|  |
| --- |
| MineMap JavaScript API开发指南 V1.0 |
| **北京世纪高通科技有限公司**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **文件编号：** |  | **附 录：** | |  | | **文件状态：** | 创建 | **生效日期：** | | 2017/03/13 | | **总页数：** |  | **正 文**： | |  | | **编 制：** | 麻广伟 | **批 准：** |  | | |

目 录

[概述 6](#_Toc478371214)

[1.1 MineMap JavaScript API简介 6](#_Toc478371215)

[1.2 功能介绍 6](#_Toc478371216)

[1.3 面向对象 6](#_Toc478371217)

[1.4 浏览器支持 6](#_Toc478371218)

[入门 7](#_Toc478371219)

[2. 个性化地图 10](#_Toc478371220)

[2.1 设置配图方案 10](#_Toc478371221)

[2.2 初始化定制地图 14](#_Toc478371222)

[3. 地图控件 14](#_Toc478371223)

[3.1 添加控件 15](#_Toc478371224)

[3.2 控制控件位置 15](#_Toc478371225)

[3.3 自定义控件 15](#_Toc478371226)

[4. 地图覆盖物 16](#_Toc478371227)

[4.1 生成自定义标注 16](#_Toc478371228)

[4.2 监听标注事件 17](#_Toc478371229)

[4.3 删除标注 17](#_Toc478371230)

[4.4 添加信息窗体 18](#_Toc478371231)

[4.5 删除信息窗体 19](#_Toc478371232)

[5. 地图图层 19](#_Toc478371233)

[5.1 添加矢量图层 20](#_Toc478371234)

[5.1.1 面图层样式添加 20](#_Toc478371235)

[5.1.2 线图层样式添加 23](#_Toc478371236)

[5.1.3 标识图层样式添加 24](#_Toc478371237)

[5.1.4 圆图层样式添加 28](#_Toc478371238)

[5.1.5 粒子图层样式添加 29](#_Toc478371239)

[5.1.6 建筑物图层添加 30](#_Toc478371240)

[5.2 添加GeoJson图层 32](#_Toc478371241)

[5.3 添加栅格图层 32](#_Toc478371242)

[5.4 添加图像图层 33](#_Toc478371243)

[5.5 添加背景图层 34](#_Toc478371244)

[5.6 删除图层 35](#_Toc478371245)

[6. 地图交互 35](#_Toc478371246)

[6.1 地图事件类型 35](#_Toc478371247)

[6.2 事件的绑定和移除 37](#_Toc478371248)

[6.3 飞行 37](#_Toc478371249)

[6.4 图层交互 38](#_Toc478371250)

[6.4.1 获取图层要素信息 38](#_Toc478371251)

[6.4.2 修改图层样式 40](#_Toc478371252)

[6.5 地图控制 40](#_Toc478371253)

[7. 服务接口 41](#_Toc478371254)

[7.1 位置搜索服务 42](#_Toc478371255)

[7.1.1 根据查询的数据类型进行关键字搜索 42](#_Toc478371256)

[7.1.2 根据关键字搜索所有符合条件的数据 43](#_Toc478371257)

[7.1.3 根据关键字搜索poi数据 43](#_Toc478371258)

[7.1.4 根据关键字搜索道路数据 44](#_Toc478371259)

[7.1.5 根据关键字和坐标点进行周边搜索 44](#_Toc478371260)

[7.1.6 根据关键字和矩形框进行范围搜索 45](#_Toc478371261)

[7.2 周边查找服务 46](#_Toc478371262)

[7.2.1 根据坐标点进行周边查找 46](#_Toc478371263)

[7.3 路径规划服务 47](#_Toc478371264)

[7.3.1 根据起终点进行驾车路径规划搜索 47](#_Toc478371265)

[7.3.2 根据起终点进行公交路径规划搜索 49](#_Toc478371266)

[7.4 行政区划服务 51](#_Toc478371267)

[7.4.1 根据坐标点进行查找其所在的行政区划 51](#_Toc478371268)

[7.5 geocoding服务 52](#_Toc478371269)

[7.5.1 根据地址查找geocoding信息 52](#_Toc478371270)

[7.5.2 根据地址查找地址描述信息 53](#_Toc478371271)

[7.6 内容补全服务 54](#_Toc478371272)

[7.6.1 根据输入内容查找智能提示信息 54](#_Toc478371273)

[8. 常见问题 55](#_Toc478371274)

# 概述

## MineMap JavaScript API简介

MineMap API是一套由JavaScript语言编写的基于WebGl的应用程序接口，可帮助您在网站中构建功能丰富、交互性强、可定制的地图应用。当前版本为V1.2。

## 功能介绍

MineMap API包含了构建地图相关功能的各种接口，具体如下：

* 地图展示功能：展示2D图、3D图、栅格图、矢量图；
* 地图控件功能：可以在地图上添加/删除比例尺、缩放以及自定义控件，并可以设置各类控件的显示位置；
* 覆盖物功能：支持在地图上添加/删除标注marker、信息窗体Popup等覆盖物功能；
* 工具类功能：提供经纬度坐标与屏幕坐标互转功能；配合插件库提供测距、面积计算等功能；
* 鼠标交互功能：支持动态修改鼠标样式、鼠标拖拽/缩放地图及鼠标绘制等功能；
* 地图控制功能：支持地图平移、缩放、拖拽、旋转、俯仰；
* 图层功能：支持设置标注、圆、线、粒子、面、建筑物、栅格、背景等图层，并能设置图层渲染样式；
* 地图服务功能：提供数据信息检索的接口，如位置搜索、路径规划、地址解析、智能提示等服务；

## 面向对象

MineMap API 是面向具有一定 JavaScript 技能和web相关开发经验以及具有一些地图行业知识的读者

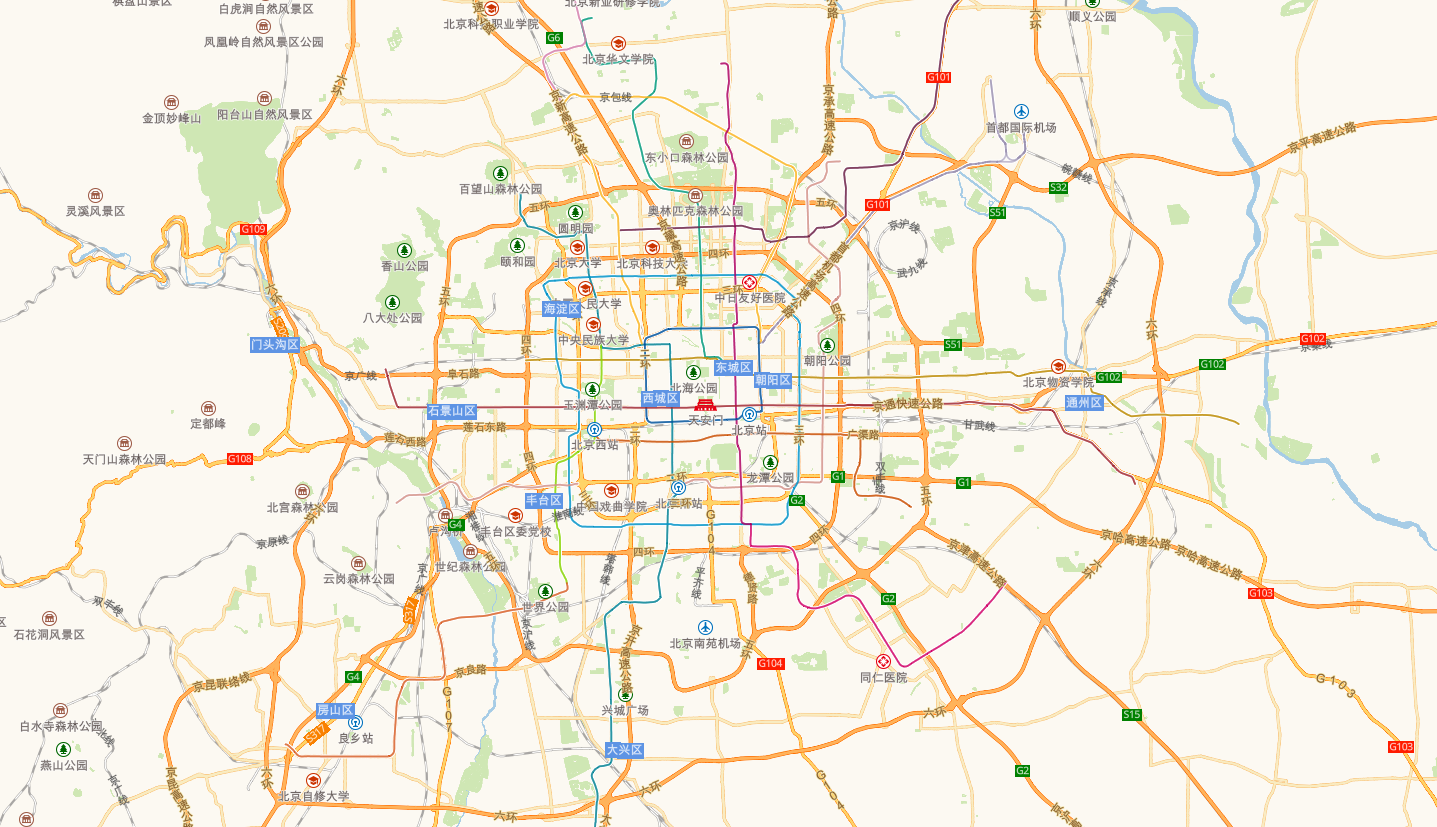
## 浏览器支持

PC浏览器：Chrome >= 56.0.2924.87、 Safari 、 Opera 40.0.2308.90、 Firefox 49.0.1、Microsoft Edge 25.1、Internet Explorer 11

移动浏览器：QQ浏览器v7.0.2、Chrome 56.0.2924.87、Safari、 Firefox49.0 、不支持UC

# 入门

入门指南带您迅速了解MineMap API的基本使用方法，掌握如何基于MineMap API开始地图应用的开发。



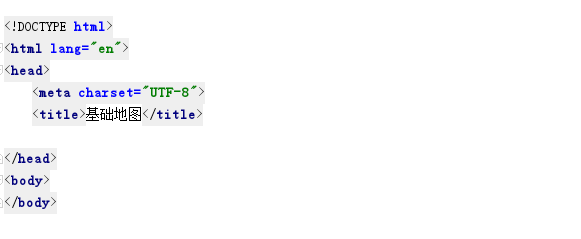
以下代码可以实现一个简单的全屏展示的地图Demo，Demo网址：http://minedata.cn/minemapapi/demo/index.html



