
            });
        }
    });
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getZoomRange

Статическая функция.

Определяет доступный диапазон уровней масштабирования в указанной точке для указанного типа карты. Возвращает объект-promise, который будет подтверждён и получит в качестве параметра массив из двух чисел — максимальный и минимальный уровень масштабирования в данной точке.

**Возвращает:** Объект-promise. (тип **vow.Promise**).

```
getZoomRange(mapType, coords, customizable)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| mapType* | — | Тип: String \| **MapType**<br><br>Тип карты. Строка-ключ из **mapType.storage**, либо экземпляр класса **MapType**. |
| coords* | — | Тип: Number[]<br><br>Координаты точки, для которой нужно узнать диапазон доступных уровней масштабирования. |
| customizable* | — | Тип: **ICustomizable**=null<br><br>Объект, который содержит менеджер опций. Опции объекта будут учитываться при получении результата. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Пусть мы хотим инициализировать карту на максимальном масштабе.
var myMap;
ymaps.getZoomRange('yandex#map', [55.750516, 37.615924]).then(function (result)
{
     myMap = new ymaps.Map('mapContainer', {
         center: [55.750516, 37.615924],
         zoom: result[1]
     });
});
```

**2.**

```
// Инициализируем карту с помощью геокодера, с центром на доме 16 по улице Льва
Толстого
// на максимально возможном масштабе.
var myMap;
ymaps.geocode("Москва, Льва Толстого, 16").then(function (geoData) {
    var coords = geoData.geoObjects.get(0).geometry.getCoordinates();
    ymaps.getZoomRange('yandex#map', coords).then(function (zoomRange) {
        myMap = new ymaps.Map('mapContainer', {
           center: coords,
           zoom: zoomRange[1]
        });
    });
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# graphics

## graphics.style

### graphics.style.color

Статический объект.

Задает цвет графической фигуры в форматах #RGB,#RGBA,#RRGGBB,#RRGGBBAA,rgb(r,b,a),rgba(r,g,b,a)

```
graphics.style.color()
```

**Пример:**

```
strokeColor: '#F00';
strokeColor: '#FF0000';
strokeColor: '#FF0000AA';
strokeColor: 'rgba(255,0,0,1)'.
```

### graphics.style.stroke

Статический объект.

Позволяет изменить стиль линии. Задавать значение можно как используя ключи описанные ниже, так и в формате массива. Следует учесть, что в режиме отображениея VML(IE<9) могут быть использованны только ключи. Также в стиле линии можно указать смещение начала пунктира через поле offset.

Поля

```
graphics.style.stroke()
```

**Пример:**

```
strokeStyle: 'dot';
strokeStyle: [1,2];
strokeStyle: {
  style: 'dot',
  offset: 10
}
```

#### Поля

| Имя | Тип | Описание |
|---|---|---|
| **dash** | | Тире |
| **dashdot** | | Длинное тире-короткое тире |
| **dot** | | Точки |
| **longdash** | | Длинные тире |
| **longdashdot** | | Очень длинное тире — точка |
| **longdashdotdot** | | Длинное тире-точка-точка |
| **shortdash** | | Короткие тире |
| **shortdashdot** | | Тире-точка |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| shortdashdotdot | | Тире-точка-точка |
| shortdot | | Точки через двойной интервал |
| solid | | Сплошная линия |

## Описание полей

### dash

```
dash
```

Тире

### dashdot

```
dashdot
```

Длинное тире-короткое тире

### dot

```
dot
```

Точки

### longdash

```
longdash
```

Длинные тире

### longdashdot

```
longdashdot
```

Очень длинное тире — точка

### longdashdotdot

```
longdashdotdot
```

Длинное тире-точка-точка

### shortdash

```
shortdash
```

Короткие тире

### shortdashdot

```
shortdashdot
```

Тире-точка

### shortdashdotdot

```
shortdashdotdot
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Тире-точка-точка

## shortdot

```
shortdot
```

Точки через двойной интервал

## solid

```
solid
```

Сплошная линия


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# Hint

Расширяет **IHint**, **Popup**.

Хинт — всплывающая подсказка, которая может отображать любое HTML-содержимое. Обычно хинт существует на карте в единственном экземпляре, управление им происходит через специальные менеджеры (карты, геообъектов, слоёв активных областей и т.д.). Не создавайте их самостоятельно, если в этом нет необходимости.

Конструктор | Поля | События | Методы

## Конструктор

```
Hint(map[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Ссылка на объект карты. |
| options | — | Тип: Object<br><br>Опции. |
| options.contentLayout | — | Тип: Function \| String<br><br>Макет для содержимого хинта. (Тип: конструктор объекта с интерфейсом ILayout). |
| options.fitPane | true | Тип: Boolean<br><br>Флаг, заставляющий инфо-объекта сдвигать свою позицию, чтобы не выходить за границы контейнера. |
| options.holdByMouse | true | Тип: Boolean<br><br>Флаг, отменяющий закрытие хинта, находящегося под курсором. |
| options.layout | islands#hint | Тип: Function \| String<br><br>Внешний макет хинта. (Тип: конструктор объекта с интерфейсом ILayout). |
| options.offset | — | Тип: Number[]<br><br>Дополнительное смещение позиции относительно точки привязки. |
| options.pane | 'outerHint' | Тип: String<br><br>Ключ пейна, в который помещается оверлей хинта. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Создаем независимый экземпляр хинта и показываем его с собстенным макетом в
центре карты с помощью указания гео-координат.
(new ymaps.Hint(myMap, {
    projection: ymaps.projection.wgs84Mercator,
    layout: ymaps.templateLayoutFactory.createClass('{{ content }}')
})).open(myMap.getCenter(), 'Hello');
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| close | Закрытие инфо-объекта.<br>Унаследовано от IPopup. |
| open | Открытие инфо-объекта.<br>Унаследовано от IPopup. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| close([force]) | vow.Promise | Закрывает инфо-объект.<br>Унаследован от IPopup. |
| getData() | | Возвращает данные инфо-объекта.<br>Унаследован от IPopup. |
| getOverlay() | vow.Promise | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от IPopup. |
| getOverlaySync() | IOverlay | Возвращает оверлей, если тот существует.<br>Унаследован от IPopup. |
| getPosition() | | Возвращает координаты инфо-объекта.<br>Унаследован от IPopup. |
| isOpen() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от IPopup. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **open**([position[, data]]) | **vow.Promise** | Открывает инфо-объект в указанной позиции. Если инфо-объект уже открыт, перемещает его в указанную точку. Формат и суть координат определяется проекцией **IProjection**, которая содержится в опциях.<br>Унаследован от **IPopup**. |
| **setData**(data) | **vow.Promise** | Задаёт инфо-объекту новые данные.<br>Унаследован от **IPopup**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopup**. |

# Hotspot

Расширяет **IHotspot**.

Активная область.

Конструктор | Поля | События | Методы

## Конструктор

```
Hotspot(shape[, zIndex])
```

Активная область.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| shape * | — | Тип: **IShape**<br>Фигура активной области. |
| zIndex | 0 | Тип: Number<br>zIndex активной области. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getShape() | IShape | Возвращает фигуру активной области.<br>Унаследован от IHotspot. |
| getZIndex() | Number | Возвращает zIndex активной области.<br>Унаследован от IHotspot. |
| setShape(shape) | | Задает фигуру активной области.<br>Унаследован от IHotspot. |
| setZIndex(zIndex) | | Задает z-index активной области.<br>Унаследован от IHotspot. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# hotspot

## hotspot.layer

## hotspot.layer.addon

### hotspot.layer.addon.balloon

**Примечание:**
Конструктор класса hotspot.layer.addon.balloon скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Статический объект.

Методы

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get(layer) | IPopupManager | Возвращает менеджер балуна хотспотного слоя. |

## Описание методов

## get

```
{IPopupManager} get(layer)
```

**Возвращает** менеджер балуна хотспотного слоя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| layer* | — | Тип: **hotspot.Layer**<br><br>Хотспотный слой. |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.hotspot.layer.addon.balloon.get(layer)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# hotspot.layer.addon.hint

**Примечание:**
Конструктор класса hotspot.layer.addon.hint скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Статический объект.

Методы

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **get**(layer) | **IPopupManager** | Возвращает менеджер хинта хотспотного слоя. |

## Описание методов

## get

```
{IPopupManager} get(layer)
```

**Возвращает** менеджер хинта хотспотного слоя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| layer* | — | Тип: **hotspot.Layer**<br><br>Хотспотный слой. |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.hotspot.layer.addon.hint.get(layer)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# hotspot.layer.Balloon

Расширяет **IBalloonManager**.

Менеджер балуна хотспотного слоя. Позволяет управлять балуном хотспотного слоя, открывать и скрывать его. Использует внутри себя менеджер балуна карты **map.Balloon**. Хотспотные слои содержат экземпляр этого класса, доступный как myHotspotLayer.balloon. Не создавайте экземпляры этого класса без необходимости.

**См.: Balloon hotspot.Layer.balloon**

Конструктор | Поля | События | Методы

## Конструктор

```
hotspot.layer.Balloon(hotspotLayer)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| hotspotLayer * | — | Тип: Object<br><br>Хотспотный слой. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **autopanbegin** | Начало автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **autopanend** | Окончание автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **beforeuserclose** | Событие, предшествующее Balloon.userclose. Позволяет отменить пользовательское действие посредством вызова метода preventDefault. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **userclose** | Закрытие балуна пользователем. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• target — ссылка на объект **IBalloonOwner**.<br>Унаследовано от **IBalloonManager**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **autoPan**() | **vow.Promise** | Перемещает карту таким образом, чтобы балун оказался видимым.<br>Унаследован от **IBalloonManager**. |
| **close**([force]) | **vow.Promise** | Закрывает инфо-объект.<br>Унаследован от **IPopupManager**. |
| **destroy**() | | Отключает менеджер инфо-объекта.<br>Унаследован от **IPopupManager**. |
| **getData**() | Object \| null | Возвращает данные инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **getOptions**() | **IOptionManager** \| null | Возвращает менеджер опций или `null`.<br>Унаследован от **IPopupManager**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от **IPopupManager**. |
| **getOverlaySync**() | **IOverlay** \| null | Возвращает оверлей, если тот существует.<br>Унаследован от **IPopupManager**. |
| **getPosition**() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **isOpen**() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от **IPopupManager**. |
| **open**([position[, data[, options]]]) | **vow.Promise** | Открывает балун в указанной позиции. |
| **setData**(data) | **vow.Promise** | Задаёт инфо-объекту новые данные.<br>Унаследован от **IPopupManager**. |
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopupManager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Описание методов

#### open

```
{vow.Promise} open([position[, data[, options]]])
```

Открывает балун в указанной позиции.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position | — | Тип: Number[]<br><br>Координаты открытия в глобальных пиксельных координатах. |
| data | — | Тип: Object<br><br>Данные. |
| options | — | Тип: Object<br><br>Опции. |

## hotspot.layer.Hint

Расширяет **IHintManager**.

Менеджер хинта хотспотного слоя. Позволяет управлять хинтом хотспотного слоя, открывать и скрывать его. Использует внутри себя менеджер хинтов карты **map.Hint**. Хотспотные слои содержат экземпляр этого класса, доступный как myHotspotLayer.hint. Не создавайте экземпляры этого класса без необходимости.

**См.:** **Hint hotspot.Layer.hint**

Конструктор | Поля | Методы

### Конструктор

```
hotspot.layer.Hint(hotspotLayer)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| hotspotLayer * | — | Тип: Object<br><br>Хотспотный слой. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| close([force]) | vow.Promise | Закрывает инфо-объект.<br>Унаследован от IPopupManager. |
| destroy() | | Отключает менеджер инфо-объекта.<br>Унаследован от IPopupManager. |
| getData() | Object \| null | Возвращает данные инфо-объекта или `null`.<br>Унаследован от IPopupManager. |
| getOptions() | IOptionManager \| null | Возвращает менеджер опций или `null`.<br>Унаследован от IPopupManager. |
| getOverlay() | vow.Promise | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от IPopupManager. |
| getOverlaySync() | IOverlay \| null | Возвращает оверлей, если тот существует.<br>Унаследован от IPopupManager. |
| getPosition() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от IPopupManager. |
| isOpen() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от IPopupManager. |
| open([position[, data[, options]]]) | vow.Promise | Открывает всплывающую подсказку в указанной позиции. |
| setData(data) | vow.Promise | Задаёт инфо-объекту новые данные.<br>Унаследован от IPopupManager. |
| setOptions(options) | vow.Promise | Задаёт инфо-объекту новые опции.<br>Унаследован от IPopupManager. |
| setPosition(position) | vow.Promise | Задаёт инфо-объекту новую позицию.<br>Унаследован от IPopupManager. |

## Описание методов

### open

```
{vow.Promise} open([position[, data[, options]]])
```

Открывает всплывающую подсказку в указанной позиции.

**Возвращает** объект-promise.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position | — | Тип: Number[]<br><br>Координаты открытия в глобальных пиксельных координатах. |
| data | — | Тип: Object<br><br>Данные. |
| options | — | Тип: Object<br><br>Опции. |

# hotspot.layer.Object

Расширяет **IHotspotLayerObject**.

Объект хотспотного слоя.

Конструктор | Поля | События | Методы

## Конструктор

```
hotspot.layer.Object(shape, feature, options)
```

Создаёт объект хотспотного слоя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| shape * | — | Тип: **IShape**<br><br>Фигура активной области. |
| feature * | — | Тип: Object<br><br>Описание объекта {@see GeoObject}. |
| options * | — | Тип: Object<br><br>Опции объекта. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchmove | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры, описывающей активную область. Экземпляр класса Event.<br>Унаследовано от IHotspotLayerObject. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getGeometry() | Object | Возвращает реальную геометрию объекта.<br>Унаследован от IHotspotLayerObject. |
| getHotspot() | IHotspot | Возвращает объект, описывающий активную область.<br>Унаследован от IHotspotLayerObject. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getId**() | Number | Возвращает идентификатор объекта.<br>Унаследован от **IHotspotLayerObject**. |
| **getProperties**() | Object | Возвращает данные объекта.<br>Унаследован от **IHotspotLayerObject**. |
| **setGeometry**(geometry) | | Задает реальную геометрию объекта.<br>Унаследован от **IHotspotLayerObject**. |
| **setId**(id) | | Задает идентификатор объекта.<br>Унаследован от **IHotspotLayerObject**. |
| **setProperties**(properties) | | Задает данные объекта.<br>Унаследован от **IHotspotLayerObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# hotspot.Layer

Расширяет **IChildOnMap**, **ICustomizable**.

Слой активных областей.

Конструктор | Поля | События | Методы

## Конструктор

```
hotspot.Layer(objectSource[, options])
```

Создает слой активных областей. Каждая отдельная область в составе слоя представляет собой объект — **hotspot.layer.Object**.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objectSource * | — | Тип: **IHotspotObjectSource**<br><br>Источник объектов слоя. |
| options | — | Тип: Object<br><br>Опции слоя. Опции для балунов **Balloon** и хинтов **Hint** хотспотного слоя нужно указывать с префиксами 'balloon' и 'hint'. |
| options.cursor | 'pointer' | Тип: String<br><br>Тип курсора. |
| options.dontChangeCursor | false | Тип: Boolean<br><br>Опция, позволяющая запретить изменение курсора при наведении на объект слоя. По умолчанию курсоры изменяются. |
| options.hasBalloon | true | Тип: Boolean<br><br>Флаг, показывающий, есть ли у слоя балуны. Если значение флага false, у слоя не будет создано поле .balloon. |
| options.hasHint | true | Тип: Boolean<br><br>Флаг, показывающий, есть ли у слоя хинты. Если значение флага false, у слоя не будет создано поле .hint. |
| options.interactivityModel | 'default#layer' | Тип: String<br><br>Модель интерактивности слоя. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.openBalloonOnClick | true | Тип: Boolean<br><br>Опция, позволяющая запретить открытие балуна при клике на хотспотном объекте. По умолчанию открытие балуна разрешено. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.openEmptyBalloon | false | Тип: String<br><br>Открывать балун с пустым содержимым. |
| options.openEmptyHint | false | Тип: String<br><br>Показывать всплывающую подсказку с пустым содержимым. |
| options.openHintOnHover | true | Тип: Boolean<br><br>Опция, позволяющая запретить показ хинта при наведении на хотспотный объект. По умолчанию показ хинтов разрешен. |
| options.pane | 'events' | Тип: **IEventPane**<br><br>Контейнер элементов карты, на котором слушаются события. |
| options.zIndex | — | Тип: Number<br><br>Значение zIndex слоя. Когда активный объект ищется в пределах одного слоя, то при одновременном попадании курсора в два объекта выбирается объект с наибольшим значением zIndex. При дальнейшей работе — например при определении, фигура какого слоя является более приоритетной, используется значение zIndex слоя. Обратите внимание, что zIndex слоя будет определять взаимное расположение объектов слоя и одиночных объектов, добавленных в определенные пейны карты. Например, если вы хотите наложить хотспотные объекты поверх площадных объектов, стоит задать хотспотному слою zIndex=201. См. **map.pane.Manager**. |

* Обязательный параметр/опция.

**Пример:**

```
// Создаем источник хотспотных данных. Не указываем значение ключа,
// в результате чего имя функции-обработчика (padding jsonp запроса) будет
генерироваться динамически.
var objectSource = new ymaps.hotspot.ObjectSource('tiles/?%c'),
    hotspotLayer = new ymaps.hotspot.Layer(objectSource, {
        zIndex: 100,
        // Разрешаем открывать балуны без содержимого.
        showEmptyBalloon: true
    });
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **balloon** | **hotspot.layer.Balloon** | Балун хотспотного слоя. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| hint | hotspot.layer.Hint | Хинт хотспотного слоя. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| addtomap | Слой добавлен на карту. |
| balloonclose | Закрытие балуна. Экземпляр класса Event. |
| balloonopen | Открытие балуна на хотспотном слое. Экземпляр класса Event. |
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• activeObject (hotspot.layer.Object) — объект слоя, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• activeObject (hotspot.layer.Object) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• activeObject (hotspot.layer.Object) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |
| hintclose | Закрытие хинта. Экземпляр класса Event. |
| hintopen | Открытие хинта на хотспотном слое. Экземпляр класса Event. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• activeObject (hotspot.layer.Object) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• activeObject (hotspot.layer.Object) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• activeObject (hotspot.layer.Object) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |
| mousemove | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• activeObject (hotspot.layer.Object) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |
| mouseup | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• activeObject (hotspot.layer.Object) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:<br>• activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:<br>• activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:<br>• activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| removefrommap | Слой удален с карты. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• activeObject (hotspot.layer.Object) — объект, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту. |
| getObjectInPosition(coords) | vow.Promise | Метод, позволяющий получить объект слоя в указанной географической точке на текущем масштабе. Если данные для тайла, в который попадает точка, не подгружены, вызов метода спровоцирует их загрузку. После этого поиск будет производиться по загруженным данным. |
| getObjectSource() | IHotspotObjectSource | Возвращает источник объектов слоя активных областей. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| update() | | Обновляет слой активных областей. После выполнения этой команды загруженные ранее объекты удаляются из контейнера и запрашиваются новые данные. |

## Описание полей

### balloon

```
{hotspot.layer.Balloon} balloon
```

Балун хотспотного слоя.

### hint

```
{hotspot.layer.Hint} hint
```

Хинт хотспотного слоя.

## Описание событий

### addtomap

Слой добавлен на карту.

### balloonclose

Закрытие балуна. Экземпляр класса **Event**.

### balloonopen

Открытие балуна на хотспотном слое. Экземпляр класса **Event**.

### click

Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- activeObject (**hotspot.layer.Object**) — объект слоя, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## contextmenu

Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## dblclick

Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## hintclose

Закрытие хинта. Экземпляр класса **Event**.

## hintopen

Открытие хинта на хотспотном слое. Экземпляр класса **Event**.

## mousedown

Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## mouseenter

Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## mouseleave

Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## mousemove

Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## mouseup

Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## multitouchend

Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## multitouchmove

Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## multitouchstart

Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## removefrommap

Слой удален с карты.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### wheel

Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- activeObject (**hotspot.layer.Object**) — объект, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## Описание методов

### getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

### getObjectInPosition

```
{vow.Promise} getObjectInPosition(coords)
```

Метод, позволяющий получить объект слоя в указанной географической точке на текущем масштабе. Если данные для тайла, в который попадает точка, не подгружены, вызов метода спровоцирует их загрузку. После этого поиск будет производиться по загруженным данным.

**Возвращает** объект-обещание, который будет зарезолвлен объектом слоя, в который попадает точка. Если точка попадает в несколько объектов слоя, вернется объект с максимальным значением zIndex.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coords* | — | Тип: Number[]<br><br>Географические координаты точки. |

* Обязательный параметр/опция.

### getObjectSource

```
{IHotspotObjectSource} getObjectSource()
```

**Возвращает** источник объектов слоя активных областей.

**Пример:**

```
// Обновление источника данных хотспотного слоя.
hotspotLayer.getObjectSource().setTileUrlTemplate('newSource/?%c');
hotspotLayer.update();
```

### update

```
update()
```

Обновляет слой активных областей. После выполнения этой команды загруженные ранее объекты удаляются из контейнера и запрашиваются новые данные.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# hotspot.ObjectSource

Расширяет **IHotspotObjectSource**.

Стандартная реализация интерфейса **IHotspotObjectSource**. Работает со стандартным форматом ответа сервера.

Конструктор | Поля | События | Методы

## Конструктор

```
hotspot.ObjectSource(tileUrlTemplate[, keyTemplate[, options]])
```

Возвращает источник данных для слоя активных областей.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tileUrlTemplate * | — | Тип: String \| Function<br><br>Шаблон URL данных, соответствующих тайлу. Поддерживаются специальные конструкции и возможность использования функции по аналогии с **Layer**. |
| keyTemplate | — | Тип: String \| Function<br><br>Шаблон идентификатора тайла. Используется для формирования padding в jsonp-запросе за данными. Задается по аналогии с предыдущим параметром. Все символы, не являющиеся буквой, цифрой или символом '_', будут заменены на '_'. Если параметр не задан, то padding запроса будет сгенерирован автоматически. Примеры преобразований при tileNumber=[3, 1], zoom=9:<br>• 'myCallback=%x' => 'myCallback_3'<br>• '%c' => 'x_3_y_1_z_9'<br>• 'callback2_%c' => 'callback2_x_3_y_1_z_9'<br>• 'callback%test' => 'callback_test' |
| options | — | Тип: Object<br><br>Опции источника данных. |
| options.bounds | null | Тип: Bounds<br><br>Область на карте, для которой есть данные или null, если данные есть везде. Опция для стандартной реализации метода restrict. |
| options.maxZoom | 0 | Тип: Bounds<br><br>Максимальный уровень масштабирования, для которого делаются запросы за данными. Опция для стандартной реализации метода restrict. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.minZoom | 0 | Тип: Bounds<br><br>Минимальный уровень масштабирования, для которого делаются запросы за данными. Опция для стандартной реализации метода restrict. |
| options.noCache | false | Тип: Boolean<br><br>false — использовать встроенный кэш браузера, true — не использовать (к каждому URL будет автоматически приписываться случайный GET-параметр, чтобы исключить кэширование тайлов браузером). |

* Обязательный параметр/опция.

**Пример:**

```
// Пример стандартного формата ответа сервера
hotspot_callback({
    // ответ может содержать поля "data" или "error".
    "data": {
        "type": "FeatureCollection",
        // Массив хотспотных объектов.
        "features": [{
            "type": "Feature",
            // Данные объекта.
            // Передаются в конструктор hotspot.layer.Object в качестве данных.
            "properties": {
                "hintContent": "Содержимое текстовой подсказки.",
                "balloonContentBody": "Содержимое балуна.",
                "balloonContentHeader": "Заголовок балуна.",
                "balloonContentFooter": "Нижняя часть балуна.",
                // Можно задавать свойство balloonContent вместо Body/Header/
Footer

                // Обязательное поле, описывающее hotspot.layer.Object.
                "HotspotMetaData": {
                    // Идентификатор хотспотного объекта.
                  // hotspot.layer.Object.getId по умолчанию возвращает значение
этого поля.
                    "id": 10469893,
                    "zIndex": 10,
                    // Данные, на основе которых создается геометрия
hotspot.layer.Object.
                    // Обязательное поле.
                   // Стандартная реализация позволяет передавать следующие типы
геометрий:
                    //    "Rectangle" - прямоугольник.
                    //      Будет создана геометрия geometry.pixel.Rectangle.
                    //    "Polygon" - многоконтурный многоугольник.
                    //      Будет создана геометрия geometry.pixel.Polygon.
                   //    "MultiPolygon" - сложная фигура, состоящая из нескольких
многоконтурных многоугольников.
                    //      Будет создана геометрия geometry.pixel.MultiPolygon.
                    //    "ConvexPolygon" - многоконтурный многоугольник. Все
контуры должны быть выпуклыми.
                    //       Работает быстрее, чем "Polygon".
                    //       Будет создана геометрия geometry.pixel.Polygon.
                    //    "MultiConvexPolygon" - сложная фигура, состоящая из
нескольких многоконтурных многоугольников.
                    //      Все контуры должны быть выпуклыми. Работает быстрее,
чем "MultiPolygon".
                    //      Будет создана геометрия geometry.pixel.MultiPolygon.
                    "RenderedGeometry": {
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
                        "type": "Polygon",
                        // координаты передаются в пиксельных координатах,
отсчитываемых от верхнего левого угла тайла.
                        "coordinates": [
                            // Первый контур многоугольника.
                            [
                                [-315, 280], [32, 442], [141, 208], [-206, 46],
[-315, 280]
                            ],
                            // Второй контур многоугольника.
                            [
                                [-186, 155], [-238, 265], [-152, 306], [-100,
196], [-186, 155]
                            ]
                        ]
                    }
                }
            },
            // Реальная геометрия объекта.
            // Необязательное поле.
            // Передается в hotspot.layer.Object и доступна через метод
hotspot.layer.Object.getGeometry.
            "geometry": {
                "type": "Polygon",
                "coordinates": [
                    // Первый контур многоугольника.
                    [
                        [29.176096525, 40.904183940],
                        [29.177027467, 40.903854324],
                        [29.177319900, 40.904329679],
                        [29.176389040, 40.904659406],
                        [29.176096525, 40.904183940]
                    ],
                    // Второй контур многоугольника.
                    [
                        [29.176442530, 40.904437683],
                        [29.176303055, 40.904213830],
                        [29.176533525, 40.904131168],
                        [29.176673032, 40.904355010],
                        [29.176442530, 40.904437683]
                    ]
                ]
            }
        },
        {
            "type": "Feature",
            "properties": {
                // описание следующего хотспотного объекта...
            }
        }]
    }
});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| cancelLastRequest() | | Отменяет последний запрос за данными.<br>Унаследован от IHotspotObjectSource. |
| getKey(tileNumber, zoom) | String | Возвращает имя callback-функции (padding) в jsonp-запросе, если оно статическое, либо null, если можно использовать динамический jsonp-callback. Шаблоны поддерживают те же подстановки, что и в шаблоне url тайла. Все символы, не являющиеся буквой, цифрой или символом '_', будут заменены на '_'. |
| getKeyTemplate() | String | Возвращает шаблон идентификатора тайла. |
| getTileUrl(tileNumber, zoom) | String | Возвращает URL данных для конкретного тайла. |
| getTileUrlTemplate() | String | Возвращает шаблон URL данных для тайла. |
| parseResponse(layer, res, callback, tileNumber, zoom) | | Разбирает ответ сервера. Возвращает в callback массив объектов-активных областей, сформированных по json-описанию. |
| requestObjects(layer, tileNumber, zoom, callback) | | Строит массив объектов IHotspotLayerObject, соответствующих определенному слою, тайлу и коэффициенту масштабирования карты, и передает его в функцию callback.<br>Унаследован от IHotspotObjectSource. |
| restrict(layer, tileNumber, zoom) | Boolean | Метод, который вызывается перед тем, как отправить запрос за данными для тайла. Если метод возвращает true, то запрос на сервер посылаться не будет, и в качестве ответа вернется пустой массив объектов. |
| setKeyTemplate(template) | | Задает новый шаблон идентификатора тайла. |
| setTileUrlTemplate(template) | | Задает новый шаблон URL данных для тайла. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### getKey

```
{String} getKey(tileNumber, zoom)
```

Возвращает имя callback-функции (padding) в jsonp-запросе, если оно статическое, либо null, если можно использовать динамический jsonp-callback. Шаблоны поддерживают те же подстановки, что и в шаблоне url тайла. Все символы, не являющиеся буквой, цифрой или символом '_', будут заменены на '_'.

**Возвращает** идентификатор тайла. Используется при формировании padding в jsonp-запросе за данными.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tileNumber* | — | Тип: Number[]<br><br>Номер тайла (тайловые координаты). |
| zoom* | — | Тип: Integer<br><br>Уровень масштаба. |

* Обязательный параметр/опция.

### getKeyTemplate

```
{String} getKeyTemplate()
```

**Возвращает** шаблон идентификатора тайла.

### getTileUrl

```
{String} getTileUrl(tileNumber, zoom)
```

**Возвращает** URL данных для конкретного тайла.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tileNumber* | — | Тип: Number[]<br><br>Номер тайла (тайловые координаты). |
| zoom* | — | Тип: Integer<br><br>Уровень масштабирования. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
var hotspotObjectSource = new ymaps.hotspot.ObjectSource('dataSource/?%c');
hotspotObjectSource.getTileUrl = function (tileNumber, zoom) {
    if (zoom > 10) {
        // Для крупных масштабов используем выставленный путь к данным.
        return ymaps.hotspot.ObjectSource.prototype.call(this, tileNumber,
zoom);
    } else {
        // Для мелких масштабов используем другой путь.
        return 'otherSource/getHotspots.xml?z=' + zoom + '&x=' + tileNumber[0] +
'&y=' + tileNumber[1]';
    }
};
```

## getTileUrlTemplate

```
{String} getTileUrlTemplate()
```

**Возвращает** шаблон URL данных для тайла.

## parseResponse

```
parseResponse(layer, res, callback, tileNumber, zoom)
```

Разбирает ответ сервера. Возвращает в callback массив объектов-активных областей, сформированных по json-описанию.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| layer* | — | Тип: **hotspot.Layer**<br><br>Слой, которому принадлежат объекты. |
| res* | — | Тип: Object<br><br>Ответ сервера. |
| callback* | — | Тип: Function<br><br>Функция-обработчик. |
| tileNumber* | — | Тип: Number[]<br><br>Номер тайла, для которого пришел ответ. |
| zoom* | — | Тип: Number<br><br>Коэффициент масштабирования, для которого пришел ответ. массива объектов. |

* Обязательный параметр/опция.

## restrict

```
{Boolean} restrict(layer, tileNumber, zoom)
```

Метод, который вызывается перед тем, как отправить запрос за данными для тайла. Если метод возвращает true, то запрос на сервер посылаться не будет, и в качестве ответа вернется пустой массив объектов. Стандартная реализация метода проверяет, лежит ли параметр zoom в пределах [options.minZoom, options.maxZoom]. А также проверяет, попадает ли центр запрашиваемого тайла в указанные options.bounds. Если options.bounds=null, то проверка для этого параметра не осуществляется.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** true — тайл вышел за пределы области данных (для данного тайла данных нет), false — не вышел (данные есть).

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| layer* | — | Тип: **hotspot.Layer**<br><br>Слой активных областей. |
| tileNumber* | — | Тип: Number[]<br><br>Номер тайла. |
| zoom* | — | Тип: Integer<br><br>Коэффициент масштабирования. |

* Обязательный параметр/опция.

**Пример:**

```
// Пример переопределения метода restrict
// Допустим, что данные есть только для Мурманска и Новосибирска.
var myMap = new ymaps.Map('map', {center: [32.5, 68.9] , zoom: 9}),
    geoBounds = [
         [[31.729958, 69.369182], [34.203324, 68.666473]], // Мурманск
         [[82.179084, 55.341085], [83.725642, 54.670738]]  // Новосибирск
    ],
    projection = myMap.options.get('projection');

var myHotspotSource = new ymaps.hotspot.ObjectSource('http://www.myDomain.ru/
tiles/?%c', '%c');
myHotspotSource.restrict = function(layer, tileNumber, zoom) {

   // Вычислим пиксельные границы городов для данного уровня масштабирования.
   var boundsFromPoints = ymaps.util.bounds.fromPoints,
       toGlobalPixels = projection.toGlobalPixels,
       pixelBounds = [
           boundsFromPoints(
               toGlobalPixels(geoBounds[0][0]], zoom),
               toGlobalPixels(geoBounds[0][1]], zoom)
           ),
           boundsFromPoints(
               toGlobalPixels(geoBounds[1][0]], zoom),
               toGlobalPixels(geoBounds[1][1]], zoom)
           )
       ],

       // Вычислим пиксельные границы тайла
       leftTop = [tileNumber[0] * 256, tileNumber[1] * 256],
       tileBounds = [leftTop, [leftTop[0] + 256, leftTop[1] + 256]],
       intersects = ymaps.util.bounds.intersects;
   // Если пиксельные границы тайла пересекаются с пиксельными границами
указанных областей,
   // то нужно будет отправить запрос за данными
   if (intersects(pixelBounds[0], tileBounds) || (intersects(pixelBounds[1],
tileBounds)) {
         return false;
   }

   // Для всех остальных тайлов данных у этого источника нет.
   return true;
}
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## setKeyTemplate

```
setKeyTemplate(template)
```

Задает новый шаблон идентификатора тайла.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| template* | — | Тип: String<br><br>Шаблон идентификатора. |

* Обязательный параметр/опция.

## setTileUrlTemplate

```
setTileUrlTemplate(template)
```

Задает новый шаблон URL данных для тайла.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| template* | — | Тип: String<br><br>Шаблон URL |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# interactivityModel

## interactivityModel.storage

Статический объект.

Экземпляр класса **util.Storage**

Хранилище моделей интерактивности. Модели интерактивности позволяют объектам по-разному обрабатывать DOM-события. Список доступных ключей моделей интерактивности:

- 'default#opaque' — объект генерирует все DOM-события и не прокидывает их на карту. Поведения карты не будут работать при наведении или клике на объекты с данной моделью интерактивности.
- 'default#geoObject' — объект генерирует все DOM-события. На карту прокидываются события 'wheel', 'mousedown', 'dblclick', 'contextmenu', 'multitouchstart', 'multitouchmove' и 'multitouchend'. Если на карте включены поведения 'scrollZoom', 'dblClickZoom' или 'magnifier', они будут работать через объекты с данной моделью интерактивности, в отличие от объектов с моделью 'default#opaque'.
- 'default#layer' — объект генерирует все DOM-события. На карту прокидываются события 'wheel', 'mousedown', 'contextmenu', 'multitouchstart', 'multitouchmove' и 'multitouchend'. Если на карте включены поведения 'scrollZoom', 'drag' или 'magnifier', они будут работать через объекты с данной моделью интерактивности.
- 'default#transparent' — объект генерирует все DOM-события, а затем прокидывает их на карту.
- 'default#silent' — объект не генерирует DOM-события, но прокидывает их на карту.

Методы

```
interactivityModel.storage()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# layer

## layer.storage

Статический объект.

Экземпляр класса **util.Storage**

Хранилище слоев.

Методы

```
layer.storage()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |

# layer.tile

## layer.tile.CanvasTile

Расширяет **ICanvasTile**.

Картиночный canvas-тайл. Умеет отрисовывать указанное изображение через метод drawImage у 2d контекста canvas-элемента.

Конструктор | Поля | События | Методы

### Конструктор

```
layer.tile.CanvasTile(url[, options])
```

Создаёт картиночный canvas-тайл. Доступен по ключу "default#canvas" в хранилище классов тайлов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| url * | — | Тип: String<br><br>URL изображения. |
| options | — | Тип: Object<br><br>Опции. |
| options.notFoundTile | null | Тип: String \| null |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опция, указывающая по какому url нужно загрузить картинку, если картинка для тайла не загрузилась. Если значение null, то будет показан стандартный тайл с текстовым сообщением. Для прозрачных тайлов опция notFoundTile не применяется, и вместо незагруженных тайлов ничего показываться не будет. |
| `options.tileAnimationDuration` | — | Тип: Number<br><br>Продолжительность анимации прозрачности изображения. Значение по умолчанию зависит от браузера. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий тайла.<br>Унаследовано от **ICanvasTile**. |

## События

| Имя | Описание |
|---|---|
| **ready** | Событие готовности тайла.<br>Унаследовано от **ICanvasTile**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает тайл.<br>Унаследован от **ICanvasTile**. |
| **isReady**() | Boolean | Проверяет готовность тайла.<br>Унаследован от **ICanvasTile**. |
| **renderAt**(context, canvasSize, bounds[, animate]) | | Отрисовывает картиночный тайл на 2d-контексте объекта canvas.<br>Унаследован от **ICanvasTile**. |

# layer.tile.DomTile

Расширяет **IDomTile**.

Картиночный DOM-тайл. Умеет отрисовывать указанное изображение через css свойство background у DOM-элемента.

Конструктор | Поля | События | Методы


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Конструктор

```
layer.tile.DomTile(url[, options])
```

Создаёт картиночный dom-тайл. Доступен по ключу "default#dom" в хранилище классов тайлов. Создаёт картиночный DOM-тайл.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| `url` * | — | Тип: String<br><br>URL изображения. |
| `options` | — | Тип: Object<br><br>Опции. |
| `options.notFoundTile` | null | Тип: String \| null<br><br>Опция, указывающая по какому url нужно загрузить картинку, если картинка для тайла не загрузилась. Если значение null, то будет показан стандартный тайл с текстовым сообщением. Для прозрачных тайлов опция notFoundTile не применяется, и вместо незагруженных тайлов ничего показываться не будет. |
| `options.tileAnimationDuration` | — | Тип: Number<br><br>Продолжительность анимации прозрачности изображения при рисовании в мс (применяется только в браузерах, поддерживающих CSS Transition для свойства opacity). Значение по умолчанию зависит от браузера. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий тайла.<br><br>Унаследовано от **IDomTile**. |

## События

| Имя | Описание |
|---|---|
| **ready** | Событие готовности тайла.<br><br>Унаследовано от **IDomTile**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает тайл. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IDomTile**. |
| **isReady**() | Boolean | Проверяет готовность тайла.<br>Унаследован от **IDomTile**. |
| **renderAt**(context, clientBounds, animate) | | Добавляет тайл в родительский HTML-элемент.<br>Унаследован от **IDomTile**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# layer.tileContainer

## layer.tileContainer.CanvasContainer

Расширяет **IChildOnMap**.

Контейнер тайлов на canvas

Конструктор | Поля | События | Методы

### Конструктор

```
layer.tileContainer.CanvasContainer(layer[, options])
```

Создаёт контейнер тайлов на canvas. Доступен по ключу "default#canvas" в хранилище классов тайловых контейнеров.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| layer * | — | Тип: **ILayer**<br><br>Слой. |
| options | — | Тип: Object<br><br>Опции контейнера. |
| options.notFoundTile | null | Тип: String \| null<br><br>Опция, указывающая по какому url нужно загрузить картинку, если картинка для тайла не загрузилась. Если значение null, то будет показан стандартный тайл с текстовым сообщением. Для прозрачных тайлов опция notFoundTile не применяется, и вместо незагруженных тайлов ничего показываться не будет. |
| options.tileClass | 'default#canvas' | Тип: **ICanvasTile**<br><br>Класс тайлов, используемых контейнером. Должен реализовывать интерфейс **ICanvasTile**. |
| options.tileTransparent | false | Тип: Boolean<br><br>Флаг, показывающий, являются ли тайлы контейнера прозрачными. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| getTile(tileNumber, tileZoom, priority) | ICanvasTile | Фабричная функция создания тайлов. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание методов

### getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

### getTile

```
{ICanvasTile} getTile(tileNumber, tileZoom, priority)
```

Фабричная функция создания тайлов.

**Возвращает** экземпляр тайла.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tileNumber* | — | Тип: Number[]<br>Номер тайла. |
| tileZoom* | — | Тип: Number<br>Масштаб тайла. |
| priority* | — | Тип: Number |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Приоритет загрузки. |

* Обязательный параметр/опция.

# layer.tileContainer.DomContainer

Расширяет **IChildOnMap**.

Контейнер тайлов типа IDomTile

Конструктор | Поля | События | Методы

## Конструктор

```
layer.tileContainer.DomContainer(layer[, options])
```

Создаёт контейнер DOM-тайлов. Доступен по ключу "default#dom" в хранилище классов тайловых контейнеров.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| layer * | — | Тип: **ILayer**<br><br>Слой. |
| options | — | Тип: Object<br><br>Опции контейнера. |
| options.notFoundTile | null | Тип: String \| null<br><br>Опция, указывающая по какому url нужно загрузить картинку, если картинка для тайла не загрузилась. Если значение null, то будет показан стандартный тайл с текстовым сообщением. Для прозрачных тайлов опция notFoundTile не применяется, и вместо незагруженных тайлов ничего показываться не будет. |
| options.tileClass | 'default#dom' | Тип: **IDomTile**<br><br>Класс тайлов, используемых контейнером. Должен реализовывать интерфейс **IDomTile**. |
| options.tileTransparent | false | Тип: Boolean<br><br>Флаг, показывающий, являются ли тайлы контейнера прозрачными. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **ready** | Событие готовности всех тайлов. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getMap**() | **Map** | Возвращает ссылку на карту. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getTile**(tileNumber, tileZoom, priority) | **IDomTile** | Фабричная функция создания тайлов. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание событий

### ready

Событие готовности всех тайлов.

## Описание методов

### getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

### getTile

```
{IDomTile} getTile(tileNumber, tileZoom, priority)
```

Фабричная функция создания тайлов.

**Возвращает** экземпляр тайла.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tileNumber* | — | Тип: Number[]<br><br>Номер тайла. |
| tileZoom* | — | Тип: Number<br><br>Масштаб тайла. |
| priority* | — | Тип: Number<br><br>Приоритет загрузки. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# Layer

Расширяет **ILayer**, **IParentOnMap**, **IPositioningContext**.

Тайловый слой. Позволяет отображать на карте подложку, состоящую из тайлов.

Конструктор | Поля | События | Методы

## Конструктор

```
Layer(tileUrlTemplate[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tileUrlTemplate * | — | Тип: String \| Function<br><br>Строковый шаблон URL тайла, либо функция, которая генерирует URL тайла. Для строкового шаблона поддерживается подстановка:<br>• %c — заменяется на x=номер[0]&y=номер[1]&z=уровень масштабирования;<br>• %x — заменяется на номер[0]<br>• %y — заменяется на номер[1]<br>• %z — заменяется на уровень масштабирования<br>• %l — заменяется на lang=язык;<br>• %d или %d\|n — заменяется на число от 1 до n, в зависимости от номера тайла. n — количество доменов. Используется для распределения нагрузки между несколькими доменами. В качестве n нужно указывать степени двойки (2, 4, 16 и т.д.). Если шаблон имеет вид %d, то n=4.<br><br>Функция-шаблон получает на вход три параметра:<br>• tileNumber — массив из двух чисел, номера тайла по x и y;<br>• tileZoom — уровень масштабирования;<br>• Возвращает строку-URL. |
| options | — | Тип: Object<br><br>Опции. |
| options.brightness | 0.5 | Тип: Number<br><br>Яркость слоя. Задается числом от 0 до 1. 0 соответствует черному цвету, 1 — белому. |
| options.notFoundTile | null | Тип: String \| null |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опция, указывающая по какому url нужно загрузить картинку, если картинка для тайла не загрузилась. Если значение null, то будет показан стандартный тайл с текстовым сообщением. Для прозрачных тайлов опция notFoundTile не применяется, и вместо незагруженных тайлов ничего показываться не будет. |
| options.pane | 'ground' | Тип: **IPane**<br><br>Пейн слоя. |
| options.projection | — | Тип: Object<br><br>Проекция слоя. |
| options.tileSize | [256, 256] | Тип: Number[]<br><br>Размер тайлов слоя. |
| options.tileTransparent | false | Тип: Boolean<br><br>Флаг, показывающий, являются ли тайлы слоя прозрачными. |
| options.zIndex | constants.zIndex.layer | Тип: Number<br><br>Z-index слоя в контейнере слоёв. |

* Обязательный параметр/опция.

**Пример:**

```
// Добавим на карту слой OSM
map.layers.add(new ymaps.Layer(
'http://tile.openstreetmap.org/%z/%x/%y.png', {
    projection: ymaps.projection.sphericalMercator
}));
map.copyrights.add('&copy; OpenStreetMap contributors, CC-BY-SA');
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **brightnesschange** | Событие изменения яркости слоя.<br><br>Унаследовано от **ILayer**. |
| **copyrightschange** | Событие изменения доступной информации об авторских правах.<br><br>Унаследовано от **ILayer**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **zoomrangechange** | Событие изменения доступной информации о диапазоне уровней масштабирования.<br>Унаследовано от **ILayer**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **clientPixelsToNumber**(clientPixelPoint, tileZoom) | Number[] | Возвращает номер тайла, в который попадает указанная точка при заданном уровне масштаба тайлов. |
| **fromClientPixels**(clientPixelPoint) | Number[] | Преобразует клиентские пиксельные координаты в глобальные.<br>Унаследован от **IPositioningContext**. |
| **getBrightness**() | Number | Необязательный метод. Возвращает яркость слоя. В зависимости от суммарной яркости слоев, добавленных на карту, выбирается цвет логотипа и копирайтов на карте.<br>Унаследован от **ILayer**. |
| **getCopyrights**(coords, zoom) | **vow.Promise** | Необязательный метод. Запрашивает информацию об авторских правах в указанной точке с указанным коэффициентом масштабирования. Возвращает объект-promise, который будет подтверждён и передаст в качестве результата массив строк или DOM-элементов с информацией об авторских правах.<br>Унаследован от **ILayer**. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getPane**() | **IPane** | Возвращает контейнер, в котором находится слой. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IChildOnMap**. |
| **getTileSize**(zoom) | Number[] | Возвращает размеры тайлов в зависимости от масштаба. |
| **getTileUrl**(tileNumber, tileZoom) | String \| null | Возвращает URL тайла по его номеру и уровню масштабирования или null если для запрашиваемого участка нет данных. |
| **getTileUrlTemplate**() | String \| Function | Возвращает строковый шаблон URL тайла или функцию, которая его генерирует. |
| **getZoom**() | Number | Возвращает текущий уровень масштабирования, в котором работает контекст позиционирования.<br>Унаследован от **IPositioningContext**. |
| **getZoomRange**(point) | **vow.Promise** | Необязательный метод. Определяет доступный в указанной точке диапазон уровней масштабирования. Если данные есть, то возвращенный объект-promise будет подтвержден и передаст в качестве результата массив из двух чисел — минимальный и максимальный доступный в точке уровень масштабирования. Если данных нет, то promise отклоняется с указанием ошибки.<br>Унаследован от **ILayer**. |
| **numberToClientBounds**(tileNumber, tileZoom) | Number[][] | Преобразует номер и уровень масштаба тайла в занимаемую тайлом область в клиентских координатах родительского контейнера. |
| **restrict**(number, tileZoom) | Integer[] \| null | Накладывает ограничения на зону видимости тайлов (в том числе зацикленность карты по осям x, y). |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |
| **setTileUrlTemplate**(tileUrlTemplate) | | |
| **toClientPixels**(globalPixelPoint) | Number[] | Преобразует глобальные пиксельные координаты в клиентские.<br>Унаследован от **IPositioningContext**. |
| **update**() | | Удаляет старые тайлы и запрашивает новые. |

## Описание методов

### clientPixelsToNumber

```
{Number[]} clientPixelsToNumber(clientPixelPoint, tileZoom)
```

**Возвращает** номер тайла, в который попадает указанная точка при заданном уровне масштаба тайлов.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| clientPixelPoint* | — | Тип: Number<br><br>Точка в клиентских пиксельных координатах. |
| tileZoom* | — | Тип: Number<br><br>Уровень масштаба тайла. |

* Обязательный параметр/опция.

## getPane

```
{IPane} getPane()
```

Возвращает контейнер, в котором находится слой.

**Возвращает** контейнер.

## getTileSize

```
{Number[]} getTileSize(zoom)
```

Возвращает размеры тайлов в зависимости от масштаба.

**Возвращает** размеры тайла по горизонтали и вертикали для указанного коэффициента масштабирования.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| zoom* | — | Тип: Number<br><br>Значение коэффициента мастабирования. |

* Обязательный параметр/опция.

**Пример:**

```
// Будем показывать тайлы для большего уровня масштабирования,
// растянутые в два раза до размера 512x512 пикселей
// Например, для экономии трафика.
var layer = new ymaps.Layer('', {
    projection: ymaps.projection.sphericalMercator
});
layer.getTileUrl = function (tileNumber, zoom) {
    return [
        'http://tile.openstreetmap.org',
        Math.max(zoom - 1, 0), tileNumber[0], tileNumber[1]
    ].join('/') + '.png';
}
layer.getTileSize = function (zoom) {
    if (zoom == 0) {
        return [256, 256];
    }
    return [512, 512];
}
map.copyrights.add('&copy; OpenStreetMap contributors, CC-BY-SA');
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getTileUrl

```
{String | null} getTileUrl(tileNumber, tileZoom)
```

**Возвращает** URL тайла по его номеру и уровню масштабирования или null если для запрашиваемого участка нет данных.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tileNumber* | — | Тип: |
| tileZoom* | — | Тип: |

* Обязательный параметр/опция.

**Пример:**

```
// Определим свою функцию для формирования URL тайла
var layer = new ymaps.Layer('');
layer.getTileUrl = function (tileNumber, zoom) {
    return [
        'http://tile.openstreetmap.org',
        zoom, tileNumber[0], tileNumber[1]
    ].join('/') + '.png';
}
```

## getTileUrlTemplate

```
{String | Function} getTileUrlTemplate()
```

**Возвращает** строковый шаблон URL тайла или функцию, которая его генерирует.

## numberToClientBounds

```
{Number[][]} numberToClientBounds(tileNumber, tileZoom)
```

Преобразует номер и уровень масштаба тайла в занимаемую тайлом область в клиентских координатах родительского контейнера.

**Возвращает** область в клиентских пиксельных координатах.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tileNumber* | — | Тип: Integer[]<br><br>Номер тайла. |
| tileZoom* | — | Тип: Integer<br><br>Уровень масштаба тайла. |

* Обязательный параметр/опция.

## restrict

```
{Integer[] | null} restrict(number, tileZoom)
```

Накладывает ограничения на зону видимости тайлов (в том числе зацикленность карты по осям x, y).

**Возвращает** новый номер тайла с учётом ограничений либо null, если тайл не входит в зону видимости.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| number* | — | Тип: Integer[]<br><br>Номер тайла. |
| tileZoom* | — | Тип: Integer<br><br>Уровень масштабирования тайла. |

* Обязательный параметр/опция.

### setTileUrlTemplate

```
setTileUrlTemplate(tileUrlTemplate)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tileUrlTemplate* | — | Тип: String \| Function<br><br>Строковый шаблон URL тайла, либо функция, которая его генерирует. |

* Обязательный параметр/опция.

### update

```
update()
```

Удаляет старые тайлы и запрашивает новые.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| updateBounds* | — | Тип: |

* Обязательный параметр/опция.

## LayerCollection

Расширяет **ILayer**, **IMapObjectCollection**.

Коллекция слоев.

Конструктор | Поля | События | Методы

### Конструктор

```
LayerCollection()
```

Коллекция слоев.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| add | Произошло добавление дочернего объекта.<br>Унаследовано от ICollection. |
| brightnesschange | Событие изменения яркости слоя.<br>Унаследовано от ILayer. |
| copyrightschange | Событие изменения доступной информации об авторских правах.<br>Унаследовано от ILayer. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| remove | Произошло удаление дочернего объекта.<br>Унаследовано от ICollection. |
| zoomrangechange | Событие изменения доступной информации о диапазоне уровней масштабирования.<br>Унаследовано от ILayer. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| add(child) | LayerCollection | Добавляет в коллекцию дочерний объект. |
| each(callback[, context]) | | Перебирает все элементы коллекции, вызывая для каждого функцию-обработчик. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getBrightness**() | Number | Возвращает яркость слоя в виде числа от 0 до 1. |
| **getCopyrights**([coords[, zoom]]) | **vow.Promise** | Запрашивает информацию об авторских правах в указанной точке с указанным коэффициентом масштабирования. Возвращает объект-promise, который будет подтверждён и передаст в качестве результата массив строк или DOM-элементов с информацией об авторских правах. Если точка и коэффициент масштабирования не указаны, будут использованы центр и коэффициент масштабирования карты. |
| **getIterator**() | **IIterator** | Возвращает итератор по коллекции.<br>Унаследован от **ICollection**. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getZoomRange**() | **vow.Promise** | Определяет доступный в указанной точке диапазон уровней масштабирования. Если данные есть, то возвращенный объект-promise будет подтвержден и передаст в качестве результата массив из двух чисел — минимальный и максимальный доступный в точке уровень масштабирования. Если данных нет, то promise отклоняется с указанием ошибки. Если у коллекции нет ни одного потомка, предоставляющего информацию о диапазоне уровней масштабирования, то promise будет отклонён с сообщением 'noProvider'. |
| **remove**(child) | **LayerCollection** | Удаляет из коллекции дочерний объект. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание методов

### add

```
{LayerCollection} add(child)
```

Добавляет в коллекцию дочерний объект.

**Возвращает** ссылку на себя.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **ILayer** \| String<br><br>Слой (ключ хранилища **layer.storage** или экземпляр класса **ILayer**), который нужно добавить. |

* Обязательный параметр/опция.

**Пример:**

```
// Допустим, мы хотим добавить в свою коллекцию несколько слоёв.
var layerCollection = new ymaps.LayerCollection(),
    customLayer = new ymaps.Layer(
        'http://tile.openstreetmap.org/%z/%x/%y.png', {
        projection: ymaps.projection.sphericalMercator
    }),
    // Слой можно задать ключём из layer.storage.
    satelliteLayer = 'yandex#satellite';
// Добавим слои в нашу коллекцию.
layerCollection
    .add(customLayer)
    .add(satelliteLayer);
```

## each

```
each(callback[, context])
```

Перебирает все элементы коллекции, вызывая для каждого функцию-обработчик.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик. |
| context | — | Тип: Object<br><br>Контекст выполнения функции. |

* Обязательный параметр/опция.

## getBrightness

```
{Number} getBrightness()
```

**Возвращает** яркость слоя в виде числа от 0 до 1.

## getCopyrights

```
{vow.Promise} getCopyrights([coords[, zoom]])
```

Запрашивает информацию об авторских правах в указанной точке с указанным коэффициентом масштабирования. Возвращает объект-promise, который будет подтверждён и передаст в качестве результата массив строк или DOM-элементов с информацией об авторских правах. Если точка и коэффициент масштабирования не указаны, будут использованы центр и коэффициент масштабирования карты.

**Возвращает** объект-promise.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coords | — | Тип: Number[]<br><br>Точка на карте, для которой запрашивается информация об авторских правах. |
| zoom | — | Тип: Number<br><br>Уровень масштабирования, для которого запрашивается информация об авторских правах. |

**Пример:**

```
// Пусть у нас есть сервис, умеющий по координатам
// и уровню масштабирования возвращать копирайты
myLayer.getCopyrights = function (coords, zoom) {
    var deferred = ymaps.vow.defer();
    $.ajax('url/to/copyrights/provider?ll=' +
        (coords || map.getCenter()).join(',') + '&z=' +
        (zoom || map.getZoom()),
    function (res) {
        deferred.resolve(res || []);
    });
    return deferred.promise();
};
```

## getZoomRange

```
{vow.Promise} getZoomRange()
```

Определяет доступный в указанной точке диапазон уровней масштабирования. Если данные есть, то возвращенный объект-promise будет подтвержден и передаст в качестве результата массив из двух чисел — минимальный и максимальный доступный в точке уровень масштабирования. Если данных нет, то promise отклоняется с указанием ошибки. Если у коллекции нет ни одного потомка, предоставляющего информацию о диапазоне уровней масштабирования, то promise будет отклонён с сообщением 'noProvider'.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coords* | — | Тип: |

* Обязательный параметр/опция.

**Пример:**

```
// Допустим, наш слой отрисован для масштабов 2-15 по всей земле
myLayer.getZoomRange = function () {
    return ymaps.vow.resolve([2, 15]);
}
```

## remove

```
{LayerCollection} remove(child)
```

Удаляет из коллекции дочерний объект.

**Возвращает** ссылку на себя.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **ILayer** \| String<br><br>Слой (строка-ключ из **layer.storage** или экземпляр класса **ILayer**), который нужно удалить. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# layout

## layout.Image

Расширяет **ILayout**.

Класс для создания макетов, содержащих картинку.

Конструктор | Поля | События | Методы

### Конструктор

```
layout.Image(data)
```

Создает экземпляр макета-картинки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data * | — | Тип: **ILayout**<br><br>Данные макета. |
| data.options | — | Тип: **ILayout**<br><br>Опции макета. |
| data.options.imageClipRect | — | Тип: Number[][]<br><br>Координаты отображаемой области исходного изображения в пикселях. |
| data.options.imageHref | — | Тип: String<br><br>URL графического файла. |
| data.options.imageOffset | — | Тип: Number[]<br><br>Смещение картинки относительно точки привязки. |
| data.options.imageSize | — | Тип: Number[]<br><br>Размеры слоя с картинкой. |
| data.options.shape | — | Тип: **IShape** \| Object \| null<br><br>Фигура активной области. Может быть задана экземпляром класса, реализующего интерфейс IShape или JSON-описанием пиксельной геометрии иконки. Если не задана, то автоматически будет рассчитана прямоугольная фигура на основе размера и смещения картинки. Координаты геометрии фигуры отсчитываются от точки привязки. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Создадим круглую метку радиусом 20 пикселей.
var placemark = new ymaps.Placemark([59.936952, 30.343334], null, {
        iconLayout: 'default#image',
        iconImageHref: './images/roundImage.png',
        iconImageSize: [40, 40],
        iconImageOffset: [-20, -20],
        // Определим интерактивную область над картинкой.
        iconShape: {
            type: 'Circle',
            coordinates: [0, 0],
            radius: 20
        },
    });
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| emptinesschange | Изменение признака пустоты макета. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mousemove | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **parentelementchange** | Изменение родительского элемента. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **shapechange** | Изменение фигуры области охватывающей макет. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Деструктор. Вызывается при завершении работы с макетом.<br>Унаследован от **ILayout**. |
| **getData**() | Object | Возвращает объект данных макета.<br>Унаследован от **ILayout**. |
| **getParentElement**() | HTMLElement | Возвращает родительский html-элемент.<br>Унаследован от **ILayout**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.<br>Унаследован от **ILayout**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **ILayout**. |
| **setData**(data) | | Задает данные макета.<br>Унаследован от **ILayout**. |
| **setParentElement**(parent) | | Добавляет макет в DOM-дерево.<br>Унаследован от **ILayout**. |

# layout.ImageWithContent

Расширяет **layout.Image**.

Класс для создания макетов, состоящих из картинки и содержимого.

Конструктор | Поля | События | Методы

## Конструктор

```
layout.ImageWithContent(data)
```

Создает экземпляр макета-картинки с содержимым.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data * | — | Тип: **ILayout**<br>Данные макета. |
| data.options | — | Тип: **ILayout** |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опции макета. |
| data.options.contentLayout | — | Тип: Function \| String<br><br>Макет содержимого. (Тип: конструктор объекта с интерфейсом ILayout или его ключ в хранилище). |
| data.options.contentOffset | — | Тип: Number[]<br><br>Смещение слоя с содержимым относительно слоя с картинкой. |
| data.options.contentSize | — | Тип: Number[]<br><br>Размеры слоя с содержимым. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | IEventManager | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты макета. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **parentelementchange** | Изменение родительского элемента. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **shapechange** | Изменение фигуры области охватывающей макет. Экземпляр класса **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **ILayout**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Деструктор. Вызывается при завершении работы с макетом.<br>Унаследован от **ILayout**. |
| **getData**() | Object | Возвращает объект данных макета.<br>Унаследован от **ILayout**. |
| **getParentElement**() | HTMLElement | Возвращает родительский html-элемент.<br>Унаследован от **ILayout**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.<br>Унаследован от **ILayout**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **ILayout**. |
| **setData**(data) | | Задает данные макета.<br>Унаследован от **ILayout**. |
| **setParentElement**(parent) | | Добавляет макет в DOM-дерево.<br>Унаследован от **ILayout**. |

# layout.storage

Статический объект.

Экземпляр класса **util.AsyncStorage**

Хранилище классов лейаутов.

Методы

```
layout.storage()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |
| **define**(key[, depends, resolveCallback[, context]]) | **util.AsyncStorage** | Определение в хранилище асинхронного значения. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **isDefined**(key) | Boolean | Проверка доступности ключа в хранилище. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |
| **require**(keys[, successCallback[, errorCallback[, context]]]) | **vow.Promise** | Асинхронный запрос на получение значений из хранилища. |

# layout.templateBased

## layout.templateBased.Base

Расширяет **ILayout**.

Базовый класс макета на основе шаблонов. Этот класс используется фабрикой макетов как базовый для создания пользовательских макетов.

**См.: templateLayoutFactory**

Конструктор | Поля | События | Методы

### Конструктор

```
layout.templateBased.Base(data)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data * | — | Тип: Object<br><br>Набор разнородных данных, на основе которых строится макет. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты макета. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **parentelementchange** | Изменение родительского элемента. Экземпляр класса **Event**.<br>Унаследовано от **ILayout**. |
| **shapechange** | Изменение фигуры области охватывающей макет. Экземпляр класса **Event**.<br>Унаследовано от **ILayout**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **build**() | | Строит экземпляр макета на основе шаблона и добавляет его в родительский HTML-элемент. |
| **clear**() | | Удаляет содержимое макета из DOM. |
| **destroy**() | | Деструктор. Вызывается при завершении работы с макетом.<br>Унаследован от **ILayout**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getData**() | Object | Возвращает объект данных макета.<br>Унаследован от **ILayout**. |
| **getParentElement**() | HTMLElement | Возвращает родительский html-элемент.<br>Унаследован от **ILayout**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. По умолчанию пытается построить фигуру через опцию shape — this.getData().options.get('shape').<br>В опции можно задать как инстанцию класса, реализующего интерфейс IShape, так и JSON-описание пиксельной геометрии фигуры. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **ILayout**. |
| **rebuild**() | | Перестраивает макет. |
| **setData**(data) | | Задает данные макета.<br>Унаследован от **ILayout**. |
| **setParentElement**(parent) | | Добавляет макет в DOM-дерево.<br>Унаследован от **ILayout**. |

## Описание методов

### build

```
build()
```

Строит экземпляр макета на основе шаблона и добавляет его в родительский HTML-элемент.

### clear

```
clear()
```

Удаляет содержимое макета из DOM.

### getShape

```
{IShape | null} getShape()
```

**Возвращает** фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. По умолчанию пытается построить фигуру через опцию shape — this.getData().options.get('shape'). В опции можно задать как инстанцию класса, реализующего интерфейс IShape, так и JSON-описание пиксельной геометрии фигуры. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Пример:**

```
// Создадим метку и зададим ей круглую интерактивную область.
var MyLayoutClass = ymaps.templateLayoutFactory.createClass('<div
class="imageIcon">{{name}}</div>'),
    myPlacemark = new ymaps.Placemark([22, 34], {name: 'Кафе Маяк'}, {
        iconLayout: MyLayoutClass,
       iconShape: {type: 'Circle', coordinates: [0, 0], radius: 20}
   });
```

## rebuild

```
rebuild()
```

Перестраивает макет.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# LoadingObjectManager

Расширяет **ICustomizable**, **IEventEmitter**, **IGeoObject**, **IParentOnMap**.

Менеджер объектов, осуществляющий их оптимальную подгрузку с сервера. Позволяет оптимально загружать, отображать, кластеризовать и управлять видимостью объектов. Менеджер отправляет запрос за данными по указанному url в JSONP-формате. Описание формата соответствует формату добавляемых в ObjectManager объектов ObjectManager.add. Обратите внимание, что у объектов, отрисованных на карте через данный менеджер, нельзя включать режимы редактирования и перетаскивания.

Конструктор | Поля | События | Методы

## Конструктор

```
LoadingObjectManager(urlTemplate[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| urlTemplate * | — | Тип: String<br><br>Шаблон URL данных. Поддерживаются специальные конструкции по аналогии с **Layer**. Также поддерживаются подстановки:<br>• %b заменяется на массив географических координат, описывающих прямоугольную область, для которой требуется загрузить данные.<br>• %t заменяется на массив номеров тайлов, описывающих прямоугольную область, для которой требуется загрузить данные. |
| options | — | Тип: Object<br><br>Опции.<br>• Можно задавать все опции, указанные в описании **Clusterer**, за исключением опций hasBalloon и hasHint.<br>• Опции для кластеров задаются с префиксом cluster. Список опций указан в описании класса **ClusterPlacemark**;<br>• Опции для одиночных объектов задаются с префиксом geoObject. Список опций указан в описании класса **Placemark**. |
| options.clusterize | false | Тип: Boolean<br><br>Флаг, показывающий, нужно ли кластеризовать объекты. Обратите внимание, что на данный момент кластеризация работает только для точечных объектов. При включенном режиме кластеризации все неточечные объекты будут игнорироваться. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.paddingParamName | 'callback' | Тип: Boolean<br><br>Имя GET-параметра, который содержит значение jsonp-колбека. |
| options.paddingTemplate | null | Тип: String<br><br>Шаблон для jsonp-колбека. Поддерживает те же подстановки, что и urlTemplate. Все символы, не являющиеся буквой или цифрой, будут заменены на '_'. Если параметр не задан, то имя jsonp-колбека будет сгенерировано автоматически. Примеры преобразований при tileNumber=[3, 1], zoom=9:<br>• 'myCallback=%x' => 'myCallback_3'<br>• '%c' => 'x_3_y_1_z_9'<br>• 'callback2_%c' => 'callback2_x_3_y_1_z_9'<br>• 'callback%test' => 'callback_test'<br>• 'callback_%b' => 'callback_85_0841__180_0000_85_0841_180_0000'<br><br>Обратите внимание, что если не использовать в значении опции подстановки, то это может привести к ошибке. Все запросы будут обращаться к одной callback-функции. |
| options.splitRequests | false | Тип: Boolean<br><br>Разделять запросы за данными на запросы за одиночными тайлами. По умолчанию запросы делаются за данными для прямоугольной области, содержащей несколько тайлов. |
| options.syncOverlayInit | false | Тип: Boolean<br><br>Флаг, разрешающий создавать оверлеи для объектов синхронно. Обратите внимание, что при синхронном создании оверлея нужно самостоятельно обеспечить загрузку нужного класса, реализующего интерфейс IOverlay. По умолчанию оверлеи создаются асинхронно, при этом класс оверлея загружается по требованию. |
| options.viewportMargin | 128 | Тип: Number \| Number[]<br><br>Отступ для области, в которой показываются объекты. С помощью данной опции область показа объектов расширяется по отношению к видимой области карты. |

* Обязательный параметр/опция.

**Примеры:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

1.

```
var objectManager = new ymaps.LoadingObjectManager('http://myServer.com/tile?
bbox=%b', {
    // Включаем кластеризацию.
    clusterize: true,
    // Опции кластеров задаются с префиксом cluster.
    clusterHasBalloon: false,
    // Опции геообъектов задаются с префиксом geoObject
    geoObjectOpenBalloonOnClick: false
});

// Можно задавать опции напрямую в дочерние коллекции.
objectManager.clusters.options.set({
    preset: 'islands#grayClusterIcons',
    hintContentLayout: ymaps.templateLayoutFactory.createClass('Группа
объектов')
});
objectManager.objects.options.set('preset', 'islands#grayIcon');
```

2.

```
Пример ответа для LoadingObjectManager
jsonp_callback({
    // Ответ содержит поля error и data. В случае ошибки в поле error
    // пишется код ошибки или ее описание.
    error: null,
    data: {
        type: 'FeatureCollection',
        features: [
            {
                type: 'Feature',
                geometry: {
                    type: 'Point',
                    coordinates: [55, 35]
                },
                id: 23,
                properties: {
                    balloonContent: 'Содержимое балуна метки',
                    iconContent: 'Содержимое метки'
                },
                options: {
                    preset: 'islands#yellowIcon'
                }
            }
        ]
    }
});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **clusters** | **objectManager.ClusterCollection** | Коллекция кластеров, сгенерированных менеджером. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **objects** | **objectManager.ObjectCollection** | Коллекция объектов, добавленных в слой. |
| **options** | **IOptionManager** | Менеджер опций. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **ICustomizable**. |
| **properties** | **IDataManager** | Данные геообъекта.<br>Унаследовано от **IGeoObject**. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Вычисляет границы области в геокоординатах, охватывающей все загруженные объекты в составе менеджера. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getObjectState**(id) | Object | Получение информации о текущем состоянии объекта, добавленного в менеджер. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getPixelBounds**() | Number[][] \| null | Вычисляет границы области в глобальных пиксельных координатах, охватывающей все загруженные объекты в составе менеджера. |
| **getTileUrl**() | String \| null | Возвращает URL тайла с данными. |
| **getUrlTemplate**() | String | Возвращает шаблон URL данных. |
| **reloadData**() | | Метод, удаляющий все загруженные ранее данные и отправляющий запрос за новыми данными. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |
| setUrlTemplate(urlTemplate) | | |

## Описание полей

### clusters

```
{objectManager.ClusterCollection} clusters
```

Коллекция кластеров, сгенерированных менеджером.

**Пример:**

```
objectManager.clusters.events.add('click', function (e) {
    var objectId = e.get('objectId');
    objectManager.clusters.balloon.open(objectId);
});
```

### objects

```
{objectManager.ObjectCollection} objects
```

Коллекция объектов, добавленных в слой.

**Пример:**

```
objectManager.objects.events.add('click', function (e) {
    var objectId = e.get('objectId');
    objectManager.objects.balloon.open(objectId);
});
```

## Описание методов

### getBounds

```
{Number[][] | null} getBounds()
```

Вычисляет границы области в геокоординатах, охватывающей все загруженные объекты в составе менеджера.

**Возвращает** массив координат области или null, если менеджер не добавлен на карту.

### getObjectState

```
{Object} getObjectState(id)
```

Получение информации о текущем состоянии объекта, добавленного в менеджер.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** объект с полями:

- found — признак, указывающий, существует ли объект с переданным идентификатором. Тип: Boolean;
- isShown — признак, указывающий, находится ли объект в видимой области карты. Тип: Boolean;
- cluster — JSON-описание кластера, в который добавлен объект. Помимо обязательных полей, содержит поле properties.geoObjects с массивом объектов, попавших в кластер. Поле возвращается только при включенной кластеризации;
- isClustered — признак, указывающий, попал ли объект в состав кластера; Поле возвращается только при включенной кластеризации. Тип: Boolean;
- isFilteredOut — признак, указывающий, прошел ли объект фильтрацию. Если фильтр не задан или объект прошел фильтрацию, значение поля будет false. Тип: Boolean.

.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| id* | — | Тип: Object<br><br>Идентификатор объекта, для которого нужно получить состояние. |

* Обязательный параметр/опция.

**Пример:**

```
// Открытие балуна кластера с выбранным объектом.
// Получим данные о состоянии объекта внутри кластера.
var objectState = objectManager.getObjectState(myObjects[1].id);
// Проверяем, находится ли объект в видимой области карты.
if (objectState.found && objectState.isShown) {
    // Если объект попадает в кластер, открываем балун кластера с нужным выбранным
объектом.
    if (objectState.isClustered) {
        objectManager.clusters.state.set('activeObject', myObjects[1]);
        objectManager.clusters.balloon.open(objectState.cluster.id);
    } else {
        // Если объект не попал в кластер, открываем его собственный балун.
        objectManager.objects.balloon.open(myObjects[i].id);
    }
}
```

## getPixelBounds

```
{Number[][] | null} getPixelBounds()
```

Вычисляет границы области в глобальных пиксельных координатах, охватывающей все загруженные объекты в составе менеджера.

**Возвращает** массив координат области или null, если менеджер не добавлен на карту.

## getTileUrl

```
{String | null} getTileUrl()
```

**Возвращает** URL тайла с данными.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters* | — | Тип: |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

**Пример:**

```
var objectManager = new ymaps.LoadingObjectManager('http://myServer.com/tile?
bbox=%b');
objectManager.getTileUrl = function (parameters) {
   var boundingBox = parameters.boundingBox.join('~');
   return this.getUrlTemplate().replace(/%b/g, boundingBox);
};
```

### getUrlTemplate

```
{String} getUrlTemplate()
```

**Возвращает** шаблон URL данных.

### reloadData

```
reloadData()
```

Метод, удаляющий все загруженные ранее данные и отправляющий запрос за новыми данными.

### setUrlTemplate

```
setUrlTemplate(urlTemplate)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| urlTemplate* | — | Тип: String<br><br>шаблон URL данных. |

* Обязательный параметр/опция.

## Map

Расширяет **IDomEventEmitter**.

Класс для создания и управления картой.

Конструктор | Поля | События | Методы

### Конструктор

```
Map(element, state[, options])
```

Создает экземпляр карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| element * | — | Тип: Object \| String<br><br>Ссылка на HTML-элемент, в котором размещается карта, или идентификатор этого HTML-элемента. |
| state * | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Параметры карты. |
| state.behaviors | ['default'] | Тип: String[]<br><br>Включенные поведения карты. По умолчанию включаются: перетаскивание карты, масштабирование карты мультисенсорным касанием и двойным кликом на устройствах с сенсорным экраном; перетаскивание карты мышью, масштабирование двойным щелчком и выделением области правой кнопкой мыши на остальных устройствах. Можно указывать любые ключи, которые поддерживаются behavior.Manager. |
| state.bounds | — | Тип: Number[][]<br><br>Геокоординаты начальной области показа карты. При инициализации карты может быть задана либо пара state.zoom-state.center, либо данная область. Если совместно со state.bounds будут заданы параметры state.zoom, state.center, они будут проигнорированы. |
| state.center | — | Тип: Number[]<br><br>Геокоординаты центра карты. Должны быть заданы совместно со state.zoom. |
| state.controls | ['default'] | Тип: String[]<br><br>Добавленные по умолчанию элементы управления карты. Можно указывать любые ключи, поддерживаемые **control.Manager**. Список доступных ключей и наборы элементов управления по умолчанию описаны в control.Manager.add. |
| state.type | 'yandex#map' | Тип: String \| **MapType**<br><br>Тип карты. Может быть ключом или экземпляром класса **MapType**. Список доступных ключей:<br>• 'yandex#map' — тип карты "схема";<br>• 'yandex#satellite' — тип карты "спутник";<br>• 'yandex#hybrid' — тип карты "гибрид". |
| state.zoom | — | Тип: Number<br><br>Коэффициент масштабирования карты. Должен быть задан совместно со state.center. |
| options | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опции карты. Через опции карты можно задавать настройки как самой карты, так и добавленных на неё объектов:<br>• опции поведений карты;<br>• опции балуна карты с префиксом `balloon`;<br>• опции хинта карты с префиксом `hint`;<br>• опции гео-объектов с префиксом `geoObject`;<br>• опции слоев с префиксом `layer`;<br>• опции слоев хотспотов с префиксом `hotspotLayer`;<br>Опции, которые трактует непосредственно сама карта, перечислены ниже. |
| `options.autoFitToViewport` | 'ifNull' | Тип: String<br>Автоматическое слежение за контейнером карты. По умолчанию карта перестроится автоматически в случае, когда она инициализируется из скрытого контейнера, либо мы программно поменяли его размеры, иначе необходимо вызвать map.container.fitToViewport. Доступны следующие значения:<br>• 'none' — не следить за изменением отображения контейнера при инициализации из скрытого контейнера или при программном изменении его размеров;<br>• 'ifNull' — как только контейнер получит CSS значение «display», отличное от «none», автоматически запустится map.container.fitToViewport, чтобы подстроить карту под него. После слежение будет остановлено;<br>• 'always' — постоянно следить за изменением состояния отображения контейнера карты. |
| `options.avoidFractionalZoom` | true | Тип: Boolean<br>true — карта не будет останавливаться на дробных значениях коэффициента масштабирования, false — будет.<br>**Примечание:**<br>Для десктопных браузеров данная опция по умолчанию принимает значение true, для мобильных — false. |
| `options.exitFullscreenByEsc` | true | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Включение возможности выхода из «полноэкранного режима» с помощью нажатия на клавишу ESC. True — выходить из «полноэкранного режима» по нажатию на ESC, false — не выходить. |
| options.fullscreenZIndex | 10000 | Тип: Number<br><br>Значение z-index контейнера карты в «полноэкранном режиме». |
| options.mapAutoFocus | true | Тип: Boolean<br><br>true — при клике на карту будет убираться фокус с текущего активного dom-элемента, false — при клике на карту фокус будет оставаться в активном элементе. |
| options.maxAnimationZoomDifference | 5 | Тип: Number<br><br>Максимальная разница между текущим и выставляемым коэффициентом масштабирования карты, при котором масштабирование будет происходить плавно. |
| options.maxZoom | 23 | Тип: Number<br><br>Максимальный коэффициент масштабирования карты. |
| options.minZoom | 0 | Тип: Number<br><br>Минимальный коэффициент масштабирования карты. |
| options.projection | ymaps.projection.wgs84Mercator | Тип: **IProjection**<br><br>Проекция карты. |
| options.restrictMapArea | false | Тип: Boolean \| Number[][]<br><br>Устанавливает область просмотра карты так, чтобы пользователь не мог выйти за пределы этой области. True — установить область в соответствии с текущей областью отображения, false — отключить опцию. Если необходимо ограничить область с известными координатами, нужно указать эту прямоугольную область. |
| options.suppressMapOpenBlock | false | Тип: Boolean<br><br>Нужно ли скрывать предложение открыть текущую карту в Яндекс.Картах, максимально сохранив всю имеющуюся информацию о ней. True — скрывать, false — не скрывать. Ссылка на Яндекс.Карты отображается в левом нижнем углу карты. |
| options.suppressObsoleteBrowserNotifier | false | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Нужно ли в устаревших версиях браузеров скрывать уведомление с предложением обновить браузер. True — скрывать, false — не скрывать. Уведомление отображается в левом нижнем углу карты. |
| options.yandexMapAutoSwitch | true | Тип: Boolean<br><br>true — включить автоматическое переключение на Народную Карту там, где схема недостаточно подробна, false — выключить. Эта опция работает только при значениях параметра lang=ru-RU и lang=uk-UA. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Инициализация карты из известных центра и уровня масштабирования
var myMap = new ymaps.Map('map', {
        center: [55.74954, 37.621587],
        zoom: 10
    });
```

**2.**

```
// Инициализация карты по известной области просмотра
// Полагаем, что на странице подключен jQuery
var $mapElement = $('#map'),
    myMap = new ymaps.Map(
        $mapElement[0],
        ymaps.util.bounds.getCenterAndZoom(
            [[55.7, 37.6], [55.8, 37.7]],
            [$mapElement.width(), $mapElement.height()]
        )
    );
```

**3.**

```
// Инициализация карты из результата геокодирования
var myMap;
ymaps.geocode('Москва').then(function (res) {
    myMap = new ymaps.Map('map', {
        center: res.geoObjects.get(0).geometry.getCoordinates(),
        zoom : 10
    });
});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **action** | **map.action.Manager** | Менеджер действий карты. |
| **balloon** | **map.Balloon** | Балун карты. |
| **behaviors** | **map.behavior.Manager** | Менеджер поведений карты. Позволяет включать и выключать поведения, а также предоставляет доступ к их методам и свойствам. |
| **container** | **map.Container** | Контейнер карты. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| controls | control.Manager | Элементы управления картой. |
| converter | map.Converter | Преобразователь пиксельных точек карты из глобальных в локальные и наоборот. |
| copyrights | map.Copyrights | Менеджер размещённой на карте информации об авторских правах. |
| cursors | util.cursor.Manager | Менеджер курсоров карты. |
| events | event.Manager | Менеджер событий карты. Поддерживает подписку с приоритетами. Бросает события типа MapEvent. |
| geoObjects | map.GeoObjects | Менеджер геообъектов карты. |
| hint | map.Hint | Хинт карты. |
| layers | map.layer.Manager | Менеджер слоёв карты. |
| margin | map.margin.Manager | Менеджер отступов карты. |
| options | option.Manager | Опции карты. |
| panes | map.pane.Manager | Менеджер контейнеров объектов карты. |
| zoomRange | map.ZoomRange | Объект, предоставляющий доступ к информации о доступных в точке уровнях масштабирования. |

## События

| Имя | Описание |
|---|---|
| actionbegin | Началось новое плавное движение карты. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• action — запущенное действие. |
| actionbreak | Событие, возникающее при досрочном прекращении шага действия (например, из-за выполнения другого действия или нового шага того же действия). Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• action — действие. |
| actionend | Окончание плавного движения карты. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• action — остановленное действие. |
| actiontick | Начало исполнения нового шага плавного движения (например, сдвиг карты пользователем или шаг анимации плавного масштабирования). Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• action — выполняемое в данный момент действие;<br>• tick — описание шага действия в виде объекта с полями globalPixelCenter, zoom, duration и timingFunction. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **actiontickcomplete** | Окончание исполнения шага плавного движения. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• action — выполняемое в данный момент действие;<br>• tick — описание шага действия в виде объекта с полями globalPixelCenter, zoom, duration и timingFunction. |
| **balloonclose** | Закрытие балуна. Экземпляр класса **Event**. |
| **balloonopen** | Открытие балуна на карте. Экземпляр класса **Event**. |
| **boundschange** | Событие изменения области просмотра карты (в результате изменения центра или уровня масштабирования). Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCenter — предыдущий центр карты в геокоординатах;<br>• newCenter — новый центр карты в геокоординатах;<br>• oldZoom — предыдущий уровень машсштабирования;<br>• newZoom — новый уровень масштабирования;<br>• oldGlobalPixelCenter — предыдущий центр карты в глобальных пикселах;<br>• newGlobalPixelCenter — новый центр карты в глобальных пикселах;<br>• oldBounds — предыдущая область показа;<br>• newBounds — новая область показа. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **destroy** | Карта уничтожена. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**. |
| **hintopen** | Открытие хинта на карте. Экземпляр класса **Event**. |
| **marginchange** | Изменились отступы карты. Экземпляр класса **Event**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменились опции карты. |
| **sizechange** | Изменился размер карты. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• oldSize — предыдущий размер карты;<br>• newSize — новый размер карты. |
| **typechange** | Изменился тип карты. Экземпляр класса **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает карту. |
| **getBounds**([options]) | Number[][] | Возвращает двумерный массив геокоординат левого нижнего и правого верхнего углов области показа карты. |
| **getCenter**([options]) | Number[] | Возвращает географические координаты текущего центра карты. |
| **getGlobalPixelCenter**([options]) | Number[] | Возвращает глобальные пиксельные координаты текущего центра карты. |
| **getType**() | String \| **MapType** | Возвращает текущий тип карты. |
| **getZoom**() | Number | Возвращает текущий коэффициент масштабирования карты. |
| **panTo**(center[, options]) | **vow.Promise** | Устанавливает центр карты. Если передан массив точек, то карта будет перемещаться от одной точки к другой. |
| **setBounds**(bounds[, options]) | **vow.Promise** | Позиционирует карту для показа переданного района. |
| **setCenter**(center[, zoom[, options]]) | **vow.Promise** | Устанавливает центр и коэффициент масштабирования карты. Центр задается в географических координатах. |
| **setGlobalPixelCenter**(globalPixelCenter[, zoom[, options]]) | **vow.Promise** | Устанавливает центр и коэффициент масштабирования карты. Центр задается в глобальных пиксельных координатах. |
| **setType**(type[, options]) | **vow.Promise** | Устанавливает тип карты. |
| **setZoom**(zoom[, options]) | **vow.Promise** | Устанавливает коэффициент масштабирования карты. |

## Описание полей

### action

```
{map.action.Manager} action
```

Менеджер действий карты.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
var myAction = new ymaps.map.action.Single({
          center: [0, 0],
          zoom: 4,
          duration: 1000,
          timingFunction: "ease-in"
    });

myMap.action.execute(myAction);
```

## balloon

```
{map.Balloon} balloon
```

Балун карты.

## behaviors

```
{map.behavior.Manager} behaviors
```

Менеджер поведений карты. Позволяет включать и выключать поведения, а также предоставляет доступ к их методам и свойствам.

**Пример:**

```
// Включим масштабирование колесом мыши
myMap.behaviors.enable('scrollZoom');
```

## container

```
{map.Container} container
```

Контейнер карты.

**Пример:**

```
// Подгоним размер карты под новый размер контейнера
// (например, если изменилась верстка страницы или карта была инициализирована
// в скрытом состоянии)
map.container.fitToViewport();
```

## controls

```
{control.Manager} controls
```

Элементы управления картой.

**Пример:**

```
myMap.controls.add('zoomControl', {
    float: 'none',
    position: {
        right: 40,
       top: 5
   }
});
```

## converter

```
{map.Converter} converter
```

Преобразователь пиксельных точек карты из глобальных в локальные и наоборот.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Преобразуем координаты мыши в географические.
var projection = map.options.get('projection');
$('#map').bind('click', function (e) {
    console.log(projection.fromGlobalPixels(
        map.converter.pageToGlobal([e.pageX, e.pageY]), map.getZoom()
    ));
});
```

## copyrights

```
{map.Copyrights} copyrights
```

Менеджер размещённой на карте информации об авторских правах.

**Пример:**

```
// Добавим информацию об авторе
map.copyrights.add('&copy; Vassily Poupkine');
```

## cursors

```
{util.cursor.Manager} cursors
```

Менеджер курсоров карты.

## events

```
{event.Manager} events
```

Менеджер событий карты. Поддерживает подписку с приоритетами. Бросает события типа **MapEvent**.

**Примеры:**

**1.**

```
// Поставим метку по клику над картой
map.events.add('click', function (e) {
    // Географические координаты точки клика можно узнать
    // посредством вызова .get('coords')
    var position = e.get('coords');
    map.geoObjects.add(new ymaps.Placemark(position));
});
```

**2.**

```
// Будем отслеживать центр и зум карты при плавных перемещениях.
map.events.add('actiontick', function () {
    var state = map.action.getCurrentState();
    console.log(state.zoom, state.globalPixelCenter);
});
```

## geoObjects

```
{map.GeoObjects} geoObjects
```

Менеджер геообъектов карты.

## hint

```
{map.Hint} hint
```

Хинт карты.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## layers

{map.layer.Manager} layers

Менеджер слоёв карты.

**См.:** Layer

**Пример:**

```
// Добавим слой пользовательских тайлов на карту
map.layers.add(new ymaps.Layer('http://some.server/tiles?&c'));
```

## margin

{map.margin.Manager} margin

Менеджер отступов карты.

## options

{option.Manager} options

Опции карты.

**Пример:**

```
// Запретим всем объектам на карте открывать балуны по щелчку мыши
map.options.set('openBalloonOnClick', false);
```

## panes

{map.pane.Manager} panes

Менеджер контейнеров объектов карты.

**Пример:**

```
// Добавим свой элемент в контейнер элементов управления карты
$('<div><input type="button" value="Click!"/></div>')
    .css({ position: 'absolute', left: '5px', top: '50px'})
    .appendTo(map.panes.get('controls').getElement());
```

## zoomRange

{map.ZoomRange} zoomRange

Объект, предоставляющий доступ к информации о доступных в точке уровнях масштабирования.

**Пример:**

```
// узнаем максимальный доступный в точке [20, 30] уровень
// масштабирования карты
map.zoomRange.get([20, 30]).then(function (zoomRange) {
    alert(zoomRange[1]);
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание событий

### actionbegin

Началось новое плавное движение карты. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- action — запущенное действие.

### actionbreak

Событие, возникающее при досрочном прекращении шага действия (например, из-за выполнения другого действия или нового шага того же действия). Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- action — действие.

### actionend

Окончание плавного движения карты. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- action — остановленное действие.

### actiontick

Начало исполнения нового шага плавного движения (например, сдвиг карты пользователем или шаг анимации плавного масштабирования). Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- action — выполняемое в данный момент действие;
- tick — описание шага действия в виде объекта с полями globalPixelCenter, zoom, duration и timingFunction.

**Пример:**

```
// Отслеживает все перемещения карты, даже перетаскивание пользователем
// и плавное масштабирование
map.events.add('actiontick', function (e) {
    var tick = e.get('tick');
    console.log('Сейчас карта переместится в точку (' +
        map.options.get('projection').fromGlobalPixels(tick.globalPixelCenter,
tick.zoom).join(',') +
        ') в течение ' + e.get('tick').duration + ' миллисекунд');
});
```

### actiontickcomplete

Окончание исполнения шага плавного движения. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- action — выполняемое в данный момент действие;
- tick — описание шага действия в виде объекта с полями globalPixelCenter, zoom, duration и timingFunction.

### balloonclose

Закрытие балуна. Экземпляр класса **Event**.

### balloonopen

Открытие балуна на карте. Экземпляр класса **Event**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## boundschange

Событие изменения области просмотра карты (в результате изменения центра или уровня масштабирования). Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- oldCenter — предыдущий центр карты в геокоординатах;
- newCenter — новый центр карты в геокоординатах;
- oldZoom — предыдущий уровень машсштабирования;
- newZoom — новый уровень масштабирования;
- oldGlobalPixelCenter — предыдущий центр карты в глобальных пикселах;
- newGlobalPixelCenter — новый центр карты в глобальных пикселах;
- oldBounds — предыдущая область показа;
- newBounds — новая область показа.

**Пример:**

```
// Будем отслеживать изменение уровня масштабирования карты
map.events.add('boundschange', function (event) {
    if (event.get('newZoom') != event.get('oldZoom')) {
        alert('Уровень масштабирования изменился');
    }
});
```

## destroy

Карта уничтожена.

## hintclose

Закрытие хинта. Экземпляр класса **Event**.

## hintopen

Открытие хинта на карте. Экземпляр класса **Event**.

## marginchange

Изменились отступы карты. Экземпляр класса **Event**.

## optionschange

Изменились опции карты.

## sizechange

Изменился размер карты. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- oldSize — предыдущий размер карты;
- newSize — новый размер карты.

## typechange

Изменился тип карты. Экземпляр класса **Event**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### destroy

```
destroy()
```

Уничтожает карту.

### getBounds

```
{Number[][]} getBounds([options])
```

**Возвращает** двумерный массив геокоординат левого нижнего и правого верхнего углов области показа карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции. |
| options.useMapMargin | false | Тип: Boolean<br><br>Нужно ли учитывать отступы, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

### getCenter

```
{Number[]} getCenter([options])
```

**Возвращает** географические координаты текущего центра карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции. |
| options.useMapMargin | false | Тип: Boolean<br><br>Нужно ли учитывать отступы, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

### getGlobalPixelCenter

```
{Number[]} getGlobalPixelCenter([options])
```

**Возвращает** глобальные пиксельные координаты текущего центра карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.useMapMargin | false | Тип: Boolean<br><br>Нужно ли учитывать отступы, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

**Пример:**

```
// Сдвинем карту на 10 пикселей влево
var position = map.getGlobalPixelCenter();
map.setGlobalPixelCenter([ position[0] - 10, position[1] ]);
```

## getType

```
{String | MapType} getType()
```

**Возвращает** текущий тип карты.

## getZoom

```
{Number} getZoom()
```

**Возвращает** текущий коэффициент масштабирования карты.

## panTo

```
{vow.Promise} panTo(center[, options])
```

Устанавливает центр карты. Если передан массив точек, то карта будет перемещаться от одной точки к другой.

**Возвращает** объект-промис. В случае корректного выполнения действия будет возвращен resolve без значения. В случае ошибки будет возвращен reject с описанием ошибки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| center* | — | Тип: Number[] \| Object[]<br><br>Центр карты или массив точек для последовательного обхода. |
| options | — | Тип: Object<br><br>Опции. |
| options.checkZoomRange | false | Тип: Boolean<br><br>Проверка возможности установить текущий коэффициент масштабирования при изменении центра карты. Если значение опции равно true, метод вызывается асинхронно. Производится обращение к серверу, который возвращает интервал допустимых для данного центра значений коэффициентов масштабирования. После этого выставляется указанный центр и корректный зум. |
| options.delay | 1000 | Тип: Number |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Время задержки между перемещениями в миллисекундах. |
| options.duration | 500 | Тип: Number<br><br>Время анимации в миллисекундах. |
| options.flying | true | Тип: Boolean<br><br>Разрешить уменьшать и затем увеличивать зум карты при перемещении между точками. Если расстояние между точками не превышает двух размеров контейнера карты, то карта плавно переместит центр в указанную точку без изменения масштаба. Если расстояние больше, то карта выставит себе меньший коэффициент масштабирования, затем сместится к конечной точке и выставит требуемый коэффициент масштабирования (эмуляция полета над картой). |
| options.safe | true | Тип: Boolean<br><br>Безопасное смещение карты. Используется при значении опции flying равной false. Если расстояние между точками не превышает двух размеров контейнера карты, то карта плавно переместит центр в указанную точку. Если расстояние больше, то произойдет мгновенный переход в выбранную точку. При помощи этого режима удобно перемещать карту без отображения незагруженных тайлов. |
| options.timingFunction | 'ease-in-out' | Тип: String<br><br>Тайминг-функция. Аналогично значению CSS свойства transition-timing-function. Полный список значений: http://www.w3.org/TR/css3-transitions/#transition-timing-function_tag |
| options.useMapMargin | false | Тип: Boolean<br><br>Нужно ли учитывать отступы, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Полёт из Калининграда во Владивосток через Москву
map.setCenter([54.704815, 20.466380], 10);
map.panTo([
    [55.751574, 37.573856],
    [43.134091, 131.928478]
]).then(function () {
 alert('Прилетели!');
}, function (err) {
 alert('Произошла ошибка ' + err);
}, this);
```

## setBounds

```
{vow.Promise} setBounds(bounds[, options])
```

Позиционирует карту для показа переданного района.

