Google Maps JavaScript API V3 Reference

Release Version

Last updated Tuesday, July 29, 2014

> This reference documents version 3.19 (the ***release*** version) of the Maps Javascript API released February 15, 2015. This release version of the API is a feature-stable version of the API whose interfaces are guaranteed to remain as documented within these pages until this version is retired.
>
> To consult the ***latest*** (experimental) version of the Maps Javascript API, see the Experimental Development Reference.

Reference Table of Contents

- Data.RemovePropertyEvent

Overlays

- Marker
- MarkerOptions
- Icon
- MarkerShape
- Symbol
- SymbolPath
- Animation
- InfoWindow
- InfoWindowOptions
- Polyline
- PolylineOptions
- IconSequence
- Polygon
- PolygonOptions
- PolyMouseEvent
- Rectangle
- RectangleOptions
- Circle
- CircleOptions
- StrokePosition
- GroundOverlay
- GroundOverlayOptions
- OverlayView
- MapPanes
- MapCanvasProjection

Services

- Geocoder
- GeocoderRequest
- GeocoderComponentRestrictions
- GeocoderStatus
- GeocoderResult
- GeocoderAddressComponent
- GeocoderGeometry
- GeocoderLocationType
- DirectionsRenderer
- DirectionsRendererOptions
- DirectionsService
- DirectionsRequest
- TravelMode
- UnitSystem
- TransitOptions
- TransitMode
- TransitRoutePreference
- TransitFare
- DirectionsWaypoint
- DirectionsStatus
- DirectionsResult
- DirectionsRoute
- DirectionsLeg
- DirectionsStep
- Distance
- Duration
- Time
- TransitDetails
- TransitStop
- TransitLine
- TransitAgency
- TransitVehicle
- VehicleType
- ElevationService
- LocationElevationRequest

- PathElevationRequest
- ElevationResult
- ElevationStatus
- MaxZoomService
- MaxZoomResult
- MaxZoomStatus
- DistanceMatrixService
- DistanceMatrixRequest
- DistanceMatrixResponse
- DistanceMatrixResponseRow
- DistanceMatrixResponseElement
- DistanceMatrixStatus
- DistanceMatrixElementStatus

Save to Google Maps

- Attribution
- Place
- SaveWidget
- SaveWidgetOptions

Map Types

- MapType
- MapTypeRegistry
- Projection
- ImageMapType
- ImageMapTypeOptions
- StyledMapType
- StyledMapTypeOptions
- MapTypeStyle
- MapTypeStyleFeatureType
- MapTypeStyleElementType
- MapTypeStyler

Layers

- BicyclingLayer
- FusionTablesLayer
- FusionTablesLayerOptions
- FusionTablesQuery
- FusionTablesStyle
- FusionTablesHeatmap
- FusionTablesMarkerOptions
- FusionTablesPolygonOptions
- FusionTablesPolylineOptions
- FusionTablesMouseEvent
- FusionTablesCell
- KmlLayer
- KmlLayerOptions
- KmlLayerMetadata
- KmlLayerStatus
- KmlMouseEvent
- KmlFeatureData
- KmlAuthor
- TrafficLayer
- TransitLayer

Street View

- StreetViewPanorama
- StreetViewPanoramaOptions
- StreetViewAddressControlOptions
- StreetViewLink
- StreetViewPov
- StreetViewPanoramaData
- StreetViewLocation

- StreetViewTileData
- StreetViewService
- StreetViewStatus
- StreetViewCoverageLayer

Events

- MapsEventListener
- event
- MouseEvent

Base

- LatLng
- LatLngLiteral
- LatLngBounds
- Point
- Size

MVC

- MVCObject
- MVCArray

Geometry Library

- encoding
- spherical
- poly

AdSense Library

- AdUnit
- AdUnitOptions
- AdFormat

Panoramio Library

- PanoramioLayer
- PanoramioLayerOptions
- PanoramioFeature
- PanoramioMouseEvent

Places Library

- Autocomplete
- AutocompleteOptions
- AutocompletePrediction
- PredictionTerm
- PredictionSubstring
- AutocompleteService
- AutocompletionRequest
- ComponentRestrictions
- PlaceAspectRating
- PlaceDetailsRequest
- PlaceGeometry
- PlacePhoto
- PhotoOptions
- PlaceResult
- PlaceReview
- PlaceSearchPagination
- PlaceSearchRequest
- PlacesService
- PlacesServiceStatus
- QueryAutocompletePrediction
- QueryAutocompletionRequest
- RadarSearchRequest

- RankBy
- SearchBox
- SearchBoxOptions
- TextSearchRequest

Drawing Library

- DrawingManager
- DrawingManagerOptions
- DrawingControlOptions
- OverlayCompleteEvent
- OverlayType

Weather Library

- CloudLayer
- WeatherLayer
- WeatherLayerOptions
- TemperatureUnit
- WindSpeedUnit
- LabelColor
- WeatherMouseEvent
- WeatherFeature
- WeatherConditions
- WeatherForecast

Visualization Library

- MapsEngineLayer
- MapsEngineLayerOptions
- MapsEngineLayerProperties
- MapsEngineMouseEvent
- MapsEngineStatus
- DynamicMapsEngineLayer
- DynamicMapsEngineLayerOptions
- DynamicMapsEngineMouseEvent
- FeatureStyle
- HeatmapLayer
- HeatmapLayerOptions
- WeightedLocation

google.maps.Map class

This class extends `MVCObject`.

Constructor

Constructor	Description
`Map(mapDiv:Node, opts?:`*`MapOptions`*`)`	Creates a new map inside of the given HTML container, which is typically a `DIV` element.

Methods

Methods	Return Value	Description
`fitBounds(bounds:LatLngBounds)`	`None`	Sets the viewport to contain the given bounds.
`getBounds()`	`LatLngBounds`	Returns the lat/lng bounds of the current viewport. If more than one copy of the world is visible, the bounds range in longitude from -180 to 180 degrees inclusive. If the map is not yet initialized (i.e. the mapType is still null), or center and zoom have not been set then the result is `null` or `undefined`.
`getCenter()`	`LatLng`	Returns the position displayed at the center of the map. Note that this `LatLng` object is *not* wrapped. See `LatLng` for more information.

getDiv()	Node	
getHeading()	number	Returns the compass heading of aerial imagery. The heading value is measured in degrees (clockwise) from cardinal direction North.
getMapTypeId()	MapTypeId|string	
getProjection()	Projection	Returns the current Projection. If the map is not yet initialized (i.e. the mapType is still null) then the result is null. Listen to projection_changed and check its value to ensure it is not null.
getStreetView()	StreetViewPanorama	Returns the default StreetViewPanorama bound to the map, which may be a default panorama embedded within the map, or the panorama set using setStreetView(). Changes to the map's streetViewControl will be reflected in the display of such a bound panorama.
getTilt()	number	Returns the current angle of incidence of the map, in degrees from the viewport plane to the map plane. The result will be 0 for imagery taken directly overhead or 45 for 45° imagery. 45° imagery is only available for SATELLITE and HYBRID map types, within some locations, and at some zoom levels. **Note:** This method does not return the value set by setTilt. See setTilt for details.
getZoom()	number	
panBy(x:number, y:number)	None	Changes the center of the map by the given distance in pixels. If the distance is less than both the width and height of the map, the transition will be smoothly animated. Note that the map coordinate system increases from west to east (for x values) and north to south (for y values).
panTo(latLng:LatLng|LatLngLiteral)	None	Changes the center of the map to the given LatLng. If the change is less than both the width and height of the map, the transition will be smoothly animated.
panToBounds(latLngBounds:LatLngBounds)	None	Pans the map by the minimum amount necessary to contain the given LatLngBounds. It makes no guarantee where on the map the bounds will be, except that as much of the bounds as possible will be visible. The bounds will be positioned inside the area bounded by the map type and navigation (pan, zoom, and Street View) controls, if they are present on the map. If the bounds is larger than the map, the map will be shifted to include the northwest corner of the bounds. If the change in the map's position is less than both the width and height of the map, the transition will be smoothly animated.
setCenter(latlng:LatLng|LatLngLiteral)	None	
setHeading(heading:number)	None	Sets the compass heading for aerial imagery measured in degrees from cardinal direction North.
setMapTypeId(mapTypeId:MapTypeId|string)	None	
setOptions(options:*MapOptions*)	None	
setStreetView(panorama:StreetViewPanorama)	None	Binds a StreetViewPanorama to the map. This panorama overrides the default StreetViewPanorama, allowing the map to bind to an external panorama outside of the map. Setting the panorama to null binds the default embedded panorama back to the map.
setTilt(tilt:number)	None	Controls the automatic switching behavior for the angle of incidence of the map. The only allowed values are 0 and 45. setTilt(0) causes the map to always use a 0° overhead view regardless of the zoom level and viewport. setTilt(45) causes the tilt angle to automatically switch to 45 whenever 45° imagery is available for the current zoom level and viewport, and switch back to 0 whenever 45° imagery is not available (this is the default behavior). 45° imagery is only available for

		`SATELLITE` and `HYBRID` map types, within some locations, and at some zoom levels. **Note:** `getTilt` returns the current tilt angle, not the value set by `setTilt`. Because `getTilt` and `setTilt` refer to different things, do not `bind()` the `tilt` property; doing so may yield unpredictable effects.
`setZoom(zoom:number)`	`None`	

Properties

Properties	Type	Description
`controls`	`Array<MVCArray<Node>>`	Additional controls to attach to the map. To add a control to the map, add the control's `<div>` to the `MVCArray` corresponding to the `ControlPosition` where it should be rendered.
`data`	`Data`	An instance of `Data`, bound to the map. Add features to this `Data` object to conveniently display them on this map.
`mapTypes`	`MapTypeRegistry`	A registry of `MapType` instances by string ID.
`overlayMapTypes`	`MVCArray<MapType>`	Additional map types to overlay.

Events

Events	Arguments	Description
`bounds_changed`	`None`	This event is fired when the viewport bounds have changed.
`center_changed`	`None`	This event is fired when the map center property changes.
`click`	*`MouseEvent`*	This event is fired when the user clicks on the map (but not when they click on a marker or infowindow).
`dblclick`	*`MouseEvent`*	This event is fired when the user double-clicks on the map. Note that the click event will also fire, right before this one.
`drag`	`None`	This event is repeatedly fired while the user drags the map.
`dragend`	`None`	This event is fired when the user stops dragging the map.
`dragstart`	`None`	This event is fired when the user starts dragging the map.
`heading_changed`	`None`	This event is fired when the map heading property changes.
`idle`	`None`	This event is fired when the map becomes idle after panning or zooming.
`maptypeid_changed`	`None`	This event is fired when the mapTypeId property changes.
`mousemove`	*`MouseEvent`*	This event is fired whenever the user's mouse moves over the map container.
`mouseout`	*`MouseEvent`*	This event is fired when the user's mouse exits the map container.
`mouseover`	*`MouseEvent`*	This event is fired when the user's mouse enters the map container.
`projection_changed`	`None`	This event is fired when the projection has changed.
`resize`	`None`	Developers should trigger this event on the map when the div changes size: `google.maps.event.trigger(map, 'resize')` .
`rightclick`	*`MouseEvent`*	This event is fired when the DOM contextmenu event is fired on the map container.
`tilesloaded`	`None`	This event is fired when the visible tiles have finished loading.
`tilt_changed`	`None`	This event is fired when the map tilt property changes.
`zoom_changed`	`None`	This event is fired when the map zoom property changes.

google.maps.MapOptions object specification

Properties

Properties	Type	Description
`backgroundColor`	`string`	Color used for the background of the Map div. This color will be visible when tiles have not yet loaded as the user pans. This option can only be set when

		the map is initialized.
`center`	`LatLng`	The initial Map center. Required.
`disableDefaultUI`	`boolean`	Enables/disables all default UI. May be overridden individually.
`disableDoubleClickZoom`	`boolean`	Enables/disables zoom and center on double click. Enabled by default.
`draggable`	`boolean`	If false, prevents the map from being dragged. Dragging is enabled by default.
`draggableCursor`	`string`	The name or url of the cursor to display when mousing over a draggable map. This property uses the css `cursor` attribute to change the icon. As with the css property, you must specify at least one fallback cursor that is not a URL. For example: `draggableCursor: 'url(http://www.example.com/icon.png), auto;'`.
`draggingCursor`	`string`	The name or url of the cursor to display when the map is being dragged. This property uses the css `cursor` attribute to change the icon. As with the css property, you must specify at least one fallback cursor that is not a URL. For example: `draggingCursor: 'url(http://www.example.com/icon.png), auto;'`.
`heading`	`number`	The heading for aerial imagery in degrees measured clockwise from cardinal direction North. Headings are snapped to the nearest available angle for which imagery is available.
`keyboardShortcuts`	`boolean`	If false, prevents the map from being controlled by the keyboard. Keyboard shortcuts are enabled by default.
`mapMaker`	`boolean`	True if Map Maker tiles should be used instead of regular tiles.
`mapTypeControl`	`boolean`	The initial enabled/disabled state of the Map type control.
`mapTypeControlOptions`	`MapTypeControlOptions`	The initial display options for the Map type control.
`mapTypeId`	`MapTypeId`	The initial Map mapTypeId. Defaults to `ROADMAP`.
`maxZoom`	`number`	The maximum zoom level which will be displayed on the map. If omitted, or set to null, the maximum zoom from the current map type is used instead.
`minZoom`	`number`	The minimum zoom level which will be displayed on the map. If omitted, or set to null, the minimum zoom from the current map type is used instead.
`noClear`	`boolean`	If true, do not clear the contents of the Map div.
`overviewMapControl`	`boolean`	The enabled/disabled state of the Overview Map control.
`overviewMapControlOptions`	`OverviewMapControlOptions`	The display options for the Overview Map control.
`panControl`	`boolean`	The enabled/disabled state of the Pan control.
`panControlOptions`	`PanControlOptions`	The display options for the Pan control.
`rotateControl`	`boolean`	The enabled/disabled state of the Rotate control.
`rotateControlOptions`	`RotateControlOptions`	The display options for the Rotate control.
`scaleControl`	`boolean`	The initial enabled/disabled state of the Scale control.
`scaleControlOptions`	`ScaleControlOptions`	The initial display options for the Scale control.
`scrollwheel`	`boolean`	If false, disables scrollwheel zooming on the map. The scrollwheel is enabled by default.
`streetView`	`StreetViewPanorama`	A `StreetViewPanorama` to display when the Street View pegman is dropped on the map. If no panorama is specified, a default `StreetViewPanorama` will be displayed in the map's `div` when the pegman is dropped.
`streetViewControl`	`boolean`	The initial enabled/disabled state of the Street View Pegman control. This control is part of the default UI, and should be set to `false` when displaying a map type on which the Street View road overlay should not appear (e.g. a non-Earth map type).
`streetViewControlOptions`	`StreetViewControlOptions`	The initial display options for the Street View Pegman control.
`styles`	`Array<MapTypeStyle>`	Styles to apply to each of the default map types. Note that for Satellite/Hybrid and Terrain modes, these styles will only apply to labels and

		geometry.
`tilt`	`number`	Controls the automatic switching behavior for the angle of incidence of the map. The only allowed values are `0` and `45`. The value `0` causes the map to always use a 0° overhead view regardless of the zoom level and viewport. The value `45` causes the tilt angle to automatically switch to 45 whenever 45° imagery is available for the current zoom level and viewport, and switch back to 0 whenever 45° imagery is not available (this is the default behavior). 45° imagery is only available for `SATELLITE` and `HYBRID` map types, within some locations, and at some zoom levels. **Note:** `getTilt` returns the current tilt angle, not the value specified by this option. Because `getTilt` and this option refer to different things, do not `bind()` the `tilt` property; doing so may yield unpredictable effects.
`zoom`	`number`	The initial Map zoom level. Required.
`zoomControl`	`boolean`	The enabled/disabled state of the Zoom control.
`zoomControlOptions`	`ZoomControlOptions`	The display options for the Zoom control.

google.maps.MapTypeId class

Identifiers for common MapTypes.

Constant

Constant	Description
`HYBRID`	This map type displays a transparent layer of major streets on satellite images.
`ROADMAP`	This map type displays a normal street map.
`SATELLITE`	This map type displays satellite images.
`TERRAIN`	This map type displays maps with physical features such as terrain and vegetation.

google.maps.MapTypeControlOptions object specification

Options for the rendering of the map type control.

Properties

Properties	Type	Description
`mapTypeIds`	`Array<MapTypeId>|Array<string>`	IDs of map types to show in the control.
`position`	`ControlPosition`	Position id. Used to specify the position of the control on the map. The default position is `TOP_RIGHT`.
`style`	`MapTypeControlStyle`	Style id. Used to select what style of map type control to display.

google.maps.MapTypeControlStyle class

Identifiers for common MapTypesControls.

Constant

Constant	Description
`DEFAULT`	Uses the default map type control. The control which DEFAULT maps to will vary according to window size and other factors. It may change in future versions of the API.
`DROPDOWN_MENU`	A dropdown menu for the screen realestate conscious.
`HORIZONTAL_BAR`	The standard horizontal radio buttons bar.

google.maps.OverviewMapControlOptions object specification

Options for the rendering of the Overview Map control.

Properties

Properties	Type	Description
`opened`	`boolean`	Whether the control should display in opened mode or collapsed (minimized) mode. By default, the control is closed.

google.maps.PanControlOptions object specification

Options for the rendering of the pan control.

Properties

Properties	Type	Description
`position`	`ControlPosition`	Position id. Used to specify the position of the control on the map. The default position is `TOP_LEFT`.

google.maps.RotateControlOptions object specification

Options for the rendering of the rotate control.

Properties

Properties	Type	Description
`position`	`ControlPosition`	Position id. Used to specify the position of the control on the map. The default position is `TOP_LEFT`.

google.maps.ScaleControlOptions object specification

Options for the rendering of the scale control.

Properties

Properties	Type	Description
`style`	`ScaleControlStyle`	Style id. Used to select what style of scale control to display.

google.maps.ScaleControlStyle class

Identifiers for scale control ids.

Constant

Constant	Description
`DEFAULT`	The standard scale control.

google.maps.StreetViewControlOptions object specification

Options for the rendering of the Street View pegman control on the map.

Properties

Properties	Type	Description
`position`	`ControlPosition`	Position id. Used to specify the position of the control on the map. The default position is embedded within the navigation (zoom and pan) controls. If this position is empty or the same as that specified in the `zoomControlOptions` or `panControlOptions`, the Street View control will be displayed as part of the navigation controls. Otherwise, it will be displayed separately.

google.maps.ZoomControlOptions object specification

Options for the rendering of the zoom control.

Properties

Properties	Type	Description
position	ControlPosition	Position id. Used to specify the position of the control on the map. The default position is TOP_LEFT.
style	ZoomControlStyle	Style id. Used to select what style of zoom control to display.

google.maps.ZoomControlStyle class

Identifiers for the zoom control.

Constant

Constant	Description
DEFAULT	The default zoom control. The control which DEFAULT maps to will vary according to map size and other factors. It may change in future versions of the API.
LARGE	The larger control, with the zoom slider in addition to +/- buttons.
SMALL	A small control with buttons to zoom in and out.

google.maps.ControlPosition class

Identifiers used to specify the placement of controls on the map. Controls are positioned relative to other controls in the same layout position. Controls that are added first are positioned closer to the edge of the map.

```
  +----------------+
  + TL    TC    TR +
  + LT          RT +
  +               +
  + LC          RC +
  +               +
  + LB          RB +
  + BL    BC    BR +
  +----------------+
```

Elements in the top or bottom row flow towards the middle of the row. Elements in the left or right column flow towards the middle of the column.

Constant

Constant	Description
BOTTOM_CENTER	Elements are positioned in the center of the bottom row.
BOTTOM_LEFT	Elements are positioned in the bottom left and flow towards the middle. Elements are positioned to the right of the Google logo.
BOTTOM_RIGHT	Elements are positioned in the bottom right and flow towards the middle. Elements are positioned to the left of the copyrights.
LEFT_BOTTOM	Elements are positioned on the left, above bottom-left elements, and flow upwards.
LEFT_CENTER	Elements are positioned in the center of the left side.
LEFT_TOP	Elements are positioned on the left, below top-left elements, and flow downwards.
RIGHT_BOTTOM	Elements are positioned on the right, above bottom-right elements, and flow upwards.
RIGHT_CENTER	Elements are positioned in the center of the right side.
RIGHT_TOP	Elements are positioned on the right, below top-right elements, and flow downwards.
TOP_CENTER	Elements are positioned in the center of the top row.
TOP_LEFT	Elements are positioned in the top left and flow towards the middle.

`TOP_RIGHT`	Elements are positioned in the top right and flow towards the middle.

google.maps.Data class

A layer for displaying geospatial data. Points, line-strings and polygons can be displayed.

Every `Map` has a `Data` object by default, so most of the time there is no need to construct one. For example:

```
var myMap = new google.maps.Map(...);
myMap.data.addGeoJson(...);
myMap.data.setStyle(...);
```

The `Data` object is a collection of `Features`.

This class extends `MVCObject`.

Constructor

Constructor	Description
`Data(options?:Data.DataOptions)`	Creates an empty collection, with the given `DataOptions`.

Methods

Methods	Return Value	Description
`add(feature:Data.Feature|Data.FeatureOptions)`	`Data.Feature`	Adds a feature to the collection, and returns the added feature. If the feature has an ID, it will replace any existing feature in the collection with the same ID. If no feature is given, a new feature will be created with null geometry and no properties. If `FeatureOptions` are given, a new feature will be created with the specified properties. Note that the IDs `1234` and `'1234'` are equivalent. Adding a feature with ID `1234` will replace a feature with ID `'1234'`, and vice versa.
`addGeoJson(geoJson:Object, options?:Data.GeoJsonOptions)`	`Array<Data.Feature>`	Adds GeoJSON features to the collection. Give this method a parsed JSON. The imported features are returned. Throws an exception if the GeoJSON could not be imported.
`contains(feature:Data.Feature)`	`boolean`	Checks whether the given feature is in the collection.
`forEach(callback:function(Data.Feature))`	`None`	Repeatedly invokes the

		given function, passing a feature in the collection to the function on each invocation. The order of iteration through the features is undefined.
`getFeatureById(id:number|string)`	`Data.Feature|undefined`	Returns the feature with the given ID, if it exists in the collection. Otherwise returns `undefined`. Note that the IDs `1234` and `'1234'` are equivalent. Either can be used to look up the same feature.
`getMap()`	`Map`	Returns the map on which the features are displayed.
`getStyle()`	`Data.StylingFunction|Data.StyleOptions`	Gets the style for all features in the collection.
`loadGeoJson(url:string, options?:Data.GeoJsonOptions, callback?:function(Array<Data.Feature>))`	`None`	Loads GeoJSON from a URL, and adds the features to the collection.
`overrideStyle(feature:Data.Feature, style:Data.StyleOptions)`	`None`	Changes the style of a feature. These changes are applied on top of the style specified by `setStyle()`. Style properties set to null revert to the value specified via `setStyle()`.
`remove(feature:Data.Feature)`	`None`	Removes a feature from the collection.
`revertStyle(feature?:Data.Feature)`	`None`	Removes the effect of previous `overrideStyle()` calls. The style of the given feature reverts to the style specified by `setStyle()`. If no feature is given, all features have their style reverted.
`setMap(map:Map)`	`None`	Renders the features on the specified map. If map is set to null, the features will be removed from the map.
`setStyle(style:Data.StylingFunction|Data.StyleOptions)`	`None`	Sets the style for all features in the collection. Styles specified on a per-

		feature basis via `overrideStyle()` continue to apply. Pass either an object with the desired style options, or a function that computes the style for each feature. The function will be called every time a feature's properties are updated.
`toGeoJson(callback:function(Object))`	`None`	Exports the features in the collection to a GeoJSON object.

Events

Events	Arguments	Description
`addfeature`	*`Data.AddFeatureEvent`*	This event is fired when a feature is added to the collection.
`click`	*`Data.MouseEvent`*	This event is fired for a click on the geometry.
`dblclick`	*`Data.MouseEvent`*	This event is fired for a double click on the geometry.
`mousedown`	*`Data.MouseEvent`*	This event is fired for a mousedown on the geometry.
`mouseout`	*`Data.MouseEvent`*	This event is fired when the mouse leaves the area of the geometry.
`mouseover`	*`Data.MouseEvent`*	This event is fired when the mouse enters the area of the geometry.
`mouseup`	*`Data.MouseEvent`*	This event is fired for a mouseup on the geometry.
`removefeature`	*`Data.RemoveFeatureEvent`*	This event is fired when a feature is removed from the collection.
`removeproperty`	*`Data.RemovePropertyEvent`*	This event is fired when a feature's property is removed.
`rightclick`	*`Data.MouseEvent`*	This event is fired for a rightclick on the geometry.
`setgeometry`	*`Data.SetGeometryEvent`*	This event is fired when a feature's geometry is set.
`setproperty`	*`Data.SetPropertyEvent`*	This event is fired when a feature's property is set.

google.maps.Data.DataOptions object specification

DataOptions object used to define the properties that a developer can set on a `Data` object.