下面我们分步骤介绍如何快捷地开始地图开发：

**第一步 准备页面**

我们建议使用最新的符合HTML5规范的文档声明：



**第二步 加载API文件**

引用MineMap1.2版本：



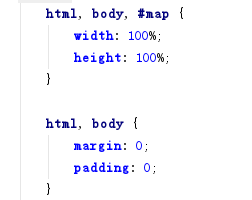
**第三步 创建地图容器**

地图需要一个HTML元素作为容器，这样才能展现到页面上。这里我们创建了一个div元素：



**第四步 指定地图容器大小**

地图的展示必须要给地图容器指定大小，否则无法显示



**第五步 地图初始化**

获取地图方案的accessToken与solution，然后通过MineMap API初始化地图，地图初始

化时必须设置地图容器container、地图style、地图中心点center、当前地图级别zoom的参数值，否则地图无法显示：



**第六步 地图配置与操作**

一般情况下，我们会给地图按需设定中心点坐标、地图视角等属性，可以通过两种方式实现：（1）地图初始化时设置相应参数



（2）地图初始化之后通过相应的API改变参数。



同时还可以添加一些地图操作，比如放大、缩小、比例尺等操作按钮：



# 个性化地图

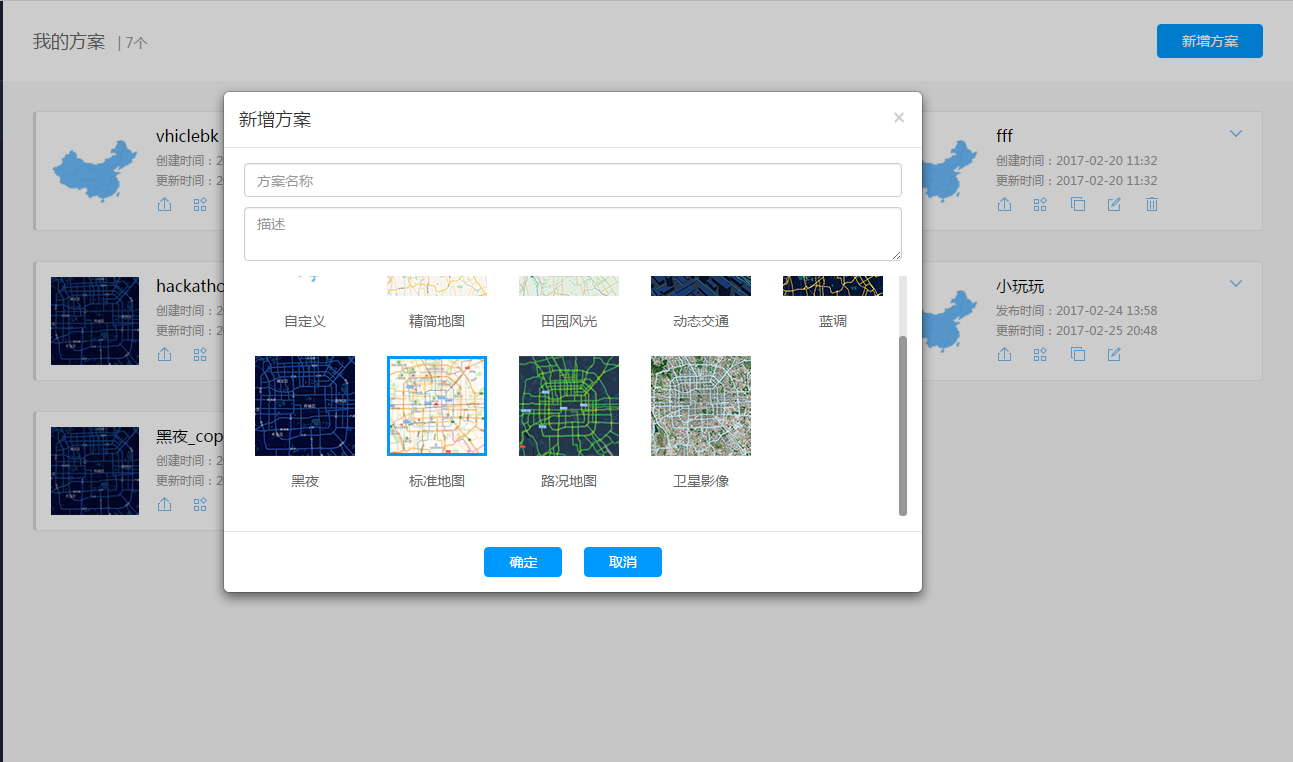
MineData平台可以配置自定义的个性化地图，我们可以注册登录MineData平台，进入工作台页面进行地图配置。

## 设置配图方案

下面将介绍设置配图方案的步骤：

**第一步 选择模板**

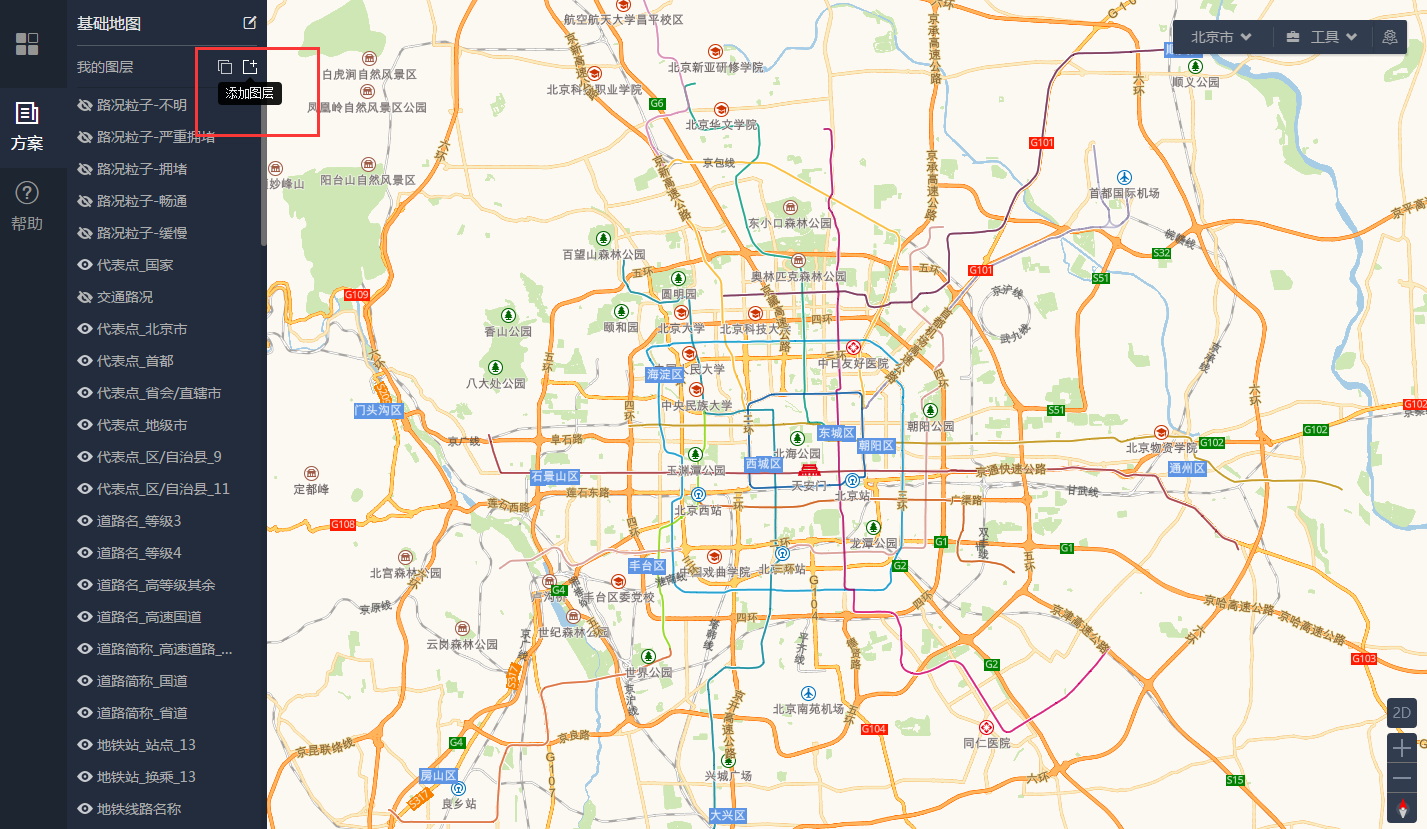
进入工作台，点击新建方案，在弹框中新增地图方案。可自定义模板，也可在已有的模板上进行修改。