**Возвращает** объект-промис. В случае корректного выполнения действия будет возвращен resolve без значения. В случае ошибки будет возвращен reject с описанием ошибки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds* | — | Тип: Number[][]<br><br>Границы области показа. |
| options | — | Тип: Object<br><br>Опции. |
| options.checkZoomRange | false | Тип: Boolean<br><br>Проверка возможности установить указанный коэффициент масштабирования. Если значение опции равно true, метод вызывается асинхронно. Производится обращение к серверу, который возвращает интервал допустимых для данного центра значений коэффициентов масштабирования. После этого выставляется указанный центр и корректный зум. |
| options.duration | 0 | Тип: Number<br><br>Время анимации в миллисекундах. |
| options.preciseZoom | false | Тип: Boolean<br><br>Возможность использования дробных коэффициентов масштабирования. |
| options.timingFunction | 'linear' | Тип: String<br><br>Тайминг-функция. Аналогично значению CSS свойства transition-timing-function. Полный список значений: http://www.w3.org/TR/css3-transitions/#transition-timing-function_tag |
| options.useMapMargin | false | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Нужно ли учитывать отступы, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |
| options.zoomMargin | 0 | Тип: Number \| Number[]<br><br>Отступы от границ видимой области карты. Если задано одно число — оно применяется ко всем сторонам. Если задано два — то это горизонтальные и вертикальные отступы соответственно. Если задан массив из 4х чисел, то это отступы top, right, bottom, left. При включенной опции "useMapMargin" значение "zoomMargin" складывается со значениями, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

* Обязательный параметр/опция.

**Пример:**

```
// Новые центр и коэффициент масштабирования карты вычисляются на основе
текущего
// состояния карты.
// Если текущий zoomRange не совпадает с zoomRange для нового центра карты,
// могут быть отображены серые тайлы в случае, если область показа очень мала.
// Чтобы избежать этой проблемы, используйте опцию checkZoomRange.
map.setBounds([[60,-40], [20,60]], {
 checkZoomRange: true,
}).then(function () {
    // Действие было успешно завершено.
}, function (err) {
    // Не удалось показать заданный регион
    // ...
 }, this);
```

## setCenter

```
{vow.Promise} setCenter(center[, zoom[, options]])
```

Устанавливает центр и коэффициент масштабирования карты. Центр задается в географических координатах.

**Возвращает** объект-промис. В случае корректного выполнения действия будет возвращен resolve без значения. В случае ошибки будет возвращен reject с описанием ошибки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| center* | — | Тип: Number[]<br><br>Геокоординаты центра карты. |
| zoom | — | Тип: Number<br><br>Коэффициент масштабирования карты. |
| options | — | Тип: Object<br><br>Опции. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.checkZoomRange | false | Тип: Boolean<br><br>Проверка возможности установить указанный коэффициент масштабирования. Если значение опции равно true, метод вызывается асинхронно. Производится обращение к серверу, который возвращает интервал допустимых для данного центра значений коэффициентов масштабирования. После этого выставляется указанный центр и корректный зум. |
| options.duration | 0 | Тип: Number<br><br>Время анимации в миллисекундах. |
| options.timingFunction | 'linear' | Тип: String<br><br>Тайминг-функция. Аналогично значению CSS свойства transition-timing-function. Полный список значений: http://www.w3.org/TR/css3-transitions/#transition-timing-function_tag |
| options.useMapMargin | false | Тип: Boolean<br><br>Нужно ли учитывать отступы, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

* Обязательный параметр/опция.

**Пример:**

```
myMap.setCenter([40, 50], 3, {
    checkZoomRange: true
});
```

## setGlobalPixelCenter

```
{vow.Promise} setGlobalPixelCenter(globalPixelCenter[, zoom[, options]])
```

Устанавливает центр и коэффициент масштабирования карты. Центр задается в глобальных пиксельных координатах.

**Возвращает** объект-промис. В случае корректного выполнения действия будет возвращен resolve без значения. В случае ошибки будет возвращен reject с описанием ошибки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| globalPixelCenter* | — | Тип: Number[]<br><br>Пиксельные координаты нового центра карты. |
| zoom | — | Тип: Number<br><br>Коэффициент масштабирования карты. |
| options | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опции. |
| options.checkZoomRange | false | Тип: Boolean<br><br>Проверка возможности установить указанный коэффициент масштабирования. Если значение опции равно true, метод вызывается асинхронно. Производится обращение к серверу, который возвращает интервал допустимых для данного центра значений коэффициентов масштабирования. После этого выставляется указанный центр и корректный зум. |
| options.duration | 0 | Тип: Number<br><br>Время анимации в миллисекундах. |
| options.timingFunction | 'linear' | Тип: String<br><br>Тайминг-функция. Аналогично значению CSS свойства transition-timing-function. Полный список значений: http://www.w3.org/TR/css3-transitions/#transition-timing-function_tag |
| options.useMapMargin | false | Тип: Boolean<br><br>Нужно ли учитывать отступы, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

* Обязательный параметр/опция.

**Пример:**

```
// Сдвинем карту на 10 пикселей влево
var position = map.getGlobalPixelCenter();
map.setGlobalPixelCenter([ position[0] - 10, position[1] ]);
```

## setType

```
{vow.Promise} setType(type[, options])
```

Устанавливает тип карты.

**Возвращает** объект-промис. В случае корректного выполнения действия будет возвращен resolve без значения. В случае ошибки будет возвращен reject с описанием ошибки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| type* | — | Тип: String \| **MapType** |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Тип карты. Может быть ключом или или экземпляром класса **MapType**. Список доступных ключей:<br>• 'yandex#map' — тип карты "схема";<br>• 'yandex#satellite' — тип карты "спутник";<br>• 'yandex#hybrid' — тип карты "гибрид". |
| options | — | Тип: Object<br><br>Опции карты. |
| options.checkZoomRange | false | Тип: Boolean<br><br>Проверка возможности установить указанный тип карты при данном уровне масштабирования. При значении true производится обращение к серверу, который возвращает интервал допустимых для данного типа карты значений коэффициентов масштабирования. После этого выставляется указанный центр и корректный зум. |

* Обязательный параметр/опция.

**Пример:**

```
map.setType('yandex#hybrid', {
 checkZoomRange: true
}).then(function () {
 // Тип карты был установен с допустимым уровнем масштабирования.
}, this);
```

## setZoom

```
{vow.Promise} setZoom(zoom[, options])
```

Устанавливает коэффициент масштабирования карты.

**Возвращает** объект-промис. В случае корректного выполнения действия будет возвращен resolve без значения. В случае ошибки будет возвращен reject с описанием ошибки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| zoom* | — | Тип: Number<br><br>Коэффициент масштабирования карты. |
| options | — | Тип: Object<br><br>Опции. |
| options.checkZoomRange | false | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Проверка возможности установить указанный коэффициент масштабирования. |
| options.duration | 0 | Тип: Number<br><br>Время анимации в миллисекундах. |
| options.useMapMargin | false | Тип: Boolean<br><br>Нужно ли учитывать отступы, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

* Обязательный параметр/опция.

**Пример:**

```
myMap.setZoom(4, {duration: 1000});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# map

## map.action

### map.action.Continuous

Расширяет **IMapAction**.

Движение карты, состоящее из одного или более шагов. Предназначено для реализации сложных движений карты.

Конструктор | Поля | События | Методы

#### Конструктор

```
map.action.Continuous()
```

#### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

#### События

| Имя | Описание |
|---|---|
| **end** | Событие, оповещающее карту об окончании движения.<br>Унаследовано от **IMapAction**. |
| **tick** | Событие, оповещающее карту о следующем шаге движения. Содержит поля:<br>• globalPixelCenter — новый центр карты в глобальных пикселах;<br>• zoom — новый коэффициент масштабирования карты;<br>• duration — время, которое отводится на выполнение шага;<br>• timingFunction — функция, описывающая вид движения.<br>Унаследовано от **IMapAction**. |

#### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **begin**(mapActionManager) | | Запускает движение на исполнение картой. Этот метод автоматически вызывается менеджером движений карты. С момента вызова IMapAction.begin менеджер движений слушает события IMapAction.tick и IMapAction.end и исполняет их.<br>Унаследован от **IMapAction**. |
| **end**() | | Останавливает движение.<br>Унаследован от **IMapAction**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| isActive() | Boolean | Проверяет, исполняется ли движение картой в данный момент. |
| tick(tick) | map.action.Continuous | Выполняет один шаг движения карты. |

## Описание методов

### isActive

```
{Boolean} isActive()
```

Проверяет, исполняется ли движение картой в данный момент.

**Возвращает** true — движение в данный момент исполняется, false — нет.

### tick

```
{map.action.Continuous} tick(tick)
```

Выполняет один шаг движения карты.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tick* | — | Тип: Object<br><br>Параметры движения. |
| tick.duration | 0 | Тип: Number<br><br>Длительность исполнения движения, в миллисекундах. |
| tick.globalPixelCenter | — | Тип: Number[]<br><br>Новый центр карты в глобальных пикселах. Должен быть задан один из параметров: либо pixelOffset, либо globalPixelCenter |
| tick.pixelOffset | — | Тип: Number[]<br><br>Смещение в пикселях относительно предыдущего центра. Должен быть задан один из параметров: либо pixelOffset, либо globalPixelCenter. |
| tick.timingFunction | 'linear' | Тип: String<br><br>Тайминг-функция. |
| tick.zoom | — | Тип: Number<br><br>Новый коэффициент масштабирования карты. Если не указан, масштаб карты не изменяется. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# map.action.Manager

Расширяет **IEventEmitter**.

Менеджер действий карты. Позволяет выполнять сложные движения на карте и гарантирует, что сложные движения не наложатся друг на друга. У каждой карты уже есть собственный менеджер действий, доступный как **Map.action**. Не создавайте экземпляры этого класса без необходимости.

**См.: Map.action**

Конструктор | Поля | События | Методы

## Конструктор

```
map.action.Manager(map)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Карта. |

* Обязательный параметр/опция.

**Пример:**

```
// Создадим сложное движение: каждые 100 мс центр
// карты смещается на случайную величину.

// Создадим экземпляр сложного движения.
var action = new ymaps.map.action.Continuous();
// Запустим его выполнение на карте.
myMap.action.execute(action);

// Запомним пиксельный центр карты и уровень масштабирования.
var center = myMap.getGlobalPixelCenter(),
    zoom = myMap.getZoom();
// Каждые 100 миллисекунд будем генерировать случайный сдвиг.
var interval = window.setInterval(function () {
    center[0] += Math.round(Math.random() * 100) - 50;
    center[1] += Math.round(Math.random() * 100) - 50;
    // Генерируем новый сдвиг карты.
    action.tick({
        globalPixelCenter: center,
        zoom: zoom
    });
}, 100);

// Как только пользователь сдвинет карту, наше движение
// прекратит выполняться и произойдёт событие end.
var listener = action.events.once('end', function () {
    window.clearInterval(interval);
});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| **begin** | Событие, возникающее, когда действие началось. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• action — запущенное действие. |
| **break** | Событие, возникающее при досрочном прекращении шага действия (например, из-за выполнения другого действия или нового шага того же действия). Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• action — действие. |
| **end** | Событие, возникающее при остановке действия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• action — остановленное действие. |
| **tick** | Событие, возникающее, когда очередной шаг действия начинает исполняться. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• action — выполняемое в данный момент действие;<br>• tick — описание шага действия в виде объекта с полями globalPixelCenter, zoom, duration и timingFunction. |
| **tickcomplete** | Событие, возникающее при окончании выполнения шага действия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• action — выполняемое в данный момент действие;<br>• tick — описание шага действия в виде объекта с полями globalPixelCenter, zoom, duration и timingFunction. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **breakTick**() | | Прерывает очередной шаг действия. |
| **execute**(action) | | Запускает выполнение действия на карте. Если в этот момент на карте выполняется какое-то другое движение, то оно останавливается (вызывается метод end). Новое движение запускается посредством вызова метода begin. |
| **getCurrentState**() | Object | Определяет состояние карты в момент ее плавного движения. |
| **getMap**() | **Map** | Возвращает ссылку на карту. |
| **setCorrection**(userFunction) | | Позволяет устанавливать пользовательские коррекции для сложных движений на карте. По окончанию коррекции необходимо вернуть скорректированные значения. |
| **stop**() | | Останавливает выполнение действия на карте. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание событий

### begin

Событие, возникающее, когда действие началось. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- action — запущенное действие.

### break

Событие, возникающее при досрочном прекращении шага действия (например, из-за выполнения другого действия или нового шага того же действия). Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- action — действие.

### end

Событие, возникающее при остановке действия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- action — остановленное действие.

### tick

Событие, возникающее, когда очередной шаг действия начинает исполняться. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- action — выполняемое в данный момент действие;
- tick — описание шага действия в виде объекта с полями globalPixelCenter, zoom, duration и timingFunction.

**Пример:**

```
// Отслеживает все перемещения карты, даже перетаскивание пользователем
// и плавное масштабирование.
myMap.action.events.add('tick', function (e) {
    var tick = e.get('tick');
    console.log('Сейчас карта переместится в точку (' +
        myMap.options.get('projection').fromGlobalPixels(tick.globalPixelCenter,
tick.zoom).join(',') +
        ') в течение ' + e.get('tick').duration + ' миллисекунд');
});
```

### tickcomplete

Событие, возникающее при окончании выполнения шага действия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- action — выполняемое в данный момент действие;
- tick — описание шага действия в виде объекта с полями globalPixelCenter, zoom, duration и timingFunction.

## Описание методов

### breakTick

```
breakTick()
```

Прерывает очередной шаг действия.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## execute

```
execute(action)
```

Запускает выполнение действия на карте. Если в этот момент на карте выполняется какое-то другое движение, то оно останавливается (вызывается метод end). Новое движение запускается посредством вызова метода begin.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| action* | — | Тип: **IMapAction**<br><br>Действие. |

* Обязательный параметр/опция.

## getCurrentState

```
{Object} getCurrentState()
```

Определяет состояние карты в момент ее плавного движения.

**Возвращает** объект с полями: isTicking — выполняется ли сейчас шаг плавного движения; tickProgress — какая часть текущего шага выполнена; zoom — масштаб карты во время текущего шага; globalPixelCenter — центр карты в глобальных пикселях в момент текущего шага.

**Пример:**

```
// Логирует текущий центр карты.
// Работает даже во время плавного масштабирования или незавершенного
// перетаскивания карты пользователем.
window.setInterval(function () {
    console.log(myMap.action.getCurrentState().center.join(', '));
}, 100);
```

## getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

## setCorrection

```
setCorrection(userFunction)
```

Позволяет устанавливать пользовательские коррекции для сложных движений на карте. По окончанию коррекции необходимо вернуть скорректированные значения.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| userFunction* | — | Тип: Function<br><br>Пользовательская функция коррекции шагов. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Сделаем так, чтобы пользователь не мог перетащить центр карты
// за пределы МКАД.
var mkad = [
    [55.785017, 37.841576],
    [55.861979, 37.765992],
    [55.898533, 37.635961],
    [55.888897, 37.48861],
    [55.83251, 37.395275],
    [55.744789, 37.370248],
    [55.660424, 37.434424],
    [55.5922, 37.526366],
    [55.574019, 37.683167],
    [55.62913, 37.802473],
    [55.712203, 37.837121]
],
mkadPolygon = new ymaps.Polygon([mkad], {}, {
    fillColor: '#FFFF00',
    opacity: .4
});
myMap.geoObjects.add(mkadPolygon);
myMap.action.setCorrection(function (tick) {
    var projection = myMap.options.get('projection'),
        tickCenter = projection.fromGlobalPixels(tick.globalPixelCenter,
tick.zoom);
    // Если центр карты не входит в нашу область.
    if (!mkadPolygon.geometry.contains(tickCenter)) {
        tick.globalPixelCenter = projection.toGlobalPixels(
            mkadPolygon.geometry.getClosest(tickCenter).position,
            tick.zoom
        );
        tick.duration = 0;
    }
    return tick;
});
```

### stop

```
stop()
```

Останавливает выполнение действия на карте.

## map.action.Single

Расширяет **IMapAction**.

Простое движение карты. Движение выполняется сразу после передачи его в map.action.Manager.

Конструктор | Поля | События | Методы

### Конструктор

```
map.action.Single(tick)
```

Создаёт простое (один шаг) движение карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| tick* | — | Тип: Object<br><br>Параметры движения. |
| tick.callback | — | Тип: Function |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Функция, которая будет вызвана после выполнения действия. Принимает в качестве параметра ошибку или null, если действие выполнилось успешно. |
| tick.center | — | Тип: Number[]<br><br>Новый центр карты в геокоординатах. |
| tick.checkZoomRange | false | Тип: Boolean<br><br>Флаг, показывающий, нужно ли проверять новый коэффициент мастабирования карты. Если значение флага true, то перед выполнением действия будет запрошен интервал допустимых коэффициентов масштабирования в новой точке. Если указанный коэффициент выходит за интервал допустимых значений, он будет скорректирован. При этом будет изменено значение нового центра карты в глобальных пиксельных координатах. |
| tick.duration | 0 | Тип: Number<br><br>Время исполнения движения, в миллисекундах. |
| tick.globalPixelCenter | — | Тип: Number[]<br><br>Новый центр карты в глобальных пикселах. При одновременном задании параметров center и globalPixelCenter, параметр center игнорируется. |
| tick.timingFunction | 'linear' | Тип: String<br><br>Тайминг-функция. |
| tick.zoom | — | Тип: Number<br><br>Новый коэффициент масштабирования карты. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
var myCallback = function(err) {
        if (err) {
            throw err;
        }
    },
    myAction = new ymaps.map.action.Single({
          center: [0, 0],
          zoom: 4,
          duration: 1000,
          timingFunction: 'ease-in',
          checkZoomRange: true,
          callback: myCallback
    });

// Действие выполнится сразу после вызова метода execute.
myMap.action.execute(myAction);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **end** | Событие, оповещающее карту об окончании движения.<br>Унаследовано от **IMapAction**. |
| **tick** | Событие, оповещающее карту о следующем шаге движения. Содержит поля:<br>• globalPixelCenter — новый центр карты в глобальных пикселах;<br>• zoom — новый коэффициент масштабирования карты;<br>• duration — время, которое отводится на выполнение шага;<br>• timingFunction — функция, описывающая вид движения.<br>Унаследовано от **IMapAction**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **begin**(mapActionManager) | | Запускает движение на исполнение картой. Этот метод автоматически вызывается менеджером движений карты. С момента вызова IMapAction.begin менеджер движений слушает события IMapAction.tick и IMapAction.end и исполняет их.<br>Унаследован от **IMapAction**. |
| **end**() | | Останавливает движение.<br>Унаследован от **IMapAction**. |
| **isActive**() | Boolean | Проверяет, исполняется ли движение картой в данный момент. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### isActive

```
{Boolean} isActive()
```

Проверяет, исполняется ли движение картой в данный момент.

**Возвращает** true — движение в данный момент исполняется, false — нет.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# map.addon

## map.addon.balloon

**Примечание:**
Конструктор класса map.addon.balloon скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Статический объект.

Модуль, который предоставляет возможность использовать балун у карты. Добавляет карте (**Map**) интерфейс **IBalloonOwner**. При подключении package.full (стандартный набор модулей) доступен по умолчанию. Если **Map** подключается отдельно, то необходимо явно указать этот модуль в загрузчике. Если **map.addon.balloon** подключается отдельно после создания **Map**, то интерфейс **IBalloonOwner** не будет добавлен. Тогда для инициализации менеджера балуна нужно будет использовать метод map.addon.balloon#get.

Методы

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get() | IPopupManager | Возвращает менеджер балуна карты. |

### Описание методов

#### get

```
{IPopupManager} get()
```

**Возвращает** менеджер балуна карты.

**Пример:**

```
ymaps.map.addon.balloon.get(myMap)
```

## map.addon.hint

**Примечание:**
Конструктор класса map.addon.hint скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Статический объект.

Модуль, который предоставляет возможность использовать хинт у карты. Добавляет карте (**Map**) интерфейс **IHintOwner**. При подключении package.full (стандартный набор модулей) доступен по умолчанию. Если **Map** подключается отдельно, то необходимо явно указать этот модуль в загрузчике. Если


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**map.addon.hint** подключается отдельно после создания **Map**, то интерфейс **IHintOwner** не будет добавлен. Тогда для инициализации менеджера балуна нужно будет использовать метод map.addon.hint#get.

Методы

---

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

---

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get() | **IPopupManager** | Возвращает менеджер хинта карты. |

## Описание методов

### get

```
{IPopupManager} get()
```

**Возвращает** менеджер хинта карты.

**Пример:**

```
ymaps.map.addon.hint.get(myMap)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# map.Balloon

Расширяет **IBalloonManager**, IBalloonSharingManager.

Менеджер балуна карты. У каждой карты уже есть собственный менеджер балуна, доступный как myMap.balloon. В один момент на карте может быть открыт только один балун, управляемый менеджером. Не создавайте экземпляры этого класса без необходимости.

**См.: Balloon Map.balloon**

Конструктор | Поля | События | Методы

## Конструктор

```
map.Balloon(map)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Ссылка на объект карты. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **autopanbegin** | Начало автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **autopanend** | Окончание автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **beforeuserclose** | Событие, предшествующее Balloon.userclose. Позволяет отменить пользовательское действие посредством вызова метода preventDefault. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **userclose** | Закрытие балуна пользователем. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IBalloonManager**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **autoPan**() | **vow.Promise** | Перемещает карту таким образом, чтобы балун оказался видимым.<br>Унаследован от **IBalloonManager**. |
| **close**([force]) | **vow.Promise** | Закрывает инфо-объект.<br>Унаследован от **IPopupManager**. |
| **destroy**() | | Отключает менеджер инфо-объекта.<br>Унаследован от **IPopupManager**. |
| **getData**() | Object \| null | Возвращает данные инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **getOptions**() | **IOptionManager** \| null | Возвращает менеджер опций или `null`.<br>Унаследован от **IPopupManager**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от **IPopupManager**. |
| **getOverlaySync**() | **IOverlay** \| null | Возвращает оверлей, если тот существует.<br>Унаследован от **IPopupManager**. |
| **getPosition**() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **isOpen**() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от **IPopupManager**. |
| **open**([position[, data[, options]]]) | **vow.Promise** | Открывает инфо-объект в указанной позиции.<br>Унаследован от **IPopupManager**. |
| **setData**(data) | **vow.Promise** | Задаёт инфо-объекту новые данные.<br>Унаследован от **IPopupManager**. |
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopupManager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# map.behavior

## map.behavior.Manager

Расширяет **ICustomizable**, **IEventEmitter**, **IParentOnMap**.

Менеджер поведений карты. Позволяет включать и отключать поведения. У каждой карты уже есть собственный менеджер поведений, доступный как map.behaviors. Не инстанцируйте экземпляры этого класса без необходимости

**См.: Map.behaviors**

Конструктор | Поля | События | Методы

### Конструктор

```
map.behavior.Manager(map[, behaviors[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Карта. |
| behaviors | — | Тип: String \| String[]<br><br>Список поведений карты, включенных сразу при создании карты. По умолчанию — "drag", "dblClickZoom", "rightMouseButtonMagnifier" для настольных браузеров, "drag", "dblClickZoom" и "multiTouch" — для мобильных. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Возможные значения ключей:<br>• "default" — короткий синоним для включения/отключения поведений карты по умолчанию;<br>• "drag" — перемещение карты при нажатой левой кнопке мыши либо одиночным касанием **behavior.Drag**;<br>• "scrollZoom" — изменение масштаба колесом мыши **behavior.ScrollZoom**;<br>• "dblClickZoom" — масштабирование карты двойным щелчком кнопки мыши **behavior.DblClickZoom**;<br>• "multiTouch" — масштабирование карты двойным касанием (например, пальцами на сенсорном экране) **behavior.MultiTouch**;<br>• "rightMouseButtonMagnifier" — увеличение области, выделенной правой кнопкой мыши (только для настольных браузеров), **behavior.RightMouseButtonMagnifier**;<br>• "leftMouseButtonMagnifier" — увеличение области, выделенной левой кнопкой мыши либо одиночным касанием, **behavior.LeftMouseButtonMagnifier**;<br>• "ruler" — измерение расстояния **behavior.Ruler**;<br>• "routeEditor" — редактор маршрутов **behavior.RouteEditor**;<br>Добавлять и удалять классы поведений можно через хранилище поведений **behavior.storage**. |
| `options` | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опции поведений. Можно задавать следующие опции:<br>• опции поведения behavior.Drag с префиксом `drag`;<br>• опции поведения behavior.ScrollZoom с префиксом `scrollZoom`;<br>• опции поведения behavior.DblClickZoom с префиксом `dblClickZoom`;<br>• опции поведения behavior.MultiTouch с префиксом `multiTouch`;<br>• опции поведения magnifier.RightMouseButtonMagnifier, с префиксом `rightMouseButtonMagnifier`;<br>• опции поведения behavior.LeftMouseButtonMagnifier, с префиксом `leftMouseButtonMagnifier`;<br>• опции поведения behavior.Ruler с префиксом `ruler`. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **disable**(behaviors) | **map.behavior.Manager** | Отключает поведения на карте. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **enable**(behaviors) | **map.behavior.Manager** | Включает поведения на карте. |
| **get**(behaviorName) | **IBehavior** | Возвращает экземпляр поведения по ключу. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **isEnabled**(behaviorName) | Boolean | Проверяет, включено ли в настоящий момент поведение. |

## Описание методов

### disable

```
{map.behavior.Manager} disable(behaviors)
```

Отключает поведения на карте.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| behaviors* | — | Тип: String \| String[]<br>Список отключаемых поведений. |

* Обязательный параметр/опция.

**Пример:**

```
myMap.behaviors.disable('drag');
```

### enable

```
{map.behavior.Manager} enable(behaviors)
```

Включает поведения на карте.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| behaviors* | — | Тип: String \| String[]<br>Список подключаемых поведений. |

* Обязательный параметр/опция.

**Пример:**

```
myMap.behaviors.enable(['ruler', 'multiTouch']);
```

### get

```
{IBehavior} get(behaviorName)
```

**Возвращает** экземпляр поведения по ключу.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| behaviorName* | — | Тип: String<br><br>Имя поведения. |

* Обязательный параметр/опция.

**Пример:**

```
myMap.behaviors.get('drag');
```

## isEnabled

```
{Boolean} isEnabled(behaviorName)
```

Проверяет, включено ли в настоящий момент поведение.

**Возвращает** true, если поведение включено, в противном случае — false.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| behaviorName* | — | Тип: String<br><br>Идентификатор поведения. |

* Обязательный параметр/опция.

**Пример:**

```
// Если поведение "drag" отключено, то подключаем его
if (!(myMap.behaviors.isEnabled('drag'))) {
    myMap.behaviors.enable('drag');
}
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# map.Container

Расширяет **IDomEventEmitter**.

Менеджер контейнера карты. У каждой карты уже есть свой собственный менеджер контейнера, доступный как map.container. Не инстанцируйте экземпляры этого класса без необходимости.

**См.: Map.container**

Конструктор | Поля | События | Методы

## Конструктор

```
map.Container(parentElement)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parentElement * | — | Тип: String \| HTMLElement<br><br>HTML-элемент, в котором будет создана карта или его id. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **beforefullscreenexit** | Событие, предшествующее событию fullscreenexit. Если у данного события вызвать метод Event.preventDefault, то последующее событие fullscreenexit будет отменено. Экземпляр класса **Event**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **fullscreenenter** | Карта перешла в полноэкранный режим. Экземпляр класса **Event**. |
| **fullscreenexit** | Карта вышла из полноэкранного режима. Экземпляр класса **Event**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| | Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **sizechange** | Изменение размера контейнера карты. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldSize: Number[];<br>• newSize: Number[];<br>• oldOffset: Number[];<br>• newOffset: Number[];<br>• preservePixelPosition: Boolean. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **enterFullscreen**() | | Позволяет переводить карту в полноэкранный режим. |
| **exitFullscreen**() | | Позволяет вывести карту из полноэкранного режима. |
| **fitToViewport**([preservePixelPosition]) | | Вызывается при изменении размера контейнера карты для того, чтобы карта применила новый размер. |
| **getElement**() | HTMLElement | Возвращает HTML-элемент карты. |
| **getOffset**() | Number[] | Возвращает смещение контейнера карты в пикселях относительно левого верхнего угла документа. |
| **getSize**() | Number[] | Возвращает размеры контейнера карты в пикселях. |
| **isFullscreen**() | Boolean | Возвращает признак находится ли карта в полноэкранном режиме. |

## Описание событий

### beforefullscreenexit

Событие, предшествующее событию fullscreenexit. Если у данного события вызвать метод Event.preventDefault, то последующее событие fullscreenexit будет отменено. Экземпляр класса **Event**.

### fullscreenenter

Карта перешла в полноэкранный режим. Экземпляр класса **Event**.

### fullscreenexit

Карта вышла из полноэкранного режима. Экземпляр класса **Event**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### sizechange

Изменение размера контейнера карты. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- oldSize: Number[];
- newSize: Number[];
- oldOffset: Number[];
- newOffset: Number[];
- preservePixelPosition: Boolean.

## Описание методов

### enterFullscreen

```
enterFullscreen()
```

Позволяет переводить карту в полноэкранный режим.

### exitFullscreen

```
exitFullscreen()
```

Позволяет вывести карту из полноэкранного режима.

### fitToViewport

```
fitToViewport([preservePixelPosition])
```

Вызывается при изменении размера контейнера карты для того, чтобы карта применила новый размер.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| preservePixelPosition | — | Тип: Boolean<br><br>Сохранять положение центра карты. |

**Пример:**

```
// Изменим размеры контейнера карты
map.container.getElement().style.width = '300px';
// Инициируем пересчет размеров
map.container.fitToViewport();
```

### getElement

```
{HTMLElement} getElement()
```

**Возвращает** HTML-элемент карты.

### getOffset

```
{Number[]} getOffset()
```

**Возвращает** смещение контейнера карты в пикселях относительно левого верхнего угла документа.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### getSize

```
{Number[]} getSize()
```

**Возвращает** размеры контейнера карты в пикселях.

### isFullscreen

```
{Boolean} isFullscreen()
```

**Возвращает** признак находится ли карта в полноэкранном режиме.

## map.Converter

Класс для преобразования глобальных пиксельных координат точки (считаются от левого верхнего угла мира) в локальные (считаются от левого верхнего угла окна) и наоборот. У каждой карты уже есть собственный конвертер, доступный как map.converter. Не инстанцируйте экземпляры этого класса без необходимости.

**См.:** **Map.converter**

Конструктор | Методы

### Конструктор

```
map.Converter(map)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Ссылка на карту. |

* Обязательный параметр/опция.

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **globalToPage**(globalPixelPoint) | Number[] | Преобразовывает глобальные пиксельные координаты точки в локальные. |
| **pageToGlobal**(pagePixelPoint) | Number[] | Преобразовывает локальные пиксельные координаты точки в глобальные. |

## Описание методов

### globalToPage

```
{Number[]} globalToPage(globalPixelPoint)
```

Преобразовывает глобальные пиксельные координаты точки в локальные.

**Возвращает** преобразованные координаты.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| globalPixelPoint* | — | Тип: Number[]<br><br>Пиксельные координаты точки, которые необходимо преобразовать. |

* Обязательный параметр/опция.

**Пример:**

```
// Преобразуем географические координаты в пиксели окна браузера
var projection = map.options.get('projection');
console.log(map.converter.globalToPage(
    projection.toGlobalPixels(
        // географические координаты
        [55, 37],
        map.getZoom()
    )
));
```

## pageToGlobal

```
{Number[]} pageToGlobal(pagePixelPoint)
```

Преобразовывает локальные пиксельные координаты точки в глобальные.

**Возвращает** преобразованные координаты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pagePixelPoint* | — | Тип: Number[]<br><br>Пиксельные координаты точки, которые необходимо преобразовать. |

* Обязательный параметр/опция.

**Пример:**

```
// Преобразуем координаты курсора мыши в геокоординаты
var projection = map.options.get('projection');
$('#map').bind('click', function (e) {
    console.log(projection.fromGlobalPixels(
        map.converter.pageToGlobal([e.pageX, e.pageY]), map.getZoom()
    ).join(', '));
});
```

# map.Copyrights

Менеджер размещённой на карте информации об авторских правах. У каждой карты уже есть собственный менеджер информации об авторских правах, доступный как map.copyrights. Не инстанцируйте экземпляры этого класса без необходимости.

**См.:** **Map.copyrights**

Конструктор | Поля | События | Методы

## Конструктор

```
map.Copyrights(map)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Карта. |

* Обязательный параметр/опция.

**Пример:**

```
// Добавим статическую информацию об авторских правах на карту
var accessor = map.copyrights.add('&copy; Герард Меркатор');
// ...
// Удалим информацию об авторских правах
accessor.remove();
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **event.Manager** | Менеджер событий. Бросает события типа **MapEvent**. |

## События

| Имя | Описание |
|---|---|
| **change** | Событие изменения размещённой на карте информации об авторских правах. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCopyrights — старый массив информации об авторских правах;<br>• newCopyrights — новый массив информации об авторских правах; |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(customCopyrights) | **ICopyrightsAccessor** | Добавляет на карту статическую (не зависящую от текущего центра и масштаба карты) информацию об авторских правах. |
| **addProvider**(provider) | **map.Copyrights** | Добавляет нового провайдера информации об авторских правах. |
| **get**([point[, zoom]]) | **vow.Promise** | Определяет информацию об авторских правах в указанной точке. Возвращает объект-promise, который будет подтверждён и передаст в качестве параметра массив строк/DOM-элементов. |
| **getPromoLink**() | String | Возвращает внешнюю ссылку из блока "Открыть в Яндекс.Картах". |
| **removeProvider**(provider) | **map.Copyrights** | Удаляет провайдера информации об авторских правах. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание полей

### events

```
{event.Manager} events
```

Менеджер событий. Бросает события типа **MapEvent**.

## Описание событий

### change

Событие изменения размещённой на карте информации об авторских правах. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- oldCopyrights — старый массив информации об авторских правах;
- newCopyrights — новый массив информации об авторских правах;

## Описание методов

### add

```
{ICopyrightsAccessor} add(customCopyrights)
```

Добавляет на карту статическую (не зависящую от текущего центра и масштаба карты) информацию об авторских правах.

**Возвращает** объект, позволяющий управлять добавленной информацией.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| customCopyrights* | — | Тип: String \| HTMLElement \| String[] \| HTMLElement[]<br><br>Информация об авторских правах в виде строки, DOM-элемента либо массива строк/DOM-элементов. |

* Обязательный параметр/опция.

### addProvider

```
{map.Copyrights} addProvider(provider)
```

Добавляет нового провайдера информации об авторских правах.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| provider* | — | Тип: **ICopyrightsProvider**<br><br>Провайдер. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Создадим динамический провайдер информации об авторских правах,
// который будет получать актуальную информацию об авторских правах для
// конкретной точки карты с удаленного сервера.
var myProvider = {
        getCopyrights: function (center, zoom) {
            var deferred = ymaps.vow.defer();
            $.ajax('http://some.server/copyrights/?ll=' +
                center.join(',') = '&z=' + zoom, {
                // Сервер должен вернуть массив строк
                success: function (res) {
                    deferred.resolve(res);
                }
            });
            return deferred.promise();
        }
    };
// Добавим провайдер на карту; теперь при изменении центра или уровня
// масштабирования менеджер информации об авторских правах карты будет
// перезапрашивать акутальную информацию с удаленного сервера и отображать
// полученные данные автоматически.
map.copyrights.addProvider(myProvider);
```

## get

```
{vow.Promise} get([point[, zoom]])
```

Определяет информацию об авторских правах в указанной точке. Возвращает объект-promise, который будет подтверждён и передаст в качестве параметра массив строк/DOM-элементов.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| point | — | Тип: Number[]<br><br>Точка (в географических координатах), для которой требуется определить информацию об авторских правах. Если не указана, берётся текущий центр карты. |
| zoom | — | Тип: Number<br><br>Уровень масштабирования, для которого требуется определить информацию об авторских правах. Если не указан, берётся текущий уровень масштабирования карты. |

## getPromoLink

```
{String} getPromoLink()
```

**Возвращает** внешнюю ссылку из блока "Открыть в Яндекс.Картах".

## removeProvider

```
{map.Copyrights} removeProvider(provider)
```

Удаляет провайдера информации об авторских правах.

**Возвращает** ссылку на себя.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| provider* | — | Тип: **ICopyrightsProvider**<br><br>Провайдер. |

* Обязательный параметр/опция.

# map.GeoObjects

Расширяет **IGeoObjectCollection**.

Коллекция геообъектов карты. У каждой карты уже есть своя коллекция геообъектов, доступная как map.geoObjects. Не инстанцируйте экземпляры этого класса без необходимости.

**См.: Map.geoObjects**

Конструктор | Поля | События | Методы

## Конструктор

```
map.GeoObjects(map[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Карта. |
| options | — | Тип: Object<br><br>Опции коллекции геообъектов. Через опции map.geoObjects можно задавать настройки геообъектов, добавленных на карту:<br>• опции кластеризаторов с префиксом `clusterer`;<br>• опции кластеров с префиксом `cluster`. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| add | Добавлен (вставлен) дочерний геообъект. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• index: Integer — индекс добавленного геообъекта;<br>• child: IGeoObject — ссылка на добавленный геообъект.<br>Унаследовано от IGeoObjectCollection. |
| boundschange | Изменение координат географической области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса Event.<br>Унаследовано от IGeoObjectCollection. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| pixelboundschange | Изменение пиксельных координат области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса Event.<br>Унаследовано от IGeoObjectCollection. |
| remove | Удален дочерний геообъект. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• index: Integer — индекс удаленного геообъекта;<br>• child: IGeoObject — ссылка на удаленный геообъект.<br>Унаследовано от IGeoObjectCollection. |
| set | В коллекцию записан новый дочерний геообъект. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• index: Integer — индекс геообъекта;<br>• child: IGeoObject — ссылка на новый геообъект;<br>• prevChild: IGeoObject — ссылка на предыдущее значение для данного индекса.<br>Унаследовано от IGeoObjectCollection. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| add(child[, index]) | map.GeoObjects | Добавляет (вставляет) дочерний геообъект в коллекцию. |
| each(callback[, context]) | | Для каждого дочернего геообъекта, вызывает функцию-обработчик. |
| get(index) | IGeoObject | Возвращает дочерний геообъект с заданным индексом. |
| getBounds() | Number[][] \| null | Возвращает географические координаты области, охватывающей коллекцию и ее дочерние геообъекты. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getIterator**() | **IIterator** | Возвращает итератор по коллекции. |
| **getLength**() | Integer | Возвращает длину коллекции. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getPixelBounds**() | Number[][] \| null | Возвращает глобальные пиксельные координаты области, охватывающей коллекцию и ее дочерние геообъекты. |
| **indexOf**(object) | Integer | Возвращает индекс дочернего геообъекта. Если геообъект в коллекции не найден, возвращается -1. |
| **remove**(child) | **map.GeoObjects** | Удаляет дочерний геообъект из коллекции. |
| **removeAll**() | **map.GeoObjects** | Очищает коллекцию. |
| **set**(index, child) | **map.GeoObjects** | Записывает в коллекцию новый дочерний геообъект. |
| **splice**(index, number) | **GeoObjectCollection** | Удаляет геообъекты из коллекции. Если необходимо, вставляет на их место другие. Объекты, которые будут вставлены вместо удаляемых, передаются как дополнительные параметры (после параметра number). |

## Описание методов

### add

```
{map.GeoObjects} add(child[, index])
```

Добавляет (вставляет) дочерний геообъект в коллекцию.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **IGeoObject**<br>Дочерний геообъект. |
| index | — | Тип: Integer<br>Индекс, в который добавляется новый объект. По умолчанию объект добавляется в конец коллекции. |

* Обязательный параметр/опция.

### each

```
each(callback[, context])
```

Для каждого дочернего геообъекта, вызывает функцию-обработчик.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик. |
| context | — | Тип: Object<br><br>Контекст выполнения функции-обработчика. |

* Обязательный параметр/опция.

**Пример:**

```
// Отображаем индекс геообъекта в коллекции как содержимое его иконки.
myGeoObjects.events.add(["add", "remove", "set"], function () {
    this.each(function (el, i) {
        el.properties.set("iconContent", i);
    })
}, myGeoObjects);
```

## get

```
{IGeoObject} get(index)
```

Возвращает дочерний геообъект с заданным индексом.

**Возвращает** дочерний геообъект.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс. |

* Обязательный параметр/опция.

## getBounds

```
{Number[][] | null} getBounds()
```

**Возвращает** географические координаты области, охватывающей коллекцию и ее дочерние геообъекты.

## getIterator

```
{IIterator} getIterator()
```

**Возвращает** итератор по коллекции.

## getLength

```
{Integer} getLength()
```

**Возвращает** длину коллекции.

## getPixelBounds

```
{Number[][] | null} getPixelBounds()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** глобальные пиксельные координаты области, охватывающей коллекцию и ее дочерние геообъекты.

## indexOf

```
{Integer} indexOf(object)
```

**Возвращает** индекс дочернего геообъекта. Если геообъект в коллекции не найден, возвращается -1.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| object* | — | Тип: Object<br><br>Дочерний геообъект. |

* Обязательный параметр/опция.

## remove

```
{map.GeoObjects} remove(child)
```

Удаляет дочерний геообъект из коллекции.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **IGeoObject**<br><br>Удаляемый геообъект. |

* Обязательный параметр/опция.

**Пример:**

```
// При клике по геообъекту удаляем его из коллекции.
myGeoObjects.events.add("click", function (e) {
    if (e.get("target").getParent() == this) {
        this.remove(e.get("target"));
    }
}, myGeoObjects);
```

## removeAll

```
{map.GeoObjects} removeAll()
```

Очищает коллекцию.

**Возвращает** ссылку на себя.

## set

```
{map.GeoObjects} set(index, child)
```

Записывает в коллекцию новый дочерний геообъект.

**Возвращает** ссылку на себя.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс. |
| child* | — | Тип: **IGeoObject**<br><br>Дочерний геообъект. |

* Обязательный параметр/опция.

## splice

```
{GeoObjectCollection} splice(index, number)
```

Удаляет геообъекты из коллекции. Если необходимо, вставляет на их место другие. Объекты, которые будут вставлены вместо удаляемых, передаются как дополнительные параметры (после параметра number).

**Возвращает** коллекция удаленных геообъектов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс геообъекта, с которого начинается удаление. |
| number* | — | Тип: Integer<br><br>Количество удаляемых геообъектов. |

* Обязательный параметр/опция.

**Пример:**

```
// Удаляет второй объект.
myGeoObjects.splice(1, 1);
// Вставляет новый объект obj во вторую позицию.
myGeoObjects.splice(1, 0, obj);
// Заменяет второй объект на новый объект obj.
myGeoObjects.splice(1, 1, obj);
```

# map.Hint

Расширяет **IHintManager**, IHintSharingManager.

Менеджер хинта карты. У каждой карты уже есть собственный менеджер хинта, доступный как myMap.hint. В один момент на карте может быть открыт только один хинт, управляемый менеджером. Не создавайте экземпляры этого класса без необходимости.

**См.: Hint Map.hint**

Конструктор | Поля | Методы

## Конструктор

```
map.Hint(map)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Ссылка на объект карты. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **close**([force]) | **vow.Promise** | Закрывает инфо-объект.<br>Унаследован от **IPopupManager**. |
| **destroy**() | | Отключает менеджер инфо-объекта.<br>Унаследован от **IPopupManager**. |
| **getData**() | Object \| null | Возвращает данные инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **getOptions**() | **IOptionManager** \| null | Возвращает менеджер опций или `null`.<br>Унаследован от **IPopupManager**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от **IPopupManager**. |
| **getOverlaySync**() | **IOverlay** \| null | Возвращает оверлей, если тот существует.<br>Унаследован от **IPopupManager**. |
| **getPosition**() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **isOpen**() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от **IPopupManager**. |
| **open**([position[, data[, options]]]) | **vow.Promise** | Открывает инфо-объект в указанной позиции.<br>Унаследован от **IPopupManager**. |
| **setData**(data) | **vow.Promise** | Задаёт инфо-объекту новые данные.<br>Унаследован от **IPopupManager**. |
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br><br>Унаследован от **IPopupManager**. |

# map.layer

## map.layer.Manager

Расширяет **ILayer**, **IMapObjectCollection**.

Менеджер слоев карты.

**См.:** **Map.layers**

Конструктор | Поля | События | Методы

### Конструктор

```
map.layer.Manager(map[, options])
```

Создает экземпляр класса для работы со слоями карты. У любой карты уже есть свой менеджер слоёв, доступный как map.layers. Не инстанцируйте экземпляры этого класса без необходимости.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Карта |
| options | — | Тип: Object<br><br>Опции слоев карты. Через опции map.layers можно задавать настройки слоев, добавленных на карту. Опции слоев хотспотов задаются с префиксом 'hotspotLayer'. |
| options.trafficImageZIndex | 201 | Тип: Number<br><br>Z-index картиночного слоя пробок. |
| options.trafficInfoZIndex | 1 | Тип: Number<br><br>Приоритет хотспотного слоя инфоточек. |
| options.trafficJamZIndex | 0 | Тип: Number<br><br>Приоритет хотспотного слоя пробок. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **add** | Произошло добавление дочернего объекта.<br>Унаследовано от **ICollection**. |
| **brightnesschange** | Событие изменения яркости слоя.<br>Унаследовано от **ILayer**. |
| **copyrightschange** | Событие изменения доступной информации об авторских правах.<br>Унаследовано от **ILayer**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **remove** | Произошло удаление дочернего объекта.<br>Унаследовано от **ICollection**. |
| **zoomrangechange** | Событие изменения доступной информации о диапазоне уровней масштабирования.<br>Унаследовано от **ILayer**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(object) | **ICollection** | Добавляет в коллекцию дочерний объект.<br>Унаследован от **ICollection**. |
| **each**(callback[, context]) | | Перебирает все элементы коллекции, вызывая для каждого функцию-обработчик. |
| **getBrightness**() | Number | Необязательный метод. Возвращает яркость слоя. В зависимости от суммарной яркости слоев, добавленных на карту, выбирается цвет логотипа и копирайтов на карте. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **ILayer**. |
| **getCopyrights**(coords, zoom) | **vow.Promise** | Необязательный метод. Запрашивает информацию об авторских правах в указанной точке с указанным коэффициентом масштабирования. Возвращает объект-promise, который будет подтверждён и передаст в качестве результата массив строк или DOM-элементов с информацией об авторских правах.<br>Унаследован от **ILayer**. |
| **getIterator**() | **IIterator** | Возвращает итератор по коллекции.<br>Унаследован от **ICollection**. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getZoomRange**(point) | **vow.Promise** | Необязательный метод. Определяет доступный в указанной точке диапазон уровней масштабирования. Если данные есть, то возвращенный объект-promise будет подтвержден и передаст в качестве результата массив из двух чисел — минимальный и максимальный доступный в точке уровень масштабирования. Если данных нет, то promise отклоняется с указанием ошибки.<br>Унаследован от **ILayer**. |
| **remove**(object) | **ICollection** | Удаляет из коллекции дочерний объект.<br>Унаследован от **ICollection**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание методов

### each

```
each(callback[, context])
```

Перебирает все элементы коллекции, вызывая для каждого функцию-обработчик.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Функция-обработчик. |
| context | — | Тип: Object<br><br>Контекст выполнения функции. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# map.margin

## map.margin.Accessor

Объект, который предоставляет доступ к прямоугольной области в менеджере отступов.

Конструктор | Поля | События | Методы

### Конструктор

```
map.margin.Accessor(screenArea)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| screenArea * | — | Тип: Object<br><br>Прямоугольная область, которая задана в экранных координатах. Область описывается в виде объекта, который содержит информаци о смещении от краев карты (left, top, right, bottom) и размеры области (width, height). Значения могут быть заданы в виде процентов от ширины/высоты контейнера карты.<br><br>Не инстанцируйте экземпляры этого класса без необходимости. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | | Менеджер событий. |

### События

| Имя | Описание |
|---|---|
| **change** | Прямоугольная область была изменена. |
| **remove** | Аксесссор был удален из менеджера отступов. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getArea**() | Object | Возвращает прямоугольная область. |
| **remove**() | **map.margin.Accessor** | Удаление прямоугольной области из менеджера отступов. |
| **setArea**(screenArea) | **map.margin.Accessor** | Возвращает ссылку на себя. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание полей

### events

```
events
```

Менеджер событий.

## Описание событий

### change

Прямоугольная область была изменена.

### remove

Аксесссор был удален из менеджера отступов.

## Описание методов

### getArea

```
{Object} getArea()
```

**Возвращает** прямоугольная область.

### remove

```
{map.margin.Accessor} remove()
```

Удаление прямоугольной области из менеджера отступов.

**Возвращает** ссылку на себя.

### setArea

```
{map.margin.Accessor} setArea(screenArea)
```

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| screenArea* | — | Тип: Object<br><br>Прямоугольная область, которая задана в экранных координатах. Область описывается в виде объекта, который содержит информаци о смещении от краев карты (left, top, right, bottom) и размеры области (width, height). Значения могут быть заданы в виде процентов от ширины/высоты контейнера карты. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
accessor.setArea({
    top: 50,
    left: 50,
    width: 100,
    height: 100
});
```

# map.margin.Manager

Расширяет **IEventEmitter**.

Менеджер отступов карты.

Менеджер для рассчета оптимальных отступов от краев контейнера карты.

Менеджер принимает на вход описание прямоугольных областей в экранных координатах, которые обозначают занятые области контейнера карты.

Отступы могут учитываться при установке текущей видимой области карты, чтобы добиться наилучшего отображения данных на карте. Данные на карте никогда не попадут под занятые области контейнера карты. Map.setBounds

Не инстанцируйте экземпляры этого класса без необходимости.

Конструктор | Поля | События | Методы

## Конструктор

```
map.margin.Manager()
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map** |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение отступов карты. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **addArea**(screenArea) | **map.margin.Accessor** | Добавление новой прямоугольной области. |
| **destroy**() | **map.margin.Manager** | Уничтожение менеджера отступов. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getMargin() | Number[] | Возвращает текущее значение отступов от краев карты. Значения в массиве идут в следующем порядке: верхний отступ, правый отступ, нижний отступ, левый отступ. |

## Описание событий

### change

Изменение отступов карты.

## Описание методов

### addArea

```
{map.margin.Accessor} addArea(screenArea)
```

Добавление новой прямоугольной области.

**Возвращает** объект, который предоставляет доступ к добавленной области. Добавленную прямоугольную область можно удалить только через полученный объект.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| screenArea* | — | Тип: Object<br><br>Прямоугольная область, которая задана в локальных координатах. Область описывается в виде объекта, который содержит информацию о смещении от краев карты (left, top, right, bottom) и размеры области (width, height). Значения могут быть заданы в виде процентов от ширины/высоты контейнера карты. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Смещение от верхнего левого угла.
map.margin.addArea({
    left: 0,
    top: 0,
    width: 78,
    height: 101
});
console.log(map.margin.getMargin()); // [0, 0, 0, 78]
```

**2.**

```
// Смещение от нижнего правого угла.
map.margin.addArea({
    right: 50,
    bottom: 50,
    width: 26,
    height: 25
});
console.log(map.margin.getMargin()); // [0, 0, 75, 0]
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**3.**

```
// Задание смещения в процентах.
map.margin.addArea({
    left: 50,
    bottom: '1%',
    width: 20,
    height: 20
});
console.log(map.margin.getMargin()); // [0, 0, 26, 0]
```

**4.**

```
// Задание размера в процентах.
map.margin.addArea({
    left: 0,
    top: 50,
    width: '100%',
    height: 25
});
console.log(map.margin.getMargin()); // [75, 0, 0, 0]
```

**5.**

```
// Задание нескольких областей.
map.margin.addArea({
    top: 10,
    left: 10,
    width: 20,
    height: 20
});
map.margin.addArea({
    top: 20,
    right: 40,
    width: 100,
    height: 100
});
map.margin.addArea({
    bottom: 20,
    left: 30,
    width: 120,
    height: 30
});
console.log(map.margin.getMargin()); // [120, 0, 50, 0]
```

## destroy

```
{map.margin.Manager} destroy()
```

Уничтожение менеджера отступов.

**Возвращает** ссылку на себя.

## getMargin

```
{Number[]} getMargin()
```

**Возвращает** текущее значение отступов от краев карты. Значения в массиве идут в следующем порядке: верхний отступ, правый отступ, нижний отступ, левый отступ.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# map.pane

## map.pane.Manager

Менеджер пейнов карты. У каждой карты уже есть свой менеджер пейнов, доступный как map.panes. Не создавайте экземпляры этого класса без необходимости. Список ключей доступных по умолчанию пейнов карты и их zIndex'ы:

- 'ground': **pane.MovablePane** (zIndex: 100) — самый нижний пейн, предназначенный для размещения подложки карты;
- 'areas': **pane.MovablePane** (zIndex: 200) — пейн площадных объектов, таких как многоугольники;
- 'shadows': **pane.MovablePane** (zIndex: 300) — пейн теней объектов карты, находящихся выше;
- 'places': **pane.MovablePane** (zIndex: 400) — пейн точечных объектов, таких как метки;
- 'events': **pane.EventsPane** (zIndex: 500) — пейн, предназначенный для слушания событий карты;
- 'overlaps': **pane.MovablePane** (zIndex: 600) — пейн для объектов, не требующих использования активных областей для реализации своей интерактивности;
- 'balloon': **pane.MovablePane** (zIndex: 700) — пейн балуна;
- 'outerBalloon': **pane.MovablePane** (zIndex: 800) — внешний пейн балуна;
- 'controls': **pane.StaticPane** (zIndex: 900) — пейн элементов управления карты;
- 'copyrights': **pane.StaticPane** (zIndex: 1000) — пейн копирайтов;
- 'hint': **pane.StaticPane** (zIndex: 1100) — пейн хинта;
- 'outerHint': **pane.StaticPane** (zIndex: 1200) — внешний пейн хинта.

**См.:** **Map.panes**

Конструктор | Методы

### Конструктор

```
map.pane.Manager(map)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Карта. |

* Обязательный параметр/опция.

**Пример:**

```
// Добавим watermark над контейнером карты.
// Для этого изменим background у контейнера событий.
map.panes.get('events').getElement().style.backgroundImage = 'my/background/
image';
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **append**(key, pane) | | Добавляет новый пейн в карту. Ключ добавляемого пейна должен быть уникальным в рамках текущего набора ключей пейнов карты. |
| **destroy**() | | Деструктор. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| get(key) | IPane \| null | Возвращает пейн карты с заданным ключом, либо null, если такой пейн в данной карте отсутствует. |
| getLower() | String | Принимает в качестве аргументов ключи пейнов карты и возвращает ключ самого нижнего пейна из переданного набора. Если ни один ключ не передан, то будет произведен поиск по всему набору пейнов карты. |
| getUpper() | String | Принимает в качестве аргументов ключи пейнов карты и возвращает ключ самого верхнего пейна из переданного набора. Если ни один ключ не передан, то будет произведен поиск по всему набору пейнов карты. |
| insertBefore(key, pane, referenceKey) | | Вставляет новый пейн перед другим пейном карты. Ключ добавляемого пейна должен быть уникальным в рамках текущего набора ключей пейнов карты. |
| remove(pane) | | Удаляет пейн из карты. |

## Описание методов

### append

```
append(key, pane)
```

Добавляет новый пейн в карту. Ключ добавляемого пейна должен быть уникальным в рамках текущего набора ключей пейнов карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Ключ добавляемого пейна. |
| pane* | — | Тип: IPane<br><br>Добавляемый пейн. |

* Обязательный параметр/опция.

### destroy

```
destroy()
```

Деструктор.

### get

```
{IPane | null} get(key)
```

**Возвращает** пейн карты с заданным ключом, либо null, если такой пейн в данной карте отсутствует.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Ключ пейна. |

* Обязательный параметр/опция.

## getLower

```
{String} getLower()
```

Принимает в качестве аргументов ключи пейнов карты и возвращает ключ самого нижнего пейна из переданного набора. Если ни один ключ не передан, то будет произведен поиск по всему набору пейнов карты.

**Возвращает** ключ самого нижнего пейна карты.

## getUpper

```
{String} getUpper()
```

Принимает в качестве аргументов ключи пейнов карты и возвращает ключ самого верхнего пейна из переданного набора. Если ни один ключ не передан, то будет произведен поиск по всему набору пейнов карты.

**Возвращает** ключ самого верхнего пейна карты.

## insertBefore

```
insertBefore(key, pane, referenceKey)
```

Вставляет новый пейн перед другим пейном карты. Ключ добавляемого пейна должен быть уникальным в рамках текущего набора ключей пейнов карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Ключ добавляемого пейна. |
| pane* | — | Тип: **IPane**<br><br>Добавляемый пейн. |
| referenceKey* | — | Тип: String<br><br>Ключ пейна, перед которым осуществляется вставка. |

* Обязательный параметр/опция.

## remove

```
remove(pane)
```

Удаляет пейн из карты.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pane* | — | Тип: **IPane**<br><br>Удаляемый пейн. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# map.ZoomRange

Расширяет **IEventEmitter**.

Менеджер коэффициентов масштабирования карты. У каждой карты уже есть свой менеджер коэффициентов масштабирования, доступный как map.zoomRange. Не инстанцируйте экземпляры этого класса без необходимости.

**См.: Map.zoomRange**

Конструктор | Поля | События | Методы

## Конструктор

```
map.ZoomRange(map, constraints)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Карта. |
| constraints * | — | Тип: Number[]<br><br>Массив, содержащий минимальный и максимальный зум карты. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **change** | Произошло изменение диапазона масштабов карты. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **get**() | **vow.Promise** | Определяет доступный в указанной точке диапазон уровней масштабирования. Возвращает объект-promise, который будет подтверждён и получит в качестве параметра массив из двух чисел — максимальный и минимальный уровень масштабирования в данной точке. |
| **getCurrent**() | Number[] | Возвращает текущее (последнее полученное) значение минимального и максимального уровня масштабирования карты. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание событий

### change

Произошло изменение диапазона масштабов карты.

## Описание методов

### get

```
{vow.Promise} get()
```

Определяет доступный в указанной точке диапазон уровней масштабирования. Возвращает объект-promise, который будет подтверждён и получит в качестве параметра массив из двух чисел — максимальный и минимальный уровень масштабирования в данной точке.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coords* | — | Тип: |

* Обязательный параметр/опция.

**Пример:**

```
// Найдем координаты офиса компании Яндекс и покажем его на карте
// на максимально возможном уровне масштабирования.
ymaps.geocode('Москва, улица Льва Толстого, 16').then(function (res) {
    var coords = res.geoObjects.get(0).geometry.getCoordinates();
    map.zoomRange.get(coords).then(function (range) {
        map.setCenter(coords, range[1]);
    });
});
```

### getCurrent

```
{Number[]} getCurrent()
```

**Возвращает** текущее (последнее полученное) значение минимального и максимального уровня масштабирования карты.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# MapEvent

Расширяет **Event**.

Объект, описывающий произошедшее над картой событие. Имена полей, доступных через метод Event.get:

- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы (также доступно под именем position);
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

Конструктор | Методы

## Конструктор

```
MapEvent(originalEvent[, sourceEvent])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| originalEvent * | — | Тип: Object<br><br>Данные, связанные с событием. Должны содержать поле "map" указывающее на карту, над которой произошло событие. |
| sourceEvent | — | Тип: **IEvent**<br><br>Исходное событие. |

* Обязательный параметр/опция.

**Пример:**

```
// Откроем балун в точке клика по карте
map.events.add('click', function (e) {
    map.balloon.open(e.get('coords'), 'Щелк!');
});
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **allowMapEvent**() | | Разрешает распространение данного события на карту.<br><br>Унаследован от **IEvent**. |
| **callMethod**(name) | Object | Вызывает указанный метод. Операция эквивалентна поиску поля через get и вызову с передачей в качестве контекста originalEvent. Все аргументы после первого передаются в качестве параметров вызываемому методу.<br><br>Унаследован от **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **get**(name) | Object | Возвращает значение поля из originalEvent. originalEvent всегда имеет следующие поля:<br>• type — строковый тип события;<br>• target — ссылка на объект, породивший событие.<br>Унаследован от **Event**. |
| **getSourceEvent**() | **IEvent** \| null | Возвращает исходное событие.<br>Унаследован от **IEvent**. |
| **isDefaultPrevented**() | Boolean | Проверяет, отменена ли реакция по умолчанию на событие в рамках системы событий API Яндекс.Карт.<br>Унаследован от **Event**. |
| **isImmediatePropagationStopped**() | Boolean | Определяет, прекращено ли распространение события в системе событий API Яндекс.Карт.<br>Унаследован от **Event**. |
| **isMapEventAllowed**() | Boolean | Возвращает true, если событие карты разрешено.<br>Унаследован от **IEvent**. |
| **isPropagationStopped**() | Boolean | Проверяет, остановлено ли распространение события по иерархии объектов и коллекций в системе событий API Яндекс.Карт.<br>Унаследован от **Event**. |
| **preventDefault**() | | Отменяет реакцию по умолчанию на событие в рамках системы событий API Яндекс.Карт. Вызов этого метода никак не влияет на распространение исходного DOM-события (если оно есть) по DOM-дереву.<br>Унаследован от **Event**. |
| **stopImmediatePropagation**() | | Прекращает распространение события в системе событий API Яндекс.Карт. Вызов этого метода никак не влияет на распространение исходного DOM-события (если оно есть) по DOM-дереву.<br>Унаследован от **Event**. |
| **stopPropagation**() | | Прекращает распространение события по иерархии объектов и коллекций в системе событий API Яндекс.Карт. Вызов этого метода никак не влияет на распространение исходного DOM-события (если оно есть) по DOM-дереву.<br>Унаследован от **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# mapType

## mapType.storage

Статический объект.

Экземпляр класса **util.Storage**

Хранилище типов карты.

Методы

```
mapType.storage()
```

**Пример:**

```
// Добавим свой тип карты

// Допустим, у нас есть свой слой карты
var myLayer = function () {
    return new ymaps.Layer('http://some.server/?%c');
}
// Добавляем его в хранилище слоёв
ymaps.layer.storage.add('my#layer', myLayer);
// Создаём свой тип карты, состоящий из одного слоя
var myType = new ymaps.MapType('Мой тип карты', ['my#layer']);
// Добавляем его в хранилище типов карты
ymaps.mapType.storage.add('my#mapType', myType);
// Теперь можем устанавливать свой тип карте
map.setType('my#mapType');
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# MapType

Тип карты.

Конструктор | Методы

## Конструктор

```
MapType(name, layers)
```

Создает экземпляр типа карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name * | — | Тип: String<br><br>Название типа. |
| layers * | — | Тип: Function[] \| String[]<br><br>Массив, содержащий конструкторы слоев или ключи. |

* Обязательный параметр/опция.

**Пример:**

```
// Создадим собственный тип карты, состоящий из спутниковых снимков MapQuest
// с наложенным поверх Яндекс.Гибридом.

// Класс тайлов MapQuest
var MQLayer = function () {
    var layer = new ymaps.Layer('http://oatile%d.mqcdn.com/naip//%z/%x/%y.jpg');
    // Копирайты
    layer.getCopyrights = function () {
        return ymaps.vow.resolve('Data, imagery and map information provided by
MapQuest, Open Street Map and contributors, CC-BY-SA');
    };
    // Диапазон доступных масштабов
    layer.getZoomRange = function () {
        return ymaps.vow.resolve([0, 18]);
    };

    return layer;
};
// Добавим слой под ключом
ymaps.layer.storage.add('mq#aerial', MQLayer);
// Создадим тип карты, состоящий из слоёв 'mq#aerial' и 'yandex#skeleton'
var myMapType = new ymaps.MapType('MQ + Ya', ['mq#aerial', 'yandex#skeleton']);
// Добавим в хранилище типов карты
ymaps.mapType.storage.add('mq_ya#hybrid', myMapType);
// Теперь мы можем задавать наш тип карты любой карте
map.setType('mq_ya#hybrid');
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getLayers() | Function[] \| String[] | Возвращает список слоев данного типа карты — массив конструкторов или ключей для слоев. |
| getName() | String | Возвращает название типа карты. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### getLayers

```
{Function[] | String[]} getLayers()
```

**Возвращает** список слоев данного типа карты — массив конструкторов или ключей для слоев.

### getName

```
{String} getName()
```

**Возвращает** название типа карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: |

* Обязательный параметр/опция.

## meta

Статический объект.

Информация об API.

Поля

```
meta()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **coordinatesOrder** | String | Порядок координат, который используется в API. Возможные значения:<br>• latlong — [широта, долгота];<br>• longlat — [долгота, широта].<br>Задается GET-параметром "coordorder" при подключении API.<br>Подробнее про параметры подключения API |
| **countryCode** | String | Двузначный код страны. Возвращается в формате ISO 3166-1.<br>Задается GET-параметром "lang" при подключении API.<br>Подробнее про параметры подключения API |
| **languageCode** | String | Двузначный код языка. Возвращается в формате ISO 639-1.<br>Задается GET-параметром "lang" при подключении API.<br>Подробнее про параметры подключения API |
| **mode** | String | Режим работы API Яндекс.Карт. Возможные значения:<br>• release;<br>• debug. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Задается GET-параметром "mode" при подключении API.<br>Подробнее про параметры подключения API |
| **ns** | Object | Ссылка на пространство имен API Яндекс.Карт.<br>Имеет значение независимо от значения параметра "ns" при подключении API. |
| **version** | String | Версия API Яндекс.Карт. |

## Описание полей

### coordinatesOrder

```
{String} coordinatesOrder
```

Порядок координат, который используется в API. Возможные значения:

- latlong — [широта, долгота];
- longlat — [долгота, широта].

Задается GET-параметром "coordorder" при подключении API.

Подробнее про параметры подключения API

### countryCode

```
{String} countryCode
```

Двузначный код страны. Возвращается в формате ISO 3166-1.

Задается GET-параметром "lang" при подключении API.

Подробнее про параметры подключения API

### languageCode

```
{String} languageCode
```

Двузначный код языка. Возвращается в формате ISO 639-1.

Задается GET-параметром "lang" при подключении API.

Подробнее про параметры подключения API

### mode

```
{String} mode
```

Режим работы API Яндекс.Карт. Возможные значения:

- release;
- debug.

Задается GET-параметром "mode" при подключении API.

Подробнее про параметры подключения API

### ns

```
{Object} ns
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Ссылка на пространство имен API Яндекс.Карт.

Имеет значение независимо от значения параметра "ns" при подключении API.

## version

```
{String} version
```

Версия API Яндекс.Карт.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# modules

Статический объект.

Модульная система, на основе которой работает API Яндекс.Карт.

API Яндекс.Карт состоит из большого количества взаимосвязанных модулей. Модуль — это некоторая программная единица. К примеру, класс, конкретная реализация класса, статический объект или функция. Модульная система гарантирует, что при инициализации конкретного модуля все необходимые ему для работы модули уже будут проинициализированы.

Модульная система предоставляет асинхронный доступ, так как может потребоваться произвести загрузку недостающих модулей.

В модульную систему можно добавлять свои собственные модули.

```
modules()
```

# modules.define

Статическая функция.

Определение модуля в модульной системе.

---

**Примечание:**
Рекомендуется создавать собственные модули в своем пространстве имен, чтобы случано не заменить необходимые для работы API модули.

---

**Возвращает:** Ссылку на самого себя. (тип **modules**).

```
modules.define(module[, depends, resolveCallback[, context]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| module* | — | Тип: String<br><br>Название модуля. |
| depends | — | Тип: String[]<br><br>Массив названий необходимых модулей. Этот аргумент может быть пропущен. |
| resolveCallback* | — | Тип: Function |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Функция, которая определяет модуль. Первым аргументом в resolveCallback поступит provide-функция, в которую нужно будет передать модуль. Вызов provide-функции может быть отложен во времени. Последующие аргументы — это модули, указанные в зависимостях. Порядок модулей будет соответствовать порядку в массиве depends. В случае если модуль не может быть зарезолвен, то необходимо об этом уведомить модульную систему. Это можно сделать передав в provide-функцию второй аргумент. Второй аргумент будет передан в errorCallback и в promise как ошибка в запрос модуля. Впоследствии модуль можно будет запросить повторно. |
| context | — | Тип: Object<br><br>Контекст исполнения функции. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Задание собственного модуля. 'plugin.*' - это собственное пространство имен.
ymaps.modules.define('plugin.CustomModule', function (provide) {
    var CustomModule = function (defValue) {
        this.field = defValue;
    };
    provide(CustomModule);
});
// Запрос модулей.
ymaps.modules.require(['plugin.CustomModule'])
    .spread(
        function (CustomModule) {
            // ...
        },
        this
    );
```

**2.**

```
// Задание собственного асинхронного модуля.
ymaps.modules.define('CustomAsyncModule', function (provide) {
    // Для работы модуля необходимо загрузить внешний скрипт.
    $.getScript( "ajax/test.js")
        .done(function( script, textStatus ) {
            function CustomAsyncModule () {
                // ...
            }
            // Вызов provide-функции после загрузки скрипта.
            provide(CustomAsyncModule);
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
        });
});
// Запрос модулей
ymaps.modules.require(['CustomAsyncModule'])
    .spread(
        function (CustomAsyncModule) {
            // ...
        },
        this
    );
```

**3.**

```
// Реализуем собственный асинхронный модуль с учетом ошибки.
ymaps.modules.define('plugin.CustomModule', function (provide) {
    $.getScript( "ajax/test.js" )
        .done(function( script, textStatus ) {
            // ... Обработка удачной загрузки скрипта.
            provide({ a: 1 });
        })
        .fail(function( jqxhr, settings, exception ) {
            // Уведомляем модульную систему о том, что модуль не может быть
подготовлен сейчас.
            provide(null, new Error('Произошла ошибка при загрузке'));
        });
});

// Запрос модулей с обработкой ошибки.
ymaps.modules.require(['plugin.CustomModule'])
    .spread(
        function (CustomModule) {
            // ...
        },
        function (error) {
            console.log(error.messsage); // "Произошла ошибка при загрузке".
            // Код обработки ошибки получения модуля. Модуль можно запросить
повторно.
        },
        this
    );
```

**4.**

```
// Задание собственного модуля с зависимостями.
ymaps.modules.define('plugin.CustomLayoutModule', [
     'templateLayoutFactory',
     'layout.storage'
], function (provide, templateLayoutFactory, layoutStorage) {
    var customLayoutClass = templateLayoutFactory.createClass(
        '<div>My Layout</div>',
        {
            build: function () {
                 customLayoutClass.superclass.build.call(this);
                 // ...
            },
            clear: function () {
                 // ...
                 customLayoutClass.superclass.clear.call(this);
            }
        }
    );
    layoutStorage.add('customLayout', customLayoutClass);
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
    provide(customLayoutClass);
});
// Запрос своего макета.
ymaps.modules.require(['plugin.CustomLayoutModule'])
    .spread(
        function (CustomLayoutModule) {
            // ...
        },
        this
    );
```

## modules.isDefined

Статическая функция.

Проверка доступности модуля по названию.

**Возвращает:** true — модуль был определен, false — нет. (тип Boolean).

```
modules.isDefined(module)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| module* | — | Тип: String<br><br>Модуль |

* Обязательный параметр/опция.

**Пример:**

```
if (ymaps.modules.isDefined('plugin.CustomModule')) {
    ymaps.modules.require(['plugin.CustomModule'])
        .spread(
            function (CustomModule) {
                // ...
            }
        );
}
```

## modules.require

Статическая функция.

Запрос получения модулей.

**Возвращает:** Объект-обещание, который подтверждается запрошенными модулями. Либо отклоняется в случае, если произошла ошибка. К примеру, один из запрошенных модулей отсутствует в модульной системе. (тип **vow.Promise**).

```
modules.require(modules[, successCallback[, errorCallback[, context]]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| modules* | — | Тип: String \| String[]<br><br>Название модуля или массив названий модулей. |
| successCallback | — | Тип: Function |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Функция, которая будет вызвана после получения всех модулей. В функцию как аргументы будут переданы запрошенные сущности. Порядок аргументов будет соответствовать порядку в массиве modules. |
| errorCallback | — | Тип: Function<br><br>Функция, которая будет вызвана в случае ошибки. |
| context | — | Тип: Object<br><br>Контекст исполнения функции обратного вызова. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Запрос модулей 'Map' и 'Placemark' при помощи объекта-обещания.
ymaps.modules.require(['Map', 'Placemark'])
    .spread(
        function (Map, Placemark) {
            var myMap = new Map("map", {
                center: [55.72, 37.64],
                zoom: 5
            });
            myMap.geoObjects.add(
                new Placemark(myMap.getCenter())
            );
        },
        function (error) {
            // Обработка ошибки.
        },
        this
    );
```

**2.**

```
// Запрос модуля 'Map'.
ymaps.modules.require('Map')
    .spread(
        function (Map) {
            var myMap = new Map("map", {
                center: [55.72, 37.64],
                zoom: 5
            });
        },
        this
    );
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# Monitor

Объект, отслеживающий изменения определенных полей данных на заданном менеджере данных. Также может использоваться для отслеживания изменений опций.

Конструктор | Методы

## Конструктор

```
Monitor(dataManager)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| dataManager * | — | Тип: IDataManager \| IOptionManager<br><br>Менеджер данных. |

* Обязательный параметр/опция.

**Пример:**

```
// Отслеживание изменений опции метки.
var placemark = new ymaps.Placemark([0, 0]),
    optionMonitor = new ymaps.Monitor(placemark.options);
optionMonitor.add("cursor", function (newValue) {
    alert("cursor: " + newValue);
});
myMap.geoObjects.add(placemark);
// Выведет строку "cursor: arrow".
myMap.options.set({
    geoObjectCursor: "arrow"
});
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(name, changeCallback[, context[, params]]) | **Monitor** | Включает наблюдение за определенным полем, либо группой полей данных. |
| **forceChange**() | **Monitor** | Инициирует проверку изменения значений наблюдаемых полей данных. |
| **get**(name) | Object | Возвращает текущее значение одного из наблюдаемых полей данных. |
| **remove**(name) | **Monitor** | Отключает наблюдение за определенным полем, либо группой полей данных. |
| **removeAll**() | **Monitor** | Отключает наблюдение для всех полей данных. |

## Описание методов

### add

```
{Monitor} add(name, changeCallback[, context[, params]])
```

Включает наблюдение за определенным полем, либо группой полей данных.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String \| String[]<br><br>Имя, либо массив имен полей данных, за которыми устанавливается наблюдение. |
| changeCallback* | — | Тип: Function<br><br>Обработчик изменения поля данных, либо одного из полей данных группы. |
| context | — | Тип: Object<br><br>Контекст исполнения обработчика изменений данных, а также опциональных обработчиков. |
| params | — | Тип: Object<br><br>Опциональные параметры. |
| params.compareCallback | — | Тип: Function<br><br>Обработчик сравнивающий старое и новое значения поля данных. Принимает два аргумента: старое значение, новое значение. Менее приоритетен по отношению к обработчикам заданным с помощью параметра compareCallbacks. |
| params.compareCallbacks | — | Тип: Object<br><br>Хэш вида {имя поля данных: ссылка на обработчик}. Данный параметр позволяет задать индивидуальные обработчики сравнения значений для различных полей данных в группе. |
| params.defaultValue | — | Тип: Object<br><br>Значение по умолчанию, используемое в том случае, если поле данных не определено. |
| params.defaultValues | — | Тип: Object<br><br>Хэш вида {имя поля данных: значение по умолчанию}. Данный параметр позволяет задать индивидуальные значения по умолчанию для различных полей данных в группе. |
| params.resolveCallback | — | Тип: Function |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Обработчик разрешающий значение поля данных. Принимает два аргумента: имя поля данных, ссылку на менеджер данных. Менее приоритетен по отношению к обработчикам заданным с помощью параметра resolveCallbacks. |
| params.resolveCallbacks | — | Тип: Object<br><br>Хэш вида {имя поля данных: ссылка на обработчик}. Данный параметр позволяет задать индивидуальные обработчики разрешения значений для различных полей данных в группе. |

* Обязательный параметр/опция.

## forceChange

```
{Monitor} forceChange()
```

Инициирует проверку изменения значений наблюдаемых полей данных.

**Возвращает** ссылку на себя.

## get

```
{Object} get(name)
```

**Возвращает** текущее значение одного из наблюдаемых полей данных.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя поля данных. |

* Обязательный параметр/опция.

## remove

```
{Monitor} remove(name)
```

Отключает наблюдение за определенным полем, либо группой полей данных.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String \| String[]<br><br>Имя, либо массив имен полей данных, для которых отключается наблюдение. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## removeAll

```
{Monitor} removeAll()
```

Отключает наблюдение для всех полей данных.

**Возвращает** ссылку на себя.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter

## multiRouter.driving

### multiRouter.driving.Path

**Примечание:**
Конструктор класса multiRouter.driving.Path скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение пути автомобильного мультимаршрута. Единичный маршрут может содержать несколько путей, каждый путь соединяет две путевые точки.

Поля | События | Методы

Создает отображение пути автомобильного мультимаршрута.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.driving.PathModel** | Модель данных пути мультимаршрута. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **properties** | **data.Manager** | Данные пути мультимаршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;<br>• type: String — идентификатор типа маршрута, принимающий значение "driving" для автомобильных маршрутов;<br>• distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда пути в секундах;<br>• durationInTraffic: Object — объект с полями text и value, описывающий время проезда пути в секундах с учетом пробок;<br>• coordinates: Number[][] — координаты всех точек пути;<br>• encodedCoordinates: String — строка закодированных в base64 координат всех точек пути. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br><br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **update** | Обновление отображения пути. Экземпляр класса **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br><br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br><br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br><br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br><br>Унаследован от IChildOnMap. |
| getSegments() | GeoObjectCollection | Возвращает дочернюю коллекцию сегментов, из которых состоит путь. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br><br>Унаследован от IChildOnMap. |

## Описание полей

### model

```
{multiRouter.driving.PathModel} model
```

Модель данных пути мультимаршрута.

### properties

```
{data.Manager} properties
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Данные пути мультимаршрута. Доступны следующие поля:

- index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;
- type: String — идентификатор типа маршрута, принимающий значение "driving" для автомобильных маршрутов;
- distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда пути в секундах;
- durationInTraffic: Object — объект с полями text и value, описывающий время проезда пути в секундах с учетом пробок;
- coordinates: Number[][] — координаты всех точек пути;
- encodedCoordinates: String — строка закодированных в base64 координат всех точек пути.

### Описание событий

#### update

Обновление отображения пути. Экземпляр класса **Event**.

### Описание методов

#### getSegments

```
{GeoObjectCollection} getSegments()
```

**Возвращает** дочернюю коллекцию сегментов, из которых состоит путь.

## multiRouter.driving.PathModel

**Примечание:**
Конструктор класса multiRouter.driving.PathModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IEventEmitter**.

Модель данных пути автомобильного маршрута. Единичный маршрут может содержать несколько путей, каждый путь соединяет две путевые точки.

Поля | События | Методы

Создает модель данных пути автомобильного маршрута.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| properties | data.Manager | Данные пути мультимаршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;<br>• type: String — идентификатор типа маршрута, принимающий значение "driving" для автомобильных маршрутов;<br>• distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда пути в секундах;<br>• durationInTraffic: Object — объект с полями text и value, описывающий время проезда пути в секундах с учетом пробок;<br>• coordinates: Number[][] — координаты всех точек пути;<br>• encodedCoordinates: String — строка закодированных в base64 координат всех точек пути. |
| route | multiRouter.driving.RouteModel | Ссылка на родительскую модель маршрута. |

## События

| Имя | Описание |
|---|---|
| update | Обновление модели новыми данными. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• segmentsChange: Boolean — признак изменений в наборе сегментов. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает модель. |
| getSegments() | multiRouter.driving.SegmentModel[] | Возвращает массив сегментов пути. |
| getType() | String | Возвращает идентификатор типа пути маршрута. Для автомобильных маршрутов возвращается строка "driving". |
| update(pathJson) | | Обновляет состояние модели. |

## Описание полей

### properties

```
{data.Manager} properties
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Данные пути мультимаршрута. Доступны следующие поля:

- index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;
- type: String — идентификатор типа маршрута, принимающий значение "driving" для автомобильных маршрутов;
- distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда пути в секундах;
- durationInTraffic: Object — объект с полями text и value, описывающий время проезда пути в секундах с учетом пробок;
- coordinates: Number[][] — координаты всех точек пути;
- encodedCoordinates: String — строка закодированных в base64 координат всех точек пути.

### route

```
{multiRouter.driving.RouteModel} route
```

Ссылка на родительскую модель маршрута.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- segmentsChange: Boolean — признак изменений в наборе сегментов.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getSegments

```
{multiRouter.driving.SegmentModel[]} getSegments()
```

**Возвращает** массив сегментов пути.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа пути маршрута. Для автомобильных маршрутов возвращается строка "driving".

### update

```
update(pathJson)
```

Обновляет состояние модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pathJson* | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | JSON-данные. |

* Обязательный параметр/опция.

# multiRouter.driving.Route

**Примечание:**
Конструктор класса multiRouter.driving.Route скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение единичного автомобильного маршрута. Мультимаршрут может состоять из нескольких единичных маршрутов.

Поля | События | Методы

Создает отображение единичного автомобильного маршрута.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.driving.RouteModel** | Модель данных единичного маршрута. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **properties** | **data.Manager** | Данные маршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер маршрута в мультимаршруте;<br>• type: String — идентификатор типа маршрута, принимающий значение "driving" для автомобильных маршрутов;<br>• blocked: Boolean — признак того, что маршрут содержит перекрытые участки;<br>• distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда маршрута в секундах;<br>• durationInTraffic: Object — объект с полями text и value, описывающий время проезда маршрута в секундах с учетом пробок;<br>• boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего данный маршрут. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br><br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **update** | Обновление отображения маршрута. Экземпляр класса **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br><br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br><br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br><br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br><br>Унаследован от IChildOnMap. |
| getPaths() | GeoObjectCollection | Возвращает дочернюю коллекцию путей, из которых состоит маршрут. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br><br>Унаследован от IChildOnMap. |

## Описание полей

### model

```
{multiRouter.driving.RouteModel} model
```

Модель данных единичного маршрута.

### properties

```
{data.Manager} properties
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Данные маршрута. Доступны следующие поля:

- index: Integer — порядковый номер маршрута в мультимаршруте;
- type: String — идентификатор типа маршрута, принимающий значение "driving" для автомобильных маршрутов;
- blocked: Boolean — признак того, что маршрут содержит перекрытые участки;
- distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда маршрута в секундах;
- durationInTraffic: Object — объект с полями text и value, описывающий время проезда маршрута в секундах с учетом пробок;
- boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего данный маршрут.

## Описание событий

### update

Обновление отображения маршрута. Экземпляр класса **Event**.

## Описание методов

### getPaths

```
{GeoObjectCollection} getPaths()
```

**Возвращает** дочернюю коллекцию путей, из которых состоит маршрут.

# multiRouter.driving.RouteModel

**Примечание:**
Конструктор класса multiRouter.driving.RouteModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IEventEmitter**.

Модель данных единичного автомобильного маршрута. Мультимаршрут может состоять из нескольких единичных маршрутов.

Поля | События | Методы

Создает модель данных единичного автомобильного маршрута.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **multiRoute** | **multiRouter.MultiRouteModel** | Ссылка на родительскую модель мультимаршрута. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| properties | data.Manager | Данные маршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер маршрута в мультимаршруте;<br>• type: String — идентификатор типа маршрута, принимающий значение "driving" для автомобильных маршрутов;<br>• blocked: Boolean — признак того, что маршрут содержит перекрытые участки;<br>• distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда маршрута в секундах;<br>• durationInTraffic: Object — объект с полями text и value, описывающий время проезда маршрута в секундах с учетом пробок;<br>• boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего данный маршрут. |

## События

| Имя | Описание |
|---|---|
| update | Обновление модели новыми данными. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• pathsChange: Boolean — признак изменений в наборе путей. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает модель. |
| getPaths() | multiRouter.driving.PathModel[] | Возвращает массив путей маршрута. |
| getType() | String | Возвращает идентификатор типа маршрута. Для автомобильных маршрутов возвращается строка "driving". |
| update(routeJson) | | Обновляет состояние модели. |

# Описание полей

## multiRoute

```
{multiRouter.MultiRouteModel} multiRoute
```

Ссылка на родительскую модель мультимаршрута.

## properties

```
{data.Manager} properties
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Данные маршрута. Доступны следующие поля:

- index: Integer — порядковый номер маршрута в мультимаршруте;
- type: String — идентификатор типа маршрута, принимающий значение "driving" для автомобильных маршрутов;
- blocked: Boolean — признак того, что маршрут содержит перекрытые участки;
- distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда маршрута в секундах;
- durationInTraffic: Object — объект с полями text и value, описывающий время проезда маршрута в секундах с учетом пробок;
- boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего данный маршрут.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pathsChange: Boolean — признак изменений в наборе путей.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getPaths

```
{multiRouter.driving.PathModel[]} getPaths()
```

**Возвращает** массив путей маршрута.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа маршрута. Для автомобильных маршрутов возвращается строка "driving".

### update

```
update(routeJson)
```

Обновляет состояние модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| routeJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter.driving.Segment

**Примечание:**
Конструктор класса multiRouter.driving.Segment скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение сегмента автомобильного маршрута. Сегментом автомобильного маршрута называется часть пути от одного манёвра до другого.

Поля | События | Методы

Создает отображение сегмента автомобильного маршрута.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.driving.SegmentModel** | Модель данных сегмента. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **properties** | **data.Manager** | Данные сегмента. Доступны следующие поля:<br>• index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;<br>• type: String — идентификатор типа сегмента, принимающий значение "driving" для автомобильных сегментов;<br>• street: String — текстовое описание улицы, по которой проходит сегмент;<br>• action: Object — объект с полями text и value, описывающий завершающий маневр сегмента;<br>• distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;<br>• durationInTraffic: Object — объект с полями text и value, описывающий время проезда сегмента в секундах с учетом пробок;<br>• text: String — текстовое описание сегмента;<br>• viaPoints: Integer[] — индексы транзитных точек, лежащих на данном сегменте;<br>• lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br><br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **update** | Обновление отображения сегмента. Экземпляр класса **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br><br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br><br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br><br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br><br>Унаследован от IChildOnMap. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br><br>Унаследован от IChildOnMap. |

## Описание полей

### model

```
{multiRouter.driving.SegmentModel} model
```

Модель данных сегмента.