Properties

Properties	Type	Description
`map`	`Map`	Map on which to display the features in the collection.
`style`	`Data.StylingFunction`|`Data.StyleOptions`	Style for all features in the collection. For more details, see the `setStyle()` method above.

google.maps.Data.GeoJsonOptions object specification

Optional parameters for importing GeoJSON.

Properties

Properties	Type	Description
`idPropertyName`	`string`	The name of the Feature property to use as the feature ID. If not specified, the GeoJSON Feature id will be used.

google.maps.Data.StyleOptions object specification

These options specify the way a Feature should appear when displayed on a map.

Properties

Properties	Type	Description
`clickable`	`boolean`	If `true`, the marker receives mouse and touch events. Default value is `true`.
`cursor`	`string`	Mouse cursor to show on hover. Only applies to point geometries.
`fillColor`	`string`	The fill color. All CSS3 colors are supported except for extended named colors. Only applies to polygon geometries.
`fillOpacity`	`number`	The fill opacity between 0.0 and 1.0. Only applies to polygon geometries.
`icon`	`string|Icon|Symbol`	Icon for the foreground. If a string is provided, it is treated as though it were an `Icon` with the string as `url`. Only applies to point geometries.
`shape`	`MarkerShape`	Defines the image map used for hit detection. Only applies to point geometries.
`strokeColor`	`string`	The stroke color. All CSS3 colors are supported except for extended named colors. Only applies to line and polygon geometries.
`strokeOpacity`	`number`	The stroke opacity between 0.0 and 1.0. Only applies to line and polygon geometries.
`strokeWeight`	`number`	The stroke width in pixels. Only applies to line and polygon geometries.
`title`	`string`	Rollover text. Only applies to point geometries.
`visible`	`boolean`	Whether the feature is visible. Defaults to `true`.
`zIndex`	`number`	All features are displayed on the map in order of their zIndex, with higher values displaying in front of features with lower values. Markers are always displayed in front of line-strings and polygons.

google.maps.Data.StylingFunction typedef

A function that computes the appearance of a feature.

The `Data.setStyle()` method can accept a styling function. Use this when features should appear differently depending on their properties. You can find more information about styling features in the developer's guide.

```
function(Data.Feature):Data.StyleOptions
```

google.maps.Data.Feature class

A feature has a geometry, an id, and a set of properties.

Constructor

Constructor	Description
`Data.Feature(options?:Data.FeatureOptions)`	Constructs a Feature with the given options.

Methods

Methods	Return Value	Description
`forEachProperty(callback:function(*, string))`	`None`	Repeatedly invokes the given function, passing a property value and name on each invocation. The order of iteration through the properties is undefined.
`getGeometry()`	`Data.Geometry`	Returns the feature's geometry.
`getId()`	`number|string|undefined`	Returns the feature ID.
`getProperty(name:string)`	`*`	Returns the value of the requested property, or undefined if the property

		does not exist.
`removeProperty(name:string)`	`None`	Removes the property with the given name.
`setGeometry(newGeometry:Data.Geometry|LatLng|LatLngLiteral)`	`None`	Sets the feature's geometry.
`setProperty(name:string, newValue:*)`	`None`	Sets the value of the specified property. If `newValue` is `undefined` this is equivalent to calling `removeProperty`.
`toGeoJson(callback:function(Object))`	`None`	Exports the feature to a GeoJSON object.

Events

Events	Arguments	Description
`removeproperty`	`Data.RemovePropertyEvent`	This event is triggered when a feature's property is removed.
`setgeometry`	`Data.SetGeometryEvent`	This event is triggered when a feature's geometry is set.
`setproperty`	`Data.SetPropertyEvent`	This event is triggered when a feature's property is set.

google.maps.Data.FeatureOptions object specification

Optional parameters for creating `Data.Feature` objects.

Properties

Properties	Type	Description
`geometry`	`Data.Geometry|LatLng|LatLngLiteral`	The feature geometry. If none is specified when a feature is constructed, the feature's geometry will be null. If a `LatLng` object or `LatLngLiteral` is given, this will be converted to a `Data.Point` geometry.
`id`	`number|string`	Feature ID is optional. If provided, it can be used to look up the feature in a `Data` object using the `getFeatureById()` method. Note that a feature's ID cannot be subsequently changed.
`properties`	`Object`	The feature properties. This is an arbitrary mapping of property names to values.

google.maps.Data.Geometry class

A superclass for the various geometry objects.

Methods

Methods	Return Value	Description
`getType()`	`string`	Returns the type of the geometry object. Possibilities are `"Point"`, `"MultiPoint"`, `"LineString"`, `"MultiLineString"`, `"LinearRing"`, `"Polygon"`, `"MultiPolygon"`, or `"GeometryCollection"`.

google.maps.Data.Point class

A Point geometry contains a single `LatLng`.

This class extends `Data.Geometry`.

Constructor

Constructor	Description
`Data.Point(latLng:LatLng|LatLngLiteral)`	Constructs a `Data.Point` from the given `LatLng` or `LatLngLiteral`.

Methods

Methods	Return Value	Description
`get()`	`LatLng`	Returns the contained `LatLng`.
`getType()`	`string`	Returns the string `"Point"`.

google.maps.Data.MultiPoint class

A MultiPoint geometry contains a number of `LatLng`s.

This class extends `Data.Geometry`.

Constructor

Constructor	Description
`Data.MultiPoint(elements:Array<LatLng|LatLngLiteral>)`	Constructs a `Data.MultiPoint` from the given `LatLng`s or `LatLngLiteral`s.

Methods

Methods	Return Value	Description
`getArray()`	`Array<LatLng>`	Returns an array of the contained `LatLng`s. A new array is returned each time `getArray()` is called.
`getAt(n:number)`	`LatLng`	Returns the n-th contained `LatLng`.
`getLength()`	`number`	Returns the number of contained `LatLng`s.
`getType()`	`string`	Returns the string `"MultiPoint"`.

google.maps.Data.LineString class

A LineString geometry contains a number of `LatLng`s.

This class extends `Data.Geometry`.

Constructor

Constructor	Description
`Data.LineString(elements:Array<LatLng|LatLngLiteral>)`	Constructs a `Data.LineString` from the given `LatLng`s or `LatLngLiteral`s.

Methods

Methods	Return Value	Description
`getArray()`	`Array<LatLng>`	Returns an array of the contained `LatLng`s. A new array is returned each time `getArray()` is called.
`getAt(n:number)`	`LatLng`	Returns the n-th contained `LatLng`.
`getLength()`	`number`	Returns the number of contained `LatLng`s.
`getType()`	`string`	Returns the string `"LineString"`.

google.maps.Data.MultiLineString class

A MultiLineString geometry contains a number of `LineString`s.

This class extends `Data.Geometry`.

Constructor

Constructor	Description
`Data.MultiLineString(elements:Array<Data.LineString|Array<LatLng|LatLngLiteral>>)`	Constructs a `Data.MultiLineString` from the given `Data.LineString`s or arrays of positions.

Methods

Methods	Return Value	Description
`getArray()`	`Array<Data.LineString>`	Returns an array of the contained `Data.LineString`s. A new array is returned each time `getArray()` is called.
`getAt(n:number)`	`Data.LineString`	Returns the `n`-th contained `Data.LineString`.
`getLength()`	`number`	Returns the number of contained `Data.LineString`s.
`getType()`	`string`	Returns the string `"MultiLineString"`.

google.maps.Data.LinearRing class

A LinearRing geometry contains a number of `LatLng`s, representing a closed LineString. There is no need to make the first `LatLng` equal to the last `LatLng`. The LinearRing is closed implicitly.

This class extends `Data.Geometry`.

Constructor

Constructor	Description
`Data.LinearRing(elements:Array<LatLng|LatLngLiteral>)`	Constructs a `Data.LinearRing` from the given `LatLng`s or `LatLngLiteral`s.

Methods

Methods	Return Value	Description
`getArray()`	`Array<LatLng>`	Returns an array of the contained `LatLng`s. A new array is returned each time `getArray()` is called.
`getAt(n:number)`	`LatLng`	Returns the `n`-th contained `LatLng`.
`getLength()`	`number`	Returns the number of contained `LatLng`s.
`getType()`	`string`	Returns the string `"LinearRing"`.

google.maps.Data.Polygon class

A Polygon geometry contains a number of `Data.LinearRing`s. The first linear-ring must be the polygon exterior boundary, and subsequent linear-rings must be interior boundaries or "holes".

This class extends `Data.Geometry`.

Constructor

Constructor	Description
`Data.Polygon(elements:Array<Data.LinearRing|Array<LatLng|LatLngLiteral>>)`	Constructs a `Data.Polygon` from the given `Data.LinearRing`s or arrays of positions.

Methods

Methods	Return Value	Description
`getArray()`	`Array<Data.LinearRing>`	Returns an array of the contained `Data.LinearRing`s. A new array is returned each time `getArray()` is called.
`getAt(n:number)`	`Data.LinearRing`	Returns the `n`-th contained `Data.LinearRing`.

`getLength()`	`number`	Returns the number of contained `Data.LinearRing`s.
`getType()`	`string`	Returns the string `"Polygon"`.

google.maps.Data.MultiPolygon class

A MultiPolygon geometry contains a number of `Data.Polygon`s.

This class extends `Data.Geometry`.

Constructor

Constructor	Description
`Data.MultiPolygon(elements:Array<Data.Polygon|Array<Data.LinearRing|Array<LatLng|LatLngLiteral>>>)`	Constructs a `Data.MultiPolygon` from the given `Data.Polygon`s or arrays of positions.

Methods

Methods	Return Value	Description
`getArray()`	`Array<Data.Polygon>`	Returns an array of the contained `Data.Polygon`s. A new array is returned each time `getArray()` is called.
`getAt(n:number)`	`Data.Polygon`	Returns the n-th contained `Data.Polygon`.
`getLength()`	`number`	Returns the number of contained `Data.Polygon`s.
`getType()`	`string`	Returns the string `"MultiPolygon"`.

google.maps.Data.GeometryCollection class

A GeometryCollection contains a number of geometry objects. Any `LatLng` or `LatLngLiteral` objects are automatically converted to `Data.Point` geometry objects.

This class extends `Data.Geometry`.

Constructor

Constructor	Description
`Data.GeometryCollection(elements:Array<Data.Geometry|LatLng|LatLngLiteral>)`	Constructs a `Data.GeometryCollection` from the given geometry objects or `LatLng`s.

Methods

Methods	Return Value	Description
`getArray()`	`Array<Data.Geometry>`	Returns an array of the contained geometry objects. A new array is returned each time `getArray()` is called.
`getAt(n:number)`	`Data.Geometry`	Returns the n-th contained geometry object.
`getLength()`	`number`	Returns the number of contained geometry objects.
`getType()`	`string`	Returns the string `"GeometryCollection"`.

google.maps.Data.MouseEvent object specification

This object is passed to mouse event handlers on a `Data` object.

This object extends `MouseEvent`.

Properties

Properties	Type	Description
`feature`	`Data.Feature`	The feature which generated the mouse event.

google.maps.Data.AddFeatureEvent object specification

The properties of a `addfeature` event.

Properties

Properties	Type	Description
`feature`	`Data.Feature`	The feature that was added to the `FeatureCollection`.

google.maps.Data.RemoveFeatureEvent object specification

The properties of a `removefeature` event.

Properties

Properties	Type	Description
`feature`	`Data.Feature`	The feature that was removed from the `FeatureCollection`.

google.maps.Data.SetGeometryEvent object specification

The properties of a `setgeometry` event.

Properties

Properties	Type	Description
`feature`	`Data.Feature`	The feature whose geometry was set.
`newGeometry`	`Data.Geometry`	The new feature geometry.
`oldGeometry`	`Data.Geometry`	The previous feature geometry.

google.maps.Data.SetPropertyEvent object specification

The properties of a `setproperty` event.

Properties

Properties	Type	Description
`feature`	`Data.Feature`	The feature whose property was set.
`name`	`string`	The property name.
`newValue`	`*`	The new value.
`oldValue`	`*`	The previous value. Will be `undefined` if the property was added.

google.maps.Data.RemovePropertyEvent object specification

The properties of a `removeproperty` event.

Properties

Properties	Type	Description

feature	Data.Feature	The feature whose property was removed.
name	string	The property name.
oldValue	*	The previous value.

google.maps.Marker class

This class extends MVCObject.

Constructor

Constructor	Description
Marker(opts?:*MarkerOptions*)	Creates a marker with the options specified. If a map is specified, the marker is added to the map upon construction. Note that the position must be set for the marker to display.

Methods

Methods	Return Value	Description
getAnimation()	Animation	
getAttribution()	Attribution	
getClickable()	boolean	
getCursor()	string	
getDraggable()	boolean	
getIcon()	string|Icon|Symbol	
getMap()	Map|StreetViewPanorama	
getOpacity()	number	
getPlace()	Place	
getPosition()	LatLng	
getShape()	MarkerShape	
getTitle()	string	
getVisible()	boolean	
getZIndex()	number	
setAnimation(animation:Animation)	None	Start an animation. Any ongoing animation will be cancelled. Currently supported animations are: BOUNCE, DROP. Passing in null will cause any animation to stop.
setAttribution(attribution:*Attribution*)	None	
setClickable(flag:boolean)	None	
setCursor(cursor:string)	None	
setDraggable(flag:boolean)	None	
setIcon(icon:string|Icon|Symbol)	None	
setMap(map:Map|StreetViewPanorama)	None	Renders the marker on the specified map or panorama. If map is set to null, the marker will be removed.
setOpacity(opacity:number)	None	
setOptions(options:*MarkerOptions*)	None	
setPlace(place:*Place*)	None	
setPosition(latlng:LatLng|LatLngLiteral)	None	
setShape(shape:*MarkerShape*)	None	
setTitle(title:string)	None	

setVisible(visible:boolean)	None	
setZIndex(zIndex:number)	None	

Constant

Constant	Description
MAX_ZINDEX	The maximum default z-index that the API will assign to a marker. You may set a higher z-index to bring a marker to the front.

Events

Events	Arguments	Description
animation_changed	None	This event is fired when the marker's animation property changes.
click	*MouseEvent*	This event is fired when the marker icon was clicked.
clickable_changed	None	This event is fired when the marker's clickable property changes.
cursor_changed	None	This event is fired when the marker's cursor property changes.
dblclick	*MouseEvent*	This event is fired when the marker icon was double clicked.
drag	*MouseEvent*	This event is repeatedly fired while the user drags the marker.
dragend	*MouseEvent*	This event is fired when the user stops dragging the marker.
draggable_changed	None	This event is fired when the marker's draggable property changes.
dragstart	*MouseEvent*	This event is fired when the user starts dragging the marker.
flat_changed	None	This event is fired when the marker's flat property changes.
icon_changed	None	This event is fired when the marker icon property changes.
mousedown	*MouseEvent*	This event is fired for a mousedown on the marker.
mouseout	*MouseEvent*	This event is fired when the mouse leaves the area of the marker icon.
mouseover	*MouseEvent*	This event is fired when the mouse enters the area of the marker icon.
mouseup	*MouseEvent*	This event is fired for a mouseup on the marker.
position_changed	None	This event is fired when the marker position property changes.
rightclick	*MouseEvent*	This event is fired for a rightclick on the marker.
shape_changed	None	This event is fired when the marker's shape property changes.
title_changed	None	This event is fired when the marker title property changes.
visible_changed	None	This event is fired when the marker's visible property changes.
zindex_changed	None	This event is fired when the marker's zIndex property changes.

google.maps.MarkerOptions object specification

Properties

Properties	Type	Description
anchorPoint	Point	The offset from the marker's position to the tip of an InfoWindow that has been opened with the marker as anchor.
animation	Animation	Which animation to play when marker is added to a map.
attribution	Attribution	Contains all the information needed to identify your application as the source of a save. In this context, 'place' means a business, point of interest or geographic location. attribution must be specified with a place in order to enable a save.
clickable	boolean	If true, the marker receives mouse and touch events. Default value is true.
crossOnDrag	boolean	If false, disables cross that appears beneath the marker when dragging. This option is true by default.

cursor	string	Mouse cursor to show on hover
draggable	boolean	If true, the marker can be dragged. Default value is false.
icon	string|Icon|Symbol	Icon for the foreground. If a string is provided, it is treated as though it were an Icon with the string as url.
map	Map|StreetViewPanorama	Map on which to display Marker.
opacity	number	The marker's opacity between 0.0 and 1.0.
optimized	boolean	Optimization renders many markers as a single static element. Optimized rendering is enabled by default. Disable optimized rendering for animated GIFs or PNGs, or when each marker must be rendered as a separate DOM element (advanced usage only).
place	Place	Place information, used to identify and describe the place associated with this Marker. In this context, 'place' means a business, point of interest or geographic location. To allow a user to save this place, open an info window anchored on this marker. The info window will contain information about the place and an option for the user to save it. Only one of position or place can be specified.
position	LatLng	Marker position. Required.
shape	MarkerShape	Image map region definition used for drag/click.
title	string	Rollover text
visible	boolean	If true, the marker is visible
zIndex	number	All markers are displayed on the map in order of their zIndex, with higher values displaying in front of markers with lower values. By default, markers are displayed according to their vertical position on screen, with lower markers appearing in front of markers further up the screen.

google.maps.Icon object specification

Properties

Properties	Type	Description
anchor	Point	The position at which to anchor an image in correspondence to the location of the marker on the map. By default, the anchor is located along the center point of the bottom of the image.
origin	Point	The position of the image within a sprite, if any. By default, the origin is located at the top left corner of the image (0, 0).
scaledSize	Size	The size of the entire image after scaling, if any. Use this property to stretch/shrink an image or a sprite.
size	Size	The display size of the sprite or image. When using sprites, you must specify the sprite size. If the size is not provided, it will be set when the image loads.
url	string	The URL of the image or sprite sheet.

google.maps.MarkerShape object specification

This object defines the clickable region of a marker image for browsers other than Internet Explorer. The shape consists of two properties — type and coord — which define the non-transparent region of an image. A MarkerShape object is not required on Internet Explorer since the browser does not fire events on the transparent region of an image by default.

Properties

Properties	Type	Description
coords	Array<number>	The format of this attribute depends on the value of the type and follows the w3 AREA coords specification found at http://www.w3.org/TR/REC-html40/struct/objects.html#adef-coords. The coords attribute is an array of integers that specify the pixel position of the shape relative to the top-left corner of the target image. The coordinates depend on the value of type as follows: - circle: coords is [x1,y1,r] where x1,y2 are the coordinates of the center of the circle, and r is the radius of the circle. - poly: coords is [x1,y1,x2,y2...xn,yn] where each x,y pair contains the coordinates of one vertex of the

		polygon. - `rect`: coords is `[x1,y1,x2,y2]` where x1,y1 are the coordinates of the upper-left corner of the rectangle and x2,y2 are the coordinates of the lower-right coordinates of the rectangle.
`type`	`string`	Describes the shape's type and can be `circle`, `poly` or `rect`.

google.maps.Symbol object specification

Properties

Properties	Type	Description
`anchor`	`Point`	The position of the symbol relative to the marker or polyline. The coordinates of the symbol's path are translated left and up by the anchor's x and y coordinates respectively. By default, a symbol is anchored at `(0, 0)`. The position is expressed in the same coordinate system as the symbol's path.
`fillColor`	`string`	The symbol's fill color. All CSS3 colors are supported except for extended named colors. For symbol markers, this defaults to 'black'. For symbols on polylines, this defaults to the stroke color of the corresponding polyline.
`fillOpacity`	`number`	The symbol's fill opacity. Defaults to 0.
`path`	`SymbolPath|string`	The symbol's path, which is a built-in symbol path, or a custom path expressed using SVG path notation. Required.
`rotation`	`number`	The angle by which to rotate the symbol, expressed clockwise in degrees. Defaults to 0. A symbol in an `IconSequence` where `fixedRotation` is `false` is rotated relative to the angle of the edge on which it lies.
`scale`	`number`	The amount by which the symbol is scaled in size. For symbol markers, this defaults to 1; after scaling, the symbol may be of any size. For symbols on a polyline, this defaults to the stroke weight of the polyline; after scaling, the symbol must lie inside a square 22 pixels in size centered at the symbol's anchor.
`strokeColor`	`string`	The symbol's stroke color. All CSS3 colors are supported except for extended named colors. For symbol markers, this defaults to 'black'. For symbols on a polyline, this defaults to the stroke color of the polyline.
`strokeOpacity`	`number`	The symbol's stroke opacity. For symbol markers, this defaults to 1. For symbols on a polyline, this defaults to the stroke opacity of the polyline.
`strokeWeight`	`number`	The symbol's stroke weight. Defaults to the `scale` of the symbol.

google.maps.SymbolPath class

Built-in symbol paths.

Constant

Constant	Description
`BACKWARD_CLOSED_ARROW`	A backward-pointing closed arrow.
`BACKWARD_OPEN_ARROW`	A backward-pointing open arrow.
`CIRCLE`	A circle.
`FORWARD_CLOSED_ARROW`	A forward-pointing closed arrow.
`FORWARD_OPEN_ARROW`	A forward-pointing open arrow.

google.maps.Animation class

Animations that can be played on a marker. Use the `setAnimation` method on Marker or the `animation` option to play an animation.

Constant

Constant	Description

`BOUNCE`	Marker bounces until animation is stopped.
`DROP`	Marker falls from the top of the map ending with a small bounce.

google.maps.InfoWindow class

An overlay that looks like a bubble and is often connected to a marker.

This class extends `MVCObject`.

Constructor

Constructor	Description
`InfoWindow(opts?:InfoWindowOptions)`	Creates an info window with the given options. An InfoWindow can be placed on a map at a particular position or above a marker, depending on what is specified in the options. Unless auto-pan is disabled, an InfoWindow will pan the map to make itself visible when it is opened. After constructing an InfoWindow, you must call open to display it on the map. The user can click the close button on the InfoWindow to remove it from the map, or the developer can call close() for the same effect.

Methods

Methods	Return Value	Description
`close()`	`None`	Closes this InfoWindow by removing it from the DOM structure.
`getContent()`	`string|Node`	
`getPosition()`	`LatLng`	
`getZIndex()`	`number`	
`open(map?:Map|StreetViewPanorama, anchor?:MVCObject)`	`None`	Opens this InfoWindow on the given map. Optionally, an InfoWindow can be associated with an anchor. In the core API, the only anchor is the Marker class. However, an anchor can be any MVCObject that exposes a LatLng `position` property and optionally a Point `anchorPoint` property for calculating the `pixelOffset` (see InfoWindowOptions). The `anchorPoint` is the offset from the anchor's position to the tip of the InfoWindow.
`setContent(content:string|Node)`	`None`	
`setOptions(options:InfoWindowOptions)`	`None`	
`setPosition(position:LatLng)`	`None`	
`setZIndex(zIndex:number)`	`None`	

Events

Events	Arguments	Description
`closeclick`	`None`	This event is fired when the close button was clicked.
`content_changed`	`None`	This event is fired when the content property changes.
`domready`	`None`	This event is fired when the `<div>` containing the `InfoWindow`'s content is attached to the DOM. You may wish to monitor this event if you are building out your info window content dynamically.
`position_changed`	`None`	This event is fired when the position property changes.
`zindex_changed`	`None`	This event is fired when the InfoWindow's zIndex changes.

google.maps.InfoWindowOptions object specification

Properties

Properties	Type	Description

content	string|Node	Content to display in the InfoWindow. This can be an HTML element, a plain-text string, or a string containing HTML. The InfoWindow will be sized according to the content. To set an explicit size for the content, set content to be a HTML element with that size.
disableAutoPan	boolean	Disable auto-pan on open. By default, the info window will pan the map so that it is fully visible when it opens.
maxWidth	number	Maximum width of the infowindow, regardless of content's width. This value is only considered if it is set before a call to open. To change the maximum width when changing content, call close, setOptions, and then open.
pixelOffset	Size	The offset, in pixels, of the tip of the info window from the point on the map at whose geographical coordinates the info window is anchored. If an InfoWindow is opened with an anchor, the pixelOffset will be calculated from the anchor's anchorPoint property.
position	LatLng|LatLngLiteral	The LatLng at which to display this InfoWindow. If the InfoWindow is opened with an anchor, the anchor's position will be used instead.
zIndex	number	All InfoWindows are displayed on the map in order of their zIndex, with higher values displaying in front of InfoWindows with lower values. By default, InfoWindows are displayed according to their latitude, with InfoWindows of lower latitudes appearing in front of InfoWindows at higher latitudes. InfoWindows are always displayed in front of markers.

google.maps.Polyline class

A polyline is a linear overlay of connected line segments on the map.

This class extends MVCObject.

Constructor

Constructor	Description
Polyline(opts?:*PolylineOptions*)	Create a polyline using the passed *PolylineOptions*, which specify both the path of the polyline and the stroke style to use when drawing the polyline. You may pass either an array of LatLngs or an MVCArray of LatLngs when constructing a polyline, though simple arrays are converted to MVCArrays within the polyline upon instantiation.