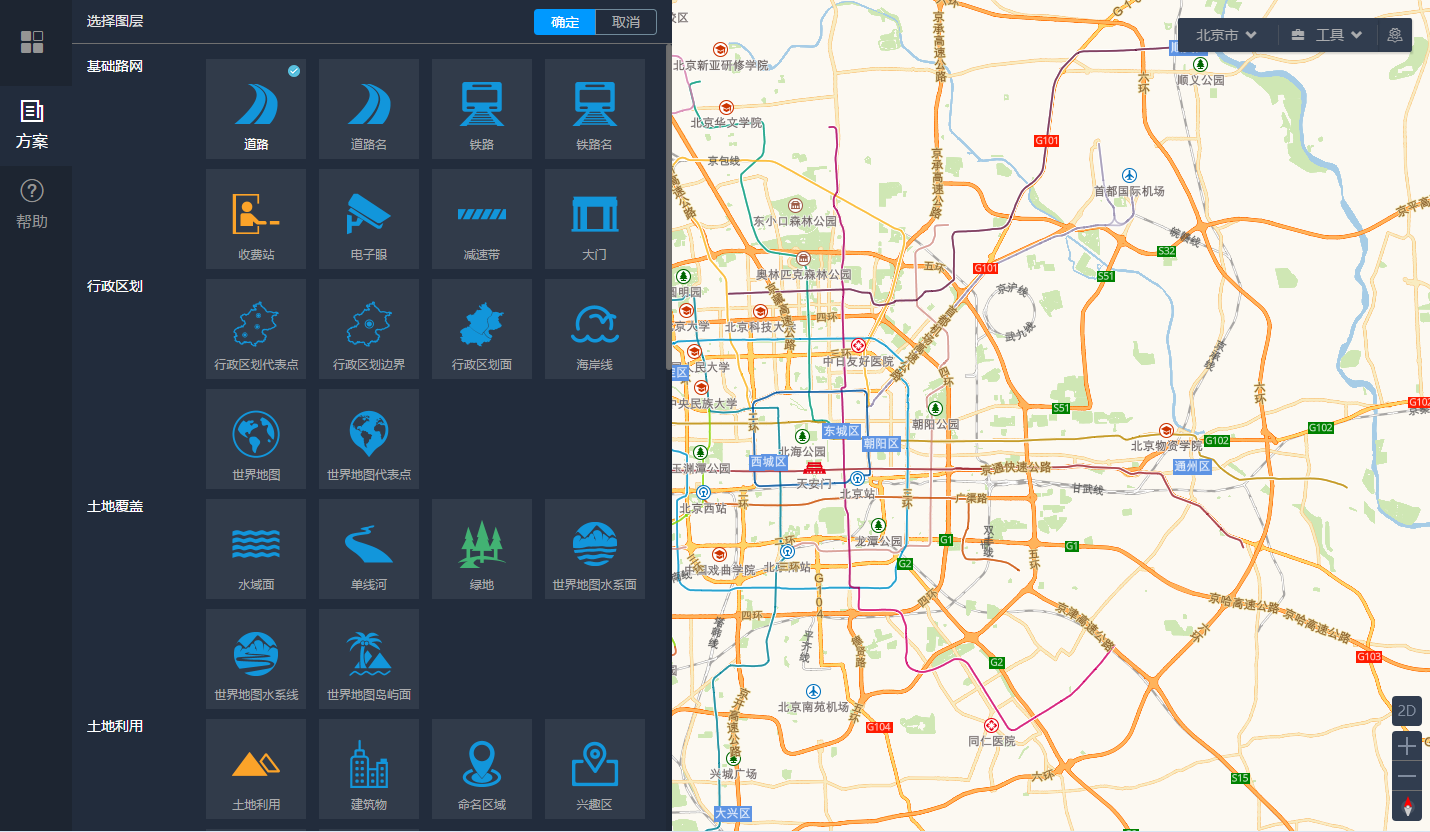


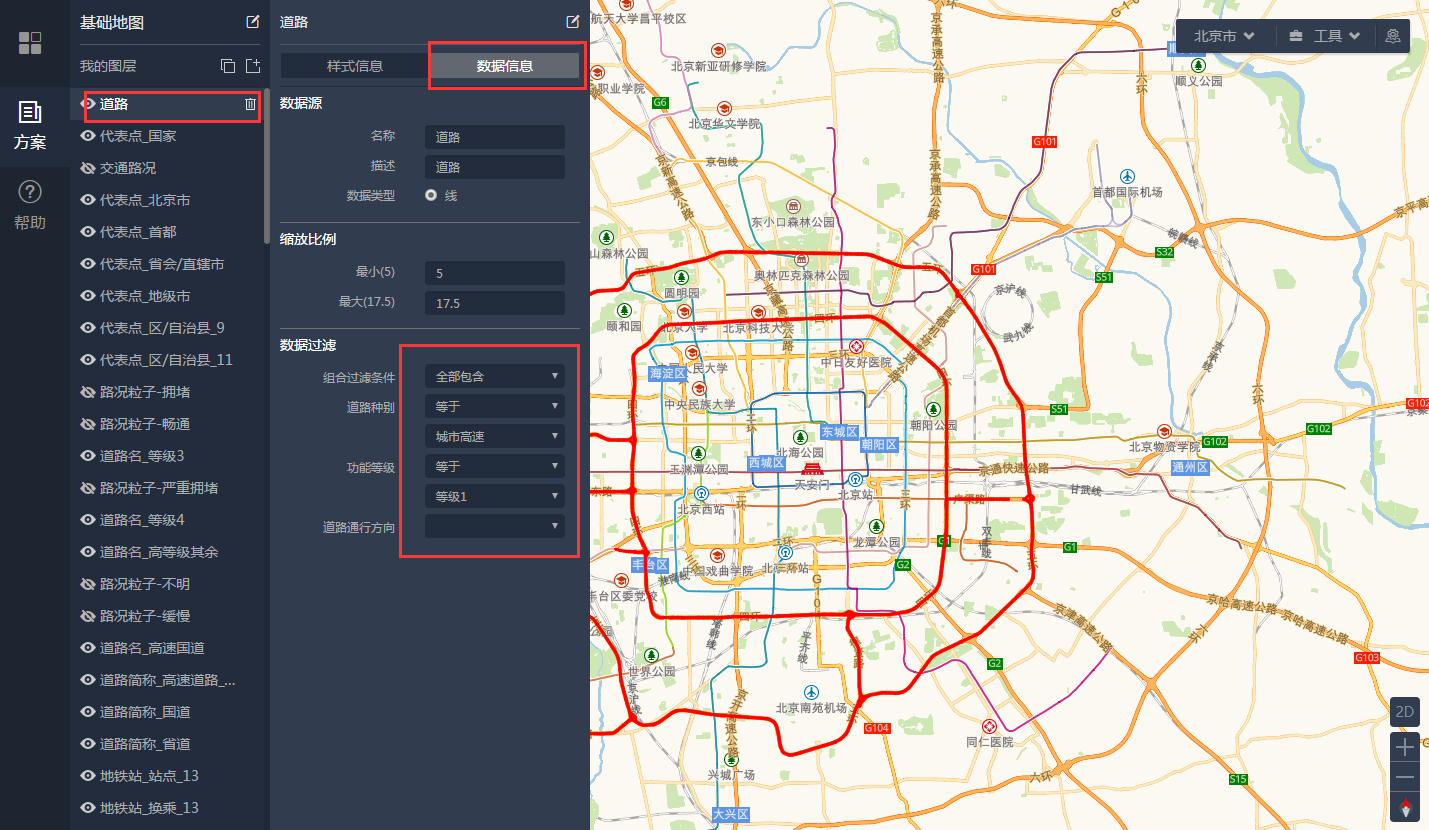
**第二步 编辑地图样式**

模板选择之后，进入编辑页面，可进行添加图层、修改图层样式、删除图层等操作。

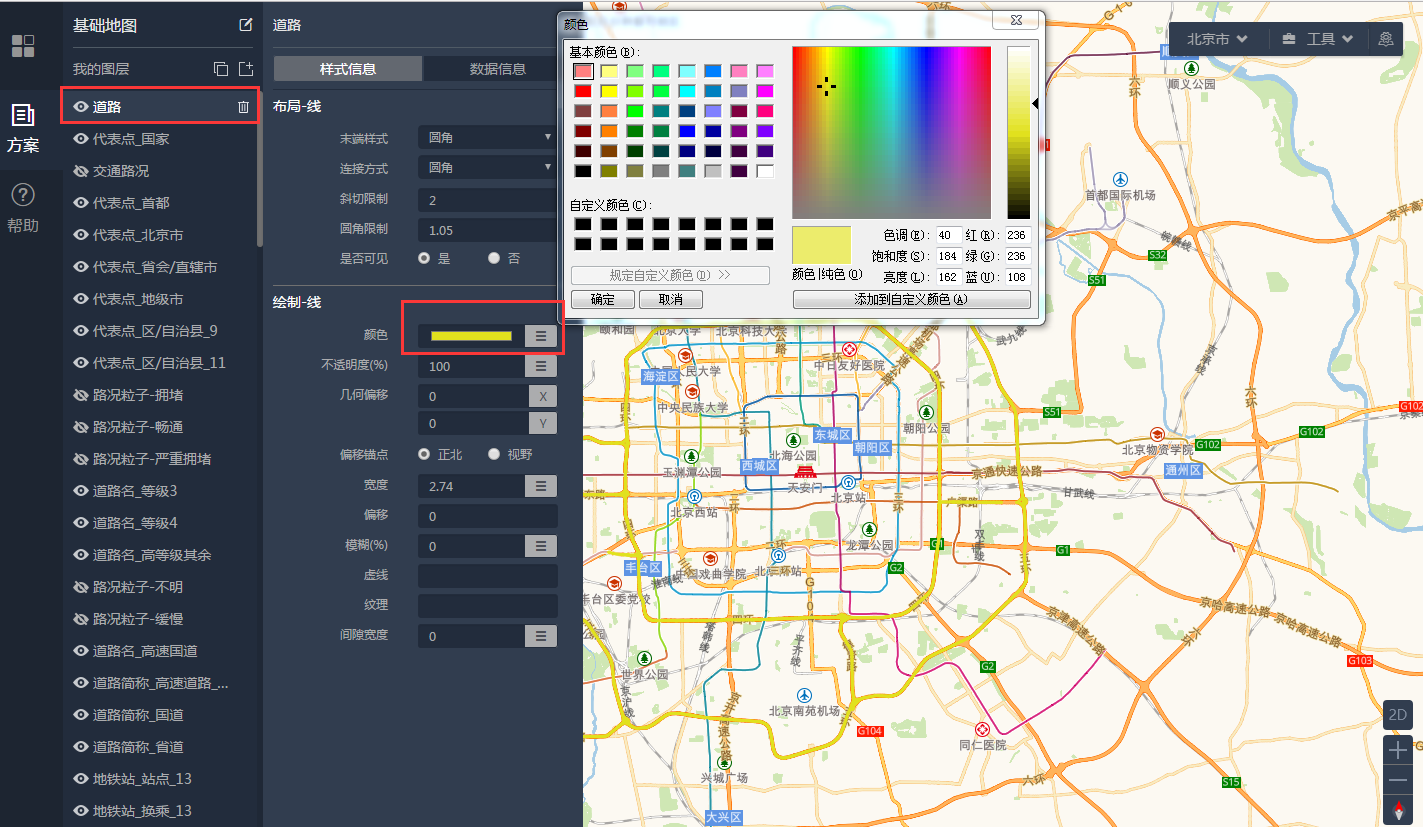
添加图层：



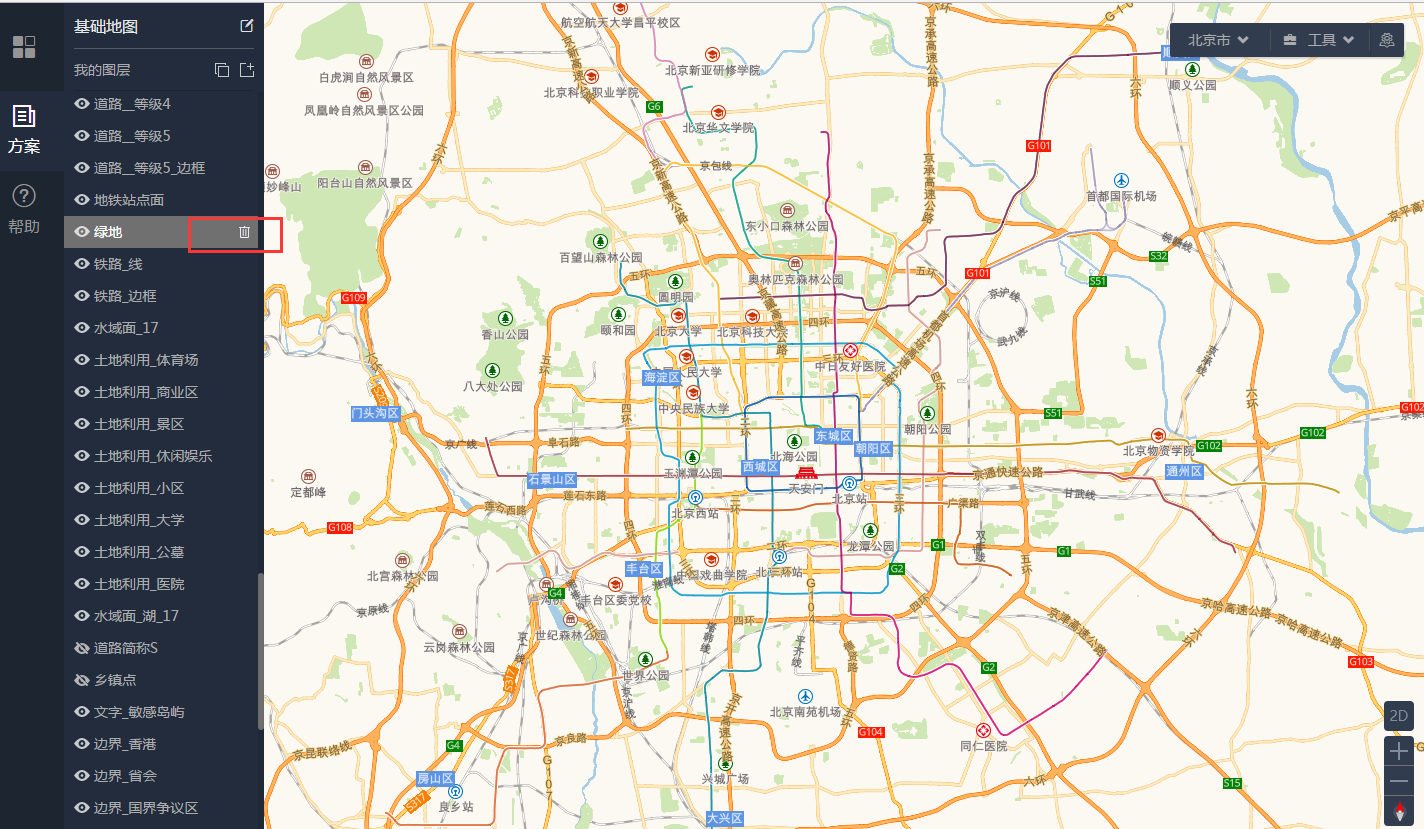




修改图层样式：

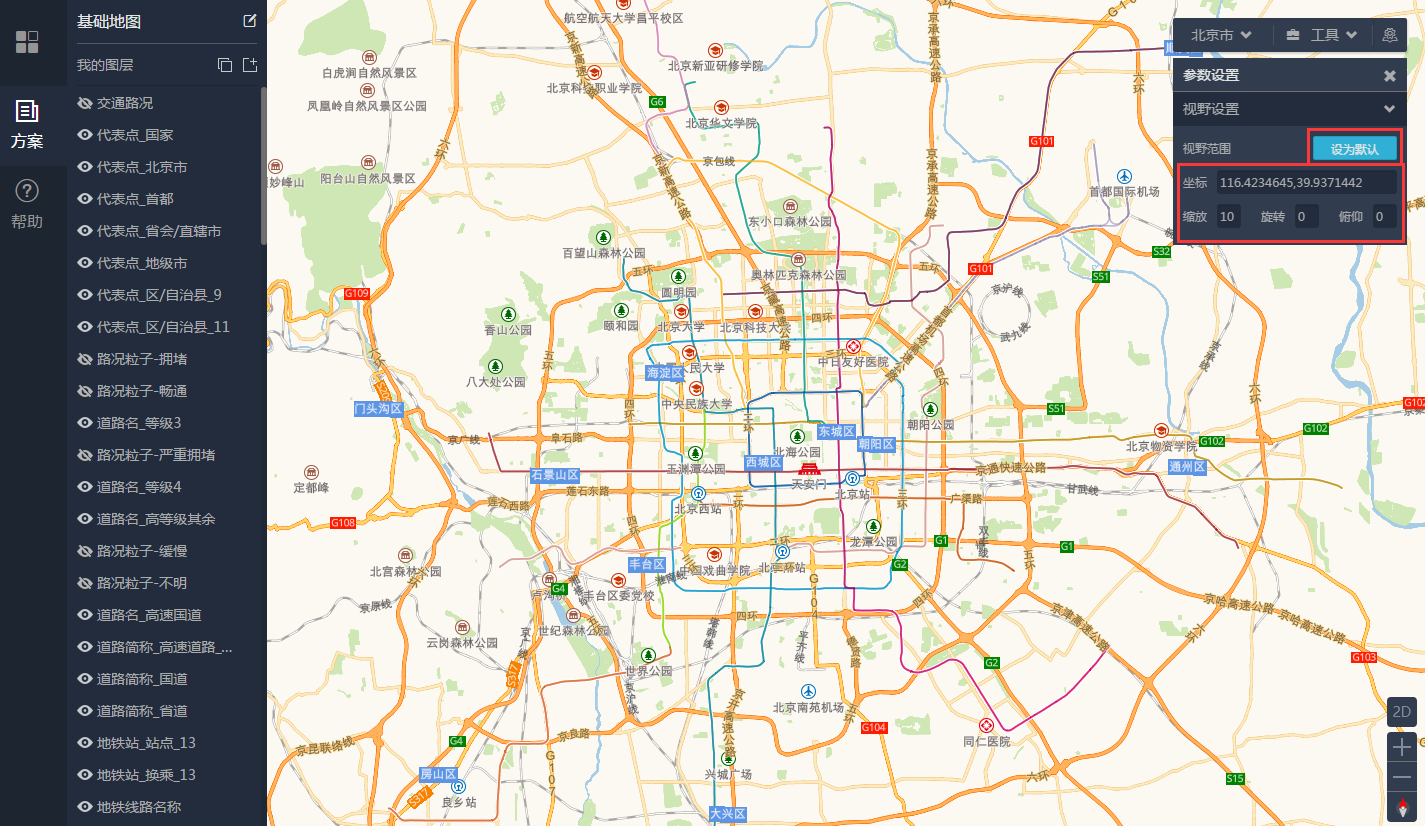


删除图层：



第三步 更改地图初始化信息

地图在初始化时会设置默认的参数，我们可以通过地图右上侧的参数设置模块修改初始化信息，例如中心点坐标、当前底图级别等参数。



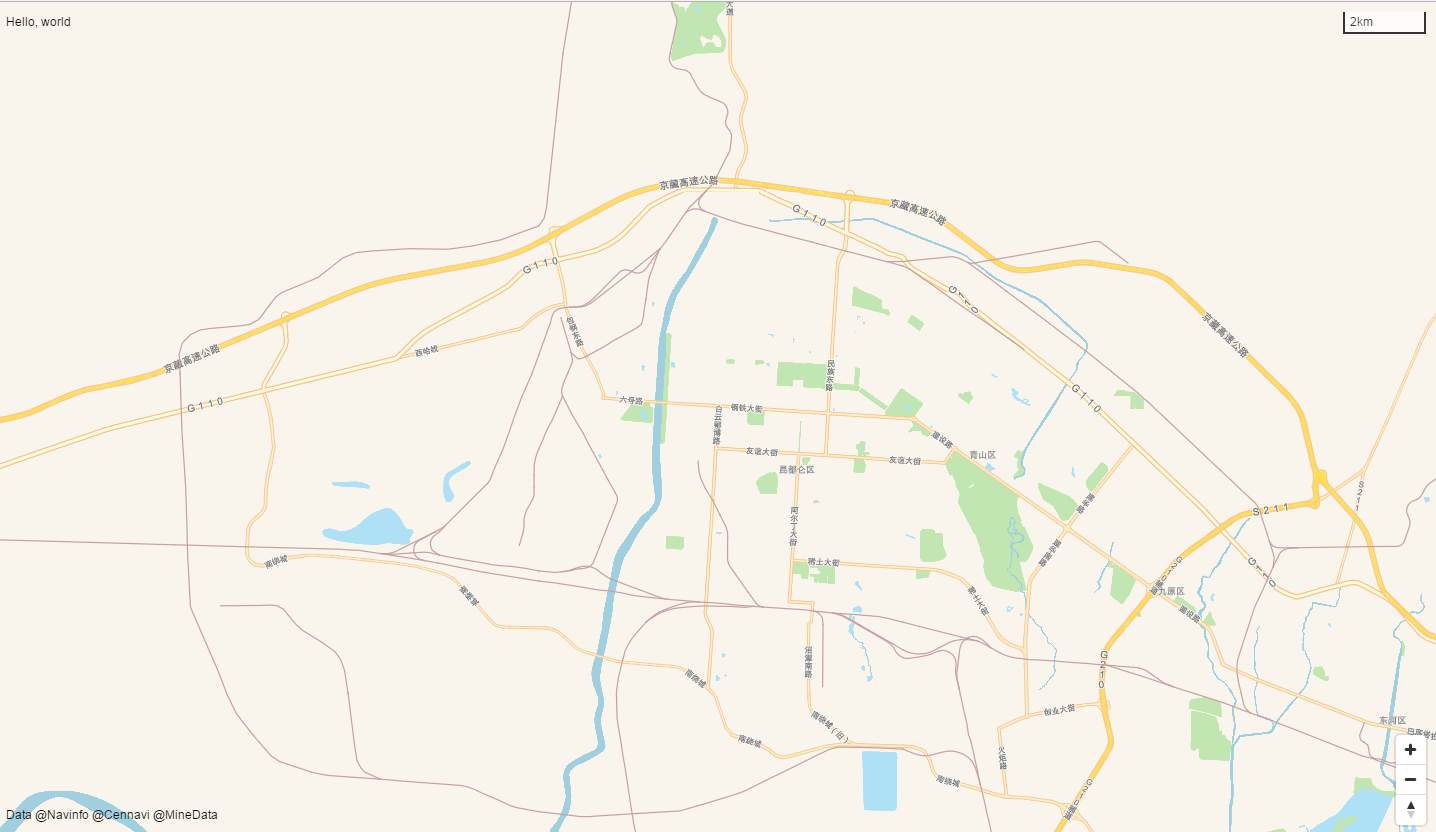