### properties

```
{data.Manager} properties
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Данные сегмента. Доступны следующие поля:

- index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;
- type: String — идентификатор типа сегмента, принимающий значение "driving" для автомобильных сегментов;
- street: String — текстовое описание улицы, по которой проходит сегмент;
- action: Object — объект с полями text и value, описывающий завершающий маневр сегмента;
- distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;
- durationInTraffic: Object — объект с полями text и value, описывающий время проезда сегмента в секундах с учетом пробок;
- text: String — текстовое описание сегмента;
- viaPoints: Integer[] — индексы транзитных точек, лежащих на данном сегменте;
- lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута.

### Описание событий

#### update

Обновление отображения сегмента. Экземпляр класса **Event**.

## multiRouter.driving.SegmentModel

---

**Примечание:**
Конструктор класса multiRouter.driving.SegmentModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет **IEventEmitter**.

Модель данных сегмента пути автомобильного маршрута. Сегментом автомобильного маршрута называется часть пути от одного манёвра до другого.

Поля | События | Методы

Создает модель данных сегмента пути автомобильного маршрута.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **geometry** | **geometry.base.LineString** | Геометрия сегмента. |
| **path** | **multiRouter.driving.PathModel** | Ссылка на родительскую модель пути. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| properties | data.Manager | Данные сегмента. Доступны следующие поля:<br>• index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;<br>• type: String — идентификатор типа сегмента, принимающий значение "driving" для автомобильных сегментов;<br>• street: String — текстовое описание улицы, по которой проходит сегмент;<br>• action: Object — объект с полями text и value, описывающий завершающий маневр сегмента;<br>• distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;<br>• durationInTraffic: Object — объект с полями text и value, описывающий время проезда сегмента в секундах с учетом пробок;<br>• text: String — текстовое описание сегмента;<br>• viaPoints: Integer[] — индексы транзитных точек, лежащих на данном сегменте;<br>• lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута. |

## События

| Имя | Описание |
|---|---|
| update | Обновление модели новыми данными. Экземпляр класса Event. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает модель. |
| getType() | String | Возвращает идентификатор типа сегмента. Для сегментов автомобильных маршрутов возвращается строка "driving". |
| getViaPoints() | multiRouter.ViaPointModel[] | Возвращает массив транзитных точек сегмента. |
| update(segmentJson) | | Обновляет состояние модели. |

## Описание полей

### geometry

```
{geometry.base.LineString} geometry
```

Геометрия сегмента.

### path

```
{multiRouter.driving.PathModel} path
```

Ссылка на родительскую модель пути.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### properties

```
{data.Manager} properties
```

Данные сегмента. Доступны следующие поля:

- index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;
- type: String — идентификатор типа сегмента, принимающий значение "driving" для автомобильных сегментов;
- street: String — текстовое описание улицы, по которой проходит сегмент;
- action: Object — объект с полями text и value, описывающий завершающий маневр сегмента;
- distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;
- durationInTraffic: Object — объект с полями text и value, описывающий время проезда сегмента в секундах с учетом пробок;
- text: String — текстовое описание сегмента;
- viaPoints: Integer[] — индексы транзитных точек, лежащих на данном сегменте;
- lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа сегмента. Для сегментов автомобильных маршрутов возвращается строка "driving".

### getViaPoints

```
{multiRouter.ViaPointModel[]} getViaPoints()
```

**Возвращает** массив транзитных точек сегмента.

### update

```
update(segmentJson)
```

Обновляет состояние модели.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| segmentJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter.Editor

Расширяет **ICustomizable**, **IEventEmitter**.

Редактор мультимаршрута.

Конструктор | Поля | События | Методы

## Конструктор

```
multiRouter.Editor(multiRoute[, state[, options]])
```

Создает редактор мультимаршрута.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| multiRoute * | — | Тип: **multiRouter.MultiRoute**<br><br>Редактируемый мультимаршрут. |
| state | — | Тип: Object<br><br>Объект, описывающий начальное состояние редактора. Список доступных полей смотрите в описании **multiRouter.Editor.state**. |
| options | — | Тип: Object<br><br>Опции. |
| options.drawOver | true | Тип: Object<br><br>В режиме добавления новых путевых точек разрешает ставить точки поверх объектов карты. |
| options.midPointsType | "way" | Тип: String<br><br>Определяет тип добавляемых точек при перетаскивании маркера, появляющегося при наведении курсора мыши на активный маршрут. Может принимать одно из следующих строковых значений:<br>• "way" — добавлять путевые точки;<br>• "via" — добавлять транзитные точки.<br><br>Также смотрите описание поля addMidPoints менеджера состояния **multiRouter.Editor.state**. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **state** | **data.Manager** | Менеджер состояния редактора мультимаршрута.<br>Доступные поля:<br>• addWayPoints: Boolean — разрешает добавление новых путевых точек при клике на карту. Значение по умолчанию: false.<br>• dragWayPoints: Boolean — разрешает перетаскивание уже существующих путевых точек. Значение по умолчанию: true.<br>• removeWayPoints: Boolean — разрешает удаление путевых точек при двойном клике по ним. Значение по умолчанию: false.<br>• dragViaPoints: Boolean — разрешает перетаскивание уже существующих транзитных точек. Значение по умолчанию: true.<br>• removeViaPoints: Boolean — разрешает удаление транзитных точек при двойном клике по ним. Значение по умолчанию: true.<br>• addMidPoints: Boolean — разрешает добавление промежуточных транзитных или путевых точек посредством перетаскивания маркера, появляющегося при наведении курсора мыши на активный маршрут. Тип добавляемых точек задается опцией midPointsType. Значение по умолчанию: true. |

## События

| Имя | Описание |
|---|---|
| **beforemidpointadd** | Событие, предшествующее событию midpointadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• coords: Number[] — координаты добавления промежуточной точки;<br>• pointType: String — идентификатор типа добавляемой промежуточной точки;<br>• insertIndex: Integer — индекс вставки промежуточной точки во множестве опорных точек мультимаршрута.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPointType — метод, позволяющий задать тип добавляемой точки. В качестве аргумента принимает строковый идентификатор типа (смотрите описание опции midPointsType).<br>• setInsertIndex — метод, позволяющий скорректировать индекс вставки промежуточной точки, который будет фактически применен. В качестве аргумента принимает новый индекс.<br>Если у данного события вызвать метод Event.preventDefault, то последующее добавление промежуточной точки маршрута, как и событие midpointadd, будет отменено. |
| **beforemidpointdrag** | Событие, предшествующее событию midpointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие midpointdrag будет отменено. |
| **beforemidpointpinshow** | Событие, предшествующее событию midpointpinshow. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pin: **Placemark** — ссылка на объект маркера;<br>• globalPixels: Number[] — глобальные пиксельные координаты маркера;<br>• segment: **multiRouter.driving.Segment** — ссылка на сегмент маршрута, над которым должен появится маркер.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие midpointpinshow будет отменено, а маркер будет скрыт. |
| **beforeviapointdrag** | Событие, предшествующее событию viapointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие viapointdrag будет отменено. |
| **beforeviapointdragstart** | Событие, предшествующее событию viapointdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие viapointdragstart, будет отменено. |
| **beforeviapointremove** | Событие, предшествующее событию viapointremove. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект удаляемой транзитной точки.<br>Если у данного события вызвать метод Event.preventDefault, то удаление транзитной точки, как и последующее событие viapointremove, будет отменено. |
| **beforewaypointadd** | Событие, предшествующее событию waypointadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• coords: Number[] — координаты добавленной путевой точки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Имена методов, доступных через метод Event.callMethod:<br>• setCoords — метод, позволяющий скорректировать координаты добавляемой путевой точки. В качестве аргумента принимает новые координаты в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то добавление путевой точки, как и последующее событие waypointadd, будет отменено. |
| **beforewaypointdrag** | Событие, предшествующее событию waypointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие waypointdrag будет отменено. |
| **beforewaypointdragstart** | Событие, предшествующее событию waypointdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие waypointdragstart, будет отменено. |
| **beforewaypointremove** | Событие, предшествующее событию waypointremove. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект удаляемой путевой точки.<br>Если у данного события вызвать метод Event.preventDefault, то удаление путевой точки, как и последующее событие waypointremove, будет отменено. |
| **midpointadd** | Добавление промежуточной точки маршрута. Тип точки определяется значением опции midPointsType. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• coords: Number[] — координаты добавления промежуточной точки;<br>• pointType: String — идентификатор типа добавляемой промежуточной точки;<br>• insertIndex: Integer — индекс вставки промежуточной точки во множестве опорных точек мультимаршрута. |
| **midpointdrag** | Перетаскивание добавленной промежуточной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге. |
| **midpointdragend** | Завершение перетаскивания добавленной промежуточной точки. Экземпляр класса **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **midpointpinshow** | Появление перетаскиваемого маркера при наведении на активный маршрут. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pin: **Placemark** — ссылка на объект маркера;<br>• globalPixels: Number[] — глобальные пиксельные координаты маркера;<br>• segment: **multiRouter.driving.Segment** — ссылка на сегмент маршрута, над которым появился маркер. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **viapointdrag** | Перетаскивание транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки. |
| **viapointdragend** | Завершение перетаскивания транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки. |
| **viapointdragstart** | Начало перетаскивания транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки. |
| **viapointremove** | Удаление транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект удаленной транзитной точки. |
| **waypointadd** | Добавление путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• coords: Number[] — координаты добавленной путевой точки. |
| **waypointdrag** | Перетаскивание путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки. |
| **waypointdragend** | Завершение перетаскивания путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки. |
| **waypointdragstart** | Начало перетаскивания путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| waypointremove | Удаление путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект удаленной путевой точки. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает редактор мультимаршрута. |
| getMultiRoute() | multiRouter.MultiRoute | Возвращает ссылку на редактируемый мультимаршрут. |

## Описание полей

### state

```
{data.Manager} state
```

Менеджер состояния редактора мультимаршрута.

Доступные поля:

- addWayPoints: Boolean — разрешает добавление новых путевых точек при клике на карту. Значение по умолчанию: false.
- dragWayPoints: Boolean — разрешает перетаскивание уже существующих путевых точек. Значение по умолчанию: true.
- removeWayPoints: Boolean — разрешает удаление путевых точек при двойном клике по ним. Значение по умолчанию: false.
- dragViaPoints: Boolean — разрешает перетаскивание уже существующих транзитных точек. Значение по умолчанию: true.
- removeViaPoints: Boolean — разрешает удаление транзитных точек при двойном клике по ним. Значение по умолчанию: true.
- addMidPoints: Boolean — разрешает добавление промежуточных транзитных или путевых точек посредством перетаскивания маркера, появляющегося при наведении курсора мыши на активный маршрут. Тип добавляемых точек задается опцией midPointsType. Значение по умолчанию: true.

## Описание событий

### beforemidpointadd

Событие, предшествующее событию midpointadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- coords: Number[] — координаты добавления промежуточной точки;
- pointType: String — идентификатор типа добавляемой промежуточной точки;
- insertIndex: Integer — индекс вставки промежуточной точки во множестве опорных точек мультимаршрута.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Имена методов, доступных через метод Event.callMethod:

- setPointType — метод, позволяющий задать тип добавляемой точки. В качестве аргумента принимает строковый идентификатор типа (смотрите описание опции midPointsType).
- setInsertIndex — метод, позволяющий скорректировать индекс вставки промежуточной точки, который будет фактически применен. В качестве аргумента принимает новый индекс.

Если у данного события вызвать метод Event.preventDefault, то последующее добавление промежуточной точки маршрута, как и событие midpointadd, будет отменено.

## beforemidpointdrag

Событие, предшествующее событию midpointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге.

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие midpointdrag будет отменено.

## beforemidpointpinshow

Событие, предшествующее событию midpointpinshow. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pin: **Placemark** — ссылка на объект маркера;
- globalPixels: Number[] — глобальные пиксельные координаты маркера;
- segment: **multiRouter.driving.Segment** — ссылка на сегмент маршрута, над которым должен появится маркер.

Если у данного события вызвать метод Event.preventDefault, то последующее событие midpointpinshow будет отменено, а маркер будет скрыт.

## beforeviapointdrag

Событие, предшествующее событию viapointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;
- viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие viapointdrag будет отменено.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## beforeviapointdragstart

Событие, предшествующее событию viapointdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;
- viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.

Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие viapointdragstart, будет отменено.

## beforeviapointremove

Событие, предшествующее событию viapointremove. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- viaPoint: **multiRouter.ViaPoint** — ссылка на объект удаляемой транзитной точки.

Если у данного события вызвать метод Event.preventDefault, то удаление транзитной точки, как и последующее событие viapointremove, будет отменено.

## beforewaypointadd

Событие, предшествующее событию waypointadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- coords: Number[] — координаты добавленной путевой точки.

Имена методов, доступных через метод Event.callMethod:

- setCoords — метод, позволяющий скорректировать координаты добавляемой путевой точки. В качестве аргумента принимает новые координаты в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то добавление путевой точки, как и последующее событие waypointadd, будет отменено.

## beforewaypointdrag

Событие, предшествующее событию waypointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;
- wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие waypointdrag будет отменено.

## beforewaypointdragstart

Событие, предшествующее событию waypointdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;
- wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие waypointdragstart, будет отменено.

## beforewaypointremove

Событие, предшествующее событию waypointremove. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- wayPoint: **multiRouter.WayPoint** — ссылка на объект удаляемой путевой точки.

Если у данного события вызвать метод Event.preventDefault, то удаление путевой точки, как и последующее событие waypointremove, будет отменено.

## midpointadd

Добавление промежуточной точки маршрута. Тип точки определяется значением опции midPointsType. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- coords: Number[] — координаты добавления промежуточной точки;
- pointType: String — идентификатор типа добавляемой промежуточной точки;
- insertIndex: Integer — индекс вставки промежуточной точки во множестве опорных точек мультимаршрута.

## midpointdrag

Перетаскивание добавленной промежуточной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге.

## midpointdragend

Завершение перетаскивания добавленной промежуточной точки. Экземпляр класса **Event**.

## midpointpinshow

Появление перетаскиваемого маркера при наведении на активный маршрут. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pin: **Placemark** — ссылка на объект маркера;
- globalPixels: Number[] — глобальные пиксельные координаты маркера;
- segment: **multiRouter.driving.Segment** — ссылка на сегмент маршрута, над которым появился маркер.

## viapointdrag

Перетаскивание транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;
- viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.

## viapointdragend

Завершение перетаскивания транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### viapointdragstart

Начало перетаскивания транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;
- viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.

### viapointremove

Удаление транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- viaPoint: **multiRouter.ViaPoint** — ссылка на объект удаленной транзитной точки.

### waypointadd

Добавление путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- coords: Number[] — координаты добавленной путевой точки.

### waypointdrag

Перетаскивание путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;
- wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.

### waypointdragend

Завершение перетаскивания путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.

### waypointdragstart

Начало перетаскивания путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;
- wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.

### waypointremove

Удаление путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- wayPoint: **multiRouter.WayPoint** — ссылка на объект удаленной путевой точки.

## Описание методов

### destroy

```
destroy()
```

Уничтожает редактор мультимаршрута.

### getMultiRoute

```
{multiRouter.MultiRoute} getMultiRoute()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** ссылку на редактируемый мультимаршрут.

# multiRouter.EditorAddon

**Примечание:**
Конструктор класса multiRouter.EditorAddon скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **ICustomizable**, **IEventEmitter**.

Аддон редактора мультимаршрута.

Поля | События | Методы

Создает аддон редактора мультимаршрута.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **state** | **data.Manager** | Менеджер состояния редактора мультимаршрута. |

## События

| Имя | Описание |
|---|---|
| **beforemidpointadd** | Событие, предшествующее событию midpointadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• coords: Number[] — координаты добавления промежуточной точки;<br>• pointType: String — идентификатор типа добавляемой промежуточной точки;<br>• insertIndex: Integer — индекс вставки промежуточной точки во множестве опорных точек мультимаршрута.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPointType — метод, позволяющий задать тип добавляемой точки. В качестве аргумента принимает строковый идентификатор типа (смотрите описание опции midPointsType).<br>• setInsertIndex — метод, позволяющий скорректировать индекс вставки промежуточной точки, который будет фактически применен. В качестве аргумента принимает новый индекс.<br>Если у данного события вызвать метод Event.preventDefault, то последующее добавление промежуточной точки маршрута, как и событие midpointadd, будет отменено. |
| **beforemidpointdrag** | Событие, предшествующее событию midpointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие midpointdrag будет отменено. |
| **beforemidpointpinshow** | Событие, предшествующее событию midpointpinshow. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pin: **Placemark** — ссылка на объект маркера;<br>• globalPixels: Number[] — глобальные пиксельные координаты маркера;<br>• segment: **multiRouter.driving.Segment** — ссылка на сегмент маршрута, над которым должен появится маркер.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие midpointpinshow будет отменено, а маркер будет скрыт. |
| **beforeviapointdrag** | Событие, предшествующее событию viapointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие viapointdrag будет отменено. |
| **beforeviapointdragstart** | Событие, предшествующее событию viapointdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие viapointdragstart, будет отменено. |
| **beforeviapointremove** | Событие, предшествующее событию viapointremove. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект удаляемой транзитной точки.<br>Если у данного события вызвать метод Event.preventDefault, то удаление транзитной точки, как и последующее событие viapointremove, будет отменено. |
| **beforewaypointadd** | Событие, предшествующее событию waypointadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• coords: Number[] — координаты добавленной путевой точки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Имена методов, доступных через метод Event.callMethod:<br>• setCoords — метод, позволяющий скорректировать координаты добавляемой путевой точки. В качестве аргумента принимает новые координаты в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то добавление путевой точки, как и последующее событие waypointadd, будет отменено. |
| **beforewaypointdrag** | Событие, предшествующее событию waypointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие waypointdrag будет отменено. |
| **beforewaypointdragstart** | Событие, предшествующее событию waypointdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие waypointdragstart, будет отменено. |
| **beforewaypointremove** | Событие, предшествующее событию waypointremove. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект удаляемой путевой точки.<br>Если у данного события вызвать метод Event.preventDefault, то удаление путевой точки, как и последующее событие waypointremove, будет отменено. |
| **midpointadd** | Добавление промежуточной точки маршрута. Тип точки определяется значением опции midPointsType. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• coords: Number[] — координаты добавления промежуточной точки;<br>• pointType: String — идентификатор типа добавляемой промежуточной точки;<br>• insertIndex: Integer — индекс вставки промежуточной точки во множестве опорных точек мультимаршрута. |
| **midpointdrag** | Перетаскивание добавленной промежуточной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге. |
| **midpointdragend** | Завершение перетаскивания добавленной промежуточной точки. Экземпляр класса **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **midpointpinshow** | Появление перетаскиваемого маркера при наведении на активный маршрут. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pin: **Placemark** — ссылка на объект маркера;<br>• globalPixels: Number[] — глобальные пиксельные координаты маркера;<br>• segment: **multiRouter.driving.Segment** — ссылка на сегмент маршрута, над которым появился маркер. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **start** | Включение редактора. Экземпляр класса **Event**. |
| **stop** | Отключение редактора. Экземпляр класса **Event**. |
| **viapointdrag** | Перетаскивание транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки. |
| **viapointdragend** | Завершение перетаскивания транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки. |
| **viapointdragstart** | Начало перетаскивания транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки. |
| **viapointremove** | Удаление транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• viaPoint: **multiRouter.ViaPoint** — ссылка на объект удаленной транзитной точки. |
| **waypointadd** | Добавление путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• coords: Number[] — координаты добавленной путевой точки. |
| **waypointdrag** | Перетаскивание путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки. |
| **waypointdragend** | Завершение перетаскивания путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **waypointdragstart** | Начало перетаскивания путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки. |
| **waypointremove** | Удаление путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• wayPoint: **multiRouter.WayPoint** — ссылка на объект удаленной путевой точки. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **isActive**() | Boolean | Возвращает признак того, что редактор в данный момент включен. |
| **start**(state) | | Включает редактор. |
| **stop**() | | Отключает редактор. |

## Описание полей

### state

```
{data.Manager} state
```

Менеджер состояния редактора мультимаршрута.

## Описание событий

### beforemidpointadd

Событие, предшествующее событию midpointadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- coords: Number[] — координаты добавления промежуточной точки;
- pointType: String — идентификатор типа добавляемой промежуточной точки;
- insertIndex: Integer — индекс вставки промежуточной точки во множестве опорных точек мультимаршрута.

Имена методов, доступных через метод Event.callMethod:

- setPointType — метод, позволяющий задать тип добавляемой точки. В качестве аргумента принимает строковый идентификатор типа (смотрите описание опции midPointsType).
- setInsertIndex — метод, позволяющий скорректировать индекс вставки промежуточной точки, который будет фактически применен. В качестве аргумента принимает новый индекс.

Если у данного события вызвать метод Event.preventDefault, то последующее добавление промежуточной точки маршрута, как и событие midpointadd, будет отменено.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## beforemidpointdrag

Событие, предшествующее событию midpointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге.

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие midpointdrag будет отменено.

## beforemidpointpinshow

Событие, предшествующее событию midpointpinshow. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pin: **Placemark** — ссылка на объект маркера;
- globalPixels: Number[] — глобальные пиксельные координаты маркера;
- segment: **multiRouter.driving.Segment** — ссылка на сегмент маршрута, над которым должен появится маркер.

Если у данного события вызвать метод Event.preventDefault, то последующее событие midpointpinshow будет отменено, а маркер будет скрыт.

## beforeviapointdrag

Событие, предшествующее событию viapointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;
- viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие viapointdrag будет отменено.

## beforeviapointdragstart

Событие, предшествующее событию viapointdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;
- viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.

Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие viapointdragstart, будет отменено.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## beforeviapointremove

Событие, предшествующее событию viapointremove. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- viaPoint: **multiRouter.ViaPoint** — ссылка на объект удаляемой транзитной точки.

Если у данного события вызвать метод Event.preventDefault, то удаление транзитной точки, как и последующее событие viapointremove, будет отменено.

## beforewaypointadd

Событие, предшествующее событию waypointadd. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- coords: Number[] — координаты добавленной путевой точки.

Имена методов, доступных через метод Event.callMethod:

- setCoords — метод, позволяющий скорректировать координаты добавляемой путевой точки. В качестве аргумента принимает новые координаты в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то добавление путевой точки, как и последующее событие waypointadd, будет отменено.

## beforewaypointdrag

Событие, предшествующее событию waypointdrag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;
- wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.

Имена методов, доступных через метод Event.callMethod:

- setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.

Если у данного события вызвать метод Event.preventDefault, то последующее событие waypointdrag будет отменено.

## beforewaypointdragstart

Событие, предшествующее событию waypointdragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;
- wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.

Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие waypointdragstart, будет отменено.

## beforewaypointremove

Событие, предшествующее событию waypointremove. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- wayPoint: **multiRouter.WayPoint** — ссылка на объект удаляемой путевой точки.

Если у данного события вызвать метод Event.preventDefault, то удаление путевой точки, как и последующее событие waypointremove, будет отменено.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## midpointadd

Добавление промежуточной точки маршрута. Тип точки определяется значением опции midPointsType. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- coords: Number[] — координаты добавления промежуточной точки;
- pointType: String — идентификатор типа добавляемой промежуточной точки;
- insertIndex: Integer — индекс вставки промежуточной точки во множестве опорных точек мультимаршрута.

## midpointdrag

Перетаскивание добавленной промежуточной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге.

## midpointdragend

Завершение перетаскивания добавленной промежуточной точки. Экземпляр класса **Event**.

## midpointpinshow

Появление перетаскиваемого маркера при наведении на активный маршрут. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pin: **Placemark** — ссылка на объект маркера;
- globalPixels: Number[] — глобальные пиксельные координаты маркера;
- segment: **multiRouter.driving.Segment** — ссылка на сегмент маршрута, над которым появился маркер.

## start

Включение редактора. Экземпляр класса **Event**.

## stop

Отключение редактора. Экземпляр класса **Event**.

## viapointdrag

Перетаскивание транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;
- viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.

## viapointdragend

Завершение перетаскивания транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### viapointdragstart

Начало перетаскивания транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;
- viaPoint: **multiRouter.ViaPoint** — ссылка на объект перетаскиваемой транзитной точки.

### viapointremove

Удаление транзитной точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- viaPoint: **multiRouter.ViaPoint** — ссылка на объект удаленной транзитной точки.

### waypointadd

Добавление путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- coords: Number[] — координаты добавленной путевой точки.

### waypointdrag

Перетаскивание путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelOffset: Number[] — пиксельный сдвиг по осям на данном шаге;
- wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.

### waypointdragend

Завершение перетаскивания путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.

### waypointdragstart

Начало перетаскивания путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- domEvent: **DomEvent** — исходное DOM-событие, если оно имеется;
- wayPoint: **multiRouter.WayPoint** — ссылка на объект перетаскиваемой путевой точки.

### waypointremove

Удаление путевой точки. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- wayPoint: **multiRouter.WayPoint** — ссылка на объект удаленной путевой точки.

## Описание методов

### isActive

```
{Boolean} isActive()
```

**Возвращает** признак того, что редактор в данный момент включен.

### start

```
start(state)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Включает редактор.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| state* | — | Тип: Object<br><br>Начальное состояние редактора. |

* Обязательный параметр/опция.

### stop

```
stop()
```

Отключает редактор.

# multiRouter.masstransit

## multiRouter.masstransit.Path

**Примечание:**
Конструктор класса multiRouter.masstransit.Path скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение пути маршрута общественного транспорта. Единичный маршрут может содержать несколько путей, каждый путь соединяет две путевые точки.

Поля | События | Методы

Создает отображение пути маршрута общественного транспорта.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.masstransit.PathModel** | Модель данных пути мультимаршрута. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| properties | data.Manager | Данные пути мультимаршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;<br>• type: String — идентификатор типа маршрута, принимающий значение "masstransit" для маршрутов общественного транспорта;<br>• distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда пути в секундах;<br>• coordinates: Number[][] — координаты всех точек пути;<br>• encodedCoordinates: String — строка закодированных в base64 координат всех точек пути. |
| state | IDataManager | Состояние геообъекта.<br>Унаследовано от IGeoObject. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| geometrychange | Изменение геометрии геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие геометрии.<br>Унаследовано от IGeoObject. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| update | Обновление отображения пути. Экземпляр класса Event. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| getSegmentMarkers() | GeoObjectCollection | Возвращает дочернюю коллекцию маркеров сегментов. |
| getSegments() | GeoObjectCollection | Возвращает дочернюю коллекцию сегментов, из которых состоит путь. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание полей

### model

```
{multiRouter.masstransit.PathModel} model
```

Модель данных пути мультимаршрута.

### properties

```
{data.Manager} properties
```

Данные пути мультимаршрута. Доступны следующие поля:

- index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;
- type: String — идентификатор типа маршрута, принимающий значение "masstransit" для маршрутов общественного транспорта;
- distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда пути в секундах;
- coordinates: Number[][] — координаты всех точек пути;
- encodedCoordinates: String — строка закодированных в base64 координат всех точек пути.

## Описание событий

### update

Обновление отображения пути. Экземпляр класса Event.

## Описание методов

### getSegmentMarkers

```
{GeoObjectCollection} getSegmentMarkers()
```

**Возвращает** дочернюю коллекцию маркеров сегментов.

### getSegments

```
{GeoObjectCollection} getSegments()
```

**Возвращает** дочернюю коллекцию сегментов, из которых состоит путь.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter.masstransit.PathModel

**Примечание:**
Конструктор класса multiRouter.masstransit.PathModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IEventEmitter**.

Модель данных пути маршрута общественного транспорта. Единичный маршрут может содержать несколько путей, каждый путь соединяет две путевые точки.

Поля | События | Методы

Создает модель данных пути маршрута общественного транспорта.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **properties** | **data.Manager** | Данные пути мультимаршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;<br>• type: String — идентификатор типа маршрута, принимающий значение "masstransit" для маршрутов общественного транспорта;<br>• distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда пути в секундах;<br>• coordinates: Number[][] — координаты всех точек пути;<br>• encodedCoordinates: String — строка закодированных в base64 координат всех точек пути. |
| **route** | **multiRouter.masstransit.RouteModel** | Ссылка на родительскую модель маршрута. |

## События

| Имя | Описание |
|---|---|
| **update** | Обновление модели новыми данными. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• segmentsChange: Boolean — признак изменений в наборе сегментов. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает модель. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getSegments() | (multiRouter.masstransit.TransferSegmentModel \| multiRouter.masstransit.TransportSegmentModel \| multiRouter.masstransit.WalkSegmentModel)[] | Возвращает массив сегментов пути. |
| getType() | String | Возвращает идентификатор типа маршрута. Для маршрутов общественного транспорта возвращается строка "masstransit". |
| update(pathJson) | | Обновляет состояние модели. |

## Описание полей

### properties

```
{data.Manager} properties
```

Данные пути мультимаршрута. Доступны следующие поля:

- index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;
- type: String — идентификатор типа маршрута, принимающий значение "masstransit" для маршрутов общественного транспорта;
- distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда пути в секундах;
- coordinates: Number[][] — координаты всех точек пути;
- encodedCoordinates: String — строка закодированных в base64 координат всех точек пути.

### route

```
{multiRouter.masstransit.RouteModel} route
```

Ссылка на родительскую модель маршрута.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- segmentsChange: Boolean — признак изменений в наборе сегментов.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getSegments

```
{(multiRouter.masstransit.TransferSegmentModel |
multiRouter.masstransit.TransportSegmentModel |
multiRouter.masstransit.WalkSegmentModel)[]} getSegments()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** массив сегментов пути.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа маршрута. Для маршрутов общественного транспорта возвращается строка "masstransit".

### update

```
update(pathJson)
```

Обновляет состояние модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pathJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.

## multiRouter.masstransit.Route

**Примечание:**
Конструктор класса multiRouter.masstransit.Route скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение единичного маршрута общественного транспорта. Мультимаршрут может состоять из нескольких единичных маршрутов.

Поля | События | Методы

Создает отображение единичного маршрута общественного транспорта.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.masstransit.RouteModel** | Модель данных единичного маршрута. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| properties | data.Manager | Данные маршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер маршрута в мультимаршруте;<br>• type: String — идентификатор типа маршрута, принимающий значение "masstransit" для маршрутов общественного транспорта;<br>• distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда маршрута в секундах;<br>• boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего данный маршрут. |
| state | IDataManager | Состояние геообъекта.<br>Унаследовано от IGeoObject. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| geometrychange | Изменение геометрии геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие геометрии.<br>Унаследовано от IGeoObject. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **update** | Обновление отображения маршрута. Экземпляр класса **Event**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getPaths**() | **GeoObjectCollection** | Возвращает дочернюю коллекцию путей, из которых состоит маршрут. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание полей

### model

```
{multiRouter.masstransit.RouteModel} model
```

Модель данных единичного маршрута.

### properties

```
{data.Manager} properties
```

Данные маршрута. Доступны следующие поля:

- index: Integer — порядковый номер маршрута в мультимаршруте;
- type: String — идентификатор типа маршрута, принимающий значение "masstransit" для маршрутов общественного транспорта;
- distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда маршрута в секундах;
- boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего данный маршрут.

## Описание событий

### update

Обновление отображения маршрута. Экземпляр класса **Event**.

## Описание методов

### getPaths

```
{GeoObjectCollection} getPaths()
```

**Возвращает** дочернюю коллекцию путей, из которых состоит маршрут.

# multiRouter.masstransit.RouteModel

---

**Примечание:**
Конструктор класса multiRouter.masstransit.RouteModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет **IEventEmitter**.

Модель данных единичного маршрута общественного транспорта. Мультимаршрут может состоять из нескольких единичных маршрутов.

Поля | События | Методы

Создает модель данных единичного маршрута общественного транспорта.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| multiRoute | multiRouter.MultiRouteModel | Ссылка на родительскую модель мультимаршрута. |
| properties | data.Manager | Данные маршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер маршрута в мультимаршруте;<br>• type: String — идентификатор типа маршрута, принимающий значение "masstransit" для маршрутов общественного транспорта;<br>• distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда маршрута в секундах;<br>• boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего данный маршрут. |

## События

| Имя | Описание |
|---|---|
| update | Обновление модели новыми данными. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• pathsChange: Boolean — признак изменений в наборе путей. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает модель. |
| getPaths() | multiRouter.masstransit.PathModel[] | Возвращает массив путей маршрута. |
| getType() | String | Возвращает идентификатор типа маршрута. Для маршрутов общественного транспорта возвращается строка "masstransit". |
| update(routeJson) | | Обновляет состояние модели. |

# Описание полей

## multiRoute

```
{multiRouter.MultiRouteModel} multiRoute
```

Ссылка на родительскую модель мультимаршрута.

## properties

```
{data.Manager} properties
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Данные маршрута. Доступны следующие поля:

- index: Integer — порядковый номер маршрута в мультимаршруте;
- type: String — идентификатор типа маршрута, принимающий значение "masstransit" для маршрутов общественного транспорта;
- distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда маршрута в секундах;
- boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего данный маршрут.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pathsChange: Boolean — признак изменений в наборе путей.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getPaths

```
{multiRouter.masstransit.PathModel[]} getPaths()
```

**Возвращает** массив путей маршрута.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа маршрута. Для маршрутов общественного транспорта возвращается строка "masstransit".

### update

```
update(routeJson)
```

Обновляет состояние модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| routeJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter.masstransit.StopModel

**Примечание:**
Конструктор класса multiRouter.masstransit.StopModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IEventEmitter**.

Модель данных остановки транспортного сегмента маршрута общественного транспорта.

Поля | События | Методы

Создает модель данных остановки транспортного сегмента.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **geometry** | **geometry.base.Point** | Геометрия остановки. |
| **properties** | **data.Manager** | Данные остановки. Доступны следующие поля:<br>• index: Integer — порядковый номер остановки во множестве остановок соответствующего транспортного сегмента;<br>• id: String — идентификатор остановки;<br>• name: String — название остановки;<br>• lodIndex: Integer — порядковый номер точки остановки во множестве всех координат соответствующего ей пути маршрута. |
| **segment** | **multiRouter.masstransit.TransportSegmentModel** | Ссылка на родительскую модель транспортного сегмента. |

## События

| Имя | Описание |
|---|---|
| **update** | Обновление модели новыми данными. Экземпляр класса **Event**. |

## Методы

| Имя | Описание |
|---|---|
| **update**(stopJson) | Обновляет состояние модели. |

## Описание полей

### geometry

```
{geometry.base.Point} geometry
```

Геометрия остановки.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### properties

```
{data.Manager} properties
```

Данные остановки. Доступны следующие поля:

- index: Integer — порядковый номер остановки во множестве остановок соответствующего транспортного сегмента;
- id: String — идентификатор остановки;
- name: String — название остановки;
- lodIndex: Integer — порядковый номер точки остановки во множестве всех координат соответствующего ей пути маршрута.

### segment

```
{multiRouter.masstransit.TransportSegmentModel} segment
```

Ссылка на родительскую модель транспортного сегмента.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**.

## Описание методов

### update

```
update(stopJson)
```

Обновляет состояние модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| stopJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.

# multiRouter.masstransit.TransferSegment

**Примечание:**
Конструктор класса multiRouter.masstransit.TransferSegment скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение сегмента перехода маршрута общественного транспорта. Сегментом пути маршрута общественного транспорта называется часть пути от одной пересадки до другой.

Поля | События | Методы


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Создает отображение сегмента перехода маршрута общественного транспорта.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |
| geometry | IGeometry \| null | Геометрия геообъекта.<br>Унаследовано от IGeoObject. |
| model | multiRouter.masstransit.TransferSegmentModel | Модель данных сегмента. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |
| properties | data.Manager | Данные сегмента. Доступны следующие поля:<br>• index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;<br>• type: String — идентификатор типа сегмента, принимающий значение "transfer" для сегментов переходов;<br>• text: String — текстовое описание сегмента;<br>• distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;<br>• lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута. |
| state | IDataManager | Состояние геообъекта.<br>Унаследовано от IGeoObject. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| geometrychange | Изменение геометрии геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие геометрии. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| update | Обновление отображения сегмента. Экземпляр класса Event. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание полей

### model

```
{multiRouter.masstransit.TransferSegmentModel} model
```

Модель данных сегмента.

### properties

```
{data.Manager} properties
```

Данные сегмента. Доступны следующие поля:

- index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;
- type: String — идентификатор типа сегмента, принимающий значение "transfer" для сегментов переходов;
- text: String — текстовое описание сегмента;
- distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;
- lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута.

## Описание событий

### update

Обновление отображения сегмента. Экземпляр класса **Event**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter.masstransit.TransferSegmentModel

**Примечание:**
Конструктор класса multiRouter.masstransit.TransferSegmentModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IEventEmitter**.

Модель данных сегмента перехода пути маршрута общественного транспорта. Сегментом пути маршрута общественного транспорта называется часть пути от одной пересадки до другой.

Поля | События | Методы

Создает модель данных сегмента перехода пути маршрута общественного транспорта.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **geometry** | **geometry.base.LineString** | Геометрия сегмента. |
| **path** | **multiRouter.masstransit.PathModel** | Ссылка на родительскую модель пути. |
| **properties** | **data.Manager** | Данные сегмента. Доступны следующие поля:<br>• index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;<br>• type: String — идентификатор типа сегмента, принимающий значение "transfer" для сегментов переходов;<br>• text: String — текстовое описание сегмента;<br>• distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;<br>• lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута. |

## События

| Имя | Описание |
|---|---|
| **update** | Обновление модели новыми данными. Экземпляр класса **Event**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**(segmentJson) | | Обновляет состояние модели. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getType() | String | Возвращает идентификатор типа сегмента. Для сегментов перехода маршрутов общественного транспорта возвращается строка "transfer". |

## Описание полей

### geometry

```
{geometry.base.LineString} geometry
```

Геометрия сегмента.

### path

```
{multiRouter.masstransit.PathModel} path
```

Ссылка на родительскую модель пути.

### properties

```
{data.Manager} properties
```

Данные сегмента. Доступны следующие поля:

- index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;
- type: String — идентификатор типа сегмента, принимающий значение "transfer" для сегментов переходов;
- text: String — текстовое описание сегмента;
- distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;
- lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**.

## Описание методов

### destroy

```
destroy(segmentJson)
```

Обновляет состояние модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| segmentJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа сегмента. Для сегментов перехода маршрутов общественного транспорта возвращается строка "transfer".

## multiRouter.masstransit.TransportSegment

**Примечание:**
Конструктор класса multiRouter.masstransit.TransportSegment скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение транспортного сегмента маршрута общественного транспорта. Сегментом пути маршрута общественного транспорта называется часть пути от одной пересадки до другой.

Поля | События | Методы

Создает отображение транспортного сегмента маршрута общественного транспорта.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.masstransit.TransportSegmentModel** | Модель данных сегмента. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| **properties** | **data.Manager** | Данные сегмента. Доступны следующие поля:<br>• index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;<br>• type: String — идентификатор типа сегмента, принимающий значение "transport" для транспортных сегментов;<br>• text: String — текстовое описание сегмента;<br>• transports: **ITransportProperties**[] — массив, описывающий конкретные транспортные средства, с помощью которых возможно перемещение по данному сегменту;<br>• stops: Object — описание остановок в формате GeoJson:FeatureCollection;<br>• distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;<br>• lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| update | Обновление отображения сегмента. Экземпляр класса Event. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание полей

### model

```
{multiRouter.masstransit.TransportSegmentModel} model
```

Модель данных сегмента.

### properties

```
{data.Manager} properties
```

Данные сегмента. Доступны следующие поля:

- index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;
- type: String — идентификатор типа сегмента, принимающий значение "transport" для транспортных сегментов;
- text: String — текстовое описание сегмента;
- transports: **ITransportProperties**[] — массив, описывающий конкретные транспортные средства, с помощью которых возможно перемещение по данному сегменту;
- stops: Object — описание остановок в формате GeoJson:FeatureCollection;
- distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;
- lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута.

## Описание событий

### update

Обновление отображения сегмента. Экземпляр класса **Event**.

# multiRouter.masstransit.TransportSegmentModel

---

**Примечание:**
Конструктор класса multiRouter.masstransit.TransportSegmentModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет **IEventEmitter**.

Модель данных транспортного сегмента пути маршрута общественного транспорта. Сегментом пути маршрута общественного транспорта называется часть пути от одной пересадки до другой.

Поля | События | Методы

Создает модель данных транспортного сегмента пути маршрута общественного транспорта.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **IEventEmitter**. |
| **geometry** | **geometry.base.LineString** | Геометрия сегмента. |
| **path** | **multiRouter.masstransit.PathModel** | Ссылка на родительскую модель пути. |
| **properties** | **data.Manager** | Данные сегмента. Доступны следующие поля:<br>• index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;<br>• type: String — идентификатор типа сегмента, принимающий значение "transport" для транспортных сегментов;<br>• text: String — текстовое описание сегмента;<br>• transports: **ITransportProperties**[] — массив, описывающий конкретные транспортные средства, с помощью которых возможно перемещение по данному сегменту;<br>• stops: Object — описание остановок в формате GeoJson:FeatureCollection;<br>• distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;<br>• lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута. |

## События

| Имя | Описание |
|---|---|
| **update** | Обновление модели новыми данными. Экземпляр класса **Event**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает модель. |
| **getStops**() | **multiRouter.masstransit.StopModel**[] | Возвращает массив остановок транспортного сегмента. |
| **getType**() | String | Возвращает идентификатор типа сегмента. Для транспортных сегментов маршрутов общественного транспорта возвращается строка "transport". |
| **update**(segmentJson) | | Обновляет состояние модели. |

## Описание полей

### geometry

```
{geometry.base.LineString} geometry
```

Геометрия сегмента.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### path

```
{multiRouter.masstransit.PathModel} path
```

Ссылка на родительскую модель пути.

### properties

```
{data.Manager} properties
```

Данные сегмента. Доступны следующие поля:

- index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;
- type: String — идентификатор типа сегмента, принимающий значение "transport" для транспортных сегментов;
- text: String — текстовое описание сегмента;
- transports: **ITransportProperties**[] — массив, описывающий конкретные транспортные средства, с помощью которых возможно перемещение по данному сегменту;
- stops: Object — описание остановок в формате GeoJson:FeatureCollection;
- distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;
- lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getStops

```
{multiRouter.masstransit.StopModel[]} getStops()
```

**Возвращает** массив остановок транспортного сегмента.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа сегмента. Для транспортных сегментов маршрутов общественного транспорта возвращается строка "transport".

### update

```
update(segmentJson)
```

Обновляет состояние модели.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| segmentJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.

## multiRouter.masstransit.WalkSegment

**Примечание:**
Конструктор класса multiRouter.masstransit.WalkSegment скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение пешего сегмента маршрута общественного транспорта. Сегментом пути маршрута общественного транспорта называется часть пути от одной пересадки до другой.

Поля | События | Методы

Создает отображение пешего сегмента маршрута общественного транспорта.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.masstransit.WalkSegmentModel** | Модель данных сегмента. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |
| **properties** | **data.Manager** | Данные сегмента. Доступны следующие поля:<br><br>• index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;<br>• type: String — идентификатор типа сегмента, принимающий значение "walk" для пеших сегментов;<br>• text: String — текстовое описание сегмента;<br>• distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;<br>• lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута. |
| **state** | **IDataManager** | Состояние геообъекта. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **IGeoObject**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br><br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br><br>Поля данных:<br><br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br><br>Унаследовано от **IChild**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| update | Обновление отображения сегмента. Экземпляр класса Event. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание полей

### model

```
{multiRouter.masstransit.WalkSegmentModel} model
```

Модель данных сегмента.

### properties

```
{data.Manager} properties
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Данные сегмента. Доступны следующие поля:

- index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;
- type: String — идентификатор типа сегмента, принимающий значение "walk" для пеших сегментов;
- text: String — текстовое описание сегмента;
- distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;
- lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута.

### Описание событий

#### update

Обновление отображения сегмента. Экземпляр класса **Event**.

## multiRouter.masstransit.WalkSegmentModel

---

**Примечание:**
Конструктор класса multiRouter.masstransit.WalkSegmentModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет **IEventEmitter**.

Модель данных пешего сегмента пути маршрута общественного транспорта. Сегментом пути маршрута общественного транспорта называется часть пути от одной пересадки до другой.

Поля | События | Методы

Создает модель данных пешего сегмента пути маршрута общественного транспорта.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **geometry** | **geometry.base.LineString** | Геометрия сегмента. |
| **path** | **multiRouter.masstransit.PathModel** | Ссылка на родительскую модель пути. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| properties | data.Manager | Данные сегмента. Доступны следующие поля:<br>• index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;<br>• type: String — идентификатор типа сегмента, принимающий значение "walk" для пеших сегментов;<br>• text: String — текстовое описание сегмента;<br>• distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;<br>• duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;<br>• lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута. |

## События

| Имя | Описание |
|---|---|
| update | Обновление модели новыми данными. Экземпляр класса Event. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает модель. |
| getType() | String | Возвращает идентификатор типа сегмента. Для пеших сегментов маршрутов общественного транспорта возвращается строка "walk". |

## Описание полей

### geometry

```
{geometry.base.LineString} geometry
```

Геометрия сегмента.

### path

```
{multiRouter.masstransit.PathModel} path
```

Ссылка на родительскую модель пути.

### properties

```
{data.Manager} properties
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Данные сегмента. Доступны следующие поля:

- index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;
- type: String — идентификатор типа сегмента, принимающий значение "walk" для пеших сегментов;
- text: String — текстовое описание сегмента;
- distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;
- duration: Object — объект с полями text и value, описывающий время проезда сегмента в секундах;
- lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа сегмента. Для пеших сегментов маршрутов общественного транспорта возвращается строка "walk".


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter.MultiRoute

Расширяет **IGeoObject**.

Мультимаршрут на карте. Позволяет отображать на карте маршрут и несколько альтернатив к нему. Предоставляет интерфейс выбора активного маршрута.

Конструктор | Поля | События | Методы

## Конструктор

```
multiRouter.MultiRoute(model[, options])
```

Создает мультимаршрут на карте.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| `model` * | — | Тип: **multiRouter.MultiRouteModel** \| **IMultiRouteModelJson**<br><br>Модель данных мультимаршрута, либо объект описания модели. В случае передачи объекта описания, модель на его основе создается автоматически. |
| `options` | — | Тип: Object<br><br>Опции мультимаршрута. Также для задания опций составных частей мультимаршрута доступны следующие префиксы:<br>• `wayPoint` — опции путевых точек;<br>• `wayPointStart` — опции отображения начальной путевой точки;<br>• `wayPointFinish` — опции отображения конечной путевой точки;<br>• `viaPoint` — опции транзитных точек;<br>• `pin` — опции точечных маркеров на маршруте;<br>• `editor` — опции редактора мультимаршрута (см. **multiRouter.Editor**).<br><br>Опции единичных маршрутов в рамках мультимаршрута задаются с помощью префиксов:<br>• `route` — опции маршрутов в том числе неактивных;<br>• `routeActive` — опции активного маршрута. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Причем опции заданные с префиксом "routeActive" более приоритетны по отношению к опциям с префиксом "route". Для задания опций составных частей маршрутов общественного транспорта доступны следующие префиксы.<br>• `routeMarker` — опции меток сегментов маршрута;<br>• `routeWalkMarker` — опции меток пеших сегментов маршрута;<br>• `routeTransferMarker` — опции меток сегментов переходов;<br>• `routeTransportMarker` — опции меток транспортных сегментов маршрута;<br>• `routeWalkSegment` — опции линий пеших сегментов маршрута;<br>• `routeTransferSegment` — опции линий сегментов переходов;<br>• `routeTransportSegment` — опции линий транспортных сегментов маршрута;<br>Также данные префиксы доступны в "routeActive*" варианте. |
| `options.activeRouteAutoSelection` | true | Тип: Boolean<br>После обновления данных автоматически выбирать в качестве активного маршрут с минимальным временем проезда. |
| `options.boundsAutoApply` | false | Тип: Boolean<br>При добавлении мультимаршрута на карту позволяет автоматически установить центр и коэффициент масштабирования так, чтобы мультимаршрут был виден целиком. |
| `options.useMapMargin` | true | Тип: Boolean<br>Нужно ли учитывать отступы карты **map.margin.Manager**. |
| `options.zoomMargin` | 0 | Тип: Number \| Number[] |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Отступы от границ видимой области карты при изменении коэффициента масштабирования. Если задано одно число — оно применяется ко всем сторонам. Если задано два — то это горизонтальные и вертикальные отступы соответственно. Если задан массив из 4х чисел, то это отступы top, right, bottom, left. При включенной опции "useMapMargin" значение "zoomMargin" складывается со значениями, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

* Обязательный параметр/опция.

**Пример:**

```
// Создаем мультимаршрут и добавляем его на карту.
var multiRoute = new ymaps.multiRouter.MultiRoute({
    referencePoints: ['Москва, Ленинский проспект', 'Москва, Кулаков переулок'],
}, {
    editorDrawOver: false,
    wayPointDraggable: true,
    viaPointDraggable: true,
    // Задаем собственное оформление линий мультимаршрута.
    routeStrokeColor: "000088",
    routeActiveStrokeColor: "ff0000",
    pinIconFillColor: "ff0000",
    boundsAutoApply: true,
    zoomMargin: 30
});
myMap.geoObjects.add(multiRoute);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **editor** | **multiRouter.EditorAddon** | Редактор мультимаршрута. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.MultiRouteModel** | Модель данных мультимаршрута. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **properties** | **IDataManager** | Данные геообъекта.<br>Унаследовано от **IGeoObject**. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| activeroutechange | Изменение активного маршрута. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldActiveRoute: multiRouter.driving.Route\|multiRouter.masstransit.Route\|null — предыдущий активный маршрут. |
| boundsautoapply | Событие возникает в момент установки центра и коэффициента масштабирования карты для оптимального отображения мультимаршрута. Также смотрите описание опции boundsAutoApply. Экземпляр класса **Event**. |
| boundschange | Изменение координат географической области, охватывающей мультимаршрут. Экземпляр класса **Event**. |
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| geometrychange | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| mousemove | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **pixelboundschange** | Изменение пиксельных координат области, охватывающей мультимаршрут. Экземпляр класса **Event**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **update** | Обновление мультимаршрута. Экземпляр класса **Event**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getActiveRoute**() | **multiRouter.driving.Route** \| **multiRouter.masstransit.Route** \| null | Возвращает активный маршрут. |
| **getBounds**() | Number[][] \| null | Возвращает географические координаты области, охватывающей мультимаршрут. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getPixelBounds**() | Number[][] \| null | Возвращает глобальные пиксельные координаты области, охватывающей мультимаршрут. |
| **getRoutes**() | **GeoObjectCollection** | Возвращает дочернюю коллекцию единичных маршрутов мультимаршрута. |
| **getViaPoints**() | **GeoObjectCollection** | Возвращает дочернюю коллекцию транзитных точек мультимаршрута. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getWayPoints() | GeoObjectCollection | Возвращает дочернюю коллекцию путевых точек мультимаршрута. |
| setActiveRoute(route) | | Устанавливает активный маршрут. При этом предыдущий активный маршрут деактивируется, и генерируется событие multiRouter.MultiRoute.activeroutechange. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание полей

### editor

```
{multiRouter.EditorAddon} editor
```

Редактор мультимаршрута.

**Пример:**

```
// Начало редактирования маршрута.
ymaps.route(['Москва', 'Питер'], { multiRoute: true })
    .done(function (multiRoute) {
        myMap.geoObjects.add(multiRoute);
        multiRoute.editor.start({
            addWayPoints: true,
            removeWayPoints: true
        });
        // ...
        // Завершение редактирования маршрута.
        multiRoute.editor.stop();
    });
```

### model

```
{multiRouter.MultiRouteModel} model
```

Модель данных мультимаршрута.

## Описание событий

### activeroutechange

Изменение активного маршрута. Экземпляр класса Event. Имена полей, доступных через метод Event.get:

- oldActiveRoute: multiRouter.driving.Route|multiRouter.masstransit.Route|null — предыдущий активный маршрут.

### boundsautoapply

Событие возникает в момент установки центра и коэффициента масштабирования карты для оптимального отображения мультимаршрута. Также смотрите описание опции boundsAutoApply. Экземпляр класса Event.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### boundschange

Изменение координат географической области, охватывающей мультимаршрут. Экземпляр класса **Event**.

### pixelboundschange

Изменение пиксельных координат области, охватывающей мультимаршрут. Экземпляр класса **Event**.

### update

Обновление мультимаршрута. Экземпляр класса **Event**.

## Описание методов

### getActiveRoute

```
{multiRouter.driving.Route | multiRouter.masstransit.Route | null} getActiveRoute()
```

**Возвращает** активный маршрут.

### getBounds

```
{Number[][] | null} getBounds()
```

**Возвращает** географические координаты области, охватывающей мультимаршрут.

### getPixelBounds

```
{Number[][] | null} getPixelBounds()
```

**Возвращает** глобальные пиксельные координаты области, охватывающей мультимаршрут.

### getRoutes

```
{GeoObjectCollection} getRoutes()
```

**Возвращает** дочернюю коллекцию единичных маршрутов мультимаршрута.

### getViaPoints

```
{GeoObjectCollection} getViaPoints()
```

**Возвращает** дочернюю коллекцию транзитных точек мультимаршрута.

### getWayPoints

```
{GeoObjectCollection} getWayPoints()
```

**Возвращает** дочернюю коллекцию путевых точек мультимаршрута.

### setActiveRoute

```
setActiveRoute(route)
```

Устанавливает активный маршрут. При этом предыдущий активный маршрут деактивируется, и генерируется событие multiRouter.MultiRoute.activeroutechange.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| route* | — | Тип: multiRouter.driving.Route \| multiRouter.masstransit.Route \| null<br><br>Активируемый маршрут. |

* Обязательный параметр/опция.

## multiRouter.MultiRouteModel

Расширяет **IEventEmitter**.

Модель данных мультимаршрута.

Конструктор | Поля | События | Методы

### Конструктор

```
multiRouter.MultiRouteModel(referencePoints[, params])
```

Создает модель данных мультимаршрута.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| referencePoints * | — | Тип: IMultiRouteReferencePoint[]<br><br>Описание опорных точек мультимаршрута. |
| params | — | Тип: IMultiRouteParams<br><br>Параметры маршрутизации. |

* Обязательный параметр/опция.

**Пример:**

```
// Создаем модель мультимаршрута.
var multiRouteModel = new ymaps.multiRouter.MultiRouteModel(['Москва', 'Тверь',
'Питер'], {
    avoidTrafficJams: true,
    viaIndexes: [1]
});

// Создаем отображение мультимаршрута на основе модели.
var multiRouteView = new ymaps.multiRouter.MultiRoute(multiRouteModel);
myMap.geoObjects.add(multiRouteView);

// Подписываемся на события модели мультимаршрута.
multiRouteView.model.events
    .add("requestsuccess", function (event) {
        var routes = event.get("target").getRoutes();
        console.log("Найдено маршрутов: " + routes.length);
        for (var i = 0, l = routes.length; i < l; i++) {
            console.log("Длина маршрута " + (i + 1) + ": " +
routes[i].properties.get("distance").text);
        }
    })
    .add("requestfail", function (event) {
        console.log("Ошибка: " + event.get("error").message);
    });
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| properties | data.Manager | Данные мультимаршрута. |

## События

| Имя | Описание |
|---|---|
| requestcancel | Запрос за данными отменен. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• referencePoints: Object[] — массив, описывающий множество опорных точек;<br>• params: Object — параметры маршрутизации. |
| requestchange | Изменились опорные данные модели мультимаршрута (опорные точки, либо параметры маршрутизации). В результате формируется новый запрос за данными к сервису маршрутизации. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• referencePoints: Object[] — массив, описывающий множество опорных точек;<br>• params: Object — параметры маршрутизации;<br>• oldReferencePoints: Object[] — массив, описывающий предыдущее множество опорных точек;<br>• oldParams: Object — предыдущие параметры маршрутизации. |
| requestfail | Запрос за данными завершился с ошибкой. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• error: Error — объект ошибки. |
| requestsend | Отправлен новый запрос за данными модели мультимаршрута. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• referencePoints: Object[] — массив, описывающий множество опорных точек;<br>• params: Object — параметры маршрутизации. |
| requestsuccess | Запрос за данными завершился успешно, модель данных обновлена. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• init: Boolean — признак инициализирующего запроса;<br>• rough: Boolean — признак промежуточного запроса (используется для оптимизации размера ответа сервера при редактировании маршрута);<br>• wayPointsChange: Boolean — признак изменений в наборе путевых точек;<br>• viaPointsChange: Boolean — признак изменений в наборе транзитных точек;<br>• routesChange: Boolean — признак изменений в наборе маршрутов. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает модель. |
| getAllPoints() | (multiRouter.WayPointModel \| multiRouter.ViaPointModel)[] | Возвращает объединенный массив моделей транзитных и путевых точек в порядке, соответствующем порядку следования соответствующих опорных точек. |
| getJson() | Object | Возвращает JSON-данные модели мультимаршрута. |
| getParams() | IMultiRouteParams | Возвращает объект, описывающий текущие значения параметров маршрутизации. |
| getPoints() | (multiRouter.WayPointModel \| multiRouter.ViaPointModel)[] | Устаревшее название метода multiRouter.MultiRouteModel.getAllPoints. Не рекомендуется к использованию. |
| getReferencePointIndexes() | Object | Возвращает объект, содержащий следующие поля:<br>• way: Integer[] — индексы опорных точек, соответствующие множеству путевых точек модели;<br>• via: Integer[] — индексы опорных точек, соответствующие множеству транзитных точек модели.<br>. |
| getReferencePoints() | IMultiRouteReferencePoint[] | Возвращает массив опорных точек мультимаршрута. |
| getRoutes() | multiRouter.driving.RouteModel[] \| multiRouter.masstransit.RouteModel[] | Возвращает массив моделей дочерних маршрутов. |
| getViaPoints() | multiRouter.ViaPointModel[] | Возвращает массив моделей дочерних транзитных точек. |
| getWayPoints() | multiRouter.WayPointModel[] | Возвращает массив моделей дочерних путевых точек. |
| setParams(params[, extend[, clearRequests]]) | | Задает параметры маршрутизации. |
| setReferencePoints(referencePoints[, viaIndexes[, clearRequests]]) | | Задает опорные точки мультимаршрута. |

## Описание полей

### properties

```
{data.Manager} properties
```

Данные мультимаршрута.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание событий

### requestcancel

Запрос за данными отменен. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- referencePoints: Object[] — массив, описывающий множество опорных точек;
- params: Object — параметры маршрутизации.

### requestchange

Изменились опорные данные модели мультимаршрута (опорные точки, либо параметры маршрутизации). В результате формируется новый запрос за данными к сервису маршрутизации. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- referencePoints: Object[] — массив, описывающий множество опорных точек;
- params: Object — параметры маршрутизации;
- oldReferencePoints: Object[] — массив, описывающий предыдущее множество опорных точек;
- oldParams: Object — предыдущие параметры маршрутизации.

### requestfail

Запрос за данными завершился с ошибкой. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- error: Error — объект ошибки.

### requestsend

Отправлен новый запрос за данными модели мультимаршрута. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- referencePoints: Object[] — массив, описывающий множество опорных точек;
- params: Object — параметры маршрутизации.

### requestsuccess

Запрос за данными завершился успешно, модель данных обновлена. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- init: Boolean — признак инициализирующего запроса;
- rough: Boolean — признак промежуточного запроса (используется для оптимизации размера ответа сервера при редактировании маршрута);
- wayPointsChange: Boolean — признак изменений в наборе путевых точек;
- viaPointsChange: Boolean — признак изменений в наборе транзитных точек;
- routesChange: Boolean — признак изменений в наборе маршрутов.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getAllPoints

```
{(multiRouter.WayPointModel | multiRouter.ViaPointModel)[]} getAllPoints()
```

**Возвращает** объединенный массив моделей транзитных и путевых точек в порядке, соответствующем порядку следования соответствующих опорных точек.

## getJson

```
{Object} getJson()
```

**Возвращает** JSON-данные модели мультимаршрута.

## getParams

```
{IMultiRouteParams} getParams()
```

**Возвращает** объект, описывающий текущие значения параметров маршрутизации.

## getPoints

```
{(multiRouter.WayPointModel | multiRouter.ViaPointModel)[]} getPoints()
```

Устаревшее название метода multiRouter.MultiRouteModel.getAllPoints. Не рекомендуется к использованию.

**Возвращает** объединенный массив моделей транзитных и путевых точек в порядке, соответствующем порядку следования соответствующих опорных точек.

## getReferencePointIndexes

```
{Object} getReferencePointIndexes()
```

**Возвращает** объект, содержащий следующие поля:

- way: Integer[] — индексы опорных точек, соответствующие множеству путевых точек модели;
- via: Integer[] — индексы опорных точек, соответствующие множеству транзитных точек модели.

.

## getReferencePoints

```
{IMultiRouteReferencePoint[]} getReferencePoints()
```

**Возвращает** массив опорных точек мультимаршрута.

## getRoutes

```
{multiRouter.driving.RouteModel[] | multiRouter.masstransit.RouteModel[]}
getRoutes()
```

**Возвращает** массив моделей дочерних маршрутов.

## getViaPoints

```
{multiRouter.ViaPointModel[]} getViaPoints()
```

**Возвращает** массив моделей дочерних транзитных точек.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getWayPoints

```
{multiRouter.WayPointModel[]} getWayPoints()
```

**Возвращает** массив моделей дочерних путевых точек.

## setParams

```
setParams(params[, extend[, clearRequests]])
```

Задает параметры маршрутизации.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| params* | — | Тип: **IMultiRouteParams**<br><br>Параметры маршрутизации. |
| extend | false | Тип: Boolean<br><br>Позволяет изменить лишь часть параметров, а значения не переданных оставить прежним. |
| clearRequests | false | Тип: Boolean<br><br>Позволяет очистить очередь предыдущих запросов к серверу. |

* Обязательный параметр/опция.

## setReferencePoints

```
setReferencePoints(referencePoints[, viaIndexes[, clearRequests]])
```

Задает опорные точки мультимаршрута.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| referencePoints* | — | Тип: **IMultiRouteReferencePoint**[]<br><br>Массив опорных точек. |
| viaIndexes | — | Тип: Integer[]<br><br>Индексы транзитных точек в массиве опорных точек. |
| clearRequests | false | Тип: Boolean<br><br>Позволяет очистить очередь предыдущих запросов к серверу. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter.pedestrian

## multiRouter.pedestrian.Path

**Примечание:**
Конструктор класса multiRouter.pedestrian.Path скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение пути пешеходного маршрута. Единичный маршрут может содержать несколько путей, каждый путь соединяет две путевые точки.

Поля | События | Методы

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.pedestrian.PathModel** | Модель данных пути мультимаршрута. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **properties** | **data.Manager** | Данные пути мультимаршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;<br>• type: String — идентификатор типа маршрута, принимающий значение "pedestrian" для пешеходных маршрутов;<br>• distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;<br>• duration: Object — объект с полями text и value, описывающий время движения по пути в секундах;<br>• coordinates: Number[][] — координаты всех точек пути;<br>• encodedCoordinates: String — строка закодированных в base64 координат всех точек пути. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |

### События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br><br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **update** | Обновление отображения пути. Экземпляр класса **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br><br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br><br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br><br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br><br>Унаследован от IChildOnMap. |
| getSegmentMarkers() | GeoObjectCollection | Возвращает дочернюю коллекцию маркеров сегментов. |
| getSegments() | GeoObjectCollection | Возвращает дочернюю коллекцию сегментов, из которых состоит путь. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br><br>Унаследован от IChildOnMap. |

## Описание полей

### model

```
{multiRouter.pedestrian.PathModel} model
```

Модель данных пути мультимаршрута.

### properties

```
{data.Manager} properties
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Данные пути мультимаршрута. Доступны следующие поля:

- index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;
- type: String — идентификатор типа маршрута, принимающий значение "pedestrian" для пешеходных маршрутов;
- distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;
- duration: Object — объект с полями text и value, описывающий время движения по пути в секундах;
- coordinates: Number[][] — координаты всех точек пути;
- encodedCoordinates: String — строка закодированных в base64 координат всех точек пути.

## Описание событий

### update

Обновление отображения пути. Экземпляр класса **Event**.

## Описание методов

### getSegmentMarkers

```
{GeoObjectCollection} getSegmentMarkers()
```

**Возвращает** дочернюю коллекцию маркеров сегментов.

### getSegments

```
{GeoObjectCollection} getSegments()
```

**Возвращает** дочернюю коллекцию сегментов, из которых состоит путь.

# multiRouter.pedestrian.PathModel

---

**Примечание:**
Конструктор класса multiRouter.pedestrian.PathModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет **IEventEmitter**.

Модель данных пути пешеходного маршрута. Единичный маршрут может содержать несколько путей, каждый путь соединяет две путевые точки.

Поля | События | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| properties | data.Manager | Данные пути мультимаршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;<br>• type: String — идентификатор типа маршрута, принимающий значение "pedestrian" для пешеходных маршрутов;<br>• distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;<br>• duration: Object — объект с полями text и value, описывающий время движения по пути в секундах;<br>• coordinates: Number[][] — координаты всех точек пути;<br>• encodedCoordinates: String — строка закодированных в base64 координат всех точек пути. |
| route | multiRouter.pedestrian.RouteModel | Ссылка на родительскую модель маршрута. |

## События

| Имя | Описание |
|---|---|
| update | Обновление модели новыми данными. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• segmentsChange: Boolean — признак изменений в наборе сегментов. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает модель. |
| getSegments() | multiRouter.pedestrian.SegmentModel[] | Возвращает массив сегментов пути. |
| getType() | String | Возвращает идентификатор типа маршрута. Для пешеходных маршрутов возвращается строка "pedestrian". |
| update(pathJson) | | Обновляет состояние модели. |

## Описание полей

### properties

```
{data.Manager} properties
```

Данные пути мультимаршрута. Доступны следующие поля:

- index: Integer — порядковый номер пути в соответствующем маршруте мультимаршрута;
- type: String — идентификатор типа маршрута, принимающий значение "pedestrian" для пешеходных маршрутов;
- distance: Object — объект с полями text и value, описывающий протяженность пути в метрах;
- duration: Object — объект с полями text и value, описывающий время движения по пути в секундах;
- coordinates: Number[][] — координаты всех точек пути;
- encodedCoordinates: String — строка закодированных в base64 координат всех точек пути.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### route

```
{multiRouter.pedestrian.RouteModel} route
```

Ссылка на родительскую модель маршрута.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- segmentsChange: Boolean — признак изменений в наборе сегментов.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getSegments

```
{multiRouter.pedestrian.SegmentModel[]} getSegments()
```

**Возвращает** массив сегментов пути.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа маршрута. Для пешеходных маршрутов возвращается строка "pedestrian".

### update

```
update(pathJson)
```

Обновляет состояние модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pathJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter.pedestrian.Route

**Примечание:**
Конструктор класса multiRouter.pedestrian.Route скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение единичного пешеходного маршрута. Мультимаршрут может состоять из нескольких единичных маршрутов.

Поля | События | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.pedestrian.RouteModel** | Модель данных единичного маршрута. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **properties** | **data.Manager** | Данные маршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер маршрута в мультимаршруте;<br>• type: String — идентификатор типа маршрута, принимающий значение "pedestrian" для пешеходных маршрутов;<br>• distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;<br>• duration: Object — объект с полями text и value, описывающий время движения по маршруту в секундах;<br>• boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего маршрут. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| geometrychange | Изменение геометрии геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br><br>• originalEvent: IEvent — оригинальное событие геометрии.<br><br>Унаследовано от IGeoObject. |
| mapchange | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от IParentOnMap. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| mousemove | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| mouseup | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| multitouchend | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от IDomEventEmitter. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **update** | Обновление отображения маршрута. Экземпляр класса **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br><br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br><br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br><br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br><br>Унаследован от IChildOnMap. |
| getPaths() | GeoObjectCollection | Возвращает дочернюю коллекцию путей, из которых состоит маршрут. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br><br>Унаследован от IChildOnMap. |

## Описание полей

### model

```
{multiRouter.pedestrian.RouteModel} model
```

Модель данных единичного маршрута.

### properties

```
{data.Manager} properties
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Данные маршрута. Доступны следующие поля:

- index: Integer — порядковый номер маршрута в мультимаршруте;
- type: String — идентификатор типа маршрута, принимающий значение "pedestrian" для пешеходных маршрутов;
- distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;
- duration: Object — объект с полями text и value, описывающий время движения по маршруту в секундах;
- boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего маршрут.

## Описание событий

### update

Обновление отображения маршрута. Экземпляр класса **Event**.

## Описание методов

### getPaths

```
{GeoObjectCollection} getPaths()
```

**Возвращает** дочернюю коллекцию путей, из которых состоит маршрут.

# multiRouter.pedestrian.RouteModel

**Примечание:**
Конструктор класса multiRouter.pedestrian.RouteModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IEventEmitter**.

Модель данных единичного пешеходного маршрута. Мультимаршрут может состоять из нескольких единичных маршрутов.

Поля | События | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **multiRoute** | **multiRouter.MultiRouteModel** | Ссылка на родительскую модель мультимаршрута. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| properties | data.Manager | Данные маршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер маршрута в мультимаршруте;<br>• type: String — идентификатор типа маршрута, принимающий значение "pedestrian" для пешеходных маршрутов;<br>• distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;<br>• duration: Object — объект с полями text и value, описывающий время движения по маршруту в секундах;<br>• boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего маршрут. |

## События

| Имя | Описание |
|---|---|
| update | Обновление модели новыми данными. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• pathsChange: Boolean — признак изменений в наборе путей. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает модель. |
| getPaths() | multiRouter.pedestrian.PathModel[] | Возвращает массив путей маршрута. |
| getType() | String | Возвращает идентификатор типа маршрута. Для пешеходных маршрутов возвращается строка "pedestrian". |
| update(routeJson) | | Обновляет состояние модели. |

# Описание полей

## multiRoute

```
{multiRouter.MultiRouteModel} multiRoute
```

Ссылка на родительскую модель мультимаршрута.

## properties

```
{data.Manager} properties
```

Данные маршрута. Доступны следующие поля:

- index: Integer — порядковый номер маршрута в мультимаршруте;
- type: String — идентификатор типа маршрута, принимающий значение "pedestrian" для пешеходных маршрутов;
- distance: Object — объект с полями text и value, описывающий протяженность маршрута в метрах;
- duration: Object — объект с полями text и value, описывающий время движения по маршруту в секундах;
- boundedBy: Number[][] — координаты нижнего и верхнего углов прямоугольника, ограничивающего маршрут.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pathsChange: Boolean — признак изменений в наборе путей.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getPaths

```
{multiRouter.pedestrian.PathModel[]} getPaths()
```

**Возвращает** массив путей маршрута.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа маршрута. Для пешеходных маршрутов возвращается строка "pedestrian".

### update

```
update(routeJson)
```

Обновляет состояние модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| routeJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.

# multiRouter.pedestrian.Segment

**Примечание:**
Конструктор класса multiRouter.pedestrian.Segment скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение пешего сегмента пешеходного маршрута.

Поля | События | Методы


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |
| geometry | IGeometry \| null | Геометрия геообъекта.<br>Унаследовано от IGeoObject. |
| model | multiRouter.pedestrian.SegmentModel | Модель данных сегмента. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |
| properties | data.Manager | Данные сегмента. Доступны следующие поля:<br>• index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;<br>• type: String — идентификатор типа сегмента, принимающий значение "pedestrian" для пеших сегментов;<br>• text: String — текстовое описание сегмента;<br>• distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;<br>• duration: Object — объект с полями text и value, описывающий время движения по сегменту в секундах;<br>• lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута. |
| state | IDataManager | Состояние геообъекта.<br>Унаследовано от IGeoObject. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| geometrychange | Изменение геометрии геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие геометрии.<br>Унаследовано от IGeoObject. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| update | Обновление отображения сегмента. Экземпляр класса Event. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание полей

### model

```
{multiRouter.pedestrian.SegmentModel} model
```

Модель данных сегмента.

### properties

```
{data.Manager} properties
```

Данные сегмента. Доступны следующие поля:

- index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;
- type: String — идентификатор типа сегмента, принимающий значение "pedestrian" для пеших сегментов;
- text: String — текстовое описание сегмента;
- distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;
- duration: Object — объект с полями text и value, описывающий время движения по сегменту в секундах;
- lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута.

## Описание событий

### update

Обновление отображения сегмента. Экземпляр класса **Event**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter.pedestrian.SegmentModel

**Примечание:**
Конструктор класса multiRouter.pedestrian.SegmentModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IEventEmitter**.

Модель данных пешего сегмента пешеходного маршрута.

Поля | События | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **geometry** | **geometry.base.LineString** | Геометрия сегмента. |
| **path** | **multiRouter.pedestrian.PathModel** | Ссылка на родительскую модель пути. |
| **properties** | **data.Manager** | Данные сегмента. Доступны следующие поля:<br>• index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;<br>• type: String — идентификатор типа сегмента, принимающий значение "pedestrian" для пеших сегментов;<br>• text: String — текстовое описание сегмента;<br>• distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;<br>• duration: Object — объект с полями text и value, описывающий время движения по сегменту в секундах;<br>• lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута. |

## События

| Имя | Описание |
|---|---|
| **update** | Обновление модели новыми данными. Экземпляр класса **Event**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает модель. |
| **getType**() | String | Возвращает идентификатор типа сегмента. Для пеших сегментов пешеходных маршрутов возвращается строка "pedestrian". |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание полей

### geometry

```
{geometry.base.LineString} geometry
```

Геометрия сегмента.

### path

```
{multiRouter.pedestrian.PathModel} path
```

Ссылка на родительскую модель пути.

### properties

```
{data.Manager} properties
```

Данные сегмента. Доступны следующие поля:

- index: Integer — порядковый номер сегмента во множестве сегментов соответствующего пути маршрута;
- type: String — идентификатор типа сегмента, принимающий значение "pedestrian" для пеших сегментов;
- text: String — текстовое описание сегмента;
- distance: Object — объект с полями text и value, описывающий протяженность сегмента в метрах;
- duration: Object — объект с полями text и value, описывающий время движения по сегменту в секундах;
- lodIndex: Integer — порядковый номер первой точки сегмента во множестве всех координат соответствующего ему пути маршрута.

## Описание событий

### update

Обновление модели новыми данными. Экземпляр класса **Event**.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа сегмента. Для пеших сегментов пешеходных маршрутов возвращается строка "pedestrian".


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# multiRouter.ViaPoint

**Примечание:**
Конструктор класса multiRouter.ViaPoint скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение транзитной точки. Транзитные точки не подразумевают остановку. Таким образом, при прохождении через транзитную точку, сегмент пути маршрута не прерывается.

Поля | События | Методы

Создает отображение транзитной точки.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **model** | **multiRouter.ViaPointModel** | Модель данных транзитной точки. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **properties** | **data.Manager** | Данные транзитной точки мультимаршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер точки;<br>• lodIndex: Integer — порядковый номер транзитной точки во множестве всех координат соответствующего ей пути маршрута. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| | Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **update** | Обновление отображения транзитной точки. Экземпляр класса **Event**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание полей

### model

```
{multiRouter.ViaPointModel} model
```

Модель данных транзитной точки.

### properties

```
{data.Manager} properties
```

Данные транзитной точки мультимаршрута. Доступны следующие поля:

- index: Integer — порядковый номер точки;
- lodIndex: Integer — порядковый номер транзитной точки во множестве всех координат соответствующего ей пути маршрута.

## Описание событий

### update

Обновление отображения транзитной точки. Экземпляр класса Event.

# multiRouter.ViaPointModel

---

**Примечание:**
Конструктор класса multiRouter.ViaPointModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет IEventEmitter.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Модель данных транзитной точки мультимаршрута. Транзитные точки не подразумевают остановку. Таким образом, при прохождении через транзитную точку, сегмент пути маршрута не прерывается.

Поля | События | Методы

Создает модель данных транзитной точки мультимаршрута.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **geometry** | **geometry.base.Point** | Геометрия транзитной точки мультимаршрута. |
| **multiRoute** | **multiRouter.MultiRouteModel** | Ссылка на родительскую модель мультимаршрута. |
| **properties** | **data.Manager** | Данные транзитной точки мультимаршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер точки;<br>• lodIndex: Integer — порядковый номер транзитной точки во множестве всех координат соответствующего ей пути маршрута. |

## События

| Имя | Описание |
|---|---|
| **referencepointchange** | Изменена опорная точка. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldReferencePoint: Object — описание предыдущей опорной точки. |
| **update** | Обновление модели новыми данными. Экземпляр класса **Event**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает модель. |
| **getReferencePoint**() | Object | Возвращает соответствующую опорную точку. |
| **getReferencePointIndex**() | Integer | Возвращает индекс соответствующей опорной точки во множестве опорных точек родительского мультимаршрута. |
| **setReferencePoint**(referencePoint) | | Задает описание соответствующей опорной точки. Опорная точка может быть задана одним из следующих способов:<br>• строка, содержащая почтовый адрес опорной точки;<br>• массив, содержащий широту и долготу опорной точки;<br>• геометрия **geometry.Point**, описывающая опорную точку. |
| **update**(viaPointJson) | | Обновляет модель новыми данными. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание полей

### geometry

```
{geometry.base.Point} geometry
```

Геометрия транзитной точки мультимаршрута.

### multiRoute

```
{multiRouter.MultiRouteModel} multiRoute
```

Ссылка на родительскую модель мультимаршрута.

### properties

```
{data.Manager} properties
```

Данные транзитной точки мультимаршрута. Доступны следующие поля:

- index: Integer — порядковый номер точки;
- lodIndex: Integer — порядковый номер транзитной точки во множестве всех координат соответствующего ей пути маршрута.

## Описание событий

### referencepointchange

Изменена опорная точка. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- oldReferencePoint: Object — описание предыдущей опорной точки.

### update

Обновление модели новыми данными. Экземпляр класса **Event**.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getReferencePoint

```
{Object} getReferencePoint()
```

**Возвращает** соответствующую опорную точку.

### getReferencePointIndex

```
{Integer} getReferencePointIndex()
```

**Возвращает** индекс соответствующей опорной точки во множестве опорных точек родительского мультимаршрута.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## setReferencePoint

```
setReferencePoint(referencePoint)
```

Задает описание соответствующей опорной точки. Опорная точка может быть задана одним из следующих способов:

- строка, содержащая почтовый адрес опорной точки;
- массив, содержащий широту и долготу опорной точки;
- геометрия **geometry.Point**, описывающая опорную точку.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| referencePoint* | — | Тип: Object<br><br>Описание опорной точки. |

* Обязательный параметр/опция.

## update

```
update(viaPointJson)
```

Обновляет модель новыми данными.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| viaPointJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.

# multiRouter.WayPoint

**Примечание:**
Конструктор класса multiRouter.WayPoint скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **IGeoObject**.

Отображение путевой точки. Путевые точки подразумевают остановку и делят маршрут на т.н. пути.

Поля | События | Методы

Создает отображение путевой точки.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| geometry | IGeometry \| null | Геометрия геообъекта.<br>Унаследовано от IGeoObject. |
| model | multiRouter.WayPointModel | Модель данных путевой точки. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |
| properties | data.Manager | Данные путевой точки мультимаршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер точки;<br>• request: String — строка запроса соответствующего точке запроса;<br>• address: String — строка почтового адреса точки;<br>• description: String — описание точки;<br>• name: String — название точки;<br>• geocoderMetaData: Object — метаданные геокодера. |
| state | IDataManager | Состояние геообъекта.<br>Унаследовано от IGeoObject. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| geometrychange | Изменение геометрии геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие геометрии.<br>Унаследовано от IGeoObject. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| update | Обновление отображения путевой точки. Экземпляр класса Event. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание полей

### model

```
{multiRouter.WayPointModel} model
```

Модель данных путевой точки.

### properties

```
{data.Manager} properties
```

Данные путевой точки мультимаршрута. Доступны следующие поля:

- index: Integer — порядковый номер точки;
- request: String — строка запроса соответствующего точке запроса;
- address: String — строка почтового адреса точки;
- description: String — описание точки;
- name: String — название точки;
- geocoderMetaData: Object — метаданные геокодера.

## Описание событий

### update

Обновление отображения путевой точки. Экземпляр класса **Event**.

# multiRouter.WayPointModel

---

**Примечание:**
Конструктор класса multiRouter.WayPointModel скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет **IEventEmitter**.

Модель данных путевой точки мультимаршрута. Путевые точки подразумевают остановку и делят маршрут на т.н. пути.

Поля | События | Методы

Создает модель данных путевой точки мультимаршрута.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **geometry** | **geometry.base.Point** | Геометрия путевой точки мультимаршрута. |
| **multiRoute** | **multiRouter.MultiRouteModel** | Ссылка на родительскую модель мультимаршрута. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| properties | data.Manager | Данные путевой точки мультимаршрута. Доступны следующие поля:<br>• index: Integer — порядковый номер точки;<br>• request: String — строка запроса соответствующего точке запроса;<br>• address: String — строка почтового адреса точки;<br>• description: String — описание точки;<br>• name: String — название точки;<br>• geocoderMetaData: Object — метаданные геокодера. |

## События

| Имя | Описание |
|---|---|
| referencepointchange | Изменена опорная точка. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldReferencePoint: Object — описание предыдущей опорной точки. |
| update | Обновление модели новыми данными. Экземпляр класса Event. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает модель. |
| getReferencePoint() | Object | Возвращает соответствующую опорную точку. |
| getReferencePointIndex() | Integer | Возвращает индекс соответствующей опорной точки во множестве опорных точек родительского мультимаршрута. |
| setReferencePoint(referencePoint) | | Задает описание соответствующей опорной точки. Опорная точка может быть задана одним из следующих способов:<br>• строка, содержащая почтовый адрес опорной точки;<br>• массив, содержащий широту и долготу опорной точки;<br>• геометрия geometry.Point, описывающая опорную точку. |
| update(wayPointJson) | | Обновляет модель новыми данными. |

## Описание полей

### geometry

```
{geometry.base.Point} geometry
```

Геометрия путевой точки мультимаршрута.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### multiRoute

```
{multiRouter.MultiRouteModel} multiRoute
```

Ссылка на родительскую модель мультимаршрута.

### properties

```
{data.Manager} properties
```

Данные путевой точки мультимаршрута. Доступны следующие поля:

- index: Integer — порядковый номер точки;
- request: String — строка запроса соответствующего точке запроса;
- address: String — строка почтового адреса точки;
- description: String — описание точки;
- name: String — название точки;
- geocoderMetaData: Object — метаданные геокодера.

## Описание событий

### referencepointchange

Изменена опорная точка. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- oldReferencePoint: Object — описание предыдущей опорной точки.

### update

Обновление модели новыми данными. Экземпляр класса **Event**.

## Описание методов

### destroy

```
destroy()
```

Уничтожает модель.

### getReferencePoint

```
{Object} getReferencePoint()
```

**Возвращает** соответствующую опорную точку.

### getReferencePointIndex

```
{Integer} getReferencePointIndex()
```

**Возвращает** индекс соответствующей опорной точки во множестве опорных точек родительского мультимаршрута.

### setReferencePoint

```
setReferencePoint(referencePoint)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Задает описание соответствующей опорной точки. Опорная точка может быть задана одним из следующих способов:

- строка, содержащая почтовый адрес опорной точки;
- массив, содержащий широту и долготу опорной точки;
- геометрия **geometry.Point**, описывающая опорную точку.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| referencePoint* | — | Тип: Object<br><br>Описание опорной точки. |

* Обязательный параметр/опция.

## update

```
update(wayPointJson)
```

Обновляет модель новыми данными.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| wayPointJson* | — | Тип: Object<br><br>JSON-данные. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# ObjectManager

Расширяет **ICustomizable**, **IEventEmitter**, **IGeoObject**, **IParentOnMap**.

Менеджер объектов. Позволяет оптимально отображать, кластеризовать и управлять видимостью объектов. Обратите внимание, что у объектов, отрисованных на карте через данный менеджер, нельзя включать режимы редактирования и перетаскивания.

Конструктор | Поля | События | Методы

## Конструктор

```
ObjectManager([options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции.<br>• Можно задавать все опции, указанные в описании **Clusterer**, за исключением опций hasBalloon и hasHint.<br>• Опции для кластеров задаются с префиксом cluster. Список опций указан в описании класса **ClusterPlacemark**;<br>• Опции для одиночных объектов задаются с префиксом geoObject. Список опций указан в описании класса **Placemark**. |
| options.clusterize | false | Тип: Boolean<br><br>Флаг, показывающий, нужно ли кластеризовать объекты. Обратите внимание, что на данный момент кластеризация работает только для точечных объектов. При включенном режиме кластеризации все неточечные объекты будут игнорироваться. |
| options.syncOverlayInit | false | Тип: Boolean<br><br>Флаг, разрешающий создавать оверлеи для объектов синхронно. Обратите внимание, что при синхронном создании оверлея нужно самостоятельно обеспечить загрузку нужного класса, реализующего интерфейс IOverlay. По умолчанию оверлеи создаются асинхронно, при этом класс оверлея загружается по требованию. |
| options.viewportMargin | 128 | Тип: Number \| Number[]<br><br>Отступ для области, в которой показываются объекты. С помощью данной опции область показа объектов расширяется по отношению к видимой области карты. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Пример:**

```
var objectManager = new ymaps.ObjectManager({
    // Включаем кластеризацию.
    clusterize: true,
    // Опции кластеров задаются с префиксом cluster.
    clusterHasBalloon: false,
    // Опции геообъектов задаются с префиксом geoObject
    geoObjectOpenBalloonOnClick: false
});

// Можно задавать опции напрямую в дочерние коллекции.
objectManager.clusters.options.set({
    preset: 'islands#grayClusterIcons',
    hintContentLayout: ymaps.templateLayoutFactory.createClass('Группа
объектов')
});
objectManager.objects.options.set('preset', 'islands#grayIcon');
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **clusters** | **objectManager.ClusterCollection** | Коллекция кластеров, сгенерированных менеджером. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **objects** | **objectManager.ObjectCollection** | Коллекция объектов, добавленных в слой. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **properties** | **IDataManager** | Данные геообъекта.<br>Унаследовано от **IGeoObject**. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| add(objects) | ObjectManager | Возвращает ссылку на менеджер объектов. |
| getBounds() | Number[][] \| null | Вычисляет границы области в геокоординатах, охватывающей все объекты в составе менеджера. |
| getFilter() | String \| Function \| null | Возвращает заданную функцию-фильтр. |
| getMap() | Map | Возвращает ссылку на карту. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IParentOnMap**. |
| **getObjectState**(id) | Object | Получение информации о текущем состоянии объекта, добавленного в менеджер. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getPixelBounds**() | Number[][] \| null | Вычисляет границы области в глобальных пиксельных координатах, охватывающей все объекты в составе менеджера. |
| **remove**(objects) | **ObjectManager** | Возвращает ссылку на менеджер объектов. |
| **removeAll**() | **ObjectManager** | Удаление всех объектов из менеджера. |
| **setFilter**(filterFunction) | | Устанавливает функцию-фильтр для объектов. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание полей

### clusters

```
{objectManager.ClusterCollection} clusters
```

Коллекция кластеров, сгенерированных менеджером.

**Пример:**

```
objectManager.clusters.events.add('click', function (e) {
    var objectId = e.get('objectId');
    objectManager.clusters.balloon.open(objectId);
});
```

### objects

```
{objectManager.ObjectCollection} objects
```

Коллекция объектов, добавленных в слой.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Пример:**

```
objectManager.objects.events.add('click', function (e) {
    var objectId = e.get('objectId');
    objectManager.objects.balloon.open(objectId);
});
```

## Описание методов

### add

```
{ObjectManager} add(objects)
```

**Возвращает** ссылку на менеджер объектов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objects* | — | Тип: Object \| Object[] \| String<br><br>Строка или объект с JSON-описанием объектов. JSON-описание объектов строится по следующему принципу (см. пример ниже). Объектом может быть сущность или коллекция сущностей. Коллекция сущностей представляет из себя объект с полями:<br>• type — тип объекта. Значение поля должно быть равно "FeatureCollection";<br>• features — массив дочерних сущностей коллекции. Дочерние объекты могут быть сущностями или вложенными коллекциями сущностей.<br><br>Сущность представляет из себя объект с полями:<br>• id — уникальный идентификатор объекта. Обязательное поле;<br>• type — тип объекта. Значение поля должно быть равно "Feature";<br>• geometry — геометрия объекта. Содержит поля type и coordinates. Значение соответствует, передаваемому в конструктор **GeoObject**;<br>• options — опции геообъекта;<br>• properties — данные геообъекта. |

* Обязательный параметр/опция.

**Примеры:**

**1.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
var myObjectManager = new ymaps.ObjectManager({ clusterize: true }),
    currentId = 0;

// Добавление единичного объекта.
myObjectManager.add({
    type: 'Feature',
    id: currentId++,
    geometry: {
        type: 'Point',
        coordinates: [56.23, 34.79]
    },
    properties: {
        hintContent: 'Текст всплывающей подсказки',
        balloonContent: 'Содержимое балуна'
    }
});
map.geoObjects.add(objectManager);
```

**2.**

```
// Добавление массива точечных объектов.
var myObjects = [];
for (var i = 0, l = coordinates.length; i < l; i++) {
    myObjects.push({
        type: 'Feature',
        id: currentId++,
        geometry: {
            type: 'Point',
            coordinates: coordinates[i]
        }
    });
}
myObjectManager.add(myObjects);
map.geoObjects.add(objectManager);
```

**3.**

```
// Добавление коллекции объектов.
var collection = {
    type: 'FeatureCollection',
    features: [{
            type: 'Feature',
            id: currentId++,
            geometry: {
                type: 'Point',
                coordinates: [24.34, 65.24]
            }
        }, {
            type: 'Feature',
            id: currentId++,
            geometry: {
                type: 'Point',
                coordinates: [25.34, 63.24]
            }
        }
   ]
}
myObjectManager.add(collection);
map.geoObjects.add(objectManager);
```

**4.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Добавление неточечных объектов
var myObjects = [];

// Добавляем круг
myObjects.push({
    type: 'Feature',
    id: 1,
    geometry: {
        type: 'Circle',
        coordinates: [55.755381, 37.619044],
        radius: 300
    }
});

// Добавляем прямоугольник
myObjects.push({
    type: 'Feature',
    id: 2,
    geometry: {
        type: 'Rectangle',
        coordinates: [
            [55.764286, 37.606169],
            [55.759688, 37.620588]
        ]
    },
    options: {
        fillColor: '#FFFFFF',
        opacity: 0.8
    }
});

// Добавляем ломанную
myObjects.push({
    type: 'Feature',
    id: 3,
    geometry: {
        type: 'LineString',
        coordinates: [
            [55.75901100, 37.6308886],
            [55.7516538, 37.6299444],
            [55.74603822, 37.6380125]
        ]
    },
    options: {
        strokeColor: "#FF0000",
        strokeWidth: 5
    }
});

// Добавляем полигон
myObjects.push({
    type: 'Feature',
    id: 4,
    geometry: {
        type: 'Polygon',
        coordinates: [[
            [55.75175065, 37.6041094],
            [55.75005637, 37.6137224],
            [55.742891186, 37.6166407],
            [55.74153546, 37.60342281],
            [55.74700649, 37.59775798]
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
        ]]
    },
    options: {
        opacity: 0.2,
        strokeWidth: 2,
        fillColor: '#00FF00'
    }
});

objectManager.add(myObjects);
```

## getBounds

```
{Number[][] | null} getBounds()
```

Вычисляет границы области в геокоординатах, охватывающей все объекты в составе менеджера.

**Возвращает** массив координат области или null, если менеджер не добавлен на карту.

## getFilter

```
{String | Function | null} getFilter()
```

**Возвращает** заданную функцию-фильтр.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| * | — | Тип: |

* Обязательный параметр/опция.

## getObjectState

```
{Object} getObjectState(id)
```

Получение информации о текущем состоянии объекта, добавленного в менеджер.

**Возвращает** объект с полями:

- found — признак, указывающий, существует ли объект с переданным идентификатором. Тип: Boolean;
- isShown — признак, указывающий, находится ли объект в видимой области карты. Тип: Boolean;
- cluster — JSON-описание кластера, в который добавлен объект. Помимо обязательных полей, содержит поле properties.geoObjects с массивом объектов, попавших в кластер. Поле возвращается только при включенной кластеризации;
- isClustered — признак, указывающий, попал ли объект в состав кластера; Поле возвращается только при включенной кластеризации. Тип: Boolean;
- isFilteredOut — признак, указывающий, прошел ли объект фильтрацию. Если фильтр не задан или объект прошел фильтрацию, значение поля будет false. Тип: Boolean.

.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| id* | — | Тип: Object<br><br>Идентификатор объекта, для которого нужно получить состояние. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

**Пример:**

```
// Открытие балуна кластера с выбранным объектом.
// Получим данные о состоянии объекта внутри кластера.
var objectState = objectManager.getObjectState(myObjects[1].id);
// Проверяем, находится ли объект в видимой области карты.
if (objectState.found && objectState.isShown) {
    // Если объект попадает в кластер, открываем балун кластера с нужным выбранным
объектом.
    if (objectState.isClustered) {
        objectManager.clusters.state.set('activeObject', myObjects[1]);
        objectManager.clusters.balloon.open(objectState.cluster.id);
    } else {
        // Если объект не попал в кластер, открываем его собственный балун.
        objectManager.objects.balloon.open(myObjects[i].id);
    }
}
```

## getPixelBounds

```
{Number[][] | null} getPixelBounds()
```

Вычисляет границы области в глобальных пиксельных координатах, охватывающей все объекты в составе менеджера.

**Возвращает** массив координат области или null, если менеджер не добавлен на карту.

## remove

```
{ObjectManager} remove(objects)
```

**Возвращает** ссылку на менеджер объектов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objects* | — | Тип: Object \| Object[] \| String<br><br>Строка, объект с JSON-описанием объектов или массив идентификаторов удаляемых объектов. Формат описания объектов см. в описании метода ObjectManager.add |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Удаляем объекты с идентификаторами 0 и 56.
myObjectManager.remove([0, 56]);
```

**2.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Удаляем массив объектов.
myObjectManager.remove([
    {
        type: 'Feature',
        id: 23,
        geometry: {
            type: 'Point',
            coordinates: [45.33, 24.33]
        }
    },
    {
        type: 'Feature',
        id: 52,
        geometry: {
            type: 'Point',
            coordinates: [45.23, 24.34]
        }
    },
]);
```

**3.**

```
// Удаляем коллекцию объектов.
myObjectManager.remove({
    type: 'FeatureCollection',
    features: [
         // Описание объектов в составе коллекции.
         ...
    ]
});
```

## removeAll

```
{ObjectManager} removeAll()
```

Удаление всех объектов из менеджера.

**Возвращает** ссылку на себя.

**Пример:**

```
myObjectManager.removeAll();
```

## setFilter

```
setFilter(filterFunction)
```

Устанавливает функцию-фильтр для объектов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| filterFunction* | — | Тип: Function \| String |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | функция-фильтр. Получает на вход единичный объект, добавленный в ObjectManager. Если функция возвращает true, объект будет обработан. Если false — объект будет исключен из дальнейшей обработки. Также в качестве фильтра можно передавать строку. В строке-фильтре доступны следующие ключевые слова:<br>• options — обращение к опциям объекта;<br>• properties — обращение к данным объекта;<br>• geometry — обращение к геометрии объекта;<br>• id — обращение к идентификатору объекта;<br>В качестве фильтра можно указывать выражение, возвращающее true или false. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Пропустить объекты, id которых меньше 100.
objectManager.setFilter('id > 100');
```

**2.**

```
// На карте будут отображаться только объекты с заданными типами.
objectManager.setFilter('properties.type == "кафе" || properties.type ==
"аптека"');
```

**3.**

```
// Можно задавать функцию-фильтр.
objectManager.setFilter(function (object) {
    return object.properties.name != 'Тот, кого нельзя показывать.';
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# objectManager

## objectManager.addon

### objectManager.addon.clustersBalloon

**Примечание:**
Конструктор класса objectManager.addon.clustersBalloon скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Статический объект.

Методы

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

**Пример:**

```
<script src="http://api-maps.yandex.ru/2.1/?
load=Map,ObjectManager,objectManager.addon.clustersBalloon"type="text/
javascript"></script>
```

#### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get() | IPopupManager | Возвращает менеджер балуна менеджера объектов. |

#### Описание методов

##### get

```
{IPopupManager} get()
```

**Возвращает** менеджер балуна менеджера объектов.

**Пример:**

```
ymaps.objectManager.addon.clustersBalloon.get(myMap)
```

### objectManager.addon.clustersHint

**Примечание:**
Конструктор класса objectManager.addon.clustersHint скрыт, так как данный класс не предназначен для самостоятельной инициализации.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Статический объект.

Методы

---

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

---

**Пример:**

```
<script src="http://api-maps.yandex.ru/2.1/?
load=Map,ObjectManager,objectManager.addon.clustersHint"type="text/
javascript"></script>
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get() | IPopupManager | Возвращает менеджер хинта ObjectManager. |

## Описание методов

### get

```
{IPopupManager} get()
```

**Возвращает** менеджер хинта **ObjectManager**.

**Пример:**

```
ymaps.objectManager.addon.clustersHint.get(myMap)
```

# objectManager.addon.objectsBalloon

---

**Примечание:**
Конструктор класса objectManager.addon.objectsBalloon скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Статический объект.

Методы

---

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

---

**Пример:**

```
<script src="http://api-maps.yandex.ru/2.1/?
load=Map,ObjectManager,objectManager.addon.objectsBalloon"type="text/
javascript"></script>
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get(collection) | IPopupManager | Возвращает менеджер балуна менеджера объектов. |

## Описание методов

### get

```
{IPopupManager} get(collection)
```

**Возвращает** менеджер балуна менеджера объектов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| collection* | — | Тип: objectManager.ObjectCollection<br><br>Коллекция |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.objectManager.addon.objectsBalloon.get(myMap)
```

# objectManager.addon.objectsHint

**Примечание:**
Конструктор класса objectManager.addon.objectsHint скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Статический объект.

Методы

**Внимание!**
Не рекомендуется создавать экземпляры данного класса самостоятельно.

**Пример:**

```
<script src="http://api-maps.yandex.ru/2.1/?
load=Map,ObjectManager,objectManager.addon.objectsHint"type="text/javascript"></
script>
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get() | IPopupManager | Возвращает менеджер хинта ObjectManager. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### get

```
{IPopupManager} get()
```

**Возвращает** менеджер хинта **ObjectManager**.

**Пример:**

```
ymaps.objectManager.addon.objectsHint.get(myMap)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# objectManager.Balloon

Расширяет **IBalloonManager**.

Менеджер балуна коллекций ObjectManager. Позволяет управлять балуном объекта, открывать и скрывать его. Использует внутри себя менеджер балуна карты **map.Balloon**. Коллекции объектов в составе **ObjectManager** содержат экземпляр этого класса, доступный как myObjectManager.objects.balloon и myObjectsLayer.clusters.balloon. Не создавайте экземпляры этого класса без необходимости.

**См.: Balloon**

Конструктор | Поля | События | Методы

## Конструктор

```
objectManager.Balloon(collection)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| collection * | — | Тип: IReadOnlyCollection<br><br>Коллекция объектов. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **autopanbegin** | Начало автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **autopanend** | Окончание автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |
| **beforeuserclose** | Событие, предшествующее Balloon.userclose. Позволяет отменить пользовательское действие посредством вызова метода preventDefault. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• target — ссылка на объект **IBalloonOwner**.<br><br>Унаследовано от **IBalloonManager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **userclose** | Закрытие балуна пользователем. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• target — ссылка на объект **IBalloonOwner**.<br>Унаследовано от **IBalloonManager**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **autoPan**() | **vow.Promise** | Перемещает карту таким образом, чтобы балун оказался видимым.<br>Унаследован от **IBalloonManager**. |
| **close**([force]) | **vow.Promise** | Закрывает открытый балун. |
| **destroy**() | | Отключает менеджер инфо-объекта.<br>Унаследован от **IPopupManager**. |
| **getData**() | Object \| null | Возвращает хэш с описание объекта, на котором открыт балун или null, если балун не был открыт. |
| **getOptions**() | **IOptionManager** \| null | Возвращает менеджер опций или `null`.<br>Унаследован от **IPopupManager**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от **IPopupManager**. |
| **getOverlaySync**() | **IOverlay** \| null | Возвращает оверлей, если тот существует.<br>Унаследован от **IPopupManager**. |
| **getPosition**() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **isOpen**(id) | Boolean | Метод, определяющий, открыт ли балун на объекте с переданным идентификатором. |
| **open**(objectId, anchorPixelPosition) | **vow.Promise** | Открывает балун на объекте с переданным идентификатором. |
| **setData**(objectData) | **vow.Promise** | Задаёт новые данные для отображения балуна. |
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopupManager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### close

```
{vow.Promise} close([force])
```

Закрывает открытый балун.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| force | false | Тип: Boolean<br><br>Мгновенное закрытие. |

**Пример:**

```
// Закроем все балуны на слое.
objectManager.objects.balloon.close();
objectManager.clusters.balloon.close();
```

### getData

```
{Object | null} getData()
```

**Возвращает** хэш с описание объекта, на котором открыт балун или null, если балун не был открыт.

**Пример:**

```
var cluster = objectManager.clusters.balloon.getData();
if (cluster) {
    alert('Сейчас открыт балун кластера с ' +
cluster.properties.geoObjects.length + ' объектами.');
}
```

### isOpen

```
{Boolean} isOpen(id)
```

Метод, определяющий, открыт ли балун на объекте с переданным идентификатором.