Methods

Methods	Return Value	Description
getDraggable()	boolean	Returns whether this shape can be dragged by the user.
getEditable()	boolean	Returns whether this shape can be edited by the user.
getMap()	Map	Returns the map on which this shape is attached.
getPath()	MVCArray<LatLng>	Retrieves the path.
getVisible()	boolean	Returns whether this poly is visible on the map.
setDraggable(draggable:boolean)	None	If set to true, the user can drag this shape over the map. The geodesic property defines the mode of dragging.
setEditable(editable:boolean)	None	If set to true, the user can edit this shape by dragging the control points shown at the vertices and on each segment.
setMap(map:Map)	None	Renders this shape on the specified map. If map is set to null, the shape will be removed.
setOptions(options:*PolylineOptions*)	None	
setPath(path:MVCArray<LatLng>|Array<LatLng|LatLngLiteral>)	None	Sets the path. See PolylineOptions for more details.

`setVisible(visible:boolean)`	`None`	Hides this poly if set to `false`.

Events

Events	Arguments	Description
`click`	*`PolyMouseEvent`*	This event is fired when the DOM click event is fired on the Polyline.
`dblclick`	*`PolyMouseEvent`*	This event is fired when the DOM dblclick event is fired on the Polyline.
`drag`	*`MouseEvent`*	This event is repeatedly fired while the user drags the polyline.
`dragend`	*`MouseEvent`*	This event is fired when the user stops dragging the polyline.
`dragstart`	*`MouseEvent`*	This event is fired when the user starts dragging the polyline.
`mousedown`	*`PolyMouseEvent`*	This event is fired when the DOM mousedown event is fired on the Polyline.
`mousemove`	*`PolyMouseEvent`*	This event is fired when the DOM mousemove event is fired on the Polyline.
`mouseout`	*`PolyMouseEvent`*	This event is fired on Polyline mouseout.
`mouseover`	*`PolyMouseEvent`*	This event is fired on Polyline mouseover.
`mouseup`	*`PolyMouseEvent`*	This event is fired when the DOM mouseup event is fired on the Polyline.
`rightclick`	*`PolyMouseEvent`*	This event is fired when the Polyline is right-clicked on.

google.maps.PolylineOptions object specification

Properties

Properties	Type	Description
`clickable`	`boolean`	Indicates whether this `Polyline` handles mouse events. Defaults to `true`.
`draggable`	`boolean`	If set to true, the user can drag this shape over the map. The `geodesic` property defines the mode of dragging. Defaults to `false`.
`editable`	`boolean`	If set to true, the user can edit this shape by dragging the control points shown at the vertices and on each segment. Defaults to `false`.
`geodesic`	`boolean`	When true, edges of the polygon are interpreted as geodesic and will follow the curvature of the Earth. When false, edges of the polygon are rendered as straight lines in screen space. Note that the shape of a geodesic polygon may appear to change when dragged, as the dimensions are maintained relative to the surface of the earth. Defaults to `false`.
`icons`	`Array<IconSequence>`	The icons to be rendered along the polyline.
`map`	`Map`	Map on which to display Polyline.
`path`	`MVCArray<LatLng>|Array<LatLng|LatLngLiteral>`	The ordered sequence of coordinates of the Polyline. This path may be specified using either a simple array of `LatLng`s, or an `MVCArray` of `LatLng`s. Note that if you pass a simple array, it will be converted to an `MVCArray` Inserting or removing LatLngs in the `MVCArray` will automatically update the polyline on the map.
`strokeColor`	`string`	The stroke color. All CSS3 colors are supported except for extended named colors.
`strokeOpacity`	`number`	The stroke opacity between 0.0 and 1.0.
`strokeWeight`	`number`	The stroke width in pixels.
`visible`	`boolean`	Whether this polyline is visible on the map. Defaults to `true`.
`zIndex`	`number`	The zIndex compared to other polys.

google.maps.IconSequence object specification

Describes how icons are to be rendered on a line.

If your polyline is geodesic, then the distances specified for both offset and repeat are calculated in meters by default. Setting either offset or repeat to a pixel value will cause the distances to be calculated in pixels on the screen.

Properties

Properties	Type	Description
`fixedRotation`	`boolean`	If `true`, each icon in the sequence has the same fixed rotation regardless of the angle of the edge on which it lies. Defaults to `false`, in which case each icon in the sequence is rotated to align with its edge.
`icon`	`Symbol`	The icon to render on the line.
`offset`	`string`	The distance from the start of the line at which an icon is to be rendered. This distance may be expressed as a percentage of line's length (e.g. '50%') or in pixels (e.g. '50px'). Defaults to '100%'.
`repeat`	`string`	The distance between consecutive icons on the line. This distance may be expressed as a percentage of the line's length (e.g. '50%') or in pixels (e.g. '50px'). To disable repeating of the icon, specify '0'. Defaults to '0'.

google.maps.Polygon class

A polygon (like a polyline) defines a series of connected coordinates in an ordered sequence; additionally, polygons form a closed loop and define a filled region.

This class extends `MVCObject`.

Constructor

Constructor	Description
`Polygon(opts?:PolygonOptions)`	Create a polygon using the passed `PolygonOptions`, which specify the polygon's path, the stroke style for the polygon's edges, and the fill style for the polygon's interior regions. A polygon may contain one or more paths, where each path consists of an array of `LatLng`s. You may pass either an array of LatLngs or an `MVCArray` of `LatLng`s when constructing these paths. Arrays are converted to `MVCArray`s within the polygon upon instantiation.

Methods

Methods	Return Valu
`getDraggable()`	`boolean`
`getEditable()`	`boolean`
`getMap()`	`Map`
`getPath()`	`MVCArray<`
`getPaths()`	`MVCArray<`
`getVisible()`	`boolean`

setDraggable(draggable:boolean)	None
setEditable(editable:boolean)	None
setMap(map:Map)	None
setOptions(options:*PolygonOptions*)	None
setPath(path:MVCArray<LatLng>|Array<LatLng|LatLngLiteral>)	None
setPaths(paths:MVCArray<MVCArray<LatLng>>|MVCArray<LatLng>|Array<Array<LatLng|LatLngLiteral>>|Array<LatLng|LatLngLiteral>)	None
setVisible(visible:boolean)	None

Events

Events	Arguments	Description
click	*PolyMouseEvent*	This event is fired when the DOM click event is fired on the Polygon.
dblclick	*PolyMouseEvent*	This event is fired when the DOM dblclick event is fired on the Polygon.
drag	*MouseEvent*	This event is repeatedly fired while the user drags the polygon.
dragend	*MouseEvent*	This event is fired when the user stops dragging the polygon.
dragstart	*MouseEvent*	This event is fired when the user starts dragging the polygon.
mousedown	*PolyMouseEvent*	This event is fired when the DOM mousedown event is fired on the Polygon.
mousemove	*PolyMouseEvent*	This event is fired when the DOM mousemove event is fired on the Polygon.
mouseout	*PolyMouseEvent*	This event is fired on Polygon mouseout.
mouseover	*PolyMouseEvent*	This event is fired on Polygon mouseover.
mouseup	*PolyMouseEvent*	This event is fired when the DOM mouseup event is fired on the Polygon.
rightclick	*PolyMouseEvent*	This event is fired when the Polygon is right-clicked on.

google.maps.PolygonOptions object specification

Properties

Properties	Type	Descriptio

`clickable`	`boolean`	Indicates whether th `Polygon` handles mouse events. Defaults to `true`.
`draggable`	`boolean`	If set to tru the user ca drag this shape ove the map. T `geodesic` property defines the mode of dragging. Defaults to `false`.
`editable`	`boolean`	If set to tru the user ca edit this shape by dragging t control po shown at t vertices ar on each segment. Defaults to `false`.
`fillColor`	`string`	The fill col All CSS3 colors are supported except for extended named colors.
`fillOpacity`	`number`	The fill opacity between 0 and 1.0
`geodesic`	`boolean`	When true, edges of th polygon ar interpreted geodesic a will follow curvature the Earth. When false edges of th polygon ar rendered a straight lin in screen space. Not that the shape of a geodesic polygon m appear to

		change wh dragged, a the dimension are maintaine relative to surface of earth. Defaults to `false`.
`map`	`Map`	Map on wh to display Polygon.
`paths`	`MVCArray<MVCArray<LatLng>>|MVCArray<LatLng>|Array<Array<LatLng|LatLngLiteral>>|Array<LatLng|LatLngLiteral>`	The ordere sequence coordinate that designates closed loo Unlike polylines, a polygon m consist of one or mo paths. As a result, the paths property m specify on more array of `LatLng` coordinate Paths are closed automatic do not rep the first vertex of th path as the last vertex Simple polygons r be defined using a sir array of `LatLng`s. More complex polygons r specify an array of arrays. An simple arra are conver into `MVCArray`s Inserting o removing `LatLng`s fr the `MVCAr` will automatic

		update the polygon or the map.
strokeColor	string	The stroke color. All CSS3 colo are suppor except for extended named colors.
strokeOpacity	number	The stroke opacity between 0 and 1.0
strokePosition	StrokePosition	The stroke position. Defaults to CENTER. T property is not suppor on Interne Explorer 8 and earlier
strokeWeight	number	The stroke width in pixels.
visible	boolean	Whether th polygon is visible on t map. Defa to true.
zIndex	number	The zIndex compared other poly

google.maps.PolyMouseEvent object specification

This object is returned from mouse events on polylines and polygons.

This object extends MouseEvent.

Properties

Properties	Type	Description
edge	number	The index of the edge within the path beneath the cursor when the event occurred, if the event occurred on a mid-point on an editable polygon.
path	number	The index of the path beneath the cursor when the event occurred, if the event occurred on a vertex and the polygon is editable. Otherwise undefined.
vertex	number	The index of the vertex beneath the cursor when the event occurred, if the event occurred on a vertex and the polyline or polygon is editable. If the event does not occur on a vertex, the value is undefined.

google.maps.Rectangle class

A rectangle overlay.

This class extends MVCObject.

Constructor

Constructor	Description
`Rectangle(opts?:RectangleOptions)`	Create a rectangle using the passed *RectangleOptions*, which specify the bounds and style.

Methods

Methods	Return Value	Description
`getBounds()`	`LatLngBounds`	Returns the bounds of this rectangle.
`getDraggable()`	`boolean`	Returns whether this rectangle can be dragged by the user.
`getEditable()`	`boolean`	Returns whether this rectangle can be edited by the user.
`getMap()`	`Map`	Returns the map on which this rectangle is displayed.
`getVisible()`	`boolean`	Returns whether this rectangle is visible on the map.
`setBounds(bounds:LatLngBounds)`	`None`	Sets the bounds of this rectangle.
`setDraggable(draggable:boolean)`	`None`	If set to true, the user can drag this rectangle over the map.
`setEditable(editable:boolean)`	`None`	If set to true, the user can edit this rectangle by dragging the control points shown at the corners and on each edge.
`setMap(map:Map)`	`None`	Renders the rectangle on the specified map. If map is set to null, the rectangle will be removed.
`setOptions(options:RectangleOptions)`	`None`	
`setVisible(visible:boolean)`	`None`	Hides this rectangle if set to `false`.

Events

Events	Arguments	Description
`bounds_changed`	`None`	This event is fired when the rectangle's bounds are changed.
`click`	*MouseEvent*	This event is fired when the DOM click event is fired on the rectangle.
`dblclick`	*MouseEvent*	This event is fired when the DOM dblclick event is fired on the rectangle.
`drag`	*MouseEvent*	This event is repeatedly fired while the user drags the rectangle.
`dragend`	*MouseEvent*	This event is fired when the user stops dragging the rectangle.
`dragstart`	*MouseEvent*	This event is fired when the user starts dragging the rectangle.
`mousedown`	*MouseEvent*	This event is fired when the DOM mousedown event is fired on the rectangle.
`mousemove`	*MouseEvent*	This event is fired when the DOM mousemove event is fired on the rectangle.
`mouseout`	*MouseEvent*	This event is fired on rectangle mouseout.
`mouseover`	*MouseEvent*	This event is fired on rectangle mouseover.
`mouseup`	*MouseEvent*	This event is fired when the DOM mouseup event is fired on the rectangle.
`rightclick`	*MouseEvent*	This event is fired when the rectangle is right-clicked on.

google.maps.RectangleOptions object specification

Properties

Properties	Type	Description
`bounds`	`LatLngBounds`	The bounds.
`clickable`	`boolean`	Indicates whether this `Rectangle` handles mouse events. Defaults to `true`.
`draggable`	`boolean`	If set to true, the user can drag this rectangle over the map. Defaults to `false`.
`editable`	`boolean`	If set to true, the user can edit this rectangle by dragging the control points shown at the corners and on

		each edge. Defaults to `false`.
`fillColor`	`string`	The fill color. All CSS3 colors are supported except for extended named colors.
`fillOpacity`	`number`	The fill opacity between 0.0 and 1.0
`map`	`Map`	Map on which to display Rectangle.
`strokeColor`	`string`	The stroke color. All CSS3 colors are supported except for extended named colors.
`strokeOpacity`	`number`	The stroke opacity between 0.0 and 1.0
`strokePosition`	`StrokePosition`	The stroke position. Defaults to CENTER. This property is not supported on Internet Explorer 8 and earlier.
`strokeWeight`	`number`	The stroke width in pixels.
`visible`	`boolean`	Whether this rectangle is visible on the map. Defaults to `true`.
`zIndex`	`number`	The zIndex compared to other polys.

google.maps.Circle class

A circle on the Earth's surface; also known as a "spherical cap".

This class extends `MVCObject`.

Constructor

Constructor	Description
`Circle(opts?:CircleOptions)`	Create a circle using the passed `CircleOptions`, which specify the center, radius, and style.

Methods

Methods	Return Value	Description
`getBounds()`	`LatLngBounds`	Gets the `LatLngBounds` of this Circle.
`getCenter()`	`LatLng`	Returns the center of this circle.
`getDraggable()`	`boolean`	Returns whether this circle can be dragged by the user.
`getEditable()`	`boolean`	Returns whether this circle can be edited by the user.
`getMap()`	`Map`	Returns the map on which this circle is displayed.
`getRadius()`	`number`	Returns the radius of this circle (in meters).
`getVisible()`	`boolean`	Returns whether this circle is visible on the map.
`setCenter(center:LatLng|LatLngLiteral)`	`None`	Sets the center of this circle.
`setDraggable(draggable:boolean)`	`None`	If set to true, the user can drag this circle over the map.
`setEditable(editable:boolean)`	`None`	If set to true, the user can edit this circle by dragging the control points shown at the center and around the circumference of the circle.
`setMap(map:Map)`	`None`	Renders the circle on the specified map. If map is set to null, the circle will be removed.
`setOptions(options:CircleOptions)`	`None`	
`setRadius(radius:number)`	`None`	Sets the radius of this circle (in meters).
`setVisible(visible:boolean)`	`None`	Hides this circle if set to `false`.

Events

Events	Arguments	Description
`center_changed`	`None`	This event is fired when the circle's center is changed.
`click`	`MouseEvent`	This event is fired when the DOM click event is fired on the circle.

dblclick	*MouseEvent*	This event is fired when the DOM dblclick event is fired on the circle.
drag	*MouseEvent*	This event is repeatedly fired while the user drags the circle.
dragend	*MouseEvent*	This event is fired when the user stops dragging the circle.
dragstart	*MouseEvent*	This event is fired when the user starts dragging the circle.
mousedown	*MouseEvent*	This event is fired when the DOM mousedown event is fired on the circle.
mousemove	*MouseEvent*	This event is fired when the DOM mousemove event is fired on the circle.
mouseout	*MouseEvent*	This event is fired on circle mouseout.
mouseover	*MouseEvent*	This event is fired on circle mouseover.
mouseup	*MouseEvent*	This event is fired when the DOM mouseup event is fired on the circle.
radius_changed	None	This event is fired when the circle's radius is changed.
rightclick	*MouseEvent*	This event is fired when the circle is right-clicked on.

google.maps.CircleOptions object specification

Properties

Properties	Type	Description
center	LatLng	The center
clickable	boolean	Indicates whether this Circle handles mouse events. Defaults to true.
draggable	boolean	If set to true, the user can drag this circle over the map. Defaults to false.
editable	boolean	If set to true, the user can edit this circle by dragging the control points shown at the center and around the circumference of the circle. Defaults to false.
fillColor	string	The fill color. All CSS3 colors are supported except for extended named colors.
fillOpacity	number	The fill opacity between 0.0 and 1.0
map	Map	Map on which to display Circle.
radius	number	The radius in meters on the Earth's surface
strokeColor	string	The stroke color. All CSS3 colors are supported except for extended named colors.
strokeOpacity	number	The stroke opacity between 0.0 and 1.0
strokePosition	StrokePosition	The stroke position. Defaults to CENTER. This property is not supported on Internet Explorer 8 and earlier.
strokeWeight	number	The stroke width in pixels.
visible	boolean	Whether this circle is visible on the map. Defaults to true.
zIndex	number	The zIndex compared to other polys.

google.maps.StrokePosition object specification

The possible positions of the stroke on a polygon.

Constant

Constant	Description
CENTER	The stroke is centered on the polygon's path, with half the stroke inside the polygon and half the stroke outside the polygon.
INSIDE	The stroke lies inside the polygon.
OUTSIDE	The stroke lies outside the polygon.

google.maps.GroundOverlay class

A rectangular image overlay on the map.

This class extends `MVCObject`.

Constructor

Constructor	Description
`GroundOverlay(url:string, bounds:LatLngBounds, opts?:GroundOverlayOptions)`	Creates a ground overlay from the provided image URL and its `LatLngBounds`. The image is scaled to fit the current bounds, and projected using the current map projection.

Methods

Methods	Return Value	Description
`getBounds()`	`LatLngBounds`	Gets the `LatLngBounds` of this overlay.
`getMap()`	`Map`	Returns the map on which this ground overlay is displayed.
`getOpacity()`	`number`	Returns the opacity of this ground overlay.
`getUrl()`	`string`	Gets the url of the projected image.
`setMap(map:Map)`	`None`	Renders the ground overlay on the specified map. If map is set to null, the overlay is removed.
`setOpacity(opacity:number)`	`None`	Sets the opacity of this ground overlay.

Events

Events	Arguments	Description
`click`	*`MouseEvent`*	This event is fired when the DOM click event is fired on the `GroundOverlay`.
`dblclick`	*`MouseEvent`*	This event is fired when the DOM dblclick event is fired on the `GroundOverlay`.

google.maps.GroundOverlayOptions object specification

This object defines the properties that can be set on a `GroundOverlay` object.

Properties

Properties	Type	Description
`clickable`	`boolean`	If `true`, the ground overlay can receive mouse events.
`map`	`Map`	The map on which to display the overlay.
`opacity`	`number`	The opacity of the overlay, expressed as a number between 0 and 1. Optional. Defaults to 1.

google.maps.OverlayView class

You can implement this class if you want to display custom types of overlay objects on the map.

Inherit from this class by setting your overlay's prototype: `MyOverlay.prototype = new google.maps.OverlayView();`. The `OverlayView` constructor is guaranteed to be an empty function.

You must implement three methods: `onAdd()`, `draw()`, and `onRemove()`.

- In the `onAdd()` method, you should create DOM objects and append them as children of the panes.
- In the `draw()` method, you should position these elements.
- In the `onRemove()` method, you should remove the objects from the DOM.

You must call `setMap()` with a valid `Map` object to trigger the call to the `onAdd()` method and `setMap(null)` in order to trigger the `onRemove()` method. The `setMap()` method can be called at the time of construction or at any point afterward when the overlay should be re-shown after removing. The `draw()` method will then be called whenever a map property changes that could change the position of the element, such as zoom,

center, or map type.

This class extends `MVCObject`.

Constructor

Constructor	Description
`OverlayView()`	Creates an `OverlayView`.

Methods

Methods	Return Value	Description
`draw()`	`None`	Implement this method to draw or update the overlay. This method is called after onAdd() and when the position from projection.fromLatLngToPixel() would return a new value for a given LatLng. This can happen on change of zoom, center, or map type. It is not necessarily called on drag or resize.
`getMap()`	`Map`|`StreetViewPanorama`	
`getPanes()`	`MapPanes`	Returns the panes in which this OverlayView can be rendered. The panes are not initialized until `onAdd` is called by the API.
`getProjection()`	`MapCanvasProjection`	Returns the `MapCanvasProjection` object associated with this `OverlayView`. The projection is not initialized until `onAdd` is called by the API.
`onAdd()`	`None`	Implement this method to initialize the overlay DOM elements. This method is called once after setMap() is called with a valid map. At this point, panes and projection will have been initialized.
`onRemove()`	`None`	Implement this method to remove your elements from the DOM. This method is called once following a call to setMap(null).
`setMap(map:Map|StreetViewPanorama)`	`None`	Adds the overlay to the map or panorama.

google.maps.MapPanes object specification

This object contains the DOM elements in which overlays are rendered. They are listed below with 'Pane 0' at the bottom and 'Pane 4' at the top.

Properties

Properties	Type	Description
`floatPane`	`Node`	This pane contains the info window. It is above all map overlays. (Pane 4).
`mapPane`	`Node`	This pane is the lowest pane and is above the tiles. It may not receive DOM events. (Pane 0).
`markerLayer`	`Node`	This pane contains markers. It may not receive DOM events. (Pane 2).
`overlayLayer`	`Node`	This pane contains polylines, polygons, ground overlays and tile layer overlays. It may not receive DOM events. (Pane 1).
`overlayMouseTarget`	`Node`	This pane contains elements that receive DOM events. (Pane 3).

google.maps.MapCanvasProjection object specification

This object is made available to the `OverlayView` from within the draw method. It is not guaranteed to be initialized until draw is called.

This object extends `MVCObject`.

Methods

Methods	Return Value	Description

`fromContainerPixelToLatLng(pixel:Point, nowrap?:boolean)`	`LatLng`	Computes the geographical coordinates from pixel coordinates in the map's container.
`fromDivPixelToLatLng(pixel:Point, nowrap?:boolean)`	`LatLng`	Computes the geographical coordinates from pixel coordinates in the div that holds the draggable map.
`fromLatLngToContainerPixel(latLng:LatLng)`	`Point`	Computes the pixel coordinates of the given geographical location in the map's container element.
`fromLatLngToDivPixel(latLng:LatLng)`	`Point`	Computes the pixel coordinates of the given geographical location in the DOM element that holds the draggable map.
`getWorldWidth()`	`number`	The width of the world in pixels in the current zoom level. For projections with a heading angle of either 90 or 270 degrees, this corresponds to the pixel span in the Y-axis.

google.maps.Geocoder class

A service for converting between an address and a `LatLng`.

Constructor

Constructor	Description
`Geocoder()`	Creates a new instance of a `Geocoder` that sends geocode requests to Google servers.

Methods

Methods	Return Value	Description
`geocode(request:GeocoderRequest, callback:function(Array<GeocoderResult>, GeocoderStatus))`	`None`	Geocode a request.

google.maps.GeocoderRequest object specification

The specification for a geocoding request to be sent to the `Geocoder`.

Properties

Properties	Type	Description
`address`	`string`	Address. Optional.
`bounds`	`LatLngBounds`	LatLngBounds within which to search. Optional.
`componentRestrictions`	`GeocoderComponentRestrictions`	Components are used to restrict results to a specific area. A filter consists of one or more of: `route`, `locality`, `administrativeArea`, `postalCode`, `country`. Only the results that match all the filters will be returned. Filter values support the same methods of spelling correction and partial matching as other geocoding requests. Optional.
`location`	`LatLng|LatLngLiteral`	LatLng about which to search. Optional.
`region`	`string`	Country code used to bias the search, specified as a Unicode region subtag / CLDR identifier. Optional.

google.maps.GeocoderComponentRestrictions object specification

`GeocoderComponentRestrictions` represent a set of filters that resolve to a specific area. For details on how this works, see Geocoding Component Filtering.

Properties

Properties	Type	Description

administrativeArea	string	Matches all the administrative_area levels. Optional.
country	string	Matches a country name or a two letter ISO 3166-1 country code. Optional.
locality	string	Matches against both locality and sublocality types. Optional.
postalCode	string	Matches postal_code and postal_code_prefix. Optional.
route	string	Matches the long or short name of a route. Optional.

google.maps.GeocoderStatus class

The status returned by the Geocoder on the completion of a call to geocode().

Constant

Constant	Description
ERROR	There was a problem contacting the Google servers.
INVALID_REQUEST	This GeocoderRequest was invalid.
OK	The response contains a valid GeocoderResponse.
OVER_QUERY_LIMIT	The webpage has gone over the requests limit in too short a period of time.
REQUEST_DENIED	The webpage is not allowed to use the geocoder.
UNKNOWN_ERROR	A geocoding request could not be processed due to a server error. The request may succeed if you try again.
ZERO_RESULTS	No result was found for this GeocoderRequest.

google.maps.GeocoderResult object specification

A single geocoder result retrieved from the geocode server. A geocode request may return multiple result objects. Note that though this result is "JSON-like," it is not strictly JSON, as it indirectly includes a LatLng object.