## 初始化定制地图

地图配置好之后，可在URL中获取solution与accessToken的值，可然后通过MineMap API初始化地图，地图初始化时必须设置地图容器container、地图style、地图中心点center、当前地图级别zoom的参数值，否则地图无法显示：



# 地图控件

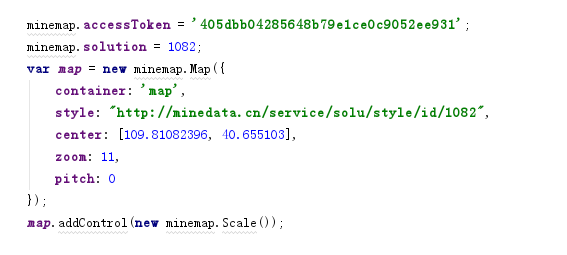
MineMap API提供了比例尺、地图导航工具条等常用的控件，可对地图进行一些操作。



## 添加控件

MineMap通过addControl方法进行控件的初始化与添加。

比例尺控件：



导航控件：



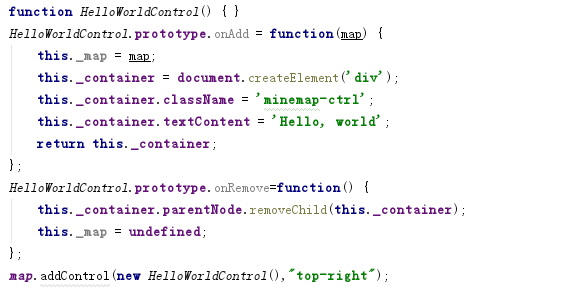
## 控制控件位置

控件的位置是在初始化时，通过参数的形式添加的，位置参数有“top-left”、“top-right”、“bottom-left”、“bottom-right”，例如给导航控件设置位置：