**Возвращает** состояние балуна: открыт/закрыт.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| id* | — | Тип: Object<br><br>Идентификатор объекта. |

* Обязательный параметр/опция.

**Пример:**

```
// При повторном клике на метку будем закрывать балун.
objectManager.objects.options.set('hideIconOnBalloonOpen', false);
objectManager.objects.events.add('click', function (e) {
    var objectId = e.get('objectId');
    if (objectManager.objects.balloon.isOpen(objectId)) {
        objectManager.objects.balloon.close();
    }
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## open

```
{vow.Promise} open(objectId, anchorPixelPosition)
```

Открывает балун на объекте с переданным идентификатором.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objectId* | — | Тип: Object<br><br>Идентификатор объекта, на котором надо открыть балун. |
| anchorPixelPosition* | — | Тип: |

* Обязательный параметр/опция.

**Пример:**

```
// Загрузка данных объекта перед открытием балуна.
// Отменим автоматическое открытие балуна по клику на объекте.
objectManager.options.set('geoObjectOpenBalloonOnClick', false);
objectManager.objects.events.add('click', function (e) {
    var objectId = e.get('objectId');
    // По клику на объекте вызовем подгрузку данных для объекта.
    getObjectData.then(function (data) {
        // Как только данные готовы, зададим их объекту и откроем балун.
        objectManager.objects.getById(objectId).properties = data;
        objectManager.objects.balloon.open(objectId);
    });
});
```

## setData

```
{vow.Promise} setData(objectData)
```

Задаёт новые данные для отображения балуна.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objectData* | — | Тип: Object<br><br>Хэш с описанием объекта, на котором открыт балун. Соответствует описанию объекта, который подается на вход ObjectManager.add. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Подгрузка данных после открытия балуна на объекте.
objectManager.objects.events.add('balloonopen', function (e) {
    var objectId = e.get('objectId'),
        object = objectManager.objects.getById(objectId);
    if (!object.properties) {
        loadObjectData(objectId).then(function (data) {
            if (objectManager.balloon.isOpen(objectid)) {
                object.properties = data;
                objectManager.objects.balloon.setData(object);
            }
        });
    }
});
```

# objectManager.ClusterCollection

Расширяет **ICustomizable**, **IEventEmitter**.

Коллекция кластеров, сгенерированных **ObjectManager**. Кластеры добавляются и удаляются из коллекции автоматически в ходе работы и доступны только на чтение. Объект-кластер представляет из себя JSON-структуру, аналогичную объектам в составе слоя. Поля объекта-кластера:

- id — уникальный идентификатор кластера;
- geometry — описание геометрии кластера;
- properties — описание данных кластера. В поле properties.geoObjects хранится массив объектов, входящих в состав кластера;
- options — опции кластера. Необязательное поле.

Конструктор | Поля | События | Методы

## Конструктор

```
objectManager.ClusterCollection()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **balloon** | **objectManager.Balloon** | Балун кластера в составе менеджера. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| **options** | **option.Manager** | Менеджер опций. Имена полей, доступных через метод option.Manager.get:<br>• hasBalloon — флаг наличия у коллекции поля .balloon. Если при клике на кластер не нужно открывать балун, рекомендуется установить эту опцию в значение false, чтобы избежать лишних инициализаций;<br>• hasHint — флаг наличия у коллекции поля .hint. Если при наведении на кластер не нужно показывать всплывающую подсказку, рекомендуется установить эту опцию в значение false, чтобы избежать лишних инициализаций;<br>• hideIconOnBalloonOpen — скрывать иконку при открытии балуна. Значение по умолчанию true;<br>• openBalloonOnClick — опция, позволяющая запретить открытие балуна при клике на кластер. По умолчанию открытие балуна разрешено;<br>• openHintOnHover — опция, позволяющая запретить показ всплывающей подсказки при наведении на кластер. По умолчанию показ хинтов разрешен. |
| **overlays** | **objectManager.OverlayCollection** | Коллекция оверлеев кластеров. Все события, за исключением событий add и remove, пропагируются от коллекции оверлеев в коллекцию кластеров. |
| **state** | **data.Manager** | Состояние коллекции кластеров. Определяется следующими полями:<br>• activeObject — JSON-описание объекта, выбранного в балуне кластера. |

## События

| Имя | Описание |
|---|---|
| **add** | Добавление кластера в коллекцию. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор добавленного объекта;<br>• child — добавленный объект. |
| **clusteroptionschange** | Изменение опций кластера через метод objectManager.ClusterCollection.setClusterOptions Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор кластера, у которого заменили опции. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **remove** | Удаление кластера из коллекции. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор удаленного объекта;<br>• child — удаленный объект. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **each**(callback, context) | | |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getAll() | Object[] | Возвращает массив объектов, содержащихся в коллекции. |
| getById(id) | Object \| null | Возвращает объект-кластер с указанным идентификатором или null, если такого кластера не существует. |
| getIterator() | IIterator | Возвращает итератор по коллекции. |
| getLength() | Number | Возвращает количество объектов в коллекции. |
| getObjectManager() | ObjectManager | Возвращает родительский слой объектов коллекции. |
| setClusterOptions(objectId, options) | objectManager.ObjectCollection | Возвращает ссылку на себя. |

## Описание полей

### balloon

```
{objectManager.Balloon} balloon
```

Балун кластера в составе менеджера.

### options

```
{option.Manager} options
```

Менеджер опций. Имена полей, доступных через метод option.Manager.get:

- hasBalloon — флаг наличия у коллекции поля .balloon. Если при клике на кластер не нужно открывать балун, рекомендуется установить эту опцию в значение false, чтобы избежать лишних инициализаций;
- hasHint — флаг наличия у коллекции поля .hint. Если при наведении на кластер не нужно показывать всплывающую подсказку, рекомендуется установить эту опцию в значение false, чтобы избежать лишних инициализаций;
- hideIconOnBalloonOpen — скрывать иконку при открытии балуна. Значение по умолчанию true;
- openBalloonOnClick — опция, позволяющая запретить открытие балуна при клике на кластер. По умолчанию открытие балуна разрешено;
- openHintOnHover — опция, позволяющая запретить показ всплывающей подсказки при наведении на кластер. По умолчанию показ хинтов разрешен.

**Пример:**

```
objectManager.objects.options.set({
    preset: 'islands#greenDotIcon',
    hintContentLayout:
ymaps.templateLayoutFactory.createClass('{{propeties.name}}')
});
```

### overlays

```
{objectManager.OverlayCollection} overlays
```

Коллекция оверлеев кластеров. Все события, за исключением событий add и remove, пропагируются от коллекции оверлеев в коллекцию кластеров.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Меняем цвет иконки кластера при наведении.
objectManager.clusters.events.add(['mouseenter', 'mouseleave'], function (e) {
    var objectId = e.get('objectId'),
        overlay = objectManager.clusters.overlays.getById(objectId);
    if (e.get('type') == 'mouseenter') {
        setRedColor(objectId);
        overlay.events.add('mapchange', onMapChange);
    } else {
        setGreenColor(objectId);
        overlay.events.remove('mapchange', onMapChange);
    }
});

function onMapChange (e) {
    setGreenColor(objectManager.clusters.overlays.getId(e.get('target')));
}

function setGreenColor (objectId) {
    objectManager.clusters.setClusterOptions(objectId, {
        preset: 'islands#greenClusterIcons'
    });
}

function setRedColor (objectId) {
    objectManager.clusters.setClusterOptions(objectId, {
        preset: 'islands#redClusterIcons'
    });
}
```

### state

```
{data.Manager} state
```

Состояние коллекции кластеров. Определяется следующими полями:

- activeObject — JSON-описание объекта, выбранного в балуне кластера.

**Пример:**

```
// Откроем балун кластера с нужным выбранным объектом.
var objectState = objectManager.getObjectState(myObjects[i]);
if (objectState.isClustered) {
    objectManager.clusters.state.set('activeObject', myObjects[i]);
    objectManager.clusters.balloon.open(objectState.cluster.id);
}
```

## Описание событий

### add

Добавление кластера в коллекцию. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор добавленного объекта;
- child — добавленный объект.

### clusteroptionschange

Изменение опций кластера через метод objectManager.ClusterCollection.setClusterOptions Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор кластера, у которого заменили опции.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### remove

Удаление кластера из коллекции. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор удаленного объекта;
- child — удаленный объект.

## Описание методов

### each

```
each(callback, context)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик, в которую передаются объекты коллекции. |
| context* | — | Тип: Object<br><br>Контекст исполнения функции-обработчика. |

* Обязательный параметр/опция.

**Пример:**

```
var clusterizedObjectsCounter = 0;
objectManager.clusters.each(function (cluster) {
    clusterizedObjectsCounter += cluster.properties.geoObjects.length;
});
alert('На карте показано ' + clusterizedObjectsCounter + ' откластеризованных
объектов.');
```

### getAll

```
{Object[]} getAll()
```

**Возвращает** массив объектов, содержащихся в коллекции.

**Пример:**

```
var clusterArray = objectManager.clusters.getAll();
```

### getById

```
{Object | null} getById(id)
```

**Возвращает** объект-кластер с указанным идентификатором или null, если такого кластера не существует.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| id* | — | Тип: String<br><br>Идентификатор кластера. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

**Пример:**

```
// Будем менять цвет кластера, если в нем более 20 объектов.
objectManager.clusters.events.add('add', function (e) {
    var cluster = objectManager.clusters.getById(e.get('objectId')),
        objects = cluster.properties.geoObjects;
    if (objects.length > 20) {
        objectManager.clusters.setClusterOptions({
            preset: 'islands#redClusterIcons'
        });
    }
});
```

## getIterator

```
{IIterator} getIterator()
```

**Возвращает** итератор по коллекции.

**Пример:**

```
var clusterizedObjectsCounter = 0,
    it = objectManager.clusters.getIterator(),
    cluster;
while ((cluster = it.getNext()) != it.STOP_ITERATION)
    clusterizedObjectsCounter += cluster.properties.geoObjects.length;
});
alert('На карте показано ' + clusterizedObjectsCounter + ' откластеризованных
объектов.');
```

## getLength

```
{Number} getLength()
```

**Возвращает** количество объектов в коллекции.

**Пример:**

```
alert('На карте показано ' + objectManager.clusters.getLength() + ' кластеров.');
```

## getObjectManager

```
{ObjectManager} getObjectManager()
```

Возвращает родительский слой объектов коллекции.

**Возвращает** ObjectManager.

## setClusterOptions

```
{objectManager.ObjectCollection} setClusterOptions(objectId, options)
```

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objectId* | — | Тип: String<br><br>Идентификатор кластера. |
| options* | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Объект с опциями кластера. |

* Обязательный параметр/опция.

**Пример:**

```
// Будем менять цвет кластера, если в нем более 20 объектов.
objectManager.clusters.events.add('add', function (e) {
    var cluster = objectManager.clusters.getById(e.get('objectId')),
        objects = cluster.properties.geoObjects;
    if (objects.length > 20) {
        objectManager.clusters.setClusterOptions({
            preset: 'islands#redClusterIcons'
        });
    }
});
```

# objectManager.Hint

Расширяет **IHintManager**.

Менеджер хинта объектного слоя. Позволяет управлять хинтом объектного слоя, открывать и скрывать его. Использует внутри себя менеджер хинтов карты **map.Hint**. Объектные слои содержат экземпляры этого класса, доступные как myObjectManager.objects.hint и myObjectManager.clusters.hint. Не создавайте экземпляры этого класса без необходимости.

**См.: Hint**

Конструктор | Поля | Методы

## Конструктор

```
objectManager.Hint(collection)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| collection * | — | Тип: IReadOnlyCollection<br><br>Объектный слой. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **close**([force]) | **vow.Promise** | Скрывает всплывающую подсказку. |
| **destroy**() | | Отключает менеджер инфо-объекта.<br><br>Унаследован от **IPopupManager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getData**() | Object \| null | Возвращает хэш с описание объекта, на котором показан хинт или null, если хинт не был показан. |
| **getOptions**() | **IOptionManager** \| null | Возвращает менеджер опций или `null`.<br>Унаследован от **IPopupManager**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от **IPopupManager**. |
| **getOverlaySync**() | **IOverlay** \| null | Возвращает оверлей, если тот существует.<br>Унаследован от **IPopupManager**. |
| **getPosition**() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **isOpen**(id) | Boolean | Метод, определяющий, показана ли всплывающая подсказка на объекте с переданным идентификатором. |
| **open**(objectId[, position]) | **vow.Promise** | Показывает всплывающую подсказку на объекте с переданным идентификатором. |
| **setData**(objectData) | **vow.Promise** | Задаёт новые данные для отображения всплывающей подсказки. |
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopupManager**. |

## Описание методов

### close

```
{vow.Promise} close([force])
```

Скрывает всплывающую подсказку.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| force | false | Тип: Boolean<br>Мгновенное закрытие. |

**Пример:**

```
// Закроем все всплывающие подсказки на слое.
objectManager.objects.hint.close();
objectManager.clusters.hint.close();
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getData

```
{Object | null} getData()
```

**Возвращает** хэш с описание объекта, на котором показан хинт или null, если хинт не был показан.

**Пример:**

```
var cluster = objectManager.clusters.hint.getData();
if (cluster) {
    alert('На кластере показан хинт.');
}
```

## isOpen

```
{Boolean} isOpen(id)
```

Метод, определяющий, показана ли всплывающая подсказка на объекте с переданным идентификатором.

**Возвращает** состояние подсказки: показана/скрыта.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| id* | — | Тип: Object<br><br>Идентификатор объекта. |

* Обязательный параметр/опция.

**Пример:**

```
// Скроем подсказку при клике на объекте.
objectManager.objects.add('click', function (e) {
    var objectId = e.get('objectId');
    if (objectManager.objects.hint.isOpen(objectId)) {
        objectManager.objects.hint.close();
    }
});
```

## open

```
{vow.Promise} open(objectId[, position])
```

Показывает всплывающую подсказку на объекте с переданным идентификатором.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objectId* | — | Тип: Object<br><br>Идентификатор объекта, на котором надо открыть подсказку. |
| position | — | Тип: Number[]<br><br>Позиция, на которой надо показать всплывающую подсказку в глобальных пиксельных координатах. Если значение не задано, хинт покажется на геометрическом центре объекта. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

**Пример:**

```
objectManager.clusters.hint.open(objectId);
```

## setData

```
{vow.Promise} setData(objectData)
```

Задаёт новые данные для отображения всплывающей подсказки.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objectData* | — | Тип: Object<br><br>Хэш с описанием объекта, на котором открыт хинт. Соответствует описанию объекта, который подается на вход ObjectManager.add. |

* Обязательный параметр/опция.

**Пример:**

```
objectManager.objects.hint.setData(objectManager.objects.getById(objectId));
```

# objectManager.ObjectCollection

Расширяет **ICollection**, **ICustomizable**.

Коллекция объектов, добавленных в слой.

Конструктор | Поля | События | Методы

## Конструктор

```
objectManager.ObjectCollection()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **balloon** | **objectManager.Balloon** | Балун объекта в составе менеджера. |
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **hint** | **objectManager.Hint** | Хинт объекта в составе **ObjectManager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| options | option.Manager | Менеджер опций. Имена полей, доступных через метод option.Manager.get:<br>• hasBalloon — флаг наличия у коллекции поля .balloon. Если при клике на объекте не нужно открывать балун, рекомендуется установить эту опцию в значение false, чтобы избежать лишних инициализаций;<br>• hasHint — флаг наличия у коллекции поля .hint. Если при наведении на объект не нужно показывать всплывающую подсказку, рекомендуется установить эту опцию в значение false, чтобы избежать лишних инициализаций;<br>• hideIconOnBalloonOpen — скрывать иконку при открытии балуна. Значение по умолчанию true;<br>• openBalloonOnClick — показывать балун при клике на объект. Значение по умолчанию true. |
| overlays | objectManager.OverlayCollection | Коллекция оверлеев одиночных объектов. Все события, за исключением событий add и remove, пропагируются от коллекции оверлеев в коллекцию объектов. |

## События

| Имя | Описание |
|---|---|
| add | Добавление объекта в коллекцию. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор добавленного объекта;<br>• child — добавленный объект. |
| objectoptionschange | Изменение опций объекта через метод objectManager.ObjectCollection.setObjectOptions Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, у которого заменили опции. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| remove | Удаление объекта из коллекции. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор удаленного объекта;<br>• child — удаленный объект. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| add(data) | objectManager.ObjectCollection | Метод полностью дублирует логику ObjectManager.add. |
| each(callback, context) | | Метод, вызывающий переданную функцию-обработчик для всех элементов коллекции. |
| getAll() | Object[] | Возвращает массив объектов, содержащихся в коллекции. |
| getById(id) | Object \| null | Возвращает объект или null, если объекта с переданным id не существует. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getIterator() | IIterator | Возвращает итератор по коллекции.<br>Унаследован от ICollection. |
| getLength() | Number | Возвращает количество объектов в коллекции. |
| getObjectManager() | ObjectManager | Возвращает родительский слой объектов коллекции. |
| remove(data) | objectManager.ObjectCollection | Метод полностью дублирует логику ObjectManager.remove. |
| removeAll() | objectManager.ObjectCollection | Удаляет все элементы коллекции. |
| setObjectOptions(objectId, options) | objectManager.ObjectCollection | Метод, позволяющий динамически обновлять опции объекта. Метод следует использовать в случаях, когда вы хотите, чтобы новые опции мгновенно применились к отображению объекта на карте. |

## Описание полей

### balloon

```
{objectManager.Balloon} balloon
```

Балун объекта в составе менеджера.

### hint

```
{objectManager.Hint} hint
```

Хинт объекта в составе ObjectManager.

### options

```
{option.Manager} options
```

Менеджер опций. Имена полей, доступных через метод option.Manager.get:

- hasBalloon — флаг наличия у коллекции поля .balloon. Если при клике на объекте не нужно открывать балун, рекомендуется установить эту опцию в значение false, чтобы избежать лишних инициализаций;
- hasHint — флаг наличия у коллекции поля .hint. Если при наведении на объект не нужно показывать всплывающую подсказку, рекомендуется установить эту опцию в значение false, чтобы избежать лишних инициализаций;
- hideIconOnBalloonOpen — скрывать иконку при открытии балуна. Значение по умолчанию true;
- openBalloonOnClick — показывать балун при клике на объект. Значение по умолчанию true.

**Пример:**

```
// Зададим стили для одиночных объектов в составе коллекции.
objectManager.objects.options.set({
    preset: 'islands#redIcon',
    hasBalloon: false,
    zIndex: 500
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### overlays

```
{objectManager.OverlayCollection} overlays
```

Коллекция оверлеев одиночных объектов. Все события, за исключением событий add и remove, пропагируются от коллекции оверлеев в коллекцию объектов.

**Пример:**

```
objectManager.objects.overlays.events.add('add', function (e) {
    alert('Объект ' + e.get('objectId') + ' показан на карте.');
});
```

## Описание событий

### add

Добавление объекта в коллекцию. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор добавленного объекта;
- child — добавленный объект.

### objectoptionschange

Изменение опций объекта через метод objectManager.ObjectCollection.setObjectOptions Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, у которого заменили опции.

### remove

Удаление объекта из коллекции. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор удаленного объекта;
- child — удаленный объект.

## Описание методов

### add

```
{objectManager.ObjectCollection} add(data)
```

Метод полностью дублирует логику ObjectManager.add.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data* | — | Тип: Object \| Object[] \| String<br><br>Объекты, добавляемые в слой. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
objectManager.objects.add({
    type: 'Feature',
    geometry: {
        type: 'Point',
        coordinates: [55.33, 36.64]
    }
});
```

## each

```
each(callback, context)
```

Метод, вызывающий переданную функцию-обработчик для всех элементов коллекции.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик, в которую передаются объекты коллекции. |
| context* | — | Тип: Object<br><br>Контекст исполнения функции-обработчика. |

* Обязательный параметр/опция.

**Пример:**

```
var singleCounter = 0,
    clusterCounter = 0;
objectManager.objects.each(function (object) {
    var objectState = objectManager.getObjectState(object.id);
    if (objectState.isClustered) {
        clusterCounter++;
    } else {
        if (objectState.isShown) {
            singleCounter++;
        }
    }
});
alert('Количество показаных единичных меток: ' + singleCounter);
alert('Количество меток, показанных в составе кластера: ' + clusterCounter);
```

## getAll

```
{Object[]} getAll()
```

**Возвращает** массив объектов, содержащихся в коллекции.

## getById

```
{Object | null} getById(id)
```

**Возвращает** объект или null, если объекта с переданным id не существует.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| id* | — | Тип: Number |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Идентификатор объекта. |

* Обязательный параметр/опция.

**Пример:**

```
objectManager.objects.add('click', function (e) {
    var objectId = e.get('objectId'),
        object = objectManager.objects.getById(objectId);
});
```

## getLength

```
{Number} getLength()
```

**Возвращает** количество объектов в коллекции.

**Пример:**

```
alert('Количество объектов в слое: ' + objectManager.objects.getLength());
```

## getObjectManager

```
{ObjectManager} getObjectManager()
```

Возвращает родительский слой объектов коллекции.

**Возвращает** родительский слой объектов.

## remove

```
{objectManager.ObjectCollection} remove(data)
```

Метод полностью дублирует логику ObjectManager.remove.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data* | — | Тип: Object \| Object[] \| String<br><br>Объекты, которые нужно удалить. |

* Обязательный параметр/опция.

**Пример:**

```
// Удалим объекты с идентификаторами 34 и 25.
objectManager.objects.remove([34, 25]);
```

## removeAll

```
{objectManager.ObjectCollection} removeAll()
```

Удаляет все элементы коллекции.

**Возвращает** ссылку на себя.

**Пример:**

```
objectManager.objects.removeAll();
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## setObjectOptions

```
{objectManager.ObjectCollection} setObjectOptions(objectId, options)
```

Метод, позволяющий динамически обновлять опции объекта. Метод следует использовать в случаях, когда вы хотите, чтобы новые опции мгновенно применились к отображению объекта на карте.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| objectId* | — | Тип: Object<br><br>Идентификатор объекта, которому нужно задать опции. |
| options* | — | Тип: Object<br><br>Новые опции объекта. |

* Обязательный параметр/опция.

**Пример:**

```
// Меняем цвет иконки при наведении.
objectManager.objects.events.add('mouseenter', function (e) {
    var objectId = e.get('objectId'),
        overlay = objectManager.objects.overlays.getById(objectId);
    setRedColor(objectId);
    overlay.events.add('mapchange', setGreenColor);
});

objectManager.objects.events.add('mouseleave', function (e) {
    var objectId = e.get('objectId'),
        overlay = objectManager.objects.overlays.getById(objectId);
    setGreenColor(objectId);
    overlay.events.remove('mapchange', setGreenColor);
});

function setGreenColor (objectId) {
    objectManager.objects.setObjectOptions(objectId, {
        preset: 'islands#greenIcon'
    });
}

function setRedColor (objectId) {
    objectManager.objects.setClusterOptions(objectId, {
        preset: 'islands#redIcon'
    });
}
```

# objectManager.OverlayCollection

Расширяет **ICustomizable**, **IEventEmitter**.

Коллекция овелеев.

Конструктор | Поля | События | Методы

## Конструктор

```
objectManager.OverlayCollection()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| add | Добавление оверлея в коллекцию. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — добавленный оверлей. |
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |
| mousemove | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |
| mouseup | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| remove | Удаление оверлея из коллекции. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — удаленный оверлей. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• objectId — идентификатор объекта, которому принадлежит оверлей;<br>• overlay — IOverlay — оверлей, на котором произошло событие;<br>• coords — географические координаты точки, в которой произошло событие;<br>• globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;<br>• pagePixels — координаты события в пикселях от левого верхнего угла страницы;<br>• clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| each(callback, context) | | |
| getAll() | Object[] | Возвращает массив объектов, содержащихся в коллекции. |
| getById(id) | Object \| null | Возвращает оверлей или null, если оверлея с переданным id не существует. |
| getId(overlay) | Number \| null | Возвращает идентификатор объекта или null, если оверлей не содержится в коллекции. |
| getIterator() | IIterator | Возвращает итератор по коллекции. |
| getLength() | Number | Возвращает количество объектов в коллекции. |

## Описание событий

### add

Добавление оверлея в коллекцию. Экземпляр класса Event. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — добавленный оверлей.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## click

Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## contextmenu

Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## dblclick

Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## mousedown

Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## mouseenter

Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## mouseleave

Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## mousemove

Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## mouseup

Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## multitouchend

Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## multitouchmove

Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию ин-


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

терфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

### multitouchstart

Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Имена полей, доступных через метод IMultiTouchEvent.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

### remove

Удаление оверлея из коллекции. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — удаленный оверлей.

### wheel

Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- objectId — идентификатор объекта, которому принадлежит оверлей;
- overlay — IOverlay — оверлей, на котором произошло событие;
- coords — географические координаты точки, в которой произошло событие;
- globalPixels — координаты события в глобальных пикселях от левого верхнего угла мира;
- pagePixels — координаты события в пикселях от левого верхнего угла страницы;
- clientPixels — координаты события в пикселях от левого верхнего угла окна браузера;
- domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.

## Описание методов

### each

```
each(callback, context)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик, в которую передаются объекты коллекции. |
| context* | — | Тип: Object<br><br>Контекст исполнения функции-обработчика. |

* Обязательный параметр/опция.

**Пример:**

```
objectManager.clusters.overlays.each(function (overlay) {
     overlay.options.set('cursor', 'help');
});
```

## getAll

```
{Object[]} getAll()
```

**Возвращает** массив объектов, содержащихся в коллекции.

**Пример:**

```
var clusterOverlayArray = objectManager.clusters.overlays.getAll();
```

## getById

```
{Object | null} getById(id)
```

**Возвращает** оверлей или null, если оверлея с переданным id не существует.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| id* | — | Тип: Number<br><br>Идентификатор объекта, которому принадлежит оверлей. |

* Обязательный параметр/опция.

**Пример:**

```
objectManager.objects.add('mouseenter', function (e) {
    var objectId = e.get('objectId'),
        overlay = objectManager.objects.overlays.getById(objectId);
    overlay.options.set('zIndex', 100);
});
```

## getId

```
{Number | null} getId(overlay)
```

**Возвращает** идентификатор объекта или null, если оверлей не содержится в коллекции.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| overlay* | — | Тип: **IOverlay**<br><br>Оверлей. |

* Обязательный параметр/опция.

**Пример:**

```
objectManager.objects.overlays.each(function (overlay) {
    var objectId = objectManager.objects.overlays.getId(overlay);
    objectManager.objects.setObjectOptions(objectId, {
        preset: 'islands#redIcon'
    });
});
```

## getIterator

```
{IIterator} getIterator()
```

**Возвращает** итератор по коллекции.

**Пример:**

```
var it = objectManager.objects.overlays.getIterator(),
    overlay;
while ((overlay = it.getNext()) != it.STOP_ITERATION) {
    overlay.options.set('zIndex', 10);
}
```

## getLength

```
{Number} getLength()
```

**Возвращает** количество объектов в коллекции.

**Пример:**

```
var objectsNumber = objectManager.objects.getLength(),
    overlaysNumber = objectManager.objects.overlays.getNumber();
alert('На данный момент на карте показано ' + overlaysNumber + ' из ' +
objectsNumber.);
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# option

## option.Manager

Расширяет **IOptionManager**.

Менеджер опций. Позволяет задавать и получать значения опций по строковому ключу, а также разрешать значения опций в контексте существующей иерархии менеджеров опций.

Специальный ключ "preset" позволяет задать набор опций по умолчанию для данного менеджера. Значением опции "preset" может быть хэш вида {"имя опции": "значение опции"}, либо строковый идентификатор хэша опций в хранилище **option.presetStorage**. Данный хэш опций может также содержать поле с именем "preset", что позволяет наследовать значения опций от других наборов.

При поиске значения в иерархии сначала проверяются собственные опции, затем опции заданные с помощью ключа "preset", после чего происходит обращение к родителю, если он есть.

Для отслеживания изменений некоторых опций можно использовать **Monitor**.

Конструктор | Поля | События | Методы

### Конструктор

```
option.Manager([options[, parent[, name]]])
```

Создает менеджер опций.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Хэш опций. |
| parent | — | Тип: **IOptionManager**<br><br>Родительский менеджер опций. |
| name | — | Тип: String<br><br>Имя менеджера опций. |

**Примеры:**

**1.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Пример построения иерархии менеджеров опций.
var parentManager = new ymaps.option.Manager({
        key1: '123'
    }),
    childManager = new ymaps.option.Manager({
        key2: '234'
    }, parentManager);
// Выведет 123, значение берется из manager1.
alert(childManager.get('key1'));
// Выведет 234, значение берется из manager2.
alert(childManager.get('key2'));
// Перекрываем опцию.
childManager.set('key1', '345');
// Выведет 345, значение берется из manager2.
alert(childManager.get('key1'));
// Выведет 123, значение берется из manager1.
alert(parentManager.get('key1'));
```

**2.**

```
// Пример использования опции "preset".
var optionManager = new ymaps.option.Manager({
        preset: 'islands#blueIcon'
    }),
    subOptionManager = new ymaps.option.Manager();
// Данных нет, так как subOptionManager пуст.
alert(subOptionManager.get('iconImageSize'));
// Связываем два менеджера.
subOptionManager.setParent(optionManager);
// [37, 42] - значение получено из пресета в родительском менеджере.
alert(subOptionManager.get('iconImageSize'));
// Перекрываем значение iconImageSize на уровне subOptionManager.
subOptionManager.set('iconImageSize', [10, 12]);
// [10, 12] - значение получено из subOptionManager.
alert(subOptionManager.get('iconImageSize'));
// Отменяем перекрытие iconImageSize.
subOptionManager.unset('iconImageSize');
// [37, 42] - значение снова получено из пресета в родительском менеджере.
alert(subOptionManager.get('iconImageSize'));
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий объекта.<br>Унаследовано от **IFreezable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение опций.<br>Унаследовано от **IOptionManager**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| freeze() | IFreezable | Переводит объект в "замороженный" режим.<br>Унаследован от IFreezable. |
| get(key[, defaultValue]) | | Возвращает значение заданной опции в контексте существующей иерархии наследования опций. При вызове данного метода сначала происходит поиск значения в текущем менеджере опций, а затем, если значение не определено, поиск продолжается в иерархии родительских менеджеров.<br>Унаследован от IOptionManager. |
| getAll() | Object | Возвращает ссылку на внутренний хэш хранящий значения опций.<br>Унаследован от IOptionManager. |
| getName() | String | Возвращает имя менеджера опций.<br>Унаследован от IOptionManager. |
| getNative(key) | Object | Возвращает значение заданной опции, определенное на данном уровне иерархии опций, т.е. в данном менеджере.<br>Унаследован от IOptionManager. |
| getParent() | IOptionManager \| null | Возвращает родительский менеджер опций.<br>Унаследован от IOptionManager. |
| isFrozen() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от IFreezable. |
| resolve(key[, name]) | Object | Метод, предназначенный для вызова дочерними менеджерами опций.<br>Унаследован от IOptionManager. |
| set(key[, value]) | option.Manager | Задает значения опций для данного менеджера. Поддерживаются две сигнатуры:<br>• один аргумент, представляющий собой хеш вида {"имя опции": "значение опции"};<br>• два аргумента, первый из которых — имя опции, второй — значение. |
| setName(name) | | Задает имя менеджера опций.<br>Унаследован от IOptionManager. |
| setParent(parent) | IChild | Задает родительский менеджер опций. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IOptionManager**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |
| **unset**(keys) | **option.Manager** | Стирает значения заданных опций в данном менеджере. |
| **unsetAll**() | **option.Manager** | Стирает значения всех опции в данном менеджере. |

## Описание методов

### set

```
{option.Manager} set(key[, value])
```

Задает значения опций для данного менеджера. Поддерживаются две сигнатуры:

- один аргумент, представляющий собой хеш вида {"имя опции": "значение опции"};
- два аргумента, первый из которых — имя опции, второй — значение.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: Object \| String<br><br>Имя опции, либо хеш вида {"имя опции": "значение опции"}. |
| value | — | Тип: Object<br><br>Значение опции, если в качестве первого аргумента было передано имя. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
//Задание нескольких опций через хэш.
myMap.options.set({
    dblClickZoomCentering: true,
    dblClickFloatZoom: true
});
// Генерируется одно событие OptionManager.change.
```

**2.**

```
// Задание опций раздельно.
myMap.options
    .set("dblClickZoomCentering", true)
    .set("dblClickFloatZoom", true);
// После каждого set будет генерироваться событие OptionManager.change.
```

**3.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Использование freeze для минимизации количества событий OptionManager.change.
myMap.options.freeze();
myMap.options.set({
    dblClickZoomCentering: true,
    dblClickFloatZoom: true
});
myMap.options.set('cursor', 'zoom');
myMap.unfreeze();
// Генерируется одно событие OptionManager.change.
```

## unset

```
{option.Manager} unset(keys)
```

Стирает значения заданных опций в данном менеджере.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| keys* | — | Тип: String \| String[]<br><br>Имя опции или массив имен опций, значения которых должны быть аннулированы. |

* Обязательный параметр/опция.

**Пример:**

```
map.options.unset(['dblClickZoomCentering', 'dblClickFloatZoom']);
```

## unsetAll

```
{option.Manager} unsetAll()
```

Стирает значения всех опции в данном менеджере.

**Возвращает** ссылку на себя.

**Пример:**

```
var geoObject = new ymaps.Placemark([37, 55], {}, {preset:'islands#blueIcon'});
myMap.geoObjects.add(geoObject);
// Произвольным образом меняем стиль.
geoObject.options.set({
    iconLayout: 'default#image',
    iconImageHref: 'http://mysite.ru/icon.png',
    iconImageSize: [16, 16]
});
// Восстанавливаем первоначальный вид.
geoObject.options
    // Во избежание двойной реакции геообъекта
    // на изменение опций сначала вызываем freeze, а после задания
    // всех значений вызываем unfreeze.
    .freeze()
    .unsetAll()
    .set('preset','islands#blueIcon')
    .unfreeze();
```

# option.presetStorage

Статический объект.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Экземпляр класса **util.Storage**

Хранилище предустановленных опций. Список доступных ключей:

**Метки с текстом**

Содержимое метки задается через поле properties.iconContent.

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueIcon' | | 'islands#darkGreenIcon' |
| | 'islands#redIcon' | | 'islands#violetIcon' |
| | 'islands#darkOrangeIcon' | | 'islands#blackIcon' |
| | 'islands#nightIcon' | | 'islands#yellowIcon' |
| | 'islands#darkBlueIcon' | | 'islands#greenIcon' |
| | 'islands#pinkIcon' | | 'islands#orangeIcon' |
| | 'islands#grayIcon' | | 'islands#lightBlueIcon' |
| | 'islands#brownIcon' | | 'islands#oliveIcon' |

**Метки с текстом (иконки тянутся под контент)**

Содержимое метки задается через поле properties.iconContent.

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueStretchyIcon' | | 'islands#darkGreenStretchyIcon' |
| | 'islands#redStretchyIcon' | | 'islands#violetStretchyIcon' |



Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#darkOrangeStretchyIcon' | | 'islands#blackStretchyIcon' |
| | 'islands#nightStretchyIcon' | | 'islands#yellowStretchyIcon' |
| | 'islands#darkBlueStretchyIcon' | | 'islands#greenStretchyIcon' |
| | 'islands#pinkStretchyIcon' | | 'islands#orangeStretchyIcon' |
| | 'islands#grayStretchyIcon' | | 'islands#lightBlueStretchyIcon' |
| | 'islands#brownStretchyIcon' | | 'islands#oliveStretchyIcon' |

**Метки без содержимого с точкой в центре**

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueDotIcon' | | 'islands#darkGreenDotIcon' |
| | 'islands#redDotIcon' | | 'islands#violetDotIcon' |
| | 'islands#darkOrangeDotIcon' | | 'islands#blackDotIcon' |
| | 'islands#nightDotIcon' | | 'islands#yellowDotIcon' |
| | 'islands#darkBlueDotIcon' | | 'islands#greenDotIcon' |
| | 'islands#pinkDotIcon' | | 'islands#orangeDotIcon' |
| | 'islands#grayDotIcon' | | 'islands#lightBlueDotIcon' |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#brownDotIcon' | | 'islands#oliveDotIcon' |

### Метки в виде кругов с текстом

Содержимое метки задается через поле properties.iconContent.

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueCircleIcon' | | 'islands#darkGreenCircleIcon' |
| | 'islands#redCircleIcon' | | 'islands#violetCircleIcon' |
| | 'islands#darkOrangeCircleIcon' | | 'islands#blackCircleIcon' |
| | 'islands#nightCircleIcon' | | 'islands#yellowCircleIcon' |
| | 'islands#darkBlueCircleIcon' | | 'islands#greenCircleIcon' |
| | 'islands#pinkCircleIcon' | | 'islands#orangeCircleIcon' |
| | 'islands#grayCircleIcon' | | 'islands#lightBlueCircleIcon' |
| | 'islands#brownCircleIcon' | | 'islands#oliveCircleIcon' |

### Метки в виде кругов с точкой в центре

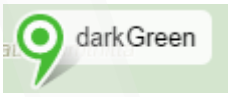
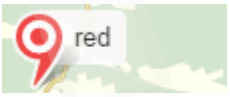
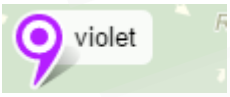
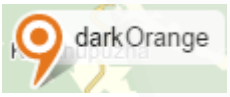
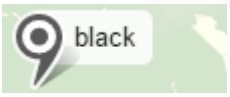
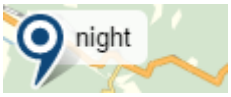
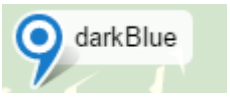
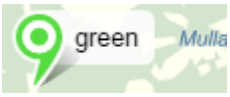
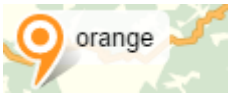
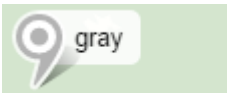
| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueCircleDotIcon' | | 'islands#darkGreenCircleDotIcon' |
| | 'islands#redCircleDotIcon' | | 'islands#violetCircleDotIcon' |
| | 'islands#darkOrangeCircleDotIcon' | | 'islands#blackCircleDotIcon' |
| | 'islands#nightCircleDotIcon' |  | 'islands#yellowCircleDotIcon' |
| | 'islands#darkBlueCircleDotIcon' | | 'islands#greenCircleDotIcon' |
| | 'islands#pinkCircleDotIcon' | | 'islands#orangeCircleDotIcon' |
| | 'islands#grayCircleDotIcon' | | 'islands#lightBlueCircleDotIcon' |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

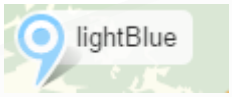
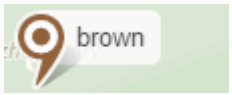
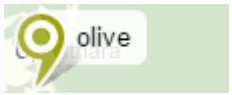
| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#brownCircleDotIcon' | | 'islands#oliveCircleDotIcon' |

**Метки с точкой в центре. Подпись в блоке справа**

Подпись задается через поле properties.iconCaption.

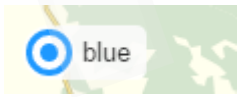
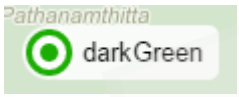
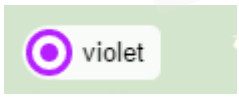
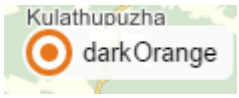
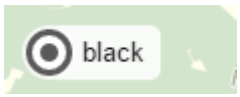
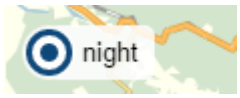
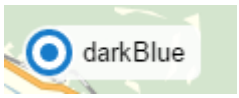
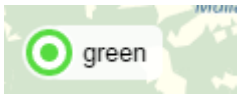
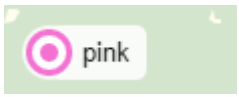
| Значок | Ключ |
|---|---|
| | 'islands#blueDotIconWithCaption' |
|  | 'islands#darkGreenDotIconWithCaption' |
|  | 'islands#redDotIconWithCaption' |
|  | 'islands#violetDotIconWithCaption' |
|  | 'islands#darkOrangeDotIconWithCaption' |
|  | 'islands#blackDotIconWithCaption' |
|  | 'islands#nightDotIconWithCaption' |
| | 'islands#yellowDotIconWithCaption' |
|  | 'islands#darkBlueDotIconWithCaption' |
|  | 'islands#greenDotIconWithCaption' |
| | 'islands#pinkDotIconWithCaption' |
|  | 'islands#orangeDotIconWithCaption' |
|  | 'islands#grayDotIconWithCaption' |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

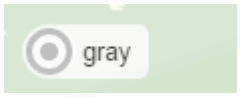
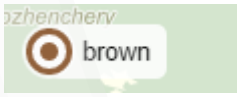
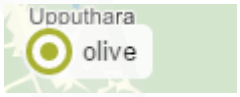
| Значок | Ключ |
| --- | --- |
|  | 'islands#lightBlueDotIconWithCaption' |
|  | 'islands#brownDotIconWithCaption' |
|  | 'islands#oliveDotIconWithCaption' |

**Метки в виде кругов с точкой в центре. Подпись в блоке справа**

Подпись задается через поле properties.iconCaption.

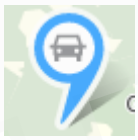
| Значок | Ключ |
| --- | --- |
|  | 'islands#blueCircleDotIconWithCaption' |
|  | 'islands#darkGreenCircleDotIconWithCaption' |
| | 'islands#redCircleDotIconWithCaption' |
|  | 'islands#violetCircleDotIconWithCaption' |
|  | 'islands#darkOrangeCircleDotIconWithCaption' |
|  | 'islands#blackCircleDotIconWithCaption' |
|  | 'islands#nightCircleDotIconWithCaption' |
| | 'islands#yellowCircleDotIconWithCaption' |
|  | 'islands#darkBlueCircleDotIconWithCaption' |
|  | 'islands#greenCircleDotIconWithCaption' |
|  | 'islands#pinkCircleDotIconWithCaption' |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

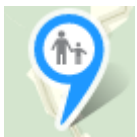
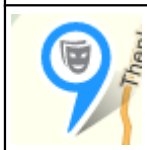
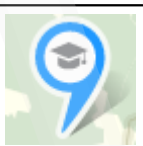
| Значок | Ключ |
|---|---|
| | 'islands#orangeCircleDotIconWithCaption' |
|   | 'islands#grayCircleDotIconWithCaption' |
| | 'islands#lightBlueCircleDotIconWithCaption' |
|   | 'islands#brownCircleDotIconWithCaption' |
|   | 'islands#oliveCircleDotIconWithCaption' |

### Метки со значком

Ключ для данного типа меток формируется по следующему правилу: 'islands#{цвет}{значок}Icon'. Например, 'islands#blueHomeIcon'. Ниже приведен список доступных значков и соответствующих для них ключей. Список доступных цветов можно посмотреть в любой другой таблице.

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueHomeIcon' | | 'islands#blueScienceIcon' |
| | 'islands#blueAirportIcon' |  | 'islands#blueAutoIcon' |
| | 'islands#blueBarIcon' | | 'islands#blueShoppingIcon' |
| | 'islands#blueFoodIcon' | | 'islands#blueSportIcon' |
| | 'islands#blueCinemaIcon' | | 'islands#blueVideoIcon' |
| | 'islands#blueMassTransitIcon' | | 'islands#blueRailwayIcon' |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueToiletIcon' | | 'islands#blueParkIcon' |
| | 'islands#blueBeachIcon' | | 'islands#bluePocketIcon' |
| | 'islands#blueZooIcon' | | 'islands#blueNightClubIcon' |
| | 'islands#blueUnderpassIcon' | | 'islands#bluePoolIcon' |
| | 'islands#blueRunIcon' | | 'islands#blueMedicalIcon' |
| | 'islands#blueBicycleIcon' | | 'islands#blueBicycle2Icon' |
| | 'islands#blueGardenIcon' | | 'islands#blueVegetationIcon' |
| | 'islands#blueObservationIcon' | | 'islands#blueGovernmentIcon' |
| | 'islands#blueEntertainmentCenterIcon' | | 'islands#blueCircusIcon' |
|  | 'islands#blueFamilyIcon' | | 'islands#blueRapidTransitIcon' |
|  | 'islands#blueTheaterIcon' |  | 'islands#blueEducationIcon' |
| | 'islands#blueBookIcon' | | 'islands#blueMountainIcon' |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueWaterwayIcon' | | 'islands#blueCarWashIcon' |
| | 'islands#blueFuelStationIcon' | | 'islands#blueFactoryIcon' |
| | 'islands#blueRepairShopIcon' | | 'islands#blueCourtIcon' |
| | 'islands#bluePostIcon' | | 'islands#blueHotelIcon' |
| | 'islands#blueWaterParkIcon' | | 'islands#blueChristianIcon' |
| | 'islands#blueWorshipIcon' | | 'islands#blueLaundryIcon' |
| | 'islands#blueFashionIcon' | | 'islands#blueSouvenirsIcon' |
| | 'islands#blueWasteIcon' | | 'islands#blueDogIcon' |
| | 'islands#blueMoneyIcon' | | 'islands#blueLeisureIcon' |
| | 'islands#blueHydroIcon' | | |

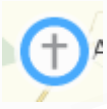
**Метки в виде круга со значком**

Ключ для данного типа меток формируется по следующему правилу: 'islands#{цвет}{значок}CircleIcon'. Например, 'islands#blueHomeCircleIcon'. Ниже приведен список доступных значков и соответствующих для них ключей. Список доступных цветов можно посмотреть в любой другой таблице.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueHomeCircleIcon' | | 'islands#blueScienceCircleIcon' |
| | 'islands#blueAirportCircleIcon' | | 'islands#blueAutoCircleIcon' |
| | 'islands#blueBarCircleIcon' | | 'islands#blueShoppingCircleIcon' |
| | 'islands#blueFoodCircleIcon' | | 'islands#blueSportCircleIcon' |
| | 'islands#blueCinemaCircleIcon' | | 'islands#blueVideoCircleIcon' |
| | 'islands#blueMassTransitCircleIcon' | | 'islands#blueRailwayCircleIcon' |
| | 'islands#blueToiletCircleIcon' | | 'islands#blueParkCircleIcon' |
| | 'islands#blueBeachCircleIcon' | | 'islands#bluePocketCircleIcon' |
| | 'islands#blueZooCircleIcon' | | 'islands#blueNightClubCircleIcon' |
| | 'islands#blueUnderpassCircleIcon' | | 'islands#bluePoolCircleIcon' |
| | 'islands#blueRunCircleIcon' | | 'islands#blueMedicalCircleIcon' |
| | 'islands#blueBicycleCircleIcon' | | 'islands#blueBicycle2CircleIcon' |
| | 'islands#blueGardenCircleIcon' | | 'islands#blueVegetationCircleIcon' |
| | 'islands#blueObservationCircleIcon' | | 'islands#blueGovernmentCircleIcon' |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueEntertainmentCenterCircleIcon' | | 'islands#blueCircusCircleIcon' |
| | 'islands#blueFamilyCircleIcon' | | 'islands#blueRapidTransitCircleIcon' |
| | 'islands#blueTheaterCircleIcon' | | 'islands#blueEducationCircleIcon' |
| | 'islands#blueBookCircleIcon' | | 'islands#blueMountainIcon' |
| | 'islands#blueWaterwayCircleIcon' | | 'islands#blueCarWashCircleIcon' |
| | 'islands#blueFuelStationCircleIcon' | | 'islands#blueFactoryCircleIcon' |
| | 'islands#blueRepairShopCircleIcon' | | 'islands#blueCourtCircleIcon' |
| | 'islands#bluePostCircleIcon' | | 'islands#blueHotelCircleIcon' |
| | 'islands#blueWaterParkCircleIcon' |  | 'islands#blueChristianCircleIcon' |
| | 'islands#blueWorshipCircleIcon' | | 'islands#blueLaundryCircleIcon' |
| | 'islands#blueFashionCircleIcon' | | 'islands#blueSouvenirsCircleIcon' |
| | 'islands#blueWasteCircleIcon' | | 'islands#blueDogCircleIcon' |
| | 'islands#blueMoneyCircleIcon' | | 'islands#blueLeisureCircleIcon' |
| | 'islands#blueHydroCircleIcon' | | |

**Пиктограммы**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Значок | Ключ |
|---|---|
| | 'islands#geolocationIcon' |

### Значки кластеров

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#blueClusterIcons' | | 'islands#invertedBlueClusterIcons' |
| | 'islands#redClusterIcons' | | 'islands#invertedRedClusterIcons' |
| | 'islands#darkOrangeClusterIcons' | | 'islands#invertedDarkOrangeClusterIcons' |
| | 'islands#nightClusterIcons' | | 'islands#invertedNightClusterIcons' |
| | 'islands#darkBlueClusterIcons' | | 'islands#invertedDarkBlueClusterIcons' |
| | 'islands#pinkClusterIcons' | | 'islands#invertedPinkClusterIcons' |
| | 'islands#grayClusterIcons' | | 'islands#invertedGrayClusterIcons' |
| | 'islands#brownClusterIcons' | | 'islands#invertedBrownClusterIcons' |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Значок | Ключ | Значок | Ключ |
|---|---|---|---|
| | 'islands#darkGreenClusterIcons' | | 'islands#invertedDarkGreenClusterIcons' |
| | 'islands#violetClusterIcons' | | 'islands#invertedVioletClusterIcons' |
| | 'islands#blackClusterIcons' | | 'islands#invertedBlackClusterIcons' |
| | 'islands#yellowClusterIcons' |  | 'islands#invertedYellowClusterIcons' |
| | 'islands#greenClusterIcons' | | 'islands#invertedGreenClusterIcons' |
| | 'islands#orangeClusterIcons' | | 'islands#invertedOrangeClusterIcons' |
| | 'islands#lightBlueClusterIcons' | | 'islands#invertedLightBlueClusterIcons' |
| | 'islands#oliveClusterIcons' | | 'islands#invertedOliveClusterIcons' |

Методы

```
option.presetStorage()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# overlay

## overlay.Circle

Расширяет **IOverlay**.

Интерактивный оверлей круга. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор | Поля | События | Методы

### Конструктор

```
overlay.Circle(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelCircleGeometry**<br><br>Пиксельная геометрия фигуры. |
| data | — | Тип: Object<br><br>Данные. |
| options | — | Тип: Object<br><br>Опции. |
| options.fill | — | Тип: Boolean<br><br>Наличие заливки **graphics.style.color**. |
| options.fillColor | — | Тип: String<br><br>Цвет заливки. |
| options.fillImageHref | — | Тип: String<br><br>Фоновое изображение. При включении данной опции значение fillColor игнорируется. |
| options.fillMethod | 'stretch' | Тип: String<br><br>Тип заливки фоном. Может принимать одно из двух значений:<br>• stretch — фоновое изображение растягивается по размеру оверлея.<br>• tile — фоновое изображение повторяется без изменений размера. Аналог background-repeat в css. Можно использовать для заливки фигуры неким шаблоном. |
| options.fillOpacity | — | Тип: Number |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Прозрачность заливки. |
| options.interactive | true | Тип: Boolean<br><br>Опция, позволяющая отключить реакцию объекта на dom-события. |
| options.opacity | — | Тип: Number<br><br>Общая прозрачность. |
| options.outline | — | Тип: Boolean<br><br>Наличие обводки. |
| options.separateContainer | — | Тип: Boolean<br><br>Производит отрисовку в отдельном слое. |
| options.strokeColor | — | Тип: String<br><br>Цвет линии **graphics.style.color**. |
| options.strokeOpacity | — | Тип: Number<br><br>Прозрачность контура. |
| options.strokeStyle | — | Тип: Number[] \| String<br><br>Стиль контура(не поддерживается в режиме Canvas) **graphics.style.stroke**. |
| options.strokeWidth | — | Тип: Number<br><br>Ширина линии. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **datachange** | Изменение данных. Поля данных:<br><br>• oldData — старые данные;<br>• newData — новые данные.<br><br>Унаследовано от **IOverlay**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты оверлея. Экземпляр класса **Event**.<br><br>Унаследовано от **IOverlay**. |
| **geometrychange** | Изменение геометрии. Поля данных:<br><br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия.<br><br>Унаследовано от **IOverlay**. |
| **mapchange** | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IOverlay**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchend | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от IDomEventEmitter. |
| multitouchmove | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br><br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры области охватывающей оверлей. Экземпляр класса Event.<br><br>Унаследовано от IOverlay. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getData() | Object | Возвращает объект данных оверлея.<br><br>Унаследован от IOverlay. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getGeometry**() | **IPixelGeometry** | Возвращает текущую пиксельную геометрию.<br>Унаследован от **IOverlay**. |
| **getMap**() | **Map** \| null | Возвращает ссылку на текущую карту.<br>Унаследован от **IOverlay**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.<br>Унаследован от **IOverlay**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **IOverlay**. |
| **setData**(data) | | Задает данные оверлея.<br>Унаследован от **IOverlay**. |
| **setGeometry**(geometry) | | Задает пиксельную геометрию оверлея.<br>Унаследован от **IOverlay**. |
| **setMap**(map) | | Задает карту, на которой должен отобразиться оверлей.<br>Унаследован от **IOverlay**. |

# overlay.hotspot

## overlay.hotspot.Circle

Расширяет overlay.hotspot.Base.

Оверлей круглой активной области. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор

### Конструктор

```
overlay.hotspot.Circle(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| `geometry` * | — | Тип: **IPixelCircleGeometry**<br>Геометрия. |
| `data` | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Данные. |
| options | — | Тип: Object<br><br>Опции. |

* Обязательный параметр/опция.

## overlay.hotspot.Placemark

Расширяет overlay.hotspot.Base.

Оверлей точечной активной области. По умолчанию оверлеи не добавлены в package.full (стандартный набор модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор

### Конструктор

```
overlay.hotspot.Placemark(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelPointGeometry**<br><br>Пиксельная геометрия фигуры. |
| data | — | Тип: Object<br><br>Данные. |
| options | — | Тип: Object<br><br>Опции. |

* Обязательный параметр/опция.

## overlay.hotspot.Polygon

Расширяет overlay.hotspot.Base.

Оверлей активной области в форме многоугольника. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор

### Конструктор

```
overlay.hotspot.Polygon(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelPolygonGeometry** |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Геометрия. |
| data | — | Тип: Object<br>Данные. |
| options | — | Тип: Object<br>Опции. |

* Обязательный параметр/опция.

## overlay.hotspot.Polyline

Расширяет overlay.hotspot.Base.

Оверлей активной области в форме ломаной линии. По умолчанию оверлеи не добавлены в package.full (стандартный набор модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор

### Конструктор

```
overlay.hotspot.Polyline(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelLineStringGeometry**<br>Пиксельная геометрия фигуры. |
| data | — | Тип: Object<br>Данные. |
| options | — | Тип: Object<br>Опции. |

* Обязательный параметр/опция.

## overlay.hotspot.Rectangle

Расширяет overlay.hotspot.Base.

Оверлей прямоугольной активной области. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор

### Конструктор

```
overlay.hotspot.Rectangle(geometry[, data[, options]])
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| `geometry` * | — | Тип: **IPixelRectangleGeometry**<br><br>Пиксельная геометрия фигуры. |
| `data` | — | Тип: Object<br><br>Данные. |
| `options` | — | Тип: Object<br><br>Опции. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# overlay.html

## overlay.html.Balloon

Расширяет **IOverlay**.

HTML-оверлей балуна. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор | Поля | События | Методы

### Конструктор

```
overlay.html.Balloon(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelPointGeometry**<br><br>Пиксельная геометрия фигуры. |
| data | — | Тип: Object<br><br>Данные. |
| options | — | Тип: Object<br><br>Опции. |
| options.cursor | — | Тип: String<br><br>Курсор при наведении мыши. |
| options.interactivityModel | "default#opaque" | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.layout | — | Тип: Function \| String<br><br>Макет. (Тип: конструктор объекта с интерфейсом ILayout). |
| options.offset | [0,0] | Тип: Array<br><br>Смещение в пикселях. |
| options.pane | "balloon" | Тип: String<br><br>Контейнер, в который будет помещен макет балуна. |
| options.shadow | true | Тип: Boolean<br><br>Флаг наличия тени. |
| options.shadowLayout | — | Тип: Function \| String<br><br>Макет тени (Тип: конструктор объекта с интерфейсом ILayout). |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.shadowOffset | [0,0] | Тип: Array<br><br>Смещение тени в пикселях. |
| options.shadowsPane | "shadows" | Тип: Array<br><br>Контейнер, в который будет помещен макет тени балуна. |
| options.zIndex | — | Тип: Number<br><br>z-index элемента. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **datachange** | Изменение данных. Поля данных:<br><br>• oldData — старые данные;<br>• newData — новые данные.<br><br>Унаследовано от **IOverlay**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты оверлея. Экземпляр класса **Event**.<br><br>Унаследовано от **IOverlay**. |
| **geometrychange** | Изменение геометрии. Поля данных:<br><br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия.<br><br>Унаследовано от **IOverlay**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IOverlay**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры области охватывающей оверлей. Экземпляр класса Event.<br>Унаследовано от IOverlay. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getBalloonElement() | HTMLElement | Возвращает родительский элемент макета балуна. |
| getBalloonLayout() | vow.Promise | Возвращает объект-обещание вернуть макет балуна. |
| getBalloonLayoutSync() | ILayout | Возвращает макет балуна. |
| getData() | Object | Возвращает объект данных оверлея.<br>Унаследован от IOverlay. |
| getElement() | HTMLElement | Возвращает родительский элемент макета балуна. |
| getGeometry() | IPixelGeometry | Возвращает текущую пиксельную геометрию.<br>Унаследован от IOverlay. |
| getLayout() | vow.Promise | Возвращает объект-обещание вернуть макет балуна. |
| getLayoutSync() | ILayout \| null | Возвращает макет балуна. |
| getMap() | Map \| null | Возвращает ссылку на текущую карту.<br>Унаследован от IOverlay. |
| getMode() | String | Возвращает текущий режим балуна. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getShadowElement() | HTMLElement | Возвращает родительский элемент макета тени балуна. |
| getShadowLayout() | vow.Promise | Возвращает объект-обещание вернуть макет тени балуна. |
| getShadowLayoutSync() | ILayout \| null | Возвращает макет тени балуна. |
| getShape() | IShape \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.<br>Унаследован от IOverlay. |
| isEmpty() | Boolean | Возвращает true, если макет пуст или если макет еще не был загружен, т.е. не имеет содержимого. |
| setData(data) | | Задает данные оверлея.<br>Унаследован от IOverlay. |
| setGeometry(geometry) | | Задает пиксельную геометрию оверлея.<br>Унаследован от IOverlay. |
| setMap(map) | | Задает карту, на которой должен отобразиться оверлей.<br>Унаследован от IOverlay. |

## Описание методов

### getBalloonElement

```
{HTMLElement} getBalloonElement()
```

**Возвращает** родительский элемент макета балуна.

### getBalloonLayout

```
{vow.Promise} getBalloonLayout()
```

**Возвращает** объект-обещание вернуть макет балуна.

### getBalloonLayoutSync

```
{ILayout} getBalloonLayoutSync()
```

**Возвращает** макет балуна.

### getElement

```
{HTMLElement} getElement()
```

**Возвращает** родительский элемент макета балуна.

### getLayout

```
{vow.Promise} getLayout()
```

**Возвращает** объект-обещание вернуть макет балуна.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### getLayoutSync

```
{ILayout | null} getLayoutSync()
```

**Возвращает** макет балуна.

### getMode

```
{String} getMode()
```

Возвращает текущий режим балуна.

**Возвращает** режим балуна: "panel" — режим панели "standard" — стандартное отображение.

### getShadowElement

```
{HTMLElement} getShadowElement()
```

**Возвращает** родительский элемент макета тени балуна.

### getShadowLayout

```
{vow.Promise} getShadowLayout()
```

**Возвращает** объект-обещание вернуть макет тени балуна.

### getShadowLayoutSync

```
{ILayout | null} getShadowLayoutSync()
```

**Возвращает** макет тени балуна.

### isEmpty

```
{Boolean} isEmpty()
```

**Возвращает** true, если макет пуст или если макет еще не был загружен, т.е. не имеет содержимого.

## overlay.html.Hint

Расширяет **IOverlay**.

Простой HTML-оверлей. По умолчанию оверлеи не добавлены в package.full (стандартный набор модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор | Поля | События | Методы

### Конструктор

```
overlay.html.Hint(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelPointGeometry**<br><br>Пиксельная геометрия фигуры. |
| data | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Данные. |
| options | — | Тип: Object<br><br>Опции. |
| options.cursor | — | Тип: String<br><br>Курсор при наведении мыши. |
| options.interactivityModel | 'default#opaque' | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.layout | — | Тип: **ILayout** \| String<br><br>Макет. |
| options.pane | "outerHint" | Тип: String<br><br>Контейнер, в который будет помещен оверлей. |
| options.zIndex | — | Тип: Number<br><br>z-index элемента. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **datachange** | Изменение данных. Поля данных:<br><br>• oldData — старые данные;<br>• newData — новые данные. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IOverlay**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты оверлея. Экземпляр класса **Event**.<br>Унаследовано от **IOverlay**. |
| **geometrychange** | Изменение геометрии. Поля данных:<br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия.<br>Унаследовано от **IOverlay**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IOverlay**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchmove | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры области охватывающей оверлей. Экземпляр класса Event.<br>Унаследовано от IOverlay. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getData() | Object | Возвращает объект данных оверлея.<br>Унаследован от IOverlay. |
| getElement() | HTMLElement | Возвращает родительский элемент макета иконки. |
| getGeometry() | IPixelGeometry | Возвращает текущую пиксельную геометрию.<br>Унаследован от IOverlay. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getLayout() | vow.Promise | Возвращает объект-обещание вернуть макет иконки. |
| getLayoutSync() | ILayout \| null | Возвращает макет иконки. |
| getMap() | Map \| null | Возвращает ссылку на текущую карту.<br>Унаследован от IOverlay. |
| getShape() | IShape \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.<br>Унаследован от IOverlay. |
| isEmpty() | Boolean | Возвращает true, если макет пуст или если макет еще не был загружен, т.е. не имеет содержимого. |
| setData(data) | | Задает данные оверлея.<br>Унаследован от IOverlay. |
| setGeometry(geometry) | | Задает пиксельную геометрию оверлея.<br>Унаследован от IOverlay. |
| setMap(map) | | Задает карту, на которой должен отобразиться оверлей.<br>Унаследован от IOverlay. |

## Описание методов

### getElement

```
{HTMLElement} getElement()
```

**Возвращает** родительский элемент макета иконки.

### getLayout

```
{vow.Promise} getLayout()
```

**Возвращает** объект-обещание вернуть макет иконки.

### getLayoutSync

```
{ILayout | null} getLayoutSync()
```

**Возвращает** макет иконки.

### isEmpty

```
{Boolean} isEmpty()
```

**Возвращает** true, если макет пуст или если макет еще не был загружен, т.е. не имеет содержимого.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# overlay.html.Placemark

Расширяет **IOverlay**.

HTML-оверлей метки. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор | Поля | События | Методы

## Конструктор

```
overlay.html.Placemark(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelPointGeometry**<br><br>Пиксельная геометрия. |
| data | — | Тип: Object<br><br>Данные. |
| options | — | Тип: Object<br><br>Опции. |
| options.cursor | — | Тип: String<br><br>Курсор при наведении мыши. |
| options.interactivityModel | 'default#geoObject' | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.layout | — | Тип: Function \| String<br><br>Макет. (Тип: конструктор объекта с интерфейсом ILayout). |
| options.offset | [0,0] | Тип: Array<br><br>Смещение в пикселях. |
| options.pane | "places" | Тип: String<br><br>Контейнер, в который будет помещен макет метки. |
| options.shadow | false | Тип: Boolean<br><br>Флаг наличия тени. |
| options.shadowLayout | — | Тип: Function \| String<br><br>Макет тени. (Тип: конструктор объекта с интерфейсом ILayout). |
| options.shadowOffset | [0,0] | Тип: Array |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Смещение тени в пикселях. |
| options.shadowsPane | "shadows" | Тип: Array<br><br>Контейнер, в который будет помещен макет тени метки. |
| options.zIndex | — | Тип: Number<br><br>z-index элемента. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| datachange | Изменение данных. Поля данных:<br>• oldData — старые данные;<br>• newData — новые данные.<br>Унаследовано от IOverlay. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| emptinesschange | Изменение признака пустоты оверлея. Экземпляр класса Event.<br>Унаследовано от IOverlay. |
| geometrychange | Изменение геометрии. Поля данных:<br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия.<br>Унаследовано от IOverlay. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IOverlay**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры области охватывающей оверлей. Экземпляр класса Event.<br>Унаследовано от IOverlay. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getData() | Object | Возвращает объект данных оверлея.<br>Унаследован от IOverlay. |
| getElement() | HTMLElement | Возвращает родительский элемент макета иконки. |
| getGeometry() | IPixelGeometry | Возвращает текущую пиксельную геометрию.<br>Унаследован от IOverlay. |
| getIconElement() | HTMLElement | Возвращает родительский элемент макета иконки. |
| getIconLayout() | vow.Promise | Возвращает объект-обещание вернуть макет иконки. |
| getIconLayoutSync() | ILayout \| null | Возвращает макет иконки. |
| getLayout() | vow.Promise | Возвращает объект-обещание вернуть макет иконки. |
| getLayoutSync() | ILayout \| null | Возвращает макет иконки. |
| getMap() | Map \| null | Возвращает ссылку на текущую карту.<br>Унаследован от IOverlay. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getShadowElement**() | HTMLElement | Возвращает родительский элемент макета тени иконки. |
| **getShadowLayout**() | **vow.Promise** | Возвращает объект-обещание вернуть макет тени иконки. |
| **getShadowLayoutSync**() | **ILayout** \| null | Возвращает макет тени иконки. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.<br>Унаследован от **IOverlay**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **IOverlay**. |
| **setData**(data) | | Задает данные оверлея.<br>Унаследован от **IOverlay**. |
| **setGeometry**(geometry) | | Задает пиксельную геометрию оверлея.<br>Унаследован от **IOverlay**. |
| **setMap**(map) | | Задает карту, на которой должен отобразиться оверлей.<br>Унаследован от **IOverlay**. |

## Описание методов

### getElement

```
{HTMLElement} getElement()
```

**Возвращает** родительский элемент макета иконки.

### getIconElement

```
{HTMLElement} getIconElement()
```

**Возвращает** родительский элемент макета иконки.

### getIconLayout

```
{vow.Promise} getIconLayout()
```

**Возвращает** объект-обещание вернуть макет иконки.

### getIconLayoutSync

```
{ILayout | null} getIconLayoutSync()
```

**Возвращает** макет иконки.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### getLayout

```
{vow.Promise} getLayout()
```

**Возвращает** объект-обещание вернуть макет иконки.

### getLayoutSync

```
{ILayout | null} getLayoutSync()
```

**Возвращает** макет иконки.

### getShadowElement

```
{HTMLElement} getShadowElement()
```

**Возвращает** родительский элемент макета тени иконки.

### getShadowLayout

```
{vow.Promise} getShadowLayout()
```

**Возвращает** объект-обещание вернуть макет тени иконки.

### getShadowLayoutSync

```
{ILayout | null} getShadowLayoutSync()
```

**Возвращает** макет тени иконки.

## overlay.html.Rectangle

Расширяет **IOverlay**.

HTML-оверлей прямоугольника. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор | Поля | События | Методы

### Конструктор

```
overlay.html.Rectangle(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelRectangleGeometry**<br>Пиксельная геометрия фигуры. |
| data | — | Тип: Object<br>Данные. |
| options | — | Тип: Object<br>Опции оверлея. |
| options.cursor | — | Тип: String |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Курсор при наведении мыши. |
| options.fillColor | — | Тип: String<br><br>Цвет заливки **graphics.style.color**. Опция стандартного макета прямоугольника. |
| options.fillImageHref | — | Тип: String<br><br>Фоновое изображение. При включении данной опции значение fillColor игнорируется. Опция стандартного макета прямоугольника. |
| options.fillMethod | 'stretch' | Тип: String<br><br>Тип заливки фоном. Может принимать одно из двух значений:<br>• stretch — Фоновое изображение растягивается по размеру оверлея.<br>• tile — Фоновое изображение повторяется без изменений размера. Аналог background-repeat в css. Можно использовать для заливки фигуры неким шаблоном.<br><br>Опция стандартного макета прямоугольника. |
| options.fillOpacity | — | Тип: Number<br><br>Прозрачность заливки. Опция стандартного макета прямоугольника. |
| options.interactivityModel | 'default#geoObject' | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.opacity | — | Тип: Number<br><br>Общая прозрачность. |
| options.pane | "areas" | Тип: String<br><br>Контейнер, в который будет помещен оверлей. |
| options.strokeColor | — | Тип: String<br><br>Цвет линии **graphics.style.color**. Опция стандартного макета прямоугольника. |
| options.strokeStyle | — | Тип: Number[] \| String |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Стиль обводки, поддерживаемый стандартным CSS-свойством *border-style*. Опция стандартного макета прямоугольника. |
| `options.strokeWidth` | — | Тип: Number<br><br>Ширина линии. Опция стандартного макета прямоугольника. |
| `options.zIndex` | — | Тип: Number<br><br>z-index элемента. |
| `dataSet.options.borderRadius` | — | Тип: Number<br><br>Радиус скругления углов. Опция стандартного макета прямоугольника. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **datachange** | Изменение данных. Поля данных:<br><br>• oldData — старые данные;<br>• newData — новые данные.<br><br>Унаследовано от **IOverlay**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты оверлея. Экземпляр класса **Event**.<br><br>Унаследовано от **IOverlay**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **geometrychange** | Изменение геометрии. Поля данных:<br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия.<br>Унаследовано от **IOverlay**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IOverlay**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры области охватывающей оверлей. Экземпляр класса Event.<br>Унаследовано от IOverlay. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getData() | Object | Возвращает объект данных оверлея.<br>Унаследован от IOverlay. |
| getElement() | HTMLElement | Возвращает родительский элемент макета прямоугольника. |
| getGeometry() | IPixelGeometry | Возвращает текущую пиксельную геометрию.<br>Унаследован от IOverlay. |
| getLayout() | vow.Promise | Возвращает объект-обещание вернуть макет прямоугольника. |
| getLayoutSync() | ILayout \| null | Возвращает макет прямоугольника. |
| getMap() | Map \| null | Возвращает ссылку на текущую карту.<br>Унаследован от IOverlay. |
| getShape() | IShape \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.<br>Унаследован от IOverlay. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **IOverlay**. |
| **setData**(data) | | Задает данные оверлея.<br>Унаследован от **IOverlay**. |
| **setGeometry**(geometry) | | Задает пиксельную геометрию оверлея.<br>Унаследован от **IOverlay**. |
| **setMap**(map) | | Задает карту, на которой должен отобразиться оверлей.<br>Унаследован от **IOverlay**. |

## Описание методов

### getElement

```
{HTMLElement} getElement()
```

**Возвращает** родительский элемент макета прямоугольника.

### getLayout

```
{vow.Promise} getLayout()
```

**Возвращает** объект-обещание вернуть макет прямоугольника.

### getLayoutSync

```
{ILayout | null} getLayoutSync()
```

**Возвращает** макет прямоугольника.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# overlay.Pin

Расширяет **IOverlay**.

Интерактивный оверлей круглой метки. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор | Поля | События | Методы

## Конструктор

```
overlay.Pin(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelPointGeometry**<br><br>Пиксельная геометрия метки. |
| data | — | Тип: Object<br><br>Данные. |
| options | — | Тип: Object<br><br>Опции. |
| options.fill | — | Тип: Boolean<br><br>Наличие заливки **graphics.style.color**. |
| options.fillColor | — | Тип: String<br><br>Цвет заливки. |
| options.fillImageHref | — | Тип: String<br><br>Фоновое изображение. При включении данной опции значение fillColor игнорируется. |
| options.fillMethod | 'stretch' | Тип: String<br><br>Тип заливки фоном. Может принимать одно из двух значений:<br>• stretch — фоновое изображение растягивается по размеру оверлея.<br>• tile — фоновое изображение повторяется без изменений размера. Аналог background-repeat в css. Можно использовать для заливки фигуры неким шаблоном. |
| options.fillOpacity | — | Тип: Number<br><br>Прозрачность заливки. |
| options.interactive | true | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опция, позволяющая отключить реакцию объекта на dom-события. |
| options.opacity | — | Тип: Number<br><br>Общая прозрачность. |
| options.outline | — | Тип: Boolean<br><br>Наличие обводки. |
| options.radius | — | Тип: Number<br><br>Радиус метки в пикселях. |
| options.separateContainer | — | Тип: Boolean<br><br>Производит отрисовку в отдельном слое. |
| options.strokeColor | — | Тип: String<br><br>Цвет линии **graphics.style.color**. |
| options.strokeOpacity | — | Тип: Number<br><br>Прозрачность контура. |
| options.strokeStyle | — | Тип: Number[] \| String<br><br>Стиль контура(не поддерживается в режиме Canvas) **graphics.style.stroke**. |
| options.strokeWidth | — | Тип: Number<br><br>Ширина линии. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **datachange** | Изменение данных. Поля данных:<br>• oldData — старые данные;<br>• newData — новые данные.<br><br>Унаследовано от **IOverlay**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты оверлея. Экземпляр класса **Event**.<br><br>Унаследовано от **IOverlay**. |
| **geometrychange** | Изменение геометрии. Поля данных:<br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия.<br><br>Унаследовано от **IOverlay**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IOverlay**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchend | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от IDomEventEmitter. |
| multitouchmove | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br><br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры области охватывающей оверлей. Экземпляр класса Event.<br><br>Унаследовано от IOverlay. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getData() | Object | Возвращает объект данных оверлея.<br><br>Унаследован от IOverlay. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getGeometry**() | **IPixelGeometry** | Возвращает текущую пиксельную геометрию.<br>Унаследован от **IOverlay**. |
| **getMap**() | **Map** \| null | Возвращает ссылку на текущую карту.<br>Унаследован от **IOverlay**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.<br>Унаследован от **IOverlay**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **IOverlay**. |
| **setData**(data) | | Задает данные оверлея.<br>Унаследован от **IOverlay**. |
| **setGeometry**(geometry) | | Задает пиксельную геометрию оверлея.<br>Унаследован от **IOverlay**. |
| **setMap**(map) | | Задает карту, на которой должен отобразиться оверлей.<br>Унаследован от **IOverlay**. |

# overlay.Placemark

Расширяет **IOverlay**.

Оверлей метки. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор | Поля | События | Методы

## Конструктор

```
overlay.Placemark(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelPointGeometry**<br>Пиксельная геометрия. |
| data | — | Тип: Object<br>Данные. |
| options | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опции. |
| options.cursor | — | Тип: String<br><br>Курсор при наведении мыши. |
| options.interactive | true | Тип: Boolean<br><br>Опция, позволяющая отключить реакцию объекта на dom-события. |
| options.interactivityModel | 'default#geoObject' | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.layout | — | Тип: Function \| String<br><br>Макет. (Тип: конструктор объекта с интерфейсом ILayout). |
| options.offset | [0,0] | Тип: Array<br><br>Смещение в пикселях. |
| options.pane | "places" | Тип: String<br><br>Контейнер, в который будет помещен макет метки. |
| options.shadow | false | Тип: Boolean<br><br>Флаг наличия тени. |
| options.shadowLayout | — | Тип: Function \| String<br><br>Макет тени. (Тип: конструктор объекта с интерфейсом ILayout). |
| options.shadowOffset | [0,0] | Тип: Array<br><br>Смещение тени в пикселях. |
| options.shadowsPane | "shadows" | Тип: Array<br><br>Контейнер, в который будет помещен макет тени метки. |
| options.zIndex | — | Тип: Number<br><br>z-index элемента. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| datachange | Изменение данных. Поля данных:<br>• oldData — старые данные;<br>• newData — новые данные.<br>Унаследовано от IOverlay. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| emptinesschange | Изменение признака пустоты оверлея. Экземпляр класса Event.<br>Унаследовано от IOverlay. |
| geometrychange | Изменение геометрии. Поля данных:<br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия.<br>Унаследовано от IOverlay. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IOverlay. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **shapechange** | Изменение фигуры области охватывающей оверлей. Экземпляр класса **Event**.<br><br>Унаследовано от **IOverlay**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getData**() | Object | Возвращает объект данных оверлея.<br>Унаследован от **IOverlay**. |
| **getElement**() | HTMLElement | Возвращает родительский элемент макета иконки. |
| **getGeometry**() | **IPixelGeometry** | Возвращает текущую пиксельную геометрию.<br>Унаследован от **IOverlay**. |
| **getIconElement**() | HTMLElement | Возвращает родительский элемент макета иконки. |
| **getIconLayout**() | **vow.Promise** | Возвращает объект-обещание вернуть макет иконки. |
| **getIconLayoutSync**() | **ILayout** \| null | Возвращает макет иконки. |
| **getLayout**() | **vow.Promise** | Возвращает объект-обещание вернуть макет иконки. |
| **getLayoutSync**() | **ILayout** \| null | Возвращает макет иконки. |
| **getMap**() | **Map** \| null | Возвращает ссылку на текущую карту.<br>Унаследован от **IOverlay**. |
| **getShadowElement**() | HTMLElement | Возвращает родительский элемент макета тени иконки. |
| **getShadowLayout**() | **vow.Promise** | Возвращает объект-обещание вернуть макет тени иконки. |
| **getShadowLayoutSync**() | **ILayout** \| null | Возвращает макет тени иконки. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.<br>Унаследован от **IOverlay**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **IOverlay**. |
| **setData**(data) | | Задает данные оверлея.<br>Унаследован от **IOverlay**. |
| **setGeometry**(geometry) | | Задает пиксельную геометрию оверлея.<br>Унаследован от **IOverlay**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| setMap(map) | | Задает карту, на которой должен отобразиться оверлей.<br><br>Унаследован от IOverlay. |

## Описание методов

### getElement

```
{HTMLElement} getElement()
```

**Возвращает** родительский элемент макета иконки.

### getIconElement

```
{HTMLElement} getIconElement()
```

**Возвращает** родительский элемент макета иконки.

### getIconLayout

```
{vow.Promise} getIconLayout()
```

**Возвращает** объект-обещание вернуть макет иконки.

### getIconLayoutSync

```
{ILayout | null} getIconLayoutSync()
```

**Возвращает** макет иконки.

### getLayout

```
{vow.Promise} getLayout()
```

**Возвращает** объект-обещание вернуть макет иконки.

### getLayoutSync

```
{ILayout | null} getLayoutSync()
```

**Возвращает** макет иконки.

### getShadowElement

```
{HTMLElement} getShadowElement()
```

**Возвращает** родительский элемент макета тени иконки.

### getShadowLayout

```
{vow.Promise} getShadowLayout()
```

**Возвращает** объект-обещание вернуть макет тени иконки.

### getShadowLayoutSync

```
{ILayout | null} getShadowLayoutSync()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** макет тени иконки.

## overlay.Polygon

Расширяет **IOverlay**.

Оверлей многоугольника.

Конструктор | Поля | События | Методы

### Конструктор

```
overlay.Polygon(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IPixelPolygonGeometry**<br><br>Пиксельная геометрия фигуры. |
| data | — | Тип: Object<br><br>Данные. |
| options | — | Тип: Object<br><br>Опции. |
| options.fill | — | Тип: String<br><br>Наличие заливки. |
| options.fillColor | — | Тип: String<br><br>Цвет заливки **graphics.style.color**. |
| options.fillImageHref | — | Тип: String<br><br>Фоновое изображение. При включении данной опции значение fillColor игнорируется. |
| options.fillMethod | 'stretch' | Тип: String<br><br>Тип заливки фоном. Может принимать одно из двух значений:<br>• stretch — Фоновое изображение растягивается по размеру оверлея.<br>• tile — Фоновое изображение повторяется без изменений размера. Аналог background-repeat в css. Можно использовать для заливки фигуры неким шаблоном. |
| options.fillOpacity | — | Тип: Number<br><br>Прозрачность заливки. |
| options.interactive | true | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опция, позволяющая отключить реакцию объекта на dom-события. |
| options.opacity | — | Тип: Number<br><br>Общая прозрачность. |
| options.outline | — | Тип: String<br><br>Наличие обводки. |
| options.separateContainer | — | Тип: Boolean<br><br>Производит отрисовку в отдельном слое. |
| options.strokeColor | — | Тип: String<br><br>Цвет линии **graphics.style.color**. |
| options.strokeOpacity | — | Тип: Number<br><br>Прозрачность контура. |
| options.strokeStyle | — | Тип: Number[] \| String<br><br>Стиль контура **graphics.style.stroke**. |
| options.strokeWidth | — | Тип: Number<br><br>Ширина линии. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **datachange** | Изменение данных. Поля данных:<br>• oldData — старые данные;<br>• newData — новые данные.<br>Унаследовано от **IOverlay**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты оверлея. Экземпляр класса **Event**.<br>Унаследовано от **IOverlay**. |
| **geometrychange** | Изменение геометрии. Поля данных:<br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия.<br>Унаследовано от **IOverlay**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IOverlay**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchmove | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры области охватывающей оверлей. Экземпляр класса Event.<br>Унаследовано от IOverlay. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getData() | Object | Возвращает объект данных оверлея.<br>Унаследован от IOverlay. |
| getGeometry() | IPixelGeometry | Возвращает текущую пиксельную геометрию.<br>Унаследован от IOverlay. |
| getMap() | Map \| null | Возвращает ссылку на текущую карту. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IOverlay**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.<br>Унаследован от **IOverlay**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **IOverlay**. |
| **setData**(data) | | Задает данные оверлея.<br>Унаследован от **IOverlay**. |
| **setGeometry**(geometry) | | Задает пиксельную геометрию оверлея.<br>Унаследован от **IOverlay**. |
| **setMap**(map) | | Задает карту, на которой должен отобразиться оверлей.<br>Унаследован от **IOverlay**. |

# overlay.Polyline

Расширяет **IOverlay**.

Оверлей линии. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор | Поля | События | Методы

## Конструктор

```
overlay.Polyline(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| `geometry` * | — | Тип: **IPixelLineStringGeometry**<br>Пиксельная геометрия фигуры. |
| `data` | — | Тип: Object<br>Данные. |
| `options` | — | Тип: Object<br>Опции. |
| `options.interactive` | true | Тип: Boolean<br>Опция, позволяющая отключить реакцию объекта на dom-события. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.opacity | — | Тип: Number<br><br>Общая прозрачность. |
| options.separateContainer | — | Тип: Boolean<br><br>Производит отрисовку в отдельном слое. |
| options.strokeColor | — | Тип: String<br><br>Цвет линии **graphics.style.color**. |
| options.strokeOpacity | — | Тип: Number<br><br>Прозрачность контура. |
| options.strokeStyle | — | Тип: Number[] \| String<br><br>Стиль контура **graphics.style.stroke**. |
| options.strokeWidth | — | Тип: Number<br><br>Ширина линии. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **datachange** | Изменение данных. Поля данных:<br><br>• oldData — старые данные;<br>• newData — новые данные.<br><br>Унаследовано от **IOverlay**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты оверлея. Экземпляр класса **Event**.<br>Унаследовано от **IOverlay**. |
| **geometrychange** | Изменение геометрии. Поля данных:<br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия.<br>Унаследовано от **IOverlay**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IOverlay**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchmove | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры области охватывающей оверлей. Экземпляр класса Event.<br>Унаследовано от IOverlay. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getData() | Object | Возвращает объект данных оверлея.<br>Унаследован от IOverlay. |
| getGeometry() | IPixelGeometry | Возвращает текущую пиксельную геометрию.<br>Унаследован от IOverlay. |
| getMap() | Map \| null | Возвращает ссылку на текущую карту. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IOverlay**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.<br>Унаследован от **IOverlay**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **IOverlay**. |
| **setData**(data) | | Задает данные оверлея.<br>Унаследован от **IOverlay**. |
| **setGeometry**(geometry) | | Задает пиксельную геометрию оверлея.<br>Унаследован от **IOverlay**. |
| **setMap**(map) | | Задает карту, на которой должен отобразиться оверлей.<br>Унаследован от **IOverlay**. |

# overlay.Rectangle

Расширяет **IOverlay**.

Оверлей прямоугольника. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). Для создания собственного экземпляра оверлея необходимо использовать **overlay.storage**.

Конструктор | Поля | События | Методы

## Конструктор

```
overlay.Rectangle(geometry[, data[, options]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| `geometry` * | — | Тип: **IPixelRectangleGeometry**<br>Пиксельная геометрия фигуры. |
| `data` | — | Тип: Object<br>Данные. |
| `options` | — | Тип: Object<br>Опции. |
| `options.borderRadius` | — | Тип: Number<br>Радиус скругления углов. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.fill | — | Тип: String<br><br>Наличие заливки. |
| options.fillColor | — | Тип: String<br><br>Цвет заливки **graphics.style.color**. |
| options.fillImageHref | — | Тип: String<br><br>Фоновое изображение. При включении данной опции значение fillColor игнорируется. |
| options.fillMethod | 'stretch' | Тип: String<br><br>Тип заливки фоном. Может принимать одно из двух значений:<br>• stretch — Фоновое изображение растягивается по размеру оверлея.<br>• tile — Фоновое изображение повторяется без изменений размера. Аналог background-repeat в css. Можно использовать для заливки фигуры неким шаблоном. |
| options.fillOpacity | — | Тип: Number<br><br>Прозрачность заливки. |
| options.interactive | true | Тип: Boolean<br><br>Опция, позволяющая отключить реакцию объекта на dom-события. |
| options.opacity | — | Тип: Number<br><br>Общая прозрачность. |
| options.outline | — | Тип: String<br><br>Наличие обводки. |
| options.separateContainer | — | Тип: Boolean<br><br>Производит отрисовку в отдельном слое. |
| options.strokeColor | — | Тип: String<br><br>Цвет линии **graphics.style.color**. |
| options.strokeOpacity | — | Тип: Number<br><br>Прозрачность контура. |
| options.strokeStyle | — | Тип: Number[] \| String<br><br>Стиль обводки (не поддерживается в режиме Canvas) **graphics.style.stroke**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.strokeWidth | — | Тип: Number<br><br>Ширина линии. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| datachange | Изменение данных. Поля данных:<br>• oldData — старые данные;<br>• newData — новые данные.<br>Унаследовано от IOverlay. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| emptinesschange | Изменение признака пустоты оверлея. Экземпляр класса Event.<br>Унаследовано от IOverlay. |
| geometrychange | Изменение геометрии. Поля данных:<br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия.<br>Унаследовано от IOverlay. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IOverlay. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры области охватывающей оверлей. Экземпляр класса Event.<br>Унаследовано от IOverlay. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getData() | Object | Возвращает объект данных оверлея.<br>Унаследован от IOverlay. |
| getGeometry() | IPixelGeometry | Возвращает текущую пиксельную геометрию.<br>Унаследован от IOverlay. |
| getMap() | Map \| null | Возвращает ссылку на текущую карту.<br>Унаследован от IOverlay. |
| getShape() | IShape \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.<br>Унаследован от IOverlay. |
| isEmpty() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от IOverlay. |
| setData(data) | | Задает данные оверлея. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IOverlay**. |
| **setGeometry**(geometry) | | Задает пиксельную геометрию оверлея.<br>Унаследован от **IOverlay**. |
| **setMap**(map) | | Задает карту, на которой должен отобразиться оверлей.<br>Унаследован от **IOverlay**. |

## overlay.storage

Статический объект.

Экземпляр класса **util.AsyncStorage**

Хранилище оверлеев. По умолчанию оверлеи не добавлены в package.full (стандартного набора модулей). При добавлении геообъекта на карту происходит загрузка необходимого оверлея. Для получения класса оверлея необходимо использовать метод require данного хранилища. По умолчанию в хранилище объявлены следующие ключи для асинхронного доступа:

- 'default#circle' — Оверлей круга **overlay.Circle**;
- 'default#polyline' — Оверлей линии **overlay.Polyline**;
- 'default#placemark' — Оверлей метки-изображения **overlay.Placemark**;
- 'default#polygon' — Оверлей многоугольника **overlay.Polygon**;
- 'default#rectangle' — Оверлей прямоугольника **overlay.Rectangle**;
- 'hotspot#circle' — Хотспотный оверлей круга **overlay.hotspot.Circle**;
- 'hotspot#polyline' — Хотспотный оверлей линии **overlay.hotspot.Polyline**;
- 'hotspot#placemark' — Хотспотный оверлей точки **overlay.hotspot.Placemark**;
- 'hotspot#polygon' — Хотспотный оверлей многоугольника **overlay.hotspot.Polygon**;
- 'hotspot#rectangle' — Хотспотный оверлей прямоугольника **overlay.hotspot.Rectangle**;
- 'html#balloon' — HTML-оверлей балуна **overlay.html.Balloon**;
- 'html#hint' — Простой HTML-оверлей **overlay.html.Hint**;
- 'html#placemark' — HTML-оверлей метки **overlay.html.Placemark**;
- 'html#rectangle' — HTML-оверлей прямоугольника **overlay.html.Rectangle**.

Методы

```
overlay.storage()
```

**Пример:**

```
ymaps.overlay.storage.require(['hotspot#circle'], function (HotspotOverlayClass)
{
    // Создаем экземпляр полученного класса.
    var overlay = new HotspotOverlayClass(
        new ymaps.geometry.Circle([30, 50], 10), {}, {}
    );
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |
| **define**(key[, depends, resolveCallback[, context]]) | **util.AsyncStorage** | Определение в хранилище асинхронного значения. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **isDefined**(key) | Boolean | Проверка доступности ключа в хранилище. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |
| **require**(keys[, successCallback[, errorCallback[, context]]]) | **vow.Promise** | Асинхронный запрос на получение значений из хранилища. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# pane

## pane.EventsPane

Расширяет **IEventPane**.

Пейн событий.

Конструктор | Поля | События | Методы

### Конструктор

```
pane.EventsPane(map, params)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Карта. |
| params * | — | Тип: Object<br><br>Параметры. |
| params.css | — | Тип: Object<br><br>CSS стили DOM-элемента пейна. |
| params.zIndex | 0 | Тип: Number<br><br>zIndex пейна. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

### События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| overflowchange | Изменение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Экземпляр IEvent.<br>Унаследовано от IPane. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| zindexchange | Изменение значения zIndex пейна. Экземпляр IEvent.<br>Унаследовано от IPane. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает пейн.<br>Унаследован от IPane. |
| getElement() | HTMLElement | Возвращает ссылку на DOM-контейнер пейна.<br>Унаследован от IPane. |
| getMap() | Map | Возвращает карту, которой принадлежит пейн.<br>Унаследован от IPane. |
| getOverflow() | String | Возвращает значение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Данный параметр может принимать одно из следующих строковых значений:<br>• "visible" — при выходе за пределы контейнера карты содержимое пейна остается видимым;<br>• "hidden" — область отображения содержимого пейна ограничена контейнером карты.<br>. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IPane**. |
| **getZIndex**() | Number | Возвращает zIndex пейна.<br>Унаследован от **IPane**. |

# pane.MovablePane

Расширяет **IContainerPane**.

Двигающийся пейн карты.

Конструктор | Поля | События | Методы

## Конструктор

```
pane.MovablePane(map, params)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br>Карта. |
| params * | — | Тип: Object<br>Параметры. |
| params.css | — | Тип: Object<br>CSS стили DOM-элемента пейна. |
| params.margin | 0 | Тип: Number<br>Дополнительный отступ от границ контейнера карты, расширяющий область отображения пейна. |
| params.overflow | "hidden" | Тип: String<br>Параметр определяющий видимость содержимого пейна при выходе за пределы контейнера карты. Данный параметр может принимать одно из следующих строковых значений:<br>• "visible" — при выходе за пределы контейнера карты содержимое пейна остается видимым;<br>• "hidden" — область отображения содержимого пейна ограничена контейнером карты.<br>. |
| params.zIndex | 0 | Тип: Number<br>zIndex пейна. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |

## События

| Имя | Описание |
|---|---|
| actionbegin | Начало движения пейна. Экземпляр IEvent.<br>Унаследовано от IContainerPane. |
| actionend | Окончание движения пейна. Экземпляр IEvent.<br>Унаследовано от IContainerPane. |
| clientpixelschange | Изменение локальной координатной системы пейна. Данное событие означает, что объекты, рассчитывающие свою позицию внутри пейна из глобальных пиксельных координат карты, должны пересчитать ее и обновить свое положение внутри DOM-элемента пейна. Экземпляр IEvent.<br>Унаследовано от IContainerPane. |
| overflowchange | Изменение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Экземпляр IEvent.<br>Унаследовано от IPane. |
| viewportchange | Изменение области показа пейна. Экземпляр IEvent.<br>Унаследовано от IContainerPane. |
| zindexchange | Изменение значения zIndex пейна. Экземпляр IEvent.<br>Унаследовано от IPane. |
| zoomchange | Изменение актуального уровня масштабирования пейна. Экземпляр IEvent.<br>Унаследовано от IContainerPane. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает пейн.<br>Унаследован от IPane. |
| fromClientPixels(clientPixelPoint) | Number[] | Преобразует клиентские пиксельные координаты в глобальные.<br>Унаследован от IPositioningContext. |
| getElement() | HTMLElement | Возвращает ссылку на DOM-контейнер пейна.<br>Унаследован от IPane. |
| getMap() | Map | Возвращает карту, которой принадлежит пейн.<br>Унаследован от IPane. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getOverflow**() | String | Возвращает значение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Данный параметр может принимать одно из следующих строковых значений:<br>• "visible" — при выходе за пределы контейнера карты содержимое пейна остается видимым;<br>• "hidden" — область отображения содержимого пейна ограничена контейнером карты.<br>.<br>Унаследован от **IPane**. |
| **getViewport**() | Number[][] | Возвращает область показа пейна в клиентских координатах.<br>Унаследован от **IContainerPane**. |
| **getZIndex**() | Number | Возвращает zIndex пейна.<br>Унаследован от **IPane**. |
| **getZoom**() | Number | Возвращает текущий уровень масштабирования, в котором работает контекст позиционирования.<br>Унаследован от **IPositioningContext**. |
| **toClientPixels**(globalPixelPoint) | Number[] | Преобразует глобальные пиксельные координаты в клиентские.<br>Унаследован от **IPositioningContext**. |

# pane.StaticPane

Расширяет **IContainerPane**.