Properties

Properties	Type	Description
address_components	Array<GeocoderAddressComponent>	An array of GeocoderAddressComponents
formatted_address	string	A string containing the human-readable address of this location.
geometry	GeocoderGeometry	A GeocoderGeometry object
partial_match	boolean	Whether the geocoder did not return an exact match for the original request, though it was able to match part of the requested address.
postcode_localities	Array<string>	An array of strings denoting all the localities contained in a postal code. This is only present when the result is a postal code that contains multiple localities.
types	Array<string>	An array of strings denoting the type of the returned geocoded element. For a list of possible strings, refer to the Address Component Types section of the Developer's Guide.

google.maps.GeocoderAddressComponent object specification

A single address component within a GeocoderResult. A full address may consist of multiple address components.

Properties

Properties	Type	Description
long_name	string	The full text of the address component
short_name	string	The abbreviated, short text of the given address component

types	Array<string>	An array of strings denoting the type of this address component. A list of valid types can be found here

google.maps.GeocoderGeometry object specification

Geometry information about this GeocoderResult

Properties

Properties	Type	Description
bounds	LatLngBounds	The precise bounds of this GeocoderResult, if applicable
location	LatLng	The latitude/longitude coordinates of this result
location_type	GeocoderLocationType	The type of location returned in location
viewport	LatLngBounds	The bounds of the recommended viewport for displaying this GeocoderResult

google.maps.GeocoderLocationType class

Describes the type of location returned from a geocode.

Constant

Constant	Description
APPROXIMATE	The returned result is approximate.
GEOMETRIC_CENTER	The returned result is the geometric center of a result such a line (e.g. street) or polygon (region).
RANGE_INTERPOLATED	The returned result reflects an approximation (usually on a road) interpolated between two precise points (such as intersections). Interpolated results are generally returned when rooftop geocodes are unavailable for a street address.
ROOFTOP	The returned result reflects a precise geocode.

google.maps.DirectionsRenderer class

Renders directions retrieved in the form of a DirectionsResult object retrieved from the DirectionsService.

This class extends MVCObject.

Constructor

Constructor	Description
DirectionsRenderer(opts?:*DirectionsRendererOptions*)	Creates the renderer with the given options. Directions can be rendered on a map (as visual overlays) or additionally on a panel (as textual instructions).

Methods

Methods	Return Value	Description
getDirections()	DirectionsResult	Returns the renderer's current set of directions.
getMap()	Map	Returns the map on which the DirectionsResult is rendered.
getPanel()	Node	Returns the panel in which the DirectionsResult is rendered.
getRouteIndex()	number	Returns the current (zero-based) route index in use by this DirectionsRenderer object.
setDirections(directions:*DirectionsResult*)	None	Set the renderer to use the result from the DirectionsService. Setting a valid set of directions in this manner will display the directions on the renderer's designated map and panel.

`setMap(map:Map)`	`None`	This method specifies the map on which directions will be rendered. Pass `null` to remove the directions from the map.
`setOptions(options:DirectionsRendererOptions)`	`None`	Change the options settings of this `DirectionsRenderer` after initialization.
`setPanel(panel:Node)`	`None`	This method renders the directions in a `<div>`. Pass `null` to remove the content from the panel.
`setRouteIndex(routeIndex:number)`	`None`	Set the (zero-based) index of the route in the `DirectionsResult` object to render. By default, the first route in the array will be rendered.

Events

Events	Arguments	Description
`directions_changed`	`None`	This event is fired when the rendered directions change, either when a new `DirectionsResult` is set or when the user finishes dragging a change to the directions path.

google.maps.DirectionsRendererOptions object specification

This object defines the properties that can be set on a `DirectionsRenderer` object.

Properties

Properties	Type	Description
`directions`	`DirectionsResult`	The directions to display on the map and/or in a `<div>` panel, retrieved as a `DirectionsResult` object from `DirectionsService`.
`draggable`	`boolean`	If true, allows the user to drag and modify the paths of routes rendered by this `DirectionsRenderer`.
`hideRouteList`	`boolean`	This property indicates whether the renderer should provide UI to select amongst alternative routes. By default, this flag is `false` and a user-selectable list of routes will be shown in the directions' associated panel. To hide that list, set `hideRouteList` to `true`.
`infoWindow`	`InfoWindow`	The `InfoWindow` in which to render text information when a marker is clicked. Existing info window content will be overwritten and its position moved. If no info window is specified, the `DirectionsRenderer` will create and use its own info window. This property will be ignored if `suppressInfoWindows` is set to true.
`map`	`Map`	Map on which to display the directions.
`markerOptions`	`MarkerOptions`	Options for the markers. All markers rendered by the `DirectionsRenderer` will use these options.
`panel`	`Node`	The `<div>` in which to display the directions steps.
`polylineOptions`	`PolylineOptions`	Options for the polylines. All polylines rendered by the `DirectionsRenderer` will use these options.
`preserveViewport`	`boolean`	By default, the input map is centered and zoomed to the bounding box of this set of directions. If this option is set to `true`, the viewport is left unchanged, unless the map's center and zoom were never set.
`routeIndex`	`number`	The index of the route within the `DirectionsResult` object. The default value is 0.
`suppressBicyclingLayer`	`boolean`	Suppress the rendering of the `BicyclingLayer` when bicycling directions are requested.
`suppressInfoWindows`	`boolean`	Suppress the rendering of info windows.
`suppressMarkers`	`boolean`	Suppress the rendering of markers.
`suppressPolylines`	`boolean`	Suppress the rendering of polylines.

google.maps.DirectionsService class

A service for computing directions between two or more places.

Constructor

Constructor	Description
`DirectionsService()`	Creates a new instance of a `DirectionsService` that sends directions queries to Google servers.

Methods

Methods	Return Value	Description
`route(request:`*`DirectionsRequest`*`, callback:function(DirectionsResult, DirectionsStatus))`	`None`	Issue a directions search request.

google.maps.DirectionsRequest object specification

A directions query to be sent to the `DirectionsService`.

Properties

Properties	Type	Description
`avoidFerries`	`boolean`	If true, instructs the Directions service to avoid ferries where possible. Optional.
`avoidHighways`	`boolean`	If true, instructs the Directions service to avoid highways where possible. Optional.
`avoidTolls`	`boolean`	If true, instructs the Directions service to avoid toll roads where possible. Optional.
`destination`	`LatLng|string`	Location of destination. This can be specified as either a string to be geocoded or a `LatLng`. Required.
`durationInTraffic`	`boolean`	Whether or not we should provide trip duration based on current traffic conditions. Only available to Maps API for Work customers.
`optimizeWaypoints`	`boolean`	If set to true, the `DirectionService` will attempt to re-order the supplied intermediate waypoints to minimize overall cost of the route. If waypoints are optimized, inspect `DirectionsRoute.waypoint_order` in the response to determine the new ordering.
`origin`	`LatLng|string`	Location of origin. This can be specified as either a string to be geocoded or a `LatLng`. Required.
`provideRouteAlternatives`	`boolean`	Whether or not route alternatives should be provided. Optional.
`region`	`string`	Region code used as a bias for geocoding requests. Optional.
`transitOptions`	`TransitOptions`	Settings that apply only to requests where `travelMode` is TRANSIT. This object will have no effect for other travel modes.
`travelMode`	`TravelMode`	Type of routing requested. Required.
`unitSystem`	`UnitSystem`	Preferred unit system to use when displaying distance. Defaults to the unit system used in the country of origin.
`waypoints`	`Array<DirectionsWaypoint>`	Array of intermediate waypoints. Directions will be calculated from the origin to the destination by way of each waypoint in this array. The maximum allowed waypoints is 8, plus the origin, and destination. Maps API for Business customers are allowed 23 waypoints, plus the origin, and destination. Waypoints are not supported for transit directions. Optional.

google.maps.TravelMode class

The valid travel modes that can be specified in a `DirectionsRequest` as well as the travel modes returned in a `DirectionsStep`.

Constant

Constant	Description
`BICYCLING`	Specifies a bicycling directions request.
`DRIVING`	Specifies a driving directions request.
`TRANSIT`	Specifies a transit directions request.
`WALKING`	Specifies a walking directions request.

google.maps.UnitSystem class

The valid unit systems that can be specified in a `DirectionsRequest`.

Constant

Constant	Description
`IMPERIAL`	Specifies that distances in the `DirectionsResult` should be expressed in imperial units.
`METRIC`	Specifies that distances in the `DirectionsResult` should be expressed in metric units.

google.maps.TransitOptions object specification

The TransitOptions object to be included in a `DirectionsRequest` when the travel mode is set to TRANSIT.

Properties

Properties	Type	Description
`arrivalTime`	`Date`	The desired arrival time for the route, specified as a Date object. The Date object measures time in milliseconds since 1 January 1970. If arrival time is specified, departure time is ignored.
`departureTime`	`Date`	The desired departure time for the route, specified as a Date object. The Date object measures time in milliseconds since 1 January 1970. If neither departure time nor arrival time is specified, the time is assumed to be "now".
`modes`	`Array<TransitMode>`	One or more preferred modes of transit, such as bus or train. If no preference is given, the API returns the default best route.
`routingPreference`	`TransitRoutePreference`	A preference that can bias the choice of transit route, such as less walking. If no preference is given, the API returns the default best route.

google.maps.TransitMode class

The valid transit mode e.g. bus that can be specified in a `TransitOptions`.

Constant

Constant	Description
`BUS`	Specifies bus as a preferred mode of transit.
`RAIL`	Specifies rail as a preferred mode of transit.
`SUBWAY`	Specifies subway as a preferred mode of transit.
`TRAIN`	Specifies train as a preferred mode of transit.
`TRAM`	Specifies tram as a preferred mode of transit.

google.maps.TransitRoutePreference class

The valid transit route type that can be specified in a `TransitOptions`.

Constant

Constant	Description
`FEWER_TRANSFERS`	Specifies that the calculated route should prefer a limited number of transfers.
`LESS_WALKING`	Specifies that the calculated route should prefer limited amounts of walking.

google.maps.TransitFare object specification

A fare of a `DirectionsRoute` consisting of value and currency.

google.maps.DirectionsWaypoint object specification

A `DirectionsWaypoint` represents a location between origin and destination through which the trip should be routed.

Properties

Properties	Type	Description
`location`	`LatLng|string`	Waypoint location. Can be an address string or `LatLng`. Optional.
`stopover`	`boolean`	If `true`, indicates that this waypoint is a stop between the origin and destination. This has the effect of splitting the route into two. This value is `true` by default. Optional.

google.maps.DirectionsStatus class

The status returned by the `DirectionsService` on the completion of a call to `route()`.

Constant

Constant	Description
`INVALID_REQUEST`	The `DirectionsRequest` provided was invalid.
`MAX_WAYPOINTS_EXCEEDED`	Too many `DirectionsWaypoint`s were provided in the `DirectionsRequest`. The total allowed waypoints is 8, plus the origin and destination. Maps API for Work customers are allowed 23 waypoints, plus the origin, and destination.
`NOT_FOUND`	At least one of the origin, destination, or waypoints could not be geocoded.
`OK`	The response contains a valid `DirectionsResult`.
`OVER_QUERY_LIMIT`	The webpage has gone over the requests limit in too short a period of time.
`REQUEST_DENIED`	The webpage is not allowed to use the directions service.
`UNKNOWN_ERROR`	A directions request could not be processed due to a server error. The request may succeed if you try again.
`ZERO_RESULTS`	No route could be found between the origin and destination.

google.maps.DirectionsResult object specification

The directions response retrieved from the directions server. You can render these using a `DirectionsRenderer` or parse this object and render it yourself. You must display the warnings and copyrights as noted in the Maps API terms of service. Note that though this result is "JSON-like," it is not strictly JSON, as it indirectly includes `LatLng` objects.

Properties

Properties	Type	Description
`routes`	`Array<DirectionsRoute>`	An array of `DirectionsRoute`s, each of which contains information about the legs and steps of which it is composed. There will only be one route unless the `DirectionsRequest` was made with `provideRouteAlternatives` set to `true`.

google.maps.DirectionsRoute object specification

A single route containing a set of legs in a `DirectionsResult`. Note that though this object is "JSON-like," it is not strictly JSON, as it directly and indirectly includes `LatLng` objects.

Properties

Properties	Type	Description
`bounds`	`LatLngBounds`	The bounds for this route.
`copyrights`	`string`	Copyrights text to be displayed for this route.
`fare`	`TransitFare`	The total fare for the whole transit trip. Only applicable to transit requests.
`legs`	`Array<DirectionsLeg>`	An array of `DirectionsLeg`s, each of which contains information about the steps of which it is composed. There will be one leg for each waypoint or destination specified. So a route with no waypoints will contain one `DirectionsLeg` and a route with one waypoint will contain two.
`overview_path`	`Array<LatLng>`	An array of `LatLng`s representing the entire course of this route. The path is simplified in order to make it suitable in contexts where a small number of vertices is required (such as Static Maps API URLs).
`overview_polyline`	`string`	An encoded polyline representation of the route in overview_path. This polyline is an approximate (smoothed) path of the resulting directions.
`warnings`	`Array<string>`	Warnings to be displayed when showing these directions.
`waypoint_order`	`Array<number>`	If `optimizeWaypoints` was set to `true`, this field will contain the re-ordered permutation of the input waypoints. For example, if the input was: Origin: Los Angeles Waypoints: Dallas, Bangor, Phoenix Destination: New York and the optimized output was ordered as follows: Origin: Los Angeles Waypoints: Phoenix, Dallas, Bangor Destination: New York then this field will be an `Array` containing the values [2, 0, 1]. Note that the numbering of waypoints is zero-based. If any of the input waypoints has `stopover` set to `false`, this field will be empty, since route optimization is not available for such queries.

google.maps.DirectionsLeg object specification

A single leg consisting of a set of steps in a `DirectionsResult`. Some fields in the leg may not be returned for all requests. Note that though this result is "JSON-like," it is not strictly JSON, as it directly and indirectly includes `LatLng` objects.

Properties

Properties	Type	Description
`arrival_time`	`Time`	An estimated arrival time for this leg. Only applicable for TRANSIT requests.
`departure_time`	`Time`	An estimated departure time for this leg. Only applicable for TRANSIT requests.
`distance`	`Distance`	The total distance covered by this leg. This property may be undefined as the distance may be unknown.
`duration`	`Duration`	The total duration of this leg. This property may be undefined as the duration may be unknown.
`duration_in_traffic`	`Duration`	The total duration of this leg, taking into account current traffic conditions. This property may be undefined as the duration may be unknown. Only available to Maps API for Work customers when `durationInTraffic` is set to `true` when making the request.
`end_address`	`string`	The address of the destination of this leg.
`end_location`	`LatLng`	The `DirectionsService` calculates directions between locations by using the nearest transportation option (usually a road) at the start and end locations. `end_location`

		indicates the actual geocoded destination, which may be different than the `end_location` of the last step if, for example, the road is not near the destination of this leg.
`start_address`	`string`	The address of the origin of this leg.
`start_location`	`LatLng`	The `DirectionsService` calculates directions between locations by using the nearest transportation option (usually a road) at the start and end locations. `start_location` indicates the actual geocoded origin, which may be different than the `start_location` of the first step if, for example, the road is not near the origin of this leg.
`steps`	`Array<DirectionsStep>`	An array of `DirectionsStep`s, each of which contains information about the individual steps in this leg.
`via_waypoints`	`Array<LatLng>`	An array of waypoints along this leg that were not specified in the original request, either as a result of a user dragging the polyline or selecting an alternate route.

google.maps.DirectionsStep object specification

A single `DirectionsStep` in a `DirectionsResult`. Some fields may be undefined. Note that though this object is "JSON-like," it is not strictly JSON, as it directly includes `LatLng` objects.

Properties

Properties	Type	Description
`distance`	`Distance`	The distance covered by this step. This property may be undefined as the distance may be unknown.
`duration`	`Duration`	The typical time required to perform this step in seconds and in text form. This property may be undefined as the duration may be unknown.
`end_location`	`LatLng`	The ending location of this step.
`instructions`	`string`	Instructions for this step.
`path`	`Array<LatLng>`	A sequence of `LatLng`s describing the course of this step.
`start_location`	`LatLng`	The starting location of this step.
`steps`	`Array<DirectionsStep>`	Sub-steps of this step. Specified for non-transit sections of transit routes.
`transit`	`TransitDetails`	Transit-specific details about this step. This property will be undefined unless the travel mode of this step is `TRANSIT`.
`travel_mode`	`TravelMode`	The mode of travel used in this step.

google.maps.Distance object specification

A representation of distance as a numeric value and a display string.

Properties

Properties	Type	Description
`text`	`string`	A string representation of the distance value, using the `UnitSystem` specified in the request.
`value`	`number`	The distance in meters.

google.maps.Duration object specification

A representation of duration as a numeric value and a display string.

Properties

Properties	Type	Description
`text`	`string`	A string representation of the duration value.

value	number	The duration in seconds.

google.maps.Time object specification

Properties

Properties	Type	Description
text	string	A string representing the time's value. The time is displayed in the time zone of the transit stop.
time_zone	string	The time zone in which this stop lies. The value is the name of the time zone as defined in the IANA Time Zone Database, e.g. "America/New_York".
value	Date	The time of this departure or arrival, specified as a JavaScript Date object.

google.maps.TransitDetails object specification

Properties

Properties	Type	Description
arrival_stop	TransitStop	The arrival stop of this transit step.
arrival_time	Time	The arrival time of this step, specified as a Time object.
departure_stop	TransitStop	The departure stop of this transit step.
departure_time	Time	The departure time of this step, specified as a Time object.
headsign	string	The direction in which to travel on this line, as it is marked on the vehicle or at the departure stop.
headway	number	The expected number of seconds between equivalent vehicles at this stop.
line	TransitLine	Details about the transit line used in this step.
num_stops	number	The number of stops on this step. Includes the arrival stop, but not the departure stop.

google.maps.TransitStop object specification

Properties

Properties	Type	Description
location	LatLng	The location of this stop.
name	string	The name of this transit stop.

google.maps.TransitLine object specification

Properties

Properties	Type	Description
agencies	Array<TransitAgency>	The transit agency that operates this transit line.
color	string	The color commonly used in signage for this transit line, represented as a hex string.
icon	string	The URL for an icon associated with this line.
name	string	The full name of this transit line, e.g. "8 Avenue Local".
short_name	string	The short name of this transit line, e.g. "E".
text_color	string	The text color commonly used in signage for this transit line, represented as a hex string.
url	string	The agency's URL which is specific to this transit line.
vehicle	TransitVehicle	The type of vehicle used, e.g. train or bus.

google.maps.TransitAgency object specification

Properties

Properties	Type	Description
name	string	The name of this transit agency.
phone	string	The transit agency's phone number.
url	string	The transit agency's URL.

google.maps.TransitVehicle object specification

Properties

Properties	Type	Description
icon	string	A URL for an icon that corresponds to the type of vehicle used on this line.
local_icon	string	A URL for an icon that corresponds to the type of vehicle used in this region instead of the more general icon.
name	string	A name for this type of TransitVehicle, e.g. "Train" or "Bus".
type	VehicleType	The type of vehicle used, e.g. train, bus, or ferry.

google.maps.VehicleType object specification

Possible values for vehicle types. These values are specifed as strings, i.e. 'BUS' or 'TRAIN'.

Constant

Constant	Description
BUS	Bus.
CABLE_CAR	A vehicle that operates on a cable, usually on the ground. Aerial cable cars may be of the type GONDOLA_LIFT.
COMMUTER_TRAIN	Commuter rail.
FERRY	Ferry.
FUNICULAR	A vehicle that is pulled up a steep incline by a cable.
GONDOLA_LIFT	An aerial cable car.
HEAVY_RAIL	Heavy rail.
HIGH_SPEED_TRAIN	High speed train.
INTERCITY_BUS	Intercity bus.
METRO_RAIL	Light rail.
MONORAIL	Monorail.
OTHER	Other vehicles.
RAIL	Rail.
SHARE_TAXI	Share taxi is a sort of bus transport with ability to drop off and pick up passengers anywhere on its route. Generally share taxi uses minibus vehicles.
SUBWAY	Underground light rail.
TRAM	Above ground light rail.
TROLLEYBUS	Trolleybus.

google.maps.ElevationService class

Defines a service class that talks directly to Google servers for requesting elevation data.

Constructor

Constructor	Description
`ElevationService()`	Creates a new instance of a `ElevationService` that sends elevation queries to Google servers.

Methods

Methods	Return Value	Description
`getElevationAlongPath(request:`*`PathElevationRequest`*`, callback:function(Array<ElevationResult>, ElevationStatus))`	`None`	Makes an elevation request along a path, where the elevation data are returned as distance-based samples along that path.
`getElevationForLocations(request:`*`LocationElevationRequest`*`, callback:function(Array<ElevationResult>, ElevationStatus))`	`None`	Makes an elevation request for a list of discrete locations.

google.maps.LocationElevationRequest object specification

An elevation request sent by the `ElevationService` containing the list of discrete coordinates (`LatLng`s) for which to return elevation data.

Properties

Properties	Type	Description
`locations`	`Array<LatLng>`	The discrete locations for which to retrieve elevations.

google.maps.PathElevationRequest object specification

An elevation query sent by the `ElevationService` containing the path along which to return sampled data. This request defines a continuous path along the earth along which elevation samples should be taken at evenly-spaced distances. All paths from vertex to vertex use segments of the great circle between those two points.

Properties

Properties	Type	Description
`path`	`Array<LatLng>`	The path along which to collect elevation values.
`samples`	`number`	Required. The number of equidistant points along the given path for which to retrieve elevation data, including the endpoints. The number of samples must be a value between 2 and 512 inclusive.

google.maps.ElevationResult object specification

The result of an `ElevationService` request, consisting of the set of elevation coordinates and their elevation values. Note that a single request may produce multiple `ElevationResult`s.

Properties

Properties	Type	Description
`elevation`	`number`	The elevation of this point on Earth, in meters above sea level.
`location`	`LatLng`	The location of this elevation result.
`resolution`	`number`	The distance, in meters, between sample points from which the elevation was interpolated. This property will be missing if the resolution is not known. Note that elevation data becomes more coarse (larger `resolution` values) when multiple points are passed. To obtain the most accurate elevation value for a point, it should be queried independently.

google.maps.ElevationStatus class

The status returned by the `ElevationService` upon completion of an elevation requerst.

Constant

Constant	Description
`INVALID_REQUEST`	This request was invalid.
`OK`	The request did not encounter any errors.
`OVER_QUERY_LIMIT`	The webpage has gone over the requests limit in too short a period of time.
`REQUEST_DENIED`	The webpage is not allowed to use the elevation service for some reason.
`UNKNOWN_ERROR`	A geocoding, directions or elevation request could not be successfully processed, yet the exact reason for the failure is not known.

google.maps.MaxZoomService class

A service for obtaining the highest zoom level at which satellite imagery is available for a given location.

Constructor

Constructor	Description
`MaxZoomService()`	Creates a new instance of a `MaxZoomService` that can be used to send queries about the maximum zoom level available for satellite imagery.

Methods

Methods	Return Value	Description
`getMaxZoomAtLatLng(latlng:LatLng|LatLngLiteral, callback:function(MaxZoomResult))`	`None`	Returns the maximum zoom level available at a particular `LatLng` for the Satellite map type. As this request is asynchronous, you must pass a `callback` function which will be executed upon completion of the request, being passed a `MaxZoomResult`.

google.maps.MaxZoomResult object specification

A MaxZoom result in JSON format retrieved from the MaxZoomService.

Properties

Properties	Type	Description
`status`	`MaxZoomStatus`	Status of the request.
`zoom`	`number`	The maximum zoom level found at the given `LatLng`.

google.maps.MaxZoomStatus class

The status returned by the `MaxZoomService` on the completion of a call to `getMaxZoomAtLatLng()`.

Constant

Constant	Description
`ERROR`	There was a problem contacting the Google servers.
`OK`	The response contains a valid `MaxZoomResult`.

google.maps.DistanceMatrixService class

A service for computing distances between multiple origins and destinations.

Constructor

Constructor	Description
`DistanceMatrixService()`	Creates a new instance of a `DistanceMatrixService` that sends distance matrix queries to Google servers.

Methods

Methods	Return Value	Description
`getDistanceMatrix(request:`*`DistanceMatrixRequest`*`, callback:function(DistanceMatrixResponse, DistanceMatrixStatus))`	`None`	Issues a distance matrix request.

google.maps.DistanceMatrixRequest object specification

A distance matrix query sent by the `DistanceMatrixService` containing arrays of origin and destination locations, and various options for computing metrics.