## 自定义控件

MineMap API还可以添加自定义控件，下列为添加自定义控件的javascript代码：



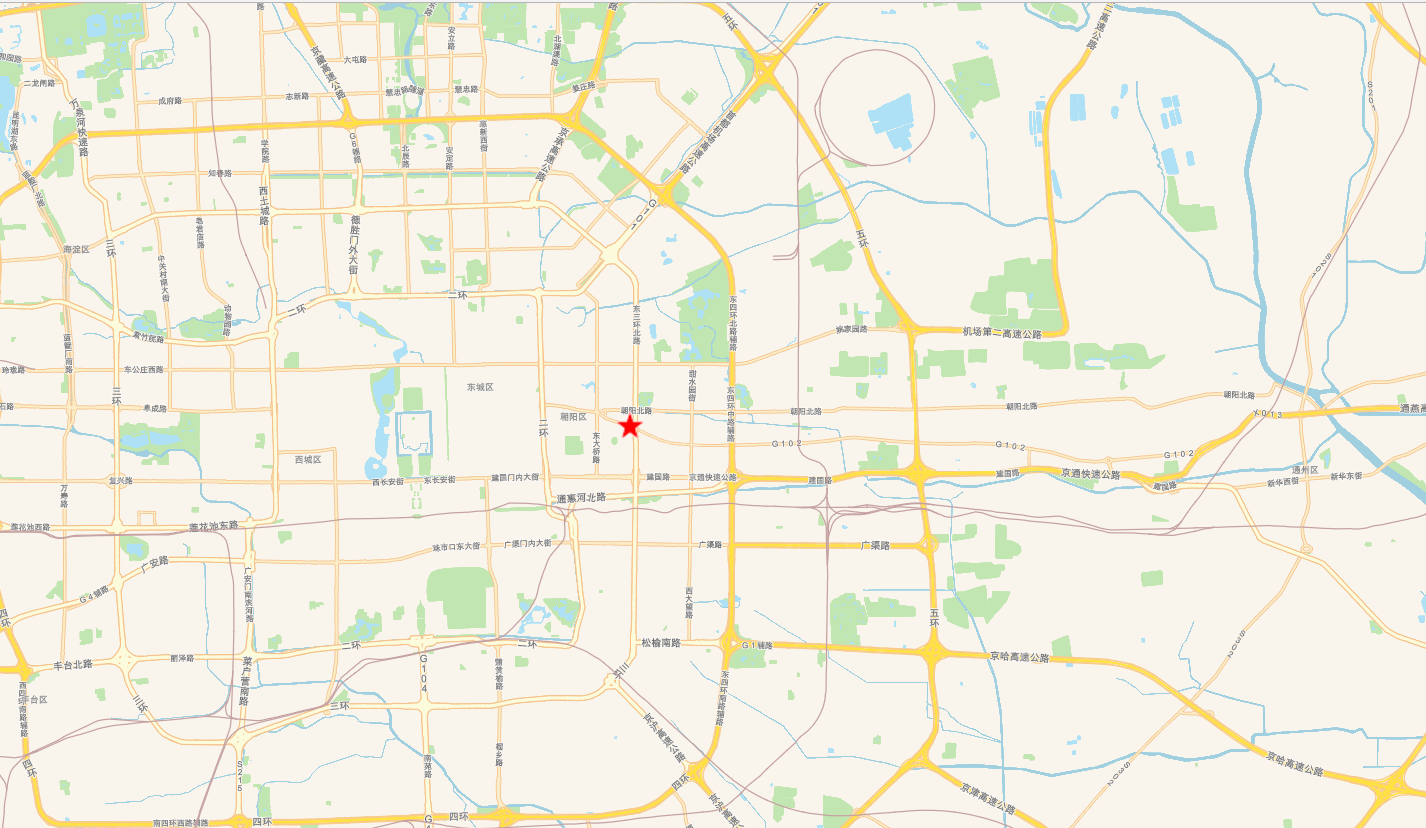
首先定义控件名称HelloWorldControl，然后定义onAdd与onRemove方法，而且控件必须有一个DOM元素，通常是div元素，然后通过addControl方法进行控件的初始化与添加。

# 地图覆盖物

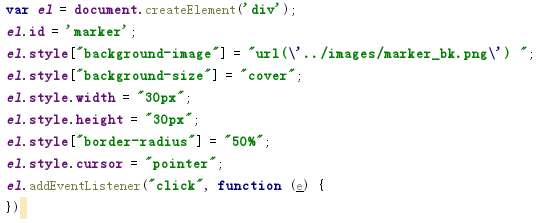
## 生成自定义标注

MineMap用Marker类表示标注，它可以将用户希望看到的兴趣点标注在地图上，Marker上可以添加任意DOM内容。

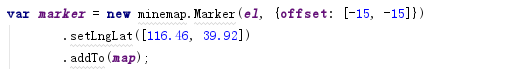
建立一个简单的Marker标注，需要设置DOM元素、Marker坐标参数，并将其添加到地图上。如下例所示：



设置DOM元素：



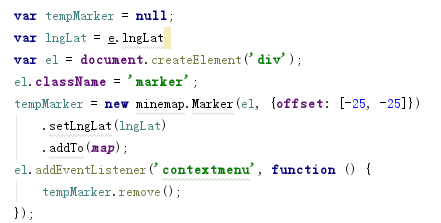
添加标注：



offset参数为标注相对于其左上角偏移像素大小。

## 监听标注事件

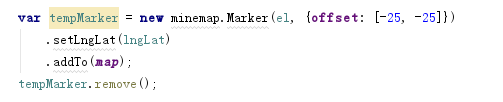
标注的事件绑定是通过Marker的DOM元素绑定的，例如监听右键事件如下例所示：



在标注的DOM元素el中绑定contextmenu事件，当在标注上右键时，该标注会被移除。

## 删除标注

标注的删除是通过MineMap API的remove方法实现的，如下例所示：



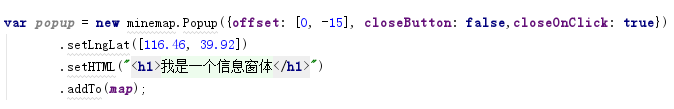
## 添加信息窗体

MineMap用Popup类表示信息窗体，信息窗体一般用来对地图要素的详细说明。

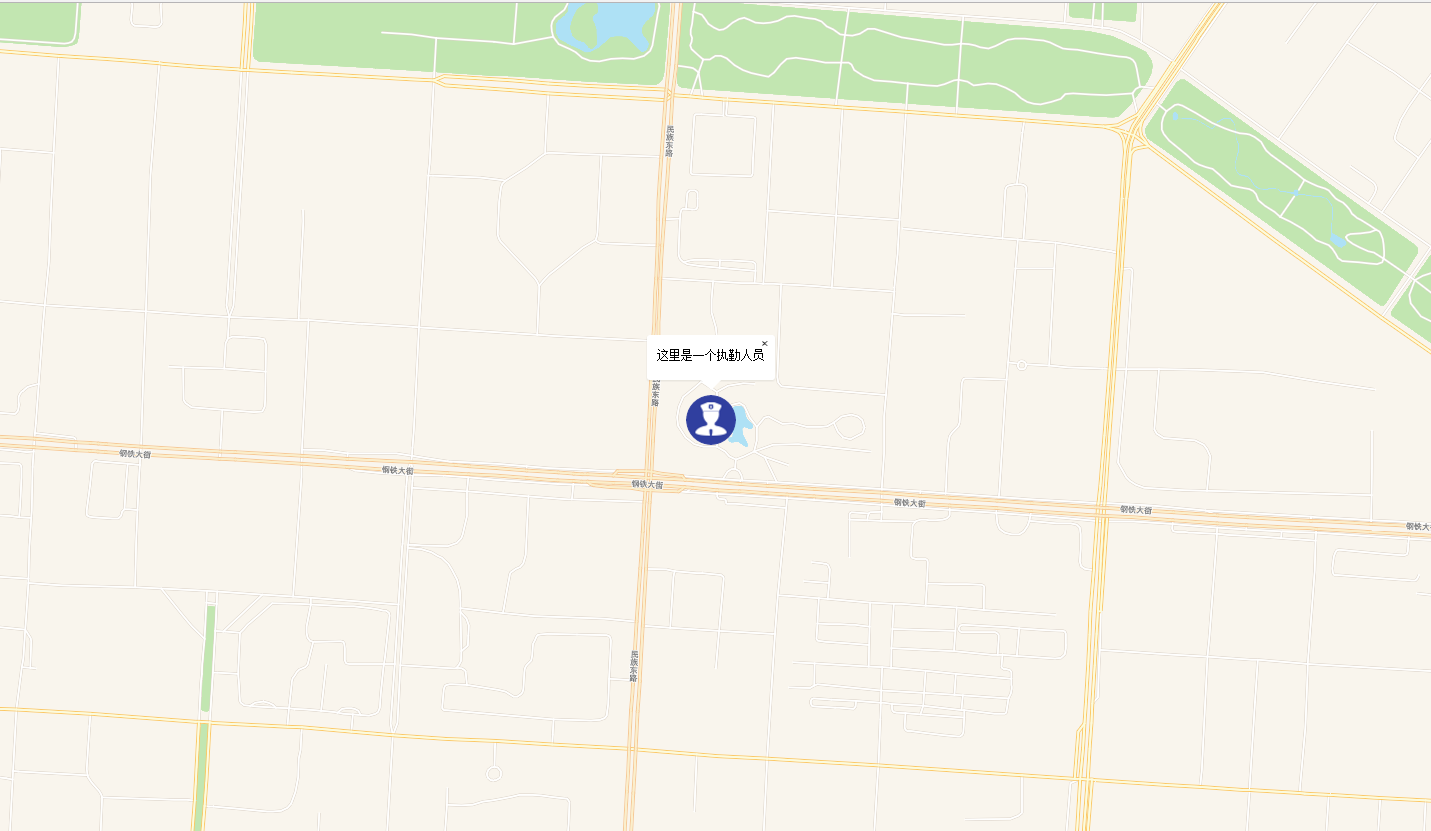
信息窗体初始化时可以设置三个参数：（1）closeButton，true表示会展示一个关闭按钮；（2）closeOnClick，设置为true表示当地图被点击时该信息窗体会被关闭；（3）offset，参数为标注相对于其左上角偏移像素大小。

信息窗体中可以添加的内容有三种（1）文本类型，通过setText方法添加，添加文本类型则无法改变文本样式；（2）DOM类型，通过setDOMContent方法添加，可以通过DOM自定义样式；（3）HTML字符串类型，通过setHTML方法添加，可以通过HTML字符自定义样式。如下例所示：





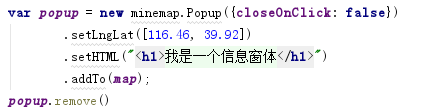
信息窗体还可以与标注结合使用，在Marker中通过setPopup方法可以为该标注添加信息窗体，如下例所示：





## 删除信息窗体

标注的删除是通过MineMap API的remove方法实现的，如下例所示：



# 地图图层

MineMap API支持丰富的图层展示，如路网图、实时交通图、3D建筑物图、图片等，同时可以添加自定义图层。图层的添加分两步完成，第一步，添加图层数据，MineMap API通过addSource实现图层数据的添加，数据源type的类别有“vector”、“raster”、“geojson”、“imge”等；第二步，添加图层样式，MineMap API通过addLayer实现图层样式的添加，数据源type的不同会对应不同的图层type，下表1详细阐述了数据源type与图层type对应关系：

表1 数据源type与图层type对应关系

|  |  |
| --- | --- |
| Source type | Layer type |
| vector | symbol、fill、line、circle、sprite、extrusion |
| geojson | symbol、fill、line、circle、sprite、extrusion |
| raster | raster |
| image | raster |

下面介绍添加矢量、Geojson、栅格、图像图层以及修改图层样式、删除图层的方法。

## 添加矢量图层

矢量图层数据添加：

MineMap API通过addSource方法实现数据源的添加，具体为addSource(id,source)，id为开发人员自定义数据源的唯一标识，source为数据源的具体信息，包括类别type和瓦片来源tiles，如下例所示：

map.**addSource**(**"regionSource"**, {  
 **"type"**: **"vector"**,  
 **"tiles"**:[**'http://123.203.124.207:8083/mapspotter/view/provinceface/{z}/{x}/{y}'**]  
})