Статичный пейн карты.

Конструктор | Поля | События | Методы

## Конструктор

```
pane.StaticPane(map, params)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| `map` * | — | Тип: **Map**<br>Карта. |
| `params` * | — | Тип: Object<br>Параметры. |
| `params.css` | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | CSS стили DOM-элемента пейна. |
| params.margin | 0 | Тип: Number<br><br>Дополнительный отступ от границ контейнера карты, расширяющий область отображения пейна. |
| params.overflow | "hidden" | Тип: String<br><br>Параметр определяющий видимость содержимого пейна при выходе за пределы контейнера карты. Данный параметр может принимать одно из следующих строковых значений:<br>• "visible" — при выходе за пределы контейнера карты содержимое пейна остается видимым;<br>• "hidden" — область отображения содержимого пейна ограничена контейнером карты.<br><br>. |
| params.zIndex | 0 | Тип: Number<br><br>zIndex пейна. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **actionbegin** | Начало движения пейна. Экземпляр **IEvent**.<br><br>Унаследовано от **IContainerPane**. |
| **actionend** | Окончание движения пейна. Экземпляр **IEvent**.<br><br>Унаследовано от **IContainerPane**. |
| **clientpixelschange** | Изменение локальной координатной системы пейна. Данное событие означает, что объекты, рассчитывающие свою позицию внутри пейна из глобальных пиксельных координат карты, должны пересчитать ее и обновить свое положение внутри DOM-элемента пейна. Экземпляр **IEvent**.<br><br>Унаследовано от **IContainerPane**. |
| **overflowchange** | Изменение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Экземпляр **IEvent**.<br><br>Унаследовано от **IPane**. |
| **viewportchange** | Изменение области показа пейна. Экземпляр **IEvent**.<br><br>Унаследовано от **IContainerPane**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **zindexchange** | Изменение значения zIndex пейна. Экземпляр **IEvent**.<br>Унаследовано от **IPane**. |
| **zoomchange** | Изменение актуального уровня масштабирования пейна. Экземпляр **IEvent**.<br>Унаследовано от **IContainerPane**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает пейн.<br>Унаследован от **IPane**. |
| **fromClientPixels**(clientPixelPoint) | Number[] | Преобразует клиентские пиксельные координаты в глобальные.<br>Унаследован от **IPositioningContext**. |
| **getElement**() | HTMLElement | Возвращает ссылку на DOM-контейнер пейна.<br>Унаследован от **IPane**. |
| **getMap**() | **Map** | Возвращает карту, которой принадлежит пейн.<br>Унаследован от **IPane**. |
| **getOverflow**() | String | Возвращает значение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Данный параметр может принимать одно из следующих строковых значений:<br>• "visible" — при выходе за пределы контейнера карты содержимое пейна остается видимым;<br>• "hidden" — область отображения содержимого пейна ограничена контейнером карты.<br>.<br>Унаследован от **IPane**. |
| **getViewport**() | Number[][] | Возвращает область показа пейна в клиентских координатах.<br>Унаследован от **IContainerPane**. |
| **getZIndex**() | Number | Возвращает zIndex пейна.<br>Унаследован от **IPane**. |
| **getZoom**() | Number | Возвращает текущий уровень масштабирования, в котором работает контекст позиционирования.<br>Унаследован от **IPositioningContext**. |
| **toClientPixels**(globalPixelPoint) | Number[] | Преобразует глобальные пиксельные координаты в клиентские.<br>Унаследован от **IPositioningContext**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# Placemark

Расширяет **GeoObject**.

Метка. Представляет собой геообъект с геометрией **geometry.Point**.

**См.: GeoObject geometry.Point**

Конструктор | Поля | События | Методы

## Конструктор

```
Placemark(geometry[, properties[, options]])
```

Создает экземпляр метки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: Number[] \| Object \| **IPointGeometry**<br><br>Координаты метки или хэш с описанием геометрии, или ссылка на объект точечной геометрии. |
| properties | — | Тип: Object \| **IDataManager**<br><br>Данные метки. Могут задаваться как экземпляр класса, реализующего интерфейс **IDataManager**, либо в виде хэша. При значениях опций по умолчанию геообъектом трактуются следующие поля данных:<br>• iconContent — содержимое иконки геообъекта;<br>• hintContent — содержимое всплывающей подсказки геообъекта;<br>• balloonContent — содержимое балуна геообъекта;<br>• balloonContentHeader — содержимое заголовка балуна геообъекта;<br>• balloonContentBody — содержимое основой части балуна геообъекта;<br>• balloonContentFooter — содержимое нижней части балуна геообъекта.<br><br>Поле balloonContent является кратким обозначением для поля balloonContentBody, но при одновременном задании balloonContentBody более приоритетен. Также вы можете дополнить данные метки своими собственными полями и использовать их везде, где это возможно. Например в макете метки или макете балуна. |
| options | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опции метки.<br><br>**Для изменения стиля и цвета иконки используются следующие опции:**<br><br>• `preset` — ключ предустановленных стилей метки. Список доступных ключей содержится в описании **option.presetStorage**. Опцию `preset` можно задавать совместно с опцией `iconColor`, только если она принимает одно из следующих значений:<br>  • 'islands#icon';<br>  • 'islands#dotIcon';<br>  • 'islands#circleIcon';<br>  • 'islands#circleDotIcon'/<br>• `iconColor` — цвет иконки метки. Можно задавать в любом формате, допустимом в CSS (например, по названию или в формате RGB).Эта опция применяется для стандартных иконок в браузерах, поддерживающих SVG.<br><br>**Примечание:**<br>В IE8 данная опция не работает.<br><br>**Если требуется задать собственный макет иконки, то необходимо указать следующие опции:**<br><br>• iconLayout — макет иконки. Тип: конструктор объекта с интерфейсом ILayout или его ключ в хранилище. Список доступных макетов:<br>  • 'default#image' — пользовательское изображение иконки;<br>  • 'default#imageWithContent' — пользовательское изображение иконки с содержимым;<br>• дополнительные опции классов "**layout.Image**" и "**layout.ImageWithContent** с префиксом icon.<br><br>**Примечание:**<br>Для задания макета тени иконки используется тот же набор опций, но с префиксом iconShadow. Например, iconShadowLayout. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | **Также через класс Placemark можно задать опции отдельных объектов:**<br>• Опции балуна метки с префиксом `balloon`.<br>• Опции всплывающей подсказки метки с префиксом `hint`.<br>• Опции редактора геометрии многоульника с префиксом `editor`. Смотрите описание класса **geometryEditor.Point**.<br>• Опции геометрии могут задаваться без префикса. Смотрите описание класса **IGeometry** геометрии **geometry.Point**.<br>Ниже приведены остальные опции данного класса. |
| `options.cursor` | "pointer" | Тип: String<br>Вид курсора над меткой. |
| `options.draggable` | false | Тип: Boolean<br>Определяет возможность перетаскивания метки. |
| `options.hasBalloon` | true | Тип: Boolean<br>Определяет наличие поля balloon у метки. |
| `options.hasHint` | true | Тип: Boolean<br>Определяет наличие поля hint у метки. |
| `options.hideIconOnBalloonOpen` | true | Тип: Boolean<br>Скрывать метку при открытии балуна. |
| `options.iconOffset` | — | Тип: Number[]<br>Пиксельное смещение иконки относительно заданной позиции. |
| `options.iconShape` | — | Тип: **IGeometryJson** \| null<br>Фигура активной области метки. Задается в виде JSON-описания пиксельной геометрии иконки. Эту опцию нужно использовать при создании своих HTML макетов. Координаты геометрии фигуры отсчитываются от точки привязки. |
| `options.interactiveZIndex` | true | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Включает режим автоматического изменения z-index метки в зависимости от ее состояния. |
| options.interactivityModel | "default#geoObject" | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.openBalloonOnClick | true | Тип: Boolean<br><br>Определяет, показывать ли балун при щелчке на метке. |
| options.openEmptyBalloon | false | Тип: Boolean<br><br>Определяет, показывать ли пустой балун при щелчке на метке. |
| options.openEmptyHint | false | Тип: Boolean<br><br>Определяет, показывать ли пустой хинт при наведении указателя мыши на метку. |
| options.openHintOnHover | true | Тип: Boolean<br><br>Определяет, показывать ли хинт при наведении указателя мыши на метку. |
| options.pane | "places" | Тип: String<br><br>Ключ пейна, в который помещается оверлей метки. |
| options.pointOverlay | "default#placemark" | Тип: String \| Function<br><br>Ключ-идентификатор из хранилища **overlay.storage** или класс оверлея. Функция-генератор принимает три параметра:<br>• geometry: **IPixelPointGeometry** — непосредственно пиксельная геометрия;<br>• data: **IDataManager** или Object — данные оверлея;<br>• options: Object — опции оверлея.<br><br>И возвращает **vow.Promise**. |
| options.syncOverlayInit | false | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Включает синхронное добавление оверлея на карту. По умолчанию добавление оверлея осуществляется асинхронно, что позволяет предотвратить зависания браузера при добавлении на карту большого числа геообъектов. Однако, асинхронное добавление не позволяет получать доступ к оверлею сразу после добавления метки на карту. |
| options.useMapMarginInDragging | true | Тип: Boolean<br><br>При перетаскивании объекта к краю карты происходит автоматическое изменение центра карты. Нужно ли учитывать отступы карты при автоматическом смещении центра карты **map.margin.Manager**. |
| options.visible | true | Тип: Boolean<br><br>Определяет видимость метки. |
| options.zIndex | — | Тип: Number<br><br>z-index иконки метки в обычном состоянии. Наименее приоритетный. |
| options.zIndexActive | — | Тип: Number<br><br>z-index иконки метки с открытым балуном. Наиболее приоритетный. |
| options.zIndexDrag | — | Тип: Number<br><br>z-index иконки метки при перетаскивании. |
| options.zIndexHover | — | Тип: Number<br><br>z-index иконки метки при наведении на него указателя мыши. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Создаем метку.
var placemark = new ymaps.Placemark([48, 40], {
    balloonContent: '<img src="http://img-fotki.yandex.ru/get/
6114/82599242.2d6/0_88b97_ec425cf5_M" />',
    iconContent: "Азербайджан"
}, {
    preset: "islands#yellowStretchyIcon",
    // Отключаем кнопку закрытия балуна.
    balloonCloseButton: false,
    // Балун будем открывать и закрывать кликом по иконке метки.
    hideIconOnBalloonOpen: false
});
geoMap.geoObjects.add(placemark);
```

**2.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
var placemark = new ymaps.Placemark([88, 43], {}, {
    // Задаем стиль метки (метка в виде круга).
    preset: "islands#dotCircleIcon",
    // Задаем цвет метки (в формате RGB).
    iconColor: '#ff0000'
});
geoMap.geoObjects.add(placemark);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| balloon | geoObject.Balloon | Балун геообъекта.<br>Унаследовано от GeoObject. |
| editor | geometryEditor.Point | Редактор геометрии "Точка". |
| events | event.Manager | Менеджер событий.<br>Унаследовано от GeoObject. |
| geometry | geometry.Point | Геометрия типа "Точка". |
| hint | geoObject.Hint | Хинт геообъекта.<br>Унаследовано от GeoObject. |
| options | option.Manager | Менеджер опций геообъекта.<br>Унаследовано от GeoObject. |
| properties | data.Manager | Менеджер данных геообъекта.<br>Унаследовано от GeoObject. |
| state | data.Manager | Состояние геообъекта. Определяется следующими полями:<br>• active: Boolean — признак того, что на геообъекте открыт балун;<br>• hover: Boolean — признак того, что в данный момент на геообъект наведен указатель мыши;<br>• drag: Boolean — признак того, что геообъект перетаскивается.<br>Унаследовано от GeoObject. |

## События

| Имя | Описание |
|---|---|
| balloonclose | Закрытие балуна. Экземпляр класса Event.<br>Унаследовано от GeoObject. |
| balloonopen | Открытие балуна на геообъекте. Экземпляр класса Event.<br>Унаследовано от GeoObject. |
| beforedrag | Событие, предшествующее событию drag. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие drag будет отменено.<br>Унаследовано от **GeoObject**. |
| **beforedragstart** | Событие, предшествующее событию dragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие dragstart, будет отменено.<br>Унаследовано от **GeoObject**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **drag** | Перетаскивание геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragend** | Завершение перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **dragstart** | Начало перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **editorstatechange** | Изменение состояния редактора геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent — оригинальное событие редактора геометрии.<br>Унаследовано от **GeoObject**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **hintopen** | Открытие хинта на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br><br>Поля данных:<br><br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br><br>Унаследовано от **IChild**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание полей

### editor

```
{geometryEditor.Point} editor
```

Редактор геометрии "Точка".

### geometry

```
{geometry.Point} geometry
```

Геометрия типа "Точка".

# Polygon

Расширяет **GeoObject**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Многоугольник. Представляет собой геообъект с геометрией **geometry.Polygon**.

**См.: GeoObject geometry.Polygon**

Конструктор | Поля | События | Методы

## Конструктор

```
Polygon(geometry[, properties[, options]])
```

Создает экземпляр многоугольника.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: Number[][][] \| Object \| **IPolygonGeometry**<br><br>Координаты вершин ломаных, определяющих внешнюю и внутренние границы многоугольника, хэш-объект с параметрами геометрии или ссылка на объект геометрии. Внутренняя граница может отсутствовать. |
| properties | — | Тип: Object \| **IDataManager**<br><br>Данные многоугольника. Могут задаваться как экземпляр класса, реализующего интерфейс **IDataManager**, либо в виде хэша. При значениях опций по умолчанию геообъектом трактуются следующие поля данных:<br>• hintContent — содержимое всплывающей подсказки многоугольника;<br>• balloonContent — содержимое балуна многоугольника;<br>• balloonContentHeader — содержимое заголовка балуна многоугольника;<br>• balloonContentBody — содержимое основой части балуна многоугольника;<br>• balloonContentFooter — содержимое нижней части балуна многоугольника.<br><br>Поле balloonContent является кратким обозначением для поля balloonContentBody, но при одновременном задании balloonContentBody более приоритетен. Также вы можете дополнить данные многоугольника своими собственными полями и использовать их везде, где это возможно. Например в макете балуна многоугольника. |
| options | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опции многоугольника. Используя этот параметр можно задавать как опции самого многоульника, так и опции его составных частей:<br>• Опции балуна многоульника с префиксом `balloon`.<br>• Опции всплывающей подсказки многоульника с префиксом `hint`.<br>• Опции редактора геометрии многоульника с префиксом `editor`. Смотрите описание класса **geometryEditor.Polygon**.<br>• Опции геометрии могут задаваться без префикса. Смотрите описание класса **IGeometry** геометрии **geometry.Polygon**. |
| `options.cursor` | "pointer" | Тип: String<br><br>Вид курсора над многоугольником. |
| `options.draggable` | false | Тип: Boolean<br><br>Определяет возможность перетаскивания многоугольника. |
| `options.fill` | true | Тип: Boolean<br><br>Наличие заливки фигуры. |
| `options.fillColor` | "0066ff99" | Тип: String<br><br>Цвет заливки. |
| `options.fillImageHref` | — | Тип: String<br><br>Фоновое изображение. При включении данной опции в режиме заливки **stretch** значение fillColor игнорируется. |
| `options.fillMethod` | 'stretch' | Тип: String<br><br>Тип заливки фоном. Может принимать одно из двух значений:<br>• stretch — Фоновое изображение растягивается по размеру оверлея.<br>• tile — Фоновое изображение повторяется без изменений размера. Аналог background-repeat в css. Можно использовать для заливки фигуры неким шаблоном. |
| `options.fillOpacity` | 1 | Тип: Number<br><br>Прозрачность заливки. |
| `options.hasBalloon` | true | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Определяет наличие поля balloon у многоугольника. |
| options.hasHint | true | Тип: Boolean<br><br>Определяет наличие поля hint у многоугольника. |
| options.interactiveZIndex | false | Тип: Boolean<br><br>Включает режим автоматического изменения z-index полигона в зависимости от его состояния. |
| options.interactivityModel | "default#geoObject" | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.opacity | 1 | Тип: Number<br><br>Прозрачность. |
| options.openBalloonOnClick | true | Тип: Boolean<br><br>Определяет, показывать ли балун при щелчке на многоугольнике. |
| options.openEmptyBalloon | false | Тип: Boolean<br><br>Определяет показывать ли пустой балун при щелчке на многоугольнике. |
| options.openEmptyHint | false | Тип: Boolean<br><br>Определяет показывать ли пустой хинт при наведении указателя мыши на многоугольник. |
| options.openHintOnHover | true | Тип: Boolean<br><br>Определяет, показывать ли хинт при наведении указателя мыши на многоугольник. |
| options.outline | true | Тип: Boolean<br><br>Наличие обводки фигуры. |
| options.pane | "areas" | Тип: String<br><br>Ключ пейна, в который помещается оверлей многоугольника. |
| options.polygonOverlay | "default#polygon" | Тип: String \| Function |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Ключ-идентификатор из хранилища **overlay.storage** или класс оверлея. Функция-генератор принимает три параметра:<br>• geometry: **IPixelPolygonGeometry** — непосредственно пиксельная геометрия;<br>• data: **IDataManager** или Object — данные оверлея;<br>• options: Object — опции оверлея.<br>И возвращает **vow.Promise**. |
| options.strokeColor | "0066ffff" | Тип: String \| String[]<br>Цвет линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeOpacity | 1 | Тип: Number \| Number[]<br>Прозрачность линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeStyle | — | Тип: String \| Object \| String[] \| Object[]<br>Стиль линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeWidth | 1 | Тип: Number \| Number[]<br>Толщина линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.syncOverlayInit | false | Тип: Boolean<br>Включает синхронное добавление оверлея на карту. По умолчанию добавление оверлея осуществляется асинхронно, что позволяет предотвратить зависания браузера при добавлении на карту большого числа геообъектов. Однако, асинхронное добавление не позволяет получать доступ к оверлею сразу после добавления многоугольника на карту. |
| options.useMapMarginInDragging | true | Тип: Boolean<br>При перетаскивании объекта к краю карты происходит автоматическое изменение центра карты. Нужно ли учитывать отступы карты при автоматическом смещении центра карты **map.margin.Manager**. |
| options.visible | true | Тип: Boolean<br>Определяет видимость многоугольника. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options.zIndex | — | Тип: Number<br><br>z-index многоугольника в обычном состоянии. Наименее приоритетный. |
| options.zIndexActive | — | Тип: Number<br><br>z-index многоугольника с открытым балуном. Наиболее приоритетный. |
| options.zIndexDrag | — | Тип: Number<br><br>z-index многоугольника при перетаскивании. |
| options.zIndexHover | — | Тип: Number<br><br>z-index многоугольника при наведении на него указателя мыши. |

* Обязательный параметр/опция.

**Пример:**

```
var polygon = new ymaps.Polygon([
    // Координаты внешнего контура.
    [[-80, 60], [-90, 50], [-60, 40], [-80, 60]],
    // Координаты внутреннего контура.
    [[-90, 80], [-90, 30], [-20, 40], [-90, 80]]
], {
    hintContent: "Многоугольник"
}, {
    fillColor: '#6699ff',
    // Делаем полигон прозрачным для событий карты.
    interactivityModel: 'default#transparent',
    strokeWidth: 8,
    opacity: 0.5
});
myMap.geoObjects.add(polygon);
myMap.setBounds(polygon.geometry.getBounds());
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **balloon** | **geoObject.Balloon** | Балун геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **editor** | **geometryEditor.Polygon** | Редактор геометрии "Многоугольник". |
| **events** | **event.Manager** | Менеджер событий.<br><br>Унаследовано от **GeoObject**. |
| **geometry** | **geometry.Polygon** | Геометрия типа "Многоугольник". |
| **hint** | **geoObject.Hint** | Хинт геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **options** | **option.Manager** | Менеджер опций геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **properties** | **data.Manager** | Менеджер данных геообъекта. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **GeoObject**. |
| **state** | **data.Manager** | Состояние геообъекта. Определяется следующими полями:<br>• active: Boolean — признак того, что на геообъекте открыт балун;<br>• hover: Boolean — признак того, что в данный момент на геообъект наведен указатель мыши;<br>• drag: Boolean — признак того, что геообъект перетаскивается.<br>Унаследовано от **GeoObject**. |

## События

| Имя | Описание |
|---|---|
| **balloonclose** | Закрытие балуна. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **balloonopen** | Открытие балуна на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **beforedrag** | Событие, предшествующее событию drag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие drag будет отменено.<br>Унаследовано от **GeoObject**. |
| **beforedragstart** | Событие, предшествующее событию dragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие dragstart, будет отменено.<br>Унаследовано от **GeoObject**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **drag** | Перетаскивание геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragend** | Завершение перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragstart** | Начало перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **editorstatechange** | Изменение состояния редактора геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent — оригинальное событие редактора геометрии.<br>Унаследовано от **GeoObject**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **hintopen** | Открытие хинта на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание полей

### editor

```
{geometryEditor.Polygon} editor
```

Редактор геометрии "Многоугольник".

### geometry

```
{geometry.Polygon} geometry
```

Геометрия типа "Многоугольник".

# Polyline

Расширяет GeoObject.

Ломаная линия. Представляет собой геообъект с геометрией geometry.LineString.

**См.:** GeoObject geometry.LineString

Конструктор | Поля | События | Методы

## Конструктор

```
Polyline(geometry[, properties[, options]])
```

Создает экземпляр ломаной.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: Number[][] \| Object \| ILineStringGeometry<br>Координаты вершин, хэш-объект с параметрами геометрии или ссылка на объект геометрии ломаной линии. |
| properties | — | Тип: Object \| IDataManager |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Данные ломаной. Могут задаваться как экземпляр класса, реализующего интерфейс **IDataManager**, либо в виде хэша. При значениях опций по умолчанию геообъектом трактуются следующие поля данных:<br>• hintContent — содержимое всплывающей подсказки ломаной;<br>• balloonContent — содержимое балуна ломаной;<br>• balloonContentHeader — содержимое заголовка балуна ломаной;<br>• balloonContentBody — содержимое основой части балуна ломаной;<br>• balloonContentFooter — содержимое нижней части балуна ломаной.<br>Поле balloonContent является кратким обозначением для поля balloonContentBody, но при одновременном задании balloonContentBody более приоритетен. Также вы можете дополнить данные ломаной своими собственными полями и использовать их, например, в макете хинта. |
| `options` | — | Тип: Object<br>Опции ломаной. Используя этот параметр можно задавать как опции самой ломаной, так и опции ее составных частей:<br>• Опции балуна ломаной с префиксом `balloon`.<br>• Опции всплывающей подсказки ломаной с префиксом `hint`.<br>• Опции редактора геометрии ломаной с префиксом `editor`. Смотрите описание класса **geometryEditor.LineString**.<br>• Опции геометрии могут задаваться без префикса. Смотрите описание класса **IGeometry** геометрии **geometry.LineString**. |
| `options.cursor` | "pointer" | Тип: String<br>Вид курсора над ломаной. |
| `options.draggable` | false | Тип: Boolean<br>Определяет возможность перетаскивания ломаной. |
| `options.hasBalloon` | true | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Определяет наличие поля balloon у ломаной. |
| options.hasHint | true | Тип: Boolean<br><br>Определяет наличие поля hint у ломаной. |
| options.interactiveZIndex | false | Тип: Boolean<br><br>Включает режим автоматического изменения z-index ломаной в зависимости от ее состояния. |
| options.interactivityModel | "default#geoObject" | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.lineStringOverlay | "default#polyline" | Тип: String \| Function<br><br>Ключ-идентификатор из хранилища **overlay.storage** или класс оверлея. Функция-генератор принимает три параметра:<br>• geometry: **IPixelLineStringGeometry** — непосредственно пиксельная геометрия;<br>• data: **IDataManager** или Object — данные оверлея;<br>• options: Object — опции оверлея.<br><br>И возвращает **vow.Promise**. |
| options.opacity | 1 | Тип: Number<br><br>Прозрачность. |
| options.openBalloonOnClick | true | Тип: Boolean<br><br>Определяет, показывать ли балун при щелчке на ломаной. |
| options.openEmptyBalloon | false | Тип: Boolean<br><br>Определяет, показывать ли пустой балун при щелчке на ломаной. |
| options.openEmptyHint | false | Тип: Boolean<br><br>Определяет, показывать ли пустой хинт при наведении указателя мыши на ломаную. |
| options.openHintOnHover | true | Тип: Boolean<br><br>Определяет, показывать ли хинт при наведении указателя мыши на ломаную. |
| options.pane | "areas" | Тип: **IPane** \| String |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Ключ пейна, в который помещается оверлей ломаной. |
| options.strokeColor | "0066ffff" | Тип: String \| String[]<br><br>Цвет линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeOpacity | 1 | Тип: Number \| Number[]<br><br>Прозрачность линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeStyle | — | Тип: String \| Object \| String[] \| Object[]<br><br>Стиль линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeWidth | 1 | Тип: Number \| Number[]<br><br>Толщина линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.syncOverlayInit | false | Тип: Boolean<br><br>Включает синхронное добавление оверлея на карту. По умолчанию добавление оверлея осуществляется асинхронно, что позволяет предотвратить зависания браузера при добавлении на карту большого числа ломаных. Однако, асинхронное добавление не позволяет получать доступ к оверлею сразу после добавления ломаной на карту. |
| options.useMapMarginInDragging | true | Тип: Boolean<br><br>При перетаскивании объекта к краю карты происходит автоматическое изменение центра карты. Нужно ли учитывать отступы карты при автоматическом смещении центра карты **map.margin.Manager**. |
| options.visible | true | Тип: Boolean<br><br>Определяет видимость ломаной. |
| options.zIndex | — | Тип: Number<br><br>z-index ломаной в обычном состоянии. Наименее приоритетный. |
| options.zIndexActive | — | Тип: Number<br><br>z-index ломаной с открытым балуном. Наиболее приоритетный. |
| options.zIndexDrag | — | Тип: Number |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | z-index ломаной при перетаскивании. |
| options.zIndexHover | — | Тип: Number<br><br>z-index ломаной при наведении на него указателя мыши. |

* Обязательный параметр/опция.

**Пример:**

```
// Создаем ломаную линию.
var polyline = new ymaps.Polyline([
    [-80, 60], [-90, 50], [-60, 40], [-80, 60]
], {
    hintContent: "Ломаная линия"
}, {
    draggable: true,
    strokeColor: '#000000',
    strokeWidth: 4,
    // Первой цифрой задаем длину штриха. Второй цифрой задаем длину разрыва.
    strokeStyle: '1 5'
});
// Добавляем линию на карту.
myMap.geoObjects.add(polyline);
// Устанавливаем карте границы линии.
myMap.setBounds(polyline.geometry.getBounds());
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **balloon** | **geoObject.Balloon** | Балун геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **editor** | **geometryEditor.LineString** | Редактор геометрии "Ломаная линия". |
| **events** | **event.Manager** | Менеджер событий.<br><br>Унаследовано от **GeoObject**. |
| **geometry** | **geometry.LineString** | Геометрия типа "Ломаная линия". |
| **hint** | **geoObject.Hint** | Хинт геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **options** | **option.Manager** | Менеджер опций геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **properties** | **data.Manager** | Менеджер данных геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **state** | **data.Manager** | Состояние геообъекта. Определяется следующими полями:<br><br>• active: Boolean — признак того, что на геообъекте открыт балун;<br>• hover: Boolean — признак того, что в данный момент на геообъект наведен указатель мыши;<br>• drag: Boolean — признак того, что геообъект перетаскивается.<br><br>Унаследовано от **GeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| **balloonclose** | Закрытие балуна. Экземпляр класса **Event**.<br><br>Унаследовано от **GeoObject**. |
| **balloonopen** | Открытие балуна на геообъекте. Экземпляр класса **Event**.<br><br>Унаследовано от **GeoObject**. |
| **beforedrag** | Событие, предшествующее событию drag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br><br>Имена методов, доступных через метод Event.callMethod:<br><br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br><br>Если у данного события вызвать метод Event.preventDefault, то последующее событие drag будет отменено.<br><br>Унаследовано от **GeoObject**. |
| **beforedragstart** | Событие, предшествующее событию dragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br><br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие dragstart, будет отменено.<br><br>Унаследовано от **GeoObject**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **drag** | Перетаскивание геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragend** | Завершение перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragstart** | Начало перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **editorstatechange** | Изменение состояния редактора геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent — оригинальное событие редактора геометрии.<br>Унаследовано от **GeoObject**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **hintopen** | Открытие хинта на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Описание полей

#### editor

```
{geometryEditor.LineString} editor
```

Редактор геометрии "Ломаная линия".

#### geometry

```
{geometry.LineString} geometry
```

Геометрия типа "Ломаная линия".

## Popup

Расширяет **IPopup**.

Класс для создания инфо-объекта.

Конструктор | Поля | События | Методы

### Конструктор

```
Popup(map[, options])
```

Инфо-объект.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map * | — | Тип: **Map**<br><br>Ссылка на карту. |
| options | — | Тип: Object<br><br>Опции. |
| options.closeTimeout | 700 | Тип: Number<br><br>Задержка закрытия (в мс). |
| options.interactivityModel | — | Тип: String<br><br>Ключ модели интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.openTimeout | 150 | Тип: Number<br><br>Задержка открытия (в мс). |
| options.pane | — | Тип: **IPane** \| String<br><br>Пейн инфо-объекта. Список доступных по умолчанию пейнов смотрите в описании **map.pane.Manager**. |
| options.projection | — | Тип: **IProjection** |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Проекция координат в глобальные пиксели. |
| options.zIndex | — | Тип: String<br><br>z-index инфо-объекта. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| close | Закрытие инфо-объекта.<br>Унаследовано от IPopup. |
| open | Открытие инфо-объекта.<br>Унаследовано от IPopup. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| close([force]) | vow.Promise | Закрывает инфо-объект.<br>Унаследован от IPopup. |
| getData() | | Возвращает данные инфо-объекта.<br>Унаследован от IPopup. |
| getOverlay() | vow.Promise | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от IPopup. |
| getOverlaySync() | IOverlay | Возвращает оверлей, если тот существует.<br>Унаследован от IPopup. |
| getPosition() | | Возвращает координаты инфо-объекта.<br>Унаследован от IPopup. |
| isOpen() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от IPopup. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **open**([position[, data]]) | **vow.Promise** | Открывает инфо-объект в указанной позиции. Если инфо-объект уже открыт, перемещает его в указанную точку. Формат и суть координат определяется проекцией **IProjection**, которая содержится в опциях.<br><br>Унаследован от **IPopup**. |
| **setData**(data) | **vow.Promise** | Задаёт инфо-объекту новые данные.<br><br>Унаследован от **IPopup**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br><br>Унаследован от **IPopup**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# projection

## projection.Cartesian

Расширяет **IProjection**.

Декартова проекция прямоугольной области. Учитывает параметр coordorder загрузки API — при значении 'latlong' в массиве координат точки на первом месте должен стоять y, а на втором x.

Конструктор | Методы

### Конструктор

```
projection.Cartesian(bounds[, cycled[, scale]])
```

Создает проекцию прямоугольной координатной области в пиксели. Размер области в пикселях всегда NxN, где N = 256 * 2^zoom.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds * | — | Тип: Object[]<br><br>Массив из двух точек — координат левого нижнего и правого верхнего углов прямоугольной координатной области. |
| cycled | [false, false] | Тип: Boolean[]<br><br>Массив из признаков зацикленности карты по x и y. |
| scale | 1 | Тип: Number \| Number[]<br><br>Масштаб одного деления на оси. Может быть числом или парой чисел для каждой из осей. |

* Обязательный параметр/опция.

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **fromGlobalPixels**(globalPixelPoint, zoom) | Number[] | Преобразует пиксельные координаты на указанном уровне масштабирования в координаты проекции.<br><br>Унаследован от **IProjection**. |
| **getCoordSystem**() | **ICoordSystem** | Возвращает координатную систему, используемую проекцией.<br><br>Унаследован от **IProjection**. |
| **isCycled**() | Boolean[] | Показатель зацикленности проекции.<br><br>Унаследован от **IProjection**. |
| **toGlobalPixels**(coordPoint, zoom) | Number[] | Преобразует координаты проекции в глобальные пиксельные на указанном уровне масштабирования. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IProjection**. |

## projection.sphericalMercator

Статический объект.

Экземпляр класса **IProjection**

Меркаторовская проекция на сферу. Используется многими картографическими сервисами, в частности, OpenStreetMap.

Методы

```
projection.sphericalMercator()
```

**Пример:**

```
// Создадим карту в сферической проекции Меркатора
var map = new ymaps.Map('YMapsID', {
    center: [55, 37],
    zoom: 6
}, {
    projection: ymaps.projection.sphericalMercator
});
map.layers.add(new ymaps.Layer('http://tile.openstreetmap.org/%z/%x/%y.png'));
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **fromGlobalPixels**(globalPixelPoint, zoom) | Number[] | Преобразует пиксельные координаты на указанном уровне масштабирования в координаты проекции. |
| **getCoordSystem**() | **ICoordSystem** | Возвращает координатную систему, используемую проекцией. |
| **isCycled**() | Boolean[] | Показатель зацикленности проекции. |
| **toGlobalPixels**(coordPoint, zoom) | Number[] | Преобразует координаты проекции в глобальные пиксельные на указанном уровне масштабирования. |

## projection.wgs84Mercator

Статический объект.

Экземпляр класса **IProjection**

Меркаторовская проекция на референсный эллипсоид wgs84. Используется Яндекс.Картами по умолчанию.

Методы

```
projection.wgs84Mercator()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **fromGlobalPixels**(globalPixelPoint, zoom) | Number[] | Преобразует пиксельные координаты на указанном уровне масштабирования в координаты проекции. |
| **getCoordSystem**() | **ICoordSystem** | Возвращает координатную систему, используемую проекцией. |
| **isCycled**() | Boolean[] | Показатель зацикленности проекции. |
| **toGlobalPixels**(coordPoint, zoom) | Number[] | Преобразует координаты проекции в глобальные пиксельные на указанном уровне масштабирования. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# ready

Статическая функция.

Выполняет переданную функцию, когда API и DOM готовы к использованию.

**Возвращает:** Объект-обещание, которое подтверждается пространством имен API, либо отклоняется, если при загрузке произошла ошибка. (тип **vow.Promise**).

```
ready([successCallback[, errorCallback[, context]]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| successCallback | — | Тип: Function \| Object<br><br>Функция, которая будет вызвана при успешной загрузке и инициализации API и DOM, или объект с параметрами, если используется расширенный синтаксис.<br><br>Доступный список параметров:<br>• require — массив дополнительных модулей, которые должны быть загружены вместе с API;<br>• successCallback — функция, которая будет вызвана при успешной загрузке API;<br>• errorCallback — функция, которая будет вызварана в случае ошибки;<br>• context — контекст исполнения функций.<br><br>Все параметры являются необязательными.<br><br>В successCallback будет передано пространство имен API. |
| errorCallback | — | Тип: Function<br><br>Функция, которая будет вызвана если произошла ошибка при инициализации. В функцию будет передана ошибка. |
| context | — | Тип: Object<br><br>Контекст исполнения функции. |

**Примеры:**

**1.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
<!DOCTYPE html>
<html>
<head>
    <title>Пример</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <script src="http://api-maps.yandex.ru/2.1/?lang=ru-RU" type="text/
javascript"></script>
    <script type="text/javascript">
    ymaps.ready(function () {
        var map = new ymaps.Map('map', {
            center: [55.7, 37.6],
            zoom: 10
        });
        // ...
    });
    </script>
</head>
<body>
    <div id="map" style="width: 500px; height: 500px;"></div>
</body>
</html>
```

**2.**

```
// Пример использования расширенного синтаксиса.
ymaps.ready({
    // successCallback будет вызван, когда произойдет загрузка API и модуля
"myModule1".
    require: ['myModule1'],
    successCallback: function (ym) {
        var map = new ymaps.Map('map', {
            center: [55.7, 37.6],
            zoom: 10
        });
        var obj = new ymaps.myModule1();
        // ...
    }
})
```

# Rectangle

Расширяет **GeoObject**.

Прямоугольник. Представляет собой геообъект с геометрией **geometry.Rectangle**.

**См.: GeoObject geometry.Rectangle**

Конструктор | Поля | События | Методы

## Конструктор

```
Rectangle(geometry[, properties[, options]])
```

Создает экземпляр прямоугольника.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: Number[][] \| Object \| **IRectangleGeometry**<br><br>Координаты двух противоположных углов, хэш-объект с параметрами геометрии или ссылка на объект геометрии прямоугольника. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| properties | — | Тип: Object \| **IDataManager**<br><br>Данные прямоугольника. Могут задаваться как экземпляр класса, реализующего интерфейс **IDataManager**, либо в виде хэша. При значениях опций по умолчанию геообъектом трактуются следующие поля данных:<br>• hintContent — содержимое всплывающей подсказки прямоугольника;<br>• balloonContent — содержимое балуна прямоугольника;<br>• balloonContentHeader — содержимое заголовка балуна прямоугольника;<br>• balloonContentBody — содержимое основной части балуна прямоугольника;<br>• balloonContentFooter — содержимое нижней части балуна прямоугольника.<br><br>Поле balloonContent является кратким обозначением для поля balloonContentBody, но при одновременном задании balloonContentBody более приоритетен. Также вы можете дополнить данные прямоугольника своими собственными полями и использовать их, например, в макете балуна. |
| options | — | Тип: Object<br><br>Опции прямоугольника. Используя этот параметр можно задавать как опции самого прямоугольника, так и опции его составных частей:<br>• Опции балуна прямоугольника с префиксом `balloon`.<br>• Опции всплывающей подсказки прямоугольника с префиксом `hint`.<br>• Опции геометрии могут задаваться без префикса. Смотрите описание класса **IGeometry** геометрии **geometry.Rectangle**. |
| options.cursor | "pointer" | Тип: String<br><br>Вид курсора над прямоугольником. |
| options.draggable | false | Тип: Boolean<br><br>Определяет возможность перетаскивания прямоугольника. |
| options.fill | true | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Наличие заливки фигуры. |
| options.fillColor | "0066ff99" | Тип: String<br><br>Цвет заливки. |
| options.fillImageHref | — | Тип: String<br><br>Фоновое изображение. При включении данной опции в режиме заливки **stretch** значение fillColor игнорируется. |
| options.fillMethod | 'stretch' | Тип: String<br><br>Тип заливки фоном. Может принимать одно из двух значений:<br>• stretch — Фоновое изображение растягивается по размеру оверлея.<br>• tile — Фоновое изображение повторяется без изменений размера. Аналог background-repeat в css. Можно использовать для заливки фигуры неким шаблоном. |
| options.fillOpacity | 1 | Тип: Number<br><br>Прозрачность заливки. |
| options.hasBalloon | true | Тип: Boolean<br><br>Определяет наличие поля balloon у прямоугольника. |
| options.hasHint | true | Тип: Boolean<br><br>Определяет наличие поля hint у прямоугольника. |
| options.interactiveZIndex | false | Тип: Boolean<br><br>Включает режим автоматического изменения z-index прямоугольника в зависимости от его состояния. |
| options.interactivityModel | "default#geoObject" | Тип: String<br><br>Модель интерактивности. Доступные ключи и их значения перечислены в описании **interactivityModel.storage**. |
| options.opacity | 1 | Тип: Number<br><br>Прозрачность. |
| options.openBalloonOnClick | true | Тип: Boolean<br><br>Определяет, показывать ли балун при щелчке на прямоугольнике. |
| options.openEmptyBalloon | false | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Определяет, показывать ли пустой балун при щелчке на прямоугольнике. |
| options.openEmptyHint | false | Тип: Boolean<br><br>Определяет, показывать ли пустой хинт при наведении указателя мыши на прямоугольник. |
| options.openHintOnHover | true | Тип: Boolean<br><br>Определяет, показывать ли хинт при наведении указателя мыши на прямоугольник. |
| options.outline | true | Тип: Boolean<br><br>Наличие обводки фигуры. |
| options.pane | "places" | Тип: String<br><br>Ключ пейна, в который помещается оверлей прямоугольника. |
| options.rectangleOverlay | "default#rectangle" | Тип: String \| Function<br><br>Ключ-идентификатор из хранилища **overlay.storage** или класс оверлея. Функция-генератор принимает три параметра:<br>• geometry: **IPixelRectangleGeometry** — непосредственно пиксельная геометрия;<br>• data: **IDataManager** или Object — данные оверлея;<br>• options: Object — опции оверлея.<br><br>И возвращает **vow.Promise**. |
| options.strokeColor | "0066ffff" | Тип: String \| String[]<br><br>Цвет линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeOpacity | 1 | Тип: Number \| Number[]<br><br>Прозрачность линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeStyle | — | Тип: String \| Object \| String[] \| Object[]<br><br>Стиль линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.strokeWidth | 1 | Тип: Number \| Number[] |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Толщина линии или обводки. Можно задать несколько значений для множественной обводки. |
| options.syncOverlayInit | false | Тип: Boolean<br><br>Включает синхронное добавление оверлея на карту. По умолчанию добавление оверлея осуществляется асинхронно, что позволяет предотвратить зависания браузера при добавлении на карту большого числа геообъектов. Однако, асинхронное добавление не позволяет получать доступ к оверлею сразу после добавления прямоугольника на карту. |
| options.useMapMarginInDragging | true | Тип: Boolean<br><br>При перетаскивании объекта к краю карты происходит автоматическое изменение центра карты. Нужно ли учитывать отступы карты при автоматическом смещении центра карты **map.margin.Manager**. |
| options.visible | true | Тип: Boolean<br><br>Определяет видимость прямоугольника. |
| options.zIndex | — | Тип: Number<br><br>z-index прямоугольника в обычном состоянии. Наименее приоритетный. |
| options.zIndexActive | — | Тип: Number<br><br>z-index прямоугольника с открытым балуном. Наиболее приоритетный. |
| options.zIndexDrag | — | Тип: Number<br><br>z-index прямоугольника при перетаскивании. |
| options.zIndexHover | — | Тип: Number<br><br>z-index прямоугольника при наведении на него указателя мыши. |

* Обязательный параметр/опция.

**Пример:**

```
// Создаем геодезический круг радиусом 1000 километров.
var circle = new ymaps.Circle([[50, 50], 1000000], {}, {
    draggable: true
});
// Добавляем круг на карту.
myMap.geoObjects.add(circle);

// Создаем прямоугольник на основе границы прямоугольника.
var rectangle = new ymaps.Rectangle(circle.geometry.getBounds(), {}, {
    fill: false,
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
    coordRendering: "boundsPath",
    strokeWidth: 4
});
// Добавляем прямоугольник на карту на карту.
myMap.geoObjects.add(rectangle);

// При изменении геометрии прямоугольника обновляем координаты прямоугольника.
circle.geometry.events.add("change", function (event) {
    this.geometry.setCoordinates(event.get("target").getBounds());
}, rectangle);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **balloon** | **geoObject.Balloon** | Балун геообъекта.<br>Унаследовано от **GeoObject**. |
| **editor** | Null | Редактор геометрии "Прямоугольник" на данный момент не реализован. |
| **events** | **event.Manager** | Менеджер событий.<br>Унаследовано от **GeoObject**. |
| **geometry** | **geometry.Rectangle** | Геометрия типа "Прямоугольник". |
| **hint** | **geoObject.Hint** | Хинт геообъекта.<br>Унаследовано от **GeoObject**. |
| **options** | **option.Manager** | Менеджер опций геообъекта.<br>Унаследовано от **GeoObject**. |
| **properties** | **data.Manager** | Менеджер данных геообъекта.<br>Унаследовано от **GeoObject**. |
| **state** | **data.Manager** | Состояние геообъекта. Определяется следующими полями:<br>• active: Boolean — признак того, что на геообъекте открыт балун;<br>• hover: Boolean — признак того, что в данный момент на геообъект наведен указатель мыши;<br>• drag: Boolean — признак того, что геообъект перетаскивается.<br>Унаследовано от **GeoObject**. |

## События

| Имя | Описание |
|---|---|
| **balloonclose** | Закрытие балуна. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **balloonopen** | Открытие балуна на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **beforedrag** | Событие, предшествующее событию drag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие drag будет отменено.<br>Унаследовано от **GeoObject**. |
| **beforedragstart** | Событие, предшествующее событию dragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие dragstart, будет отменено.<br>Унаследовано от **GeoObject**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **drag** | Перетаскивание геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **dragend** | Завершение перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragstart** | Начало перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **editorstatechange** | Изменение состояния редактора геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent — оригинальное событие редактора геометрии.<br>Унаследовано от **GeoObject**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **hintopen** | Открытие хинта на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание полей

### editor

```
{Null} editor
```

Редактор геометрии "Прямоугольник" на данный момент не реализован.

### geometry

```
{geometry.Rectangle} geometry
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Геометрия типа "Прямоугольник".


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# regions

## regions.load

Предоставляет доступ к геометрии различных регионов и стран мира.

Конструктор

### Конструктор

```
regions.load(region[, options])
```

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| region * | — | Тип: String<br><br>ISO_3166-1 код страны (RU, UA, BY, KZ) для загрузки административно-территориального деления, либо '001' для загрузки геометрии границ стран мира. |
| options | — | Тип: Object<br><br>Опции отображения. |
| options.disputedBorders | — | Тип: String<br><br>Двухбуквенный код государства, официальная позиция которого будет использоваться для определения административного подчинения спорных территорий. Допустимые значения: 'RU', 'UA', 'BY', 'KZ'. По умолчанию совпадает с кодом государства, указанным при загрузке API. Неподдерживаемые коды государств приводятся к RU. Для региона '001' (границы стран мира) поддерживается код 'UN' — мировые границы согласно позиции ООН. |
| options.lang | — | Тип: String<br><br>Язык (ru, uk, en, be). |
| options.quality | 1 | Тип: Number<br><br>Уровень качества. Доступные значения:<br>• 0 — минимальное качество;<br>• 1 — стандартное качество;<br>• 2 — повышенное качество;<br>• 3 — высокое качество.<br><br>Уровень качества влияет на точность представления кривых и обьем файла данных. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

**Пример:**

```
ymaps.regions.load('RU', {
    lang: 'en'
}).then(function (result) {
    geoMap.geoObjects.add(result.geoObjects);
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# RemoteObjectManager

Расширяет **ICustomizable**, **IEventEmitter**, **IGeoObject**, **IParentOnMap**.

Менеджер объектов, осуществляющий их оптимальную подгрузку с сервера. Менеджер отправляет запросы за данными по указанному url в JSONP-формате. Описание формата соответствует формату добавляемых в ObjectManager объектов ObjectManager.add. Также поддерживаются объекты типа 'Cluster', содержащие поля:

- type — тип объекта, для кластеров всегда 'Cluster';
- id — уникальный идентификатор кластера;
- geometry — геометрия кластера в JSON-формате;
- features — массив объектов в составе кластера. Необязательное поле;
- bbox — массив координат, описывающих прямоугольную область, содержащую все объекты в составе кластера;
- number — количество объектов в кластере;
- properties — данные кластера.

Данный модуль рассчитан на загрузку и отображение данных, предварительно обработанных на сервере. В частности, рекомендуется использовать модуль для отображения результатов серверной кластеризации. Данные перезапрашиваются заново при изменении коэффициента масштабирования карты. Модуль не осуществляет кластеризацию объектов или фильтрацию по области видимости на клиенте. При необходимости кластеризовать объекты на клиенте после загрузки используйте **LoadingObjectManager**. Обратите внимание, что у объектов, отрисованных на карте через данный менеджер, нельзя включать режимы редактирования и перетаскивания.

Конструктор | Поля | События | Методы

## Конструктор

```
RemoteObjectManager(urlTemplate[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| urlTemplate * | — | Тип: String<br><br>Шаблон URL данных. Поддерживаются специальные конструкции по аналогии с **Layer**. Также поддерживаются подстановки:<br>• %b заменяется на массив географических координат, описывающих прямоугольную область, для которой требуется загрузить данные.<br>• %t заменяется на массив номеров тайлов, описывающих прямоугольную область, для которой требуется загрузить данные. |
| options | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Опции.<br>• Можно задавать все опции, указанные в описании **Clusterer**, за исключением опций hasBalloon и hasHint.<br>• Опции для кластеров задаются с префиксом cluster. Список опций указан в описании класса **ClusterPlacemark**;<br>• Опции для одиночных объектов задаются с префиксом geoObject. Список опций указан в описании класса **Placemark**. |
| options.paddingParamName | 'callback' | Тип: Boolean<br><br>Имя GET-параметра, который содержит значение jsonp-колбека. |
| options.paddingTemplate | null | Тип: String<br><br>Шаблон для jsonp-колбека. Поддерживает те же подстановки, что и urlTemplate. Все символы, не являющиеся буквой или цифрой, будут заменены на '_'. Если параметр не задан, то имя jsonp-колбека будет сгенерировано автоматически. Примеры преобразований при tileNumber=[3, 1], zoom=9:<br>• 'myCallback=%x' => 'myCallback_3'<br>• '%c' => 'x_3_y_1_z_9'<br>• 'callback2_%c' => 'callback2_x_3_y_1_z_9'<br>• 'callback%test' => 'callback_test'<br>• 'callback_%b' => 'callback_85_0841__180_0000_85_0841_180_0000'<br><br>Обратите внимание, что если не использовать в значении опции подстановки, то это может привести к ошибке. Все запросы будут обращаться к одной callback-функции. |
| options.splitRequests | false | Тип: Boolean<br><br>Разделять запросы за данными на запросы за одиночными тайлами. По умолчанию запросы делаются за данными для прямоугольной области, содержащей несколько тайлов. |
| options.syncOverlayInit | false | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Флаг, разрешающий создавать оверлеи для объектов синхронно. Обратите внимание, что при синхронном создании оверлея нужно самостоятельно обеспечить загрузку нужного класса, реализующего интерфейс IOverlay. По умолчанию оверлеи создаются асинхронно, при этом класс оверлея загружается по требованию. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
var objectManager = new ymaps.RemoteObjectManager('http://myServer.com/tile?bbox=
%b', {
    // Опции кластеров задаются с префиксом cluster.
    clusterHasBalloon: false,
    // Опции геообъектов задаются с префиксом geoObject
    geoObjectOpenBalloonOnClick: false
});

// Можно задавать опции напрямую в дочерние коллекции.
objectManager.clusters.set({
    preset: 'islands#grayClusterIcons',
    hintContentLayout: ymaps.templateLayoutFactory.createClass('Группа
объектов')
});
objectManager.objects.set('preset', 'islands#grayIcon');
```

**2.**

```
Пример ответа для RemoteObjectManager
jsonp_callback({
    // Ответ содержит поля error и data. В случае ошибки в поле error
    // пишется код ошибки или ее описание.
    error: null,
    data: {
        type: 'FeatureCollection',
        features: [
             {
                 type: 'Feature',
                 geometry: {
                     type: 'Point',
                     coordinates: [55, 35]
                 },
                 id: 23,
                 properties: {
                     balloonContent: 'Содержимое балуна метки',
                     iconContent: 'Содержимое метки'
                 },
                 options: {
                     preset: 'islands#yellowIcon'
                 }
             },
             {
                 type: 'Cluster',
                 id: 24,
                 bbox: [[35, 46], [46, 57]],
                 number: 34,
                 // Массив, описывающий 34 объекта в составе кластера.
                 // Необязательное поле.
                 // Если поле не задано, у кластера по клику будет открываться
пустой балун.
                 features: [{
                     type: 'Feature',
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
                    id: 512,
                    properties: {
                        balloonContent: 'Содержимое балуна метки',
                        clusterCaption: 'Заголовок метки в балуне кластера'
                    },
                    ...
                ],
                geometry: {
                    type: 'Point',
                    coordinates: [40.5, 51]
                },
                properties: {
                    iconContent: 34
                }
            }
        ]
    }
});
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **clusters** | **objectManager.ClusterCollection** | Коллекция кластеров, сгенерированных менеджером. |
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **IGeoObject**. |
| **objects** | **objectManager.ObjectCollection** | Коллекция объектов, добавленных в слой. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **properties** | **IDataManager** | Данные геообъекта.<br>Унаследовано от **IGeoObject**. |
| **state** | **IDataManager** | Состояние геообъекта.<br>Унаследовано от **IGeoObject**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Вычисляет границы области в геокоординатах, охватывающей все загруженные объекты в составе менеджера. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getObjectState**(id) | Object | Получение информации о текущем состоянии объекта, добавленного в менеджер. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getPixelBounds**() | Number[][] \| null | Вычисляет границы области в глобальных пиксельных координатах, охватывающей все загруженные объекты в составе менеджера. |
| **getTileUrl**() | String \| null | Возвращает URL тайла с данными. |
| **getUrlTemplate**() | String | Возвращает шаблон URL данных. |
| **reloadData**() | | Метод, удаляющий все загруженные ранее данные и отправляющий запрос за новыми данными. |
| **setFilter**(filterFunction) | | Устанавливает функцию-фильтр для объектов. Фильтрует как одиночные объекты, так и кластеры. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |
| **setUrlTemplate**(urlTemplate) | | |

## Описание полей

### clusters

```
{objectManager.ClusterCollection} clusters
```

Коллекция кластеров, сгенерированных менеджером.

**Пример:**

```
objectManager.clusters.events.add('click', function (e) {
    var objectId = e.get('objectId');
    objectManager.clusters.balloon.open(objectId);
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### objects

```
{objectManager.ObjectCollection} objects
```

Коллекция объектов, добавленных в слой.

**Пример:**

```
objectManager.objects.events.add('click', function (e) {
    var objectId = e.get('objectId');
    objectManager.objects.balloon.open(objectId);
});
```

## Описание методов

### getBounds

```
{Number[][] | null} getBounds()
```

Вычисляет границы области в геокоординатах, охватывающей все загруженные объекты в составе менеджера.

**Возвращает** массив координат области или null, если менеджер не добавлен на карту.

### getObjectState

```
{Object} getObjectState(id)
```

Получение информации о текущем состоянии объекта, добавленного в менеджер.

**Возвращает** объект с полями:

- found — признак, указывающий, существует ли объект с переданным идентификатором в загруженных данных. Тип: Boolean;
- isShown — признак, указывающий, находится ли объект в видимой области карты. Тип: Boolean;
- isFilteredOut — признак, указывающий, прошел ли объект фильтрацию. Если фильтр не задан или объект прошел фильтрацию, значение поля будет false. Тип: Boolean.

.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| id* | — | Тип: Object<br><br>Идентификатор объекта, для которого нужно получить состояние. |

* Обязательный параметр/опция.

**Пример:**

```
remoteObjectManager.setFilter('properties.type == "shop"');
...
if (!remoteObjectManager.getObjectState(7).isFilteredOut) {
    remoteObjectManager.objects.balloon.open(7);
}
```

### getPixelBounds

```
{Number[][] | null} getPixelBounds()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Вычисляет границы области в глобальных пиксельных координатах, охватывающей все загруженные объекты в составе менеджера.

**Возвращает** массив координат области или null, если менеджер не добавлен на карту.

## getTileUrl

```
{String | null} getTileUrl()
```

**Возвращает** URL тайла с данными.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parameters* | — | Тип: |

* Обязательный параметр/опция.

**Пример:**

```
var objectManager = new ymaps.RemoteObjectManager('http://myServer.com/tile?bbox=
%b');
objectManager.getTileUrl = function (parameters) {
   var boundingBox = parameters.boundingBox.join('~');
   return this.getUrlTemplate().replace(/%b/g, boundingBox);
};
```

## getUrlTemplate

```
{String} getUrlTemplate()
```

**Возвращает** шаблон URL данных.

## reloadData

```
reloadData()
```

Метод, удаляющий все загруженные ранее данные и отправляющий запрос за новыми данными.

## setFilter

```
setFilter(filterFunction)
```

Устанавливает функцию-фильтр для объектов. Фильтрует как одиночные объекты, так и кластеры.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| filterFunction* | — | Тип: Function \| String |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | функция-фильтр. Получает на вход единичный объект, добавленный в ObjectManager. Если функция возвращает true, объект будет обработан. Если false — объект будет исключен из дальнейшей обработки. Также в качестве фильтра можно передавать строку. В строке-фильтре доступны следующие ключевые слова:<br>• options — обращение к опциям объекта;<br>• properties — обращение к данным объекта;<br>• geometry — обращение к геометрии объекта;<br>• id — обращение к идентификатору объекта;<br>В качестве фильтра можно указывать выражение, возвращающее true или false. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Выбрать кластеры с id > 100.
objectManager.setFilter('object.type == "Cluster" && id > 100');
```

**2.**

```
// На карте будут отображаться только объекты с заданными типами.
objectManager.setFilter('properties.type == "кафе" || properties.type ==
"аптека"');
```

**3.**

```
// Можно задавать функцию-фильтр.
objectManager.setFilter(function (object) {
    return object.properties.name != 'Тот, кого нельзя показывать.';
});
```

## setUrlTemplate

```
setUrlTemplate(urlTemplate)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| urlTemplate* | — | Тип: String<br>шаблон URL данных. |

* Обязательный параметр/опция.

# route

Статическая функция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Прокладывает маршрут через заданные точки.

---

**Примечание:**
Для платной версии API маршрутизация доступна по следующим странам: Россия, Украина, Беларусь, Казахстан, Грузия, Абхазия, Южная Осетия, Армения, Азербайджан, Молдова, Туркмения, Таджикистан, Узбекистан, Киргизия и Турция.

---

**Возвращает:** Объект-обещание, который подтверждается объектом маршрута **router.Route**, либо объектом мультимаршрута **multiRouter.MultiRoute**, в зависимости от значения параметра multiRoute. В случае ошибки объект-обещание отклоняется. (тип **vow.Promise**).

```
route(points[, params])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| points* | — | Тип: Object[]<br><br>Массив точек, через которые должен проходить маршрут. Каждая точка может быть задана строкой, содержащей адрес, массивом координат, и JSON-объектом со следующими полями:<br><br>• type: String — тип точки. Значение 'wayPoint' задает путевую точку маршрута. Значение 'viaPoint' задает транзитную точку, т.е. точку, через которую нужно проехать без остановки.<br>• point: Number[]\|String — Массив координат точки, либо ее адрес в виде строки. |
| params | — | Тип: Object<br><br>Параметры маршрутизации. |
| params.avoidTrafficJams | false | Тип: Boolean<br><br>Включает построение маршрута с учетом пробок. При использовании опции учитывайте, что объезд пробок не всегда возможен. |
| params.boundedBy | — | Тип: Number[][]<br><br>Область на карте, где предположительно находятся искомые объекты. Используется, если точки маршрута заданы почтовым адресом, а не координатами. |
| params.mapStateAutoApply | false | Тип: Boolean<br><br>Флаг, позволяющий автоматически установить центр и коэффициент масштабирования карты так, чтобы построенный маршрут был виден целиком. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| params.multiRoute | false | Тип: Boolean<br><br>Позволяет строить мультимаршруты. |
| params.routingMode | "auto" | Тип: String<br><br>Тип маршрутизации. Может принимать одно из трех строковых значений:<br>• "auto" — автомобильная маршрутизация;<br>• "masstransit" — маршрутизация с использованием общественного транспорта. Доступна только для мультимаршрутов (опция multiRoute должна быть выставлена в true);<br>• "pedestrian" — пешеходная маршрутизация. |
| params.searchCoordOrder | — | Тип: String<br><br>Определяет каким образом нужно интерпретировать координаты в запросе. Используется, если точки маршрута заданы координатами, а не почтовым адресом. |
| params.strictBounds | false | Тип: Boolean<br><br>Искать только внутри области, заданной опцией boundedBy. |
| params.useMapMargin | true | Тип: Boolean<br><br>Нужно ли учитывать отступы карты **map.margin.Manager**. |
| params.viaIndexes | [] | Тип: Integer[]<br><br>Индексы транзитных точек мультимаршрута. |
| params.zoomMargin | 0 | Тип: Number \| Number[]<br><br>Отступы от границ видимой области карты при изменении коэффициента масштабирования. Если задано одно число — оно применяется ко всем сторонам. Если задано два — то это горизонтальные и вертикальные отступы соответственно. Если задан массив из 4х чисел, то это отступы top, right, bottom, left. При включенном параметре "useMapMargin" значение "zoomMargin" складывается со значениями, которые были рассчитаны в менеджере отступов **map.margin.Manager**. |

* Обязательный параметр/опция.

**Примеры:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**1.**

```
// Строим маршрут из Королева в Красногорск через Химки и Мытищи,
// где Мытищи - транзитная точка. Красногорск зададим координатами.
ymaps.route([
    'Королев',
    { type: 'viaPoint', point: 'Мытищи' },
    'Химки',
    { type: 'wayPoint', point: [55.811511, 37.312518] }
], {
    mapStateAutoApply: true
}).then(function (route) {
    route.getPaths().options.set({
        // в балуне выводим только информацию о времени движения с учетом пробок
        balloonContentLayout:
ymaps.templateLayoutFactory.createClass('{{ properties.humanJamsTime }}'),
        // можно выставить настройки графики маршруту
        strokeColor: '0000ffff',
        opacity: 0.9
    });
    // добавляем маршрут на карту
    map.geoObjects.add(route);
});
```

**2.**

```
// Построим мультимаршрут и добавим его на карту с применением
автомасштабирования.
ymaps.route(['Южное Бутово', 'Москва, метро Парк Культуры'], {
    multiRoute: true
}).done(function (route) {
    route.options.set("mapStateAutoApply", true);
    myMap.geoObjects.add(route);
}, function (err) {
    throw err;
}, this);
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# router

## router.addon

### router.addon.editor

#### router.addon.editor.get

Статическая функция.

**Возвращает:** Редактор машуртизатора. (тип **router.Route**).

```
router.addon.editor.get()
```

**Пример:**

```
ymaps.router.addon.editor.get(myMap)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# router.Editor

**Примечание:**
Конструктор класса router.Editor скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Расширяет **ICustomizable**, **IEventEmitter**.

Редактор маршрутов. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Поля | События | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **state** | **IDataManager** | Менеджер состояния редактора маршрута.<br>Поля данных, доступные посредством методов get и set:<br>• routeloading: Boolean — признак того, что в данный момент идет загрузка данных из сервиса маршрутизации.<br>• waypointsdrag: Boolean — признак того, что в данный момент происходит перетаскивание путевой точки.<br>• viapointsdrag: Boolean — признак того, что в данный момент происходит перетаскивание транзитной точки. |

## События

| Имя | Описание |
|---|---|
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **routeupdate** | Обновление маршрута. Через значение флага e.get('rough') можно определить событие бросилось после завершения редактирования или во время. Если вы хотите в своем приложении обновлять информацию, связанную с маршрутом, то нужно делать проверку e.get('rough') == false, чтобы избежать частой обработки этого события. |
| **start** | Включение редактора. |
| **stop** | Выключение редактора. |
| **viapointadd** | Добавление транзитной точки. Добавляемую транзитную точку можно получить через e.get('viaPoint') |
| **viapointdragend** | Завершение перетаскивания транзитной точки. Путевую точку можно получить через e.get('viaPoint') |
| **viapointdragstart** | Начало перетаскивания транзитной точки. Путевую точку можно получить через e.get('viaPoint') |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| viapointremove | Удаление транзитной точки. Удаляемую транзитную точку можно получить через e.get('viaPoint') |
| waypointadd | Добавление путевой точки. Добавляемую путевую точку можно получить через e.get('wayPoint') |
| waypointdragend | Завершение перетаскивания путевой точки. Путевую точку можно получить через e.get('wayPoint') |
| waypointdragstart | Начало перетаскивания путевой точки. Путевую точку можно получить через e.get('wayPoint') |
| waypointremove | Удаление путевой точки. Удаляемую путевую точку можно получить через e.get('wayPoint') |

## Методы

| Имя | Описание |
|---|---|
| start([options]) | Включает редактор маршрута. |
| stop() | Выключает редактор маршрута. |

## Описание полей

### state

```
{IDataManager} state
```

Менеджер состояния редактора маршрута.

Поля данных, доступные посредством методов get и set:

- routeloading: Boolean — признак того, что в данный момент идет загрузка данных из сервиса маршрутизации.
- waypointsdrag: Boolean — признак того, что в данный момент происходит перетаскивание путевой точки.
- viapointsdrag: Boolean — признак того, что в данный момент происходит перетаскивание транзитной точки.

## Описание событий

### routeupdate

Обновление маршрута. Через значение флага e.get('rough') можно определить событие бросилось после завершения редактирования или во время. Если вы хотите в своем приложении обновлять информацию, связанную с маршрутом, то нужно делать проверку e.get('rough') == false, чтобы избежать частой обработки этого события.

### start

Включение редактора.

### stop

Выключение редактора.

### viapointadd

Добавление транзитной точки. Добавляемую транзитную точку можно получить через e.get('viaPoint')


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### viapointdragend

Завершение перетаскивания транзитной точки. Путевую точку можно получить через e.get('viaPoint')

### viapointdragstart

Начало перетаскивания транзитной точки. Путевую точку можно получить через e.get('viaPoint')

### viapointremove

Удаление транзитной точки. Удаляемую транзитную точку можно получить через e.get('viaPoint')

### waypointadd

Добавление путевой точки. Добавляемую путевую точку можно получить через e.get('wayPoint')

### waypointdragend

Завершение перетаскивания путевой точки. Путевую точку можно получить через e.get('wayPoint')

### waypointdragstart

Начало перетаскивания путевой точки. Путевую точку можно получить через e.get('wayPoint')

### waypointremove

Удаление путевой точки. Удаляемую путевую точку можно получить через e.get('wayPoint')

## Описание методов

### start

```
start([options])
```

Включает редактор маршрута.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции. |
| options.addViaPoints | true | Тип: Boolean<br><br>true — добавление транзитных точек разрешено, false — запрещено. |
| options.addWayPoints | false | Тип: Boolean<br><br>true — добавление путевых точек при клике на карте разрешено, false — запрещено. |
| options.editViaPoints | true | Тип: Boolean<br><br>true — редактирование (перемещение) транзитных точек разрешено, false — запрещено. |
| options.editWayPoints | true | Тип: Boolean |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | true — редактирование (перемещение) путевых точек разрешено, false — запрещено. |
| options.removeViaPoints | true | Тип: Boolean<br><br>true — удаление транзитных точек двойным кликом разрешено, false — запрещено. |
| options.removeWayPoints | false | Тип: Boolean<br><br>true — удаление путевых точек двойным кликом разрешено, false — запрещено. |

### stop

```
stop()
```

Выключает редактор маршрута.

## router.Path

---

**Примечание:**
Конструктор класса router.Path скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет **GeoObject**.

Объект, описывающий часть (путь) машрута. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию. Маршрут может содержать несколько путей, каждый путь соединяет две путевые точки. Для путей доступны следующие пресеты:

- 'router#interactivePath' — обеспечивает вывод информации о сегменте пути при клике по полилинии пути. При использовании этого пресета в макете балуна становятся доступными, как свойства геообъекта, следующие данные сегмента **router.Segment**: action, humanAction, angle, length, humanLength, time, humanTime, jamsTime, humanJamsTime, street. Используется по умолчанию.
- 'router#plainPath' — путь ведет себя как обычный геообъект.

**См.: route**

Поля | События | Методы

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **balloon** | **geoObject.Balloon** | Балун геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **editor** | **IGeometryEditor** | Редактор геометрии геообъекта.<br><br>Унаследовано от **GeoObject**. |
| **events** | **event.Manager** | Менеджер событий. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **GeoObject**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **GeoObject**. |
| **hint** | **geoObject.Hint** | Хинт геообъекта.<br>Унаследовано от **GeoObject**. |
| **options** | **option.Manager** | Менеджер опций геообъекта.<br>Унаследовано от **GeoObject**. |
| **properties** | **data.Manager** | Менеджер данных геообъекта.<br>Унаследовано от **GeoObject**. |
| **state** | **data.Manager** | Состояние геообъекта. Определяется следующими полями:<br>• active: Boolean — признак того, что на геообъекте открыт балун;<br>• hover: Boolean — признак того, что в данный момент на геообъект наведен указатель мыши;<br>• drag: Boolean — признак того, что геообъект перетаскивается.<br>Унаследовано от **GeoObject**. |

## События

| Имя | Описание |
|---|---|
| **balloonclose** | Закрытие балуна. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **balloonopen** | Открытие балуна на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **beforedrag** | Событие, предшествующее событию drag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие drag будет отменено.<br>Унаследовано от **GeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **beforedragstart** | Событие, предшествующее событию dragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие dragstart, будет отменено.<br>Унаследовано от **GeoObject**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **drag** | Перетаскивание геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragend** | Завершение перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragstart** | Начало перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **editorstatechange** | Изменение состояния редактора геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent — оригинальное событие редактора геометрии. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| | Унаследовано от **GeoObject**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **hintopen** | Открытие хинта на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getHumanJamsTime() | String | Возвращает строковое представление времени проезда пути с единицами измерения с учетом пробок. |
| getHumanLength() | String | Возвращает строковое представление длины пути с единицами измерения. |
| getHumanTime() | String | Возвращает строковое представление времени проезда пути с единицами измерения. |
| getJamsTime() | Integer | Возвращает время проезда пути в секундах с учетом пробок. |
| getLength() | Number | Возвращает длину пути в метрах. |
| getMap() | Map | Возвращает ссылку на карту.<br><br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br><br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br><br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br><br>Унаследован от IChildOnMap. |
| getSegments() | router.Segment[] | Возвращает сегменты пути. |
| getTime() | Integer | Возвращает время проезда пути в секундах. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br><br>Унаследован от IChildOnMap. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### getHumanJamsTime

```
{String} getHumanJamsTime()
```

**Возвращает** строковое представление времени проезда пути с единицами измерения с учетом пробок.

### getHumanLength

```
{String} getHumanLength()
```

**Возвращает** строковое представление длины пути с единицами измерения.

### getHumanTime

```
{String} getHumanTime()
```

**Возвращает** строковое представление времени проезда пути с единицами измерения.

### getJamsTime

```
{Integer} getJamsTime()
```

**Возвращает** время проезда пути в секундах с учетом пробок.

### getLength

```
{Number} getLength()
```

**Возвращает** длину пути в метрах.

### getSegments

```
{router.Segment[]} getSegments()
```

**Возвращает** сегменты пути.

### getTime

```
{Integer} getTime()
```

**Возвращает** время проезда пути в секундах.

## router.Route

---

**Примечание:**
Конструктор класса router.Route скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет **IGeoObject**.

Объект, описывающий проложенный маршрут. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

**См.:** route


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Поля | События | Методы

## Поля

| Имя | Тип | Описание |
|---|---|---|
| editor | router.Editor | Редактор маршрутов. |
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |
| geometry | IGeometry \| null | Геометрия геообъекта.<br>Унаследовано от IGeoObject. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |
| properties | IDataManager | Данные геообъекта.<br>Унаследовано от IGeoObject. |
| state | IDataManager | Состояние геообъекта.<br>Унаследовано от IGeoObject. |

## События

| Имя | Описание |
|---|---|
| boundsapply | Событие выставления границ маршрута карте при заданной опции options.mapStateAutoApply. |
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| geometrychange | Изменение геометрии геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие геометрии.<br>Унаследовано от IGeoObject. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| | Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **update** | Событие обновления маршрута при включенном редакторе маршрутов. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getHumanJamsTime**() | String | Возвращает строковое представление времени проезда маршрута с единицами измерения с учетом пробок. |
| **getHumanLength**() | String | Возвращает строковое представление длины маршрута с единицами измерения. |
| **getHumanTime**() | String | Возвращает строковое представление времени проезда маршрута с единицами измерения. |
| **getJamsTime**() | Integer | Возвращает время проезда маршрута в секундах с учетом пробок. |
| **getLength**() | Number | Возвращает длину маршрута в метрах. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| getPaths() | GeoObjectCollection | Возвращает коллекцию путей, из которых состоит маршрут. |
| getTime() | Integer | Возвращает время проезда маршрута в секундах. |
| getViaPoints() | GeoObjectCollection | Возвращает коллекцию транзитных точек маршрута. |
| getWayPoints() | GeoObjectCollection | Возвращает коллекцию путевых точек маршрута. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание полей

### editor

```
{router.Editor} editor
```

Редактор маршрутов.

**Пример:**

```
// Начало редактирования маршрута.
ymaps.route(['Москва', 'Питер'], function (route) {
    route.editor.start();
    // ...
    // Завершение редактирования маршрута.
    route.editor.stop();
});
```

## Описание событий

### boundsapply

Событие выставления границ маршрута карте при заданной опции options.mapStateAutoApply.

### update

Событие обновления маршрута при включенном редакторе маршрутов.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### getHumanJamsTime

```
{String} getHumanJamsTime()
```

**Возвращает** строковое представление времени проезда маршрута с единицами измерения с учетом пробок.

### getHumanLength

```
{String} getHumanLength()
```

**Возвращает** строковое представление длины маршрута с единицами измерения.

### getHumanTime

```
{String} getHumanTime()
```

**Возвращает** строковое представление времени проезда маршрута с единицами измерения.

### getJamsTime

```
{Integer} getJamsTime()
```

**Возвращает** время проезда маршрута в секундах с учетом пробок.

### getLength

```
{Number} getLength()
```

**Возвращает** длину маршрута в метрах.

### getPaths

```
{GeoObjectCollection} getPaths()
```

**Возвращает** коллекцию путей, из которых состоит маршрут.

### getTime

```
{Integer} getTime()
```

**Возвращает** время проезда маршрута в секундах.

### getViaPoints

```
{GeoObjectCollection} getViaPoints()
```

**Возвращает** коллекцию транзитных точек маршрута.

### getWayPoints

```
{GeoObjectCollection} getWayPoints()
```

**Возвращает** коллекцию путевых точек маршрута.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# router.Segment

**Примечание:**
Конструктор класса router.Segment скрыт, так как данный класс не предназначен для самостоятельной инициализации.

Объект, описывающий сегмент маршрута. Сегмент — часть маршрута от одного манёвра до другого. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Методы

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getAction**() | String | Возвращает направление поворота маршрута в конце сегмента. Возможные значения:<br>• left — налево;<br>• slight left — левее;<br>• hard left — резкий поворот налево;<br>• right — направо;<br>• slight right — правее;<br>• hard right — резкий поворот направо;<br>• none — прямо;<br>• back — разворот;<br>• enter roundabout — въезд на перекресток с круговым движением;<br>• leave roundabout [N] — съезд с перекрестка с круговым движением. Число N — номер поворота с круга. Данное число может отсутствовать. Например, "leave roundabout" или "leave roundabout 2";<br>• merge — въезд (например, на шоссе или магистраль). Обозначает слияние с транспортным потоком;<br>• board ferry — паромная переправа.<br>. |
| **getAngle**() | Number | Определяет угол поворота маршрута в конце отрезка. |
| **getCoordinates**() | Number[][] | Возвращает координаты ломаной, описывающей геометрию сегмента. |
| **getHumanAction**() | String | Возвращает направление поворота в виде локализованной человекопонятной строки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getHumanJamsTime() | String | Возвращает строковое представление времени проезда сегмента с единицами измерения с учетом пробок. |
| getHumanLength() | String | Возвращает строковое представление длины сегмента с единицами измерения. |
| getHumanTime() | String | Возвращает строковое представление времени проезда сегмента с единицами измерения. |
| getIndex() | Integer | Возвращает индекс данного сегмента в массиве всех сегментов пути. |
| getJamsTime() | Integer | Возвращает время проезда сегмента в секундах с учетом пробок. |
| getLength() | Number | Возвращает длину сегмента в метрах. |
| getPolylineEndIndex() | Integer | Возвращает индекс точки в геометрии пути, на которой заканчивается сегмент. |
| getPolylineStartIndex() | Integer | Возвращает индекс точки в геометрии пути, с которой начинается сегмент. |
| getStreet() | String | Возвращает название улицы, по которой проходит сегмент маршрута. |
| getTime() | Integer | Возвращает время проезда сегмента в секундах. |

## Описание методов

### getAction

```
{String} getAction()
```

**Возвращает** направление поворота маршрута в конце сегмента. Возможные значения:

- left — налево;
- slight left — левее;
- hard left — резкий поворот налево;
- right — направо;
- slight right — правее;
- hard right — резкий поворот направо;
- none — прямо;
- back — разворот;
- enter roundabout — въезд на перекресток с круговым движением;
- leave roundabout [N] — съезд с перекрестка с круговым движением. Число N — номер поворота с круга. Данное число может отсутствовать. Например, "leave roundabout" или "leave roundabout 2";
- merge — въезд (например, на шоссе или магистраль). Обозначает слияние с транспортным потоком;
- board ferry — паромная переправа.

.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getAngle

```
{Number} getAngle()
```

Определяет угол поворота маршрута в конце отрезка.

**Возвращает** угол поворота (в градусах).

## getCoordinates

```
{Number[][]} getCoordinates()
```

**Возвращает** координаты ломаной, описывающей геометрию сегмента.

## getHumanAction

```
{String} getHumanAction()
```

**Возвращает** направление поворота в виде локализованной человекопонятной строки.

## getHumanJamsTime

```
{String} getHumanJamsTime()
```

**Возвращает** строковое представление времени проезда сегмента с единицами измерения с учетом пробок.

## getHumanLength

```
{String} getHumanLength()
```

**Возвращает** строковое представление длины сегмента с единицами измерения.

## getHumanTime

```
{String} getHumanTime()
```

**Возвращает** строковое представление времени проезда сегмента с единицами измерения.

## getIndex

```
{Integer} getIndex()
```

**Возвращает** индекс данного сегмента в массиве всех сегментов пути.

## getJamsTime

```
{Integer} getJamsTime()
```

**Возвращает** время проезда сегмента в секундах с учетом пробок.

## getLength

```
{Number} getLength()
```

**Возвращает** длину сегмента в метрах.

## getPolylineEndIndex

```
{Integer} getPolylineEndIndex()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** индекс точки в геометрии пути, на которой заканчивается сегмент.

### getPolylineStartIndex

```
{Integer} getPolylineStartIndex()
```

**Возвращает** индекс точки в геометрии пути, с которой начинается сегмент.

### getStreet

```
{String} getStreet()
```

**Возвращает** название улицы, по которой проходит сегмент маршрута.

### getTime

```
{Integer} getTime()
```

**Возвращает** время проезда сегмента в секундах.

## router.ViaPoint

---

**Примечание:**
Конструктор класса router.ViaPoint скрыт, так как данный класс не предназначен для самостоятельной инициализации.

---

Расширяет **GeoObject**.

Объект, описывающий транзитную точку маршрута. Конструктор недоступен в package.full (стандартный набор модулей). Данный модуль загружается по требованию.

Поля | События | Методы

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **balloon** | **geoObject.Balloon** | Балун геообъекта.<br>Унаследовано от **GeoObject**. |
| **editor** | **IGeometryEditor** | Редактор геометрии геообъекта.<br>Унаследовано от **GeoObject**. |
| **events** | **event.Manager** | Менеджер событий.<br>Унаследовано от **GeoObject**. |
| **geometry** | **IGeometry** \| null | Геометрия геообъекта.<br>Унаследовано от **GeoObject**. |
| **hint** | **geoObject.Hint** | Хинт геообъекта.<br>Унаследовано от **GeoObject**. |
| **options** | **option.Manager** | Менеджер опций геообъекта.<br>Унаследовано от **GeoObject**. |
| **properties** | **data.Manager** | Менеджер данных геообъекта. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **GeoObject**. |
| **state** | **data.Manager** | Состояние геообъекта. Определяется следующими полями:<br>• active: Boolean — признак того, что на геообъекте открыт балун;<br>• hover: Boolean — признак того, что в данный момент на геообъект наведен указатель мыши;<br>• drag: Boolean — признак того, что геообъект перетаскивается.<br>Унаследовано от **GeoObject**. |

## События

| Имя | Описание |
|---|---|
| **balloonclose** | Закрытие балуна. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **balloonopen** | Открытие балуна на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **beforedrag** | Событие, предшествующее событию drag. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие drag будет отменено.<br>Унаследовано от **GeoObject**. |
| **beforedragstart** | Событие, предшествующее событию dragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие dragstart, будет отменено.<br>Унаследовано от **GeoObject**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **drag** | Перетаскивание геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragend** | Завершение перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragstart** | Начало перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **editorstatechange** | Изменение состояния редактора геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent — оригинальное событие редактора геометрии.<br>Унаследовано от **GeoObject**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **hintopen** | Открытие хинта на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| overlaychange | Изменение оверлея геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• overlay: IOverlay\|null — ссылка на оверлей;<br>• oldOverlay: IOverlay\|null — предыдущий оверлей геообъекта.<br>Унаследовано от IGeoObject. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |
| propertieschange | Изменение данных геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• originalEvent: IEvent — оригинальное событие менеджера данных.<br>Унаследовано от IGeoObject. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от IGeoObject. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от IGeoObject. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| getPathIndex() | Integer | Возвращает индекс пути, в котором расположена точка. |
| getSegmentIndex() | Integer | Возвращает индекс сегмента пути, в котором расположена точка. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание методов

### getPathIndex

```
{Integer} getPathIndex()
```

**Возвращает** индекс пути, в котором расположена точка.

### getSegmentIndex

```
{Integer} getSegmentIndex()
```

**Возвращает** индекс сегмента пути, в котором расположена точка.

# router.WayPoint

Расширяет GeoObject.

Объект, описывающий путевую точку маршрута.

Конструктор | Поля | События | Методы

## Конструктор

```
router.WayPoint(feature[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| feature * | — | Тип: Object<br>Свойства и геометрия. |
| options | — | Тип: Object<br>Опции. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| balloon | geoObject.Balloon | Балун геообъекта.<br>Унаследовано от GeoObject. |
| editor | IGeometryEditor | Редактор геометрии геообъекта.<br>Унаследовано от GeoObject. |
| events | event.Manager | Менеджер событий.<br>Унаследовано от GeoObject. |
| geometry | IGeometry \| null | Геометрия геообъекта.<br>Унаследовано от GeoObject. |
| hint | geoObject.Hint | Хинт геообъекта.<br>Унаследовано от GeoObject. |
| options | option.Manager | Менеджер опций геообъекта.<br>Унаследовано от GeoObject. |
| properties | data.Manager | Менеджер данных путевой точки. Если путевая точка была задана адресом, то в поле GeocoderMetaData будут содержаться метаданные геокодера. См. geocode. |
| state | data.Manager | Состояние геообъекта. Определяется следующими полями:<br>• active: Boolean — признак того, что на геообъекте открыт балун;<br>• hover: Boolean — признак того, что в данный момент на геообъект наведен указатель мыши;<br>• drag: Boolean — признак того, что геообъект перетаскивается.<br>Унаследовано от GeoObject. |

## События

| Имя | Описание |
|---|---|
| balloonclose | Закрытие балуна. Экземпляр класса Event.<br>Унаследовано от GeoObject. |
| balloonopen | Открытие балуна на геообъекте. Экземпляр класса Event.<br>Унаследовано от GeoObject. |
| beforedrag | Событие, предшествующее событию drag. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта DomEvent), если оно имеется. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Имена методов, доступных через метод Event.callMethod:<br>• setPixelOffset — метод, позволяющий скорректировать значение пиксельного сдвига, который будет фактически применен. В качестве аргумента принимает новый пиксельный сдвиг в виде массива из двух чисел.<br>Если у данного события вызвать метод Event.preventDefault, то последующее событие drag будет отменено.<br>Унаследовано от **GeoObject**. |
| **beforedragstart** | Событие, предшествующее событию dragstart. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Если у данного события вызвать метод Event.preventDefault, то последующее перетаскивание, как и событие dragstart, будет отменено.<br>Унаследовано от **GeoObject**. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **drag** | Перетаскивание геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• pixelOffset — массив из двух чисел, описывающий пиксельный сдвиг на данном шаге;<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **dragend** | Завершение перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **dragstart** | Начало перетаскивания геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массив вида [pageX, pageY];<br>• domEvent — исходное DOM-событие (в виде объекта **DomEvent**), если оно имеется.<br>Унаследовано от **GeoObject**. |
| **editorstatechange** | Изменение состояния редактора геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent — оригинальное событие редактора геометрии.<br>Унаследовано от **GeoObject**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие геометрии.<br>Унаследовано от **IGeoObject**. |
| **hintclose** | Закрытие хинта. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **hintopen** | Открытие хинта на геообъекте. Экземпляр класса **Event**.<br>Унаследовано от **GeoObject**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта.<br><br>Унаследовано от **IGeoObject**. |
| **parentchange** | Сменился родительский объект.<br><br>Поля данных:<br><br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br><br>Унаследовано от **IChild**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных.<br>Унаследовано от **IGeoObject**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.<br>Унаследован от **IGeoObject**. |
| **getOverlaySync**() | **IOverlay** \| null | Метод предоставляет синхронный доступ к оверлею.<br>Унаследован от **IGeoObject**. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание полей

### properties

```
{data.Manager} properties
```

Менеджер данных путевой точки. Если путевая точка была задана адресом, то в поле GeocoderMetaData будут содержаться метаданные геокодера. См. **geocode**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# shape

## shape.Circle

Расширяет **IShape**.

Пиксельная фигура "Круг".

Конструктор | Методы

### Конструктор

```
shape.Circle(pixelGeometry[, params])
```

Создает фигуру.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pixelGeometry * | — | Тип: **IPixelCircleGeometry**<br><br>Пиксельная геометрия фигуры. |
| params | — | Тип: Object<br><br>Параметры фигуры. |
| params.fill | true | Тип: Boolean<br><br>Флаг наличия заливки. |
| params.outline | true | Тип: Boolean<br><br>Флаг наличия линии обводки. |
| params.strokeWidth | 0 | Тип: Number<br><br>Толщина линии обводки в пикселях. |

* Обязательный параметр/опция.

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет лежит ли переданная точка внутри фигуры.<br><br>Унаследован от **IShape**. |
| **equals**(shape) | Boolean | Возвращает true, если переданная фигура эквивалентна данной.<br><br>Унаследован от **IShape**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей фигуру. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IShape**. |
| **getGeometry**() | **IPixelGeometry** | Возвращает пиксельную геометрию фигуры.<br>Унаследован от **IShape**. |
| **getType**() | String | Возвращает идентификатор типа фигуры.<br>Унаследован от **IShape**. |
| **scale**(factor) | **IShape** | Создает отмасштабированную копию фигуры.<br>Унаследован от **IShape**. |
| **shift**(offset) | **IShape** | Создает копию фигуры сдвинутую на заданную величину.<br>Унаследован от **IShape**. |

# shape.LineString

Расширяет **IShape**.

Пиксельная фигура "Ломаная линия".

Конструктор | Методы

## Конструктор

```
shape.LineString(pixelGeometry[, params])
```

Создает фигуру.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| `pixelGeometry` * | — | Тип: **IPixelLineStringGeometry**<br>Пиксельная геометрия фигуры. |
| `params` | — | Тип: Object<br>Параметры фигуры. |
| `params.strokeWidth` | 1 | Тип: Number<br>Толщина линии в пикселях. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет лежит ли переданная точка внутри фигуры.<br>Унаследован от **IShape**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **equals**(shape) | Boolean | Возвращает true, если переданная фигура эквивалентна данной.<br>Унаследован от **IShape**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей фигуру. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IShape**. |
| **getGeometry**() | **IPixelGeometry** | Возвращает пиксельную геометрию фигуры.<br>Унаследован от **IShape**. |
| **getType**() | String | Возвращает идентификатор типа фигуры.<br>Унаследован от **IShape**. |
| **scale**(factor) | **IShape** | Создает отмасштабированную копию фигуры.<br>Унаследован от **IShape**. |
| **shift**(offset) | **IShape** | Создает копию фигуры сдвинутую на заданную величину.<br>Унаследован от **IShape**. |

# shape.MultiPolygon

Расширяет **IShape**.

Пиксельная фигура "Мультиполигон".

Конструктор | Методы

## Конструктор

```
shape.MultiPolygon(pixelGeometry[, params])
```

Создает фигуру.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| `pixelGeometry` * | — | Тип: **IPixelMultiPolygonGeometry**<br>Пиксельная геометрия фигуры. |
| `params` | — | Тип: Object<br>Параметры фигуры. |
| `params.fill` | true | Тип: Boolean<br>Флаг наличия заливки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| params.outline | true | Тип: Boolean<br><br>Флаг наличия линии обводки. |
| params.strokeWidth | 0 | Тип: Number<br><br>Толщина линии обводки в пикселях. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет лежит ли переданная точка внутри фигуры.<br><br>Унаследован от **IShape**. |
| **equals**(shape) | Boolean | Возвращает true, если переданная фигура эквивалентна данной.<br><br>Унаследован от **IShape**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей фигуру. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IShape**. |
| **getGeometry**() | **IPixelGeometry** | Возвращает пиксельную геометрию фигуры.<br><br>Унаследован от **IShape**. |
| **getType**() | String | Возвращает идентификатор типа фигуры.<br><br>Унаследован от **IShape**. |
| **scale**(factor) | **IShape** | Создает отмасштабированную копию фигуры.<br><br>Унаследован от **IShape**. |
| **shift**(offset) | **IShape** | Создает копию фигуры сдвинутую на заданную величину.<br><br>Унаследован от **IShape**. |

# shape.Polygon

Расширяет **IShape**.

Пиксельная фигура "Полигон".

Конструктор | Методы


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Конструктор

```
shape.Polygon(pixelGeometry[, params])
```

Создает фигуру.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pixelGeometry * | — | Тип: **IPixelPolygonGeometry**<br><br>Пиксельная геометрия фигуры. |
| params | — | Тип: Object<br><br>Параметры фигуры. |
| params.fill | true | Тип: Boolean<br><br>Флаг наличия заливки. |
| params.outline | true | Тип: Boolean<br><br>Флаг наличия линии обводки. |
| params.strokeWidth | 0 | Тип: Number<br><br>Толщина линии обводки в пикселях. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет лежит ли переданная точка внутри фигуры.<br><br>Унаследован от **IShape**. |
| **equals**(shape) | Boolean | Возвращает true, если переданная фигура эквивалентна данной.<br><br>Унаследован от **IShape**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей фигуру. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IShape**. |
| **getGeometry**() | **IPixelGeometry** | Возвращает пиксельную геометрию фигуры.<br><br>Унаследован от **IShape**. |
| **getType**() | String | Возвращает идентификатор типа фигуры.<br><br>Унаследован от **IShape**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **scale**(factor) | IShape | Создает отмасштабированную копию фигуры.<br>Унаследован от **IShape**. |
| **shift**(offset) | IShape | Создает копию фигуры сдвинутую на заданную величину.<br>Унаследован от **IShape**. |

# shape.Rectangle

Расширяет **IShape**.

Пиксельная фигура "Прямоугольник".

Конструктор | Методы

## Конструктор

```
shape.Rectangle(pixelGeometry[, params])
```

Создает фигуру.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pixelGeometry * | — | Тип: **IPixelRectangleGeometry**<br>Пиксельная геометрия фигуры. |
| params | — | Тип: Object<br>Параметры фигуры. |
| params.fill | true | Тип: Boolean<br>Флаг наличия заливки. |
| params.outline | true | Тип: Boolean<br>Флаг наличия линии обводки. |
| params.strokeWidth | 0 | Тип: Number<br>Толщина линии обводки в пикселях. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет лежит ли переданная точка внутри фигуры.<br>Унаследован от **IShape**. |
| **equals**(shape) | Boolean | Возвращает true, если переданная фигура эквивалентна данной.<br>Унаследован от **IShape**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей фигуру. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IShape**. |
| **getGeometry**() | **IPixelGeometry** | Возвращает пиксельную геометрию фигуры.<br>Унаследован от **IShape**. |
| **getType**() | String | Возвращает идентификатор типа фигуры.<br>Унаследован от **IShape**. |
| **scale**(factor) | **IShape** | Создает отмасштабированную копию фигуры.<br>Унаследован от **IShape**. |
| **shift**(offset) | **IShape** | Создает копию фигуры сдвинутую на заданную величину.<br>Унаследован от **IShape**. |

# shape.storage

Статический объект.

Экземпляр класса **util.Storage**

Хранилище геометрий хотспотных шейпов.

Методы

```
shape.storage()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## suggest

Статическая функция.

Обрабатывает запросы за поисковыми подсказками. Возвращает объект-promise, который будет либо отклонён с ошибкой, либо подтверждён массивом объектов вида { displayName: "Мытищи, Московская область", value: "Россия, Московская область, Мытищи " }. Поле displayName отвечает за представление топонима в удобном для чтения пользователем виде, поле value — значение, которое необходимо подставить в поле ввода после выбора пользователем данной подсказки.

**Возвращает:** Объект-promise. (тип **vow.Promise**).

```
suggest(request[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| request* | — | Тип: String<br><br>Строка запроса. |
| options | — | Тип: Object<br><br>Опции. |
| options.boundedBy | — | Тип: Number[][]<br><br>Прямоугольная область на карте, где предположительно находится искомый объект. Должна быть задана как массив, например [[30, 40], [50, 50]]. |
| options.provider | 'yandex#map' | Тип: **ISuggestProvider** \| String<br><br>Провайдер поисковых подсказок. Можно воспользоваться встроенным провайдером подсказок для объектов карты 'yandex#map' или задать собственный. |
| options.results | — | Тип: Number<br><br>Максимальное количество возвращаемых результатов. |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.suggest('мыт').then(function (items) {
    // items - массив поисковых подсказок
});
```

## SuggestView

Расширяет **ICustomizable**, **IEventEmitter**.