Properties

Properties	Type	Description
`avoidFerries`	`boolean`	If true, instructs the Distance Matrix service to avoid ferries where possible. Optional.
`avoidHighways`	`boolean`	If true, instructs the Distance Matrix service to avoid highways where possible. Optional.
`avoidTolls`	`boolean`	If true, instructs the Distance Matrix service to avoid toll roads where possible. Optional.
`destinations`	`Array<LatLng>|Array<string>`	An array containing destination address strings and/or `LatLng`s, to which to calculate distance and time. Required.
`durationInTraffic`	`boolean`	Whether or not we should provide trip durations based on current traffic conditions. Only available to Maps API for Work customers.
`origins`	`Array<LatLng>|Array<string>`	An array containing origin address strings and/or `LatLng`s, from which to calculate distance and time. Required.
`region`	`string`	Region code used as a bias for geocoding requests. Optional.
`transitOptions`	`TransitOptions`	Settings that apply only to requests where `travelMode` is TRANSIT. This object will have no effect for other travel modes.
`travelMode`	`TravelMode`	Type of routing requested. Required.
`unitSystem`	`UnitSystem`	Preferred unit system to use when displaying distance. Optional; defaults to metric.

google.maps.DistanceMatrixResponse object specification

The response to a `DistanceMatrixService` request, consisting of the formatted origin and destination addresses, and a sequence of `DistanceMatrixResponseRow`s, one for each corresponding origin address.

Properties

Properties	Type	Description
`destinationAddresses`	`Array<string>`	The formatted destination addresses.
`originAddresses`	`Array<string>`	The formatted origin addresses.
`rows`	`Array<DistanceMatrixResponseRow>`	The rows of the matrix, corresponding to the origin addresses.

google.maps.DistanceMatrixResponseRow object specification

A row of the response to a `DistanceMatrixService` request, consisting of a sequence of `DistanceMatrixResponseElement`s, one for each corresponding destination address.

Properties

Properties	Type	Description
`elements`	`Array<DistanceMatrixResponseElement>`	The row's elements, corresponding to the destination addresses.

google.maps.DistanceMatrixResponseElement object specification

A single element of a response to a `DistanceMatrixService` request, which contains the duration and distance from one origin to one destination.

Properties

Properties	Type	Description
`distance`	`Distance`	The distance for this origin-destination pairing. This property may be undefined as the distance may be unknown.
`duration`	`Duration`	The duration for this origin-destination pairing. This property may be undefined as the duration may be unknown.
`fare`	`TransitFare`	The total fare for this origin-destination pairing. Only applicable to transit requests.
`status`	`DistanceMatrixElementStatus`	The status of this particular origin-destination pairing.

google.maps.DistanceMatrixStatus class

The top-level status about the request in general returned by the `DistanceMatrixService` upon completion of a distance matrix request.

Constant

Constant	Description
`INVALID_REQUEST`	The provided request was invalid.
`MAX_DIMENSIONS_EXCEEDED`	The request contains more than 25 origins, or more than 25 destinations.
`MAX_ELEMENTS_EXCEEDED`	The product of origins and destinations exceeds the per-query limit.
`OK`	The response contains a valid result.
`OVER_QUERY_LIMIT`	Too many elements have been requested within the allowed time period. The request should succeed if you try again after a reasonable amount of time.
`REQUEST_DENIED`	The service denied use of the Distance Matrix service by your web page.
`UNKNOWN_ERROR`	A Distance Matrix request could not be processed due to a server error. The request may succeed if you try again.

google.maps.DistanceMatrixElementStatus class

The element-level status about a particular origin-destination pairing returned by the `DistanceMatrixService` upon completion of a distance matrix request.

Constant

Constant	Description
`NOT_FOUND`	The origin and/or destination of this pairing could not be geocoded.
`OK`	The response contains a valid result.
`ZERO_RESULTS`	No route could be found between the origin and destination.

google.maps.Attribution object specification

Properties

Properties	Type	Description
`iosDeepLinkId`	`string`	The iOS deep link to associate with this place when a user saves the place. When the user views the place in an iOS app, this URL will serve as the link on the `source` string. If there is no deep link or the app that handles the deep link is not present, the `webURL` will be used instead.
`source`	`string`	The source (origin) to associate with this place when it is saved by a user. For example, this could be the name of your website or application. The user who saved the place will see this source when they view the place in Google Maps. `source` is required for an `Attribution` to be considered valid. If it is not provided an error will be thrown.
`webUrl`	`string`	The URL (http or https) of the page to associate with this place when a user saves the place. When the user views the place in a desktop or Android app, this URL will serve as the link on the `source` string. When the user views the place in an iOS app, and there is no deep link provided or the app that handles the deep link is not present, this URL will serve as the link on the `source` string.

google.maps.Place object specification

Properties

Properties	Type	Description
`location`	`LatLng`|`LatLngLiteral`	The `LatLng` of the entity described by this Place. This must be provided for the `Place` to be considered valid.
`placeId`	`string`	The place ID of the place (such as a business or point of interest). The place ID is a unique identifier of a place in the Google Maps database. Note that the `placeId` is the most accurate way of identifying a place. If possible, you should specify the `placeId` rather than a `placeQuery`. A place ID can be retrieved from any request to the Places API, such as a TextSearch.
`query`	`string`	A search query describing the place (such as a business or point of interest). An example query would be "Quay, Upper Level, Overseas Passenger Terminal 5 Hickson Road, The Rocks NSW". If possible, you should specify the `placeId` rather than a `placeQuery`. The API does not guarantee the accuracy of resolving the query string to a place. If both the `placeId` and `placeQuery` are provided, an error is thrown.

google.maps.SaveWidget class

A control that users can use to save a place to Google Maps from your website. In this context, 'place' means a business, point of interest or geographic location. The `SaveWidget` has a fixed height of 22px.

The `SaveWidget` is only available when `signed-in=true` has been passed as a URL parameter in the bootstrap request.

Constructor

Constructor	Description
`SaveWidget(container:Node, opts?:SaveWidgetOptions)`	Creates a new `SaveWidget`, and renders it in a given div.

Methods

Methods	Return Value	Description
`getAttribution()`	`Attribution`	Returns the `Attribution` associated with this `SaveWidget`.
`getPlace()`	`Place`	Returns the `Place` associated with this `SaveWidget`.
`setAttribution(attribution:`*`Attribution`*`)`	`None`	Sets the `Attribution` associated with this `SaveWidget`.
`setOptions(opts:`*`SaveWidgetOptions`*`)`	`None`	
`setPlace(place:`*`Place`*`)`	`None`	Changes the `Place` associated with this `SaveWidget`.

google.maps.SaveWidgetOptions object specification

Properties

Properties	Type	Description
attribution	Attribution	Contains all the information needed to identify your application as the source of a save to Google Maps. The SaveWidget will show this information when a user saves the associated Place.
place	Place	Describes a Place that a user can save from your application to their personalized map. In this context, 'place' means a business, point of interest or geographic location.

google.maps.MapType object specification

This interface defines the map type, and is typically used for custom map types. Immutable.

Methods

Methods	Return Value	Description
getTile(tileCoord:Point, zoom:number, ownerDocument:Document)	Node	Returns a tile for the given tile coordinate (x, y) and zoom level. This tile will be appended to the given ownerDocument. Not available for base map types.
releaseTile(tile:Node)	None	Releases the given tile, performing any necessary cleanup. The provided tile will have already been removed from the document. Optional.

Properties

Properties	Type	Description
alt	string	Alt text to display when this MapType's button is hovered over in the MapTypeControl. Optional.
maxZoom	number	The maximum zoom level for the map when displaying this MapType. Required for base MapTypes, ignored for overlay MapTypes.
minZoom	number	The minimum zoom level for the map when displaying this MapType. Optional; defaults to 0.
name	string	Name to display in the MapTypeControl. Optional.
projection	Projection	The Projection used to render this MapType. Optional; defaults to Mercator.
radius	number	Radius of the planet for the map, in meters. Optional; defaults to Earth's equatorial radius of 6378137 meters.
tileSize	Size	The dimensions of each tile. Required.

google.maps.MapTypeRegistry class

This class extends MVCObject.

Constructor

Constructor	Description
MapTypeRegistry()	The MapTypeRegistry holds the collection of custom map types available to the map for its use. The API consults this registry when providing the list of avaiable map types within controls, for example.

Methods

Methods	Return Value	Description
set(id:string, mapType:MapType|undefined)	None	Sets the registry to associate the passed string identifier with the passed MapType.

google.maps.Projection object specification

Methods

Methods	Return Value	Description
`fromLatLngToPoint(latLng:LatLng, point?:Point)`	`Point`	Translates from the LatLng cylinder to the Point plane. This interface specifies a function which implements translation from given `LatLng` values to world coordinates on the map projection. The Maps API calls this method when it needs to plot locations on screen. `Projection` objects must implement this method.
`fromPointToLatLng(pixel:Point, nowrap?:boolean)`	`LatLng`	This interface specifies a function which implements translation from world coordinates on a map projection to `LatLng` values. The Maps API calls this method when it needs to translate actions on screen to positions on the map. `Projection` objects must implement this method.

google.maps.ImageMapType class

This class implements the MapType interface and is provided for rendering image tiles.

This class extends `MVCObject`.

Constructor

Constructor	Description
`ImageMapType(opts:ImageMapTypeOptions)`	Constructs an ImageMapType using the provided ImageMapTypeOptions

Methods

Methods	Return Value	Description
`getOpacity()`	`number`	Returns the opacity level (`0` (transparent) to `1.0`) of the `ImageMapType` tiles.
`getTile(tileCoord:Point, zoom:number, ownerDocument:Document)`	`Node`	
`releaseTile(tile:Node)`	`None`	
`setOpacity(opacity:number)`	`None`	Sets the opacity level (`0` (transparent) to `1.0`) of the `ImageMapType` tiles.

Properties

Properties	Type	Description
`alt`	`string`	
`maxZoom`	`number`	
`minZoom`	`number`	
`name`	`string`	
`projection`	`Projection`	
`radius`	`number`	
`tileSize`	`Size`	

Events

Events	Arguments	Description
`tilesloaded`	`None`	This event is fired when the visible tiles have finished loading.

google.maps.ImageMapTypeOptions object specification

This class is used to create a MapType that renders image tiles.

Methods

Methods	Return Value	Description
`getTileUrl(coordinate:Point, zoom:number)`	`string`	Returns a string (URL) for given tile coordinate (x, y) and zoom level.

Properties

Properties	Type	Description
`alt`	`string`	Alt text to display when this MapType's button is hovered over in the MapTypeControl.
`maxZoom`	`number`	The maximum zoom level for the map when displaying this MapType.
`minZoom`	`number`	The minimum zoom level for the map when displaying this MapType. Optional.
`name`	`string`	Name to display in the MapTypeControl.
`opacity`	`number`	The opacity to apply to the tiles. The opacity should be specified as a float value between 0 and 1.0, where 0 is fully transparent and 1 is fully opaque.
`tileSize`	`Size`	The tile size.

google.maps.StyledMapType class

Creates a `MapType` with a custom style.

This class extends `MVCObject`.

Constructor

Constructor	Description
`StyledMapType(styles:Array<MapTypeStyle>, options?:StyledMapTypeOptions)`	Creates a styled `MapType` with the specified options. The `StyledMapType` takes an array of `MapTypeStyle`s, where each `MapTypeStyle` is applied to the map consecutively. A later `MapTypeStyle` that applies the same `MapTypeStyler`s to the same selectors as an earlier `MapTypeStyle` will override the earlier `MapTypeStyle`.

Methods

Methods	Return Value	Description
`getTile(tileCoord:Point, zoom:number, ownerDocument:Document)`	`Node`	
`releaseTile(tile:Node)`	`None`	

Properties

Properties	Type	Description
`alt`	`string`	
`maxZoom`	`number`	
`minZoom`	`number`	
`name`	`string`	
`projection`	`Projection`	
`radius`	`number`	
`tileSize`	`Size`	

google.maps.StyledMapTypeOptions object specification

This class is used to specify options when creating a `StyledMapType`. These options cannot be changed after the `StyledMapType` is instantiated.

Properties

Properties	Type	Description
`alt`	`string`	Text to display when this `MapType`'s button is hovered over in the map type control.
`maxZoom`	`number`	The maximum zoom level for the map when displaying this `MapType`. Optional.
`minZoom`	`number`	The minimum zoom level for the map when displaying this `MapType`. Optional.
`name`	`string`	The name to display in the map type control.

google.maps.MapTypeStyle object specification

The `MapTypeStyle` is a collection of selectors and stylers that define how the map should be styled. Selectors specify what map elements should be affected and stylers specify how those elements should be modified.

Properties

Properties	Type	Description
`elementType`	`MapTypeStyleElementType`	Selects the element type to which a styler should be applied. An element type distinguishes between the different representations of a feature. Optional; if `elementType` is not specified, the value is assumed to be `'all'`.
`featureType`	`MapTypeStyleFeatureType`	Selects the feature, or group of features, to which a styler should be applied. Optional; if `featureType` is not specified, the value is assumed to be `'all'`.
`stylers`	`Array<MapTypeStyler>`	The style rules to apply to the selectors. The rules are applied to the map's elements in the order they are listed in this array.

google.maps.MapTypeStyleFeatureType object specification

Possible values for feature types. Specify these values as strings, i.e. `'administrative'` or `'poi.park'`. Stylers applied to a parent feature type automatically apply to all child feature types. Note however that parent features may include some additional features that are not included in one of their child feature types.

Constant

Constant	Description
`administrative`	Apply the rule to administrative areas.
`administrative.country`	Apply the rule to countries.
`administrative.land_parcel`	Apply the rule to land parcels.
`administrative.locality`	Apply the rule to localities.
`administrative.neighborhood`	Apply the rule to neighborhoods.
`administrative.province`	Apply the rule to provinces.
`all`	Apply the rule to all selector types.
`landscape`	Apply the rule to landscapes.
`landscape.man_made`	Apply the rule to man made structures.
`landscape.natural`	Apply the rule to natural features.
`landscape.natural.landcover`	Apply the rule to landcover.
`landscape.natural.terrain`	Apply the rule to terrain.
`poi`	Apply the rule to points of interest.

`poi.attraction`	Apply the rule to attractions for tourists.
`poi.business`	Apply the rule to businesses.
`poi.government`	Apply the rule to government buildings.
`poi.medical`	Apply the rule to emergency services (hospitals, pharmacies, police, doctors, etc).
`poi.park`	Apply the rule to parks.
`poi.place_of_worship`	Apply the rule to places of worship, such as churches, temples, or mosques.
`poi.school`	Apply the rule to schools.
`poi.sports_complex`	Apply the rule to sports complexes.
`road`	Apply the rule to all roads.
`road.arterial`	Apply the rule to arterial roads.
`road.highway`	Apply the rule to highways.
`road.highway.controlled_access`	Apply the rule to controlled-access highways.
`road.local`	Apply the rule to local roads.
`transit`	Apply the rule to all transit stations and lines.
`transit.line`	Apply the rule to transit lines.
`transit.station`	Apply the rule to all transit stations.
`transit.station.airport`	Apply the rule to airports.
`transit.station.bus`	Apply the rule to bus stops.
`transit.station.rail`	Apply the rule to rail stations.
`water`	Apply the rule to bodies of water.

google.maps.MapTypeStyleElementType object specification

Each `MapTypeStyleElementType` distinguishes between the different representations of a feature.

Constant

Constant	Description
`all`	Apply the rule to all elements of the specified feature.
`geometry`	Apply the rule to the feature's geometry.
`geometry.fill`	Apply the rule to the fill of the feature's geometry.
`geometry.stroke`	Apply the rule to the stroke of the feature's geometry.
`labels`	Apply the rule to the feature's labels.
`labels.icon`	Apply the rule to icons within the feature's labels.
`labels.text`	Apply the rule to the text in the feature's label.
`labels.text.fill`	Apply the rule to the fill of the text in the feature's labels.
`labels.text.stroke`	Apply the rule to the stroke of the text in the feature's labels.

google.maps.MapTypeStyler object specification

A styler affects how a map's elements will be styled. Each `MapTypeStyler` should contain one and only one key. If more than one key is specified in a single `MapTypeStyler`, all but one will be ignored. For example: `var rule = {hue: '#ff0000'}`.

Properties

Properties	Type	Description

`color`	`string`	Sets the color of the feature. Valid values: An RGB hex string, i.e. `'#ff0000'`.
`gamma`	`number`	Modifies the gamma by raising the lightness to the given power. Valid values: Floating point numbers, [0.01, 10], with 1.0 representing no change.
`hue`	`string`	Sets the hue of the feature to match the hue of the color supplied. Note that the saturation and lightness of the feature is conserved, which means that the feature will not match the color supplied exactly. Valid values: An RGB hex string, i.e. `'#ff0000'`.
`invert_lightness`	`boolean`	A value of `true` will invert the lightness of the feature while preserving the hue and saturation.
`lightness`	`number`	Shifts lightness of colors by a percentage of the original value if decreasing and a percentage of the remaining value if increasing. Valid values: [-100, 100].
`saturation`	`number`	Shifts the saturation of colors by a percentage of the original value if decreasing and a percentage of the remaining value if increasing. Valid values: [-100, 100].
`visibility`	`string`	Sets the visibility of the feature. Valid values: 'on', 'off' or 'simplifed'.
`weight`	`number`	Sets the weight of the feature, in pixels. Valid values: Integers greater than or equal to zero.

google.maps.BicyclingLayer class

A layer showing bike lanes and paths.

This class extends `MVCObject`.

Constructor

Constructor	Description
`BicyclingLayer()`	A layer that displays bike lanes and paths and demotes large roads.

Methods

Methods	Return Value	Description
`getMap()`	`Map`	Returns the map on which this layer is displayed.
`setMap(map:Map)`	`None`	Renders the layer on the specified map. If map is set to null, the layer will be removed.

google.maps.FusionTablesLayer class

A `FusionTablesLayer` allows you to display data from a Google Fusion Table on a map, as a rendered layer. (See https://developers.google.com/fusiontables/ for more information about Fusion Tables).

This class extends `MVCObject`.

Constructor

Constructor	Description
`FusionTablesLayer(options:`*`FusionTablesLayerOptions`*`)`	A layer that displays data from a Fusion Table.

Methods

Methods	Return Value	Description
`getMap()`	`Map`	Returns the map on which this layer is displayed.
`setMap(map:Map)`	`None`	Renders the layer on the specified map. If map is set to null, the layer will be removed.
`setOptions(options:`*`FusionTablesLayerOptions`*`)`	`None`	

Events

Events	Arguments	Description
click	*FusionTablesMouseEvent*	This event is fired when a feature in the layer is clicked.

google.maps.FusionTablesLayerOptions object specification

This object defines the properties that can be set on a FusionTablesLayer object.

Properties

Properties	Type	Description
clickable	boolean	If true, the layer receives mouse events. Default value is true.
heatmap	FusionTablesHeatmap	Options which define the appearance of the layer as a heatmap.
map	Map	The map on which to display the layer.
query	FusionTablesQuery	Options defining the data to display.
styles	Array<FusionTablesStyle>	An array of up to 5 style specifications, which control the appearance of features within the layer.
suppressInfoWindows	boolean	Suppress the rendering of info windows when layer features are clicked.

google.maps.FusionTablesQuery object specification

Specifies the data to retrieve from a Fusion Tables.

Properties

Properties	Type	Description
from	string	The ID of the Fusion Tables table to display. This ID can be found in the table's URL, as the value of the dsrcid parameter. Required.
limit	number	Limit on the number of results returned by the query.
offset	number	Offset into the sorted results.
orderBy	string	The method by which to sort the results. Accepts either of: • A column name. The column name may be suffixed with ASC or DESC (e.g. col2 DESC) to specify ascending or descending sort. • An ST_DISTANCE spatial relationship (sort by distance). A column and the coordinate from which to calculate distance must be passed, for example, orderBy: 'ST_DISTANCE(col1, LATLNG(1.2, 3.4))'.
select	string	A column, containing geographic features to be displayed on the map. See Fusion Tables Setup in the Maps API documentation for information about valid columns.
where	string	The SQL predicate to be applied to the layer.

google.maps.FusionTablesStyle object specification

Controls the appearance of a set of features within a FusionTablesLayer. Features which match the provided SQL predicate will be styled with the supplied options.

Properties

Properties	Type	Description
markerOptions	FusionTablesMarkerOptions	Options which control the appearance of point features.
polygonOptions	FusionTablesPolygonOptions	Options which control the appearance of polygons.
polylineOptions	FusionTablesPolylineOptions	Options which control the appearance of polylines.
where	string	The SQL predicate to be applied to the layer.

google.maps.FusionTablesHeatmap object specification

Specifies the appearance for a FusionTablesLayer when rendered as a heatmap.

Properties

Properties	Type	Description
`enabled`	`boolean`	If true, render the layer as a heatmap.

google.maps.FusionTablesMarkerOptions object specification

Options which control the appearance of point features in a `FusionTablesLayer`.

Properties

Properties	Type	Description
`iconName`	`string`	The name of a Fusion Tables supported icon

google.maps.FusionTablesPolygonOptions object specification

Options which control the appearance of polygons in a `FusionTablesLayer`.

Properties

Properties	Type	Description
`fillColor`	`string`	The fill color, defined by a six-digit hexadecimal number in RRGGBB format (e.g. `#00AAFF`).
`fillOpacity`	`number`	The fill opacity between 0.0 and 1.0.
`strokeColor`	`string`	The fill color, defined by a six-digit hexadecimal number in RRGGBB format (e.g. `#00AAFF`).
`strokeOpacity`	`number`	The stroke opacity between 0.0 and 1.0.
`strokeWeight`	`number`	The stroke width in pixels, between 0 and 10.

google.maps.FusionTablesPolylineOptions object specification

Options which control the appearance of polylines in a `FusionTablesLayer`.

Properties

Properties	Type	Description
`strokeColor`	`string`	The fill color, defined by a six-digit hexadecimal number in RRGGBB format (e.g. `#00AAFF`).
`strokeOpacity`	`number`	The stroke opacity between 0.0 and 1.0.
`strokeWeight`	`number`	The stroke width in pixels.

google.maps.FusionTablesMouseEvent object specification

The properties of a mouse event on a `FusionTablesLayer`.

Properties

Properties	Type	Description
`infoWindowHtml`	`string`	Pre-rendered HTML content, as placed in the infowindow by the default UI.
`latLng`	`LatLng`	The position at which to anchor an infowindow on the clicked feature.
`pixelOffset`	`Size`	The offset to apply to an infowindow anchored on the clicked feature.

row	Object<FusionTablesCell>	A collection of FusionTablesCell objects, indexed by column name, representing the contents of the table row which included the clicked feature.

google.maps.FusionTablesCell object specification

Describes a single cell from a Fusion Tables table.

Properties

Properties	Type	Description
columnName	string	The name of the column in which the cell was located.
value	string	The contents of the cell.

google.maps.KmlLayer class

A KmlLayer adds geographic markup to the map from a KML, KMZ or GeoRSS file that is hosted on a publicly accessible web server. A KmlFeatureData object is provided for each feature when clicked.

This class extends MVCObject.

Constructor

Constructor	Description
KmlLayer(opts?:*KmlLayerOptions*)	Creates a KmlLayer which renders the contents of the specified KML/KMZ file (https://developers.google.com/kml/documentation/kmlreference) or GeoRSS file (http://www.georss.org).

Methods

Methods	Return Value	Description
getDefaultViewport()	LatLngBounds	Get the default viewport for the layer being displayed.
getMap()	Map	Get the map on which the KML Layer is being rendered.
getMetadata()	KmlLayerMetadata	Get the metadata associated with this layer, as specified in the layer markup.
getStatus()	KmlLayerStatus	Get the status of the layer, set once the requested document has loaded.
getUrl()	string	Gets the URL of the KML file being displayed.
getZIndex()	number	Gets the z-index of the KML Layer.
setMap(map:Map)	None	Renders the KML Layer on the specified map. If map is set to null, the layer is removed.
setUrl(url:string)	None	Sets the URL of the KML file to display.
setZIndex(zIndex:number)	None	Sets the z-index of the KML Layer.

Events

Events	Arguments	Description
click	*KmlMouseEvent*	This event is fired when a feature in the layer is clicked.
defaultviewport_changed	None	This event is fired when the KML layers default viewport has changed.
status_changed	None	This event is fired when the KML layer has finished loading. At this point it is safe to read the status property to determine if the layer loaded successfully.

google.maps.KmlLayerOptions object specification

This object defines the properties that can be set on a KmlLayer object.

Properties

Properties	Type	Description
clickable	boolean	If true, the layer receives mouse events. Default value is true.
map	Map	The map on which to display the layer.
preserveViewport	boolean	By default, the input map is centered and zoomed to the bounding box of the contents of the layer. If this option is set to true, the viewport is left unchanged, unless the map's center and zoom were never set.
screenOverlays	boolean	Whether to render the screen overlays. Default true.
suppressInfoWindows	boolean	Suppress the rendering of info windows when layer features are clicked.
url	string	The URL of the KML document to display.
zIndex	number	The z-index of the layer.

google.maps.KmlLayerMetadata object specification

Metadata for a single KML layer, in JSON format.