如表1所示，矢量图层的图层样式分面（fill）、线、标识、圆、粒子、建筑物这几种类型，下面将分别介绍这几种类型的图层样式添加方法。

### 面图层样式添加

map.addLayer({

**'id'**: **'cityLayer'**,

**'type'**: **'fill'**,

**'source'**: **'citySource'**,

**"source-layer"**: **"Cityface"**,

**'layout'**: {

**"visibility"**: **"none"**,

},

**'paint'**: {

**'fill-color'**: color,

**'fill-opacity'**: 0.8,

**'fill-outline-color'**: **'#fafdff'**

}, **filter**: [**"<="**, **"count"**, 100]

})

面图层样式的添加最基本的包括图层唯一标识id、图层样式类型type、数据源source、数据源图层source-layer、布局layout、绘制paint、过滤fliter等属性，下面详细阐述各个属性。

1. id

id表示该图层的唯一标识，值为字符串

1. type

type表示该图层的类别，面图层值为“fill”

1. source

source表示添加的数据源的唯一标识

1. source-layer

source-layer为数据源中定义的每个图层数据的唯一标识

1. layout

面图层layout中包含的属性只有visibility，值为“visible”表示该图层显示，为“none”表示隐藏该图层。

1. paint

面图层paint中包含的属性有：

1. 'fill-color'：表示面的填充颜色，值可以是rgb值，也可以是十六进制颜色值；
2. 'fill-outline-color'：表示面边框的颜色，值可以是rgb值，也可以是十六进制颜色值；
3. 'fill-opacity'：表示面颜色的透明度，与'fill-color'搭配使用，值为数值；
4. 'fill-pattern'：表示面填充文理，值为sprite图中某个图标的名称，该图标的长、宽值必须为2的倍数；
5. 'fill-antialias'：表示该面填充是否需要抗锯齿，值为布尔值，默认为true；
6. 'fill-translate'：表示面的几何偏移，值为数组[x,y]，x若为正值，表示向右，负值表示向左，y若为正值，表示向上，负值表示向下；
7. 'fill-translate-anchor'：表示面图层旋转点，需要与'fill-translate'值存在，否则无效，值为数组[x,y]；
8. fliter

fliter表示从所有图层过滤出特定特征的图层，有以下几种过滤形式：

1. 存在过滤：

计算形式：[way, key]

存在过滤主要有“has”、“!has”两种形式。

例：

["has", "count"]，表示过滤出存在属性"count"的所有的图层。

["has","count"]，表示过滤出不存在属性"count"的所有的图层。

1. 比较过滤

计算形式：[way, key, value]

比较过滤主要有等于“= =”、大于“>”、小于“<”、不等于“!=”、大于等于“>=”、小于等于“<=”几种形式。

例：

["==", "count", "1000"]，表示过滤出属性"count"值为1000的图层显示。

1. 成员过滤：

计算形式：[way, key, v0,v1，…，vn]

成员过滤主要有等于"in"、"!in"两种形式。

例：

["in", "name", "point", "fill", "line"]，表示过滤出属性" name "值为"point", "fill", "line"其中任一一个的图层。

1. 组合过滤：

计算形式：[way, f0,f1，…，fn]

组合过滤主要有等于"all"、"any"、"none"三种形式。"all"表示过滤出满足所有过滤条件的图层，"any"表示过滤出满足任一一个过滤条件的图层，"none"表示过滤出不满足所有过滤条件的图层。

例：

["all", ["<=", "count", 34], [">", "count", 100]]，表示过滤出属性" count"满足大于34，小于100的图层。

### 线图层样式添加

map.addLayer({

**"id"**: **"state-fills-hover"**,

**"type"**: **"line"**,  
 **"source"**: **"regionSource"**,  
 **"source-layer"**: **"Provinceface"**,  
 **"layout"**: {},  
 **"paint"**: {  
 **"line-color"**: **"#00ff00"**,  
 **"line-width"**: 1  
 },  
 **"filter"**: [**"=="**, **"c\_code"**, **""**]  
});

线图层样式的添加最基本的包括图层唯一标识id、图层样式类型type、数据源source、数据源图层source-layer、布局layout、绘制paint、过滤fliter等属性，下面详细阐述各个属性。

1） id

id表示该图层的唯一标识，值为字符串

2） type

type表示该图层的类别，线图层值为“line”

3） source

source表示添加的数据源的唯一标识

4） source-layer

source-layer为数据源中定义的每个图层数据的唯一标识

5） layout

线图层layout中包含的属性有

* + - * 1. visibility：值为“visible”表示显示该图层，值为“none”表示隐藏该图层；
        2. 'line-cap'：表示线末端的样式，值主要有三个值“butt”、“round”、“square”，分别表示默认圆角、默认方角、线宽一半长的方角；
        3. 'line-join'：表示线相交处的样式，值主要有三个值“bevel”、“round”、“miter”。“bevel”表示连接处为以线宽一半为长度的方角，“round”表示连接处为以线宽一半为半径的圆角，“miter”表示连接处以尖锐角形式连接；

6） paint

线图层paint中包含的属性有：

1. 'line-color'：表示线的填充颜色，值可以是rgb值，也可以是十六进制颜色值；
2. ' line -opacity'：表示线颜色的透明度，与'line -color'搭配使用，值为数值，默认值为1；
3. line -pattern'：表示线填充文理，值为sprite图中某个图标的名称，该图标的长、宽值必须为2的倍数；
4. line-translate'：表示线的几何偏移，值为数组[x,y]，x若为正值，表示向右，负值表示向左，y若为正值，表示向上，负值表示向下；
5. line-translate-anchor'：表示线图层旋转点，需要与line-translate'值存在，否则无效，值为数组[x,y]；
6. 'line-width'：表示线的宽度，值为数值，默认值为1；
7. 'line-gap-width'：表示线的内部间隙的宽度，值为数值，默认值为0；
8. 'line-offset'：表示线位置的偏移，默认值为0，正值表示相对于线方向向右偏移，负值表示向左偏移；
9. 'line-blur'：表示绘制线阴影，以像素为单位，默认值为0；

7） fliter

fliter表示从所有图层过滤出特定特征的图层，线图层的过滤属性用法与面图层用法一致，可参考面图层的filter属性说明内容。

### 标识图层样式添加

map.addLayer({  
 **"id"**: **"provincePointLayer"**,  
 **"type"**: **"symbol"**,  
 **"source"**: **"provincePoint"**,  
 **"layout"**: {  
 **"icon-image"**: **"circle-red-11"**,  
 **"text-field"**: **"{name}"**,  
 **"text-offset"**: [0.6, 0],  
 **"text-anchor"**: **"left"**,  
  
 },  
 **"paint"**: {  
 **"text-color"**: **"#666666"**,  
 }  
});

标识图层样式的添加最基本的包括图层唯一标识id、图层样式类型type、数据源source、数据源图层source-layer、布局layout、绘制paint、过滤fliter等属性，下面详细阐述各个属性。

1. id

id表示该图层的唯一标识，值为字符串

1. type

type表示该图层的类别，标识图层值为“symbol”

1. source

source表示添加的数据源的唯一标识

1. source-layer

source-layer为数据源中定义的每个图层数据的唯一标识

1. layout

标识图层layout中包含的属性有：

1. 'icon-image'：表示图标的图像，值为sprite图中某个图标的名称，若该名称为变量引用，则用“{tokens}”表示。
2. 'icon-size'：表示图标的大小，值为数值，1表示原始大小，3表示原始图像放大3倍；
3. 'icon-rotate'：表示顺时针旋转图标，值为数值，默认值为0，'icon-image'有值时才有效；
4. 'icon-padding：表示图标之间的距离，值为数值，默认值为2，'icon-image'有值时才有效；
5. 'icon-offset：表示图标的几何偏移，值为数组[x,y]，x若为正值，表示向右，负值表示向左，y若为正值，表示向上，负值表示向下，'icon-image'有值时才有效；
6. '[icon-allow-overlap](https://www.mapbox.com/mapbox-gl-js/style-spec/#layout-symbol-icon-allow-overlap)'：表示图标允许重叠，值为布尔值，值为true表示允许重叠，false表示不允许重叠；
7. '[icon-ignore-placement](https://www.mapbox.com/mapbox-gl-js/style-spec/#layout-symbol-icon-ignore-placement)'：表示图标忽略压盖，值为布尔值，值为true表示忽略压盖，false表示不忽略压盖；
8. "icon-optional"：表示图标掩盖文本显示，值为布尔值，值为true表示图标掩盖文本显示，false表示不掩盖文本显示；
9. "symbol-spacing"：表示图标间隔，值为数值，单位为像素（px），默认值为250px；
10. "symbol-avoid-edges"：表示是否允许图标跨瓦片，值为布尔值，值为true表示允许跨瓦片，false表示不允许跨瓦片；
11. "icon-rotation-alignment"：表示图标旋转锚点，值为字符串，值域为"map"和"viewport"；
12. "icon-padding"：表示图标内边距，值为数值，单位为像素（px），默认值为2；
13. "icon-keep-upright"：表示图标是否保持直立，值为布尔值，值为true表示保持直立，false表示不保持直立；
14. "text-font"：表示文字字体，值为字符串，有"Open Sans Regular"、"Arial Unicode MS Regular"；
15. "text-size"：表示文字大小，值为数值，单位为像素（px），默认值为10；
16. "text-letter-spacing"：表示文字间隔，值为数值，单位为像素（px），默认值为250px；
17. "text-line-height"：表示文字的行高，值为数值，默认值为1.2；
18. "text-max-width"：表示文本最大宽度，值为数值，单位为像素（px），默认值为10
19. "text-justify"：表示文本的对齐方式，值为字符串，"left"表示左对齐，"center"表示中心对齐，
20. "text-rotate"：表示文字旋转角度，值为数值，默认值为0；
21. "text-max-angle"：表示文字最大旋转角度，值为数值，单位为度，默认自为45；
22. "text-padding"：表示文本内边距，值为数值，单位为像素（px），默认值为2；
23. "text-anchor"：表示文本锚点，值为字符串，值域有"center"、"top"、"bottom"、"left"、"right"、"top-left"、"top-right"、"bottom-left"、"bottom-right"
24. "text-offset"：表示文本偏移，值为数值，默认值为0；
25. "text-pitch-alignment"：表示文本倾斜锚点，值为字符串，值域为"map"、"viewport"；
26. "text-rotation-alignment"：表示文本旋转锚点，值为字符串，值域为"map"、"viewport"；
27. "text-allow-overlap"：表示文本允许重叠，值为布尔值，值为true表示允许重叠，false表示不允许重叠；
28. "text-ignore-placement"：表示文本忽略压盖，值为布尔值，值为true表示忽略压盖，false表示不忽略压盖；
29. "text-optional"：表示文本掩盖图标显示，值为布尔值，值为true表示文本掩盖图标显示，false表示不掩盖图标显示；
30. "text-keep-upright"：表示文本是否保持直立，值为布尔值，值为true表示保持直立，false表示不保持直立；
31. "text-transform"：表示文本大小写转换，值为字符串，值域为"none"、"uppercase"、"lowercase"，"none"表示无变化，"uppercase"表示转为大写，"lowercase"表示转为小写，默认值为"none"；
    * + - 1. "visibility"：值为“visible”表示显示该图层，值为“none”表示隐藏该图层；
32. paint