Создаёт выпадающую панель с поисковыми подсказками и прикрепляет к HTML-элементу <input type="text">.

Конструктор | Поля | События | Методы


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Конструктор

```
SuggestView(element[, options])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| element * | — | Тип: HTMLElement \| String<br><br>HTML-элемент или его id. |
| options | — | Тип: Object<br><br>Опции. |
| options.boundedBy | — | Тип: Number[][]<br><br>Прямоугольная область на карте, где предположительно находится искомый объект. Должна быть задана как массив, например [[30, 40], [50, 50]]. |
| options.container | — | Тип: HTMLElement<br><br>HTML-элемент для размещения верстки панели подсказок. Если не задан, панель подсказок добавляется в родителя HTML-элемента input, для которого создаётся. |
| options.layout | 'islands#suggestView' | Тип: String \| **ISuggestViewLayout**<br><br>Макет панели.<br><br>В конструктор макета передается объект, содержащий поля:<br>• suggestView — ссылка панель с поисковыми подсказками;<br>• options — менеджер опций элемента управления suggestView.options;<br>• state — менеджер состояния элемента управления suggestView.state.<br><br>Макет меняет свой внешний вид на основе состояния и опций элемента панели подсказок. Элемент управления, в свою очередь, реагирует на интерфейсные события макета и меняет значения полей suggestView.state в зависимости от полученных команд. |
| options.offset | — | Тип: Number[] |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Отступы панели подсказок от её положения по умолчанию (по умолчанию панель подсказок прикрепляется к нижней границе элемента input и имеет одинаковую с ним ширину). Задаётся в виде смещений по горизонтали и вертикали относительно левого нижнего угла элемента input. |
| options.provider | "yandex#map" | Тип: String \| **ISuggestProvider**<br><br>Провайдер поисковых подсказок. Может быть задан объектом, реализующим интерфейс **ISuggestProvider**, либо стандартным значением "yandex#map". |
| options.results | 5 | Тип: Number<br><br>Максимальное количество показываемых подсказок. |
| options.width | — | Тип: Number<br><br>Ширина панели подсказок. По умолчанию совпадает с шириной HTML-элемента input, к которому панель привязана. |
| options.zIndex | 40000 | Тип: Number<br><br>z-index для dom-элемента панели подсказок. |

* Обязательный параметр/опция.

**Пример:**

```
<input type="text" id="suggest"/>
<script src="//api-maps.yandex.ru/2.1/?
lang=ru_RU&load=SuggestView&onload=onLoad"></script>
<script>
function onLoad (ymaps) {
    var suggestView = new ymaps.SuggestView('suggest');
}
</script>
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| state | data.Manager | Состояние панели поисковых подсказок. Имена полей, доступных через метод data.Manager.get:<br>• request: String — текущий активный запрос;<br>• items: Object[] — массив поисковых подсказок (объектов с полями value и displayName);<br>• activeIndex: Number\|null — индекс текущей активной подсказки, выделенная пользователем путём наведения мыши или с клавиатуры, или null, если ни одна подсказка не активка;<br>• panelClosed: Boolean — признак, была ли панель закрыта пользователем путём нажатия клавиши ESC или выбора одной из подсказок. |

## События

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| select | Пользователь выбрал одну из поисковых подсказок. Имена полей, доступных через метод Event.get<br>• `item` — поисковая подсказка, объект с полями displayName и value. |

## Методы

| Имя | Описание |
|---|---|
| destroy() | Уничтожает панель подсказок. |

## Описание полей

### state

```
{data.Manager} state
```

Состояние панели поисковых подсказок. Имена полей, доступных через метод data.Manager.get:

- request: String — текущий активный запрос;
- items: Object[] — массив поисковых подсказок (объектов с полями value и displayName);
- activeIndex: Number|null — индекс текущей активной подсказки, выделенная пользователем путём наведения мыши или с клавиатуры, или null, если ни одна подсказка не активка;
- panelClosed: Boolean — признак, была ли панель закрыта пользователем путём нажатия клавиши ESC или выбора одной из подсказок.

## Описание событий

### select

Пользователь выбрал одну из поисковых подсказок. Имена полей, доступных через метод Event.get

- `item` — поисковая подсказка, объект с полями displayName и value.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### destroy

```
destroy()
```

Уничтожает панель подсказок.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# template

## template.filtersStorage

Статический объект.

Экземпляр класса **util.Storage**

Хранилище фильтров для шаблонов. Фильтры из хранилища можно использовать во всех макетах, созданных с помощью **templateLayoutFactory**. По умолчанию в хранилище объявлены следующие ключи для фильтров:

- default — позволяет устанавливать значения по умолчанию. Например, так: {{ ***properties.header***| **default:*"Заголовок"*** }}

Методы

```
template.filtersStorage()
```

**Примеры:**

**1.**

```
// Напишем простой фильтр, который будет преобразовывать
// дату в формате «dd.mm.yyyy» в формат «dd month yyyy».
// Для этого нужно создать функцию-фильтр, которая возвратит новое значение.

// При вызове фильтра в функцию будут переданы следующие аргументы:
// менеджер данных шаблона data.Manager, значение и значение, заданное фильтру.
var dateFilter = function (data, dateString, filterValue) {
    var months = [
            'january', 'february', 'march', 'april',
            'may', 'june', 'july', 'august',
            'september', 'october', 'november', 'december'
        ],
        date = dateString.split('.');

    date[1] = months[parseInt(date[1], 10)];
    return date.join(' ');
};

ymaps.template.filtersStorage.add('date', dateFilter);

// После можно использовать в шаблонах конструкции вида
// {{ "21.10.2014"|date }} значение будет "21 october 2014".
```

**2.**

```
// Напишем фильтр, который будет находить и заменять подстроки в тексте.
// Формат значений для замены в фильтре «subString_newSubString».

// При вызове фильтра в функцию будут переданы следующие аргументы:
// менеджер данных шаблона data.Manager, текст и значение, заданное фильтру.
var replaceFilter = function (data, text, replace) {
    replace = replace.trim();
    // Удаляем кавычки.
    replace = replace.slice(1, replace.length - 1);

    // Будем находить в тексте часть, указанную до «_» и заменять её на то, что
указано после.
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
    var values = replace.split('_'),
        from = new RegExp(values[0], 'g'),
        to = values[1];

    return text.replace(from, to);
};

// Теперь можно использовать в шаблонах конструкции вида
// {{"text test replace"|replace: "test_replaced test" }} значение будет "text
replaced test replace".
```

**3.**

```
// В данном примере в макет добавляется значение опции "colorClass" и значение
свойства "header".
// Если значения свойства "header" нет, то вставляется строка "Заголовок".
var LayoutClass = ymaps.templateLayoutFactory.createClass(
    '<h1 class="{{ options.colorClass }}">' +
        '{{ properties.header|default:"Заголовок" }}' +
    '</h1>'
);
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# Template

Шаблонизатор. В API Яндекс.Карт поддерживается базовый синтаксис языков шаблонов Twig/Django Templates. Поддерживаются следующие операции:

- Подстановка значения — **{{ *имя_поля* }}**.
- Если запрашиваемое поле данных отсутствует или имеет пустое значение, то можно предусмотреть стандартное значение — **{{ *имя_поля*|default:*значение_по_умолчанию* }}** Значением по умолчанию может быть число, строка (записывается в кавычках) или другое поле данных.
- По умолчанию значение обрабатывается функцией escape для предотвращения XSS уязвимости. Для отмены поведения необходимо добавить фильтр **"raw"** — **{{ *имя_поля*|default:*значение_по_умолчанию*|raw }}**.
- С помощью **template.filtersStorage** можно создавать свои фильтры и использовать их аналогично описанным выше.
- Для вставки подмакета необходимо использовать конструкцию **{% include *имя_поля_или_ключ* %}**. Шаблонизатор, встретив такую конструкцию, попытается использовать значение в поле как ключ вложенного макета.
- Условие записывается в виде:
  **{% if *условие* %} ... {% else %} ... {% endif %}** или Блок **else** или **elseif** можно опустить. Внутри блоков **if**, **else** и **elseif** можно использовать любые конструкции языка шаблонов.
- Для выполнения итерации по массиву или по объекту нужно использовать конструкцию **for**.
  **{% for *значение* in *массив_или_хеш* %} ... {% endfor %}**. Внутри блока **for** можно использовать любые конструкции языка шаблонов.
- Для получения номера итерации в массиве или имени поля в хеше нужно использовать конструкцию **{% for key, value in *массив_или_хеш* %} ... {% endfor %}**.

Конструктор | Методы

## Конструктор

```
Template(text)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| text * | — | Тип: String<br><br>Строка-шаблон. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Получим имя пользователя из менеджера данных data.Manager.
// Если имя не задано, то в результате получится строка «Незарегистрированный
пользователь».
var data = new ymaps.data.Manager({
         user: {
             name: "Виктор",
             age: 25
         },
         home: [55.72725771214265, 37.640390506634006]
     }),
     template = new ymaps.Template('{{ user.name |default: "Незарегистрированный
пользователь"}}');

var result = template.build(data);
console.log(result.text); // Выведет в консоль «Виктор».
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

2.

```
// У нас есть 3 группы пользователей и в зависимости от группы нужно вывести одно
из приветствий.
var data = new ymaps.data.Manager({
         groups: {
             administrator: {
                 id: 1,
                 name: "администратор"
             },
             moderator: {
                 id: 2,
                 name: "модератор"
             },
             user: {
                 id: 3,
                 name: "пользователь"
             }
         },
         userGroupId: 2
     }),
     template = new ymaps.Template('Привет, \
         {% if (userGroupId == 1) %}{{ groups.administrator.name }}\
         {% elseif (userGroupId == 2) %}{{ groups.moderator.name }}\
         {% elseif (userGroupId == 3) %}{{ groups.user.name }}\
         {% else %}гость{% endif %}!'
     );

var result = template.build(data);
console.log(result.text); // Выведет в консоль «Привет, модератор!».
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **build**(data) | Object | Возвращает объект с полями:<br>• {String} text — результат выполнения шаблонизации;<br>• {Object[]} renderedValues — массив с использованными данными из менеджера.<br>. |

## Описание методов

### build

```
{Object} build(data)
```

**Возвращает** объект с полями:

- {String} text — результат выполнения шаблонизации;
- {Object[]} renderedValues — массив с использованными данными из менеджера.

.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data* | — | Тип: **IDataManager**<br>Менеджер данных. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

**Пример:**

```
// Получим адрес дома из имеющихся координат и выведем по шаблону
// всех его жильцов в формате: «имя: возраст».
var data = new ymaps.data.Manager({
         users: [
             {name: "Виталий", age: 40},
             {name: "Георгий", age: 20}
         ],
         home: {
             coords: [55.736652, 37.620589],
             address: null
         }
     }),
     template = new ymaps.Template('{{home.address}}: <ul>{% for user in users
%}<li>{{user.name}}: {{user.age}}</li>{% endfor %}</ul>');

// Сделаем обратное геокодирование с помощью geocode.
ymaps.geocode(data.get('home.coords')).then(function (res) {
     var address = res.geoObjects.get(0).properties.get('name');
     // Установим в менеджер полученный адрес.
     data.set('home.address', address);

     // Соберём шаблон из имеющихся данных.
     var result = template.build(data);
     // Выведем результат в консоль.
     console.log(result.text);
});
```

## templateLayoutFactory

Статический объект.

Фабрика для создания класса макета из текстового шаблона. Позволяет создавать классы, реализующие интерфейс **ILayout**, используя язык шаблонов. В API Яндекс.Карт поддерживается базовый синтаксис языков шаблонов Twig/Django Templates. Более подробно о синтаксисе можно прочитать в описании **Template**.

**См.: layout.templateBased.Base**

Методы

```
templateLayoutFactory()
```

**Примеры:**

**1.**

```
// В данном примере в макет добавляются значение опции "colorClass" и значение
свойства "header".
// Если значения свойства "header" нет, то вставляется строка "Заголовок".
var LayoutClass = ymaps.templateLayoutFactory.createClass(
     '<h1 class="{{ options.colorClass }}">' +
         '{{ properties.header|default:"Заголовок" }}' +
     '</h1>'
);
```

**2.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// В зависимости от значения опции "width" подключается один из макетов.
var LayoutClass = ymaps.templateLayoutFactory.createClass(
     '{% if options.width > 200 %}' +
         // Необходимый макет будет найден в опциях.
         '{% include options.wideLayout %}' +
     '{% else %}' +
         // Непосредственно записываем ключ.
         '{% include "cluster#balloonCarousel" %}' +
     '{% endif %}'
);
```

**3.**

```
// Вывод в макет балуна массива имен.
var CustomLayoutClass = ymaps.templateLayoutFactory.createClass(
     '<ul>' +
     '{% for name in properties.names %}' +
         // Переменная name "видна" только в блоке for ... endfor
         '<li>{{ name }}</li>' +
     '{% endfor %}' +
     '</ul>'
 );

var placemark = new ymaps.Placemark([54.83, 37.11], {
     names: ['Logan', 'Sofia', 'Mason', 'Layla']
}, {
     balloonContentLayout: CustomLayoutClass
});
```

**4.**

```
// Получение имени полей.
var CustomLayoutClass = ymaps.templateLayoutFactory.createClass(
 '<ul>' +
     '{% for key, value in properties.hash %}' +
         '<li>{{ key }} {{ value }}</li>' +
     '{% endfor %}' +
 '</ul>'
);

var placemark = new ymaps.Placemark([54.83, 37.11], {
     hash: { key1: "value1", key2: "value2", key3: "value3" }
}, {
     balloonContentLayout: CustomLayoutClass
});
```

## Методы

| Имя | Статический | Возвращает | Описание |
|---|---|---|---|
| templateLayoutFactory.**createClass**(template[, overrides, staticMethods) | ✔ | Function | Возвращает конструктор макета. Созданный класс будет унаследован от класса **layout.templateBased.Base** с переопределением списка методов, указанных в overrides. |

## Описание методов

### createClass

```
{Function}    <static>  templateLayoutFactory.createClass(template[,  overrides,
staticMethods)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** конструктор макета. Созданный класс будет унаследован от класса **layout.templateBased.Base** с переопределением списка методов, указанных в overrides.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| template* | — | Тип: String<br><br>Шаблон html-содержимого макетов. |
| overrides | — | Тип: Object<br><br>Переопределения родительских методов. Можно переопределять/дополнять методы build, clear и rebuild. |
| staticMethods* | — | Тип: |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# traffic

## traffic.provider

### traffic.provider.Actual

Расширяет **ITrafficProvider**.

Провайдер актуальных пробок. Доступен в хранилище провайдеров под ключом 'traffic#actual'.

Конструктор | Поля | События | Методы

#### Конструктор

```
traffic.provider.Actual([options[, state]])
```

Создает провайдер актуальных пробок.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции провайдера. Опции для слоев провайдера задаются через опции глобальной коллекции слоев **Map.layers**.<br>• Опции для картиночного слоя **Layer** задаются с префиксом 'trafficImage';<br>• Опции для слоя хотспотов **hotspot.Layer** задаются с префиксом 'trafficJam';<br>• Опции для слоя инфоточек задаются с префиксом 'trafficInfo'. Слой инфоточек является экземпляром класса hotspot.Layer. |
| options.autoUpdate | true | Тип: Boolean<br><br>Флаг, включающий автоматическое обновление данных о пробках. Автоматическое обновление происходит только в том случае, если на карте каждые 4 минуты возникает событие mousemove. Если событие не возникает, пробки перестают обновляться до появления нового события. |
| state | — | Тип: Object<br><br>Состояние провайдера. |
| state.infoLayerShown | false | Тип: Boolean<br><br>Флаг, включающий отображение слоя дорожных событий. |

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Создадим провайдер пробок "сейчас" с включенным слоем дорожных событий
// и разместим его на карте.
var actualProvider = new ymaps.traffic.provider.Actual({}, {infoLayerShown:
true});
actualProvider.setMap(myMap);

// Запретим показ балунов по клику для слоя инфоточек.
myMap.layers.options.set({
    // Название опции сформировано из опции хотспотного слоя
    // 'openBalloonOnClick' путем добавления префикса 'trafficInfo'.
    trafficInfoOpenBalloonOnClick: false
});

// ...
// Удаление провайдера с карты.
actualProvider.setMap(null);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| **state** | **data.Manager** | Состояние провайдера. Имена полей, доступных через метод data.Manager.get:<br>• isInited — признак, готов ли провайдер предоставлять данные;<br>• infoLayerShown — признак, показан ли слой дорожных событий;<br>• timestamp — текущее время во временной зоне UTC+0 в секундах;<br>• localtime — локальное время, для которого сервер в текущий момент отдает данные, в формате 'ЧЧ:ММ';<br>• level — уровень загруженности дорог в баллах — от 0 до 10;<br>• isotime — строка, содержащая текущую дату в формате "YYYY-MM-DDThh:mm:ss±hhmm". |

## События

| Имя | Описание |
|---|---|
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getMap**() | **Map** \| null | Возвращает ссылку на карту.<br>Унаследован от **ITrafficProvider**. |
| **setMap**(Ссылка) | | |
| **update**() | | Шлёт запрос на обновление пробок. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание полей

### state

```
{data.Manager} state
```

Состояние провайдера. Имена полей, доступных через метод data.Manager.get:

- isInited — признак, готов ли провайдер предоставлять данные;
- infoLayerShown — признак, показан ли слой дорожных событий;
- timestamp — текущее время во временной зоне UTC+0 в секундах;
- localtime — локальное время, для которого сервер в текущий момент отдает данные, в формате 'ЧЧ:ММ';
- level — уровень загруженности дорог в баллах — от 0 до 10;
- isotime — строка, содержащая текущую дату в формате "YYYY-MM-DDThh:mm:ss±hhmm".

**Пример:**

```
var actualProvider = new ymaps.traffic.provider.Actual();
actualProvider.setMap(myMap);
actualProvider.state.events.add('change', function () {
    if (actualProvider.state.get('isInited')) {
        alert('Провайдер готов предоставлять данные.');
    }
});
```

## Описание методов

### update

```
update()
```

Шлёт запрос на обновление пробок.

**Пример:**

```
var trafficControl = new ymaps.control.TrafficControl({shown: true});
map.controls.add(trafficControl);
function updateProvider () {
     trafficControl.getProvider('traffic#actual').update();
}
// Будем слать запрос на обновление данных каждые 4 минуты.
window.setInterval(updateProvider, 4 * 60 * 1000);
```

# traffic.provider.Archive

Расширяет **ITrafficProvider**.

Провайдер архива пробок. Позволяет показывать состояние пробок, типичное для данного региона в конкретный день недели и конкретное время суток.

Конструктор | Поля | События | Методы

## Конструктор

```
traffic.provider.Archive([options[, state]])
```

Создает экземпляр провайдера архива пробок.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции провайдера. Опции для слоев провайдера задаются через опции глобальной коллекции слоев **Map.layers**.<br><br>• Опции для картиночного слоя **Layer** задаются с префиксом 'trafficImage';<br>• Опции для слоя хотспотов **hotspot.Layer** задаются с префиксом 'trafficJam'. |
| options.showCurrentTimeFirst | true | Тип: Boolean<br><br>При первом показе архивных данных выставить время, близкое к текущему. |
| state | — | Тип: Object<br><br>Состояние провайдера. |
| state.timestamp | — | Тип: Number<br><br>Время, для которого показываются пробки "обычно". Это время, прошедшее с 00:00 понедельника до требуемого времени в секундах. Должно быть кратно 60 * 15 = 900, так как данные на сервере доступны для моментов времени с разницей в 15 минут. Время задается для нулевого часового пояса (UTC+0). |

**Пример:**

```
// Создадим провайдер пробок "обычно" и зададим ему timestamp для 17:47 среды
// в нулевом часовом поясе. Обратите внимание, что местное время будет зависеть
// от расположения центра карты.
// К примеру, 17:47 в нулевом часовом поясе будет соответствовать 21:47 в
Москве.

// Вычислим значение параметра timestamp для требуемого времени.
var timestamp = 2 * 24 * 60 * 60 + // дважды по 24 часа - это время за понедельник
и вторник
        17 * 60 * 60 + // 17 часов прошло с 00:00 среды
        45 * 60, // поскольку время должно быть кратно 15 минутам, берем не 47
минут, а 45.
    archiveProvider = new ymaps.traffic.provider.Archive({
        // Запретим при первом открытии выставлять время, близкое к текущему
        showCurrentTimeFirst: false
    }, {
        // Зададим начальное время самостоятельно.
        timestamp: timestamp
    });
archiveProvider.setMap(map);
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Запретим показ всплывающих подсказок для слоя пробок.
myMap.layers.options.set({
    // Название опции сформировано из опции хотспотного слоя
    // 'openHintOnHover' путем добавления префикса 'trafficJam'.
    trafficJamOpenHintOnHover: false
});

// ...
// Удаление провайдера с карты.
archiveProvider.setMap(null);
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |
| state | data.Manager | Состояние провайдера. Имена полей, доступных через метод data.Manager.get:<br>• isInited — признак, готов ли провайдер предоставлять данные;<br>• timeZone — смещение по времени для текущего временного пояса относительно UTC+0. Измеряется в секундах.;<br>• dst — признак перехода на зимнее/летнее время. При dst='dst' время летнее;<br>• timestamp — текущее время во временной зоне UTC+0 в секундах;<br>• localtime — локальное время, которое возвращает в ответе сервер;<br>• level — уровень загруженности дорог в баллах — от 0 до 10. |

## События

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map \| null | Возвращает ссылку на карту.<br>Унаследован от ITrafficProvider. |
| getTime() | Object \| null | Возвращает день недели, часы и минуты состояния провайдера с учетом часового пояса и перехода на зимнее/летнее время. То есть то время, которое пользователь видит в элементе управления пробками. |
| setMap(Ссылка) | | |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **setTime**(time[, callback]) | | Позволяет задавать время для архивного провайдера в минутах, часах и днях недели. Выставляет локальное время только после того, как провайдер инициализирует поля timeZone и dst.<br>• timeZone — поле, которое показывает, в каком часовом поясе на данный момент находится центр карты. При перемещении центра карты из одного часового пояса в другой локальное время может изменяться.<br>• dst — признак перехода на летнее/зимнее время. При dst='dst' время летнее.<br>Постоянной составляющей времени для провайдера пробок "обычно" служит поле timestamp, которое отражает текущее время в нулевом часовом поясе (UTC+0). При переходе из одного часового пояса в другой timestamp не изменяется. Значения полей timestamp, dst и timeZone можно получить через поле **traffic.provider.Archive.state**. |

## Описание полей

### state

```
{data.Manager} state
```

Состояние провайдера. Имена полей, доступных через метод data.Manager.get:

- isInited — признак, готов ли провайдер предоставлять данные;
- timeZone — смещение по времени для текущего временного пояса относительно UTC+0. Измеряется в секундах.;
- dst — признак перехода на зимнее/летнее время. При dst='dst' время летнее;
- timestamp — текущее время во временной зоне UTC+0 в секундах;
- localtime — локальное время, которое возвращает в ответе сервер;
- level — уровень загруженности дорог в баллах — от 0 до 10.

**Пример:**

```
var archiveProvider = new ymaps.traffic.provider.Archive();
archiveProvider.setMap(myMap);
archiveProvider.state.events.add('change', function () {
    if (archiveProvider.state.get('isInited')) {
        alert('Провайдер готов предоставлять данные.');
    }
});
```

## Описание методов

### getTime

```
{Object | null} getTime()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Возвращает день недели, часы и минуты состояния провайдера с учетом часового пояса и перехода на зимнее/летнее время. То есть то время, которое пользователь видит в элементе управления пробками.

**Возвращает** объект с полями

- dayOfWeek — Сокращенное название дня недели. 'mon', 'tue', 'wed', 'thu', 'fri', 'sat', 'sun';
- hours — часы;
- minutes — минуты.

Если центр карты находится в точке, для которой не удается опередилить часовой пояс, функция возвращает null — если мы не знаем, в каком часовом поясе находимся, мы не можем узнать местное время.

## setTime

```
setTime(time[, callback])
```

Позволяет задавать время для архивного провайдера в минутах, часах и днях недели. Выставляет локальное время только после того, как провайдер инициализирует поля timeZone и dst.

- timeZone — поле, которое показывает, в каком часовом поясе на данный момент находится центр карты. При перемещении центра карты из одного часового пояса в другой локальное время может изменяться.
- dst — признак перехода на летнее/зимнее время. При dst='dst' время летнее.

Постоянной составляющей времени для провайдера пробок "обычно" служит поле timestamp, которое отражает текущее время в нулевом часовом поясе (UTC+0). При переходе из одного часового пояса в другой timestamp не изменяется. Значения полей timestamp, dst и timeZone можно получить через поле **traffic.provider.Archive.state**.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| time* | — | Тип: Object<br><br>Объект с задаваемыми параметрами. |
| time.dayOfWeek | — | Тип: String<br><br>Сокращенное название дня недели. 'mon', 'tue', 'wed', 'thu', 'fri', 'sat', 'sun'. |
| time.hours | — | Тип: Number<br><br>Часы. |
| time.minutes | — | Тип: Number<br><br>Минуты. |
| callback | — | Тип: Function<br><br>Функция, которая вызывается после того, как было выставлено время. Принимает на вход хэш с выставленными данными. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Создадим элемент управления, который сразу показывает на карте
// провайдер пробок "обычно".
var trafficControl = new ymaps.control.TrafficControl({
        shown: true,
        providerKey: 'traffic#archive'
    });
map.controls.add(trafficControl);
// Как только провайдер получит данные о текущей временной зоне,
// будет выставлено локальное время.
trafficControl.getProvider('traffic#archive').setTime({
    dayOfWeek: 'fri',
    hours: 9,
    minutes: 15
}, function (time) {
    alert('Локальное время ' + time.hours + ':' + time.minutes + '
выставлено!');
});
```

# traffic.provider.Forecast

Расширяет **ITrafficProvider**.

Провайдер прогноза пробок. Доступен в хранилище провайдеров под ключом 'traffic#forecast'.

Конструктор | Поля | События | Методы

## Конструктор

```
traffic.provider.Forecast([options[, state]])
```

Создает провайдер прогноза пробок.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции провайдера. Опции для слоев провайдера задаются через опции глобальной коллекции слоев **Map.layers**.<br>• Опции для картиночного слоя **Layer** задаются с префиксом 'trafficImage';<br>• Опции для слоя хотспотов **hotspot.Layer** задаются с префиксом 'trafficJam'. |
| options.autoUpdate | true | Тип: Boolean<br><br>Флаг, включающий автоматическое обновление данных о пробках. Автоматическое обновление происходит только в том случае, если на карте каждые 4 минуты возникает событие mousemove. Если событие не возникает, пробки перестают обновляться до появления нового события. |
| state | — | Тип: Object<br><br>Состояние провайдера. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| state.timeOffset | 900 | Тип: Boolean<br><br>временной интервал между текущим временем и временем прогноза. Измеряется в секундах и кратен 15 минутам (900 секундам).<br><br>`// Создадим провайдер пробок с прогнозом на полчаса вперед`<br>`// и разместим его на карте.`<br>`var forecastProvider = new ymaps.traffic.provider.Forecast({}, {timeOffset: 30 * 60});`<br>`forecastProvider.setMap(myMap);`<br><br>`// Запретим показ всплывающих подсказок для пробок.`<br>`myMap.layers.options.set({`<br>`// Название опции сформировано из опции хотспотного слоя`<br>`// 'openHintOnHover' путем добавления префикса 'trafficJam'.`<br><br>`trafficJamOpenHintOnHover: false`<br>`});`<br>`// ...`<br>`// Удаление провайдера с карты.`<br>`forecastProvider.setMap(null);` |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |
| **state** | **data.Manager** | Состояние провайдера. Имена полей, доступных через метод data.Manager.get:<br><br>• isInited — признак, готов ли провайдер предоставлять данные;<br>• timestamp — текущее время в формате unix timestamp в секундах;<br>• localtime — локальное время, для которого сервер в текущий момент отдает данные, в формате 'ЧЧ:ММ';<br>• level — уровень загруженности дорог в баллах — от 0 до 10.<br>• timeOffset — временной интервал между текущим временем и временем прогноза. Измеряется в секундах и кратен 15 минутам (900 секундам)<br>• isotime — строка, содержащая текущую дату в формате "YYYY-MM-DDThh:mm:ss±hhmm". |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### События

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map \| null | Возвращает ссылку на карту.<br>Унаследован от ITrafficProvider. |
| getTime() | Object \| null | Возвращает время, которое пользователь видит в элементе управления пробками. |
| setMap(Ссылка) | | |

## Описание полей

### state

```
{data.Manager} state
```

Состояние провайдера. Имена полей, доступных через метод data.Manager.get:

- isInited — признак, готов ли провайдер предоставлять данные;
- timestamp — текущее время в формате unix timestamp в секундах;
- localtime — локальное время, для которого сервер в текущий момент отдает данные, в формате 'ЧЧ:ММ';
- level — уровень загруженности дорог в баллах — от 0 до 10.
- timeOffset — временной интервал между текущим временем и временем прогноза. Измеряется в секундах и кратен 15 минутам (900 секундам)
- isotime — строка, содержащая текущую дату в формате "YYYY-MM-DDThh:mm:ss±hhmm".

## Описание методов

### getTime

```
{Object | null} getTime()
```

Возвращает время, которое пользователь видит в элементе управления пробками.

**Возвращает** объект с полями

- dayOfWeek — Сокращенное название дня недели. 'mon', 'tue', 'wed', 'thu', 'fri', 'sat', 'sun';
- hours — часы;
- minutes — минуты.

Если центр карты находится в точке, для которой не удается опеределить часовой пояс, функция возвращает null — если мы не знаем, в каком часовом поясе находимся, мы не можем узнать местное время.

# traffic.provider.storage

Статический объект.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Экземпляр класса **util.Storage**

Хранилище провайдеров.

Методы

```
traffic.provider.storage()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# util

## util.AsyncStorage

Расширяет **util.Storage**.

Хранилище, которое предоставляет асинхронный доступ к значениям по ключам.

Конструктор | Методы

### Конструктор

```
util.AsyncStorage()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | **util.Storage** | Добавляет объект в хранилище.<br>Унаследован от **util.Storage**. |
| **define**(key[, depends, resolveCallback[, context]]) | **util.AsyncStorage** | Определение в хранилище асинхронного значения. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка.<br>Унаследован от **util.Storage**. |
| **isDefined**(key) | Boolean | Проверка доступности ключа в хранилище. |
| **remove**(key) | **util.Storage** | Удаляет пару "ключ: значение" из хранилища.<br>Унаследован от **util.Storage**. |
| **require**(keys[, successCallback[, errorCallback[, context]]]) | **vow.Promise** | Асинхронный запрос на получение значений из хранилища. |

### Описание методов

#### define

```
{util.AsyncStorage} define(key[, depends, resolveCallback[, context]])
```

Определение в хранилище асинхронного значения.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br>Ключ, по которому можно будет произвести асинхронный вызов. |
| depends | — | Тип: String[] |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Массив ключей значений из текущего хранилища, которые должны быть проинициализированы перед текущим. Этот аргумент может быть пропущен. |
| resolveCallback* | — | Тип: Function<br><br>Функция, которая определяет значение, доступное по ключу. Первым аргументом в resolveCallback поступит provide-функция, в которую нужно будет передать значение. Вызов provide-функции может быть отложен во времени. Последующие аргументы — это значения из хранилища, указанные в зависимостях. Порядок модулей будет соответствовать порядку в массиве depends. |
| context | — | Тип: Object<br><br>Контекст исполнения функции. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
asyncStorage
    .define('red', function (provide) {
        provide('#FF0000');
     });
```

**2.**

```
asyncStorage
    .define('green', function (provide) {
        // Вызов provide-функции может быть асинхронным.
        setTimeout(function () {
            provide('#008000');
        }, 400);
    });
```

**3.**

```
asyncStorage
    .define('yellow', function (provide) {
         provide('#FFFF00');
    })
    // Для определения значения ключа 'violet' необходимо значение ключа
'yellow'.
    .define('violet', ['yellow'], function (provide, yellow) {
        console.log(yellow); // #FFFF00
        setTimeout(function () {
            provide('#9B30FF');
        }, 400);
    });
```

**4.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
var asyncStorage = new ymaps.util.AsyncStorage();
asyncStorage
    .define('red', function (provide) {
        provide('#FF0000');
     })
    .define('green', function (provide) {
        setTimeout(function () {
            provide('#008000');
        }, 400);
    })
    .define('yellow', function (provide) {
         provide('#FFFF00');
    })
    .define('violet', ['yellow'], function (provide, yellow) {
        setTimeout(function () {
            provide('#9B30FF');
        }, 400);
    });
// Запрашиваем
asyncStorage.require(['red', 'green', 'violet'])
    .spread(function (red, green, violet) {
         console.log(red, green, violet); // #FF0000 #008000 #9B30FF
         // После перевого асинхронного обращения значения будут доступны через
синхронный интерфейс.
         console.log(asyncStorage.get('red'), asyncStorage.get('green'),
asyncStorage.get('violet')); // #FF0000 #008000
         // Значение ключа 'yellow' теперь тоже содержится в хранилище, так как
было необходимо для определения 'violet'.
         console.log(asyncStorage.get('yellow')); // #FFFF00
    });
```

## isDefined

```
{Boolean} isDefined(key)
```

Проверка доступности ключа в хранилище.

**Возвращает** true — ключ определ, false — нет.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Ключ значения. |

* Обязательный параметр/опция.

**Пример:**

```
if (asyncStorage.isDefined('red')) {
    asyncStorage.require('red')
        .spread(function (red) {
            // ...
        });
}
```

## require

```
{vow.Promise} require(keys[, successCallback[, errorCallback[, context]]])
```

Асинхронный запрос на получение значений из хранилища.

**Возвращает** объект-promise, который предоставляет асинхронный доступ к значению.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| keys* | — | Тип: String \| String[]<br><br>Ключ или массив ключей. |
| successCallback | — | Тип: Function<br><br>Функция обратного вызова, которая будет вызвана после получения всех значений. В функцию как аргументы будут переданы значения из хранилища. Порядок аргументов будет соответствовать порядку в массиве keys. |
| errorCallback | — | Тип: Function<br><br>Функция обратного вызова, которая будет вызвана в случае ошибки. В функцию будет передан объект ошибки. |
| context | — | Тип: Object<br><br>Контекст исполнения функции обратного вызова. |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
asyncStorage.require(['green'])
    .spread(function (green) {
        // ...
    });
```

**2.**

```
var asyncStorage = new ymaps.util.AsyncStorage();
asyncStorage
    .define('red', function (provide) {
        provide('#FF0000');
     })
    .define('green', function (provide) {
        setTimeout(function () {
            provide('#008000');
        }, 400);
    });
// Запрашиваем
asyncStorage.require(['red', 'green'])
    .spread(function (red, green) {
         console.log(red, green); // #FF0000 #008000
    });
```

## util.augment

Статическая функция.

Базовая функция, реализующая наследование в JavaScript. Реализует наследование прототипа без исполнения конструктора родителя. К дочернему классу добавляется поле 'superclass', указывающее на прототип родительского класса, и поле 'constructor', которое указывает на конструктор класса. Через поле 'constructor' объекта 'superclass' можно обратится к конструктору родительского класса.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает:** Прототип дочернего класса. (тип Object).

```
util.augment(ChildClass, ParentClass, override)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| ChildClass* | — | Тип: Function<br><br>Дочерний класс. |
| ParentClass* | — | Тип: Function<br><br>Родительский класс. |
| override* | — | Тип: Object<br><br>Набор дополнительных полей и функций, которые будут приписаны к прототипу дочернего класса. |

* Обязательный параметр/опция.

**Пример:**

```
// Родительский класс
var ParentClass = function (param1, param2) {
    this.param1 = param1;
    this.param2 = param2;
};

ParentClass.prototype = {
    foo: function () {
        alert('Parent!');
    }
};
// Дочерний класс
var ChildClass = function (param1, param2, param3) {
    // Вызываем конструктор родителя
    ChildClass.superclass.constructor.call(this, param1, param2);
    this._param3 = param3;
}

// наследуем ChildClass от ParentClass
ymaps.util.augment(ChildClass, ParentClass, {
    // переопределяем в наследнике метод foo
    foo: function () {
        // Вызываем метод родительского класса
        ChildClass.superclass.foo.call(this);
        alert('Child!');
    }
});
```

## util.bind

Статическая функция.

Связывает переданную функцию с переданным контекстом исполнения.

```
util.bind(callback, context)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция. |
| context* | — | Тип: Object<br><br>Контекст исполнения. |

* Обязательный параметр/опция.

**Пример:**

```
var myObject = {
        name: 'test!'
    };
ymaps.geocode.load('Москва')
    then(ymaps.util.bind(function (res) {
        alert(this.name); // test!
    }, myObject));
```

# util.bounds

Статический объект.

Набор статических методов для работы с прямоугольными областями, представленными в виде двух противоположных точек в координатной системе проекции.

Методы

```
util.bounds()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **areIntersecting**(bounds1, bounds2[, projection]) | Boolean | Определяет пересекаются ли две прямоугольные области. |
| **containsBounds**(outer, inner[, projection]) | Boolean | Определяет содержат ли прямоугольная область другую прямоугольную область полностью. |
| **containsPoint**(bounds, point[, projection]) | Boolean | Определяет содержат ли прямоугольная область точку. |
| **fromBounds**(sourceBounds[, projection]) | Number[][] | Вычисляет прямоугольную область, в которую попадают все переданные. |
| **fromGlobalPixelBounds**(pixelBounds, zoom[, projection]) | Number[][] | Производит перевод прямоугольной области из пиксельных координат в геокоординаты с учетом масштаба. |
| **fromPoints**(points[, projection]) | Number[][] | Вычисляет минимальную прямоугольную область, в которую попадают все переданные точки. |
| **getCenter**(bounds[, projection]) | Number[] | Вычисляет центр прямоугольникой области в координатной системе проекции. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getCenterAndZoom**(bounds, containerSize[, projection[, params]]) | Object | Вычисляет центр и уровень масштабирования, которые необходимо установить карте для того, чтобы полностью отобразить переданную область. |
| **getIntersections**(bounds1, bounds2[, projection]) | Number[][][] | Возвращает все пересечения двух прямоугольных областей. В случае, если данные передаются в геокоординатах, то пересечений может быть больше одного.<br>Если области не пересекаются, то возвращается пустой массив. |
| **getSize**(bounds[, projection]) | Number[] | Вычисляет размеры прямоугольной области в координатной системе проекции. |
| **toGlobalPixelBounds**(geoBounds, zoom[, projection]) | Number[][] | Производит перевод границ из геокоординат в глобальные пиксели с учетом масштаба. |

## Описание методов

### areIntersecting

```
{Boolean} areIntersecting(bounds1, bounds2[, projection])
```

Определяет пересекаются ли две прямоугольные области.

**Возвращает** признак пересечения.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds1* | — | Тип: Number[][]<br>Первая область. |
| bounds2* | — | Тип: Number[][]<br>Вторая область. |
| projection | **projection.wgs84Mercator** | Тип: **IProjection**<br>Проекция. |

* Обязательный параметр/опция.

### containsBounds

```
{Boolean} containsBounds(outer, inner[, projection])
```

Определяет содержат ли прямоугольная область другую прямоугольную область полностью.

**Возвращает** признак попадания.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| outer* | — | Тип: Number[][] |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Внешняя область |
| inner* | — | Тип: Number[][]<br><br>Проверяемая область. |
| projection | **projection.wgs84Mercator** | Тип: **IProjection**<br><br>Проекция. |

* Обязательный параметр/опция.

## containsPoint

```
{Boolean} containsPoint(bounds, point[, projection])
```

Определяет содержат ли прямоугольная область точку.

**Возвращает** признак попадания.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds* | — | Тип: Number[][]<br><br>Внешняя область |
| point* | — | Тип: Number[]<br><br>Проверяемая точка. |
| projection | **projection.wgs84Mercator** | Тип: **IProjection**<br><br>Проекция. |

* Обязательный параметр/опция.

## fromBounds

```
{Number[][]} fromBounds(sourceBounds[, projection])
```

Вычисляет прямоугольную область, в которую попадают все переданные.

**Возвращает** вычисленная область.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| sourceBounds* | — | Тип: Number[][][]<br><br>Массив прямоугольных областей |
| projection | **projection.wgs84Mercator** | Тип: **IProjection**<br><br>Проекция. |

* Обязательный параметр/опция.

## fromGlobalPixelBounds

```
{Number[][]} fromGlobalPixelBounds(pixelBounds, zoom[, projection])
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Производит перевод прямоугольной области из пиксельных координат в геокоординаты с учетом масштаба.

**Возвращает** вычисленные границы в геокоординатах.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pixelBounds* | — | Тип: Number[][]<br><br>Исходные границы. |
| zoom* | — | Тип: Number<br><br>Масштаб. |
| projection | **projection.wgs84Mercator** | Тип: **IProjection**<br><br>Проекция, которая будет использоваться для вычисления геокоординат. |

* Обязательный параметр/опция.

## fromPoints

```
{Number[][]} fromPoints(points[, projection])
```

Вычисляет минимальную прямоугольную область, в которую попадают все переданные точки.

**Возвращает** вычисленная область.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| points* | — | Тип: Number[][]<br><br>Массив точек. |
| projection | **projection.wgs84Mercator** | Тип: **IProjection**<br><br>Проекция. |

* Обязательный параметр/опция.

## getCenter

```
{Number[]} getCenter(bounds[, projection])
```

Вычисляет центр прямоугольникой области в координатной системе проекции.

**Возвращает** точку-центр в координатной системе входящих данных.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds* | — | Тип: Number[][]<br><br>Область. |
| projection | **projection.wgs84Mercator** | Тип: **IProjection**<br><br>Проекция. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

## getCenterAndZoom

```
{Object} getCenterAndZoom(bounds, containerSize[, projection[, params]])
```

Вычисляет центр и уровень масштабирования, которые необходимо установить карте для того, чтобы полностью отобразить переданную область.

**Возвращает** объект с полями center (Number[]) и zoom (Number).

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds* | — | Тип: Number[][]<br><br>Область, заданная в географических координатах. Первая точка содержит минимальные значения широты и долготы, вторая точка — максимальные. |
| containerSize* | — | Тип: Number[]<br><br>Размер контейнера карты. |
| projection | **projection.wgs84Mercator** | Тип: **IProjection**<br><br>Проекция. |
| params | — | Тип: Boolean \| Object<br><br>Параметры или значение опции preciseZoom. |
| params.inscribe | true | Тип: Boolean<br><br>true — вписывать область в карту, false — вписывать карту в область. |
| params.margin | 0 | Тип: Number \| Number[]<br><br>Отступы от границ видимой области карты. Если задано одно число — оно применяется ко всем сторонам. Если задано два — то это горизонтальные и вертикальные отступы соответственно. Если задан массив из 4х чисел, то это отступы top, right, bottom, left. |
| params.preciseZoom | false | Тип: Boolean<br><br>При значении false уровень масштабирования будет округлен в меньшую сторону. |

* Обязательный параметр/опция.

## getIntersections

```
{Number[][][]} getIntersections(bounds1, bounds2[, projection])
```

Возвращает все пересечения двух прямоугольных областей. В случае, если данные передаются в геокоординатах, то пересечений может быть больше одного.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Если области не пересекаются, то возвращается пустой массив.

**Возвращает** прямоугольные области, которые образуются пересечением исходных.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds1* | — | Тип: Number[][]<br>Первая область. |
| bounds2* | — | Тип: Number[][]<br>Вторая область. |
| projection | **projection.wgs84Mercator** | Тип: **IProjection**<br>Проекция. |

* Обязательный параметр/опция.

## getSize

```
{Number[]} getSize(bounds[, projection])
```

Вычисляет размеры прямоугольной области в координатной системе проекции.

**Возвращает** размер области.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds* | — | Тип: Number[][]<br>Область. |
| projection | **projection.wgs84Mercator** | Тип: **IProjection**<br>Проекция. |

* Обязательный параметр/опция.

## toGlobalPixelBounds

```
{Number[][]} toGlobalPixelBounds(geoBounds, zoom[, projection])
```

Производит перевод границ из геокоординат в глобальные пиксели с учетом масштаба.

**Возвращает** вычисленные пиксельные границы.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geoBounds* | — | Тип: Number[][]<br>Исходные границы. |
| zoom* | — | Тип: Number<br>Масштаб. |
| projection | **projection.wgs84Mercator** | Тип: **IProjection** |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Проекция, в координатной системе которой заданы геокоординаты. |

* Обязательный параметр/опция.

# util.cursor

## util.cursor.Accessor

Объект, предоставляющий доступ к добавленному на карту курсору.

Конструктор | Методы

### Конструктор

```
util.cursor.Accessor(key)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key * | — | Тип: String<br><br>Ключ, под которым хранится курсор в хранилище курсоров. |

* Обязательный параметр/опция.

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getKey**() | String | Возвращает текущий ключ для доступа к курсору в хранилище курсоров. |
| **remove**() | | Удаляет курсор с карты. |
| **setKey**() | | Задает новый ключ для доступа к курсору. По этому ключу должен находиться соответствующий курсор в хранилище курсоров. В случае, если данный курсор активен, то он сразу сменится. |

### Описание методов

#### getKey

```
{String} getKey()
```

Возвращает текущий ключ для доступа к курсору в хранилище курсоров.

**Возвращает** String.

#### remove

```
remove()
```

Удаляет курсор с карты.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### setKey

```
setKey()
```

Задает новый ключ для доступа к курсору. По этому ключу должен находиться соответствующий курсор в хранилище курсоров. В случае, если данный курсор активен, то он сразу сменится.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String |

* Обязательный параметр/опция.

## util.cursor.Manager

Менеджер курсоров.

Конструктор | Поля | События | Методы

### Конструктор

```
util.cursor.Manager(element)
```

Менеджер курсоров над DOM-элементом. Используется прямое присваивание через style.cursor.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| element * | — | Тип: HTMLElement<br><br>DOM-элемент, для которого выполняется установка курсоров. |

* Обязательный параметр/опция.

**Пример:**

```
// поставим курсор "стрелка" над картой
var cursor = map.cursors.push('arrow');
setTimeout(function () {
    // через 5 секунд установим курсору другой ключ
    cursor.setKey('grabbing');
    setTimeout(function () {
        // еще через 5 секунд удалим этот курсор с карты
        cursor.remove();
    }, 5000);
}, 5000);
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| events | event.Manager | Менеджер событий менеджера курсоров. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### События

| Имя | Описание |
|---|---|
| change | Изменение курсора. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| push(key) | util.cursor.Accessor | Устанавливает новый курсор и дописывает его в стэк курсоров объекта. |

## Описание полей

### events

```
{event.Manager} events
```

Менеджер событий менеджера курсоров.

## Описание событий

### change

Изменение курсора.

## Описание методов

### push

```
{util.cursor.Accessor} push(key)
```

Устанавливает новый курсор и дописывает его в стэк курсоров объекта.

**Возвращает** объект, предоставляющий доступ к добавленному на карту курсору.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Курсор. Допустимые значения:<br>• "arrow" — стрелка;<br>• "crosshair" — перекрестье;<br>• "grab" — "рука";<br>• "grabbing" — сжатая "рука"<br>• "help" — стрелка со знаком вопроса;<br>• "zoom" — лупа;<br>• "move" — курсор в виде четырех стрелок;<br>• "pointer" — "палец";<br>• "inherit" — унаследовать курсор от родителя. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# util.defineClass

Статическая функция.

Базовая функция, реализующая объявление классов в API Яндекс.Карт. При помощи этой функции можно объявить новый класс, указать у этого класса набор методов и произвести наследование от другого класса.

К дочернему классу приписывается поле superclass, указывающее на прототип родительского класса. Через поле 'constructor' объекта 'superclass' можно обратится к конструктору родительского класса.

**Возвращает:** Класс. (тип Object).

```
util.defineClass(constructor[, parentClass[, override]])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| constructor* | — | Тип: Function<br><br>Конструктор класса. |
| parentClass | — | Тип: Function<br><br>Родительский класс, от которого необходимо произвести наследование. Этот аргумент может быть пропущен. |
| override | — | Тип: Object<br><br>Набор дополнительных полей и функций, которые будут приписаны к прототипу класса. Источников может быть несколько (функция может иметь произвольное число параметров), данные копируются справа налево (последний аргумент имеет наивысший приоритет при копировании). |

* Обязательный параметр/опция.

**Примеры:**

**1.**

```
// Объявление класса с методами.
var MyClass = ymaps.util.defineClass(function () {
   this.field = 'fieldValue';
    }, {
   doSomethingAwesome: function () {
  return 'methodResult';
   },
   stop: function () {

   }
    }),
    object = new MyClass();
    console.log(object.field); // 'fieldValue'
    console.log(object.doSomethingAwesome()); // 'methodResult'
```

**2.**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
// Реализуем собственный класс кнопки на основе наследования от базового класса
кнопки.
// Кнопка по нажатию переключает тип тайлов карты.
var CustomControl = ymaps.util.defineClass(function () {
    // Задаем жесткий набор опций без возможности установки извне.
    var data = {
   data: { content: 'Изменить тип карты' },
   options: {
  selectOnClick: false,
  maxWidth: 150
   }
    };
    CustomControl.superclass.constructor.call(this, data);
}, ymaps.control.Button, {
    // Указываем список методов класса.

    // Переопределяем методы кнопки enable и disable.
    enable: function () {
   // Нужно обязательно вызывать методы базового класса,
   // чтобы не нарушить логику работы кнопки.
   CustomControl.superclass.enable.call(this);
   // Включаем и выключаем поведение кнопки.
   this.events.add('click', this.switchType, this);
    },

    disable: function () {
   this.events.remove('click', this.switchType, this);
   CustomControl.superclass.disable.call(this);
    },

    // Реализуем собственные методы.
    switchType: function () {
   var map = this.getMap();
   if (map) {
  if (map.getType() == 'yandex#map') {
      this.setSatelliteMapType();
  } else {
      this.setSchemeMapType();
  }
   }
    },

    setSchemeMapType: function () {
   var map = this.getMap();
   if (map) {
  map.setType('yandex#map');
   }
    },

    setSatelliteMapType: function () {
   var map = this.getMap();
   if (map) {
  map.setType('yandex#satellite');
   }
    }
});
// Создаем экземпляр нового класса и добавляем его на карту.
var typeSwitcherButton = new CustomControl();
// Код создания карты в данном примере был опущен.
myMap.controls.add(typeSwitcherButton);
// Вызываем метод экземпляра класса.
typeSwitcherButton.setSatelliteMapType();
```

## util.Dragger

Расширяет **IEventEmitter**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Специальный инструмент, который предоставляет механизм для реализации перетаскивания элементов на странице. При использовании следует обратить внимание, что во время работы драггера в системе событий API Яндекс.Карт не будут поступать события mousemove и mouseup.

Конструктор | Поля | События | Методы

## Конструктор

```
util.Dragger([params])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| params | — | Тип: Object<br><br>Параметры драггера. |
| params.autoStartElement | — | Тип: HTMLElement \| **IDomEventEmitter**<br><br>DOM-элемент или реализация интерфейса IDomEventEmitter, нажатие на который запустит драггер. |
| params.byRightButton | false | Тип: Boolean<br><br>Для начала работы драггера при помощи параметра 'autoStartElement' используется правая кнопка мыши. На устройствах без поддержки мыши перетаскивание правой кнопкой не будет работать. |
| params.tremor | 3 | Тип: Number<br><br>Минимальное смещение в пикселях от стартовой точки, необходимое для запуска драггера. |

**Пример:**

```
// Пример перетаскивания DOM-элемента на карту.
// На месте отпускания DOM-элемента создается Placemark.
var myMap = new ymaps.Map('map', {center: [35.76, 37.67], zoom: 5}),
// У DOM-элемента должно быть выставлено CSS-свойство position: absolute.
    element = document.getElementById('someId'),
    dragger = new ymaps.util.Dragger({
        autoStartElement: element
    }),
    draggerEventsGroup = dragger.events.group();

draggerEventsGroup
    .add('start', function (event) {
        var pos = event.get('position');
        positionElement(pos[0], pos[1]);
        console.log('start');
    })
    .add('move', function (event) {
        var pos = event.get('position');
        positionElement(pos[0], pos[1]);
        console.log('move');
    })
    .add('stop', function (event) {
        draggerEventsGroup.removeAll();
        element.parentElement.removeChild(element);
        // Получаем географические координаты по точке окончания работы
драггера.
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
        var placemarkPosition =
myMap.options.get('projection').fromGlobalPixels(
            myMap.converter.pageToGlobal(event.get('position')),
            myMap.getZoom()
        );
        myMap.geoObjects.add(
            new ymaps.Placemark(placemarkPosition)
        );
        console.log('stop');
    });

function positionElement (x, y) {
    element.style.left = x + 'px';
    element.style.top = y + 'px';
}
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |

## События

| Имя | Описание |
|---|---|
| cancel | Отмена работы драггера. Это событие происходит если драггер прекратил свою работу без отправки события util.Dragger.start. К примеру, если было прекращено нажатие без смещения. Экземпляр класса Event. Имена полей, доступных через метод Event.get: |
| move | Изменение положения. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массива вида [pageX, pageY];<br>• delta — разница между положениями текущего и предыдущего события драггера. |
| start | Начало работы драггера. Это событие происходит не в момент нажатия, а при первом измении положения после нажатия. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массива вида [pageX, pageY]. |
| stop | Завершение работы драггера. Это событие не может произойти без события util.Dragger.start. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• position — координаты относительно документа. Массива вида [pageX, pageY];<br>• delta — разница между положениями текущего и предыдущего события драггера. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Останавливает работу драггера и удаляет прослушивание события "mousedown" у элемента autoStartElement. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **isDragging**() | Boolean | Возвращает работает ли сейчас драггер. |
| **start**(event) | | Запускает драггер. Этот метод автоматически вызывается по событию "mousedown" элемента autoStartElement, если он задан. Данный метод можно использовать для инициализации по требованию. К примеру, при получении ответа от сервера. Для корректной работы при инициализации по требованию необходимо чтобы передаваемое событие поддерживало поле 'position' в методе get. |
| **stop**() | | Останавливает драггер. Этот метод можно использовать для досрочной остановки работы драггера. |

## Описание событий

### cancel

Отмена работы драггера. Это событие происходит если драггер прекратил свою работу без отправки события util.Dragger.start. К примеру, если было прекращено нажатие без смещения. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

### move

Изменение положения. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- position — координаты относительно документа. Массива вида [pageX, pageY];
- delta — разница между положениями текущего и предыдущего события драггера.

### start

Начало работы драггера. Это событие происходит не в момент нажатия, а при первом измении положения после нажатия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- position — координаты относительно документа. Массива вида [pageX, pageY].

### stop

Завершение работы драггера. Это событие не может произойти без события util.Dragger.start. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- position — координаты относительно документа. Массива вида [pageX, pageY];
- delta — разница между положениями текущего и предыдущего события драггера.

## Описание методов

### destroy

```
destroy()
```

Останавливает работу драггера и удаляет прослушивание события "mousedown" у элемента autoStartElement.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### isDragging

```
{Boolean} isDragging()
```

**Возвращает** работает ли сейчас драггер.

### start

```
start(event)
```

Запускает драггер. Этот метод автоматически вызывается по событию "mousedown" элемента autoStartElement, если он задан. Данный метод можно использовать для инициализации по требованию. К примеру, при получении ответа от сервера. Для корректной работы при инициализации по требованию необходимо чтобы передаваемое событие поддерживало поле 'position' в методе get.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| event* | — | Тип: **IDomEvent**<br><br>Инициализирующее событие. |

* Обязательный параметр/опция.

### stop

```
stop()
```

Останавливает драггер. Этот метод можно использовать для досрочной остановки работы драггера.

## util.extend

Статическая функция.

Функция, копирующая свойства из одного или нескольких JavaScript-объектов в другой JavaScript-объект.

```
util.extend(target, source)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| target* | — | Тип: Object<br><br>Целевой JavaScript-объект. Будет модифицирован в результате работы функции. |
| source* | — | Тип: Object<br><br>JavaScript-объект — источник. Все его свойства будут скопированы. Источников может быть несколько (функция может иметь произвольное число параметров), данные копируются справа налево (последний аргумент имеет наивысший приоритет при копировании). |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Пример:**

```
var options = ymaps.util.extend({
     prop1: 'a',
     prop2: 'b'
}, {
     prop2: 'c',
     prop3: 'd'
}, {
     prop3: 'e'
});
// Получим в итоге: {
//     prop1: 'a',
//     prop2: 'c',
//     prop3: 'e'
// }
```

# util.hd

Статический объект.

Позволяет работать с HD экранами различных устройств.

Методы

```
util.hd()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getPixelRatio**() | Number | Возвращает отношение виртуальных пикселей на экране к физическим. |
| **selectRatio**(hash) | Number | Возвращает результат выборки — пиксельное соотношение, либо 1, если передана, например, строка. |
| **selectValue**(hash) | Object | Возвращает из переданного хэша объект равный или наиболее близкий к ключу соотношения пикселей. В случае, если передан не хэш, а, например, строка или функция, вернется она же. |

## Описание методов

### getPixelRatio

```
{Number} getPixelRatio()
```

**Возвращает** отношение виртуальных пикселей на экране к физическим.

### selectRatio

```
{Number} selectRatio(hash)
```

**Возвращает** результат выборки — пиксельное соотношение, либо 1, если передана, например, строка.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| hash* | — | Тип: **IRatioMap** |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Объект типа **IRatioMap** для экранов с различным пиксельным соотношением. |

* Обязательный параметр/опция.

### selectValue

```
{Object} selectValue(hash)
```

Возвращает из переданного хэша объект равный или наиболее близкий к ключу соотношения пикселей. В случае, если передан не хэш, а, например, строка или функция, вернется она же.

**Возвращает** результат выборки, либо исходные данные, полученные на входе.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| hash* | — | Тип: Object \| **IRatioMap**<br><br>Объект типа **IRatioMap** для экранов с различным пиксельным соотношением. |

* Обязательный параметр/опция.

**Пример:**

```
// В балун нужно добавить картинку, чтобы она хорошо смотрелась на HD экранах.
var balloonImages = {
    // Обычная картинка.
    "1": "100x100.png",
    // Картинка для ретины.
    "2": "200x200.png"
},
 var img = ymaps.util.hd.selectValue(balloonImages);
// Добавим точку на карту.
myMap.geoObjects.add(new ymaps.GeoObject({
    geometry: "Point",
    coordinates: [55, 37],
    properties: {
        balloonContent:
            '<img src="' + img + '" width="100" alt="" >'
}, {});
```

# util.math

## util.math.areEqual

Статическая функция.

Производит сравнение двух точек с допустимой погрешностью.

**Возвращает:** Результат сравнения. (тип Boolean).

```
util.math.areEqual(first, second[, diff])
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| first* | — | Тип: Number[]<br><br>Массив для сравнения. |
| second* | — | Тип: Number[]<br><br>Массив, который будем сравнивать. |
| diff | — | Тип: Number<br><br>Точность сравнения. |

* Обязательный параметр/опция.

## util.math.cycleRestrict

Статическая функция.

Приводит цифровое значение к заданному диапазону. Считается, что диапазон значений замкнут в кольцо. Если значение выходит за одну из границ диапазона, то излишек отсчитывается по кругу, со стороны другой границы.

**Возвращает:** Ограниченное значение. (тип Number).

```
util.math.cycleRestrict(value, min, max)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| value* | — | Тип: Number<br><br>Ограничиваемое значение. |
| min* | — | Тип: Number<br><br>Минимальный предел. |
| max* | — | Тип: Number<br><br>Максимальный предел. |

* Обязательный параметр/опция.

**Пример:**

```
// Возвращает 110
ymaps.util.math.cycleRestrict(-250, -180, 180);
// Возвращает 60
ymaps.util.math.cycleRestrict(-300, -180, 180);
// Возвращает -170
ymaps.util.math.cycleRestrict(190, -180, 180);
```

## util.math.restrict

Статическая функция.

Ограничивает цифровое значение на входе заданными минимальным и максимальным пределами.

**Возвращает:** Ограниченное значение. (тип Number).


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
util.math.restrict(value, min, max)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| value* | — | Тип: Number<br><br>Ограничиваемое значение. |
| min* | — | Тип: Number<br><br>Минимальный предел. |
| max* | — | Тип: Number<br><br>Максимальный предел. |

* Обязательный параметр/опция.

**Пример:**

```
ymaps.util.math.restrict(-250, -180, 180) = -180
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# util.pixelBounds

Статический объект.

Набор статических методов для работы с прямоугольными областями, представленными в виде пары из верхней левой и нижней правой точек в декардовой координатной системе.

Методы

```
util.pixelBounds()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| areIntersecting(bounds1, bounds2) | Boolean | Определяет пересекаются ли две прямоугольные области. |
| containsBounds(outer, inner) | Boolean | Определяет содержат ли прямоугольная область другую прямоугольную область полностью. |
| containsPoint(bounds, point) | Boolean | Определяет содержат ли прямоугольная область точку. |
| fromBounds(sourceBounds) | Number[][] | Вычисляет прямоугольную область, в которую попадают все переданные. |
| fromPoints(points) | Number[][] | Вычисляет прямоугольную область, в которую попадают переданные точки. |
| getCenter(bounds) | Number[] | Вычисляет центр прямоугольникой области. |
| getIntersection(bounds1, bounds2) | Number[][] \| Null | Вычисляет пересечение двух прямоугольных областей. |
| getSize(bounds) | Number[] | Вычисляет размеры прямоугольной области. |

## Описание методов

### areIntersecting

```
{Boolean} areIntersecting(bounds1, bounds2)
```

Определяет пересекаются ли две прямоугольные области.

**Возвращает** признак пересечения.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds1* | — | Тип: Number[][]<br><br>Первая область. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds2* | — | Тип: Number[][]<br><br>Вторая область. |

* Обязательный параметр/опция.

## containsBounds

```
{Boolean} containsBounds(outer, inner)
```

Определяет содержат ли прямоугольная область другую прямоугольную область полностью.

**Возвращает** признак попадания.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| outer* | — | Тип: Number[][]<br><br>Внешняя область |
| inner* | — | Тип: Number[][]<br><br>Проверяемая область. |

* Обязательный параметр/опция.

## containsPoint

```
{Boolean} containsPoint(bounds, point)
```

Определяет содержат ли прямоугольная область точку.

**Возвращает** признак попадания.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds* | — | Тип: Number[][]<br><br>Внешняя область |
| point* | — | Тип: Number[]<br><br>Проверяемая точка. |

* Обязательный параметр/опция.

## fromBounds

```
{Number[][]} fromBounds(sourceBounds)
```

Вычисляет прямоугольную область, в которую попадают все переданные.

**Возвращает** вычисленная область.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| sourceBounds* | — | Тип: Number[][][]<br><br>Массив прямоугольных областей |

* Обязательный параметр/опция.

## fromPoints

```
{Number[][]} fromPoints(points)
```

Вычисляет прямоугольную область, в которую попадают переданные точки.

**Возвращает** вычисленная область.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| points* | — | Тип: Number[][]<br><br>Массив точек. |

* Обязательный параметр/опция.

## getCenter

```
{Number[]} getCenter(bounds)
```

Вычисляет центр прямоугольникой области.

**Возвращает** точку-центр в координатной системе входящих данных.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds* | — | Тип: Number[][]<br><br>Область. |

* Обязательный параметр/опция.

## getIntersection

```
{Number[][] | Null} getIntersection(bounds1, bounds2)
```

Вычисляет пересечение двух прямоугольных областей.

**Возвращает** прямоугольную область, которая образуется пересечением переданных, либо Null, если области не пересекаются.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds1* | — | Тип: Number[][]<br><br>Первая область. |
| bounds2* | — | Тип: Number[][] |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Вторая область. |

* Обязательный параметр/опция.

### getSize

```
{Number[]} getSize(bounds)
```

Вычисляет размеры прямоугольной области.

**Возвращает** размер области.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| bounds* | — | Тип: Number[][]<br><br>Область. |

* Обязательный параметр/опция.

## util.requireCenterAndZoom

Статическая функция.

Вычисляет оптимальный центр и уровень масштабирования карты для отображения переданной области на указанном типе карты. Уровень масштаба будет в рамках доступного диапозона масштабов.

**Возвращает:** Объект-promise, который будет подтвержден объектом с полями center и zoom или будет отклонен с ошибкой. (тип **vow.Promise**).

```
util.requireCenterAndZoom(mapType, bounds, containerSize[, params])
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| mapType* | — | Тип: String \| **MapType** \| **map.ZoomRange**<br><br>Тип карты. Строка-ключ из хранилища **mapType.storage**, либо экземпляр класса **MapType**. Или менеджер коэффициентов масштабирования карты конкретной карты. |
| bounds* | — | Тип: Number[][]<br><br>Область, заданная в географических координатах. Первая точка содержит минимальные значения широты и долготы, вторая точка — максимальные. |
| containerSize* | — | Тип: Number[]<br><br>Размер контейнера карты. |
| params | — | Тип: Object |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Параметры. |
| params.inscribe | true | Тип: Boolean<br><br>true — вписывать область в карту, false — вписывать карту в область. |
| params.margin | 0 | Тип: Number \| Number[]<br><br>Отступы от границ видимой области карты. Если задано одно число — оно применяется ко всем сторонам. Если задано два — то это горизонтальные и вертикальные отступы соответственно. Если задан массив из 4х чисел, то это отступы top, right, bottom, left. |
| params.preciseZoom | false | Тип: Boolean<br><br>При значении false уровень масштабирования будет округлен в меньшую сторону. |

* Обязательный параметр/опция.

**Пример:**

```
// Находим оптимальный центр и уровень масштабирования карты.
ymaps.util.requireCenterAndZoom(
    myMap.getType(),
    [[50.531219,31.278264], [50.966841,31.964909]],
    myMap.container.getSize()
).then(function (result) {
    // Устанавливаем карте оптимальный центр и зум
    myMap.setCenter(result.center, result.zoom);
});
```

# util.Storage

Хранилище объектов по ключу.

Конструктор | Методы

## Конструктор

```
util.Storage()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(key, object) | util.Storage | Добавляет объект в хранилище. |
| **get**(key) | Object | Возвращает объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка. |
| **remove**(key) | util.Storage | Удаляет пару "ключ: значение" из хранилища. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### add

```
{util.Storage} add(key, object)
```

Добавляет объект в хранилище.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Ключ. |
| object* | — | Тип: Object<br><br>Хранимый объект. |

* Обязательный параметр/опция.

### get

```
{Object} get(key)
```

**Возвращает** объект, хранящийся под указанным ключом или исходный key, в случае, если это не строка.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String \| Object<br><br>Ключ. |

* Обязательный параметр/опция.

### remove

```
{util.Storage} remove(key)
```

Удаляет пару "ключ: значение" из хранилища.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Ключ. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# vow

Статический объект.

Содержит методы для создания и обработки объектов-promise.

---

**Примечание:**
Данный класс является частью библиотеки Vow.

Ниже описаны только некоторые методы. Полный список методов доступен здесь: http://dfilatov.github.io/vow/.

Copyright (c) 2012-2013 Filatov Dmitry (dfilatov@yandex-team.ru). Dual licensed under the MIT and GPL licenses.

---

Методы

```
vow()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **all**(iterable) | **vow.Promise** | Возвращает объект-promise, который будет разрешен или отклонён только когда будут разрешены или отклонены все указанные объекты. |
| **defer**() | **vow.Deferred** | Создаёт новый deferred-объект. Эквивалент для `new ymaps.vow.Deferred()`. |
| **reject**(reason) | **vow.Promise** | Возвращает объект-promise, отклонённый с указанной причиной. |
| **resolve**(value) | **vow.Promise** | Возвращает объект-promise, разрешённый указанным значением. |

## Описание методов

### all

```
{vow.Promise} all(iterable)
```

**Возвращает** объект-promise, который будет разрешен или отклонён только когда будут разрешены или отклонены все указанные объекты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| iterable* | — | Тип: Object \| Object[]<br><br>Набор объектов-promise и/или значений. |

* Обязательный параметр/опция.

**Примеры:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

1.

```
var deferred1 = ymaps.vow.defer(),
    deferred2 = ymaps.vow.defer();

ymaps.vow.all([deferred1.promise(), deferred2.promise(), 3])
    .then(function(value) {
         // value => [1, 2, 3]
    });

deferred1.resolve(1);
deferred2.resolve(2);
```

2.

```
var deferred1 = ymaps.vow.defer(),
    deferred2 = ymaps.vow.defer();

ymaps.vow.all({ p1 : deferred1.promise(), p2 : deferred2.promise(), p3 : 3 })
    .then(function(value) {
         // value => { p1 : 1, p2 : 2, p3 : 3 }
    });

deferred1.resolve(1);
deferred2.resolve(2);
```

## defer

```
{vow.Deferred} defer()
```

Создаёт новый deferred-объект. Эквивалент для `new ymaps.vow.Deferred()`.

**См.:** vow.Deferred

**Возвращает** объект-deferred.

## reject

```
{vow.Promise} reject(reason)
```

**Возвращает** объект-promise, отклонённый с указанной причиной.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| reason* | — | Тип: Object<br><br>Причина отклонения. |

* Обязательный параметр/опция.

## resolve

```
{vow.Promise} resolve(value)
```

**Возвращает** объект-promise, разрешённый указанным значением.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| value* | — | Тип: Object<br><br>Значение. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

# vow.Deferred

Класс, описывающий объекты-deferred.

---

**Примечание:**
Данный класс является частью библиотеки Vow.

Ниже описаны только некоторые методы. Полный список методов доступен здесь: http://dfilatov.github.io/vow/.

Copyright (c) 2012-2013 Filatov Dmitry (dfilatov@yandex-team.ru). Dual licensed under the MIT and GPL licenses.

---

**Примечание:**
Не является отдельным модулем: доступен после подключения модуля **vow**.

---

Конструктор | Методы

## Конструктор

```
vow.Deferred()
```

Создаёт объект-deferred.

**Пример:**

```
function someAsyncMethod () {
    var deferred = new ymaps.vow.Deferred(); // или `var deferred =
ymaps.vow.defer();`

    doSomeAsyncStuff(function (err, value) {
        if (err) {
            deferred.reject(err);
            return;
        }

        deferred.resolve(value);
    });

    return deferred.promise();
}

someAsyncMethod().then(function (value) {
    console.log('Результат работы метода: ' + value);
}, function (err) {
    console.log('Ошибка: ' + err);
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| promise() | vow.Promise | Возвращает связанный объект-promise. |
| reject(reason) | | Отклоняет связанный объект-promise с указанной причиной. |
| resolve(value) | | Разрешает связанный объект-promise указанным значением. |

## Описание методов

### promise

```
{vow.Promise} promise()
```

**Возвращает** связанный объект-promise.

### reject

```
reject(reason)
```

Отклоняет связанный объект-promise с указанной причиной.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| reason* | — | Тип: Object<br><br>Причина отклонения. |

* Обязательный параметр/опция.

### resolve

```
resolve(value)
```

Разрешает связанный объект-promise указанным значением.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| value* | — | Тип: Object<br><br>Значение. |

* Обязательный параметр/опция.

## vow.Promise

Класс, описывающий объекты-promise.

Спецификация Promise/A+.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Примечание:**
Данный класс является частью библиотеки Vow.

Ниже описаны только некоторые методы. Полный список методов доступен здесь: http://dfilatov.github.io/vow/.

Copyright (c) 2012-2013 Filatov Dmitry (dfilatov@yandex-team.ru). Dual licensed under the MIT and GPL licenses.

**Примечание:**
Не является отдельным модулем: доступен после подключения модуля **vow**.

Конструктор | Методы

## Конструктор

```
vow.Promise([resolver])
```

Создаёт объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| resolver | — | Тип: Function<br><br>Функция, которая принимает в качестве параметров методы resolve и reject для установки состояния и значения создаваемому объекту-promise. |

**Пример:**

```
function someAsyncMethod () {
    return new ymaps.vow.Promise(function (resolve, reject) {
        doSomeAsyncStuff(function (err, value) {
            if (err) {
                reject(err);
                return;
            }

            resolve(value);
        });
    });
}

someAsyncMethod().then(function (value) {
    console.log('Результат работы метода: ' + value);
}, function (err) {
    console.log('Ошибка: ' + err);
});
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **done**([onFulfilled[, onRejected[, onProgress[, ctx]]]]) | | Аналог метода **vow.Promise.then**, завершающий цепочку промисов. Кидает исключение в случае отклонения объекта-promise. |
| **spread**([onFulfilled[, onRejected[, ctx]]]) | **vow.Promise** | Аналог метода **vow.Promise.then**, вызывающий функции обратного вызова с набором аргументов, соответствующим массиву, которым будет разрешён/отклонён объект-promise. Обычно используется в сочетании с методами типа **vow.all**. |
| **then**([onFulfilled[, onRejected[, onProgress[, ctx]]]]) | **vow.Promise** | Задаёт функцию-обработчик для объекта-promise. |
| **valueOf**() | Object | Возвращает значение для разрешённого объекта-promise или причину отклонения для отклонённого. |

## Описание методов

### done

```
done([onFulfilled[, onRejected[, onProgress[, ctx]]]])
```

Аналог метода **vow.Promise.then**, завершающий цепочку промисов. Кидает исключение в случае отклонения объекта-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| onFulfilled | — | Тип: Function<br><br>Функция обратного вызова, которая будет вызвана в случае если объект-promise будет разрешён. |
| onRejected | — | Тип: Function<br><br>Функция обратного вызова, которая будет вызвана в случае если объект-promise будет отклонён. |
| onProgress | — | Тип: Function<br><br>Функция обратного вызова, которая будет вызвана при "оповещении" объекта-promise. |
| ctx | — | Тип: Object<br><br>Контекст выполнения функций обратного вызова. |

**Пример:**

```
var deferred = ymaps.vow.defer();
deferred.reject(Error('Internal error'));
deferred.promise().done(); // Будет брошено исключение.
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## spread

```
{vow.Promise} spread([onFulfilled[, onRejected[, ctx]]])
```

Аналог метода **vow.Promise.then**, вызывающий функции обратного вызова с набором аргументов, соответствующим массиву, которым будет разрешён/отклонён объект-promise. Обычно используется в сочетании с методами типа **vow.all**.

**Возвращает** новый объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| onFulfilled | — | Тип: Function<br><br>Функция обратного вызова, которая будет вызвана в случае если объект-promise будет разрешён. |
| onRejected | — | Тип: Function<br><br>Функция обратного вызова, которая будет вызвана в случае если объект-promise будет отклонён. |
| ctx | — | Тип: Object<br><br>Контекст выполнения функций обратного вызова. |

**Пример:**

```
var deferred1 = ymaps.vow.defer(),
    defereder2 = ymaps.vow.defer();

ymaps.vow.all([deferred1.promise(), deferred2.promise()]).spread(function(arg1,
arg2) {
    // arg1 => 1, arg2 => 'two'
});

deferred1.resolve(1);
deferred2.resolve('two');
```

## then

```
{vow.Promise} then([onFulfilled[, onRejected[, onProgress[, ctx]]]])
```

Задаёт функцию-обработчик для объекта-promise.

**Возвращает** новый объект-promise, см. спецификацию.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| onFulfilled | — | Тип: Function<br><br>Функция обратного вызова, которая будет вызвана в случае если объект-promise будет разрешён. |
| onRejected | — | Тип: Function |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Функция обратного вызова, которая будет вызвана в случае если объект-promise будет отклонён. |
| onProgress | — | Тип: Function<br><br>Функция обратного вызова, которая будет вызвана при "оповещении" объекта-promise. |
| ctx | — | Тип: Object<br><br>Контекст выполнения функций обратного вызова. |

## valueOf

```
{Object} valueOf()
```

**Возвращает** значение для разрешённого объекта-promise или причину отклонения для отклонённого.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# Интерфейсы

## IBalloon

Расширяет **IPopup**.

Интерфейс балуна.

Конструктор | Поля | События | Методы

### Конструктор

```
IBalloon()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

### События

| Имя | Описание |
|---|---|
| **close** | Закрытие инфо-объекта.<br>Унаследовано от **IPopup**. |
| **open** | Открытие инфо-объекта.<br>Унаследовано от **IPopup**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **autoPan**() | **vow.Promise** | Перемещает карту таким образом, чтобы балун оказался видимым. |
| **close**([force]) | **vow.Promise** | Закрывает инфо-объект.<br>Унаследован от **IPopup**. |
| **getData**() | | Возвращает данные инфо-объекта.<br>Унаследован от **IPopup**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от **IPopup**. |
| **getOverlaySync**() | **IOverlay** | Возвращает оверлей, если тот существует.<br>Унаследован от **IPopup**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getPosition() | | Возвращает координаты инфо-объекта.<br>Унаследован от IPopup. |
| isOpen() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от IPopup. |
| open([position[, data]]) | vow.Promise | Открывает инфо-объект в указанной позиции. Если инфо-объект уже открыт, перемещает его в указанную точку. Формат и суть координат определяется проекцией IProjection, которая содержится в опциях.<br>Унаследован от IPopup. |
| setData(data) | vow.Promise | Задаёт инфо-объекту новые данные.<br>Унаследован от IPopup. |
| setPosition(position) | vow.Promise | Задаёт инфо-объекту новую позицию.<br>Унаследован от IPopup. |

## Описание методов

### autoPan

```
{vow.Promise} autoPan()
```

Перемещает карту таким образом, чтобы балун оказался видимым.

**Возвращает** объект-promise.

# IBalloonLayout

Расширяет ILayout.

Интерфейс макета балуна.

Конструктор | Поля | События | Методы

## Конструктор

```
IBalloonLayout()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты макета. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchmove | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| parentelementchange | Изменение родительского элемента. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| shapechange | Изменение фигуры области охватывающей макет. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| userclose | Нажата кнопка закрытия. Пользователь закрывает балун. Реализует интерфейс IEvent. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Деструктор. Вызывается при завершении работы с макетом.<br>Унаследован от ILayout. |
| getData() | Object | Возвращает объект данных макета.<br>Унаследован от ILayout. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getParentElement**() | HTMLElement | Возвращает родительский html-элемент.<br>Унаследован от **ILayout**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.<br>Унаследован от **ILayout**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **ILayout**. |
| **setData**(data) | | Задает данные макета.<br>Унаследован от **ILayout**. |
| **setParentElement**(parent) | | Добавляет макет в DOM-дерево.<br>Унаследован от **ILayout**. |

### Описание событий

#### userclose

Нажата кнопка закрытия. Пользователь закрывает балун. Реализует интерфейс **IEvent**.

## IBalloonManager

Расширяет **IPopupManager**.

Интерфейс менеджера балуна.

Конструктор | Поля | События | Методы

### Конструктор

```
IBalloonManager()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| autopanbegin | Начало автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• target — ссылка на объект IBalloonOwner. |
| autopanend | Окончание автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• target — ссылка на объект IBalloonOwner. |
| beforeuserclose | Событие, предшествующее Balloon.userclose. Позволяет отменить пользовательское действие посредством вызова метода preventDefault. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• target — ссылка на объект IBalloonOwner. |
| userclose | Закрытие балуна пользователем. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• target — ссылка на объект IBalloonOwner. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| autoPan() | vow.Promise | Перемещает карту таким образом, чтобы балун оказался видимым. |
| close([force]) | vow.Promise | Закрывает инфо-объект.<br>Унаследован от IPopupManager. |
| destroy() | | Отключает менеджер инфо-объекта.<br>Унаследован от IPopupManager. |
| getData() | Object \| null | Возвращает данные инфо-объекта или `null`.<br>Унаследован от IPopupManager. |
| getOptions() | IOptionManager \| null | Возвращает менеджер опций или `null`.<br>Унаследован от IPopupManager. |
| getOverlay() | vow.Promise | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от IPopupManager. |
| getOverlaySync() | IOverlay \| null | Возвращает оверлей, если тот существует.<br>Унаследован от IPopupManager. |
| getPosition() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от IPopupManager. |
| isOpen() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IPopupManager**. |
| **open**([position[, data[, options]]]) | **vow.Promise** | Открывает инфо-объект в указанной позиции.<br>Унаследован от **IPopupManager**. |
| **setData**(data) | **vow.Promise** | Задаёт инфо-объекту новые данные.<br>Унаследован от **IPopupManager**. |
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopupManager**. |

## Описание событий

### autopanbegin

Начало автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- target — ссылка на объект **IBalloonOwner**.

### autopanend

Окончание автоматического перемещения центра карты, вызванного методом autoPan. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- target — ссылка на объект **IBalloonOwner**.

### beforeuserclose

Событие, предшествующее Balloon.userclose. Позволяет отменить пользовательское действие посредством вызова метода preventDefault. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- target — ссылка на объект **IBalloonOwner**.

### userclose

Закрытие балуна пользователем. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- target — ссылка на объект **IBalloonOwner**.

## Описание методов

### autoPan

```
{vow.Promise} autoPan()
```

Перемещает карту таким образом, чтобы балун оказался видимым.

**Возвращает** объект-promise.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IBalloonOwner

Объект, имеющий балун, доступный через свойство "balloon".

Конструктор | Поля | События

## Конструктор

```
IBalloonOwner()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| balloon | IBalloonManager | Балун объекта. |

## События

| Имя | Описание |
|---|---|
| balloonclose | Закрытие балуна. |
| balloonopen | Открытие балуна. |

## Описание полей

### balloon

```
{IBalloonManager} balloon
```

Балун объекта.

## Описание событий

### balloonclose

Закрытие балуна.

### balloonopen

Открытие балуна.

# IBaseCircleGeometry

Расширяет **IBaseGeometry**, **ICircleGeometryAccess**.

Интерфейс геометрии "Круг".

Конструктор | Поля | События | Методы

## Конструктор

```
IBaseCircleGeometry()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |

## События

| Имя | Описание |
|---|---|
| change | Изменение координат. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты центра;<br>• newCoordinates — новые координаты центра;<br>• oldRadius — старый радиус;<br>• newRadius — новый радиус.<br>Унаследовано от ICircleGeometryAccess. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| contains(position) | Boolean | Проверяет, лежит ли переданная точка внутри круга.<br>Унаследован от ICircleGeometryAccess. |
| freeze() | IFreezable | Переводит объект в "замороженный" режим.<br>Унаследован от IFreezable. |
| getBounds() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от IBaseGeometry. |
| getClosest(anchorPosition) | Object | Ищет на окружности точку, ближайшую к anchorPosition.<br>Унаследован от ICircleGeometryAccess. |
| getCoordinates() | Number[] \| null | Возвращает координаты центра круга.<br>Унаследован от ICircleGeometryAccess. |
| getRadius() | Number | Возвращает радиус круга.<br>Унаследован от ICircleGeometryAccess. |
| getType() | String | Возвращает строку "Circle". |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| isFrozen() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от IFreezable. |
| setCoordinates(coordinates) | ICircleGeometryAccess | Задает координаты центра круга.<br>Унаследован от ICircleGeometryAccess. |
| setRadius(radius) | ICircleGeometryAccess | Задает радиус круга.<br>Унаследован от ICircleGeometryAccess. |
| unfreeze() | IFreezable | Переводит объект в активный режим.<br>Унаследован от IFreezable. |

### Описание методов

#### getType

```
{String} getType()
```

**Возвращает** строку "Circle".

## IBaseGeometry

Расширяет **IEventEmitter**.

Интерфейс базовой геометрии.

Конструктор | Поля | Методы

### Конструктор

```
IBaseGeometry()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getBounds() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат. |
| getType() | String | Возвращает идентификатор типа геометрии. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Описание методов

#### getBounds

```
{Number[][] | null} getBounds()
```

**Возвращает** координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.

#### getType

```
{String} getType()
```

**Возвращает** идентификатор типа геометрии.

## IBaseLinearRingGeometry

Расширяет **IBaseGeometry**, **ILinearRingGeometryAccess**.

Интерфейс геометрии "Замкнутый контур".

Конструктор | Поля | События | Методы

### Конструктор

```
IBaseLinearRingGeometry()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

### События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты;<br>• oldFillRule — старый алгоритм заливки;<br>• newFillRule — новые алгоритм заливки.<br>Унаследовано от **ILinearRingGeometryAccess**. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри контура.<br>Унаследован от **ILinearRingGeometryAccess**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **get**(index) | Number[] | Возвращает координаты точки с заданным индексом.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getChildGeometry**(index) | **IPointGeometryAccess** | Создает и возвращает объект **IPointGeometryAccess** для заданной вершины контура.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре точку, ближайшую к anchorPosition.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getCoordinates**() | Number[][] | Возвращает массив координат геометрии.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getFillRule**() | String | Возвращает идентификатор алгоритма заливки.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getLength**() | Integer | Возвращает количество точек в геометрии.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **getType**() | String | Возвращает строку "LinearRing". |
| **insert**(index, coordinates) | **ILinearRingGeometryAccess** | Добавляет новую точку с заданным индексом.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | Number[] | Удаляет точку с заданным индексом.<br>Унаследован от **ILinearRingGeometryAccess**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **set**(index, coordinates) | **ILinearRingGeometryAccess** | Задает координаты точки с заданным индексом.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **setCoordinates**(coordinates) | **ILinearRingGeometryAccess** | Задает массив координат геометрии.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **setFillRule**(fillRule) | **ILinearRingGeometryAccess** | Задает алгоритм заливки контура.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **splice**(index, number) | Number[][] | Удаляет определенное число точек начиная с заданного индекса. При этом на место удаляемых точек могут быть добавлены новые. Координаты новых точек можно передавать дополнительными аргументами после параметра number.<br>Унаследован от **ILinearRingGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание методов

### getType

```
{String} getType()
```

**Возвращает** строку "LinearRing".

# IBaseLineStringGeometry

Расширяет **IBaseGeometry**, **ILineStringGeometryAccess**.

Интерфейс геометрии "Ломаная линия".

Конструктор | Поля | События | Методы

## Конструктор

```
IBaseLineStringGeometry()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| change | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты.<br>Унаследовано от **ILineStringGeometryAccess**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **get**(index) | Number[] | Возвращает координаты точки с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getChildGeometry**(index) | **IPointGeometryAccess** | Создает и возвращает объект **IPointGeometryAccess** для заданной вершины ломаной линии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getClosest**(anchorPosition) | Object | Ищет на ломаной линии точку, ближайшую к anchorPosition.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getCoordinates**() | Number[][] | Возвращает массив координат геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getLength**() | Integer | Возвращает количество точек в геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getType**() | String | Возвращает строку "LineString". |
| **insert**(index, coordinates) | **ILineStringGeometryAccess** | Добавляет новую точку с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | Number[] | Удаляет точку с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **set**(index, coordinates) | **ILineStringGeometryAccess** | Задает координаты точки с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **setCoordinates**(coordinates) | **ILineStringGeometryAccess** | Задает массив координат геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **splice**(index, number) | Number[][] | Удаляет определенное число точек начиная с заданного индекса. При этом на место удаляемых точек могут быть добавлены новые. Координаты новых точек можно передавать дополнительными аргументами после параметра number.<br>Унаследован от **ILineStringGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

### Описание методов

#### getType

```
{String} getType()
```

**Возвращает** строку "LineString".

## IBasePointGeometry

Расширяет **IBaseGeometry**, **IPointGeometryAccess**.

Интерфейс геометрии "Точка".

Конструктор | Поля | События | Методы

### Конструктор

```
IBasePointGeometry()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### События

| Имя | Описание |
|---|---|
| change | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты.<br>Унаследовано от **IPointGeometryAccess**. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getCoordinates**() | Number[] \| null | Возвращает координаты точки.<br>Унаследован от **IPointGeometryAccess**. |
| **getType**() | String | Возвращает строку "Point". |
| **setCoordinates**(coordinates) | **IPointGeometryAccess** | Задает координаты точки.<br>Унаследован от **IPointGeometryAccess**. |

## Описание методов

### getType

```
{String} getType()
```

**Возвращает** строку "Point".

# IBasePolygonGeometry

Расширяет **IBaseGeometry**, **IPolygonGeometryAccess**.

Интерфейс геометрии "Многоугольник".

Конструктор | Поля | События | Методы

### Конструктор

```
IBasePolygonGeometry()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |

## События

| Имя | Описание |
|---|---|
| change | Изменение координат. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты;<br>• oldFillRule — старый алгоритм заливки;<br>• newFillRule — новые алгоритм заливки.<br>Унаследовано от IPolygonGeometryAccess. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| contains(position) | Boolean | Проверяет, лежит ли переданная точка внутри многоугольника.<br>Унаследован от IPolygonGeometryAccess. |
| freeze() | IFreezable | Переводит объект в "замороженный" режим.<br>Унаследован от IFreezable. |
| get(index) | Number[][] | Возвращает координаты контура с заданным индексом.<br>Унаследован от IPolygonGeometryAccess. |
| getBounds() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от IBaseGeometry. |
| getChildGeometry(index) | ILinearRingGeometryAccess | Создает и возвращает объект ILinearRingGeometryAccess для заданного контура.<br>Унаследован от IPolygonGeometryAccess. |
| getClosest(anchorPosition) | Object | Ищет на контуре многоугольника точку, ближайшую к anchorPosition.<br>Унаследован от IPolygonGeometryAccess. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getCoordinates**() | Number[][][] | Возвращает массив координат геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getFillRule**() | String | Возвращает идентификатор алгоритма заливки.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getLength**() | Integer | Возвращает количество контуров в геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getType**() | String | Возвращает строку "Polygon". |
| **insert**(index, path) | **IPolygonGeometryAccess** | Добавляет новый контур с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | **ILinearRingGeometryAccess** | Удаляет контур с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **set**(index, path) | **IPolygonGeometryAccess** | Задает координаты контура с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **setCoordinates**(coordinates) | **IPolygonGeometryAccess** | Задает массив координат геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **setFillRule**(fillRule) | **IPolygonGeometryAccess** | Задает алгоритм заливки многоугольника.<br>Унаследован от **IPolygonGeometryAccess**. |
| **splice**(index, number) | **ILinearRingGeometryAccess**[] | Удаляет определенное число контуров начиная с заданного индекса. При этом на место удаляемых контуров могут быть добавлены новые. Координаты новых контуров можно передавать дополнительными аргументами после параметра number.<br>Унаследован от **IPolygonGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### getType

```
{String} getType()
```

**Возвращает** строку "Polygon".

# IBaseRectangleGeometry

Расширяет **IBaseGeometry**, **IRectangleGeometryAccess**.

Интерфейс геометрии "Прямоугольник".

Конструктор | Поля | События | Методы

## Конструктор

```
IBaseRectangleGeometry()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат углов. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты углов;<br>• newCoordinates — новые координаты углов.<br>Унаследовано от **IRectangleGeometryAccess**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри прямоугольника.<br>Унаследован от **IRectangleGeometryAccess**. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре прямоугольника точку, ближайшую к anchorPosition.<br>Унаследован от **IRectangleGeometryAccess**. |
| **getCoordinates**() | Number[][] | Возвращает координаты двух противоположных углов прямоугольника.<br>Унаследован от **IRectangleGeometryAccess**. |
| **getType**() | String | Возвращает строку "Rectangle". |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **setCoordinates**(coordinates) | **IRectangleGeometryAccess** | Задает координаты двух противоположных углов прямоугольника.<br>Унаследован от **IRectangleGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание методов

### getType

```
{String} getType()
```

**Возвращает** строку "Rectangle".

# IBehavior

Расширяет **IChildOnMap**, **ICustomizable**.

Поведение карты. Добавляет к карте реакцию на определённые пользовательские действия (например, таскание, масштабирование правой кнопкой мыши, многосенсорное касание). Поведение по умолчанию выключено и включается вызовом метода enable.

Конструктор | Поля | События | Методы

## Конструктор

```
IBehavior([options])
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции поведения. |

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| disable | Выключение поведения. |
| enable | Включение поведения. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| disable() | | Выключает поведение. |
| enable() | | Включает поведение. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| isEnabled() | Boolean | Проверяет, включено ли поведение. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Описание событий

#### disable

Выключение поведения.

#### enable

Включение поведения.

### Описание методов

#### disable

```
disable()
```

Выключает поведение.

#### enable

```
enable()
```

Включает поведение.

#### isEnabled

```
{Boolean} isEnabled()
```

Проверяет, включено ли поведение.

**Возвращает** true — поведение включено, false — выключено.

## ICanvasTile

Интерфейс тайлов, умеющих отображаться на 2d-контексте объекта canvas.

Конструктор | Поля | События | Методы

### Конструктор

```
ICanvasTile(url)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| url * | — | Тип: String<br><br>URL тайла. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий тайла. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### События

| Имя | Описание |
|---|---|
| ready | Событие готовности тайла. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Уничтожает тайл. |
| isReady() | Boolean | Проверяет готовность тайла. |
| renderAt(context, canvasSize, bounds[, animate]) | | Отрисовывает картиночный тайл на 2d-контексте объекта canvas. |

## Описание полей

### events

```
{IEventManager} events
```

Менеджер событий тайла.

## Описание событий

### ready

Событие готовности тайла.

## Описание методов

### destroy

```
destroy()
```

Уничтожает тайл.

### isReady

```
{Boolean} isReady()
```

Проверяет готовность тайла.

**Возвращает** true — тайл готов, false — тайл не готов.

### renderAt

```
renderAt(context, canvasSize, bounds[, animate])
```

Отрисовывает картиночный тайл на 2d-контексте объекта canvas.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| context* | — | Тип: Object<br><br>2d-контекст объекта canvas. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| canvasSize* | — | Тип: Number[]<br><br>размеры HTML-элемента canvas. |
| bounds* | — | Тип: Number[][]<br><br>Область в клиентских координатах, в которой нужно отрисовать тайл. |
| animate | false | Тип: Boolean<br><br>true — анимировать рендеринг, false — нет. |

* Обязательный параметр/опция.