Properties

Properties	Type	Description
author	KmlAuthor	The layer's <atom:author>, extracted from the layer markup.
description	string	The layer's <description>, extracted from the layer markup.
hasScreenOverlays	boolean	Whether the layer has any screen overlays.
name	string	The layer's <name>, extracted from the layer markup.
snippet	string	The layer's <Snippet>, extracted from the layer markup

google.maps.KmlLayerStatus class

The status returned by KmlLayer on the completion of loading a document.

Constant

Constant	Description
DOCUMENT_NOT_FOUND	The document could not be found. Most likely it is an invalid URL, or the document is not publicly available.
DOCUMENT_TOO_LARGE	The document exceeds the file size limits of KmlLayer.
FETCH_ERROR	The document could not be fetched.
INVALID_DOCUMENT	The document is not a valid KML, KMZ or GeoRSS document.
INVALID_REQUEST	The KmlLayer is invalid.
LIMITS_EXCEEDED	The document exceeds the feature limits of KmlLayer.
OK	The layer loaded successfully.
TIMED_OUT	The document could not be loaded within a reasonable amount of time.
UNKNOWN	The document failed to load for an unknown reason.

google.maps.KmlMouseEvent object specification

The properties of a click event on a KML/KMZ or GeoRSS document.

Properties

Properties	Type	Description

featureData	KmlFeatureData	A KmlFeatureData object, containing information about the clicked feature.
latLng	LatLng	The position at which to anchor an infowindow on the clicked feature.
pixelOffset	Size	The offset to apply to an infowindow anchored on the clicked feature.

google.maps.KmlFeatureData object specification

Data for a single KML feature in JSON format, returned when a KML feature is clicked. The data contained in this object mirrors that associated with the feature in the KML or GeoRSS markup in which it is declared.

Properties

Properties	Type	Description
author	KmlAuthor	The feature's <atom:author>, extracted from the layer markup (if specified).
description	string	The feature's <description>, extracted from the layer markup.
id	string	The feature's <id>, extracted from the layer markup. If no <id> has been specified, a unique ID will be generated for this feature.
infoWindowHtml	string	The feature's balloon styled text, if set.
name	string	The feature's <name>, extracted from the layer markup.
snippet	string	The feature's <Snippet>, extracted from the layer markup.

google.maps.KmlAuthor object specification

Contains details of the author of a KML document or feature.

Properties

Properties	Type	Description
email	string	The author's e-mail address, or an empty string if not specified.
name	string	The author's name, or an empty string if not specified.
uri	string	The author's home page, or an empty string if not specified.

google.maps.TrafficLayer class

A traffic layer.

This class extends MVCObject.

Constructor

Constructor	Description
TrafficLayer()	A layer that displays current road traffic.

Methods

Methods	Return Value	Description
getMap()	Map	Returns the map on which this layer is displayed.
setMap(map:Map)	None	Renders the layer on the specified map. If map is set to null, the layer will be removed.

google.maps.TransitLayer class

A transit layer.

This class extends `MVCObject`.

Constructor

Constructor	Description
`TransitLayer()`	A layer that displays transit lines.

Methods

Methods	Return Value	Description
`getMap()`	`Map`	Returns the map on which this layer is displayed.
`setMap(map:Map)`	`None`	Renders the layer on the specified map. If map is set to null, the layer will be removed.

google.maps.StreetViewPanorama class

Displays the panorama for a given `LatLng` or panorama ID. A `StreetViewPanorama` object provides a Street View "viewer" which can be stand-alone within a separate `<div>` or bound to a `Map`.

This class extends `MVCObject`.

Constructor

Constructor	Description
`StreetViewPanorama(container:Node, opts?:`*`StreetViewPanoramaOptions`*`)`	Creates a panorama with the passed `StreetViewPanoramaOptions`.

Methods

Methods	Return Value	Description
`getLinks()`	`Array<StreetViewLink>`	Returns the set of navigation links for the Street View panorama.
`getLocation()`	`StreetViewLocation`	Returns the StreetViewLocation of the current panorama.
`getPano()`	`string`	Returns the current panorama ID for the Street View panorama. This id is stable within the browser's current session only.
`getPhotographerPov()`	`StreetViewPov`	Returns the heading and pitch of the photographer when this panorama was taken. For Street View panoramas on the road, this also reveals in which direction the car was travelling. This data is available after the `pano_changed` event.
`getPosition()`	`LatLng`	Returns the current `LatLng` position for the Street View panorama.
`getPov()`	`StreetViewPov`	Returns the current point of view for the Street View panorama.

`getStatus()`	`StreetViewStatus`	Returns the status of the panorama on completion of the `setPosition()` or `setPano()` request.
`getVisible()`	`boolean`	Returns `true` if the panorama is visible. It does not specify whether Street View imagery is available at the specified position.
`getZoom()`	`number`	Returns the zoom level of the panorama. Fully zoomed-out is level 0, where the field of view is 180 degrees. Zooming in increases the zoom level.
`registerPanoProvider(provider:function(string):StreetViewPanoramaData)`	`None`	Set the custom panorama provider called on pano change to load custom panoramas.
`setLinks(links:Array<StreetViewLink>)`	`None`	Sets the set of navigation links for the Street View panorama.
`setOptions(options:`*`StreetViewPanoramaOptions`*`)`	`None`	Sets a collection of key-value pairs.
`setPano(pano:string)`	`None`	Sets the current panorama ID for the Street View panorama.
`setPosition(latLng:LatLng|LatLngLiteral)`	`None`	Sets the current `LatLng` position for the Street View panorama.
`setPov(pov:`*`StreetViewPov`*`)`	`None`	Sets the point of view for the Street View panorama.
`setVisible(flag:boolean)`	`None`	Sets to `true` to make the panorama visible. If set to `false`, the panorama will be hidden whether it is embedded in the map or in its own `<div>`.
`setZoom(zoom:number)`	`None`	Sets the zoom level of the panorama. Fully zoomed-out is level 0, where the field of view is 180 degrees. Zooming in increases the zoom level.

Properties

Properties	Type	Description
`controls`	`Array<MVCArray<Node>>`	Additional controls to attach to the panorama. To add a control to the panorama, add the control's `<div>` to the `MVCArray` corresponding to the `ControlPosition` where it should be rendered.

Events

Events	Arguments	Description
`clicktogo_changed`	`None`	This event is fired when the panorama's clickToGo is enabled or disabled.
`closeclick`	`Event`	This event is fired when the close button is clicked.

links_changed	None	This event is fired when the panorama's links change. The links change asynchronously following a pano id change.
pano_changed	None	This event is fired when the panorama's pano id changes. The pano may change as the user navigates through the panorama or the position is manually set. Note that not all position changes trigger a pano_changed.
position_changed	None	This event is fired when the panorama's position changes. The position changes as the user navigates through the panorama or the position is set manually.
pov_changed	None	This event is fired when the panorama's point-of-view changes. The point of view changes as the pitch, zoom, or heading changes.
resize	None	Developers should trigger this event on the panorama when its div changes size: google.maps.event.trigger(panorama, 'resize').
scrollwheel_changed	None	This event is fired when the panorama's scrollWheel is enabled or disabled.
status_changed	None	This event is fired after every panorama lookup by id or location, via setPosition() or setPano().
visible_changed	None	This event is fired when the panorama's visibility changes. The visibility is changed when the Pegman is dragged onto the map, the close button is clicked, or setVisible() is called.
zoom_changed	None	This event is fired when the panorama's zoom level changes.

google.maps.StreetViewPanoramaOptions object specification

Options defining the properties of a StreetViewPanorama object.

Methods

Methods	Return Value	Description
panoProvider(panoId:string)	StreetViewPanoramaData	Custom panorama provider, which takes a string pano id and returns an object defining the panorama given that id. This function must be defined to specify custom panorama imagery.

Properties

Properties	Type	Description
addressControl	boolean	The enabled/disabled state of the address control.
addressControlOptions	StreetViewAddressControlOptions	The display options for the address control.
clickToGo	boolean	The enabled/disabled state of click-to-go.
disableDefaultUI	boolean	Enables/disables all default UI. May be overridden individually.
disableDoubleClickZoom	boolean	Enables/disables zoom on double click. Disabled by default.
enableCloseButton	boolean	If true, the close button is displayed. Disabled by default.
imageDateControl	boolean	The enabled/disabled state of the imagery acquisition date control. Disabled by default.
linksControl	boolean	The enabled/disabled state of the links control.
panControl	boolean	The enabled/disabled state of the pan control.
panControlOptions	PanControlOptions	The display options for the pan control.
pano	string	The panorama ID, which should be set when specifying a custom panorama.
position	LatLng|LatLngLiteral	The LatLng position of the Street View panorama.
pov	StreetViewPov	The camera orientation, specified as heading and pitch, for the panorama.
scrollwheel	boolean	If false, disables scrollwheel zooming in Street View. The scrollwheel is enabled by default.

`visible`	`boolean`	If `true`, the Street View panorama is visible on load.
`zoomControl`	`boolean`	The enabled/disabled state of the zoom control.
`zoomControlOptions`	`ZoomControlOptions`	The display options for the zoom control.

google.maps.StreetViewAddressControlOptions object specification

Options for the rendering of the Street View address control.

Properties

Properties	Type	Description
`position`	`ControlPosition`	Position id. This id is used to specify the position of the control on the map. The default position is `TOP_LEFT`.

google.maps.StreetViewLink object specification

A collection of references to adjacent Street View panos.

Properties

Properties	Type	Description
`description`	`string`	A localized string describing the link.
`heading`	`number`	The heading of the link.
`pano`	`string`	A unique identifier for the panorama. This id is stable within a session but unstable across sessions.

google.maps.StreetViewPov object specification

A point of view object which specifies the camera's orientation at the Street View panorama's position. The point of view is defined as heading and pitch.

Properties

Properties	Type	Description
`heading`	`number`	The camera heading in degrees relative to true north. True north is 0°, east is 90°, south is 180°, west is 270°.
`pitch`	`number`	The camera pitch in degrees, relative to the street view vehicle. Ranges from 90° (directly upwards) to -90° (directly downwards).

google.maps.StreetViewPanoramaData object specification

The representation of a panorama returned from the provider defined using `registerPanoProvider`.

Properties

Properties	Type	Description
`copyright`	`string`	Specifies the copyright text for this panorama.
`imageDate`	`string`	Specifies the year and month in which the imagery in this panorama was acquired. The date string is in the form YYYY-MM.
`links`	`Array<StreetViewLink>`	Specifies the navigational links to adjacent panoramas.
`location`	`StreetViewLocation`	Specifies the location meta-data for this panorama.
`tiles`	`StreetViewTileData`	Specifies the custom tiles for this panorama.

google.maps.StreetViewLocation object specification

A representation of a location in the Street View panorama.

Properties

Properties	Type	Description
`description`	`string`	A localized string describing the location.
`latLng`	`LatLng`	The latlng of the panorama.
`pano`	`string`	A unique identifier for the panorama. This is stable within a session but unstable across sessions.
`shortDescription`	`string`	Short description of the location.

google.maps.StreetViewTileData object specification

The properties of the tile set used in a Street View panorama.

Methods

Methods	Return Value	Description
`getTileUrl(pano:string, tileZoom:number, tileX:number, tileY:number)`	`string`	Gets the tile image URL for the specified tile. `pano` is the panorama ID of the Street View tile. `tileZoom` is the zoom level of the tile. `tileX` is the x-coordinate of the tile. `tileY` is the y-coordinate of the tile. Returns the URL for the tile image.

Properties

Properties	Type	Description
`centerHeading`	`number`	The heading (in degrees) at the center of the panoramic tiles.
`tileSize`	`Size`	The size (in pixels) at which tiles will be rendered.
`worldSize`	`Size`	The size (in pixels) of the whole panorama's "world".

google.maps.StreetViewService class

A `StreetViewService` object performs searches for Street View data.

Methods

Methods	Return Value	Description
`getPanoramaById(pano:string, callback:function(StreetViewPanoramaData, StreetViewStatus))`	`None`	Retrieves the data for the given pano id and passes it to the provided callback as a `StreetViewPanoramaData` object. Pano ids are unique per panorama and stable for the lifetime of a session, but are liable to change between sessions.
`getPanoramaByLocation(latlng:LatLng|LatLngLiteral, radius:number, callback:function(StreetViewPanoramaData, StreetViewStatus))`	`None`	Retrieves the `StreetViewPanoramaData` for a panorama within a given radius of the given `LatLng`. The `StreetViewPanoramaData` is passed to the provided callback. If the radius is less than 50 meters, the nearest panorama will be returned.

google.maps.StreetViewStatus class

The status returned by the `StreetViewService` on completion of a Street View request.

Constant

Constant	Description
`OK`	The request was successful.
`UNKNOWN_ERROR`	The request could not be successfully processed, yet the exact reason for failure is unknown.
`ZERO_RESULTS`	There are no nearby panoramas.

google.maps.StreetViewCoverageLayer class

A layer that illustrates the locations where Street View is available.

This class extends `MVCObject`.

Constructor

Constructor	Description
`StreetViewCoverageLayer()`	Creates a new instance of `StreetViewCoverageLayer`.

Methods

Methods	Return Value	Description
`getMap()`	`Map`	Returns the map on which this layer is displayed.
`setMap(map:Map)`	`None`	Renders the layer on the specified map. If the map is set to null, the layer will be removed.

google.maps.MapsEventListener object specification

This class is opaque. It has no methods and no constructor. Its instances are returned from addListener(), addDomListener() and are eventually passed back to removeListener().

google.maps.event namespace

Static Methods

Methods	Return Value	Description
`addDomListener(instance:Object, eventName:string, handler:Function, capture?:boolean)`	`MapsEventListener`	Cross browser event handler registration. This listener is removed by calling removeListener(handle) for the handle that is returned by this function.
`addDomListenerOnce(instance:Object, eventName:string, handler:Function, capture?:boolean)`	`MapsEventListener`	Wrapper around addDomListener that removes the listener after the first event.
`addListener(instance:Object, eventName:string, handler:Function)`	`MapsEventListener`	Adds the given listener function to the given event name for the given object instance. Returns an identifier for this listener that can be used with removeListener().
`addListenerOnce(instance:Object, eventName:string, handler:Function)`	`MapsEventListener`	Like addListener, but the handler removes itself after handling the first event.
`clearInstanceListeners(instance:Object)`	`None`	Removes all listeners for all events for the given instance.
`clearListeners(instance:Object, eventName:string)`	`None`	Removes all listeners for the given event for the given instance.
`removeListener(listener:MapsEventListener)`	`None`	Removes the given listener, which should have been returned by addListener above.
`trigger(instance:Object, eventName:string, var_args:*)`	`None`	Triggers the given event. All arguments after eventName are passed as arguments to the listeners.

google.maps.MouseEvent object specification

This object is returned from various mouse events on the map and overlays, and contains all the fields shown below.

Methods

Methods	Return Value	Description
`stop()`	`None`	Prevents this event from propagating further.

Properties

Properties	Type	Description
`latLng`	`LatLng`	The latitude/longitude that was below the cursor when the event occurred.

google.maps.LatLng class

A `LatLng` is a point in geographical coordinates: latitude and longitude.

- Latitude ranges between -90 and 90 degrees, inclusive. Values above or below this range will be clamped to the range [-90, 90]. This means that if the value specified is less than -90, it will be set to -90. And if the value is greater than 90, it will be set to 90.
- Longitude ranges between -180 and 180 degrees, inclusive. Values above or below this range will be wrapped so that they fall within the range. For example, a value of -190 will be converted to 170. A value of 190 will be converted to -170. This reflects the fact that longitudes wrap around the globe.

Although the default map projection associates longitude with the x-coordinate of the map, and latitude with the y-coordinate, the latitude coordinate is always written *first*, followed by the longitude.
Notice that you cannot modify the coordinates of a `LatLng`. If you want to compute another point, you have to create a new one.

Constructor

Constructor	Description
`LatLng(lat:number, lng:number, noWrap?:boolean)`	Creates a `LatLng` object representing a geographic point. Latitude is specified in degrees within the range [-90, 90]. Longitude is specified in degrees within the range [-180, 180]. Set `noWrap` to `true` to enable values outside of this range. Note the ordering of latitude and longitude.

Methods

Methods	Return Value	Description
`equals(other:LatLng)`	`boolean`	Comparison function.
`lat()`	`number`	Returns the latitude in degrees.
`lng()`	`number`	Returns the longitude in degrees.
`toString()`	`string`	Converts to string representation.
`toUrlValue(precision?:number)`	`string`	Returns a string of the form "lat,lng" for this LatLng. We round the lat/lng values to 6 decimal places by default.

google.maps.LatLngLiteral object specification

Object literals are accepted in place of `LatLng` objects, as a convenience, in many places. These are converted to `LatLng` objects when the Maps API encounters them.

Examples:

```
map.setCenter({lat: -34, lng: 151});
new google.maps.Marker({position: {lat: -34, lng: 151}, map: map});
```

> LatLng object literals are not supported in the Geometry library.

Properties

Properties	Type	Description
lat	number	Latitude in degrees. Values will be clamped to the range [-90, 90]. This means that if the value specified is less than -90, it will be set to -90. And if the value is greater than 90, it will be set to 90.
lng	number	Longitude in degrees. Values outside the range [-180, 180] will be wrapped so that they fall within the range. For example, a value of -190 will be converted to 170. A value of 190 will be converted to -170. This reflects the fact that longitudes wrap around the globe.

google.maps.LatLngBounds class

A LatLngBounds instance represents a rectangle in geographical coordinates, including one that crosses the 180 degrees longitudinal meridian.

Constructor

Constructor	Description
LatLngBounds(sw?:LatLng, ne?:LatLng)	Constructs a rectangle from the points at its south-west and north-east corners.

Methods

Methods	Return Value	Description
contains(latLng:LatLng)	boolean	Returns true if the given lat/lng is in this bounds.
equals(other:LatLngBounds)	boolean	Returns true if this bounds approximately equals the given bounds.
extend(point:LatLng)	LatLngBounds	Extends this bounds to contain the given point.
getCenter()	LatLng	Computes the center of this LatLngBounds
getNorthEast()	LatLng	Returns the north-east corner of this bounds.
getSouthWest()	LatLng	Returns the south-west corner of this bounds.
intersects(other:LatLngBounds)	boolean	Returns true if this bounds shares any points with this bounds.
isEmpty()	boolean	Returns if the bounds are empty.
toSpan()	LatLng	Converts the given map bounds to a lat/lng span.
toString()	string	Converts to string.
toUrlValue(precision?:number)	string	Returns a string of the form "lat_lo,lng_lo,lat_hi,lng_hi" for this bounds, where "lo" corresponds to the southwest corner of the bounding box, while "hi" corresponds to the northeast corner of that box.
union(other:LatLngBounds)	LatLngBounds	Extends this bounds to contain the union of this and the given bounds.

google.maps.Point class

Constructor

Constructor	Description
Point(x:number, y:number)	A point on a two-dimensional plane.

Methods

Methods	Return Value	Description
equals(other:Point)	boolean	Compares two Points
toString()	string	Returns a string representation of this Point.

Properties

Properties	Type	Description

x	number	The X coordinate
y	number	The Y coordinate

google.maps.Size class

Constructor

Constructor	Description
Size(width:number, height:number, widthUnit?:string, heightUnit?:string)	Two-dimensonal size, where width is the distance on the x-axis, and height is the distance on the y-axis.

Methods

Methods	Return Value	Description
equals(other:Size)	boolean	Compares two Sizes.
toString()	string	Returns a string representation of this Size.

Properties

Properties	Type	Description
height	number	The height along the y-axis, in pixels.
width	number	The width along the x-axis, in pixels.

google.maps.MVCObject class

Base class implementing KVO.

The MVCObject constructor is guaranteed to be an empty function, and so you may inherit from MVCObject by simply writing MySubclass.prototype = new google.maps.MVCObject();. Unless otherwise noted, this is not true of other classes in the API, and inheriting from other classes in the API is not supported.

Constructor

Constructor	Description
MVCObject()	Creates an MVCObject.

Methods

Methods	Return Value	Description
addListener(eventName:string, handler:Function)	MapsEventListener	Adds the given listener function to the given event name. Returns an identifier for this listener that can be used with google.maps.event.removeListener.
bindTo(key:string, target:MVCObject, targetKey?:string, noNotify?:boolean)	None	Binds a View to a Model.
changed(key:string)	None	Generic handler for state changes. Override this in derived classes to handle arbitrary state changes.
get(key:string)	*	Gets a value.
notify(key:string)	None	Notify all observers of a change on this property. This notifies both objects that are bound to the object's property as well as the object that it is bound to.
set(key:string, value:*)	None	Sets a value.
setValues(values:Object|undefined)	None	Sets a collection of key-value pairs.
unbind(key:string)	None	Removes a binding. Unbinding will set the unbound property to the

		current value. The object will not be notified, as the value has not changed.
`unbindAll()`	`None`	Removes all bindings.

google.maps.MVCArray class

This class extends `MVCObject`.

Constructor

Constructor	Description
`MVCArray(array?:Array)`	A mutable MVC Array.

Methods

Methods	Return Value	Description
`clear()`	`None`	Removes all elements from the array.
`forEach(callback:function(*, number))`	`None`	Iterate over each element, calling the provided callback. The callback is called for each element like: callback(element, index).
`getArray()`	`Array`	Returns a reference to the underlying Array. Warning: if the Array is mutated, no events will be fired by this object.
`getAt(i:number)`	`*`	Returns the element at the specified index.
`getLength()`	`number`	Returns the number of elements in this array.
`insertAt(i:number, elem:*)`	`None`	Inserts an element at the specified index.
`pop()`	`*`	Removes the last element of the array and returns that element.
`push(elem:*)`	`number`	Adds one element to the end of the array and returns the new length of the array.
`removeAt(i:number)`	`*`	Removes an element from the specified index.
`setAt(i:number, elem:*)`	`None`	Sets an element at the specified index.

Events

Events	Arguments	Description
`insert_at`	`number`	This event is fired when `insertAt()` is called. The event passes the index that was passed to `insertAt()`.
`remove_at`	`number, *`	This event is fired when `removeAt()` is called. The event passes the index that was passed to `removeAt()` and the element that was removed from the array.
`set_at`	`number, *`	This event is fired when `setAt()` is called. The event passes the index that was passed to `setAt()` and the element that was previously in the array at that index.

google.maps.geometry.encoding namespace

Utilities for polyline encoding and decoding.

Library

geometry

Static Methods

Methods	Return Value	Description
`decodePath(encodedPath:string)`	`Array<LatLng>`	Decodes an encoded path string into a sequence of LatLngs.
`encodePath(path:Array<LatLng>|MVCArray<LatLng>)`	`string`	Encodes a sequence of LatLngs into an encoded path string.

google.maps.geometry.spherical namespace

Utility functions for computing geodesic angles, distances and areas. The default radius is Earth's radius of 6378137 meters.

Library

geometry

Static Methods

Methods	Return Value	Description
`computeArea(path:Array<LatLng>|MVCArray<LatLng>, radius?:number)`	`number`	Returns the area of a closed path. The computed area uses the same units as the radius. The radius defaults to the Earth's radius in meters, in which case the area is in square meters.
`computeDistanceBetween(from:LatLng, to:LatLng, radius?:number)`	`number`	Returns the distance between two LatLngs.
`computeHeading(from:LatLng, to:LatLng)`	`number`	Returns the heading from one LatLng to another LatLng. Headings are expressed in degrees clockwise from North within the range [-180,180).
`computeLength(path:Array<LatLng>|MVCArray<LatLng>, radius?:number)`	`number`	Returns the length of the given path.
`computeOffset(from:LatLng, distance:number, heading:number, radius?:number)`	`LatLng`	Returns the LatLng resulting from moving a distance from an origin in the specified heading (expressed in degrees clockwise from north).
`computeOffsetOrigin(to:LatLng, distance:number, heading:number, radius?:number)`	`LatLng`	Returns the location of origin when provided with a LatLng destination, meters travelled and original heading. Headings are expressed in degrees clockwise from North. This function returns null when no solution is available.
`computeSignedArea(loop:Array<LatLng>|MVCArray<LatLng>, radius?:number)`	`number`	Returns the signed area of a closed path. The signed area may be used to determine the orientation of the path. The computed area uses the same units as the radius. The radius defaults to the Earth's radius in meters, in which case the area is in square meters.
`interpolate(from:LatLng, to:LatLng, fraction:number)`	`LatLng`	Returns the LatLng which lies the given fraction of the way between the origin LatLng and the destination LatLng.

google.maps.geometry.poly namespace

Utility functions for computations involving polygons and polylines.

Library

geometry

Static Methods

Methods	Return Value	Description
`containsLocation(point:LatLng, polygon:Polygon)`	`boolean`	Computes whether the given point lies inside the specified polygon.
`isLocationOnEdge(point:LatLng, poly:Polygon|Polyline, tolerance?:number)`	`boolean`	Computes whether the given point lies on or near to a polyline, or the edge of a polygon, within a specified tolerance. Returns `true` when the difference between the latitude and longitude of the supplied point, and the closest point on the edge, is less than the tolerance. The tolerance defaults to 10^{-9} degrees.

google.maps.adsense.AdUnit class

Implements AdSense for Content advertising on an associated map. To use an `AdUnit`, you must obtain and specify an AdSense for Content publisher ID within the `AdUnit`'s constructor options.