标识图层paint中包含的属性有：

1. "icon-opacity"：表示图标不透明度，值为数值，默认值为1，"icon-image"必须有值才有效；
2. "icon-color"：表示图标的颜色，默认值为"#000000"，"icon-image"必须有值才有效；
3. "icon-halo-color"：表示图标边框颜色，默认值为"#000000"，"icon-image"必须有值才有效；
4. "icon-halo-width"：表示图标边框宽度，默认值0，"icon-image"必须有值才有效；
5. "icon-halo-blur"：表示图标边框模糊度（%），默认值0，"icon-image"必须有值才有效；
6. "icon-translate"：表示图标偏移，默认值0，"icon-image"必须有值才有效；
7. "icon-translate-anchor"：表示图标偏移锚点，值为字符串，值域为"map"、"viewport"，默认值为"map"；
8. "text-opacity"：表示文本不透明度（%），值为数值，默认值为1，"text-field"必须有值才有效；
9. "text-color"：表示文本颜色，默认值为"#000000"，"text-field"必须有值才有效；
10. "text-halo-color"：表示文本边框颜色，默认值0，"text-field"必须有值才有效；
11. "text-halo-width"：表示文本边框宽度，默认值0，"text-field"必须有值才有效；
12. "text-halo-blur"：表示文本边框模糊度（%），默认值0，"text-field"必须有值才有效；
13. "text-translate"：表示文本偏移，默认值0，"text-field"必须有值才有效；
14. "text-translate-anchor"：表示文本偏移锚点，值为字符串，值域为"map"、"viewport"，默认值为"map"；
15. filter

fliter表示从所有图层过滤出特定特征的图层，线图层的过滤属性用法与面图层用法一致，可参考面图层的filter属性说明内容。

### 圆图层样式添加

map.addLayer({  
 **"id"**: **'finalspointlayer'**,  
 **"type"**: **"circle"**,  
 **"source"**: **"finalspoint"**,  
 **"source-layer"**:**"pointlayer"**,  
 **'paint'**: {  
 **'circle-radius'**: {  
 **'base'**: 1,  
 **'stops'**: [[8, 1.5], [13, 1.5]]  
 },  
 **'circle-color'**: **"#CD6600"**,  
 **'circle-opacity'**: 1,  
 },  
 **"layout"**: {  
 **"visibility"**: **"none"** },  
 **"minzoom"**: 11,  
 **"maxzoom"**: 12.5,  
});

圆图层样式的添加最基本的包括图层唯一标识id、图层样式类型type、数据源source、数据源图层source-layer、布局layout、绘制paint、过滤fliter等属性，下面详细阐述各个属性。

1. id

id表示该图层的唯一标识，值为字符串

1. type

type表示该图层的类别，线图层值为“circle”

1. source

source表示添加的数据源的唯一标识

1. source-layer

source-layer为数据源中定义的每个图层数据的唯一标识

1. layout

圆图层layout中包含的属性有：

visibility：值为“visible”表示显示该图层，值为“none”表示隐藏该图层；

1. paint
2. "circle-radius"：表示圆的半径，值为数值，默认为5；
3. "circle-color"：表示圆的颜色，默认为"#000000"；
4. "circle-blur"：表示圆的模糊度（%），值为数值，默认为0；
5. "circle-opacity"：表示圆的透明度，值为数值，默认为1；
6. "circle-translate"表示圆的偏移，值为数值，默认为0；
7. "circle-translate-anchor"：表示圆的偏移锚点，值为字符串，值域为"map"、"viewport"，默认值为"map"；
8. "circle-pitch-scale"：表示倾斜方向，值为字符串，值域为"map"、"viewport"，默认值为"map"；
9. "circle-stroke-width"：表示圆的边框宽度，值为数值，默认为0；
10. "circle-stroke-color"：表示圆的边框颜色，默认值为"#000000"；
11. "circle-stroke-opacity"：表示圆的边框透明度（%），值为数值，默认为1
12. fliter

fliter表示从所有图层过滤出特定特征的图层，线图层的过滤属性用法与面图层用法一致，可参考面图层的filter属性说明内容。

### 粒子图层样式添加

map.addLayer({  
 **"id"**: **"linkspriteLayer"**,  
 **"type"**: **"sprite"**,  
 **"source"**: **"spriteSource"**,  
 **"spurce-layer"**: **"spriteLayer"**,  
 **"layout"**: {  
 **"sprite-cap"**: **"round"**,  
 **"sprite-join"**: **"round"** },  
 **"paint"**: {  
 **"sprite-color"**: **"#ff0000"**,  
 **"sprite-width"**: 1,  
 **"sprite-speed-factor"**: 1  
 }  
})

粒子图层样式的添加最基本的包括图层唯一标识id、图层样式类型type、数据源source、数据源图层source-layer、布局layout、绘制paint、过滤fliter等属性，下面详细阐述各个属性。

1. id

id表示该图层的唯一标识，值为字符串

1. type

type表示该图层的类别，线图层值为“sprite”

1. source

source表示添加的数据源的唯一标识

1. source-layer

source-layer为数据源中定义的每个图层数据的唯一标识

1. layout
2. "sprite-cap"：表示粒子线的末端样式，值域有"butt"、"round"、"square"，分别表示对接、圆角、直角，默认值为"butt"；
3. "sprite-join"：表示粒子线的连接方式，值域有"bevel"、"round"、"miter"，分别表示斜角、圆角、直角，默认值为"bevel"；
4. "sprite-miter-limit"：表示粒子线的斜切限制，值为数值，默认值为2；
5. "sprite-round-limit"：表示粒子线的圆角限制，值为数值，默认值为1.05；
6. "visibility"：表示图层是否可见，值为“visible”表示显示该图层，值为“none”表示隐藏该图层；
7. paint
8. "sprite-color"：表示粒子线的颜色，默认为"#000000"；
9. "sprite-opacity"：表示粒子线的透明度，值为数值，默认为1；
10. "sprite-translate"：表示粒子线的偏移，值为数值，默认为0；
11. "sprite-translate-anchor"：表示粒子线的偏移锚点，值为字符串，值域为"map"、"viewport"，默认值为"map"；
12. "sprite-width"：表示粒子线的宽度，值为数值，单位为像素，默认值为1；
13. "sprite-offset"表示粒子线的偏移，值为数值，默认值为0；
14. "sprite-blur"：表示粒子线的模糊度（%），值为数值，默认为0；
15. "sprite-speed-factor"：表示路况，值为数值，值域有1、2、3、4、5，分别表示“顺畅”、“缓慢”、“拥堵”、“严重拥堵”、“不明”，默认值为1；
16. "sprite-gap-width"：表示粒子间隙宽度，值为数值，默认值为0
17. fliter

fliter表示从所有图层过滤出特定特征的图层，线图层的过滤属性用法与面图层用法一致，可参考面图层的filter属性说明内容。

### 建筑物图层添加

map.addLayer({  
 **"id"**: **"extrusionLayer"**,  
 **"type"**: **"extrusion"**,  
 **"source"**: **"extrusionSource"**,  
 **"spurce-layer"**: **"extrusionface"**,  
 **"layout"**: {  
 },  
 **"paint"**: {  
 **"extrusion-color"**: **"#FFEFD5"**,  
 **"extrusion-opacity"**: 1,  
 **"extrusion-height"**: {  
 **"type"**: **"identity"**,  
 **"property"**: **"levels"** },  
 **"extrusion-base"**: 0,  
 **"extrusion-translate"**: [0,0],  
 **"extrusion-translate-anchor"**: **"map"** }  
})

建筑物图层样式的添加最基本的包括图层唯一标识id、图层样式类型type、数据源source、数据源图层source-layer、布局layout、绘制paint、过滤fliter等属性，下面详细阐述各个属性。

* + 1. id

id表示该图层的唯一标识，值为字符串

* + 1. type

type表示该图层的类别，建筑物图层值为 “extrusion”

* + 1. source

source表示添加的数据源的唯一标识

* + 1. source-layer

source-layer为数据源中定义的每个图层数据的唯一标识

* + 1. layout

建筑物图层layout中包含的属性只有visibility，值为“visible”表示显示该图层，值为“none”表示隐藏该图层；

* + 1. paint

建筑物图层paint中包含的属性有：

1. "extrusion-color"：表示建筑物颜色，默认值为"#000000"；
2. "extrusion-opacity"：表示建筑物的透明度，值为数值，默认为1；
3. "extrusion-translate"：表示建筑物的偏移，值为数值，默认为0；
4. "extrusion-translate-anchor"：表示建筑物的偏移锚点，值为字符串，值域为"map"、"viewport"，默认值为"map"；
   * 1. filter

fliter表示从所有图层过滤出特定特征的图层，线图层的过滤属性用法与面图层用法一致，可参考面图层的filter属性说明内容。

## 添加GeoJson图层

Geojson图层数据添加：

MineMap API通过addSource方法实现数据源的添加，具体为addSource(id,source)，id为开发人员自定义数据源的唯一标识，source为数据源的具体信息，包括类别type和瓦片来源data，type值必须为“geojson”，data值为gsojson数据所在地址，如下例所示：

map.**addSource**(**"provincePoint"**, {  
 **"type"**: **"geojson"**,  
 **"data"**: **"data/index/province\_point.json"**})

如表1所示，geojson图层的图层样式分面（fill）、线、标识、圆、粒子这几种类型，与矢量图层的图层样式相同，图层的属性除了没有“source-layer”之外，其他都相同，因此添加方法也一样，**此处可参考矢量图层的图层样式添加方法**。

## 添加栅格图层

栅格图层数据添加：

MineMap API通过addSource方法实现数据源的添加，具体为addSource(id,source)，id为开发人员自定义数据源的唯一标识，source为数据源的具体信息，包括类别type和瓦片来源tiles，type值必须为“raster”，“tiles”值为栅格数据所在地址如下例所示：

map.**addSource**(**"originaltrail"**,{  
 **"type"**:**"raster"**,  
 **"tileSize"**:256,

**"tiles"**:

[**"http://minemap.navinfo.com/service160/heatmap/show?source=sogou\_0523\_0605&x={x}&y={y}&z={z}"**]  
})

如表1所示，栅格图层的图层样式主要有raster，下面将介绍图raster层样式添加方法。

raster图层样式数据添加：

map.addLayer({  
 **'id'**: **"originaltrail"**,  
 **'type'**: **'raster'**,  
 **'source'**: **'originaltrail'**,  
 **"source-layer"**: **"Sougou0originaltrail"**,  
 **"layout"**: {  
 **"visibility"**: **"visible"** },  
})

raster图层样式的添加最基本的包括图层唯一标识id、图层样式类型type、数据源source、布局layout、绘制paint等属性，下面详细阐述各个属性。

1. id

id表示该图层的唯一标识，值为字符串

1. type

type表示该图层的类别，线图层值为“raster”