# IChild

Расширяет **IEventEmitter**.

Интерфейс объекта, имеющего родителя.

Конструктор | Поля | События | Методы

## Конструктор

```
IChild()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **parentchange** | Сменился родительский объект.<br><br>Поля данных:<br><br>• oldParent — старый родитель;<br>• newParent — новый родитель. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getParent**() | Object | Возвращает ссылку на родительский объект. |
| **setParent**(parent) | **IChild** | Устанавливает родительский объект. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание событий

### parentchange

Сменился родительский объект.

Поля данных:

- oldParent — старый родитель;
- newParent — новый родитель.

## Описание методов

### getParent

```
{Object} getParent()
```

**Возвращает** ссылку на родительский объект.

### setParent

```
{IChild} setParent(parent)
```

Устанавливает родительский объект.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parent* | — | Тип: Object<br><br>Родительский объект. |

* Обязательный параметр/опция.

# IChildOnMap

Расширяет **IChild**.

Интерфейс объекта, родитель которого относится к определенной карте.

Конструктор | Поля | События | Методы

## Конструктор

```
IChildOnMap()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### События

| Имя | Описание |
|---|---|
| parentchange | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от IChild. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта. |

## Описание методов

### getParent

```
{IControlParent | null} getParent()
```

**Возвращает** ссылку на родительский объект или null, если родительский элемент не был установлен.

### setParent

```
{IChildOnMap} setParent(parent)
```

Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parent* | — | Тип: IControlParent \| null<br>Родительский объект. |

* Обязательный параметр/опция.

# ICircleGeometry

Расширяет ICircleGeometryAccess, IGeometry.

Интерфейс геометрии "Круг".

Конструктор | Поля | События | Методы


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Конструктор

```
ICircleGeometry()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| change | Изменение координат. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты центра;<br>• newCoordinates — новые координаты центра;<br>• oldRadius — старый радиус;<br>• newRadius — новый радиус.<br>Унаследовано от ICircleGeometryAccess. |
| mapchange | Сменилась карта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IGeometry. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| pixelgeometrychange | Изменилась пиксельная геометрия. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия IPixelGeometry.<br>Унаследовано от IGeometry. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| contains(position) | Boolean | Проверяет, лежит ли переданная точка внутри круга.<br>Унаследован от ICircleGeometryAccess. |
| freeze() | IFreezable | Переводит объект в "замороженный" режим.<br>Унаследован от IFreezable. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.<br>Унаследован от **IGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на окружности точку, ближайшую к anchorPosition.<br>Унаследован от **ICircleGeometryAccess**. |
| **getCoordinates**() | Number[] \| null | Возвращает координаты центра круга.<br>Унаследован от **ICircleGeometryAccess**. |
| **getMap**() | **Map** \| null | Возвращает текущую карту.<br>Унаследован от **IGeometry**. |
| **getPixelGeometry**([options]) | **IPixelGeometry** | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.<br>Унаследован от **IGeometry**. |
| **getRadius**() | Number | Возвращает радиус круга.<br>Унаследован от **ICircleGeometryAccess**. |
| **getType**() | String | Возвращает строку "Circle". |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **setCoordinates**(coordinates) | **ICircleGeometryAccess** | Задает координаты центра круга.<br>Унаследован от **ICircleGeometryAccess**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |
| **setRadius**(radius) | **ICircleGeometryAccess** | Задает радиус круга.<br>Унаследован от **ICircleGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание методов

### getType

```
{String} getType()
```

**Возвращает** строку "Circle".


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# ICircleGeometryAccess

Расширяет **IFreezable**.

Интерфейс доступа к геометрии "Круг".

Конструктор | Поля | События | Методы

## Конструктор

```
ICircleGeometryAccess()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий объекта.<br>Унаследовано от **IFreezable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты центра;<br>• newCoordinates — новые координаты центра;<br>• oldRadius — старый радиус;<br>• newRadius — новый радиус. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри круга. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **getClosest**(anchorPosition) | Object | Ищет на окружности точку, ближайшую к anchorPosition. |
| **getCoordinates**() | Number[] \| null | Возвращает координаты центра круга. |
| **getRadius**() | Number | Возвращает радиус круга. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **setCoordinates**(coordinates) | **ICircleGeometryAccess** | Задает координаты центра круга. |
| **setRadius**(radius) | **ICircleGeometryAccess** | Задает радиус круга. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание событий

### change

Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- oldCoordinates — старые координаты центра;
- newCoordinates — новые координаты центра;
- oldRadius — старый радиус;
- newRadius — новый радиус.

## Описание методов

### contains

```
{Boolean} contains(position)
```

Проверяет, лежит ли переданная точка внутри круга.

**Возвращает** признак принадлежности точки кругу.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position* | — | Тип: Number[]<br><br>Координаты точки. |

* Обязательный параметр/опция.

**Пример:**

```
var myCircle = new ymaps.geometry.base.Circle([0, 0], 10);
myCircle.contains([0, 10]); // => true
```

### getClosest

```
{Object} getClosest(anchorPosition)
```

Ищет на окружности точку, ближайшую к anchorPosition.

**Возвращает** объект со следующими полями:

- position — точка на окружности, ближайшая к anchorPosition;
- distance — расстояние от anchorPosition до position;

.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| anchorPosition* | — | Тип: Number[]<br><br>Координаты точки, для которой рассчитывается ближайшая точка на окружности. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
var myCircle = new ymaps.geometry.base.Circle([0, 0], 10);
myCircle.getClosest([0, 15]).position; // => [0, 10]
```

### getCoordinates

```
{Number[] | null} getCoordinates()
```

**Возвращает** координаты центра круга.

### getRadius

```
{Number} getRadius()
```

**Возвращает** радиус круга.

### setCoordinates

```
{ICircleGeometryAccess} setCoordinates(coordinates)
```

Задает координаты центра круга.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates* | — | Тип: Number[] \| null<br><br>Координаты центра круга. |

* Обязательный параметр/опция.

### setRadius

```
{ICircleGeometryAccess} setRadius(radius)
```

Задает радиус круга.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| radius* | — | Тип: Number<br><br>Радиус круга. |

* Обязательный параметр/опция.

## ICollection

Расширяет **IEventEmitter**.

Интерфейс коллекции.

Конструктор | Поля | События | Методы

### Конструктор

```
ICollection()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |

## События

| Имя | Описание |
|---|---|
| add | Произошло добавление дочернего объекта. |
| remove | Произошло удаление дочернего объекта. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| add(object) | ICollection | Добавляет в коллекцию дочерний объект. |
| getIterator() | IIterator | Возвращает итератор по коллекции. |
| remove(object) | ICollection | Удаляет из коллекции дочерний объект. |

# Описание событий

## add

Произошло добавление дочернего объекта.

## remove

Произошло удаление дочернего объекта.

# Описание методов

## add

```
{ICollection} add(object)
```

Добавляет в коллекцию дочерний объект.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| object* | — | Тип: Object<br>Добавляемый объект. |

* Обязательный параметр/опция.

## getIterator

```
{IIterator} getIterator()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** итератор по коллекции.

### remove

```
{ICollection} remove(object)
```

Удаляет из коллекции дочерний объект.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| object* | — | Тип: Object<br><br>Удаляемый объект. |

* Обязательный параметр/опция.

## IContainerPane

Расширяет **IPane**, **IPositioningContext**.

Интерфейс пейна карты, предназначенного для размещения отображений объектов, расположенных на поверхности карты. Предоставляет возможность перевода глобальных пикселей карты в собственную локальную координатную систему, и т.о. позволяет объекту позиционироваться внутри себя. Также информирует об изменении состояния карты и т.о. позволяет объекту обрабатывать эти изменения.

Конструктор | Поля | События | Методы

### Конструктор

```
IContainerPane()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

### События

| Имя | Описание |
|---|---|
| **actionbegin** | Начало движения пейна. Экземпляр **IEvent**. |
| **actionend** | Окончание движения пейна. Экземпляр **IEvent**. |
| **clientpixelschange** | Изменение локальной координатной системы пейна. Данное событие означает, что объекты, рассчитывающие свою позицию внутри пейна из глобальных пиксельных координат карты, должны пересчитать ее и обновить свое положение внутри DOM-элемента пейна. Экземпляр **IEvent**. |
| **overflowchange** | Изменение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Экземпляр **IEvent**.<br><br>Унаследовано от **IPane**. |
| **viewportchange** | Изменение области показа пейна. Экземпляр **IEvent**. |
| **zindexchange** | Изменение значения zIndex пейна. Экземпляр **IEvent**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IPane**. |
| **zoomchange** | Изменение актуального уровня масштабирования пейна. Экземпляр **IEvent**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает пейн.<br>Унаследован от **IPane**. |
| **fromClientPixels**(clientPixelPoint) | Number[] | Преобразует клиентские пиксельные координаты в глобальные.<br>Унаследован от **IPositioningContext**. |
| **getElement**() | HTMLElement | Возвращает ссылку на DOM-контейнер пейна.<br>Унаследован от **IPane**. |
| **getMap**() | **Map** | Возвращает карту, которой принадлежит пейн.<br>Унаследован от **IPane**. |
| **getOverflow**() | String | Возвращает значение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Данный параметр может принимать одно из следующих строковых значений:<br>• "visible" — при выходе за пределы контейнера карты содержимое пейна остается видимым;<br>• "hidden" — область отображения содержимого пейна ограничена контейнером карты.<br>.<br>Унаследован от **IPane**. |
| **getViewport**() | Number[][] | Возвращает область показа пейна в клиентских координатах. |
| **getZIndex**() | Number | Возвращает zIndex пейна.<br>Унаследован от **IPane**. |
| **getZoom**() | Number | Возвращает текущий уровень масштабирования, в котором работает контекст позиционирования.<br>Унаследован от **IPositioningContext**. |
| **toClientPixels**(globalPixelPoint) | Number[] | Преобразует глобальные пиксельные координаты в клиентские.<br>Унаследован от **IPositioningContext**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Описание событий

#### actionbegin

Начало движения пейна. Экземпляр **IEvent**.

#### actionend

Окончание движения пейна. Экземпляр **IEvent**.

#### clientpixelschange

Изменение локальной координатной системы пейна. Данное событие означает, что объекты, рассчитывающие свою позицию внутри пейна из глобальных пиксельных координат карты, должны пересчитать ее и обновить свое положение внутри DOM-элемента пейна. Экземпляр **IEvent**.

#### viewportchange

Изменение области показа пейна. Экземпляр **IEvent**.

#### zoomchange

Изменение актуального уровня масштабирования пейна. Экземпляр **IEvent**.

### Описание методов

#### getViewport

```
{Number[][]} getViewport()
```

**Возвращает** область показа пейна в клиентских координатах.

## IControl

Расширяет **IChildOnMap**.

Элемент управления.

Конструктор | Поля | События | Методы

### Конструктор

```
IControl([options])
```

Интерфейс элемента управления.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Опции элемента управления. |

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций. |

## События

| Имя | Описание |
|---|---|
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта. |

# Описание полей

## options

```
{IOptionManager} options
```

Менеджер опций.

# Описание методов

## getParent

```
{IControlParent | null} getParent()
```

**Возвращает** ссылку на родительский объект или null, если родительский элемент не был установлен.

## setParent

```
{IChildOnMap} setParent(parent)
```

Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parent* | — | Тип: **IControlParent** \| null |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Родительский объект. |

* Обязательный параметр/опция.

# IControlParent

Расширяет **IParentOnMap**.

Интерфейс родительского объекта для элемента управления.

Конструктор | Поля | События | Методы

## Конструктор

```
IControlParent()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **state** | **IDataManager** | Менеджер состояний. |

## События

| Имя | Описание |
|---|---|
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getChildElement**(child) | **vow.Promise** | Возвращает объект-обещание, который подтверждается HTML-элементом, в котором должен разместиться дочерний элемент. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |

## Описание полей

### state

```
{IDataManager} state
```

Менеджер состояний.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### getChildElement

```
{vow.Promise} getChildElement(child)
```

Возвращает объект-обещание, который подтвержается HTML-элементом, в котором должен разместиться дочерний элемент.

**Возвращает** объект-обещание.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **IControl**<br><br>Дочерний объект. |

* Обязательный параметр/опция.

# ICoordSystem

Интерфейс координатной системы карты. Этот интерфейс необходимо реализовать в случае использования нестандартных (например, цилиндрических) координат.

**См.: coordSystem.geo coordSystem.cartesian**

Конструктор | Методы

## Конструктор

```
ICoordSystem()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getDistance**(point1, point2) | Number | Возвращает кратчайшее (вдоль геодезической линии) расстояние между двумя заданными точками (в метрах). |
| **solveDirectProblem**(startPoint, direction, distance) | Object | Решает так называемую первую (прямую) геодезическую задачу: где мы окажемся, если выйдем из указанной точки в указанном направлении и пройдём, не сворачивая, указанное расстояние. |
| **solveInverseProblem**(startPoint, endPoint[, reverseDirection]) | Object | Решает так называемую вторую (обратную) геодезическую задачу: построить кратчайший маршрут между двумя точками на картографируемой поверхности и определелить расстояние и направление движения. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### getDistance

```
{Number} getDistance(point1, point2)
```

**Возвращает** кратчайшее (вдоль геодезической линии) расстояние между двумя заданными точками (в метрах).

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| point1* | — | Тип: Number[]<br><br>Первая точка. |
| point2* | — | Тип: Number[]<br><br>Вторая точка. |

* Обязательный параметр/опция.

**Пример:**

```
// Рассчитаем расстояние между Москвой и Нью-Йорком
// Координаты Москвы
ymaps.geocode('Москва').then(function (res) {
    var moscowCoords = res.geoObjects.get(0).geometry.getCoordinates();
    // Координаты Нью-Йорка
    ymaps.geocode('Нью-Йорк').then(function (res) {
        var newYorkCoords = res.geoObjects.get(0).geometry.getCoordinates();
        // Расстояние
        alert(ymaps.formatter.distance(
            ymaps.coordSystem.geo.getDistance(moscowCoords, newYorkCoords)
        ));
    });
});
```

### solveDirectProblem

```
{Object} solveDirectProblem(startPoint, direction, distance)
```

Решает так называемую первую (прямую) геодезическую задачу: где мы окажемся, если выйдем из указанной точки в указанном направлении и пройдём, не сворачивая, указанное расстояние. Решением прямой геодезической задачи являются следующие данные:

- конечная точка,
- конечное направление,
- функция пути,
- функция, позволяющая для произвольного момента времени указать, в какой точке мы будем находиться и в каком направлении двигаться.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** объект с полями:

- startPoint — начальная точка в геокоординатах,
- startDirection — начальное направление движения,
- endPoint — конечная точка в геокоординатах,
- endDirection — конечное направление движения,
- distance — расстояние в метрах,
- pathFunction — Функция, принимающая число от 0 до 1 (доля пройденного пути) и возвращающая структуру с полями point и direction.

.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| startPoint* | — | Тип: Number[]<br><br>Точка отправления. |
| direction* | — | Тип: Number[]<br><br>Направление. Задаётся в виде вектора (приращение координат) [dlat, dlon] либо [dlon, dlat], в завивимости от параметра coordorder. Для того, чтобы получить из заданного таким образом направления азимут (угол между направлением движения и направлением на север), необходимо вычислить арктангенс величины dlat/dlon (в JavaScript — стандартная функция Math.atan2(dlat, dlon)); для того, чтобы по известному азимуту a вычислить направление движения, нужно положить dlat = cos(a), dlon = sin(a). |
| distance* | — | Тип: Number<br><br>Пройденное расстояние в метрах. |

* Обязательный параметр/опция.

**Пример:**

```
// Допустим, мы вылетели из аэропорта Домодедово на северо-восток и
// пролетели 200 километров по прямой. Изобразим наш путь с помощью меток
// на карте.

// Найдём с помощью геокодирования координаты начальной точки.
ymaps.geocode('аэропорт Домодедово').then(function (res) {
    var startPoint = res.geoObjects.get(0).geometry.getCoordinates(),
    // Движемся на северо-восток, азимут 45 градусов
    // или pi/4 радиан.
        azimuth = Math.PI / 4,
    // Направление движения.
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
        direction = [Math.cos(azimuth), Math.sin(azimuth)],
    // Путевая функция
        path = ymaps.coordSystem.geo
            .solveDirectProblem(startPoint, direction, 2e5).pathFunction;

    // Изобразим путь на карте с помощью меток, проставленных через каждые 10 км
    for (var i = 0; i <= 20; i++) {
        map.geoObjects.add(new ymaps.Placemark(path(i/20).point));
    }
});
```

## solveInverseProblem

```
{Object} solveInverseProblem(startPoint, endPoint[, reverseDirection])
```

Решает так называемую вторую (обратную) геодезическую задачу: построить кратчайший маршрут между двумя точками на картографируемой поверхности и определелить расстояние и направление движения. Обратите внимание: на карте земной поверхности кратчайшие маршруты изображаются кривыми линиями. Для геообъектов в API можно включить режим отображения кратчайших расстояний между точками с помощью опции geodesic.

**Возвращает** объект с полями:

- startPoint — начальная точка в геокоординатах,
- startDirection — начальное направление движения,
- endPoint — конечная точка в геокоординатах,
- endDirection — конечное направление движения,
- distance — расстояние в метрах,
- pathFunction — Функция пути, принимающая число от 0 до 1 (доля пройденного пути) и возвращающая структуру с полями point и direction.

.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| startPoint* | — | Тип: Number[]<br><br>Точка отправления. |
| endPoint* | — | Тип: Number[]<br><br>Точка прибытия. |
| reverseDirection | false | Тип: Boolean<br><br>Направление движения. false — выбрать кратчайшую дугу, true — выбрать обратную кратчайшей. |

* Обязательный параметр/опция.

**Пример:**

```
// Построим кратчайший маршрут от Калининграда до Владивостока
// Найдём координаты Калининграда
ymaps.geocode('Калининград').then(function (res) {
    var startPoint = res.geoObjects.get(0).geometry.getCoordinates();
    // Найдём координаты Владивостока
    ymaps.geocode('Владивосток').then(function (res) {
        var endPoint = res.geoObjects.get(0).geometry.getCoordinates(),
        // Найдём функцию пути между двумя точками
        path = ymaps.coordSystem.geo
            .solveInverseProblem(startPoint, endPoint).pathFunction;
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
        // Изобразим путь 20-ю точками
        for (var i = 0; i <= 20; i++) {
            // Находим промежуточную точку
            var position = path(i/20).point;
            // Добавим в промежуточную точку метку
            map.geoObjects.add(new ymaps.Placemark(
                position, {
                    // Покажем в контенте метки пройденное расстояние
                    iconContent: ymaps.formatter.distance(
                        ymaps.coordSystem.geo.getDistance(startPoint, position)
                    )
                }, {
                    preset: 'islands#redStretchyIcon'
                }
            ));
        }
    });
});
```

# ICopyrightsAccessor

Расширяет **ICopyrightsProvider**.

Интерфейс объекта, предоставляющего доступ к пользовательской информации об авторских правах, добавленной на карту с помощью метода map.Copyrights.add.

**См.:** map.Copyrights.add

Конструктор | Поля | События | Методы

## Конструктор

```
ICopyrightsAccessor()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **copyrightschange** | Событие изменения информации об авторских правах.<br>Унаследовано от **ICopyrightsProvider**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getCopyrights**(coords, zoom) | **vow.Promise** | Запрашивает информацию об авторских правах в указанной точке с указанным коэффициентом масштабирования. Возвращает объект-promise, который будет подтверждён и передаст в качестве результата массив строк или DOM-элементов с информацией об авторских правах. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **ICopyrightsProvider**. |
| **remove**() | | Удаляет добавленную через данный объект информацию об авторских правах.<br>Унаследован от **ICopyrightsProvider**. |
| **setCopyrights**(copyrights) | | Задаёт новое значение для добавленной через данный объект информации об авторских правах.<br>Унаследован от **ICopyrightsProvider**. |

# ICopyrightsProvider

Расширяет **IEventEmitter**.

Провайдер информации об авторских правах.

Конструктор | Поля | События | Методы

## Конструктор

```
ICopyrightsProvider()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **copyrightschange** | Событие изменения информации об авторских правах. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getCopyrights**(coords, zoom) | **vow.Promise** | Запрашивает информацию об авторских правах в указанной точке с указанным коэффициентом масштабирования. Возвращает объект-promise, который будет подтверждён и передаст в качестве результата массив строк или DOM-элементов с информацией об авторских правах. |
| **remove**() | | Удаляет добавленную через данный объект информацию об авторских правах. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| setCopyrights(copyrights) | | Задаёт новое значение для добавленной через данный объект информации об авторских правах. |

## Описание событий

### copyrightschange

Событие изменения информации об авторских правах.

## Описание методов

### getCopyrights

```
{vow.Promise} getCopyrights(coords, zoom)
```

Запрашивает информацию об авторских правах в указанной точке с указанным коэффициентом масштабирования. Возвращает объект-promise, который будет подтверждён и передаст в качестве результата массив строк или DOM-элементов с информацией об авторских правах.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coords* | — | Тип: Number[]<br><br>Точка на карте, для которой запрашивается информация об авторских правах. |
| zoom* | — | Тип: Number<br><br>Уровень масштабирования, для которого запрашивается информация об авторских правах. |

* Обязательный параметр/опция.

### remove

```
remove()
```

Удаляет добавленную через данный объект информацию об авторских правах.

### setCopyrights

```
setCopyrights(copyrights)
```

Задаёт новое значение для добавленной через данный объект информации об авторских правах.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| copyrights* | — | Тип: String \| HTMLElement \| String[] \| HTMLElement[]<br><br>Информация об авторских правах. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

# ICustomizable

Расширяет **IEventEmitter**.

Интерфейс объекта, который можно настраивать с помощью глобального реестра опций.

Конструктор | Поля | События

## Конструктор

```
ICustomizable()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций. |

## События

| Имя | Описание |
|---|---|
| **optionschange** | Изменение в опциях объекта. |

## Описание полей

### options

```
{IOptionManager} options
```

Менеджер опций.

## Описание событий

### optionschange

Изменение в опциях объекта.

# IDataManager

Расширяет **IEventEmitter**.

Интерфейс менеджера произвольных данных.

Конструктор | Поля | События | Методы

## Конструктор

```
IDataManager()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |

### События

| Имя | Описание |
|---|---|
| change | Изменение данных. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| get(path, defaultValue) | Object | Возвращает значение поля данных. |

## Описание событий

### change

Изменение данных.

## Описание методов

### get

```
{Object} get(path, defaultValue)
```

**Возвращает** значение поля данных.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| path* | — | Тип: String<br>Строка с именем свойства; может содержать '.'. |
| defaultValue* | — | Тип: Object<br>Значение по умолчанию. |

* Обязательный параметр/опция.

**Пример:**

```
dataManager.get("icon.shadow.size", [10, 10]);
```

# IDomEvent

Расширяет IEvent.

Объект DOM-события в системе API Яндекс.Карт. Оборачивает оригинальное DOM-событие браузера с целью нормализации имен полей данных. Т.е. посредством методов get и callMethod можно получать доступ к полям и методам оригинального DOM-события. При этом автоматически выполняется приведение к стандарту для тех полей и методов, реализация которых различна в разных браузерах. Так,


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

event.callMethod('stopPropagation') остановит распространение DOM-события во всех браузерах, включая Internet Explorer.

Конструктор | Методы

## Конструктор

```
IDomEvent(originalEvent)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| originalEvent * | — | Тип: Object<br><br>Исходное DOM-событие. |

* Обязательный параметр/опция.

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **allowMapEvent**() | | Разрешает распространение данного события на карту.<br><br>Унаследован от **IEvent**. |
| **callMethod**(name) | | Вызывает указанный метод у исходного события. Второй и последующие аргументы передаются методу при вызове.<br><br>Унаследован от **IEvent**. |
| **get**(name) | Object | Возвращает заданное свойство события. С помощью этого метода можно получать доступ как к свойствам исходного события, так и к дополнительным свойствам, предоставляемым API карт. |
| **getSourceEvent**() | **IDomEvent** | Возвращает исходное DOM-событие. |
| **isDefaultPrevented**() | Boolean | Возвращает true, если реакция по умолчанию на событие отменена.<br><br>Унаследован от **IEvent**. |
| **isImmediatePropagationStopped**() | Boolean | Возвращает true, если обработка события прервана.<br><br>Унаследован от **IEvent**. |
| **isMapEventAllowed**() | Boolean | Возвращает true, если событие карты разрешено.<br><br>Унаследован от **IEvent**. |
| **isPropagationStopped**() | Boolean | Возвращает true, если распространение события прервано,.<br><br>Унаследован от **IEvent**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| preventDefault() | | Отменяет реакцию по умолчанию на событие в рамках системы событий API Яндекс.Карт. Вызов этого метода не влияет на обработку браузером действия по умолчанию исходного DOM-события. |
| stopImmediatePropagation() | | Прекращает обработку события в системе событий API Яндекс.Карт. Т.е. после вызова данного метода ни один обработчик данного события не будет вызван. Вызов этого метода не влияет на обработку исходного DOM-события на уровне браузера. |
| stopPropagation() | | Прекращает распространение DOM-события в системе событий API Яндекс.Карт. Вызов этого метода не влияет на распространение исходного DOM-события по DOM-дереву. |

## Описание методов

### get

```
{Object} get(name)
```

Возвращает заданное свойство события. С помощью этого метода можно получать доступ как к свойствам исходного события, так и к дополнительным свойствам, предоставляемым API карт.

**Возвращает** значение свойства.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя свойства. Поддерживаются дополнительные свойства:<br>• 'propagatedData' — данные события, которые сохраняются при его пропагировании (распространении по DOM-дереву);<br>• 'position' — необязательное поле, которое содержит координаты события относительно документа. |

* Обязательный параметр/опция.

### getSourceEvent

```
{IDomEvent} getSourceEvent()
```

**Возвращает** исходное DOM-событие.

### preventDefault

```
preventDefault()
```

Отменяет реакцию по умолчанию на событие в рамках системы событий API Яндекс.Карт. Вызов этого метода не влияет на обработку браузером действия по умолчанию исходного DOM-события.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### stopImmediatePropagation

```
stopImmediatePropagation()
```

Прекращает обработку события в системе событий API Яндекс.Карт. Т.е. после вызова данного метода ни один обработчик данного события не будет вызван. Вызов этого метода не влияет на обработку исходного DOM-события на уровне браузера.

### stopPropagation

```
stopPropagation()
```

Прекращает распространение DOM-события в системе событий API Яндекс.Карт. Вызов этого метода не влияет на распространение исходного DOM-события по DOM-дереву.

## IDomEventEmitter

Расширяет **IEventEmitter**.

Интерфейс объекта, генерирующего "DOM-события". Данный интерфейс декларирует перечень событий, используемых для отражения интерактивных действий пользователя с различными объектами в среде API Яндекс Карт. Объекты, реализующие данный интерфейс, не обязаны фактически генерировать все события декларируемые интерфейсом.

Конструктор | Поля | События

### Конструктор

```
IDomEventEmitter()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий. |

### События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. |
| mousemove | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. |
| mouseup | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. |
| multitouchend | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent. |
| multitouchmove | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. |

## Описание полей

### events

```
{IEventManager} events
```

Менеджер событий.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание событий

### click

Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.

### contextmenu

Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.

### dblclick

Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.

### mousedown

Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.

### mouseenter

Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.

### mouseleave

Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.

### mousemove

Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.

### mouseup

Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.

### multitouchend

Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.

### multitouchmove

Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:

- clientX — координата касания X относительно видимой области браузера;
- clientY — координата касания Y относительно видимой области браузера;
- pageX — координата касания X касания относительно начала документа;
- pageY — координата касания Y касания относительно начала документа.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### multitouchstart

Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:

- clientX — координата касания X относительно видимой области браузера;
- clientY — координата касания Y относительно видимой области браузера;
- pageX — координата касания X касания относительно начала документа;
- pageY — координата касания Y касания относительно начала документа.

### wheel

Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.

## IDomTile

Интерфейс тайлов, представляющих из себя DOM-объект.

Конструктор | Поля | События | Методы

### Конструктор

```
IDomTile(url)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| url * | — | Тип: String<br><br>URL тайла. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий тайла. |

### События

| Имя | Описание |
|---|---|
| **ready** | Событие готовности тайла. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает тайл. |
| **isReady**() | Boolean | Проверяет готовность тайла. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **renderAt**(context, clientBounds, animate) | | Добавляет тайл в родительский HTML-элемент. |

## Описание полей

### events

```
{IEventManager} events
```

Менеджер событий тайла.

## Описание событий

### ready

Событие готовности тайла.

## Описание методов

### destroy

```
destroy()
```

Уничтожает тайл.

### isReady

```
{Boolean} isReady()
```

Проверяет готовность тайла.

**Возвращает** true — тайл готов, false — тайл не готов.

### renderAt

```
renderAt(context, clientBounds, animate)
```

Добавляет тайл в родительский HTML-элемент.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| context* | — | Тип: HTMLElement<br><br>Родительский HTML-элемент |
| clientBounds* | — | Тип: Number[][]<br><br>Область в клиентских координатах, которую должен занять тайл. |
| animate* | — | Тип: Boolean<br><br>true — анимировать рендеринг, false — нет. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IEvent

Событие, бросаемое менеджером событий IEventManager.

Конструктор | Методы

## Конструктор

```
IEvent()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **allowMapEvent**() | | Разрешает распространение данного события на карту. |
| **callMethod**(name) | | Вызывает указанный метод у исходного события. Второй и последующие аргументы передаются методу при вызове. |
| **get**(name) | Object | Возвращает свойство события по ключу. С помощью этого метода можно получать доступ как к свойствам исходного события, так и к дополнительным свойствам, предоставляемым API карт. |
| **getSourceEvent**() | **IEvent** \| null | Возвращает исходное событие. |
| **isDefaultPrevented**() | Boolean | Возвращает true, если реакция по умолчанию на событие отменена. |
| **isImmediatePropagationStopped**() | Boolean | Возвращает true, если обработка события прервана. |
| **isMapEventAllowed**() | Boolean | Возвращает true, если событие карты разрешено. |
| **isPropagationStopped**() | Boolean | Возвращает true, если распространение события прервано,. |
| **preventDefault**() | | Отменяет реакцию по умолчанию на событие в рамках системы событий API Яндекс.Карт. |
| **stopImmediatePropagation**() | | Прекращает обработку события в системе событий API Яндекс.Карт. Т.е. после вызова данного метода ни один обработчик данного события не будет вызван. |
| **stopPropagation**() | | Прекращает распространение события в системе событий API Яндекс.Карт. |

## Описание методов

### allowMapEvent

```
allowMapEvent()
```

Разрешает распространение данного события на карту.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## callMethod

```
callMethod(name)
```

Вызывает указанный метод у исходного события. Второй и последующие аргументы передаются методу при вызове.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя метода. |

* Обязательный параметр/опция.

## get

```
{Object} get(name)
```

Возвращает свойство события по ключу. С помощью этого метода можно получать доступ как к свойствам исходного события, так и к дополнительным свойствам, предоставляемым API карт.

**Возвращает** значение свойства.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя свойства. |

* Обязательный параметр/опция.

## getSourceEvent

```
{IEvent | null} getSourceEvent()
```

**Возвращает** исходное событие.

## isDefaultPrevented

```
{Boolean} isDefaultPrevented()
```

**Возвращает** true, если реакция по умолчанию на событие отменена.

## isImmediatePropagationStopped

```
{Boolean} isImmediatePropagationStopped()
```

**Возвращает** true, если обработка события прервана.

## isMapEventAllowed

```
{Boolean} isMapEventAllowed()
```

**Возвращает** true, если событие карты разрешено.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### isPropagationStopped

```
{Boolean} isPropagationStopped()
```

**Возвращает** true, если распространение события прервано,.

### preventDefault

```
preventDefault()
```

Отменяет реакцию по умолчанию на событие в рамках системы событий API Яндекс.Карт.

### stopImmediatePropagation

```
stopImmediatePropagation()
```

Прекращает обработку события в системе событий API Яндекс.Карт. Т.е. после вызова данного метода ни один обработчик данного события не будет вызван.

### stopPropagation

```
stopPropagation()
```

Прекращает распространение события в системе событий API Яндекс.Карт.

## IEventController

Интерфейс контроллера событий. Позволяет контролировать подписки и отписки от событий на определённом менеджере событий.

Конструктор | Методы

### Конструктор

```
IEventController()
```

### Методы

| Имя | Описание |
|---|---|
| **onStartListening**(events, type) | Вызывается при первой подписке на указанный тип событий через указанный менеджер событий. Этот метод является необязательным. |
| **onStopListening**(events, type) | Вызывается, когда определённый тип событий перестаёт слушаться на указанном менеджере событий (удалена последняя подписка). Этот метод является необязательным. |

### Описание методов

### onStartListening

```
onStartListening(events, type)
```

Вызывается при первой подписке на указанный тип событий через указанный менеджер событий. Этот метод является необязательным.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| events* | — | Тип: **IEventManager**<br><br>Менеджер событий. |
| type* | — | Тип: String<br><br>Тип события. |

* Обязательный параметр/опция.

### onStopListening

```
onStopListening(events, type)
```

Вызывается, когда определённый тип событий перестаёт слушаться на указанном менеджере событий (удалена последняя подписка). Этот метод является необязательным.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| events* | — | Тип: **IEventManager**<br><br>Менеджер событий. |
| type* | — | Тип: String<br><br>Тип события. |

* Обязательный параметр/опция.

## IEventEmitter

Интерфейс объекта, у которого можно слушать события.

Конструктор | Поля

### Конструктор

```
IEventEmitter()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий. |

## Описание полей

### events

```
{IEventManager} events
```

Менеджер событий.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IEventGroup

Группа слушателей событий.

Конструктор | Методы

## Конструктор

```
IEventGroup()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(types, callback[, context[, priority]]) | **IEventGroup** | Добавляет слушатель события. |
| **remove**(types, callback[, context[, priority]]) | **IEventGroup** | Удаляет слушатель события из группы. |
| **removeAll**() | **IEventGroup** | Удаляет всех слушателей событий из группы. |

## Описание методов

### add

```
{IEventGroup} add(types, callback[, context[, priority]])
```

Добавляет слушатель события.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| types* | — | Тип: String \| String[]<br><br>Тип события или массив типов. |
| callback* | — | Тип: Function<br><br>Функция-обработчик. В качестве параметра в функцию передается объект события. |
| context | — | Тип: Object<br><br>Контекст исполнения функции-обработчика. |
| priority | 0 | Тип: Integer<br><br>Приоритет подписки. |

* Обязательный параметр/опция.

### remove

```
{IEventGroup} remove(types, callback[, context[, priority]])
```

Удаляет слушатель события из группы.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| types* | — | Тип: String \| String[]<br><br>Тип или массив типов событий. |
| callback* | — | Тип: Function<br><br>Функция-обработчик. В качестве параметра в функцию передается объект события. |
| context | — | Тип: Object<br><br>Контекст исполнения функции-обработчика. |
| priority | 0 | Тип: Integer<br><br>Приоритет подписки. |

* Обязательный параметр/опция.

### removeAll

```
{IEventGroup} removeAll()
```

Удаляет всех слушателей событий из группы.

**Возвращает** ссылку на себя.

## IEventManager

Расширяет **IEventTrigger**.

Менеджер событий. При помощи менеджера событий можно подписываться на события и отписываться от них, а также инициировать сами события.

Конструктор | Методы

### Конструктор

```
IEventManager()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(types, callback[, context[, priority]]) | **IEventManager** | Добавляет новую подписку. |
| **fire**(type[, event]) | **IEventManager** | Инициирует событие. |
| **getParent**() | **IEventManager** \| null | Возвращает ссылку на родительский менеджер событий. |
| **group**() | **IEventGroup** | Возвращает группу слушателей событий, ассоциированную с данным менеджером событий. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **remove**(types, callback[, context[, priority]]) | **IEventManager** | Удаляет существующую подписку. |
| **setParent**(parent) | | Устанавливает родительский менеджер событий. |

## Описание методов

### add

```
{IEventManager} add(types, callback[, context[, priority]])
```

Добавляет новую подписку.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| types* | — | Тип: String \| String[]<br><br>Тип или массив типов события. |
| callback* | — | Тип: Function<br><br>Функция-обработчик события. В качестве параметра в функцию передается объект, описывающий событие. Может быть либо произвольным объектом, либо реализовывать интерфейс **IEvent**. |
| context | — | Тип: Object<br><br>Контекст исполнения обработчика. |
| priority | 0 | Тип: Integer<br><br>Приоритет подписки. |

* Обязательный параметр/опция.

### fire

```
{IEventManager} fire(type[, event])
```

Инициирует событие.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| type* | — | Тип: String<br><br>Тип события. |
| event | — | Тип: Object \| **Event** |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Событие. Если передан хэш с данными, то для него будет вызван метод createEventObject, и дальнейшие действия будут производиться с новым созданным событием. |

* Обязательный параметр/опция.

## getParent

```
{IEventManager | null} getParent()
```

**Возвращает** ссылку на родительский менеджер событий.

## group

```
{IEventGroup} group()
```

**Возвращает** группу слушателей событий, ассоциированную с данным менеджером событий.

## remove

```
{IEventManager} remove(types, callback[, context[, priority]])
```

Удаляет существующую подписку.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| types* | — | Тип: String \| String[]<br><br>Тип события или массив типов. |
| callback* | — | Тип: Function<br><br>Функция-обработчик события. В качестве параметра в функцию передается объект события **IEvent**. |
| context | — | Тип: Object<br><br>Контекст исполнения обработчика. |
| priority | 0 | Тип: Integer<br><br>Приоритет подписки. |

* Обязательный параметр/опция.

## setParent

```
setParent(parent)
```

Устанавливает родительский менеджер событий.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parent* | — | Тип: **IEventManager** \| null |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Родительский менеджер событий. |

* Обязательный параметр/опция.

# IEventPane

Расширяет **IDomEventEmitter**, **IPane**.

Интерфейс пейна событий карты. Позволяет отслеживать на карте DOM-события, но, в отличии от **IContainerPane**, не позволяет размещать отображения объектов карты внутри своего DOM-элемента.

Конструктор | Поля | События | Методы

## Конструктор

```
IEventPane()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **overflowchange** | Изменение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Экземпляр **IEvent**.<br><br>Унаследовано от **IPane**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **zindexchange** | Изменение значения zIndex пейна. Экземпляр **IEvent**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IPane**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает пейн.<br>Унаследован от **IPane**. |
| **getElement**() | HTMLElement | Возвращает ссылку на DOM-контейнер пейна.<br>Унаследован от **IPane**. |
| **getMap**() | **Map** | Возвращает карту, которой принадлежит пейн.<br>Унаследован от **IPane**. |
| **getOverflow**() | String | Возвращает значение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Данный параметр может принимать одно из следующих строковых значений:<br>• "visible" — при выходе за пределы контейнера карты содержимое пейна остается видимым;<br>• "hidden" — область отображения содержимого пейна ограничена контейнером карты.<br>.<br>Унаследован от **IPane**. |
| **getZIndex**() | Number | Возвращает zIndex пейна.<br>Унаследован от **IPane**. |

# IEventTrigger

Объект инициирующий события.

Конструктор | Методы

## Конструктор

```
IEventTrigger()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **fire**(type[, eventObject]) | **IEventTrigger** | Инициирует событие. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### fire

```
{IEventTrigger} fire(type[, eventObject])
```

Инициирует событие.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| type* | — | Тип: String<br><br>Тип события. |
| eventObject | — | Тип: Object \| **IEvent**<br><br>Объект, описывающий событие. Может быть либо произвольным объектом, либо реализовывать интерфейс **IEvent**. Во втором случае после вызова каждого обработчика будет проверяться значение метода isImmediatePropagationStopped() — если оно истинно, оповещение о событии немедленно прекращается. |

* Обязательный параметр/опция.

# IEventWorkflowController

Расширяет **IEventController**.

Интерфейс контроллера событий, позволяющего влиять на распространение событий по дереву.

Конструктор | Методы

## Конструктор

```
IEventWorkflowController()
```

## Методы

| Имя | Описание |
|---|---|
| **onAfterEventFiring**(events, type[, event]) | Функция, вызываемая после окончания обработки события менеджером событий. Этот метод является необязательным. |
| **onBeforeEventFiring**(events, type[, event]) | Функция, вызываемая до начала обработки события менеджером событий. Этот метод является необязательным. |
| **onStartListening**(events, type) | Вызывается при первой подписке на указанный тип событий через указанный менеджер событий. Этот метод является необязательным.<br><br>Унаследован от **IEventController**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **onStopListening**(events, type) | Вызывается, когда определённый тип событий перестаёт слушаться на указанном менеджере событий (удалена последняя подписка). Этот метод является необязательным.<br><br>Унаследован от **IEventController**. |

## Описание методов

### onAfterEventFiring

```
onAfterEventFiring(events, type[, event])
```

Функция, вызываемая после окончания обработки события менеджером событий. Этот метод является необязательным.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| events* | — | Тип: **IEventManager**<br><br>Менеджер событий. |
| type* | — | Тип: String<br><br>Тип события. |
| event | — | Тип: **IEvent**<br><br>Событие. |

* Обязательный параметр/опция.

### onBeforeEventFiring

```
onBeforeEventFiring(events, type[, event])
```

Функция, вызываемая до начала обработки события менеджером событий. Этот метод является необязательным.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| events* | — | Тип: **IEventManager**<br><br>Менеджер событий. |
| type* | — | Тип: String<br><br>Тип события. |
| event | — | Тип: **IEvent**<br><br>Событие. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IExpandableControlLayout

Расширяет **ILayout**.

Интерфейс макета, который может находится в свернутом и развернутом состоянии.

Конструктор | Поля | События | Методы

## Конструктор

```
IExpandableControlLayout()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **collapse** | Событие, инициирующее сворачивание объекта. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты макета. Экземпляр класса **Event**.<br>Унаследовано от **ILayout**. |
| **expand** | Событие, инициирующее разворачивание объекта. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **parentelementchange** | Изменение родительского элемента. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **shapechange** | Изменение фигуры области охватывающей макет. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Деструктор. Вызывается при завершении работы с макетом.<br>Унаследован от **ILayout**. |
| **getData**() | Object | Возвращает объект данных макета.<br>Унаследован от **ILayout**. |
| **getParentElement**() | HTMLElement | Возвращает родительский html-элемент.<br>Унаследован от **ILayout**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.<br>Унаследован от **ILayout**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **ILayout**. |
| **setData**(data) | | Задает данные макета.<br>Унаследован от **ILayout**. |
| **setParentElement**(parent) | | Добавляет макет в DOM-дерево.<br>Унаследован от **ILayout**. |

## Описание событий

### collapse

Событие, инициирующее сворачивание объекта.

### expand

Событие, инициирующее разворачивание объекта.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IFreezable

Интерфейс объекта с отключаемым событием изменения состояния. Объект, реализующий **IFreezable**, может функционировать в одном из следующих режимов:

- 1. Активный. В этом режиме каждое изменение внутреннего состояния объекта сопровождается генерацией события IFreezable.change.
- 2. Замороженный. В этом режиме изменения состояния объекта не сопровождаются событием IFreezable.change, но если изменения имели место, событие IFreezable.change будет сгенерировано один раз при переходе в активный режим.

Конструктор | Поля | События | Методы

## Конструктор

```
IFreezable()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий объекта. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение внутреннего состояния объекта. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим. |

## Описание полей

### events

```
{IEventManager} events
```

Менеджер событий объекта.

## Описание событий

### change

Изменение внутреннего состояния объекта.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### freeze

```
{IFreezable} freeze()
```

Переводит объект в "замороженный" режим.

**Возвращает** Ссылку на себя.

### isFrozen

```
{Boolean} isFrozen()
```

**Возвращает** true, если объект находится в "замороженном" режиме, иначе false.

### unfreeze

```
{IFreezable} unfreeze()
```

Переводит объект в активный режим.

**Возвращает** Ссылку на себя.

# IGeocodeProvider

Интерфейс провайдера геокодера.

Конструктор | Методы

## Конструктор

```
IGeocodeProvider()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **geocode**(request[, options]) | **vow.Promise** | Отправляет запрос на геокодирование. Добавить функцию-обработчик для обработки результата геокодирования можно через возвращаемый объект-promise. Объект, поступающий на вход функции-обработчика может содержать поля только следующего вида: geoObjects, layers, mapState, metaData. |
| **suggest**(request[, options]) | **vow.Promise** | |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Отправляет запрос за поисковыми подсказками. Возвращает объект-promise, который будет либо отклонён с ошибкой, либо подтверждён массивом объектов вида { displayName: "Мытищи, Московская область", value: "Россия, Московская область, Мытищи " }. Поле displayName отвечает за представление топонима в удобном для чтения пользователем виде, поле value — значение, которое необходимо подставить в поле ввода после выбора пользователем данной подсказки.<br><br>Этот метод является опциональным. |

## Описание методов

### geocode

```
{vow.Promise} geocode(request[, options])
```

Отправляет запрос на геокодирование. Добавить функцию-обработчик для обработки результата геокодирования можно через возвращаемый объект-promise. Объект, поступающий на вход функции-обработчика может содержать поля только следующего вида: geoObjects, layers, mapState, metaData.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| request* | — | Тип: String<br><br>Строка запроса. |
| options | — | Тип: Object<br><br>Опции. |
| options.boundedBy | — | Тип: Number[][]<br><br>Прямоугольная область на карте, где предположительно находится искомый объект. Должна быть задана как массив, например [[30, 40], [50, 50]]. |
| options.results | — | Тип: Number<br><br>Максимальное количество возвращаемых результатов. |
| options.skip | — | Тип: Number<br><br>Число результатов, которое необходимо пропустить. |
| options.strictBounds | — | Тип: Boolean<br><br>Искать только внутри области, заданной опцией boundedBy. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

## suggest

```
{vow.Promise} suggest(request[, options])
```

Отправляет запрос за поисковыми подсказками. Возвращает объект-promise, который будет либо отклонён с ошибкой, либо подтверждён массивом объектов вида { displayName: "Мытищи, Московская область", value: "Россия, Московская область, Мытищи " }. Поле displayName отвечает за представление топонима в удобном для чтения пользователем виде, поле value — значение, которое необходимо подставить в поле ввода после выбора пользователем данной подсказки.

Этот метод является опциональным.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| request* | — | Тип: String<br><br>Строка запроса. |
| options | — | Тип: Object<br><br>Опции. |
| options.boundedBy | — | Тип: Number[][]<br><br>Прямоугольная область на карте, где предположительно находится искомый объект. Должна быть задана как массив, например [[30, 40], [50, 50]]. |
| options.results | — | Тип: Number<br><br>Максимальное количество возвращаемых результатов. |
| options.strictBounds | — | Тип: Boolean<br><br>Искать только внутри области, заданной опцией boundedBy. |

* Обязательный параметр/опция.

# IGeometry

Расширяет **IBaseGeometry**, **ICustomizable**.

Интерфейс геометрии.

Конструктор | Поля | События | Методы

## Конструктор

```
IGeometry()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| mapchange | Сменилась карта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| pixelgeometrychange | Изменилась пиксельная геометрия. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия IPixelGeometry. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getBounds() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно. |
| getMap() | Map \| null | Возвращает текущую карту. |
| getPixelGeometry([options]) | IPixelGeometry | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты. |
| getType() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от IBaseGeometry. |
| setMap(map) | | Задает карту. |

## Описание событий

### mapchange

Сменилась карта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:

- oldMap — старая карта;
- newMap — новая карта.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### pixelgeometrychange

Изменилась пиксельная геометрия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- pixelGeometry — новая пиксельная геометрия **IPixelGeometry**.

## Описание методов

### getBounds

```
{Number[][] | null} getBounds()
```

**Возвращает** координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.

**Пример:**

```
// Задаем центр и масштаб карты таким образом, чтобы геометрия целиком
отобразилась в видимой области.
map.setBounds(myGeometry.getBounds());
```

### getMap

```
{Map | null} getMap()
```

**Возвращает** текущую карту.

### getPixelGeometry

```
{IPixelGeometry} getPixelGeometry([options])
```

**Возвращает** пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options | — | Тип: Object<br><br>Хэш опций, позволяющий перекрыть часть текущих опций геометрии для данного расчета. |

**Пример:**

```
// Получаем пиксельное представление геометрии с учетом геодезии, но без
оптимизации удаления невидимых точек.
myGeometry.getPixelGeometry({
    geodesic: true,
    simplification: false
}).getCoordinates();
```

### setMap

```
setMap(map)
```

Задает карту.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map** \| null<br><br>Ссылка на карту. |

* Обязательный параметр/опция.

# IGeometryEditor

Расширяет **ICustomizable**, **IEventEmitter**.

Интерфейс редактора геометрии.

Конструктор | Поля | События | Методы

## Конструктор

```
IGeometryEditor()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **geometry** | **IGeometry** | Редактируемая геометрия. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |
| **state** | **IDataManager** | Состояние редактора геометрии. |

## События

| Имя | Описание |
|---|---|
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **statechange** | Изменение состояния редактора геометрии. Экземпляр класса **Event**. |

## Методы

| Имя | Описание |
|---|---|
| **startEditing**() | Включает режим редактирования. |
| **stopEditing**() | Отключает режим редактирования. |

## Описание полей

### geometry

```
{IGeometry} geometry
```

Редактируемая геометрия.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### state

```
{IDataManager} state
```

Состояние редактора геометрии.

## Описание событий

### statechange

Изменение состояния редактора геометрии. Экземпляр класса **Event**.

## Описание методов

### startEditing

```
startEditing()
```

Включает режим редактирования.

### stopEditing

```
stopEditing()
```

Отключает режим редактирования.

# IGeometryEditorChildModel

Расширяет **IGeometryEditorModel**.

Интерфейс дочерней модели данных.

Конструктор | Поля | Методы

## Конструктор

```
IGeometryEditorChildModel(geometry, editor, pixels, parent)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IBaseGeometry**<br><br>Редактируемая дочерняя геометрия. Интерфейс IBaseGeometry не подразумевает наличие события pixelgeometrychange, поэтому пиксельные данные получаются от родительской модели данных. |
| editor * | — | Тип: **IGeometryEditor**<br><br>Ссылка на редактор геометрии. |
| pixels * | — | Тип: Number[]<br><br>Пиксельные данные модели. |
| parent * | — | Тип: **IGeometryEditorModel** |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Родительская модель данных. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| editor | IGeometryEditor | Редактор геометрии. |
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| geometry | IBaseGeometry | Геометрия модели. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Деструктор.<br>Унаследован от IGeometryEditorModel. |
| getParent() | IGeometryEditorModel | Возвращает родительскую модель данных. |
| getPixels() | Number[] | Возвращает пиксельные данные модели.<br>Унаследован от IGeometryEditorModel. |
| setPixels(pixels) | | Задает пиксельные данные модели. |

## Описание полей

### editor

```
{IGeometryEditor} editor
```

Редактор геометрии.

### geometry

```
{IBaseGeometry} geometry
```

Геометрия модели.

## Описание методов

### getParent

```
{IGeometryEditorModel} getParent()
```

**Возвращает** родительскую модель данных.

### setPixels

```
setPixels(pixels)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Задает пиксельные данные модели.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| pixels* | — | Тип: Number[]<br><br>Пиксельные данные. |

* Обязательный параметр/опция.

# IGeometryEditorModel

Расширяет **IEventEmitter**.

Интерфейс модели данных редактора геометрии.

Конструктор | Поля | Методы

## Конструктор

```
IGeometryEditorModel(geometry, editor)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IBaseGeometry**<br><br>Редактируемая геометрия. |
| editor * | — | Тип: **IGeometryEditor**<br><br>Ссылка на редактор геометрии. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Деструктор. |
| **getPixels**() | Number[] | Возвращает пиксельные данные модели. |

## Описание методов

### destroy

```
destroy()
```

Деструктор.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### getPixels

```
{Number[]} getPixels()
```

**Возвращает** пиксельные данные модели.

## IGeometryEditorRootModel

Расширяет **IGeometryEditorModel**.

Интерфейс корневой модели данных.

Конструктор | Поля | Методы

### Конструктор

```
IGeometryEditorRootModel(geometry, editor)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry * | — | Тип: **IGeometry**<br><br>Редактируемая геометрия. |
| editor * | — | Тип: **IGeometryEditor**<br><br>Ссылка на редактор геометрии. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Деструктор.<br><br>Унаследован от **IGeometryEditorModel**. |
| **getPixels**() | Number[] | Возвращает пиксельные данные модели.<br><br>Унаследован от **IGeometryEditorModel**. |

## IGeometryJson

Интерфейс объекта, описывающего JSON-представление геометрии.

Конструктор | Поля


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Конструктор

```
IGeometryJson()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| type | String | Идентификатор типа геометрии. |

## Описание полей

### type

```
{String} type
```

Идентификатор типа геометрии.

# IGeoObject

Расширяет **IChildOnMap**, **ICustomizable**, **IDomEventEmitter**, **IParentOnMap**.

Интерфейс геообъекта.

Конструктор | Поля | События | Методы

### Конструктор

```
IGeoObject()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| geometry | IGeometry \| null | Геометрия геообъекта. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от **ICustomizable**. |
| properties | IDataManager | Данные геообъекта. |
| state | IDataManager | Состояние геообъекта. |

### События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| | Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **geometrychange** | Изменение геометрии геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• originalEvent: **IEvent** — оригинальное событие геометрии. |
| **mapchange** | Сменилась карта. Поля данных:<br><br>• oldMap — старая карта;<br>• newMap — новая карта.<br><br>Унаследовано от **IParentOnMap**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **overlaychange** | Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• overlay: **IOverlay**\|null — ссылка на оверлей;<br>• oldOverlay: **IOverlay**\|null — предыдущий оверлей геообъекта. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **propertieschange** | Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• originalEvent: **IEvent** — оригинальное событие менеджера данных. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getOverlay() | vow.Promise | Возвращает объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки. |
| getOverlaySync() | IOverlay \| null | Метод предоставляет синхронный доступ к оверлею. |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IChildOnMap. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IChildOnMap. |

## Описание полей

### geometry

```
{IGeometry | null} geometry
```

Геометрия геообъекта.

### properties

```
{IDataManager} properties
```

Данные геообъекта.

### state

```
{IDataManager} state
```

Состояние геообъекта.

## Описание событий

### geometrychange

Изменение геометрии геообъекта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:

- originalEvent: IEvent — оригинальное событие геометрии.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### overlaychange

Изменение оверлея геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- overlay: **IOverlay**|null — ссылка на оверлей;
- oldOverlay: **IOverlay**|null — предыдущий оверлей геообъекта.

### propertieschange

Изменение данных геообъекта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- originalEvent: **IEvent** — оригинальное событие менеджера данных.

## Описание методов

### getOverlay

```
{vow.Promise} getOverlay()
```

**Возвращает** объект-обещание, который подтверждается объектом оверлея в момент его фактического создания, либо отклоняется с передачей соответствующей ошибки.

### getOverlaySync

```
{IOverlay | null} getOverlaySync()
```

Метод предоставляет синхронный доступ к оверлею.

**Возвращает** ссылку на оверелей, либо null, если оверлей отсутствует.

# IGeoObjectCollection

Расширяет **ICustomizable**, **IEventEmitter**, **IParentOnMap**.

Интерфейс коллекции геообъектов.

Конструктор | Поля | События | Методы

## Конструктор

```
IGeoObjectCollection()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| **add** | Добавлен (вставлен) дочерний геообъект. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• index: Integer — индекс добавленного геообъекта;<br>• child: **IGeoObject** — ссылка на добавленный геообъект. |
| **boundschange** | Изменение координат географической области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса **Event**. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IParentOnMap**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **pixelboundschange** | Изменение пиксельных координат области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса **Event**. |
| **remove** | Удален дочерний геообъект. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• index: Integer — индекс удаленного геообъекта;<br>• child: **IGeoObject** — ссылка на удаленный геообъект. |
| **set** | В коллекцию записан новый дочерний геообъект. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• index: Integer — индекс геообъекта;<br>• child: **IGeoObject** — ссылка на новый геообъект;<br>• prevChild: **IGeoObject** — ссылка на предыдущее значение для данного индекса. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **add**(child[, index]) | **IGeoObjectCollection** | Добавляет (вставляет) дочерний геообъект в коллекцию. |
| **each**(callback[, context]) | | Для каждого дочернего геообъекта, вызывает функцию-обработчик. |
| **get**(index) | **IGeoObject** | Возвращает дочерний геообъект с заданным индексом. |
| **getBounds**() | Number[][] \| null | Возвращает географические координаты области, охватывающей коллекции и ее дочерние геообъекты. |
| **getIterator**() | **IIterator** | Возвращает итератор по коллекции. |
| **getLength**() | Integer | Возвращает длину коллекции. |
| **getMap**() | **Map** | Возвращает ссылку на карту.<br>Унаследован от **IParentOnMap**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getPixelBounds**() | Number[][] \| null | Возвращает глобальные пиксельные координаты области, охватывающей коллекцию и ее дочерние геообъекты. |
| **indexOf**(object) | Integer | Возвращает индекс дочернего геообъекта. Если геообъект в коллекции не найден, возвращается -1. |
| **remove**(child) | **IGeoObjectCollection** | Удаляет дочерний геообъект из коллекции. |
| **removeAll**() | **IGeoObjectCollection** | Очищает коллекцию. |
| **set**(index, child) | **IGeoObjectCollection** | Записывает в коллекцию новый дочерний геообъект. |
| **splice**(index, number) | **IGeoObjectCollection** | Удаляет геообъекты из коллекции. Если необходимо, вставляет на их место другие. Объекты, которые будут вставлены вместо удаляемых, передаются как дополнительные параметры (после параметра number). |

## Описание событий

### add

Добавлен (вставлен) дочерний геообъект. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- index: Integer — индекс добавленного геообъекта;
- child: **IGeoObject** — ссылка на добавленный геообъект.

### boundschange

Изменение координат географической области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса **Event**.

### pixelboundschange

Изменение пиксельных координат области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса **Event**.

### remove

Удален дочерний геообъект. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- index: Integer — индекс удаленного геообъекта;
- child: **IGeoObject** — ссылка на удаленный геообъект.

### set

В коллекцию записан новый дочерний геообъект. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- index: Integer — индекс геообъекта;
- child: **IGeoObject** — ссылка на новый геообъект;
- prevChild: **IGeoObject** — ссылка на предыдущее значение для данного индекса.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### add

```
{IGeoObjectCollection} add(child[, index])
```

Добавляет (вставляет) дочерний геообъект в коллекцию.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **IGeoObject**<br><br>Дочерний геообъект. |
| index | — | Тип: Integer<br><br>Индекс, в который добавляется новый объект. По умолчанию объект добавляется в конец коллекции. |

* Обязательный параметр/опция.

### each

```
each(callback[, context])
```

Для каждого дочернего геообъекта, вызывает функцию-обработчик.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик. |
| context | — | Тип: Object<br><br>Контекст выполнения функции-обработчика. |

* Обязательный параметр/опция.

### get

```
{IGeoObject} get(index)
```

Возвращает дочерний геообъект с заданным индексом.

**Возвращает** дочерний геообъект.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getBounds

```
{Number[][] | null} getBounds()
```

**Возвращает** географические координаты области, охватывающей коллекции и ее дочерние геообъекты.

## getIterator

```
{IIterator} getIterator()
```

**Возвращает** итератор по коллекции.

## getLength

```
{Integer} getLength()
```

**Возвращает** длину коллекции.

## getPixelBounds

```
{Number[][] | null} getPixelBounds()
```

**Возвращает** глобальные пиксельные координаты области, охватывающей коллекцию и ее дочерние геообъекты.

## indexOf

```
{Integer} indexOf(object)
```

**Возвращает** индекс дочернего геообъекта. Если геообъект в коллекции не найден, возвращается -1.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| object* | — | Тип: Object<br><br>Дочерний геообъект. |

* Обязательный параметр/опция.

## remove

```
{IGeoObjectCollection} remove(child)
```

Удаляет дочерний геообъект из коллекции.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| child* | — | Тип: **IGeoObject**<br><br>Удаляемый геообъект. |

* Обязательный параметр/опция.

## removeAll

```
{IGeoObjectCollection} removeAll()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Очищает коллекцию.

**Возвращает** ссылку на себя.

### set

```
{IGeoObjectCollection} set(index, child)
```

Записывает в коллекцию новый дочерний геообъект.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс. |
| child* | — | Тип: **IGeoObject**<br><br>Дочерний геообъект. |

* Обязательный параметр/опция.

### splice

```
{IGeoObjectCollection} splice(index, number)
```

Удаляет геообъекты из коллекции. Если необходимо, вставляет на их место другие. Объекты, которые будут вставлены вместо удаляемых, передаются как дополнительные параметры (после параметра number).

**Возвращает** коллекция удаленных геообъектов.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс геообъекта, с которого начинается удаление. |
| number* | — | Тип: Integer<br><br>Количество удаляемых геообъектов. |

* Обязательный параметр/опция.

## IGeoObjectPopupData

Статический объект.

Объект с данными, передаваемый геообъектом своему инфо-объекту (например, балуну или хинту).

Поля

```
IGeoObjectPopupData()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Поля

| Имя | Тип | Описание |
|---|---|---|
| geometry | IGeometry | Ссылка на геометрию геообъекта. |
| geoObject | IGeoObject | Ссылка на геообъект. |
| properties | IDataManager | Ссылка на свойства геообъекта. |
| userData | Object | Пользовательские данные. |

## Описание полей

### geometry

```
{IGeometry} geometry
```

Ссылка на геометрию геообъекта.

### geoObject

```
{IGeoObject} geoObject
```

Ссылка на геообъект.

### properties

```
{IDataManager} properties
```

Ссылка на свойства геообъекта.

### userData

```
{Object} userData
```

Пользовательские данные.

# IGeoObjectSequence

Расширяет ICustomizable, IEventEmitter, IParentOnMap.

Интерфейс неизменяемой коллекции геообъектов.

Конструктор | Поля | События | Методы

### Конструктор

```
IGeoObjectSequence()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| boundschange | Изменение координат географической области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса Event. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| pixelboundschange | Изменение пиксельных координат области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса Event. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| each(callback[, context]) | | Для каждого дочернего геообъекта, вызывает функцию-обработчик. |
| get(index) | IGeoObject | Возвращает дочерний геообъект с заданным индексом. |
| getBounds() | Number[][] \| null | Возвращает географические координаты области, охватывающей коллекции и ее дочерние геообъекты. |
| getIterator() | IIterator | Возвращает итератор по дочерним геообъектам коллекции. |
| getLength() | Integer | Возвращает длину коллекции. |
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| getPixelBounds() | Number[][] \| null | Возвращает глобальные пиксельные координаты области, охватывающей коллекцию и ее дочерние геообъекты. |
| indexOf(object) | Integer | Возвращает индекс дочернего геообъекта. Если геообъект в коллекции не найден, возвращается -1. |

## Описание событий

### boundschange

Изменение координат географической области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса Event.

### pixelboundschange

Изменение пиксельных координат области, охватывающей коллекцию и ее дочерние геообъекты. Экземпляр класса Event.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### each

```
each(callback[, context])
```

Для каждого дочернего геообъекта, вызывает функцию-обработчик.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| callback* | — | Тип: Function<br><br>Функция-обработчик. |
| context | — | Тип: Object<br><br>Контекст выполнения функции-обработчика. |

* Обязательный параметр/опция.

### get

```
{IGeoObject} get(index)
```

Возвращает дочерний геообъект с заданным индексом.

**Возвращает** дочерний геообъект.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс. |

* Обязательный параметр/опция.

### getBounds

```
{Number[][] | null} getBounds()
```

**Возвращает** географические координаты области, охватывающей коллекции и ее дочерние геообъекты.

### getIterator

```
{IIterator} getIterator()
```

**Возвращает** итератор по дочерним геообъектам коллекции.

### getLength

```
{Integer} getLength()
```

**Возвращает** длину коллекции.

### getPixelBounds

```
{Number[][] | null} getPixelBounds()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** глобальные пиксельные координаты области, охватывающей коллекцию и ее дочерние геообъекты.

### indexOf

```
{Integer} indexOf(object)
```

**Возвращает** индекс дочернего геообъекта. Если геообъект в коллекции не найден, возвращается -1.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| object* | — | Тип: **IGeoObject**<br><br>Дочерний геообъект. |

* Обязательный параметр/опция.

## IGroupControlLayout

Расширяет **ILayout**.

Интерфейс макета группового элемента управления.

Конструктор | Поля | События | Методы

### Конструктор

```
IGroupControlLayout(data)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data * | — | Тип: Object<br><br>Данные макета. |

* Обязательный параметр/опция.

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |

### События

| Имя | Описание |
|---|---|
| **childcontainerchange** | Изменение контейнера дочерних элементов.<br><br>Поля данных:<br><br>• newChildContainerElement — новый элемент для дочерних объектов;<br>• oldChildContainerElement — старый элемент для дочерних объектов. |
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты макета. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchmove | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| parentelementchange | Изменение родительского элемента. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| shapechange | Изменение фигуры области охватывающей макет. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Деструктор. Вызывается при завершении работы с макетом.<br>Унаследован от ILayout. |
| getChildContainerElement() | Object | Возвращает ссылку на DOM-элемент, в который должны быть добавлены дочерние элементы. |
| getData() | Object | Возвращает объект данных макета. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **ILayout**. |
| **getParentElement**() | HTMLElement | Возвращает родительский html-элемент.<br>Унаследован от **ILayout**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.<br>Унаследован от **ILayout**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **ILayout**. |
| **setData**(data) | | Задает данные макета.<br>Унаследован от **ILayout**. |
| **setParentElement**(parent) | | Добавляет макет в DOM-дерево.<br>Унаследован от **ILayout**. |

## Описание событий

### childcontainerchange

Изменение контейнера дочерних элементов.

Поля данных:

- newChildContainerElement — новый элемент для дочерних объектов;
- oldChildContainerElement — старый элемент для дочерних объектов.

## Описание методов

### getChildContainerElement

```
{Object} getChildContainerElement()
```

**Возвращает** ссылку на DOM-элемент, в который должны быть добавлены дочерние элементы.

# IHint

Расширяет **IPopup**.

Интерфейс хинта.

Конструктор | Поля | События | Методы

## Конструктор

```
IHint()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| close | Закрытие инфо-объекта.<br>Унаследовано от IPopup. |
| open | Открытие инфо-объекта.<br>Унаследовано от IPopup. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| close([force]) | vow.Promise | Закрывает инфо-объект.<br>Унаследован от IPopup. |
| getData() | | Возвращает данные инфо-объекта.<br>Унаследован от IPopup. |
| getOverlay() | vow.Promise | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от IPopup. |
| getOverlaySync() | IOverlay | Возвращает оверлей, если тот существует.<br>Унаследован от IPopup. |
| getPosition() | | Возвращает координаты инфо-объекта.<br>Унаследован от IPopup. |
| isOpen() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от IPopup. |
| open([position[, data]]) | vow.Promise | Открывает инфо-объект в указанной позиции. Если инфо-объект уже открыт, перемещает его в указанную точку. Формат и суть координат определяется проекцией IProjection, которая содержится в опциях.<br>Унаследован от IPopup. |
| setData(data) | vow.Promise | Задаёт инфо-объекту новые данные. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IPopup**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopup**. |

# IHintManager

Расширяет **IPopupManager**.

Интерфейс менеджера хинта.

Конструктор | Поля | Методы

## Конструктор

```
IHintManager()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **close**([force]) | **vow.Promise** | Закрывает инфо-объект.<br>Унаследован от **IPopupManager**. |
| **destroy**() | | Отключает менеджер инфо-объекта.<br>Унаследован от **IPopupManager**. |
| **getData**() | Object \| null | Возвращает данные инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |
| **getOptions**() | **IOptionManager** \| null | Возвращает менеджер опций или `null`.<br>Унаследован от **IPopupManager**. |
| **getOverlay**() | **vow.Promise** | Возвращает объект-обещание вернуть оверлей.<br>Унаследован от **IPopupManager**. |
| **getOverlaySync**() | **IOverlay** \| null | Возвращает оверлей, если тот существует.<br>Унаследован от **IPopupManager**. |
| **getPosition**() | Number[] \| null | Возвращает координаты инфо-объекта или `null`.<br>Унаследован от **IPopupManager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **isOpen**() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт.<br>Унаследован от **IPopupManager**. |
| **open**([position[, data[, options]]]) | **vow.Promise** | Открывает инфо-объект в указанной позиции.<br>Унаследован от **IPopupManager**. |
| **setData**(data) | **vow.Promise** | Задаёт инфо-объекту новые данные.<br>Унаследован от **IPopupManager**. |
| **setOptions**(options) | **vow.Promise** | Задаёт инфо-объекту новые опции.<br>Унаследован от **IPopupManager**. |
| **setPosition**(position) | **vow.Promise** | Задаёт инфо-объекту новую позицию.<br>Унаследован от **IPopupManager**. |

# IHintOwner

Объект, имеющий хинт, доступный через свойство "hint".

Конструктор | Поля | События

## Конструктор

```
IHintOwner()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **hint** | **IHintManager** | Хинт объекта. |

## События

| Имя | Описание |
|---|---|
| **hintclose** | Скрытие хинта. |
| **hintopen** | Открытие хинта. |

## Описание полей

### hint

```
{IHintManager} hint
```

Хинт объекта.

## Описание событий

### hintclose

Скрытие хинта.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### hintopen

Открытие хинта.

## IHotspot

Расширяет **IDomEventEmitter**.

Интерфейс фигуры, определяющей геометрию активной области.

---

**Примечание:**
Фигура должна задаваться в глобальных пиксельных координатах.

---

Конструктор | Поля | События | Методы

### Конструктор

```
IHotspot()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |

### События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getShape() | IShape | Возвращает фигуру активной области. |
| getZIndex() | Number | Возвращает zIndex активной области. |
| setShape(shape) | | Задает фигуру активной области. |
| setZIndex(zIndex) | | Задает z-index активной области. |

## Описание методов

### getShape

```
{IShape} getShape()
```

**Возвращает** фигуру активной области.

### getZIndex

```
{Number} getZIndex()
```

**Возвращает** zIndex активной области.

### setShape

```
setShape(shape)
```

Задает фигуру активной области.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| shape* | — | Тип: IShape<br><br>Фигура. |

* Обязательный параметр/опция.

### setZIndex

```
setZIndex(zIndex)
```

Задает z-index активной области.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| zIndex* | — | Тип: Number<br><br>z-index, который будет задан для активной области. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IHotspotLayerObject

Расширяет **ICustomizable**, **IDomEventEmitter**.

Интерфейс объекта слоя активных областей.

---

**Примечание:**
Фигура, описывающая активную область, должна задаваться в глобальных пиксельных координатах.

---

Конструктор | Поля | События | Методы

## Конструктор

```
IHotspotLayerObject()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| | Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **shapechange** | Изменение фигуры, описывающей активную область. Экземпляр класса **Event**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getGeometry**() | Object | Возвращает реальную геометрию объекта. |
| **getHotspot**() | **IHotspot** | Возвращает объект, описывающий активную область. |
| **getId**() | Number | Возвращает идентификатор объекта. |
| **getProperties**() | Object | Возвращает данные объекта. |
| **setGeometry**(geometry) | | Задает реальную геометрию объекта. |
| **setId**(id) | | Задает идентификатор объекта. |
| **setProperties**(properties) | | Задает данные объекта. |

## Описание событий

### shapechange

Изменение фигуры, описывающей активную область. Экземпляр класса **Event**.

## Описание методов

### getGeometry

```
{Object} getGeometry()
```

**Возвращает** реальную геометрию объекта.

### getHotspot

```
{IHotspot} getHotspot()
```

**Возвращает** объект, описывающий активную область.

### getId

```
{Number} getId()
```

**Возвращает** идентификатор объекта.

### getProperties

```
{Object} getProperties()
```

**Возвращает** данные объекта.

### setGeometry

```
setGeometry(geometry)
```

Задает реальную геометрию объекта.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry* | — | Тип: Object<br><br>Реальная геометрия объекта. |

* Обязательный параметр/опция.

### setId

```
setId(id)
```

Задает идентификатор объекта.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| id* | — | Тип: Number<br><br>Идентификатор объекта. |

* Обязательный параметр/опция.

### setProperties

```
setProperties(properties)
```

Задает данные объекта.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| properties* | — | Тип: Object<br><br>Данные объекта. |

* Обязательный параметр/опция.

## IHotspotObjectSource

Расширяет **ICustomizable**.

Источник объектов для слоёв активных областей.

Конструктор | Поля | События | Методы

### Конструктор

```
IHotspotObjectSource()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |

## Методы

| Имя | Описание |
|---|---|
| cancelLastRequest() | Отменяет последний запрос за данными. |
| requestObjects(layer, tileNumber, zoom, callback) | Строит массив объектов IHotspotLayerObject, соответствующих определенному слою, тайлу и коэффициенту масштабирования карты, и передает его в функцию callback. |

# Описание методов

## cancelLastRequest

```
cancelLastRequest()
```

Отменяет последний запрос за данными.

## requestObjects

```
requestObjects(layer, tileNumber, zoom, callback)
```

Строит массив объектов IHotspotLayerObject, соответствующих определенному слою, тайлу и коэффициенту масштабирования карты, и передает его в функцию callback.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| layer* | — | Тип: hotspot.Layer<br>Слой активных областей. |
| tileNumber* | — | Тип: Number[]<br>Тайловые координаты. |
| zoom* | — | Тип: Number<br>Коэффициент масштабирования. |
| callback* | — | Тип: Function<br>Функция-обработчик. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IHotspotShape

Расширяет **ICustomizable**, **IDomEventEmitter**.

Интерфейс фигуры, определяющей геометрию активной области.

---

**Примечание:**
Не поддерживается начиная с версии 2.1.

---

Конструктор | Поля | События | Методы

## Конструктор

```
IHotspotShape()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **containerchange** | Изменение контейнера. Экземпляр класса **Event**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **optionschange** | Изменение в опциях объекта.<br><br>Унаследовано от **ICustomizable**. |
| **shapechange** | Изменение фигуры, описывающей активную область. Экземпляр класса **Event**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getContainer() | IHotspotContainer | Возвращает контейнер активных областей. |
| getGeometry() | Object | Возвращает реальную геометрию фигуры. |
| getId() | Number | Возвращает идентификатор объекта. |
| getProperties() | Object | Возвращает данные объекта. |
| getShape() | IShape | Возвращает фигуру, описывающую активную область. |
| setGeometry(geometry) | | Задает реальную геометрию фигуры. |
| setId(id) | | Задает идентификатор объекта. |
| setProperties(properties) | | Задает данные объекта. |
| setShape(shape) | | |

## Описание событий

### containerchange

Изменение контейнера. Экземпляр класса Event.

### shapechange

Изменение фигуры, описывающей активную область. Экземпляр класса Event.

## Описание методов

### getContainer

```
{IHotspotContainer} getContainer()
```

**Возвращает** контейнер активных областей.

### getGeometry

```
{Object} getGeometry()
```

**Возвращает** реальную геометрию фигуры.

### getId

```
{Number} getId()
```

**Возвращает** идентификатор объекта.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getProperties

```
{Object} getProperties()
```

**Возвращает** данные объекта.

## getShape

```
{IShape} getShape()
```

**Возвращает** фигуру, описывающую активную область.

## setGeometry

```
setGeometry(geometry)
```

Задает реальную геометрию фигуры.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry* | — | Тип: Object<br><br>Реальная геометрия фигуры. |

* Обязательный параметр/опция.

## setId

```
setId(id)
```

Задает идентификатор объекта.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| id* | — | Тип: Number<br><br>Идентификатор объекта. |

* Обязательный параметр/опция.

## setProperties

```
setProperties(properties)
```

Задает данные объекта.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| properties* | — | Тип: Object<br><br>Данные объекта. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### setShape

```
setShape(shape)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| shape* | — | Тип: **IShape**<br><br>Фигура, описывающая активную область. |

* Обязательный параметр/опция.

## IIterator

Интерфейс итератора. Итератор позволяет перебрать все составляющие элементы той или иной коллекции.

Конструктор | Поля | Методы

### Конструктор

```
IIterator()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **STOP_ITERATION** | Object | Объект, возвращаемый методом IIterator.getNext по окончанию перебора. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getNext**() | Object \| IIterator.STOP_ITERATION | Возвращает ссылку на очередной элемент коллекции. Возвращает объект IIterator.STOP_ITERATION, если перебор завершен. |

## Описание полей

### STOP_ITERATION

```
{Object} STOP_ITERATION
```

Объект, возвращаемый методом IIterator.getNext по окончанию перебора.

## Описание методов

### getNext

```
{Object | IIterator.STOP_ITERATION} getNext()
```

Возвращает ссылку на очередной элемент коллекции. Возвращает объект IIterator.STOP_ITERATION, если перебор завершен.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** ссылку на очередной объект или IIterator.STOP_ITERATION.

# ILayer

Расширяет **IChildOnMap**, **ICustomizable**, **IEventEmitter**.

Интерфейс слоя карты.

Конструктор | Поля | События | Методы

## Конструктор

```
ILayer()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **brightnesschange** | Событие изменения яркости слоя. |
| **copyrightschange** | Событие изменения доступной информации об авторских правах. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **zoomrangechange** | Событие изменения доступной информации о диапазоне уровней масштабирования. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getBrightness**() | Number | Необязательный метод. Возвращает яркость слоя. В зависимости от суммарной яркости слоев, добавленных на карту, выбирается цвет логотипа и копирайтов на карте. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getCopyrights**(coords, zoom) | **vow.Promise** | Необязательный метод. Запрашивает информацию об авторских правах в указанной точке с указанным коэффициентом масштабирования. Возвращает объект-promise, который будет подтверждён и передаст в качестве результата массив строк или DOM-элементов с информацией об авторских правах. |
| **getParent**() | **IControlParent** \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от **IChildOnMap**. |
| **getZoomRange**(point) | **vow.Promise** | Необязательный метод. Определяет доступный в указанной точке диапазон уровней масштабирования. Если данные есть, то возвращенный объект-promise будет подтвержден и передаст в качестве результата массив из двух чисел — минимальный и максимальный доступный в точке уровень масштабирования. Если данных нет, то promise отклоняется с указанием ошибки. |
| **setParent**(parent) | **IChildOnMap** | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от **IChildOnMap**. |

## Описание событий

### brightnesschange

Событие изменения яркости слоя.

### copyrightschange

Событие изменения доступной информации об авторских правах.

### zoomrangechange

Событие изменения доступной информации о диапазоне уровней масштабирования.

## Описание методов

### getBrightness

```
{Number} getBrightness()
```

Необязательный метод. Возвращает яркость слоя. В зависимости от суммарной яркости слоев, добавленных на карту, выбирается цвет логотипа и копирайтов на карте.

**Возвращает** значение яркости от 0 до 1 (0 — нулевая яркость, 1 — максимальная яркость).


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getCopyrights

```
{vow.Promise} getCopyrights(coords, zoom)
```

Необязательный метод. Запрашивает информацию об авторских правах в указанной точке с указанным коэффициентом масштабирования. Возвращает объект-promise, который будет подтверждён и передаст в качестве результата массив строк или DOM-элементов с информацией об авторских правах.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coords* | — | Тип: Number[]<br><br>Точка на карте, для которой запрашивается информация об авторских правах. |
| zoom* | — | Тип: Number<br><br>Уровень масштабирования, для которого запрашивается информация об авторских правах. |

* Обязательный параметр/опция.

## getZoomRange

```
{vow.Promise} getZoomRange(point)
```

Необязательный метод. Определяет доступный в указанной точке диапазон уровней масштабирования. Если данные есть, то возвращенный объект-promise будет подтвержден и передаст в качестве результата массив из двух чисел — минимальный и максимальный доступный в точке уровень масштабирования. Если данных нет, то promise отклоняется с указанием ошибки.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| point* | — | Тип: Number[]<br><br>Точка |

* Обязательный параметр/опция.

# ILayout

Расширяет **IDomEventEmitter**.

Интерфейс макета.

Конструктор | Поля | События | Методы

## Конструктор

```
ILayout(data)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data * | — | Тип: Object<br><br>Данные макета. |

* Обязательный параметр/опция.

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br><br>Унаследовано от IDomEventEmitter. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| emptinesschange | Изменение признака пустоты макета. Экземпляр класса Event. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| mousemove | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |
| mouseup | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchend | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от IDomEventEmitter. |
| multitouchmove | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от IDomEventEmitter. |
| parentelementchange | Изменение родительского элемента. Экземпляр класса Event. |
| shapechange | Изменение фигуры области охватывающей макет. Экземпляр класса Event. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Деструктор. Вызывается при завершении работы с макетом. |
| getData() | Object | Возвращает объект данных макета. |
| getParentElement() | HTMLElement | Возвращает родительский html-элемент. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getShape() | IShape \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета. |
| isEmpty() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие. |
| setData(data) | | Задает данные макета. |
| setParentElement(parent) | | Добавляет макет в DOM-дерево. |

## Описание событий

### emptinesschange

Изменение признака пустоты макета. Экземпляр класса **Event**.

### parentelementchange

Изменение родительского элемента. Экземпляр класса **Event**.

### shapechange

Изменение фигуры области охватывающей макет. Экземпляр класса **Event**.

## Описание методов

### destroy

```
destroy()
```

Деструктор. Вызывается при завершении работы с макетом.

### getData

```
{Object} getData()
```

**Возвращает** объект данных макета.

### getParentElement

```
{HTMLElement} getParentElement()
```

**Возвращает** родительский html-элемент.

### getShape

```
{IShape | null} getShape()
```

**Возвращает** фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### isEmpty

```
{Boolean} isEmpty()
```

**Возвращает** true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.

### setData

```
setData(data)
```

Задает данные макета.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data* | — | Тип: Object<br><br>Данные макета. |

* Обязательный параметр/опция.

### setParentElement

```
setParentElement(parent)
```

Добавляет макет в DOM-дерево.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parent* | — | Тип: HTMLElement \| null<br><br>Родительский html-элемент. Родительский элемент должен быть добавлен в DOM-дерево. Если передан null, то элемент удаляется из DOM-дерева. |

* Обязательный параметр/опция.

## ILinearRingGeometryAccess

Расширяет **IFreezable**.

Интерфейс доступа к геометрии "Замкнутый контур".

Конструктор | Поля | События | Методы

### Конструктор

```
ILinearRingGeometryAccess()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий объекта.<br><br>Унаследовано от **IFreezable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| change | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты;<br>• oldFillRule — старый алгоритм заливки;<br>• newFillRule — новые алгоритм заливки. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри контура. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **get**(index) | Number[] | Возвращает координаты точки с заданным индексом. |
| **getChildGeometry**(index) | **IPointGeometryAccess** | Создает и возвращает объект **IPointGeometryAccess** для заданной вершины контура. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре точку, ближайшую к anchorPosition. |
| **getCoordinates**() | Number[][] | Возвращает массив координат геометрии. |
| **getFillRule**() | String | Возвращает идентификатор алгоритма заливки. |
| **getLength**() | Integer | Возвращает количество точек в геометрии. |
| **insert**(index, coordinates) | **ILinearRingGeometryAccess** | Добавляет новую точку с заданным индексом. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | Number[] | Удаляет точку с заданным индексом. |
| **set**(index, coordinates) | **ILinearRingGeometryAccess** | Задает координаты точки с заданным индексом. |
| **setCoordinates**(coordinates) | **ILinearRingGeometryAccess** | Задает массив координат геометрии. |
| **setFillRule**(fillRule) | **ILinearRingGeometryAccess** | Задает алгоритм заливки контура. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **splice**(index, number) | Number[][] | Удаляет определенное число точек начиная с заданного индекса. При этом на место удаляемых точек могут быть добавлены новые. Координаты новых точек можно передавать дополнительными аргументами после параметра number. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание событий

### change

Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- oldCoordinates — старые координаты;
- newCoordinates — новые координаты;
- oldFillRule — старый алгоритм заливки;
- newFillRule — новые алгоритм заливки.

## Описание методов

### contains

```
{Boolean} contains(position)
```

Проверяет, лежит ли переданная точка внутри контура.

**Возвращает** признак принадлежности точки контуру.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position* | — | Тип: Number[]<br>Координаты точки. |

* Обязательный параметр/опция.

### get

```
{Number[]} get(index)
```

**Возвращает** координаты точки с заданным индексом.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br>Индекс точки. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getChildGeometry

```
{IPointGeometryAccess} getChildGeometry(index)
```

Создает и возвращает объект **IPointGeometryAccess** для заданной вершины контура.

**Возвращает** объект геометрии "Точка", соответствующий заданной вершине.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс вершины контура. |

* Обязательный параметр/опция.

## getClosest

```
{Object} getClosest(anchorPosition)
```

Ищет на контуре точку, ближайшую к anchorPosition.

**Возвращает** объект со следующими полями:

- position — точка на контуре ближайшая к anchorPosition;
- distance — расстояние от anchorPosition до position;
- closestPointIndex — индекс вершины, ближайшей к position;
- nextPointIndex — индекс вершины, следующей за position;
- prevPointIndex — индекс вершины, предшествующей position.

Поля nextPointIndex, prevPointIndex могут отсутствовать, если position совпадает с одной из вершин контура.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| anchorPosition* | — | Тип: Number[]<br><br>Координаты точки, для которой рассчитывается ближайшая точка на контуре. |

* Обязательный параметр/опция.

## getCoordinates

```
{Number[][]} getCoordinates()
```

**Возвращает** массив координат геометрии.

## getFillRule

```
{String} getFillRule()
```

**Возвращает** идентификатор алгоритма заливки.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getLength

```
{Integer} getLength()
```

**Возвращает** количество точек в геометрии.

## insert

```
{ILinearRingGeometryAccess} insert(index, coordinates)
```

Добавляет новую точку с заданным индексом.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс точки. |
| coordinates* | — | Тип: Number[]<br><br>Координаты точки. |

* Обязательный параметр/опция.

## remove

```
{Number[]} remove(index)
```

Удаляет точку с заданным индексом.

**Возвращает** координаты удаленной точки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс точки. |

* Обязательный параметр/опция.

## set

```
{ILinearRingGeometryAccess} set(index, coordinates)
```

Задает координаты точки с заданным индексом.

**Возвращает** ссылка на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс точки. |
| coordinates* | — | Тип: Number[]<br><br>Координаты точки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

## setCoordinates

```
{ILinearRingGeometryAccess} setCoordinates(coordinates)
```

Задает массив координат геометрии.

**Возвращает** ссылка на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates* | — | Тип: Number[][]<br><br>Координаты геометрии. |

* Обязательный параметр/опция.

## setFillRule

```
{ILinearRingGeometryAccess} setFillRule(fillRule)
```

Задает алгоритм заливки контура.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| fillRule* | — | Тип: String<br><br>Идентификатор алгоритма заливки. |

* Обязательный параметр/опция.

## splice

```
{Number[][]} splice(index, number)
```

Удаляет определенное число точек начиная с заданного индекса. При этом на место удаляемых точек могут быть добавлены новые. Координаты новых точек можно передавать дополнительными аргументами после параметра number.

**Возвращает** массив координат удаленных точек.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс, начиная с которого производится удаление и добавление точек. |
| number* | — | Тип: Integer<br><br>Количество удаляемых точек. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# ILineStringGeometry

Расширяет **IGeometry**, **ILineStringGeometryAccess**.

Интерфейс геометрии "Ломаная линия".

Конструктор | Поля | События | Методы

## Конструктор

```
ILineStringGeometry()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты.<br>Унаследовано от **ILineStringGeometryAccess**. |
| **mapchange** | Сменилась карта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IGeometry**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **pixelgeometrychange** | Изменилась пиксельная геометрия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия **IPixelGeometry**.<br>Унаследовано от **IGeometry**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **get**(index) | Number[] | Возвращает координаты точки с заданным индексом. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **ILineStringGeometryAccess**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.<br>Унаследован от **IGeometry**. |
| **getChildGeometry**(index) | **IPointGeometryAccess** | Создает и возвращает объект **IPointGeometryAccess** для заданной вершины ломаной линии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getClosest**(anchorPosition) | Object | Ищет на ломаной линии точку, ближайшую к anchorPosition.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getCoordinates**() | Number[][] | Возвращает массив координат геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getLength**() | Integer | Возвращает количество точек в геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **getMap**() | **Map** \| null | Возвращает текущую карту.<br>Унаследован от **IGeometry**. |
| **getPixelGeometry**([options]) | **IPixelGeometry** | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.<br>Унаследован от **IGeometry**. |
| **getType**() | String | Возвращает строку "LineString". |
| **insert**(index, coordinates) | **ILineStringGeometryAccess** | Добавляет новую точку с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | Number[] | Удаляет точку с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |
| **set**(index, coordinates) | **ILineStringGeometryAccess** | Задает координаты точки с заданным индексом.<br>Унаследован от **ILineStringGeometryAccess**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **setCoordinates**(coordinates) | **ILineStringGeometryAccess** | Задает массив координат геометрии.<br>Унаследован от **ILineStringGeometryAccess**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |
| **splice**(index, number) | Number[][] | Удаляет определенное число точек начиная с заданного индекса. При этом на место удаляемых точек могут быть добавлены новые. Координаты новых точек можно передавать дополнительными аргументами после параметра number.<br>Унаследован от **ILineStringGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание методов

### getType

```
{String} getType()
```

**Возвращает** строку "LineString".

# ILineStringGeometryAccess

Расширяет **IFreezable**.

Интерфейс доступа к геометрии "Ломаная линия".

Конструктор | Поля | События | Методы

## Конструктор

```
ILineStringGeometryAccess()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий объекта.<br>Унаследовано от **IFreezable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| freeze() | IFreezable | Переводит объект в "замороженный" режим.<br>Унаследован от IFreezable. |
| get(index) | Number[] | Возвращает координаты точки с заданным индексом. |
| getChildGeometry(index) | IPointGeometryAccess | Создает и возвращает объект IPointGeometryAccess для заданной вершины ломаной линии. |
| getClosest(anchorPosition) | Object | Ищет на ломаной линии точку, ближайшую к anchorPosition. |
| getCoordinates() | Number[][] | Возвращает массив координат геометрии. |
| getLength() | Integer | Возвращает количество точек в геометрии. |
| insert(index, coordinates) | ILineStringGeometryAccess | Добавляет новую точку с заданным индексом. |
| isFrozen() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от IFreezable. |
| remove(index) | Number[] | Удаляет точку с заданным индексом. |
| set(index, coordinates) | ILineStringGeometryAccess | Задает координаты точки с заданным индексом. |
| setCoordinates(coordinates) | ILineStringGeometryAccess | Задает массив координат геометрии. |
| splice(index, number) | Number[][] | Удаляет определенное число точек начиная с заданного индекса. При этом на место удаляемых точек могут быть добавлены новые. Координаты новых точек можно передавать дополнительными аргументами после параметра number. |
| unfreeze() | IFreezable | Переводит объект в активный режим.<br>Унаследован от IFreezable. |

## Описание событий

### change

Изменение координат. Экземпляр класса Event. Имена полей, доступных через метод Event.get:

- oldCoordinates — старые координаты;
- newCoordinates — новые координаты.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### get

```
{Number[]} get(index)
```

**Возвращает** координаты точки с заданным индексом.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс точки. |

* Обязательный параметр/опция.

**Пример:**

```
// Отмечаем начало линии меткой:
map.geoObjects.add(
    new ymaps.Placemark(polyline.geometry.get(0), { iconContent: 'A' })
);
```

### getChildGeometry

```
{IPointGeometryAccess} getChildGeometry(index)
```

Создает и возвращает объект **IPointGeometryAccess** для заданной вершины ломаной линии.

**Возвращает** объект геометрии "Точка", соответствующий заданной вершине.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс вершины. |

* Обязательный параметр/опция.

### getClosest

```
{Object} getClosest(anchorPosition)
```

Ищет на ломаной линии точку, ближайшую к anchorPosition.

**Возвращает** объект со следующими полями:

- position — точка на ломаной, ближайшая к anchorPosition;
- distance — расстояние от anchorPosition до position;
- closestPointIndex — индекс вершины, ближайшей к position;
- nextPointIndex — индекс вершины, следующей за position;
- prevPointIndex — индекс вершины, предшествующей position.

Поля nextPointIndex, prevPointIndex могут отсутствовать, если position совпадает с одной из вершин ломаной линии.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| anchorPosition* | — | Тип: Number[]<br><br>Координаты точки, для которой рассчитывается ближайшая точка на ломаной. |

* Обязательный параметр/опция.

**Пример:**

```
// Удаляем из линии точки по клику на ней:
myPolyline.events.add('click', function (e) {
    myPolyline.geometry.remove(
        myPolyline.geometry.getClosest(e.get('coords')).closestPointIndex
    );
});
```

## getCoordinates

```
{Number[][]} getCoordinates()
```

**Возвращает** массив координат геометрии.

## getLength

```
{Integer} getLength()
```

**Возвращает** количество точек в геометрии.

## insert

```
{ILineStringGeometryAccess} insert(index, coordinates)
```

Добавляет новую точку с заданным индексом.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс точки. |
| coordinates* | — | Тип: Number[]<br><br>Координаты точки. |

* Обязательный параметр/опция.