This class extends `MVCObject`.

Library

adsense

Constructor

Constructor	Description
`AdUnit(container:Node, opts:AdUnitOptions)`	Creates an AdSense for Content display ad on the associated map.

Methods

Methods	Return Value	Description
`getBackgroundColor()`	`string`	Returns the `AdUnit`'s background color.
`getBorderColor()`	`string`	Returns the `AdUnit`'s border color.
`getChannelNumber()`	`string`	Returns the channel number in use by this `AdUnit`.
`getContainer()`	`Node`	Returns the containing element of the AdUnit.
`getFormat()`	`AdFormat`	Returns the format in use by this `AdUnit`.
`getMap()`	`Map`	Returns the map to which this `AdUnit`'s ads are targeted.
`getPosition()`	`ControlPosition`	Returns the `ControlPosition` at which this `AdUnit` is displayed on the map.
`getPublisherId()`	`string`	Returns the specified AdSense For Content publisher ID.
`getTextColor()`	`string`	Returns the `AdUnit`'s text color.
`getTitleColor()`	`string`	Returns the `AdUnit`'s title color.
`getUrlColor()`	`string`	Returns the `AdUnit`'s URL color.
`setBackgroundColor(backgroundColor:string)`	`None`	Sets the `AdUnit`'s background color.
`setBorderColor(borderColor:string)`	`None`	Sets the `AdUnit`'s border color.
`setChannelNumber(channelNumber:string)`	`None`	Specifies the channel number for this `AdUnit`. Channel numbers are optional and can be created for Google AdSense tracking.
`setFormat(format:AdFormat)`	`None`	Specifies the display format for this `AdUnit`.
`setMap(map:Map)`	`None`	Associates this `AdUnit` with the specified map. Ads will be targeted to the map's viewport. The map must be specified in order to display ads.
`setPosition(position:ControlPosition)`	`None`	Sets the `ControlPosition` at which to display the `AdUnit` on the map. If the position is set to null, the `AdUnit` is removed from the map.
`setTextColor(textColor:string)`	`None`	Sets the `AdUnit`'s text color.
`setTitleColor(titleColor:string)`	`None`	Sets the `AdUnit`'s title color.
`setUrlColor(urlColor:string)`	`None`	Sets the `AdUnit`'s URL color.

google.maps.adsense.AdUnitOptions object specification

Library

adsense

Properties

Properties	Type	Description
backgroundColor	string	The AdUnit's background color. (Optional)
borderColor	string	The AdUnit's border color. (Optional)
channelNumber	string	The AdSense For Content channel number for tracking the performance of this AdUnit. It must be stored as a string as it will typically be a large UINT64. (Optional)
format	AdFormat	the Format of the AdUnit. See https://google.com/adsense/adformats. (Optional)
map	Map	The map associated with this AdUnit. Ads will be targeted to the location the map's viewport. (Required)
position	ControlPosition	The position of the AdUnit. If specified, the AdUnit will be displayed at this position. Otherwise, it will not be added to the map. (Optional)
publisherId	string	Your AdSense for Content publisher ID. Required and must be set at the time of initialization. (Required)
textColor	string	The AdUnit's text color. (Optional)
titleColor	string	The AdUnit's title color. (Optional)
urlColor	string	The AdUnit's URL color. (Optional)

google.maps.adsense.AdFormat class

Identifiers used to specify an AdSense For Content format. See https://google.com/adsense/adformats.

Library

adsense

Constant

Constant	Description
BANNER	A horizontal "banner" ad. (468x60px)
BUTTON	A small ad. (125x125px)
HALF_BANNER	A smaller horizontal "banner" ad. (234x60px)
LARGE_HORIZONTAL_LINK_UNIT	A large horizontal ad link unit. (728x15px)
LARGE_RECTANGLE	A large rectangular ad. (336x280px)
LARGE_VERTICAL_LINK_UNIT	A large vertical ad link unit. (180x90px)
LEADERBOARD	A fully horizontal display area. (728x90px)
MEDIUM_RECTANGLE	A medium rectangular ad. (300x250px)
MEDIUM_VERTICAL_LINK_UNIT	A medium vertical ad link unit. (160x90px)
SKYSCRAPER	A large vertical ad. (120x600px)
SMALL_HORIZONTAL_LINK_UNIT	A small horizontal ad link unit. (468x15px)
SMALL_RECTANGLE	A small rectangular ad. (180x150px)
SMALL_SQUARE	A smaller square ad. (200x200px)
SMALL_VERTICAL_LINK_UNIT	A small vertical ad link unit. (120x90px)
SQUARE	A square ad with large type. (250x250px)
VERTICAL_BANNER	A medium-sized vertical ad. (120x240px)
WIDE_SKYSCRAPER	A wide, vertical ad using larger type. (160x600px)
X_LARGE_VERTICAL_LINK_UNIT	An extra large vertical ad link unit. (200x90px)

google.maps.panoramio.PanoramioLayer class

A `PanoramioLayer` displays photos from Panoramio as a rendered layer.

This class extends `MVCObject`.

Library

panoramio

Constructor

Constructor	Description
`PanoramioLayer(opts?:PanoramioLayerOptions)`	A layer that displays data from Panoramio.

Methods

Methods	Return Value	Description
`getMap()`	`Map`	Returns the map on which this layer is displayed.
`getTag()`	`string`	
`getUserId()`	`string`	
`setMap(map:Map)`	`None`	Renders the layer on the specified map. If map is set to null, the layer will be removed.
`setOptions(options:PanoramioLayerOptions)`	`None`	
`setTag(tag:string)`	`None`	
`setUserId(userId:string)`	`None`	

Events

Events	Arguments	Description
`click`	*PanoramioMouseEvent*	This event is fired when a feature in the layer is clicked.

google.maps.panoramio.PanoramioLayerOptions object specification

This object defines the properties that can be set on a `PanoramioLayer` object.

Library

panoramio

Properties

Properties	Type	Description
`clickable`	`boolean`	If true, the layer receives mouse events. Default value is `true`.
`map`	`Map`	The map on which to display the layer.
`suppressInfoWindows`	`boolean`	Suppress the rendering of info windows when layer features are clicked.
`tag`	`string`	A panoramio tag used to filter the photos which are displayed. Only photos which have been tagged with the supplied string will be shown.
`userId`	`string`	A Panoramio user ID. If provided, only photos by this user will be displayed on the map. If both a tag and user ID are provided, the tag will take precedence.

google.maps.panoramio.PanoramioFeature object specification

Describes a single Panoramio feature.

Library

panoramio

Properties

Properties	Type	Description
`author`	`string`	The username of the user who uploaded this photo.
`photoId`	`string`	The unique identifier for this photo, as used in the Panoramio API (see http://www.panoramio.com/api/widget/api.html).
`title`	`string`	The title of the photo.
`url`	`string`	The URL of the photo.
`userId`	`string`	The unique identifier for the user who uploaded this photo, as used in the Panoramio API (see http://www.panoramio.com/api/widget/api.html).

google.maps.panoramio.PanoramioMouseEvent object specification

The properties of a mouse event on a `PanoramioLayer`.

Library

panoramio

Properties

Properties	Type	Description
`featureDetails`	`PanoramioFeature`	A `PanoramioFeature` object containing information about the clicked feature.
`infoWindowHtml`	`string`	Pre-rendered HTML content to display within a feature's `InfoWindow` when clicked.
`latLng`	`LatLng`	The position at which to anchor an info window on the clicked feature.
`pixelOffset`	`Size`	The offset to apply to an info window anchored on the clicked feature.

google.maps.places.Autocomplete class

A service to provide Place predictions based on a user's text input. It attaches to an input element of type `text`, and listens for text entry in that field. The list of predictions is presented as a drop-down list, and is updated as text is entered.

This class extends `MVCObject`.

Library

places

Constructor

Constructor	Description
`Autocomplete(inputField:HTMLInputElement, opts?:`*`AutocompleteOptions`*`)`	Creates a new instance of `Autocomplete` that attaches to the specified input text field with the given options.

Methods

Methods	Return Value	Description
`getBounds()`	`LatLngBounds`	Returns the bounds to which predictions are biased.
`getPlace()`	`PlaceResult`	Returns the details of the Place selected by user

		if the details were successfully retrieved. Otherwise returns a stub Place object, with the `name` property set to the current value of the input field.
`setBounds(bounds:LatLngBounds)`	`None`	Sets the preferred area within which to return Place results. Results are biased towards, but not restricted to, this area.
`setComponentRestrictions(restrictions:ComponentRestrictions)`	`None`	Sets the component restrictions. Component restrictions are used to restrict predictions to only those within the parent component. E.g., the country.
`setTypes(types:Array<string>)`	`None`	Sets the types of predictions to be returned. Supported types are '`establishment`' for businesses and '`geocode`' for addresses. If no type is specified, both types will be returned. The `setTypes` method accepts a single element array.

Events

Events	Arguments	Description
`place_changed`	`None`	This event is fired when a `PlaceResult` is made available for a Place the user has selected. If the user enters the name of a Place that was not suggested by the control and presses the Enter key, or if a Place detail request fails, a `place_changed` event will be fired that contains the user input in the `name` property, with no other properties defined.

google.maps.places.AutocompleteOptions object specification

The options that can be set on an `Autocomplete` object.

Library

places

Properties

Properties	Type	Description
`bounds`	`LatLngBounds`	The area in which to search for places. Results are biased towards, but not restricted to, places contained within these bounds.
`componentRestrictions`	`ComponentRestrictions`	The component restrictions. Component restrictions are used to restrict predictions to only those within the parent component. E.g., the country.
`types`	`Array<string>`	The types of predictions to be returned. Four types are supported: `'establishment'` for businesses, `'geocode'` for addresses, `'(regions)'` for administrative regions and `'(cities)'` for localities. If nothing is specified, all types are returned. In general only a single type is allowed. The exception is that you can safely mix the `'geocode'` and `'establishment'` types, but note that this will have the same effect as specifying no types.

google.maps.places.AutocompletePrediction object specification

Library

places

Properties

Properties	Type	Description
`description`	`string`	This is the unformatted version of the query suggested by the Places service.

`matched_substrings`	`Array<PredictionSubstring>`	A set of substrings in the place's description that match elements in the user's input, suitable for use in highlighting those substrings. Each substring is identified by an offset and a length, expressed in unicode characters.
`place_id`	`string`	A place ID that can be used to retrieve details about this place using the place details service (see `PlacesService.getDetails()`).
`terms`	`Array<PredictionTerm>`	Information about individual terms in the above description, from most to least specific. For example, "Taco Bell", "Willitis", and "CA".
`types`	`Array<string>`	An array of types that the prediction belongs to, for example `'establishment'` or `'geocode'`.

google.maps.places.PredictionTerm object specification

Library

places

Properties

Properties	Type	Description
`offset`	`number`	The offset, in unicode characters, of the start of this term in the description of the place.
`value`	`string`	The value of this term, e.g. "Taco Bell".

google.maps.places.PredictionSubstring object specification

Library

places

Properties

Properties	Type	Description
`length`	`number`	The length of the substring.
`offset`	`number`	The offset to the substring's start within the description string.

google.maps.places.AutocompleteService class

Contains methods related to retrieving Autocomplete predictions.

Library

places

Constructor

Constructor	Description
`AutocompleteService()`	Creates a new instance of the `AutocompleteService`.

Methods

Methods	Return Value	Description
`getPlacePredictions(request:`*`AutocompletionRequest`*`, callback:function(Array<AutocompletePrediction>, PlacesServiceStatus))`	`None`	Retrieves place autocomplete predictions based on the supplied autocomplete request.
`getQueryPredictions(request:`*`QueryAutocompletionRequest`*`,`	`None`	Retrieves query autocomplete predictions

`callback:function(Array<QueryAutocompletePrediction>, PlacesServiceStatus))`		based on the supplied query autocomplete request.

google.maps.places.AutocompletionRequest object specification

An Autocompletion request to be sent to the `AutocompleteService`.

Library

places

Properties

Properties	Type	Description
`bounds`	`LatLngBounds`	Bounds for prediction biasing. Predictions will be biased towards, but not restricted to, the given `bounds`. Both `location` and `radius` will be ignored if `bounds` is set.
`componentRestrictions`	`ComponentRestrictions`	The component restrictions. Component restrictions are used to restrict predictions to only those within the parent component. E.g., the country.
`input`	`string`	The user entered input string.
`location`	`LatLng`	Location for prediction biasing. Predictions will be biased towards the given `location` and `radius`. Alternatively, `bounds` can be used.
`offset`	`number`	The character position in the input term at which the service uses text for predictions (the position of the cursor in the input field).
`radius`	`number`	The radius of the area used for prediction biasing. The `radius` is specified in meters, and must always be accompanied by a `location` property. Alternatively, `bounds` can be used.
`types`	`Array<string>`	The types of predictions to be returned. Four types are supported: `'establishment'` for businesses, `'geocode'` for addresses, `'(regions)'` for administrative regions and `'(cities)'` for localities. If nothing is specified, all types are returned.

google.maps.places.ComponentRestrictions object specification

Defines the component restrictions that can be used with the autocomplete service.

Library

places

Properties

Properties	Type	Description
`country`	`string`	Restricts predictions to the specified country (ISO 3166-1 Alpha-2 country code, case insensitive). E.g., us, br, au.

google.maps.places.PlaceAspectRating object specification

Defines information about an aspect of the place that users have reviewed.

Library

places

Properties

Properties	Type	Description
`rating`	`number`	The rating of this aspect. For individual reviews this is an integer from 0 to 3. For aggregated ratings of a place this is an integer from 0 to 30.

`type`	`string`	The aspect type, e.g. "food", "decor", "service", "overall".

google.maps.places.PlaceDetailsRequest object specification

A Place details query to be sent to the `PlacesService`.

Library

places

Properties

Properties	Type	Description
`placeId`	`string`	The Place ID of the Place for which details are being requested.

google.maps.places.PlaceGeometry object specification

Defines information about the geometry of a Place.

Library

places

Properties

Properties	Type	Description
`location`	`LatLng`	The Place's position.
`viewport`	`LatLngBounds`	The preferred viewport when displaying this Place on a map. This property will be null if the preferred viewport for the Place is not known.

google.maps.places.PlacePhoto object specification

Represents a photo element of a Place.

Library

places

Methods

Methods	Return Value	Description
`getUrl(opts:PhotoOptions)`	`string`	Returns the image URL corresponding to the specified options. You must include a PhotoOptions object with at least one of maxWidth or maxHeight specified.

Properties

Properties	Type	Description
`height`	`number`	The height of the photo in pixels.
`html_attributions`	`Array<string>`	Attribution text to be displayed for this photo.
`width`	`number`	The width of the photo in pixels.

google.maps.places.PhotoOptions object specification

Defines photo-requesting options.

Library

places

Properties

Properties	Type	Description
maxHeight	number	The maximum height in pixels of the returned image.
maxWidth	number	The maximum width in pixels of the returned image.

google.maps.places.PlaceResult object specification

Defines information about a Place.

Library

places

Properties

Properties	Type	Description
address_components	Array<GeocoderAddressComponent>	The collection of address components for this Place's location.
aspects	Array<PlaceAspectRating>	The rated aspects of this Place, based on Google and Zagat user reviews. The ratings are on a scale of 0 to 30.
formatted_address	string	The Place's full address.
formatted_phone_number	string	The Place's phone number, formatted according to the number's regional convention.
geometry	PlaceGeometry	The Place's geometry-related information.
html_attributions	Array<string>	Attribution text to be displayed for this Place result.
icon	string	URL to an image resource that can be used to represent this Place's category.
international_phone_number	string	The Place's phone number in international format. International format includes the country code, and is prefixed with the plus (+) sign.
name	string	The Place's name. Note: In the case of user entered Places, this is the raw text, as typed by the user. Please exercise caution when using this data, as malicious users may try to use it as a vector for code injection attacks (See http://en.wikipedia.org/wiki/Code_injection).
permanently_closed	boolean	A flag indicating whether the Place is permanently closed. If the place is not permanently closed, the flag is not present in search or details responses.
photos	Array<PlacePhoto>	Photos of this Place. The collection will contain up to ten PlacePhoto objects.
place_id	string	A unique identifier for a place.
price_level	number	The price level of the Place, on a scale of 0 to 4. Price levels are interpreted as follows: Value / Description 0 / Free 1 / Inexpensive 2 / Moderate 3 / Expensive 4 / Very Expensive

`rating`	`number`	A rating, between 1.0 to 5.0, based on user reviews of this Place.
`reviews`	`Array<PlaceReview>`	A list of reviews of this Place.
`types`	`Array<string>`	An array of types for this Place (e.g., `["political", "locality"]` or `["restaurant", "establishment"]`).
`url`	`string`	URL of the official Google page for this place. This will be the establishment's Google+ page if the Google+ page exists, otherwise it will be the Google-owned page that contains the best available information about the place.
`vicinity`	`string`	A fragment of the Place's address for disambiguation (usually street name and locality).
`website`	`string`	The authoritative website for this Place, such as a business' homepage.

google.maps.places.PlaceReview object specification

Represents a single review of a place.

Library

places

Properties

Properties	Type	Description
`aspects`	`Array<PlaceAspectRating>`	The aspects rated by the review. The ratings on a scale of 0 to 3.
`author_name`	`string`	The name of the reviewer.
`author_url`	`string`	A link to the reviewer's profile. This will be undefined when the reviewer's profile is unavailable.
`language`	`string`	An IETF language code indicating the language in which this review is written. Note that this code includes only the main language tag without any secondary tag indicating country or region. For example, all the English reviews are tagged as 'en' rather than 'en-AU' or 'en-UK'.
`text`	`string`	The text of a review.

google.maps.places.PlaceSearchPagination object specification

An object used to fetch additional pages of Places results.

Library

places

Methods

Methods	Return Value	Description
`nextPage()`	`None`	Fetches the next page of results. Uses the same callback function that was provided to the first search request.

Properties

Properties	Type	Description
`hasNextPage`	`boolean`	Indicates if further results are available. `true` when there is an additional results page.

google.maps.places.PlaceSearchRequest object specification

A Place search query to be sent to the `PlacesService`.

Library

places

Properties

Properties	Type	Description
`bounds`	`LatLngBounds`	The bounds within which to search for Places. Both `location` and `radius` will be ignored if `bounds` is set.
`keyword`	`string`	A term to be matched against all available fields, including but not limited to name, type, and address, as well as customer reviews and other third-party content.
`location`	`LatLng|LatLngLiteral`	The location around which to search for Places.
`maxPriceLevel`	`number`	Restricts results to only those places at the specified price level or lower. Valid values are in the range from 0 (most affordable) to 4 (most expensive), inclusive. Must be greater than or equal to `minPrice` , if specified.
`minPriceLevel`	`number`	Restricts results to only those places at the specified price level or higher. Valid values are in the range from 0 (most affordable) to 4 (most expensive), inclusive. Must be less than or equal to `maxPrice`, if specified.
`name`	`string`	Restricts the Place search results to Places that include this text in the name.
`openNow`	`boolean`	Restricts results to only those places that are open right now.
`radius`	`number`	The distance from the given location within which to search for Places, in meters. The maximum allowed value is 50 000.
`rankBy`	`RankBy`	Specifies the ranking method to use when returning results.
`types`	`Array<string>`	Restricts the Place search results to Places with a type matching at least one of the specified types in this array. Valid types are given here.

google.maps.places.PlacesService class

Contains methods related to searching for Places and retrieving details about a Place.

Library

places

Constructor

Constructor	Description
`PlacesService(attrContainer:HTMLDivElement|Map)`	Creates a new instance of the `PlacesService` that renders attributions in the specified container.

Methods

Methods	Return Value	Description
`getDetails(request:`*`PlaceDetailsRequest`*`, callback:function(PlaceResult, PlacesServiceStatus))`	`None`	Retrieves details about the Place identified by the given `placeId`.
`nearbySearch(request:`*`PlaceSearchRequest`*`, callback:function(Array<PlaceResult>, PlacesServiceStatus, PlaceSearchPagination))`	`None`	Retrieves a list of Places in a given area. The `PlaceResult`s passed to the callback are stripped-down versions of a full PlaceResult. A more detailed `PlaceResult` for each Place can be obtained by sending a Place Details request with the desired Place's `placeId` value.
`radarSearch(request:`*`RadarSearchRequest`*`, callback:function(Array<PlaceResult>, PlacesServiceStatus))`	`None`	Similar to the `nearbySearch` function, with the following differences: the search response will include up to 200 Places, identified only by their geographic coordinates and `place_id`.
`textSearch(request:`*`TextSearchRequest`*`,`	`None`	Similar to the `nearbySearch` function, with the following differences: it retrieves a

`callback:function(Array<PlaceResult>, PlacesServiceStatus))`		list of Places based on the `query` attribute in the given request object; `bounds` or `location` + `radius` parameters are optional; and the region, when provided, will not restrict the results to places inside the area, only bias the response towards results near it.

google.maps.places.PlacesServiceStatus class

The status returned by the `PlacesService` on the completion of its searches.

Library

places

Constant

Constant	Description
`INVALID_REQUEST`	This request was invalid.
`OK`	The response contains a valid result.
`OVER_QUERY_LIMIT`	The application has gone over its request quota.
`REQUEST_DENIED`	The application is not allowed to use the `PlacesService`.
`UNKNOWN_ERROR`	The `PlacesService` request could not be processed due to a server error. The request may succeed if you try again.
`ZERO_RESULTS`	No result was found for this request.

google.maps.places.QueryAutocompletePrediction object specification

Represents a single Query Autocomplete prediction.

Library

places

Properties

Properties	Type	Description
`description`	`string`	This is the unformatted version of the query suggested by the Places service.
`matched_substrings`	`Array<PredictionSubstring>`	A set of substrings in the place's description that match elements in the user's input, suitable for use in highlighting those substrings. Each substring is identified by an offset and a length, expressed in unicode characters.
`place_id`	`string`	Only available if prediction is a place. A place ID that can be used to retrieve details about this place using the place details service (see `PlacesService.getDetails()`).
`terms`	`Array<PredictionTerm>`	Information about individual terms in the above description. Categorical terms come first (e.g., "restaurant"). Address terms appear from most to least specific. For example, "San Francisco", and "CA".

google.maps.places.QueryAutocompletionRequest object specification

An QueryAutocompletion request to be sent to the `QueryAutocompleteService`.

Library

places

Properties

Properties	Type	Description

`bounds`	`LatLngBounds`	Bounds for prediction biasing. Predictions will be biased towards, but not restricted to, the given `bounds`. Both `location` and `radius` will be ignored if `bounds` is set.
`input`	`string`	The user entered input string.
`location`	`LatLng`	Location for prediction biasing. Predictions will be biased towards the given `location` and `radius`. Alternatively, `bounds` can be used.
`offset`	`number`	The character position in the input term at which the service uses text for predictions (the position of the cursor in the input field).
`radius`	`number`	The radius of the area used for prediction biasing. The `radius` is specified in meters, and must always be accompanied by a `location` property. Alternatively, `bounds` can be used.

google.maps.places.RadarSearchRequest object specification

A Radar Search request to be sent to the `PlacesService`.

Library

places

Properties

Properties	Type	Description
`bounds`	`LatLngBounds`	Bounds used to bias results when searching for Places (optional). Both `location` and `radius` will be ignored if `bounds` is set. Results will not be restricted to those inside these bounds; but, results inside it will rank higher.
`keyword`	`string`	A term to be matched against all available fields, including but not limited to name, type, and address, as well as customer reviews and other third-party content.
`location`	`LatLng|LatLngLiteral`	The center of the area used to bias results when searching for Places.
`name`	`string`	Restricts results to Places that include this text in the name.
`radius`	`number`	The radius of the area used to bias results when searching for Places, in meters.
`types`	`Array<string>`	Restricts the Place search results to Places with a type matching at least one of the specified types in this array. Valid types are given here.

google.maps.places.RankBy class

Ranking options for a PlaceSearchRequest.

Library

places

Constant

Constant	Description
`DISTANCE`	Ranks place results by distance from the location.
`PROMINENCE`	Ranks place results by their prominence.

google.maps.places.SearchBox class

A service to provide query predictions based on a user's text input. It attaches to an input element of type `text`, and listens for text entry in that field. The list of predictions is presented as a drop-down list, and is updated as text is entered.

This class extends `MVCObject`.

Library

places

Constructor

Constructor	Description
`SearchBox(inputField:HTMLInputElement, opts?:SearchBoxOptions)`	Creates a new instance of `SearchBox` that attaches to the specified input text field with the given options.

Methods

Methods	Return Value	Description
`getBounds()`	`LatLngBounds`	Returns the bounds to which query predictions are biased.
`getPlaces()`	`Array<PlaceResult>`	Returns the query selected by the user, or `null` if no places have been found yet, to be used with `places_changed` event.
`setBounds(bounds:LatLngBounds)`	`None`	Sets the region to use for biasing query predictions. Results will only be biased towards this area and not be completely restricted to it.

Events

Events	Arguments	Description
`places_changed`	`None`	This event is fired when the user selects a query, `getPlaces` should be used to get new places.

google.maps.places.SearchBoxOptions object specification

The options that can be set on a `SearchBox` object.

Library

places

Properties

Properties	Type	Description
`bounds`	`LatLngBounds`	The area towards which to bias query predictions. Predictions are biased towards, but not restricted to, queries targeting these bounds.

google.maps.places.TextSearchRequest object specification

A text search request to be sent to the `PlacesService`.

Library

places

Properties

Properties	Type	Description
`bounds`	`LatLngBounds`	Bounds used to bias results when searching for Places (optional). Both `location` and `radius` will be ignored if `bounds` is set. Results will not be restricted to those inside these bounds; but, results inside it will rank higher.
`location`	`LatLng|LatLngLiteral`	The center of the area used to bias results when searching for Places.
`query`	`string`	The request's query term. e.g. the name of a place ('Eiffel Tower'), a category followed by the name of a location ('pizza in New York'), or the name of a place followed by a location disambiguator ('Starbucks in Sydney').
`radius`	`number`	The radius of the area used to bias results when searching for Places, in meters.
`types`	`Array<string>`	Restricts the Place search results to Places with a type matching at least one of the specified types in

		this array. Valid types are given here.

google.maps.drawing.DrawingManager class

Allows users to draw markers, polygons, polylines, rectangles, and circles on the map. The `DrawingManager`'s drawing mode defines the type of overlay that will be created by the user. Adds a control to the map, allowing the user to switch drawing mode.

This class extends `MVCObject`.

Library

drawing

Constructor

Constructor	Description
`DrawingManager(options?:DrawingManagerOptions)`	Creates a `DrawingManager` that allows users to draw overlays on the map, and switch between the type of overlay to be drawn with a drawing control.

Methods

Methods	Return Value	Description
`getDrawingMode()`	`OverlayType`	Returns the `DrawingManager`'s drawing mode.
`getMap()`	`Map`	Returns the `Map` to which the `DrawingManager` is attached, which is the `Map` on which the overlays created will be placed.
`setDrawingMode(drawingMode:OverlayType)`	`None`	Changes the `DrawingManager`'s drawing mode, which defines the type of overlay to be added on the map. Accepted values are `MARKER`, `POLYGON`, `POLYLINE`, `RECTANGLE`, `CIRCLE`, or `null`. A drawing mode of `null` means that the user can interact with the map as normal, and clicks do not draw anything.
`setMap(map:Map)`	`None`	Attaches the `DrawingManager` object to the specified `Map`.
`setOptions(options:DrawingManagerOptions)`	`None`	Sets the `DrawingManager`'s options.

Events

Events	Arguments	Description
`circlecomplete`	`Circle`	This event is fired when the user has finished drawing a circle.
`markercomplete`	`Marker`	This event is fired when the user has finished drawing a marker.
`overlaycomplete`	*`OverlayCompleteEvent`*	This event is fired when the user has finished drawing an overlay of any type.
`polygoncomplete`	`Polygon`	This event is fired when the user has finished drawing a polygon.
`polylinecomplete`	`Polyline`	This event is fired when the user has finished drawing a polyline.
`rectanglecomplete`	`Rectangle`	This event is fired when the user has finished drawing a rectangle.

google.maps.drawing.DrawingManagerOptions object specification

Options for the drawing manager.

Library

drawing

Properties

Properties	Type	Description
`circleOptions`	`CircleOptions`	Options to apply to any new circles created with this `DrawingManager`. The `center`

		and `radius` properties are ignored, and the `map` property of a new circle is always set to the `DrawingManager`'s map.
`drawingControl`	`boolean`	The enabled/disabled state of the drawing control. Defaults to `true`.
`drawingControlOptions`	`DrawingControlOptions`	The display options for the drawing control.
`drawingMode`	`OverlayType`	The `DrawingManager`'s drawing mode, which defines the type of overlay to be added on the map. Accepted values are `MARKER`, `POLYGON`, `POLYLINE`, `RECTANGLE`, `CIRCLE`, or `null`. A drawing mode of `null` means that the user can interact with the map as normal, and clicks do not draw anything.
`map`	`Map`	The `Map` to which the `DrawingManager` is attached, which is the `Map` on which the overlays created will be placed.
`markerOptions`	`MarkerOptions`	Options to apply to any new markers created with this `DrawingManager`. The `position` property is ignored, and the `map` property of a new marker is always set to the `DrawingManager`'s map.
`polygonOptions`	`PolygonOptions`	Options to apply to any new polygons created with this `DrawingManager`. The `paths` property is ignored, and the `map` property of a new polygon is always set to the `DrawingManager`'s map.
`polylineOptions`	`PolylineOptions`	Options to apply to any new polylines created with this `DrawingManager`. The `path` property is ignored, and the `map` property of a new polyline is always set to the `DrawingManager`'s map.
`rectangleOptions`	`RectangleOptions`	Options to apply to any new rectangles created with this `DrawingManager`. The `bounds` property is ignored, and the `map` property of a new rectangle is always set to the `DrawingManager`'s map.

google.maps.drawing.DrawingControlOptions object specification

Options for the rendering of the drawing control.

Library

drawing

Properties

Properties	Type	Description
`drawingModes`	`Array<OverlayType>`	The drawing modes to display in the drawing control, in the order in which they are to be displayed. The hand icon (which corresponds to the null drawing mode) is always available and is not to be specified in this array. Defaults to `[MARKER, POLYLINE, RECTANGLE, CIRCLE, POLYGON]`.
`position`	`ControlPosition`	Position id. Used to specify the position of the control on the map. The default position is `TOP_LEFT`.

google.maps.drawing.OverlayCompleteEvent object specification

The properties of an overlaycomplete event on a `DrawingManager`.

Library

drawing

Properties

Properties	Type	Description
`overlay`	`Marker|Polygon|Polyline|Rectangle|Circle`	The completed overlay.
`type`	`OverlayType`	The completed overlay's type.

google.maps.drawing.OverlayType class

The types of overlay that may be created by the `DrawingManager`.

Library

drawing

Constant

Constant	Description
`CIRCLE`	Specifies that the `DrawingManager` creates circles, and that the overlay given in the `overlaycomplete` event is a circle.
`MARKER`	Specifies that the `DrawingManager` creates markers, and that the overlay given in the `overlaycomplete` event is a marker.
`POLYGON`	Specifies that the `DrawingManager` creates polygons, and that the overlay given in the `overlaycomplete` event is a polygon.
`POLYLINE`	Specifies that the `DrawingManager` creates polylines, and that the overlay given in the `overlaycomplete` event is a polyline.
`RECTANGLE`	Specifies that the `DrawingManager` creates rectangles, and that the overlay given in the `overlaycomplete` event is a rectangle.

google.maps.weather.CloudLayer class

A layer showing cloud imagery.

This class extends `MVCObject`.

Library

weather

Constructor

Constructor	Description
`CloudLayer()`	Creates a new `CloudLayer` instance that displays a cloud overlay.

Methods

Methods	Return Value	Description
`getMap()`	`Map`	Returns the map on which this layer is displayed.
`setMap(map:Map)`	`None`	Renders the layer on the specified map. If map is set to null, the layer will be removed.

google.maps.weather.WeatherLayer class

A layer that displays weather icons.

This class extends `MVCObject`.

Library

weather

Constructor

Constructor	Description
`WeatherLayer(opts?:WeatherLayerOptions)`	Creates a new `WeatherLayer` instance that displays weather icons.

Methods

Methods	Return Value	Description
`getMap()`	`Map`	Returns the map on which this layer is displayed.

`setMap(map:Map)`	`None`	Renders the layer on the specified map. If map is set to null, the layer will be removed.
`setOptions(options:WeatherLayerOptions)`	`None`	Sets the `WeatherLayer`'s options.

Events

Events	Arguments	Description
`click`	*`WeatherMouseEvent`*	This event is fired when a feature in the weather layer is clicked.

google.maps.weather.WeatherLayerOptions object specification

This object defines the properties that can be set on a `WeatherLayer` object.

Library

weather

Properties

Properties	Type	Description
`clickable`	`boolean`	If true, the layer receives mouse events. Default value is true.
`labelColor`	`LabelColor`	The color of labels on the weather layer. If this is not explicitly set, the label color is chosen automatically depending on the map type.
`map`	`Map`	The map on which to display the layer.
`suppressInfoWindows`	`boolean`	Suppress the rendering of info windows when weather icons are clicked.
`temperatureUnits`	`TemperatureUnit`	The units to use for temperature.
`windSpeedUnits`	`WindSpeedUnit`	The units to use for wind speed.

google.maps.weather.TemperatureUnit class

The temperature unit displayed by the weather layer.

Library

weather

Constant

Constant	Description
`CELSIUS`	Specifies that temperatures should be displayed in degrees Celsius.
`FAHRENHEIT`	Specifies that temperatures should be displayed in degrees Fahrenheit.

google.maps.weather.WindSpeedUnit class

The wind speed unit displayed by the weather layer.

Library

weather

Constant

Constant	Description
`KILOMETERS_PER_HOUR`	Specifies that wind speeds should be displayed in kilometers per hour.

`METERS_PER_SECOND`	Specifies that wind speeds should be displayed in meters per second.
`MILES_PER_HOUR`	Specifies that wind speeds should be displayed in miles per hour.

google.maps.weather.LabelColor class

The color of the labels displayed on the weather layer.

Library

weather

Constant

Constant	Description
`BLACK`	Weather labels will be displayed as black text with a white border.
`WHITE`	Weather labels will be displayed as white text with a black border.

google.maps.weather.WeatherMouseEvent object specification

The properties of a mouse event on a `WeatherLayer`.

Library

weather

Properties

Properties	Type	Description
`featureDetails`	`WeatherFeature`	A `WeatherFeature` object containing information about the clicked feature.
`infoWindowHtml`	`string`	Pre-rendered HTML content to display within a feature's `InfoWindow` when clicked.
`latLng`	`LatLng`	The position at which to anchor an info window on the clicked feature.
`pixelOffset`	`Size`	The offset to apply to an info window anchored on the clicked feature.

google.maps.weather.WeatherFeature object specification

Describes a single Weather feature.

Library

weather

Properties

Properties	Type	Description
`current`	`WeatherConditions`	The current weather conditions at this location.
`forecast`	`Array<WeatherForecast>`	A forecast of weather conditions over the next four days. The forecast array is always in chronological order.
`location`	`string`	The location name of this feature, e.g. "San Francisco, California".
`temperatureUnit`	`TemperatureUnit`	The temperature units being used.
`windSpeedUnit`	`WindSpeedUnit`	The wind speed units being used.

google.maps.weather.WeatherConditions object specification

Describes a single weather feature.

Library

weather

Properties

Properties	Type	Description
day	string	The current day of the week in long form, e.g. "Monday".
description	string	A description of the conditions, e.g. "Partly Cloudy".
high	number	The highest temperature reached during the day.
humidity	number	The current humidity, expressed as a percentage.
low	number	The lowest temperature reached during the day.
shortDay	string	The current day of the week in short form, e.g. "M".
temperature	number	The current temperature, in the specified temperature units.
windDirection	string	The current wind direction.
windSpeed	number	The current wind speed, in the specified wind speed units.

google.maps.weather.WeatherForecast object specification

Describes a single day's weather forecast.

Library

weather

Properties

Properties	Type	Description
day	string	The day of the week in long form, e.g. "Monday".
description	string	A description of the conditions, e.g. "Partly Cloudy".
high	number	The highest temperature reached during the day.
low	number	The lowest temperature reached during the day.
shortDay	string	The day of the week in short form, e.g. "M".

google.maps.visualization.MapsEngineLayer class

A `MapsEngineLayer` allows you to display data from Google Maps Engine or the Google Earth Gallery.

This class extends `MVCObject`.

Library

visualization

Constructor

Constructor	Description
`MapsEngineLayer(options:MapsEngineLayerOptions)`	Creates a new instance of `MapsEngineLayer`.

Methods

Methods	Return Value	Description
getLayerId()	string	Returns the ID of the Maps Engine layer being

		displayed, if set.
`getLayerKey()`	`string`	Returns the key of the layer to be displayed.
`getMap()`	`Map`	Returns the map on which this layer is displayed.
`getMapId()`	`string`	Returns the ID of the Maps Engine map to which the layer belongs.
`getOpacity()`	`number`	Returns the opacity of the layer. Applies only to imagery layers.
`getProperties()`	`MapsEngineLayerProperties`	Returns properties of the Maps Engine layer, which are available once the layer has loaded.
`getStatus()`	`MapsEngineStatus`	Returns the status of the layer, which is available once the requested layer has loaded.
`getZIndex()`	`number`	Returns the z-index.
`setLayerId(layerId:string)`	`None`	Sets the ID of a single Maps Engine layer to display. Changing this value will cause the layer to be redrawn.
`setLayerKey(layerKey:string)`	`None`	Sets the key of the layer to be displayed. Maps Engine layer keys are only unique within a single map, and can be changed by map owners. Changing this value will cause the layer to be redrawn.
`setMap(map:Map)`	`None`	Renders the layer on the specified map. If map is set to null, the layer will be removed.
`setMapId(mapId:string)`	`None`	Sets the ID of the Maps Engine map that contains the layer with the given `layerKey`. Changing this value will cause the layer to be redrawn.
`setOpacity(opacity:number)`	`None`	Sets the opacity of the layer, expressed as a number between 0 and 1. Applies only to imagery layers. Note: Be careful of setting this option for other layer types, as it may become effective in the future.
`setOptions(options:`*`MapsEngineLayerOptions`*`)`	`None`	
`setZIndex(zIndex:number)`	`None`	Sets the z-index. Only applies to Vector and KML layers.

Events

Events	Arguments	Description
`click`	*`MapsEngineMouseEvent`*	This event is fired when a feature in the layer is clicked.
`properties_changed`	`None`	This event is fired when the layer has finished loading, and the layer's properties are available for reading.
`status_changed`	`None`	This event is fired when the layer has finished loading, and the status is available to determine if the layer loaded successfully.

google.maps.visualization.MapsEngineLayerOptions object specification

This object defines the properties that can be set on a `MapsEngineLayer` object. `layerId`, or both `mapId` and `layerKey`, must be set.

Library

visualization

Properties

Properties	Type	Description
`accessToken`	`string`	The authentication token returned by an OAuth 2.0 authentication request.
`clickable`	`boolean`	If true, the layer receives mouse events. Default value is true.

`fitBounds`	`boolean`	If this option is set to true, the map viewport is centered and zoomed to the bounding box of the contents of the layer. Default value is false. Applies only to KML layers. Note: Be careful when setting this option for other layer types, as it may become effective in the future.
`layerId`	`string`	The ID of a single Maps Engine layer to display.
`layerKey`	`string`	The key of the layer to display. Maps Engine layer keys are only unique within a single map, and can be changed by map owners.
`map`	`Map`	The map on which to display the layer.
`mapId`	`string`	The ID of the Maps Engine map that contains the layer with the given `layerKey`.
`opacity`	`number`	The opacity of the layer, expressed as a number between 0 and 1. Defaults to 1. Applies only to imagery layers. Note: Be careful of setting this option for other layer types, as it may become effective in the future.
`suppressInfoWindows`	`boolean`	Suppress the rendering of info windows when layer features are clicked.
`zIndex`	`number`	The z-index of the layer. Only applies to Vector and KML layers.

google.maps.visualization.MapsEngineLayerProperties object specification

This object defines the properties of a Maps Engine layer.

Library

visualization

Properties

Properties	Type	Description
`name`	`string`	The name of the layer.

google.maps.visualization.MapsEngineMouseEvent object specification

The properties of a mouse event on a `MapsEngineLayer`.

Library

visualization

Properties

Properties	Type	Description
`featureId`	`string`	The feature ID, guaranteed to be unique within the layer.
`infoWindowHtml`	`string`	Pre-rendered HTML content, as placed in the infowindow by the default UI.
`latLng`	`LatLng`	The position at which to anchor an infowindow on the clicked feature.
`pixelOffset`	`Size`	The offset to apply to an infowindow anchored on the clicked feature.

google.maps.visualization.MapsEngineStatus object specification

The status returned by `MapsEngineLayer` when a layer has loaded.

Library

visualization

Constant

Constant	Description
`INVALID_LAYER`	The requested layer is not a valid layer.

`OK`	The layer loaded successfully.
`UNKNOWN_ERROR`	The layer failed to load for an unknown reason.

google.maps.visualization.DynamicMapsEngineLayer class

A `DynamicMapsEngineLayer` allows you to display data from Google Maps Engine or the Google Earth Gallery.

This class extends `MVCObject`.

Library

visualization

Constructor

Constructor	Description
`DynamicMapsEngineLayer(options:`*`DynamicMapsEngineLayerOptions`*`)`	Creates a new instance of `DynamicMapsEngineLayer`.

Methods

Methods	Return Value	Description
`getFeatureStyle(featureId:string)`	`FeatureStyle`	Returns the style for the given feature, with which individual style properties can be retrieved or set.
`getLayerId()`	`string`	Returns the ID of the Maps Engine layer being displayed, if set.
`getLayerKey()`	`string`	Returns the key of the layer to be displayed.
`getMap()`	`Map`	Returns the map on which this layer is displayed.
`getMapId()`	`string`	Returns the ID of the Maps Engine map to which the layer belongs.
`getOpacity()`	`number`	Returns the opacity of the layer. Applies only to imagery layers.
`getStatus()`	`MapsEngineStatus`	Returns the status of the layer, set once the requested layer has loaded.
`setLayerId(layerId:string)`	`None`	Sets the ID of a single Maps Engine layer to display.
`setLayerKey(layerKey:string)`	`None`	Sets the key of the layer to be displayed. Maps Engine Layer Keys are only unique within a single map, and can be changed by map owners. Changing this value will cause the layer to be redrawn.
`setMap(map:Map)`	`None`	Renders the layer on the specified map. If map is set to null, the layer will be removed.
`setMapId(mapId:string)`	`None`	Sets the ID of the Maps Engine map to which the layer belongs. Changing this value will cause the layer to be redrawn.
`setOpacity(opacity:number)`	`None`	Sets the opacity of the layer, expressed as a number between 0 and 1. Applies only to imagery layers. Note: Be careful of setting this option for other layer types, as it may become effective in the future.
`setOptions(options:`*`DynamicMapsEngineLayerOptions`*`)`	`None`	

Events

Events	Arguments	Description
`click`	*`DynamicMapsEngineMouseEvent`*	This event is fired when a feature in the layer is clicked.
`dblclick`	*`DynamicMapsEngineMouseEvent`*	This event is fired when a feature in the layer is double clicked.

mousedown	*DynamicMapsEngineMouseEvent*	This event is fired for a mousedown on a feature in the layer.
mousemove	*DynamicMapsEngineMouseEvent*	This event is fired when the mouse moves over a feature in the layer.
mouseout	*DynamicMapsEngineMouseEvent*	This event is fired when the mouse leaves a feature in the layer.
mouseover	*DynamicMapsEngineMouseEvent*	This event is fired when the mouse enters a feature in the layer.
mouseup	*DynamicMapsEngineMouseEvent*	This event is fired for a mouseup on a feature in the layer.
properties_changed	None	This event is fired when the layer's properties are available for reading.
rightclick	*DynamicMapsEngineMouseEvent*	This event is fired for a rightclick on a feature in the layer.
status_changed	None	This event is fired when the layer has finished loading, and the status is available to determine if the layer loaded successfully.

google.maps.visualization.DynamicMapsEngineLayerOptions object specification

This object defines the properties that can be set on a `DynamicMapsEngineLayer` object. `layerId`, or both `mapId` and `layerKey` must be set.

Library

visualization

Properties

Properties	Type	Description
accessToken	string	The authentication token returned by an OAuth 2.0 authentication request.
clickable	boolean	If true, the layer receives mouse events. Default value is true.
layerId	string	The ID of the Maps Engine layer to display.
layerKey	string	The key of the layer to display from the specified map.
map	Map	The map on which to display the layer.
mapId	string	The ID of the Maps Engine map that contains the layer with the given `layerKey`.
opacity	number	The opacity of the layer, expressed as a number between 0 and 1. Defaults to 1. Applies only to imagery layers. Note: Be careful of setting this option for other layer types, as it may become effective in the future.
suppressInfoWindows	boolean	Suppress the rendering of info windows when layer features are clicked.

google.maps.visualization.DynamicMapsEngineMouseEvent object specification

The properties of a mouse event on a `DynamicMapsEngineLayer`.

Library

visualization

Methods

Methods	Return Value	Description
getDetails(callback:function(MapsEngineMouseEvent))	None	Takes a callback that will be called with details about the feature that may be used to render an info window.

Properties

Properties	Type	Description
featureId	string	The feature ID, guaranteed to be unique within the layer.
latLng	LatLng	The latitude/longitude that was below the cursor when the event occurred.

google.maps.visualization.FeatureStyle object specification

Library

visualization

Methods

Methods	Return Value	Description
`reset(property:string)`	`None`	Resets the given style property to its original value.
`resetAll()`	`None`	Resets all style properties to their original values.

Properties

Properties	Type	Description
`fillColor`	`string`	The feature's fill color. All CSS3 colors are supported except for extended named colors.
`fillOpacity`	`string`	Fill opacity, expressed as a decimal between 0 and 1 inclusive. This property may be set as a number, but it will always be returned as a string.
`iconAnchor`	`string`	The icon's anchor point is the pixel in the source image that is aligned with the point's geographical location, expressed as a whitespace-separated pair of numbers: `x y`. Defaults to the center of the icon.
`iconClip`	`string`	The rectangular region of the icon's image (in image pixel coordinates) to use, as a whitespace-separated 4-tuple of numbers: `x y width height`. For example, to use a 32x32 icon situated at (0, 64) in a sprite sheet, specify `0 64 32 32`.
`iconImage`	`string`	The image to render at the point. Currently, only `url(...)` is supported.
`iconOpacity`	`string`	Icon opacity, expressed as a decimal between 0 and 1 inclusive. This property may be set as a number, but it will always be returned as a string.
`iconSize`	`string`	Icon size, expressed as a string with two measurements (with pixel or percentage as unit) separated by whitespace.
`strokeColor`	`string`	The feature's stroke color. All CSS3 colors are supported except for extended named colors.
`strokeOpacity`	`string`	Stroke opacity, expressed as a decimal between 0 and 1 inclusive. This property may be set as a number, but it will always be returned as a string.
`strokeWidth`	`string`	Stroke width in pixels. This property may be set as a number, but it will always be returned as a string.
`zIndex`	`string`	Rendering order. Features with greater zIndex are rendered on top.

google.maps.visualization.HeatmapLayer class

A layer that provides a client-side rendered heatmap, depicting the intensity of data at geographical points.

This class extends `MVCObject`.

Library

visualization

Constructor

Constructor	Description
`HeatmapLayer(opts?:HeatmapLayerOptions)`	Creates a new instance of `HeatmapLayer`.

Methods

Methods	Return Value	Description
`getData()`	`MVCArray<LatLng|WeightedLocation>`	Returns the data points currently displayed by this heatmap.

`getMap()`	`Map`	
`setData(data:MVCArray<LatLng|WeightedLocation>|Array<LatLng|WeightedLocation>)`	`None`	Sets the data points to be displayed by this heatmap.
`setMap(map:Map)`	`None`	Renders the heatmap on the specified map. If map is set to null, the heatmap will be removed.

google.maps.visualization.HeatmapLayerOptions object specification

This object defines the properties that can be set on a `HeatmapLayer` object.

Library

visualization

Properties

Properties	Type	Description
`data`	`MVCArray<LatLng>`	The data points to display. Required.
`dissipating`	`boolean`	Specifies whether heatmaps dissipate on zoom. By default, the radius of influence of a data point is specified by the radius option only. When dissipating is disabled, the radius option is intepreted as a radius at zoom level 0.
`gradient`	`Array<string>`	The color gradient of the heatmap, specified as an array of CSS color strings. All CSS3 colors are supported except for extended named colors.
`map`	`Map`	The map on which to display the layer.
`maxIntensity`	`number`	The maximum intensity of the heatmap. By default, heatmap colors are dynamically scaled according to the greatest concentration of points at any particular pixel on the map. This property allows you to specify a fixed maximum.
`opacity`	`number`	The opacity of the heatmap, expressed as a number between 0 and 1. Defaults to 0.6.
`radius`	`number`	The radius of influence for each data point, in pixels.

google.maps.visualization.WeightedLocation object specification

A data point entry for a heatmap. This is a geographical data point with a weight attribute.

Library

visualization

Properties

Properties	Type	Description
`location`	`LatLng`	The location of the data point.
`weight`	`number`	The weighting value of the data point.

Last updated March 17, 2015.