**Пример:**

```
// добавляем новую точку в конце линии по клику на карте
myMap.events.add('click', function (e) {
    myLineString.insert(myLineString.getLength(), e.get('coords'))
});
```

## remove

```
{Number[]} remove(index)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Удаляет точку с заданным индексом.

**Возвращает** координаты удаленной точки.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс точки. |

* Обязательный параметр/опция.

## set

```
{ILineStringGeometryAccess} set(index, coordinates)
```

Задает координаты точки с заданным индексом.

**Возвращает** ссылка на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс точки. |
| coordinates* | — | Тип: Number[]<br><br>Координаты точки. |

* Обязательный параметр/опция.

## setCoordinates

```
{ILineStringGeometryAccess} setCoordinates(coordinates)
```

Задает массив координат геометрии.

**Возвращает** ссылка на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates* | — | Тип: Number[][]<br><br>Координаты геометрии. |

* Обязательный параметр/опция.

## splice

```
{Number[][]} splice(index, number)
```

Удаляет определенное число точек начиная с заданного индекса. При этом на место удаляемых точек могут быть добавлены новые. Координаты новых точек можно передавать дополнительными аргументами после параметра number.

**Возвращает** массив координат удаленных точек.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс, начиная с которого производится удаление и добавление точек. |
| number* | — | Тип: Integer<br><br>Количество удаляемых точек. |

* Обязательный параметр/опция.

**Пример:**

```
// добавляем новую точку в начало линии по клику на карте
myMap.events.add('click', function (e) {
    myLineString.splice(0, 0, myLineString.getLength(), e.get('coords'))
});
```

# IMapAction

Расширяет **IEventEmitter**.

Интерфейс объекта, управляющего движением карты.

Конструктор | Поля | События | Методы

## Конструктор

```
IMapAction()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **end** | Событие, оповещающее карту об окончании движения. |
| **tick** | Событие, оповещающее карту о следующем шаге движения. Содержит поля:<br>• globalPixelCenter — новый центр карты в глобальных пикселах;<br>• zoom — новый коэффициент масштабирования карты;<br>• duration — время, которое отводится на выполнение шага;<br>• timingFunction — функция, описывающая вид движения. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Методы

| Имя | Описание |
|---|---|
| **begin**(mapActionManager) | Запускает движение на исполнение картой. Этот метод автоматически вызывается менеджером движений карты. С момента вызова IMapAction.begin менеджер движений слушает события IMapAction.tick и IMapAction.end и исполняет их. |
| **end**() | Останавливает движение. |

## Описание событий

### end

Событие, оповещающее карту об окончании движения.

### tick

Событие, оповещающее карту о следующем шаге движения. Содержит поля:

- globalPixelCenter — новый центр карты в глобальных пикселах;
- zoom — новый коэффициент масштабирования карты;
- duration — время, которое отводится на выполнение шага;
- timingFunction — функция, описывающая вид движения.

## Описание методов

### begin

```
begin(mapActionManager)
```

Запускает движение на исполнение картой. Этот метод автоматически вызывается менеджером движений карты. С момента вызова IMapAction.begin менеджер движений слушает события IMapAction.tick и IMapAction.end и исполняет их.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| mapActionManager* | — | Тип: **map.action.Manager**<br><br>Менеджер действий карты, на которой исполняется движение. |

* Обязательный параметр/опция.

### end

```
end()
```

Останавливает движение.

# IMapObjectCollection

Расширяет **ICollection**, **ICustomizable**, **IParentOnMap**.

Коллекция объектов на карте.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Конструктор | Поля | События | Методы

## Конструктор

```
IMapObjectCollection()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| add | Произошло добавление дочернего объекта.<br>Унаследовано от ICollection. |
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IParentOnMap. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| remove | Произошло удаление дочернего объекта.<br>Унаследовано от ICollection. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| add(object) | ICollection | Добавляет в коллекцию дочерний объект.<br>Унаследован от ICollection. |
| getIterator() | IIterator | Возвращает итератор по коллекции.<br>Унаследован от ICollection. |
| getMap() | Map | Возвращает ссылку на карту.<br>Унаследован от IParentOnMap. |
| remove(object) | ICollection | Удаляет из коллекции дочерний объект.<br>Унаследован от ICollection. |

# IMapState

Объект-описание состояния карты. Объекты с таким интерфейсом возвращаются сервисом загрузки YMapsML.



Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

См.: geoXml.load

Конструктор | Методы

## Конструктор

```
IMapState()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| applyToMap(map) | vow.Promise | Применяет состояние к переданной карте. |

### Описание методов

## applyToMap

```
{vow.Promise} applyToMap(map)
```

Применяет состояние к переданной карте.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: Map<br><br>Карта. |

* Обязательный параметр/опция.

# IMultiRouteModelJson

Интерфейс объекта-описания модели мультимаршрута.

Конструктор | Поля

## Конструктор

```
IMultiRouteModelJson()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| params | IMultiRouteParams | Параметры маршрутизации. |
| referencePoints | IMultiRouteReferencePoint[] | Описание опорных точек мультимаршрута. Также смотрите описание параметра IMultiRouteParams.viaIndexes. |

### Описание полей

## params

```
{IMultiRouteParams} params
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Параметры маршрутизации.

### referencePoints

```
{IMultiRouteReferencePoint[]} referencePoints
```

Описание опорных точек мультимаршрута. Также смотрите описание параметра **IMultiRouteParams.viaIndexes**.

## IMultiRouteParams

Интерфейс объекта, описывающего формат задания параметров модели мультимаршрута.

Конструктор | Поля

### Конструктор

```
IMultiRouteParams()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **avoidTrafficJams** | Boolean | Позволяет прокладывать мультимаршрут с учетом информации о текущих пробках на дорогах. Значение по умолчанию false. |
| **boundedBy** | Number[][] \| null | Позволяет задать область на карте, где предположительно находятся искомые объекты. Используется, если точки маршрута заданы почтовым адресом, а не координатами. Значение по умолчанию null. |
| **results** | Integer | Максимально допустимое число маршрутов, возвращаемых мультимаршрутизатором. Значение по умолчанию 3. |
| **routingMode** | String | Тип маршрутизации. Может принимать одно из двух строковых значений:<br>• "auto" — автомобильная маршрутизация;<br>• "masstransit" — маршрутизация с использованием общественного транспорта;<br>• "pedestrian" — пешеходная маршрутизация.<br>Значение по умолчанию "auto". |
| **searchCoordOrder** | String | Определяет способ трактовки описаний опорных точек, заданных в виде массивов координат или геометрий. Может принимать одно из двух значений: "longlat", либо "latlong". По умолчанию используется текущее значение параметра coordorder API. |
| **strictBounds** | Boolean | Искать объекты только внутри области, заданной параметром boundedBy. Используется, если точки маршрута заданы почтовым адресом, а не координатами. Значение по умолчанию false. |
| **viaIndexes** | Integer[] | Индексы транзитных точек во множестве опорных точек мультимаршрута. По умолчанию массив пуст. |

### Описание полей

### avoidTrafficJams

```
{Boolean} avoidTrafficJams
```

Позволяет прокладывать мультимаршрут с учетом информации о текущих пробках на дорогах. Значение по умолчанию false.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### boundedBy

```
{Number[][] | null} boundedBy
```

Позволяет задать область на карте, где предположительно находятся искомые объекты. Используется, если точки маршрута заданы почтовым адресом, а не координатами. Значение по умолчанию null.

### results

```
{Integer} results
```

Максимально допустимое число маршрутов, возвращаемых мультимаршрутизатором. Значение по умолчанию 3.

### routingMode

```
{String} routingMode
```

Тип маршрутизации. Может принимать одно из двух строковых значений:

- "auto" — автомобильная маршрутизация;
- "masstransit" — маршрутизация с использованием общественного транспорта;
- "pedestrian" — пешеходная маршрутизация.

Значение по умолчанию "auto".

### searchCoordOrder

```
{String} searchCoordOrder
```

Определяет способ трактовки описаний опорных точек, заданных в виде массивов координат или геометрий. Может принимать одно из двух значений: "longlat", либо "latlong". По умолчанию используется текущее значение параметра coordorder API.

### strictBounds

```
{Boolean} strictBounds
```

Искать объекты только внутри области, заданной параметром boundedBy. Используется, если точки маршрута заданы почтовым адресом, а не координатами. Значение по умолчанию false.

### viaIndexes

```
{Integer[]} viaIndexes
```

Индексы транзитных точек во множестве опорных точек мультимаршрута. По умолчанию массив пуст.

## IMultiRouteReferencePoint

Интерфейс объекта, описывающего формат задания опорной точки мультимаршрута.

Конструктор

### Конструктор

```
IMultiRouteReferencePoint()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Каждая опорная точка мультимаршрута может быть задана одним из следующих способов:

- строка, содержащая почтовый адрес опорной точки;
- массив, содержащий широту и долготу опорной точки;
- геометрия **geometry.Point**, описывающая опорную точку.

Опорные точки мультимаршрута могут подразделяться на путевые и транзитные. Путевые точки подразумевают остановку и делят маршрут на т.н. пути. Транзитные точки остановку не подразумевают. Таким образом, при прохождении через транзитную точку, сегмент пути маршрута не прерывается.

# IOptionManager

Расширяет **IChild**, **IEventEmitter**, **IFreezable**.

Интерфейс менеджера опций. Менеджер опций позволяет задавать значения опций, строить иерархию наследования опций, а также разрешать значения опций в контексте существующей иерархии наследования.

Конструктор | Поля | События | Методы

## Конструктор

```
IOptionManager()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий объекта.<br>Унаследовано от **IFreezable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение опций. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **get**(key[, defaultValue]) | | Возвращает значение заданной опции в контексте существующей иерархии наследования опций. При вызове данного метода сначала происходит поиск значения в текущем менеджере опций, а затем, если значение не определено, поиск продолжается в иерархии родительских менеджеров. |
| **getAll**() | Object | Возвращает ссылку на внутренний хэш хранящий значения опций. |
| **getName**() | String | Возвращает имя менеджера опций. |
| **getNative**(key) | Object | Возвращает значение заданной опции, определенное на данном уровне иерархии опций, т.е. в данном менеджере. |
| **getParent**() | **IOptionManager** \| null | Возвращает родительский менеджер опций. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **resolve**(key[, name]) | Object | Метод, предназначенный для вызова дочерними менеджерами опций. |
| **setName**(name) | | Задает имя менеджера опций. |
| **setParent**(parent) | **IChild** | Задает родительский менеджер опций. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание событий

### change

Изменение опций.

## Описание методов

### get

```
get(key[, defaultValue])
```

Возвращает значение заданной опции в контексте существующей иерархии наследования опций. При вызове данного метода сначала происходит поиск значения в текущем менеджере опций, а затем, если значение не определено, поиск продолжается в иерархии родительских менеджеров.

**Возвращает** значение заданной опции, либо defaultValue, если значение опции не определено.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Имя опции. |
| defaultValue | — | Тип: Object<br><br>Значение по умолчанию. |

* Обязательный параметр/опция.

## getAll

```
{Object} getAll()
```

**Возвращает** ссылку на внутренний хэш хранящий значения опций.

## getName

```
{String} getName()
```

**Возвращает** имя менеджера опций.

## getNative

```
{Object} getNative(key)
```

**Возвращает** значение заданной опции, определенное на данном уровне иерархии опций, т.е. в данном менеджере.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Имя опции. |

* Обязательный параметр/опция.

## getParent

```
{IOptionManager | null} getParent()
```

**Возвращает** родительский менеджер опций.

## resolve

```
{Object} resolve(key[, name])
```

Метод, предназначенный для вызова дочерними менеджерами опций.

**Возвращает** значение опции в родительском контексте.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| key* | — | Тип: String<br><br>Имя опции. |
| name | — | Тип: String<br><br>Имя дочернего менеджера опций. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

* Обязательный параметр/опция.

### setName

```
setName(name)
```

Задает имя менеджера опций.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| name* | — | Тип: String<br><br>Имя менеджера опций. |

* Обязательный параметр/опция.

### setParent

```
{IChild} setParent(parent)
```

Задает родительский менеджер опций.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| parent* | — | Тип: **IOptionManager** \| null<br><br>Родительский менеджер опций. |

* Обязательный параметр/опция.

## IOverlay

Расширяет **ICustomizable**, **IDomEventEmitter**.

Интерфейс оверлея.

Конструктор | Поля | События | Методы

### Конструктор

```
IOverlay()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IDomEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **datachange** | Изменение данных. Поля данных:<br>• oldData — старые данные;<br>• newData — новые данные. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **emptinesschange** | Изменение признака пустоты оверлея. Экземпляр класса **Event**. |
| **geometrychange** | Изменение геометрии. Поля данных:<br>• oldGeometry — старая пиксельная геометрия;<br>• newGeometry — новая пиксельная геометрия. |
| **mapchange** | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| multitouchend | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от IDomEventEmitter. |
| multitouchmove | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от IDomEventEmitter. |
| optionschange | Изменение в опциях объекта.<br><br>Унаследовано от ICustomizable. |
| shapechange | Изменение фигуры области охватывающей оверлей. Экземпляр класса Event. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br><br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
| --- | --- | --- |
| getData() | Object | Возвращает объект данных оверлея. |
| getGeometry() | IPixelGeometry | Возвращает текущую пиксельную геометрию. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map \| null | Возвращает ссылку на текущую карту. |
| getShape() | IShape \| null | Возвращает фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру. |
| isEmpty() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие. |
| setData(data) | | Задает данные оверлея. |
| setGeometry(geometry) | | Задает пиксельную геометрию оверлея. |
| setMap(map) | | Задает карту, на которой должен отобразиться оверлей. |

## Описание событий

### datachange

Изменение данных. Поля данных:

- oldData — старые данные;
- newData — новые данные.

### emptinesschange

Изменение признака пустоты оверлея. Экземпляр класса **Event**.

### geometrychange

Изменение геометрии. Поля данных:

- oldGeometry — старая пиксельная геометрия;
- newGeometry — новая пиксельная геометрия.

### mapchange

Сменилась карта. Поля данных:

- oldMap — старая карта;
- newMap — новая карта.

### shapechange

Изменение фигуры области охватывающей оверлей. Экземпляр класса **Event**.

## Описание методов

### getData

```
{Object} getData()
```

**Возвращает** объект данных оверлея.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getGeometry

```
{IPixelGeometry} getGeometry()
```

**Возвращает** текущую пиксельную геометрию.

## getMap

```
{Map | null} getMap()
```

**Возвращает** ссылку на текущую карту.

## getShape

```
{IShape | null} getShape()
```

**Возвращает** фигуру, описывающую форму области, охватывающей оверлей в глобальных пиксельных координатах, либо null, если нет возможности рассчитать фигуру.

## isEmpty

```
{Boolean} isEmpty()
```

**Возвращает** true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.

## setData

```
setData(data)
```

Задает данные оверлея.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data* | — | Тип: Object<br><br>Данные оверлея. |

* Обязательный параметр/опция.

## setGeometry

```
setGeometry(geometry)
```

Задает пиксельную геометрию оверлея.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry* | — | Тип: IPixelGeometry<br><br>Геометрия в глобальных пиксельных координатах. |

* Обязательный параметр/опция.

## setMap

```
setMap(map)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Задает карту, на которой должен отобразиться оверлей.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| map* | — | Тип: **Map** \| null<br><br>Ссылка на карту. |

* Обязательный параметр/опция.

# IPane

Расширяет **IEventEmitter**.

Интерфейс пейна карты. Пейн карты представляет собой объект, размещающий внутри контейнера карты свой DOM-элемент с определенным zIndex. Пейн может служить контейнером для размещения отображений различных элементов карты, таких как тайлы, оверлеи, статические элементы управления картой и т.д. Также пейн может выполнять роль непрозрачного для DOM-событий экрана, на котором могут слушаться события мыши.

Конструктор | Поля | События | Методы

## Конструктор

```
IPane()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

## События

| Имя | Описание |
|---|---|
| **overflowchange** | Изменение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Экземпляр **IEvent**. |
| **zindexchange** | Изменение значения zIndex пейна. Экземпляр **IEvent**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Уничтожает пейн. |
| **getElement**() | HTMLElement | Возвращает ссылку на DOM-контейнер пейна. |
| **getMap**() | **Map** | Возвращает карту, которой принадлежит пейн. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getOverflow() | String | Возвращает значение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Данный параметр может принимать одно из следующих строковых значений:<br>• "visible" — при выходе за пределы контейнера карты содержимое пейна остается видимым;<br>• "hidden" — область отображения содержимого пейна ограничена контейнером карты.<br>. |
| getZIndex() | Number | Возвращает zIndex пейна. |

## Описание событий

### overflowchange

Изменение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Экземпляр **IEvent**.

### zindexchange

Изменение значения zIndex пейна. Экземпляр **IEvent**.

## Описание методов

### destroy

```
destroy()
```

Уничтожает пейн.

### getElement

```
{HTMLElement} getElement()
```

**Возвращает** ссылку на DOM-контейнер пейна.

### getMap

```
{Map} getMap()
```

**Возвращает** карту, которой принадлежит пейн.

### getOverflow

```
{String} getOverflow()
```

**Возвращает** значение параметра "overflow", определяющего видимость содержимого пейна при выходе за пределы контейнера карты. Данный параметр может принимать одно из следующих строковых значений:

- "visible" — при выходе за пределы контейнера карты содержимое пейна остается видимым;
- "hidden" — область отображения содержимого пейна ограничена контейнером карты.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

.

### getZIndex

```
{Number} getZIndex()
```

**Возвращает** zIndex пейна.

## IPanorama

Интерфейс для описания панорамы.

Конструктор | Методы

### Конструктор

```
IPanorama()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getAngularBBox**() | Number[] | Возвращает сферические координаты, которые задают на панорамной сфере область, занимаемую изображением. Координаты указываются в формате `[thetaTop, phiRight, thetaBottom, phiLeft]` (аналогично CSS). |
| **getConnections**() | **IPanoramaConnection**[] | Возвращает массив связей панорамы. |
| **getCoordSystem**() | **ICoordSystem** | Возвращает систему координат, в которой заданы позиции панорамы и всех связанных с ней маркеров и связей. |
| **getDefaultDirection**() | Number[] | Возвращает направление взгляда по умолчанию. Будет использовано плеером при открытии панорамы. |
| **getDefaultSpan**() | Number[] | Возвращает размер области обзора по умолчанию. Будет использовано плеером при открытии панорамы. |
| **getGraph**() | **IPanoramaGraph** \| null | Возвращает граф панорам, связанных с текущей панорамой, для осуществления быстрых переходов. |
| **getMarkers**() | **IPanoramaMarker**[] | Возвращает массив маркеров в панораме. |
| **getName**() | String | Возвращает название панорамы, отображаемое плеером в интерфейсе. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getPosition() | Number[] | Возвращает положение панорамы в указанной в опциях системе координат. Задается в формате `[lon, lat, height]`, `[lat, lon, height]` или `[x, y, height]` в зависимости от системы и порядка координат. `height` – высота панорамы, заданная относительно некоторого уровня (не обязательно уровня моря). |
| getThoroughfares() | IPanoramaThoroughfare[] | Возвращает массив переходов на связанные панорамы. |
| getTileLevels() | IPanoramaTileLevel[] | Возвращает массив уровней масштабирования панорамного изображения. |
| getTileSize() | Number[] | Возвращает размер тайлов, на которые нарезано панорамное изображение. |

## Описание методов

### getAngularBBox

```
{Number[]} getAngularBBox()
```

**Возвращает** сферические координаты, которые задают на панорамной сфере область, занимаемую изображением. Координаты указываются в формате `[thetaTop, phiRight, thetaBottom, phiLeft]` (аналогично CSS).

### getConnections

```
{IPanoramaConnection[]} getConnections()
```

**Возвращает** массив связей панорамы.

### getCoordSystem

```
{ICoordSystem} getCoordSystem()
```

**Возвращает** систему координат, в которой заданы позиции панорамы и всех связанных с ней маркеров и связей.

### getDefaultDirection

```
{Number[]} getDefaultDirection()
```

**Возвращает** направление взгляда по умолчанию. Будет использовано плеером при открытии панорамы.

### getDefaultSpan

```
{Number[]} getDefaultSpan()
```

**Возвращает** размер области обзора по умолчанию. Будет использовано плеером при открытии панорамы.

### getGraph

```
{IPanoramaGraph | null} getGraph()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** граф панорам, связанных с текущей панорамой, для осуществления быстрых переходов.

### getMarkers

```
{IPanoramaMarker[]} getMarkers()
```

**Возвращает** массив маркеров в панораме.

### getName

```
{String} getName()
```

**Возвращает** название панорамы, отображаемое плеером в интерфейсе.

### getPosition

```
{Number[]} getPosition()
```

**Возвращает** положение панорамы в указанной в опциях системе координат. Задается в формате `[lon, lat, height]`, `[lat, lon, height]` или `[x, y, height]` в зависимости от системы и порядка координат. `height` – высота панорамы, заданная относительно некоторого уровня (не обязательно уровня моря).

### getThoroughfares

```
{IPanoramaThoroughfare[]} getThoroughfares()
```

**Возвращает** массив переходов на связанные панорамы.

### getTileLevels

```
{IPanoramaTileLevel[]} getTileLevels()
```

**Возвращает** массив уровней масштабирования панорамного изображения.

### getTileSize

```
{Number[]} getTileSize()
```

**Возвращает** размер тайлов, на которые нарезано панорамное изображение.

## IPanoramaConnection

Расширяет **IPanoramaMarker**, **IPanoramaTransition**.

Интерфейс, описывающий связи панорамы.

Конструктор | Поля | Методы

### Конструктор

```
IPanoramaConnection()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **properties** | **data.Manager** | Дополнительные свойства маркера.<br>Унаследовано от **IPanoramaMarker**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getConnectedPanorama() | vow.Promise | Возвращает объект-обещание, который будет разрешен объектом связанной панорамы или отклонен с ошибкой.<br><br>Унаследован от IPanoramaTransition. |
| getIconSet() | vow.Promise | Возвращает объект-обещание, который будет разрешен объектом, реализующим интерфейс IPanoramaMarkerIconSet и содержащим изображения всех состояний маркера.<br><br>Унаследован от IPanoramaMarker. |
| getPanorama() | IPanorama | Возвращает панораму, к которой принадлежит маркер.<br><br>Унаследован от IPanoramaMarker. |
| getPosition() | Number[] | Возвращает положение маркера в системе координат панорамы, которой принадлежит содержащий узел граф. Задается в формате `[lon, lat, height]`, `[lat, lon, height]` или `[x, y, height]` в зависимости от системы и порядка координат. `height` – высота маркера в метрах, заданная относительно того же уровня, что и высота панорамы.<br><br>Унаследован от IPanoramaMarker. |

## IPanoramaGraph

Интерфейс, описывающий граф панорам.

Конструктор | Методы

### Конструктор

```
IPanoramaGraph()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getEdges() | IPanoramaGraphEdge[] | Возвращает массив ребер графа. |
| getNodes() | IPanoramaGraphNode[] | Возвращает массив узлов графа. |
| getPanorama() | IPanorama | Возвращает панораму, которой принадлежит граф. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### getEdges

```
{IPanoramaGraphEdge[]} getEdges()
```

**Возвращает** массив ребер графа.

### getNodes

```
{IPanoramaGraphNode[]} getNodes()
```

**Возвращает** массив узлов графа.

### getPanorama

```
{IPanorama} getPanorama()
```

**Возвращает** панораму, которой принадлежит граф.

# IPanoramaGraphEdge

Интерфейс, описывающий ребро графа панорам.

Конструктор | Методы

## Конструктор

```
IPanoramaGraphEdge()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getEndNodes() | IPanoramaGraphNode[] | Возвращает массив из двух узлов графа, связанных ребром. |

## Описание методов

### getEndNodes

```
{IPanoramaGraphNode[]} getEndNodes()
```

**Возвращает** массив из двух узлов графа, связанных ребром.

# IPanoramaGraphNode

Расширяет **IPanoramaTransition**.

Интерфейс, описывающий узел графа панорам.

Конструктор | Методы

## Конструктор

```
IPanoramaGraphNode()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getConnectedPanorama() | vow.Promise | Возвращает объект-обещание, который будет разрешен объектом связанной панорамы или отклонен с ошибкой.<br><br>Унаследован от IPanoramaTransition. |

## IPanoramaMarker

Интерфейс, описывающий маркеры на панораме.

Конструктор | Поля | Методы

### Конструктор

```
IPanoramaMarker()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| properties | data.Manager | Дополнительные свойства маркера. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getIconSet() | vow.Promise | Возвращает объект-обещание, который будет разрешен объектом, реализующим интерфейс IPanoramaMarkerIconSet и содержащим изображения всех состояний маркера. |
| getPanorama() | IPanorama | Возвращает панораму, к которой принадлежит маркер. |
| getPosition() | Number[] | Возвращает положение маркера в системе координат панорамы, которой принадлежит содержащий узел граф. Задается в формате `[lon, lat, height]`, `[lat, lon, height]` или `[x, y, height]` в зависимости от системы и порядка координат. `height` – высота маркера в метрах, заданная относительно того же уровня, что и высота панорамы. |

## Описание полей

### properties

```
{data.Manager} properties
```

Дополнительные свойства маркера.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### getIconSet

```
{vow.Promise} getIconSet()
```

**Возвращает** объект-обещание, который будет разрешен объектом, реализующим интерфейс **IPanoramaMarkerIconSet** и содержащим изображения всех состояний маркера.

### getPanorama

```
{IPanorama} getPanorama()
```

**Возвращает** панораму, к которой принадлежит маркер.

### getPosition

```
{Number[]} getPosition()
```

**Возвращает** положение маркера в системе координат панорамы, которой принадлежит содержащий узел граф. Задается в формате `[lon, lat, height]`, `[lat, lon, height]` или `[x, y, height]` в зависимости от системы и порядка координат. `height` – высота маркера в метрах, заданная относительно того же уровня, что и высота панорамы.

# IPanoramaMarkerIcon

Интерфейс, описывающий иконку маркера.

Конструктор | Поля

## Конструктор

```
IPanoramaMarkerIcon()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **image** | HTMLCanvasElement \| HTMLImageElement | Изображение иконки. |
| **offset** | Number[] | Смещение верхнего левого угла иконки в пикселах относительно проекции точки, к которой привязан маркер. |

## Описание полей

### image

```
{HTMLCanvasElement | HTMLImageElement} image
```

Изображение иконки.

### offset

```
{Number[]} offset
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Смещение верхнего левого угла иконки в пикселах относительно проекции точки, к которой привязан маркер.

# IPanoramaMarkerIconSet

Интерфейс, описывающий набор иконок маркера.

Конструктор | Поля

## Конструктор

```
IPanoramaMarkerIconSet()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **default** | **IPanoramaMarkerIcon** \| null | Иконка состояния по умолчанию. |
| **expanded** | **IPanoramaMarkerIcon** \| null | Иконка для "активного" состояния маркера. Маркер переходит в это состояние при клике по нему; при повторном клике маркер становится неактивным. |
| **expandedHovered** | **IPanoramaMarkerIcon** \| null | Иконка для "активного подсвеченного" состояния. Маркер переходит в это состояние, когда он активен и над ним находится курсор. |
| **hovered** | **IPanoramaMarkerIcon** \| null | Иконка для "подсвеченного" состояния маркера. В него маркер переходит, когда на него наводится курсор мыши. |

## Описание полей

### default

```
{IPanoramaMarkerIcon | null} default
```

Иконка состояния по умолчанию.

### expanded

```
{IPanoramaMarkerIcon | null} expanded
```

Иконка для "активного" состояния маркера. Маркер переходит в это состояние при клике по нему; при повторном клике маркер становится неактивным.

### expandedHovered

```
{IPanoramaMarkerIcon | null} expandedHovered
```

Иконка для "активного подсвеченного" состояния. Маркер переходит в это состояние, когда он активен и над ним находится курсор.

### hovered

```
{IPanoramaMarkerIcon | null} hovered
```

Иконка для "подсвеченного" состояния маркера. В него маркер переходит, когда на него наводится курсор мыши.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IPanoramaThoroughfare

Расширяет **IPanoramaTransition**.

Интерфейс, описывающий переход на панораме.

Конструктор | Поля | Методы

## Конструктор

```
IPanoramaThoroughfare()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **properties** | **data.Manager** | Дополнительные свойства перехода. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getConnectedPanorama**() | **vow.Promise** | Возвращает объект-обещание, который будет разрешен объектом связанной панорамы или отклонен с ошибкой.<br>Унаследован от **IPanoramaTransition**. |
| **getDirection**() | Number[] | Возвращает направление на панораму, на которую указывает переход из панорамы, которой он принадлежит. |
| **getPanorama**() | **IPanorama** | Возвращает панораму, которой принадлежит переход. |

## Описание полей

### properties

```
{data.Manager} properties
```

Дополнительные свойства перехода.

## Описание методов

### getDirection

```
{Number[]} getDirection()
```

**Возвращает** направление на панораму, на которую указывает переход из панорамы, которой он принадлежит.

### getPanorama

```
{IPanorama} getPanorama()
```

**Возвращает** панораму, которой принадлежит переход.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IPanoramaTileLevel

Интерфейс для описания уровней масштабирования панорамного изображения.

Конструктор | Методы

## Конструктор

```
IPanoramaTileLevel()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getImageSize() | Number[] | Возвращает размер всего изображения в пикселах. |
| getTileUrl(x, y) | String | Вычисляет URL тайла по его положению в панорамном изображении. Положение задается парой индексов: горизонтальным и вертикальным. Горизонтальный индекс отсчитывается слева с нуля и растет направо. Вертикальный отсчитывается с верха изображения и растет вниз. |

## Описание методов

### getImageSize

```
{Number[]} getImageSize()
```

**Возвращает** размер всего изображения в пикселах.

### getTileUrl

```
{String} getTileUrl(x, y)
```

Вычисляет URL тайла по его положению в панорамном изображении. Положение задается парой индексов: горизонтальным и вертикальным. Горизонтальный индекс отсчитывается слева с нуля и растет направо. Вертикальный отсчитывается с верха изображения и растет вниз.

**Возвращает** URL тайла с переданными индексами.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| x* | — | Тип: Number<br><br>Горизонтальный индекс тайла. |
| y* | — | Тип: Number<br><br>Вертикальный индекс тайла. |

* Обязательный параметр/опция.

# IPanoramaTransition

Интерфейс, описывающий сущность на панораме, связывающую ее с другой панорамой.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Конструктор | Методы

### Конструктор

```
IPanoramaTransition()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getConnectedPanorama() | vow.Promise | Возвращает объект-обещание, который будет разрешен объектом связанной панорамы или отклонен с ошибкой. |

## Описание методов

### getConnectedPanorama

```
{vow.Promise} getConnectedPanorama()
```

**Возвращает** объект-обещание, который будет разрешен объектом связанной панорамы или отклонен с ошибкой.

# IParentOnMap

Интерфейс родительского объекта, относящегося к определенному объекту карты.

Конструктор | События | Методы

### Конструктор

```
IParentOnMap()
```

### События

| Имя | Описание |
|---|---|
| mapchange | Сменилась карта. Поля данных:<br>• oldMap — старая карта;<br>• newMap — новая карта. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map | Возвращает ссылку на карту. |

## Описание событий

### mapchange

Сменилась карта. Поля данных:

- oldMap — старая карта;
- newMap — новая карта.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### getMap

```
{Map} getMap()
```

**Возвращает** ссылку на карту.

# IPixelCircleGeometry

Расширяет **IPixelGeometry**.

Интерфейс пиксельной геометрии "Круг".

Конструктор | Поля | Методы

## Конструктор

```
IPixelCircleGeometry()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getCoordinates**() | Number[] | Возвращает координаты центра круга. |
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br>Унаследован от **IPixelGeometry**. |
| **getRadius**() | Number | Возвращает радиус круга. |
| **getType**() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от **IBaseGeometry**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **scale**(factor) | **IPixelGeometry** | Создает отмасштабированную копию геометрии.<br>Унаследован от **IPixelGeometry**. |
| **shift**(offset) | **IPixelGeometry** | Создает копию геометрии сдвинутую на заданную величину.<br>Унаследован от **IPixelGeometry**. |

## Описание методов

### getCoordinates

```
{Number[]} getCoordinates()
```

**Возвращает** координаты центра круга.

### getRadius

```
{Number} getRadius()
```

**Возвращает** радиус круга.

# IPixelGeometry

Расширяет **IBaseGeometry**.

Интерфейс пиксельной геометрии.

Конструктор | Поля | Методы

## Конструктор

```
IPixelGeometry()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getMetaData() | Object | Возвращает метаданные пиксельной геометрии. |
| getType() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от IBaseGeometry. |
| scale(factor) | IPixelGeometry | Создает отмасштабированную копию геометрии. |
| shift(offset) | IPixelGeometry | Создает копию геометрии сдвинутую на заданную величину. |

## Описание методов

### equals

```
{Boolean} equals(geometry)
```

**Возвращает** true, если переданная геометрия эквивалентна данной.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| geometry* | — | Тип: IPixelGeometry<br>Геометрия, с которой производится сравнение. |

* Обязательный параметр/опция.

### getMetaData

```
{Object} getMetaData()
```

**Возвращает** метаданные пиксельной геометрии.

### scale

```
{IPixelGeometry} scale(factor)
```

Создает отмасштабированную копию геометрии.

**Возвращает** отмасштабированную копию геометрии.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| factor* | — | Тип: Number<br>Масштабный коэффициент. |

* Обязательный параметр/опция.

**Пример:**

```
// Уменьшаем геометрию в два раза
var smallCopy = myPixelGeometry.scale(0.5);
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### shift

```
{IPixelGeometry} shift(offset)
```

Создает копию геометрии сдвинутую на заданную величину.

**Возвращает** сдвинутую копию геометрии.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| offset* | — | Тип: Number[]<br><br>Величина сдвига по осям. |

* Обязательный параметр/опция.

**Пример:**

```
// Сдвигаем все координаты геометрии на 200 пикселей влево
var shifted = myPixelGeometry.shift([-200, 0]);
```

## IPixelLineStringGeometry

Расширяет **IPixelGeometry**.

Интерфейс пиксельной геометрии "Ломаная линия".

Конструктор | Поля | Методы

### Конструктор

```
IPixelLineStringGeometry()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br><br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br><br>Унаследован от **IBaseGeometry**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getClosest(anchorPosition) | Object | Ищет на ломаной линии точку, ближайшую к anchorPosition. |
| getCoordinates() | Number[][] | Возвращает координаты линии. |
| getLength() | Integer | Возвращает количество точек в геометрии. |
| getMetaData() | Object | Возвращает метаданные пиксельной геометрии.<br>Унаследован от IPixelGeometry. |
| getType() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от IBaseGeometry. |
| scale(factor) | IPixelGeometry | Создает отмасштабированную копию геометрии.<br>Унаследован от IPixelGeometry. |
| shift(offset) | IPixelGeometry | Создает копию геометрии сдвинутую на заданную величину.<br>Унаследован от IPixelGeometry. |

## Описание методов

### getClosest

```
{Object} getClosest(anchorPosition)
```

Ищет на ломаной линии точку, ближайшую к anchorPosition.

**Возвращает** объект со следующими полями:

- position — точка на ломаной ближайшая к anchorPosition;
- distance — расстояние от anchorPosition до position;
- closestPointIndex — индекс вершины, ближайшей к position;
- nextPointIndex — индекс вершины, следующей за position;
- prevPointIndex — индекс вершины, предшествующей position.

Поля nextPointIndex, prevPointIndex могут отсутствовать, если position совпадает с одной из вершин ломаной линии.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| anchorPosition* | — | Тип: Number[]<br><br>Координаты точки, для которой расчитывается ближайшая точка на ломаной. |

* Обязательный параметр/опция.

### getCoordinates

```
{Number[][]} getCoordinates()
```

**Возвращает** координаты линии.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### getLength

```
{Integer} getLength()
```

**Возвращает** количество точек в геометрии.

## IPixelMultiPolygonGeometry

Расширяет **IPixelGeometry**.

Интерфейс пиксельной геометрии "Мультимногоугольник".

Конструктор | Поля | Методы

### Конструктор

```
IPixelMultiPolygonGeometry()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет лежит ли переданная точка внутри мультимногоугольника. |
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре мультимногоугольника точку, ближайшую к anchorPosition. |
| **getCoordinates**() | Number[][][][] | Возвращает координаты мультимногоугольника. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getFillRule**() | String | Возвращает строковый идентификатор, определяющий алгоритм заливки мультимногоугольника. Идентификатор может иметь одно из двух значений:<br>• evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.<br>• nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри.<br>. |
| **getLength**() | Integer | Возвращает количество многоугольников в мультимногоугольнике. |
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br>Унаследован от **IPixelGeometry**. |
| **getType**() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от **IBaseGeometry**. |
| **scale**(factor) | **IPixelGeometry** | Создает отмасштабированную копию геометрии.<br>Унаследован от **IPixelGeometry**. |
| **shift**(offset) | **IPixelGeometry** | Создает копию геометрии сдвинутую на заданную величину.<br>Унаследован от **IPixelGeometry**. |

## Описание методов

### contains

```
{Boolean} contains(position)
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Проверяет лежит ли переданная точка внутри мультимногоугольника.

**Возвращает** признак принадлежности точки мультимногоугольнику.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position* | — | Тип: Number[]<br><br>Координаты точки. |

* Обязательный параметр/опция.

## getClosest

```
{Object} getClosest(anchorPosition)
```

Ищет на контуре мультимногоугольника точку, ближайшую к anchorPosition.

**Возвращает** объект со следующими полями:

- position — точка на контуре мультимногоугольника ближайшая к anchorPosition;
- distance — расстояние от anchorPosition до position;
- closestPointIndex — индекс вершины мультимногоугольника, ближайшей к position;
- nextPointIndex — индекс вершины мультимногоугольника, следующей за position;
- prevPointIndex — индекс вершины мультимногоугольника, предшествующей position;
- pathIndex — индекс контура мультимногоугольника, к которому относится найденная точка.

Поля nextPointIndex, prevPointIndex могут отсутствовать, если position совпадает с одной из вершин мультимногоугольника.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| anchorPosition* | — | Тип: Number[]<br><br>Координаты точки, для которой рассчитывается ближайшая точка на контуре многоугольника. |

* Обязательный параметр/опция.

## getCoordinates

```
{Number[][][][]} getCoordinates()
```

**Возвращает** координаты мультимногоугольника.

## getFillRule

```
{String} getFillRule()
```

**Возвращает** строковый идентификатор, определяющий алгоритм заливки мультимногоугольника. Идентификатор может иметь одно из двух значений:

- evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.

- nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри.

.

### getLength

```
{Integer} getLength()
```

**Возвращает** количество многоугольников в мультимногоугольнике.

## IPixelPointGeometry

Расширяет **IPixelGeometry**.

Интерфейс пиксельной геометрии "Точка".

Конструктор | Поля | Методы

### Конструктор

```
IPixelPointGeometry()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |
| **getCoordinates**() | Number[] | Возвращает координаты точки. |
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br>Унаследован от **IPixelGeometry**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getType() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от IBaseGeometry. |
| scale(factor) | IPixelGeometry | Создает отмасштабированную копию геометрии.<br>Унаследован от IPixelGeometry. |
| shift(offset) | IPixelGeometry | Создает копию геометрии сдвинутую на заданную величину.<br>Унаследован от IPixelGeometry. |

### Описание методов

#### getCoordinates

```
{Number[]} getCoordinates()
```

**Возвращает** координаты точки.

## IPixelPolygonGeometry

Расширяет IPixelGeometry.

Интерфейс пиксельной геометрии "Многоугольник".

Конструктор | Поля | Методы

### Конструктор

```
IPixelPolygonGeometry()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| contains(position) | Boolean | Проверяет лежит ли переданная точка внутри многоугольника. |
| equals(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br>Унаследован от IPixelGeometry. |
| getBounds() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IBaseGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре многоугольника точку, ближайшую к anchorPosition. |
| **getCoordinates**() | Number[][][] | Возвращает координаты многоугольника. |
| **getFillRule**() | String | Возвращает строковый идентификатор, определяющий алгоритм заливки многоугольника. Идентификатор может принимать одно из двух значений:<br>• evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.<br>• nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри.<br>. |
| **getLength**() | Integer | Возвращает количество контуров в многоугольнике. |
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br>Унаследован от **IPixelGeometry**. |
| **getType**() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от **IBaseGeometry**. |
| **scale**(factor) | **IPixelGeometry** | Создает отмасштабированную копию геометрии.<br>Унаследован от **IPixelGeometry**. |
| **shift**(offset) | **IPixelGeometry** | Создает копию геометрии сдвинутую на заданную величину.<br>Унаследован от **IPixelGeometry**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание методов

### contains

```
{Boolean} contains(position)
```

Проверяет лежит ли переданная точка внутри многоугольника.

**Возвращает** признак принадлежности точки многоугольнику.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position* | — | Тип: Number[]<br><br>Координаты точки. |

* Обязательный параметр/опция.

### getClosest

```
{Object} getClosest(anchorPosition)
```

Ищет на контуре многоугольника точку, ближайшую к anchorPosition.

**Возвращает** объект со следующими полями:

- position — точка на контуре многоугольника ближайшая к anchorPosition;
- distance — расстояние от anchorPosition до position;
- closestPointIndex — индекс вершины многоугольника, ближайшей к position;
- nextPointIndex — индекс вершины многоугольника, следующей за position;
- prevPointIndex — индекс вершины многоугольника, предшествующей position;
- pathIndex — индекс контура многоугольника, к которому относится найденная точка.

Поля nextPointIndex, prevPointIndex могут отсутствовать, если position совпадает с одной из вершин многоугольника.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| anchorPosition* | — | Тип: Number[]<br><br>Координаты точки, для которой рассчитывается ближайшая точка на контуре многоугольника. |

* Обязательный параметр/опция.

### getCoordinates

```
{Number[][][]} getCoordinates()
```

**Возвращает** координаты многоугольника.

### getFillRule

```
{String} getFillRule()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** строковый идентификатор, определяющий алгоритм заливки многоугольника. Идентификатор может принимать одно из двух значений:

- evenOdd — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и подсчета количества сегментов контура в пределах заданной фигуры, которые пересекает этот луч. Если это число нечетное, точка находится внутри; если четное, точка находится снаружи.
- nonZero — алгоритм, определяющий, находится ли точка в области заполнения, путем рисования луча от этой точки до бесконечности в любом направлении и проверки точек, в которых сегмент фигуры пересекает этот луч. Начиная с нуля, добавляется единица каждый раз, когда сегмент пересекает луч слева направо, и вычитается единица каждый раз, когда сегмент пересекает луч справа налево. Если после подсчета пересечений результат равен нулю, точка находится снаружи контура. В противном случае она находится внутри.

.

### getLength

```
{Integer} getLength()
```

**Возвращает** количество контуров в многоугольнике.

## IPixelRectangleGeometry

Расширяет **IPixelGeometry**.

Интерфейс пиксельной геометрии "Прямоугольник".

Конструктор | Поля | Методы

### Конструктор

```
IPixelRectangleGeometry()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **equals**(geometry) | Boolean | Возвращает true, если переданная геометрия эквивалентна данной.<br>Унаследован от **IPixelGeometry**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.<br>Унаследован от **IBaseGeometry**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getClosest**(anchorPosition) | Object | Ищет на прямоугольнике точку, ближайшую к anchorPosition. |
| **getCoordinates**() | Number[][] | Возвращает координаты двух противоположных углов прямоугольника. |
| **getMetaData**() | Object | Возвращает метаданные пиксельной геометрии.<br>Унаследован от **IPixelGeometry**. |
| **getType**() | String | Возвращает идентификатор типа геометрии.<br>Унаследован от **IBaseGeometry**. |
| **scale**(factor) | **IPixelGeometry** | Создает отмасштабированную копию геометрии.<br>Унаследован от **IPixelGeometry**. |
| **shift**(offset) | **IPixelGeometry** | Создает копию геометрии сдвинутую на заданную величину.<br>Унаследован от **IPixelGeometry**. |

## Описание методов

### getClosest

```
{Object} getClosest(anchorPosition)
```

Ищет на прямоугольнике точку, ближайшую к anchorPosition.

**Возвращает** объект со следующими полями:

- position — точка на прямоугольнике, ближайшая к anchorPosition;
- distance — расстояние от anchorPosition до position;
- closestPointIndex — индекс вершины прямоугольника, ближайшей к position;
- nextPointIndex — индекс вершины прямоугольника, следующей за position;
- prevPointIndex — индекс вершины прямоугольника, предшествующей position;
- pathIndex — индекс контура прямоугольника, к которому относится найденная точка.

Поля nextPointIndex, prevPointIndex могут отсутствовать, если position совпадает с одной из вершин прямоугольника.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| anchorPosition* | — | Тип: Number[]<br><br>Координаты точки, для которой рассчитывается ближайшая точка на прямоугольнике. |

* Обязательный параметр/опция.

### getCoordinates

```
{Number[][]} getCoordinates()
```

**Возвращает** координаты двух противоположных углов прямоугольника.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IPointGeometry

Расширяет **IGeometry**, **IPointGeometryAccess**.

Интерфейс геометрии "Точка".

Конструктор | Поля | События | Методы

## Конструктор

```
IPointGeometry()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты.<br>Унаследовано от **IPointGeometryAccess**. |
| **mapchange** | Сменилась карта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IGeometry**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **pixelgeometrychange** | Изменилась пиксельная геометрия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия **IPixelGeometry**.<br>Унаследовано от **IGeometry**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IGeometry**. |
| **getCoordinates**() | Number[] \| null | Возвращает координаты точки.<br>Унаследован от **IPointGeometryAccess**. |
| **getMap**() | **Map** \| null | Возвращает текущую карту.<br>Унаследован от **IGeometry**. |
| **getPixelGeometry**([options]) | **IPixelGeometry** | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.<br>Унаследован от **IGeometry**. |
| **getType**() | String | Возвращает строку "Point". |
| **setCoordinates**(coordinates) | **IPointGeometryAccess** | Задает координаты точки.<br>Унаследован от **IPointGeometryAccess**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |

## Описание методов

### getType

```
{String} getType()
```

**Возвращает** строку "Point".

# IPointGeometryAccess

Интерфейс доступа к геометрии "Точка".

Конструктор | События | Методы

## Конструктор

```
IPointGeometryAccess()
```

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **getCoordinates**() | Number[] \| null | Возвращает координаты точки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| setCoordinates(coordinates) | IPointGeometryAccess | Задает координаты точки. |

## Описание событий

### change

Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- oldCoordinates — старые координаты;
- newCoordinates — новые координаты.

## Описание методов

### getCoordinates

```
{Number[] | null} getCoordinates()
```

**Возвращает** координаты точки.

### setCoordinates

```
{IPointGeometryAccess} setCoordinates(coordinates)
```

Задает координаты точки.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates* | — | Тип: Number[] \| null<br><br>Координаты точки. |

* Обязательный параметр/опция.

# IPolygonGeometry

Расширяет **IGeometry**, **IPolygonGeometryAccess**.

Интерфейс геометрии "Многоугольник".

Конструктор | Поля | События | Методы

## Конструктор

```
IPolygonGeometry()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| options | IOptionManager | Менеджер опций.<br><br>Унаследовано от **ICustomizable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| change | Изменение координат. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты;<br>• oldFillRule — старый алгоритм заливки;<br>• newFillRule — новые алгоритм заливки.<br>Унаследовано от IPolygonGeometryAccess. |
| mapchange | Сменилась карта. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от IGeometry. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |
| pixelgeometrychange | Изменилась пиксельная геометрия. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия IPixelGeometry.<br>Унаследовано от IGeometry. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| contains(position) | Boolean | Проверяет, лежит ли переданная точка внутри многоугольника.<br>Унаследован от IPolygonGeometryAccess. |
| freeze() | IFreezable | Переводит объект в "замороженный" режим.<br>Унаследован от IFreezable. |
| get(index) | Number[][] | Возвращает координаты контура с заданным индексом.<br>Унаследован от IPolygonGeometryAccess. |
| getBounds() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.<br>Унаследован от IGeometry. |
| getChildGeometry(index) | ILinearRingGeometryAccess | Создает и возвращает объект ILinearRingGeometryAccess для заданного контура. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IPolygonGeometryAccess**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре многоугольника точку, ближайшую к anchorPosition.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getCoordinates**() | Number[][][] | Возвращает массив координат геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getFillRule**() | String | Возвращает идентификатор алгоритма заливки.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getLength**() | Integer | Возвращает количество контуров в геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **getMap**() | **Map** \| null | Возвращает текущую карту.<br>Унаследован от **IGeometry**. |
| **getPixelGeometry**([options]) | **IPixelGeometry** | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.<br>Унаследован от **IGeometry**. |
| **getType**() | String | Возвращает строку "Polygon". |
| **insert**(index, path) | **IPolygonGeometryAccess** | Добавляет новый контур с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | **ILinearRingGeometryAccess** | Удаляет контур с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **set**(index, path) | **IPolygonGeometryAccess** | Задает координаты контура с заданным индексом.<br>Унаследован от **IPolygonGeometryAccess**. |
| **setCoordinates**(coordinates) | **IPolygonGeometryAccess** | Задает массив координат геометрии.<br>Унаследован от **IPolygonGeometryAccess**. |
| **setFillRule**(fillRule) | **IPolygonGeometryAccess** | Задает алгоритм заливки многоугольника. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| | | Унаследован от **IPolygonGeometryAccess**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |
| **splice**(index, number) | **ILinearRingGeometryAccess**[] | Удаляет определенное число контуров начиная с заданного индекса. При этом на место удаляемых контуров могут быть добавлены новые. Координаты новых контуров можно передавать дополнительными аргументами после параметра number.<br>Унаследован от **IPolygonGeometryAccess**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание методов

### getType

```
{String} getType()
```

**Возвращает** строку "Polygon".

# IPolygonGeometryAccess

Расширяет **IFreezable**.

Интерфейс доступа к геометрии "Многоугольник".

Конструктор | Поля | События | Методы

## Конструктор

```
IPolygonGeometryAccess()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий объекта.<br>Унаследовано от **IFreezable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты;<br>• newCoordinates — новые координаты;<br>• oldFillRule — старый алгоритм заливки;<br>• newFillRule — новые алгоритм заливки. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри многоугольника. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **get**(index) | Number[][] | Возвращает координаты контура с заданным индексом. |
| **getChildGeometry**(index) | **ILinearRingGeometryAccess** | Создает и возвращает объект **ILinearRingGeometryAccess** для заданного контура. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре многоугольника точку, ближайшую к anchorPosition. |
| **getCoordinates**() | Number[][][] | Возвращает массив координат геометрии. |
| **getFillRule**() | String | Возвращает идентификатор алгоритма заливки. |
| **getLength**() | Integer | Возвращает количество контуров в геометрии. |
| **insert**(index, path) | **IPolygonGeometryAccess** | Добавляет новый контур с заданным индексом. |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **remove**(index) | **ILinearRingGeometryAccess** | Удаляет контур с заданным индексом. |
| **set**(index, path) | **IPolygonGeometryAccess** | Задает координаты контура с заданным индексом. |
| **setCoordinates**(coordinates) | **IPolygonGeometryAccess** | Задает массив координат геометрии. |
| **setFillRule**(fillRule) | **IPolygonGeometryAccess** | Задает алгоритм заливки многоугольника. |
| **splice**(index, number) | **ILinearRingGeometryAccess**[] | Удаляет определенное число контуров начиная с заданного индекса. При этом на место удаляемых контуров могут быть добавлены новые. Координаты новых контуров можно передавать дополнительными аргументами после параметра number. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Описание событий

### change

Изменение координат. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- oldCoordinates — старые координаты;
- newCoordinates — новые координаты;
- oldFillRule — старый алгоритм заливки;
- newFillRule — новые алгоритм заливки.

## Описание методов

### contains

```
{Boolean} contains(position)
```

Проверяет, лежит ли переданная точка внутри многоугольника.

**Возвращает** признак принадлежности точки многоугольнику.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position* | — | Тип: Number[]<br><br>Координаты точки. |

* Обязательный параметр/опция.

**Пример:**

```
var myPolygon = new ymaps.geometry.Polygon([
        [[69, 45], [68, 55], [86, 56]]
    ]);

// Метод работает только с корректно заданной картой.
myPolygon.options.setParent(myMap.options);
myPolygon.setMap(myMap);

// Проверка, входит ли точка клика в полигон, с заданной выше геометрией.
myMap.events.add('click', function (e) {
    alert(myPolygon.contains(e.get('coords')) ? 'Ранил!' : 'Мимо!');
});
```

### get

```
{Number[][]} get(index)
```

**Возвращает** координаты контура с заданным индексом.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс контура. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## getChildGeometry

```
{ILinearRingGeometryAccess} getChildGeometry(index)
```

Создает и возвращает объект **ILinearRingGeometryAccess** для заданного контура.

**Возвращает** объект геометрии, соответствующий заданному контуру.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс контура. |

* Обязательный параметр/опция.

## getClosest

```
{Object} getClosest(anchorPosition)
```

Ищет на контуре многоугольника точку, ближайшую к anchorPosition.

**Возвращает** объект со следующими полями:

- position — точка на контуре многоугольника ближайшая к anchorPosition;
- distance — расстояние от anchorPosition до position;
- closestPointIndex — индекс вершины многоугольника, ближайшей к position;
- nextPointIndex — индекс вершины многоугольника, следующей за position;
- prevPointIndex — индекс вершины многоугольника, предшествующей position;
- pathIndex — индекс контура многоугольника, к которому относится найденная точка.

Поля nextPointIndex, prevPointIndex могут отсутствовать, если position совпадает с одной из вершин многоугольника.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| anchorPosition* | — | Тип: Number[]<br><br>Координаты точки, для которой рассчитывается ближайшая точка на контуре многоугольника. |

* Обязательный параметр/опция.

**Пример:**


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

```
var myPolygon = new ymaps.Polygon([
        [[69, 45], [68, 55], [86, 56], [87, 43]],
        [[60, 48], [62, 60], [80, 62], [80, 48]]
    ]),
    myPoint = new ymaps.Placemark([74, 52]),
    // Построим кратчайшую нормаль от контура полигона до точки:
    normalVec = new ymaps.Polyline([

myPolygon.geometry.getClosest(myPoint.geometry.getCoordinates()).position,
        myPoint.geometry.getCoordinates()
    ]));

myMap.geoObjects
    .add(myPolygon)
    .add(myPoint),
    .add(normalVec);
```

## getCoordinates

```
{Number[][][]} getCoordinates()
```

**Возвращает** массив координат геометрии.

## getFillRule

```
{String} getFillRule()
```

**Возвращает** идентификатор алгоритма заливки.

**Пример:**

```
var myPolygon = new ymaps.Polygon([
        [[69, 45], [68, 55], [86, 56], [87, 43]],
        [[60, 48], [62, 60], [80, 62], [80, 48]]
    ]),
    middlePoint = [74, 52];

myMap.geoObjects.add(myPolygon);

// Проверка точки, приходящейся на пересечение двух контуров,
// по умолчанию (значение fillRule == 'evenOdd') образует вырез (дырку).
alert(myPolygon.geometry.contains(middlePoint)); // => false

// После задания значения 'nonZero', все пересечения также "закрашиваются",
// теперь точка входит в полигон.
myPolygon.geometry.setFillRule('nonZero');
alert(myPolygon.geometry.contains(middlePoint)); // => true
```

## getLength

```
{Integer} getLength()
```

**Возвращает** количество контуров в геометрии.

## insert

```
{IPolygonGeometryAccess} insert(index, path)
```

Добавляет новый контур с заданным индексом.

**Возвращает** ссылку на себя.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс контура. |
| path* | — | Тип: Number[][]<br><br>Координаты контура. |

* Обязательный параметр/опция.

## remove

```
{ILinearRingGeometryAccess} remove(index)
```

Удаляет контур с заданным индексом.

**Возвращает** удаленный контур.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс контура. |

* Обязательный параметр/опция.

## set

```
{IPolygonGeometryAccess} set(index, path)
```

Задает координаты контура с заданным индексом.

**Возвращает** ссылка на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс контура. |
| path* | — | Тип: Number[][]<br><br>Координаты контура. |

* Обязательный параметр/опция.

## setCoordinates

```
{IPolygonGeometryAccess} setCoordinates(coordinates)
```

Задает массив координат геометрии.

**Возвращает** ссылка на себя.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates* | — | Тип: Number[][][]<br><br>Координаты геометрии. |

* Обязательный параметр/опция.

## setFillRule

```
{IPolygonGeometryAccess} setFillRule(fillRule)
```

Задает алгоритм заливки многоугольника.

**Возвращает** ссылку на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| fillRule* | — | Тип: String<br><br>Идентификатор алгоритма заливки. |

* Обязательный параметр/опция.

## splice

```
{ILinearRingGeometryAccess[]} splice(index, number)
```

Удаляет определенное число контуров начиная с заданного индекса. При этом на место удаляемых контуров могут быть добавлены новые. Координаты новых контуров можно передавать дополнительными аргументами после параметра number.

**Возвращает** удаленные контуры.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| index* | — | Тип: Integer<br><br>Индекс, начиная с которого производится удаление и добавление контуров. |
| number* | — | Тип: Integer<br><br>Количество удаляемых контуров. |

* Обязательный параметр/опция.

# IPopup

Расширяет **ICustomizable**, **IEventEmitter**.

Интерфейс инфо-объекта.

Конструктор | Поля | События | Методы


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Конструктор

```
IPopup()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| close | Закрытие инфо-объекта. |
| open | Открытие инфо-объекта. |
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| close([force]) | vow.Promise | Закрывает инфо-объект. |
| getData() | | Возвращает данные инфо-объекта. |
| getOverlay() | vow.Promise | Возвращает объект-обещание вернуть оверлей. |
| getOverlaySync() | IOverlay | Возвращает оверлей, если тот существует. |
| getPosition() | | Возвращает координаты инфо-объекта. |
| isOpen() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт. |
| open([position[, data]]) | vow.Promise | Открывает инфо-объект в указанной позиции. Если инфо-объект уже открыт, перемещает его в указанную точку. Формат и суть координат определяется проекцией IProjection, которая содержится в опциях. |
| setData(data) | vow.Promise | Задаёт инфо-объекту новые данные. |
| setPosition(position) | vow.Promise | Задаёт инфо-объекту новую позицию. |

## Описание событий

### close

Закрытие инфо-объекта.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### open

Открытие инфо-объекта.

## Описание методов

### close

```
{vow.Promise} close([force])
```

Закрывает инфо-объект.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| force | false | Тип: Boolean<br><br>Мгновенное закрытие. |

### getData

```
getData()
```

**Возвращает** данные инфо-объекта.

### getOverlay

```
{vow.Promise} getOverlay()
```

**Возвращает** объект-обещание вернуть оверлей.

### getOverlaySync

```
{IOverlay} getOverlaySync()
```

**Возвращает** оверлей, если тот существует.

### getPosition

```
getPosition()
```

**Возвращает** координаты инфо-объекта.

### isOpen

```
{Boolean} isOpen()
```

**Возвращает** состояние инфо-объекта: открыт/закрыт.

### open

```
{vow.Promise} open([position[, data]])
```

Открывает инфо-объект в указанной позиции. Если инфо-объект уже открыт, перемещает его в указанную точку. Формат и суть координат определяется проекцией **IProjection**, которая содержится в опциях.

**Возвращает** объект-promise.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position | — | Тип: Number[]<br><br>Точка, где необходимо расположить балун. |
| data | — | Тип: Object \| String \| HTMLElement<br><br>Данные оверлея. |

## setData

```
{vow.Promise} setData(data)
```

Задаёт инфо-объекту новые данные.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data* | — | Тип: Object \| String \| HTMLElement<br><br>Данные инфо-объекта. |

* Обязательный параметр/опция.

## setPosition

```
{vow.Promise} setPosition(position)
```

Задаёт инфо-объекту новую позицию.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position* | — | Тип: Number[]<br><br>Координаты инфо-объекта. |

* Обязательный параметр/опция.

# IPopupManager

Расширяет **IEventEmitter**.

Интерфейс менеджера инфо-объекта.

Конструктор | Поля | Методы

## Конструктор

```
IPopupManager()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| close([force]) | vow.Promise | Закрывает инфо-объект. |
| destroy() | | Отключает менеджер инфо-объекта. |
| getData() | Object \| null | Возвращает данные инфо-объекта или `null`. |
| getOptions() | IOptionManager \| null | Возвращает менеджер опций или `null`. |
| getOverlay() | vow.Promise | Возвращает объект-обещание вернуть оверлей. |
| getOverlaySync() | IOverlay \| null | Возвращает оверлей, если тот существует. |
| getPosition() | Number[] \| null | Возвращает координаты инфо-объекта или `null`. |
| isOpen() | Boolean | Возвращает состояние инфо-объекта: открыт/закрыт. |
| open([position[, data[, options]]]) | vow.Promise | Открывает инфо-объект в указанной позиции. |
| setData(data) | vow.Promise | Задаёт инфо-объекту новые данные. |
| setOptions(options) | vow.Promise | Задаёт инфо-объекту новые опции. |
| setPosition(position) | vow.Promise | Задаёт инфо-объекту новую позицию. |

## Описание методов

### close

```
{vow.Promise} close([force])
```

Закрывает инфо-объект.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| force | false | Тип: Boolean<br>Мгновенное закрытие. |

### destroy

```
destroy()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Отключает менеджер инфо-объекта.

## getData

```
{Object | null} getData()
```

**Возвращает** данные инфо-объекта или `null`.

## getOptions

```
{IOptionManager | null} getOptions()
```

**Возвращает** менеджер опций или `null`.

## getOverlay

```
{vow.Promise} getOverlay()
```

**Возвращает** объект-обещание вернуть оверлей.

## getOverlaySync

```
{IOverlay | null} getOverlaySync()
```

**Возвращает** оверлей, если тот существует.

## getPosition

```
{Number[] | null} getPosition()
```

**Возвращает** координаты инфо-объекта или `null`.

## isOpen

```
{Boolean} isOpen()
```

**Возвращает** состояние инфо-объекта: открыт/закрыт.

## open

```
{vow.Promise} open([position[, data[, options]]])
```

Открывает инфо-объект в указанной позиции.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position | — | Тип: Number[]<br><br>Координаты открытия. |
| data | — | Тип: Object \| String \| HTMLElement<br><br>Данные. |
| options | — | Тип: Object<br><br>Опции. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### setData

```
{vow.Promise} setData(data)
```

Задаёт инфо-объекту новые данные.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| data* | — | Тип: Object \| String \| HTMLElement<br><br>Данные инфо-объекта. |

* Обязательный параметр/опция.

### setOptions

```
{vow.Promise} setOptions(options)
```

Задаёт инфо-объекту новые опции.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| options* | — | Тип: Object<br><br>Опции инфо-объекта. |

* Обязательный параметр/опция.

### setPosition

```
{vow.Promise} setPosition(position)
```

Задаёт инфо-объекту новую позицию.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position* | — | Тип: Number[]<br><br>Координаты инфо-объекта. |

* Обязательный параметр/опция.

## IPositioningContext

Интерфейс контекста позиционирования — объекта, позволяющего позиционировать внутри себя объект, заданный глобальными пиксельными координатами.

Конструктор | Методы


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Конструктор

```
IPositioningContext()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **fromClientPixels**(clientPixelPoint) | Number[] | Преобразует клиентские пиксельные координаты в глобальные. |
| **getZoom**() | Number | Возвращает текущий уровень масштабирования, в котором работает контекст позиционирования. |
| **toClientPixels**(globalPixelPoint) | Number[] | Преобразует глобальные пиксельные координаты в клиентские. |

## Описание методов

### fromClientPixels

```
{Number[]} fromClientPixels(clientPixelPoint)
```

Преобразует клиентские пиксельные координаты в глобальные.

**Возвращает** глобальные пиксельные координаты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| clientPixelPoint* | — | Тип: Number[]<br><br>Клиентские пиксельные координаты. |

* Обязательный параметр/опция.

### getZoom

```
{Number} getZoom()
```

**Возвращает** текущий уровень масштабирования, в котором работает контекст позиционирования.

### toClientPixels

```
{Number[]} toClientPixels(globalPixelPoint)
```

Преобразует глобальные пиксельные координаты в клиентские.

**Возвращает** клиентские пиксельные координаты.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| globalPixelPoint* | — | Тип: Number[]<br><br>Глобальные пиксельные координаты. |

* Обязательный параметр/опция.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

# IProjection

Проекция. Описывает, каким образом реальная карта проецируется на бесконечную пиксельную плоскость. Один "мир" должен иметь размеры 256x256 пикселей на нулевом масштабе, при этом левый верхний угол "мира" имеет координаты (0, 0), координатные оси направлены вправо и вниз. "Миры" могут быть склеенными по любой оси (либо по обеим осям сразу).

Конструктор | Методы

## Конструктор

```
IProjection()
```

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **fromGlobalPixels**(globalPixelPoint, zoom) | Number[] | Преобразует пиксельные координаты на указанном уровне масштабирования в координаты проекции. |
| **getCoordSystem**() | **ICoordSystem** | Возвращает координатную систему, используемую проекцией. |
| **isCycled**() | Boolean[] | Показатель зацикленности проекции. |
| **toGlobalPixels**(coordPoint, zoom) | Number[] | Преобразует координаты проекции в глобальные пиксельные на указанном уровне масштабирования. |

## Описание методов

### fromGlobalPixels

```
{Number[]} fromGlobalPixels(globalPixelPoint, zoom)
```

Преобразует пиксельные координаты на указанном уровне масштабирования в координаты проекции.

**Возвращает** точку в координатах проекции.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| globalPixelPoint* | — | Тип: Number[]<br><br>Точка в пиксельных координатах. |
| zoom* | — | Тип: Number<br><br>Уровень масштабирования. |

* Обязательный параметр/опция.

### getCoordSystem

```
{ICoordSystem} getCoordSystem()
```

Возвращает координатную систему, используемую проекцией.

**Возвращает** координатную систему.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### isCycled

```
{Boolean[]} isCycled()
```

Показатель зацикленности проекции.

**Возвращает** пару флагов, показывающих, склеена ли карта по пиксельным осям (x/y).

### toGlobalPixels

```
{Number[]} toGlobalPixels(coordPoint, zoom)
```

Преобразует координаты проекции в глобальные пиксельные на указанном уровне масштабирования.

**Возвращает** пару пиксельных координат.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordPoint* | — | Тип: Number[]<br><br>Точка в координатах проекции. |
| zoom* | — | Тип: Number<br><br>Уровень масштабирования. |

* Обязательный параметр/опция.

## IPromiseProvider

Объект, выполняющий "обещание".

Конструктор | Методы

### Конструктор

```
IPromiseProvider()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **then**(onResolve, onReject) | **IPromiseProvider** | Возвращает объект-обещание. |

## Описание методов

### then

```
{IPromiseProvider} then(onResolve, onReject)
```

Возвращает объект-обещание.

**Возвращает** ссылку на себя или новый объект-обещание.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| onResolve* | — | Тип: Function |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Функция-обработчик, которая вызывается, если обещание выполнено. |
| onReject* | — | Тип: Function<br><br>Функция-обработчик, которая вызывается, если обещание не выполнено (произошла ошибка). |

* Обязательный параметр/опция.

# IRatioMap

Интерфейс объекта, содержащего соотношение произвольных данных к devicePixelRatio. Используется в случаях, когда требуется поддержка экранов с отношением виртуальных пикселей к физическим более единицы. В качестве ключей в объекте должны использваться строки, состоящие из целого или дробного числа, обозначающего коэффициент плотности пикселей.

Конструктор

## Конструктор

```
IRatioMap()
```

**Пример:**

```
// Пусть, нужно корректно отобразить изображение.
var images = {
    // Для обычных экранов будем показывать нормальную картинку.
    "1": "100x100.png",
    // Для HTC Desire, Samsung Galaxy S II и других с devicePixelRatio = 1.5.
    "1.5": "150x150.png",
    // Для Apple устройств с ретина-экраном, а также, например, Sony Xperia S,
HTC One X и других
    // с отношением = 2.
    "2": "200x100.png"
}
```

# IRectangleGeometry

Расширяет **IGeometry**, **IRectangleGeometryAccess**.

Интерфейс геометрии "Прямоугольник".

Конструктор | Поля | События | Методы

## Конструктор

```
IRectangleGeometry()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br><br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Тип | Описание |
|---|---|---|
| | | Унаследовано от **ICustomizable**. |

## События

| Имя | Описание |
|---|---|
| **change** | Изменение координат углов. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты углов;<br>• newCoordinates — новые координаты углов.<br>Унаследовано от **IRectangleGeometryAccess**. |
| **mapchange** | Сменилась карта. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• oldMap — старая карта;<br>• newMap — новая карта.<br>Унаследовано от **IGeometry**. |
| **optionschange** | Изменение в опциях объекта.<br>Унаследовано от **ICustomizable**. |
| **pixelgeometrychange** | Изменилась пиксельная геометрия. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• pixelGeometry — новая пиксельная геометрия **IPixelGeometry**.<br>Унаследовано от **IGeometry**. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет, лежит ли переданная точка внутри прямоугольника.<br>Унаследован от **IRectangleGeometryAccess**. |
| **freeze**() | **IFreezable** | Переводит объект в "замороженный" режим.<br>Унаследован от **IFreezable**. |
| **getBounds**() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей геометрию. Первый элемент массива — юго-западный угол области; второй элемент — северо-восточный угол соответственно.<br>Унаследован от **IGeometry**. |
| **getClosest**(anchorPosition) | Object | Ищет на контуре прямоугольника точку, ближайшую к anchorPosition.<br>Унаследован от **IRectangleGeometryAccess**. |
| **getCoordinates**() | Number[][] | Возвращает координаты двух противоположных углов прямоугольника.<br>Унаследован от **IRectangleGeometryAccess**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getMap**() | **Map** \| null | Возвращает текущую карту.<br>Унаследован от **IGeometry**. |
| **getPixelGeometry**([options]) | **IPixelGeometry** | Возвращает пиксельную геометрию, соответствующую данной геометрии, ее опциям и состоянию карты.<br>Унаследован от **IGeometry**. |
| **getType**() | String | Возвращает строку "Rectangle". |
| **isFrozen**() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от **IFreezable**. |
| **setCoordinates**(coordinates) | **IRectangleGeometryAccess** | Задает координаты двух противоположных углов прямоугольника.<br>Унаследован от **IRectangleGeometryAccess**. |
| **setMap**(map) | | Задает карту.<br>Унаследован от **IGeometry**. |
| **unfreeze**() | **IFreezable** | Переводит объект в активный режим.<br>Унаследован от **IFreezable**. |

## Описание методов

### getType

```
{String} getType()
```

**Возвращает** строку "Rectangle".

# IRectangleGeometryAccess

Расширяет **IFreezable**.

Интерфейс доступа к геометрии "Прямоугольник".

Конструктор | Поля | События | Методы

## Конструктор

```
IRectangleGeometryAccess()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий объекта.<br>Унаследовано от **IFreezable**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## События

| Имя | Описание |
|---|---|
| change | Изменение координат углов. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• oldCoordinates — старые координаты углов;<br>• newCoordinates — новые координаты углов. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| contains(position) | Boolean | Проверяет, лежит ли переданная точка внутри прямоугольника. |
| freeze() | IFreezable | Переводит объект в "замороженный" режим.<br>Унаследован от IFreezable. |
| getClosest(anchorPosition) | Object | Ищет на контуре прямоугольника точку, ближайшую к anchorPosition. |
| getCoordinates() | Number[][] | Возвращает координаты двух противоположных углов прямоугольника. |
| isFrozen() | Boolean | Возвращает true, если объект находится в "замороженном" режиме, иначе false.<br>Унаследован от IFreezable. |
| setCoordinates(coordinates) | IRectangleGeometryAccess | Задает координаты двух противоположных углов прямоугольника. |
| unfreeze() | IFreezable | Переводит объект в активный режим.<br>Унаследован от IFreezable. |

## Описание событий

### change

Изменение координат углов. Экземпляр класса Event. Имена полей, доступных через метод Event.get:

- oldCoordinates — старые координаты углов;
- newCoordinates — новые координаты углов.

## Описание методов

### contains

```
{Boolean} contains(position)
```

Проверяет, лежит ли переданная точка внутри прямоугольника.

**Возвращает** признак принадлежности точки прямоугольнику.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position* | — | Тип: Number[]<br><br>Координаты точки. |

* Обязательный параметр/опция.

## getClosest

```
{Object} getClosest(anchorPosition)
```

Ищет на контуре прямоугольника точку, ближайшую к anchorPosition.

**Возвращает** объект со следующими полями:

- position — точка на контуре прямоугольника, ближайшая к anchorPosition;
- distance — расстояние от anchorPosition до position;

.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| anchorPosition* | — | Тип: Number[]<br><br>Координаты точки, для которой рассчитывается ближайшая точка на прямоугольнике. |

* Обязательный параметр/опция.

## getCoordinates

```
{Number[][]} getCoordinates()
```

**Возвращает** координаты двух противоположных углов прямоугольника.

## setCoordinates

```
{IRectangleGeometryAccess} setCoordinates(coordinates)
```

Задает координаты двух противоположных углов прямоугольника.

**Возвращает** ссылка на себя.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| coordinates* | — | Тип: Number[][]<br><br>Координаты углов. |

* Обязательный параметр/опция.

# ISearchControlLayout

Расширяет **IExpandableControlLayout**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Интерфейс макета элемента управления "Поиск по карте".

Конструктор | Поля | События | Методы

## Конструктор

```
ISearchControlLayout()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| collapse | Событие, инициирующее сворачивание объекта.<br>Унаследовано от IExpandableControlLayout. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| emptinesschange | Изменение признака пустоты макета. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| expand | Событие, инициирующее разворачивание объекта.<br>Унаследовано от IExpandableControlLayout. |
| loadmore | Событие, инициирующее дополнительную подгрузку результатов поиска. Экземпляр класса Event. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **parentelementchange** | Изменение родительского элемента. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **resultselect** | Событие, инициирующее показ результата поиска на карте. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• index — номер объекта в ответе сервера, который нужно показать на карте. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| search | Событие, инициирующее поиск объектов. Экземпляр класса Event. Имена полей, доступных через метод Event.get:<br>• request — строка, содержащая запрос. |
| shapechange | Изменение фигуры области охватывающей макет. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Деструктор. Вызывается при завершении работы с макетом.<br>Унаследован от ILayout. |
| getData() | Object | Возвращает объект данных макета.<br>Унаследован от ILayout. |
| getParentElement() | HTMLElement | Возвращает родительский html-элемент.<br>Унаследован от ILayout. |
| getShape() | IShape \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.<br>Унаследован от ILayout. |
| isEmpty() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от ILayout. |
| setData(data) | | Задает данные макета.<br>Унаследован от ILayout. |
| setParentElement(parent) | | Добавляет макет в DOM-дерево.<br>Унаследован от ILayout. |

## Описание событий

### loadmore

Событие, инициирующее дополнительную подгрузку результатов поиска. Экземпляр класса Event.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### resultselect

Событие, инициирующее показ результата поиска на карте. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- index — номер объекта в ответе сервера, который нужно показать на карте.

### search

Событие, инициирующее поиск объектов. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- request — строка, содержащая запрос.

## ISearchProvider

Интерфейс провайдера поиска.

Конструктор | Методы

### Конструктор

```
ISearchProvider()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **search**(request[, options]) | **vow.Promise** | Отправляет запрос на поиск геообъектов. Добавить функцию-обработчик для обработки результата геокодирования можно через возвращаемый объект-promise. Объект, поступающий на вход функции-обработчика может содержать поля только следующего вида: geoObjects, layers, mapState, metaData. |
| **suggest**(request[, options]) | **vow.Promise** | Отправляет запрос за поисковыми подсказками. Возвращает объект-promise, который будет либо отклонён с ошибкой, либо подтверждён массивом объектов вида { displayName: "Мытищи, Московская область", value: "Россия, Московская область, Мытищи " }. Поле displayName отвечает за представление топонима в удобном для чтения пользователем виде, поле value — значение, которое необходимо подставить в поле ввода после выбора пользователем данной подсказки.<br><br>Этот метод является опциональным. |

### Описание методов

### search

```
{vow.Promise} search(request[, options])
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Отправляет запрос на поиск геообъектов. Добавить функцию-обработчик для обработки результата геокодирования можно через возвращаемый объект-promise. Объект, поступающий на вход функции-обработчика может содержать поля только следующего вида: geoObjects, layers, mapState, metaData.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| request* | — | Тип: String<br><br>Строка запроса. |
| options | — | Тип: Object<br><br>Опции. |
| options.boundedBy | — | Тип: Number[][]<br><br>Прямоугольная область на карте, где предположительно находится искомый объект. Должна быть задана как массив, например [[30, 40], [50, 50]]. |
| options.results | — | Тип: Number<br><br>Максимальное количество возвращаемых результатов. |
| options.skip | — | Тип: Number<br><br>Число результатов, которое необходимо пропустить. |

* Обязательный параметр/опция.

## suggest

```
{vow.Promise} suggest(request[, options])
```

Отправляет запрос за поисковыми подсказками. Возвращает объект-promise, который будет либо отклонён с ошибкой, либо подтверждён массивом объектов вида { displayName: "Мытищи, Московская область", value: "Россия, Московская область, Мытищи " }. Поле displayName отвечает за представление топонима в удобном для чтения пользователем виде, поле value — значение, которое необходимо подставить в поле ввода после выбора пользователем данной подсказки.

Этот метод является опциональным.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| request* | — | Тип: String<br><br>Строка запроса. |
| options | — | Тип: Object<br><br>Опции. |
| options.boundedBy | — | Тип: Number[][] |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Прямоугольная область на карте, где предположительно находится искомый объект. Должна быть задана как массив, например [[30, 40], [50, 50]]. |
| options.results | — | Тип: Number<br><br>Максимальное количество возвращаемых результатов. |
| options.strictBounds | — | Тип: Boolean<br><br>Искать только внутри области, заданной опцией boundedBy. |

* Обязательный параметр/опция.

# ISelectableControl

Расширяет **IControl**.

Интерфейс переключаемого и выбираемого элемента управления.

Конструктор | Поля | События | Методы

## Конструктор

```
ISelectableControl()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IEventEmitter**. |
| **options** | **IOptionManager** | Менеджер опций.<br>Унаследовано от **IControl**. |

## События

| Имя | Описание |
|---|---|
| **deselect** | Элемент управления не выбран. |
| **disable** | Элемент управления стал недоступным. |
| **enable** | Элемент управления стал доступным. |
| **parentchange** | Сменился родительский объект.<br>Поля данных:<br>• oldParent — старый родитель;<br>• newParent — новый родитель.<br>Унаследовано от **IChild**. |
| **select** | Элемент управления выбран. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| deselect() | | Отменяет выбор элемента управления (выключает). |
| disable() | | Делает элемент управления недоступным (запрещены пользовательские действия). |
| enable() | | Делает элемент управления доступным (разрешены пользовательские действия). |
| getParent() | IControlParent \| null | Возвращает ссылку на родительский объект или null, если родительский элемент не был установлен.<br>Унаследован от IControl. |
| isEnabled() | Boolean | Возвращает true — элемент управления доступен, false — недоступен. |
| isSelected() | Boolean | Возвращает true — элемент управления выбран, false — не выбран. |
| select() | | Выбирает (включает) элемент управления. |
| setParent(parent) | IChildOnMap | Устанавливает родительский объект. Если передать значение null, то элемент управления будет только удален из текущего родительского объекта.<br>Унаследован от IControl. |

## Описание событий

### deselect

Элемент управления не выбран.

### disable

Элемент управления стал недоступным.

### enable

Элемент управления стал доступным.

### select

Элемент управления выбран.

## Описание методов

### deselect

```
deselect()
```

Отменяет выбор элемента управления (выключает).


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### disable

```
disable()
```

Делает элемент управления недоступным (запрещены пользовательские действия).

### enable

```
enable()
```

Делает элемент управления доступным (разрешены пользовательские действия).

### isEnabled

```
{Boolean} isEnabled()
```

**Возвращает** true — элемент управления доступен, false — недоступен.

### isSelected

```
{Boolean} isSelected()
```

**Возвращает** true — элемент управления выбран, false — не выбран.

### select

```
select()
```

Выбирает (включает) элемент управления.

## ISelectableControlLayout

Расширяет **ILayout**.

Интерфейс макета кнопки.

Конструктор | Поля | События | Методы

### Конструктор

```
ISelectableControlLayout()
```

### Поля

| Имя | Тип | Описание |
|---|---|---|
| **events** | **IEventManager** | Менеджер событий.<br>Унаследовано от **IDomEventEmitter**. |

### События

| Имя | Описание |
|---|---|
| **click** | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| **contextmenu** | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **dblclick** | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **deselect** | Событие, инициирующее выход элемента из состояния "выбран". |
| **emptinesschange** | Изменение признака пустоты макета. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **mousedown** | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseenter** | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseleave** | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mousemove** | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **mouseup** | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| multitouchmove | Повторящееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| multitouchstart | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br>Унаследовано от IDomEventEmitter. |
| parentelementchange | Изменение родительского элемента. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| select | Событие, инициирующее переход элемента в состояние "выбран". |
| shapechange | Изменение фигуры области охватывающей макет. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| wheel | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| destroy() | | Деструктор. Вызывается при завершении работы с макетом.<br>Унаследован от ILayout. |
| getData() | Object | Возвращает объект данных макета.<br>Унаследован от ILayout. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| **getParentElement**() | HTMLElement | Возвращает родительский html-элемент.<br>Унаследован от **ILayout**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.<br>Унаследован от **ILayout**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **ILayout**. |
| **setData**(data) | | Задает данные макета.<br>Унаследован от **ILayout**. |
| **setParentElement**(parent) | | Добавляет макет в DOM-дерево.<br>Унаследован от **ILayout**. |

### Описание событий

#### deselect

Событие, инициирующее выход элемента из состояния "выбран".

#### select

Событие, инициирующее переход элемента в состояние "выбран".

## IShape

Интерфейс фигуры. Фигура представляет совокупность пиксельной геометрии, а также математических и логических параметров ее отображения (таких как наличие контура и его ширина, наличие заливки).

Конструктор | Методы

### Конструктор

```
IShape()
```

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **contains**(position) | Boolean | Проверяет лежит ли переданная точка внутри фигуры. |
| **equals**(shape) | Boolean | Возвращает true, если переданная фигура эквивалентна данной. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Возвращает | Описание |
|---|---|---|
| getBounds() | Number[][] \| null | Возвращает координаты двух противоположных углов области, охватывающей фигуру. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат. |
| getGeometry() | IPixelGeometry | Возвращает пиксельную геометрию фигуры. |
| getType() | String | Возвращает идентификатор типа фигуры. |
| scale(factor) | IShape | Создает отмасштабированную копию фигуры. |
| shift(offset) | IShape | Создает копию фигуры сдвинутую на заданную величину. |

## Описание методов

### contains

```
{Boolean} contains(position)
```

Проверяет лежит ли переданная точка внутри фигуры.

**Возвращает** true, если переданная точка лежит внутри фигуры.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| position* | — | Тип: Number[]<br><br>Координаты точки. |

* Обязательный параметр/опция.

### equals

```
{Boolean} equals(shape)
```

**Возвращает** true, если переданная фигура эквивалентна данной.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| shape* | — | Тип: IShape<br><br>Фигура, с которой производится сравнение. |

* Обязательный параметр/опция.

### getBounds

```
{Number[][] | null} getBounds()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

**Возвращает** координаты двух противоположных углов области, охватывающей фигуру. Первый элемент массива — угол с наименьшими значениями координат относительно остальных точек области; второй элемент — угол с наибольшими значениями координат.

### getGeometry

```
{IPixelGeometry} getGeometry()
```

**Возвращает** пиксельную геометрию фигуры.

### getType

```
{String} getType()
```

**Возвращает** идентификатор типа фигуры.

### scale

```
{IShape} scale(factor)
```

Создает отмасштабированную копию фигуры.

**Возвращает** отмасштабированную копию фигуры.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| factor* | — | Тип: Number<br><br>Масштабный коэффициент. |

* Обязательный параметр/опция.

### shift

```
{IShape} shift(offset)
```

Создает копию фигуры сдвинутую на заданную величину.

**Возвращает** сдвинутую копию фигуры.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| offset* | — | Тип: Number[]<br><br>Величина сдвига по осям. |

* Обязательный параметр/опция.

## ISuggestProvider

Интерфейс провайдера поисковых подсказок.

Конструктор | Методы

### Конструктор

```
ISuggestProvider()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| suggest(request[, options]) | vow.Promise | Отправляет запрос за поисковыми подсказками. Возвращает объект-promise, который будет либо отклонён с ошибкой, либо подтверждён массивом объектов вида { displayName: "Мытищи, Московская область", value: "Россия, Московская область, Мытищи ", hl: [[0,5]] }. Поле displayName отвечает за представление топонима в удобном для чтения пользователем виде, поле value — значение, которое необходимо подставить в поле ввода после выбора пользователем данной подсказки, поле hl — массив диапазонов подсветки, необходимых, чтобы показать, какая именно часть результата совпала с запросом. Диапазон подсветки — массив из двух чисел: индексы начального и конечного символов диапазона. |

## Описание методов

### suggest

```
{vow.Promise} suggest(request[, options])
```

Отправляет запрос за поисковыми подсказками. Возвращает объект-promise, который будет либо отклонён с ошибкой, либо подтверждён массивом объектов вида { displayName: "Мытищи, Московская область", value: "Россия, Московская область, Мытищи ", hl: [[0,5]] }. Поле displayName отвечает за представление топонима в удобном для чтения пользователем виде, поле value — значение, которое необходимо подставить в поле ввода после выбора пользователем данной подсказки, поле hl — массив диапазонов подсветки, необходимых, чтобы показать, какая именно часть результата совпала с запросом. Диапазон подсветки — массив из двух чисел: индексы начального и конечного символов диапазона.

**Возвращает** объект-promise.

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| request* | — | Тип: String<br><br>Строка запроса. |
| options | — | Тип: Object<br><br>Опции. |
| options.boundedBy | — | Тип: Number[][]<br><br>Прямоугольная область на карте, где предположительно находится искомый объект. Должна быть задана как массив, например [[30, 40], [50, 50]]. |
| options.results | — | Тип: Number |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| | | Максимальное количество возвращаемых результатов. |

* Обязательный параметр/опция.

# ISuggestViewLayout

Интерфейс макета панели поисковых подсказок.

Конструктор | События

## Конструктор

```
ISuggestViewLayout()
```

## События

| Имя | Описание |
|---|---|
| **hover** | Событие, возникающее при выделении одной из подсказок в списке наведением мыши на результат или с помощью клавиатуры. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• index: Number\|null — индекс элемента в списке поисковых подсказок. |
| **select** | Событие, возникающее при выборе пользователем одной из подсказок. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br>• item: Object — объект с полями displayName и value. |

## Описание событий

### hover

Событие, возникающее при выделении одной из подсказок в списке наведением мыши на результат или с помощью клавиатуры. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- index: Number|null — индекс элемента в списке поисковых подсказок.

### select

Событие, возникающее при выборе пользователем одной из подсказок. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- item: Object — объект с полями displayName и value.

# ITrafficControlLayout

Расширяет **IExpandableControlLayout**.

Интерфейс макета элемента управления "Панель управления пробками".

Конструктор | Поля | События | Методы

## Конструктор

```
ITrafficControlLayout()
```


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| collapse | Событие, инициирующее сворачивание объекта.<br>Унаследовано от IExpandableControlLayout. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| emptinesschange | Изменение признака пустоты макета. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| expand | Событие, инициирующее разворачивание объекта.<br>Унаследовано от IExpandableControlLayout. |
| hide | Событие, инициирующее удаление пробок с карты. Скрыть пробки. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mousemove | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseup | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
|---|---|
| | Унаследовано от **IDomEventEmitter**. |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **parentelementchange** | Изменение родительского элемента. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **providerkeychange** | Событие, инициирующее смену ключа провайдера. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• newProviderKey — новое значение ключа провайдера;<br>• oldProviderKey — старое значение ключа. |
| **shapechange** | Изменение фигуры области охватывающей макет. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **show** | Событие, инициирующее показ пробок на карте. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

### Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Деструктор. Вызывается при завершении работы с макетом.<br>Унаследован от **ILayout**. |
| **getData**() | Object | Возвращает объект данных макета.<br>Унаследован от **ILayout**. |
| **getParentElement**() | HTMLElement | Возвращает родительский html-элемент.<br>Унаследован от **ILayout**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.<br>Унаследован от **ILayout**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **ILayout**. |
| **setData**(data) | | Задает данные макета.<br>Унаследован от **ILayout**. |
| **setParentElement**(parent) | | Добавляет макет в DOM-дерево.<br>Унаследован от **ILayout**. |

## Описание событий

### hide

Событие, инициирующее удаление пробок с карты. Скрыть пробки.

### providerkeychange

Событие, инициирующее смену ключа провайдера. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- newProviderKey — новое значение ключа провайдера;
- oldProviderKey — старое значение ключа.

### show

Событие, инициирующее показ пробок на карте.

# ITrafficProvider

Расширяет **ICustomizable**, **IEventEmitter**.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Интерфейс провайдера данных о пробках.

Конструктор | Поля | События | Методы

## Конструктор

```
ITrafficProvider()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IEventEmitter. |
| options | IOptionManager | Менеджер опций.<br>Унаследовано от ICustomizable. |

## События

| Имя | Описание |
|---|---|
| optionschange | Изменение в опциях объекта.<br>Унаследовано от ICustomizable. |

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| getMap() | Map \| null | Возвращает ссылку на карту. |
| setMap(Ссылка) | | |

## Описание методов

### getMap

```
{Map | null} getMap()
```

**Возвращает** ссылку на карту.

### setMap

```
setMap(Ссылка)
```

**Параметры:**

| Параметр | Значение по умолчанию | Описание |
|---|---|---|
| Ссылка* | — | Тип: Map \| null<br>на карту. |

* Обязательный параметр/опция.

# ITransportProperties

Данный класс не имеет конструктора и предназначен для описания объекта данных транспортного средства в транспортном сегменте маршрута общественного транспорта.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Конструктор | Поля

## Конструктор

```
ITransportProperties()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| id | String | Идентификатор транспортного средства. |
| name | String | Название маршрута транспортного средства. |
| type | String | Идентификатор вида транспортного средства. Может принимать одно из следующих строковых значений:<br>• "bus" — автобус;<br>• "trolleybus" — троллейбус;<br>• "tramway" — трамвай;<br>• "minibus" — маршрутное такси;<br>• "underground" — метро;<br>• "suburban" — электричка. |

## Описание полей

### id

```
{String} id
```

Идентификатор транспортного средства.

### name

```
{String} name
```

Название маршрута транспортного средства.

### type

```
{String} type
```

Идентификатор вида транспортного средства. Может принимать одно из следующих строковых значений:

- "bus" — автобус;
- "trolleybus" — троллейбус;
- "tramway" — трамвай;
- "minibus" — маршрутное такси;
- "underground" — метро;
- "suburban" — электричка.

# IZoomControlLayout

Расширяет **ILayout**.

Интерфейс макета элемента управления "Ползунок масштаба".


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Конструктор | Поля | События | Методы

## Конструктор

```
IZoomControlLayout()
```

## Поля

| Имя | Тип | Описание |
|---|---|---|
| events | IEventManager | Менеджер событий.<br>Унаследовано от IDomEventEmitter. |

## События

| Имя | Описание |
|---|---|
| click | Однократное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| contextmenu | Вызов контекстного меню на элементе. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| dblclick | Двойное нажатие левой кнопки мыши на объекте. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| emptinesschange | Изменение признака пустоты макета. Экземпляр класса Event.<br>Унаследовано от ILayout. |
| mousedown | Нажатие кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseenter | Наведение курсора на объект. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseleave | Вывод курсора за пределы объекта. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mousemove | Перемещение курсора над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |
| mouseup | Отпускание кнопки мыши над объектом. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в domEvent.manager.<br>Унаследовано от IDomEventEmitter. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

| Имя | Описание |
| --- | --- |
| **multitouchend** | Окончание мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchmove** | Повторяющееся событие при мультисенсорном управлении. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **multitouchstart** | Начало мультисенсорного управления. Данное событие доступно только на устройствах, которые поддерживают множественные сенсорные прикосновения. Возвращает реализацию интерфейса IMultiTouchEvent с информацией о прикосновениях. Определяет свойство touches, которое содержит список прикосновений. Каждое прикосновение описано объектом, которое содержит поля:<br><br>• clientX — координата касания X относительно видимой области браузера;<br>• clientY — координата касания Y относительно видимой области браузера;<br>• pageX — координата касания X касания относительно начала документа;<br>• pageY — координата касания Y касания относительно начала документа.<br><br>Унаследовано от **IDomEventEmitter**. |
| **parentelementchange** | Изменение родительского элемента. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **shapechange** | Изменение фигуры области охватывающей макет. Экземпляр класса **Event**.<br><br>Унаследовано от **ILayout**. |
| **wheel** | Скролл мышкой. При использовании следует иметь в виду, что при сенсорном вводе события мыши будут эмулироваться. Более детально описание в **domEvent.manager**.<br><br>Унаследовано от **IDomEventEmitter**. |
| **zoomchange** | Событие, инициирующее смену коэффициента масштабирования карты. Изменить коэффициент масштабирования карты. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:<br><br>• newZoom — новое значение коэффициента масштабирования;<br>• oldZoom — старое значение коэффициента масштабирования. |


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

## Методы

| Имя | Возвращает | Описание |
|---|---|---|
| **destroy**() | | Деструктор. Вызывается при завершении работы с макетом.<br>Унаследован от **ILayout**. |
| **getData**() | Object | Возвращает объект данных макета.<br>Унаследован от **ILayout**. |
| **getParentElement**() | HTMLElement | Возвращает родительский html-элемент.<br>Унаследован от **ILayout**. |
| **getShape**() | **IShape** \| null | Возвращает фигуру, описывающую форму области, охватывающей макет, либо null, если нет возможности рассчитать данную фигуру. Координаты геометрии фигуры должны отсчитываться от точки привязки родительского элемента макета.<br>Унаследован от **ILayout**. |
| **isEmpty**() | Boolean | Возвращает true, если макет пуст, т.е. не имеет содержимого. Данный признак используется для сокрытия пустых объектов, таких как хинт, балун и другие.<br>Унаследован от **ILayout**. |
| **setData**(data) | | Задает данные макета.<br>Унаследован от **ILayout**. |
| **setParentElement**(parent) | | Добавляет макет в DOM-дерево.<br>Унаследован от **ILayout**. |

## Описание событий

### zoomchange

Событие, инициирующее смену коэффициента масштабирования карты. Изменить коэффициент масштабирования карты. Экземпляр класса **Event**. Имена полей, доступных через метод Event.get:

- newZoom — новое значение коэффициента масштабирования;
- oldZoom — старое значение коэффициента масштабирования.


Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/

Яндекс

# Справочник API Яндекс.Карт

Версия 2.1.42

13.07.2016

Antenna House XSL Formatter (Evaluation) http://www.antennahouse.com/