1. source

source表示添加的数据源的唯一标识

1. layout

栅格图层layout中包含的属性有

"visibility"：表示图层是否可见，值为“visible”表示显示该图层，值为“none”表示隐藏该图；

1. paint

栅格图层paint中包含的属性有：

1. "raster-opacity"：表示栅格要素的透明度，值为数值，默认为1；
2. "raster-hue-rotate"：表示栅格要素的旋转角度，值为数值，默认值为0；
3. "raster-brightness-min"：表示栅格要素的最低亮度（%），值为数值，值域为0-100，默认值为0；
4. "raster-brightness-max"：表示栅格要素的最高亮度（%），值为数值，值域为0-100，默认值为100；
5. "raster-saturation"：表示栅格要素的饱和度（%），值为数值，值域为0-100，默认值为0；
6. "raster-contrast"：表示栅格要素的对比度（%）,值为数值，值域为0-100，默认值为0；
7. "raster-fade-duration"：表示栅格要素的图层褪去时间（ms），值为数值，默认值为300；
8. filter

fliter表示从所有图层过滤出特定特征的图层，线图层的过滤属性用法与面图层用法一致，可参考面图层的filter属性说明内容。

## 添加图像图层

图像图层数据添加：

MineMap API通过addSource方法实现数据源的添加，具体为addSource(id,source)，id为开发人员自定义数据源的唯一标识，source为数据源的具体信息，包括类别type和瓦片来源tiles，type值必须为“raster”，“tiles”值为栅格数据所在地址如下例所示：

map.**addSource**({  
 **"type"**: **"image"**,  
 **"url"**: **"http://minemap.navinfo.com/assets/radar.gif"**,  
 **"coordinates"**: [  
 [-80.425, 46.437],  
 [-71.516, 46.437],  
 [-71.516, 37.936],  
 [-80.425, 37.936]  
 ]  
})

如表1所示，栅格图层的图层样式主要有raster，下面将介绍图raster层样式添加方法。

图像图层样式添加：

map.addLayer({  
 **"id"**: **"overlay"**,  
 **"source"**: **"overlay"**,  
 **"type"**: **"raster"**,  
 **"paint"**: {  
 **"raster-opacity"**: 0.85  
 }  
})

图像图层样式的添加与栅格图层样式的添加是相同的，**此处可以参考栅格图层的添加方法**。

## 添加背景图层

背景图层的添加不需要数据，只需要添加图层就可以，如下例所示：

map.addLayer({  
 **"id"**: **"backgroundLayer"**,  
 **"type"**: **"background"**,  
 **"layout"**: {  
 **"visibility"**: **"visible"** },  
 **"paint"**: {  
 **"background-opacity"**: 0.85,  
 **"background-color"**:**"#5f4a1f"** }  
})

背景图层样式的添加最基本的包括图层唯一标识id、图层样式类型type、布局layout、绘制paint等属性，下面详细阐述各个属性。

1. id

id表示该图层的唯一标识，值为字符串

1. type

type表示该图层的类别，线图层值为“background”

1. layout

背景图层layout包含的属性只有visibility，值为“visible”表示该图层显示，为“none”表示隐藏该图层。

1. paint

背景图层paint中包含的属性有：

1. "background-color"：表示背景图层颜色，默认值为"#000000"；
2. "background-pattern"：表示背景图层的文理，值为图标；
3. "background-opacity"：表示背景图层的不透明度（%），值域为0-1，默认值为1；

## 删除图层

图层的删除分两种情况（1）只删除图层样式；（2）图层样式与图层数据都删除。MineMapAPI中用removeSource()方法删除数据源，removelayer方法删除图层样式。

删除数据源：



删除图层样式：



可以同时删除数据源和图层样式，或者只删除图层，但不可以只删除数据源，否则加载图层样式时会报错

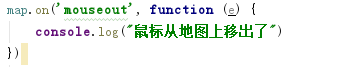
# 地图交互

## 地图事件类型

MineMap API提供了完善的事件系统来满足开发者的各种需求，主要包括地图事件与鼠标事件两大类，下面将介绍地图的事件类型。

1. mouseout

当鼠标移出地图时触发的事件



1. mousedown

鼠标按下时触发的事件



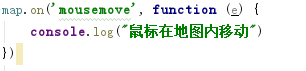
1. mouseup

鼠标按键释放时触发的事件



1. mousemove

鼠标在地图内移动时触发的事件



1. click

鼠标在地图上任一位置单击时触发的事件



1. dblclick

鼠标在地图上任一位置双击时触发的事件



1. contextmenu

鼠标在地图上任一位置右键时触发的事件



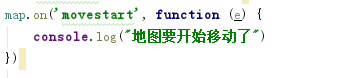
1. load

地图资源加载完成后触发的事件



1. movestart

地图要开始移动，但还未移动时触发的事件，地图移动过程中不触发



1. move

地图移动过程中触发的事件



1. moveend

地图移动一次结束时触发的事件，地图移动过程中不触发



## 事件的绑定和移除

事件的绑定通过MineMap的on()方法实现，移除通过MineMap的off()方法实现。

事件绑定：

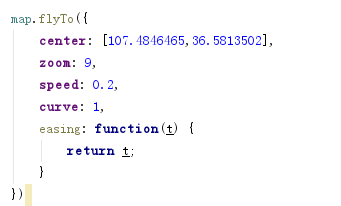
**function** *intro*(){  
 **console**.log(**"事件的绑定与删除"**)  
}  
***map***.on(**"mousemove"**,*intro*)

事件移除：

***map***.off(**"mousemove"**,*intro*)

## 飞行

MineMap API提供了地图飞行的功能，主要有jumpTo()、easeTo()、flyTo() 三种方法，它们可以采用动画的形式进行中心点、缩放级别、旋转角、倾角的改变。下面详细介绍flyTo()方法。



flyTo(options，[eventData])

参数有：

1. center：表示飞行目的地坐标，值为数组；
2. zoom：表示飞行结束后地图的zoom，值为数值；
3. speed：表示飞行速度值为数值；
4. curve：表示飞行线的弧度，值为数值；
5. easing：表示飞行过程中会触发的事件。

## 图层交互

### 获取图层要素信息

MineMap API提供了获取图层要素信息的方法，主要有queryRenderedFeatures()、querySourceFeatures()两个方法，下面将介绍这两个方法。

1. queryRenderedFeatures(geometry，[parameters])

查询指定区域的地图要素信息，以Geojson Feature对象数组形式返回查询结果。

参数说明：

1. geometry：查询区域的几何形状，可以为单个点或者由点组成的面区域。若省略此参数为e，则表示整个视图窗口。
2. [parameters]
3. **[layers]**

要查询的图层id的数组。 仅有设定的图层数据会被查询出来，如果为 null，将会查询所有的图层。

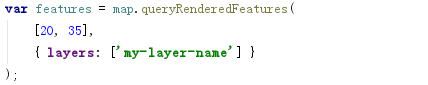
1. **[filter]**

过滤器，用于过滤查询结果。

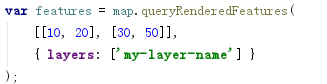
返回值：返回Geojson Feature对象数组，表示满足查询参数的地图要素信息。

例：

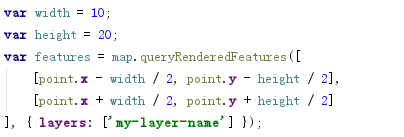
1. 单个点查询



1. 用一个给定的边界框查询地图数据信息



1. 通过制定鼠标点为中心定制一个边界框并查询边界框内的地图数据信息



1. 查询一个图层内的所有地图数据信息



1. querySourceFeatures(sourceId，params)

查询满足条件的所有地图要素信息，与区域无关，以Geojson Feature对象数组形式返回查询结果。

参数说明：

1. sourceID

数据源的id

1. params
2. [sourceLayer]

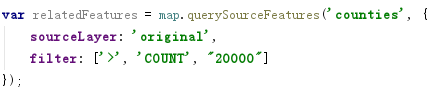
对于矢量数据来说是必须的，对与GeoJSON数据来说该项是被忽略的。

1. [filter]

过滤器，用于过滤查询结果

返回值：返回Geojson Feature对象数组，表示满足查询参数的地图要素数据源信息。

例：



### 修改图层样式

MineMap API提供了修改图层样式的方法，主要有setPaintProperty()、setLayoutProperty()等方法，下面将介绍这几个方法。

* + - 1. setPaintProperty(layer，name，value)

设置或修改图层的paint绘制样式值

1. layer

需要修改的图层id

1. name

需要修改的绘制样式名称

1. value

绘制样式的值

例：



上例修改了图层'my-layer'的'fill-color'样式值为'#faafee'

* + - 1. setLayoutProperty(layer，name，value)

设置或修改图层layout属性值

1. layer

需要修改的图层id

1. name

需要修改的layout属性

1. value

设置layout属性的值

例：



上例修改了图层'my-layer'的'**visibility**'属性值为'none'，即设置该图层不可见。

## 地图控制

MineMap API提供了地图控制的方法，主要有resize()、setMinZoom()、setMaxZoom()、setPitch()等方法，下面将介绍这几个方法。

1. resize()

根据当前容器大小调整地图尺寸，当它的容器已经发生了变化之后或者地图通过css从隐藏变成显示后才可以调用改方法。

1. setMinZoom( minZoom)

设置地图的最小缩放级别 如果当前的缩放级别小于最新设定的最小缩放级别，那么地图将会将当前的缩放级别进行改变并与之匹配。

参数：

minZoom

最小缩放级别（0-20）。如果设定为null或者undefined，地图将会设定为最小缩放级别为0。

1. setMaxZoom(maxZoom)

设置地图的最大缩放级别 如果当前的缩放级别大于最新设定的最大缩放级别，那么地图将会将当前的缩放级别进行改变并与之匹配。

参数：

maxZoom

最大缩放级别 (0-20). 如果设定为null 或 undefined，地图将会设定最大缩放级别为20。

1. setPitch(pitch，[eventData])

设置地图的俯仰角

参数：

1. **pitch**

设置地图于屏幕之间的夹角(0-60)。

1. **[eventData]**

向任何事件监听器传播的数据.

# 服务接口

MineMap API 服务接口提供了数据信息检索的接口，如位置搜索、路径规划、地址解析、智能提示等服务。

## 位置搜索服务

### 根据查询的数据类型进行关键字搜索

**接口方法**：

**minemap**.service.querySearchResultByType (adminCode, keyword, searchType, pageNumber, pageCount, callback)

**使用示例**：

**minemap**.service.querySearchResultByType(**'110000'**,**'肯德基'**,**'poi'**,1,10,**function** (error,results) {  
 *//console.log(results);*});

**参数说明：**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| adminCode | string | 行政区划编码 |
| keyword | string | 关键字 |
| searchType | string | 查询的数据类型 参数说明：all-所有、busline-公交线路、busstation-公交站点、road-道路、cross-交叉路口、poi-poi数据、area-行政区划、apt-address 门址查询 |
| pageNumber | number | 指定返回的页码 大于0 |
| pageCount | number | 指定返回的每页数据个数 大于0 默认值为10 |
| callback | Function | 回调函数 返回查询结果 示例function(error,result){} |

**返回数据格式：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | | | 类型 | | 说明 | |
| errcode | | | int | | 服务返回代码 | |
| errmsg | | | string | | 服务返回消息描述 | |
| data | pageCount | | | int | | 当前页返回记录数 |
| pageNumber | | | int | | 当前页码 |
| totalCount | | | int | | 查询结果总数 |
| rows | distance | | int | | 当周边查询时返回中心点 与poi的距离(米) |
| id | | string | | 结果数据编号 |
| name | | string | | 名称 |
| adcode | | int | | 行政区划编码 |
| resultType | | string | | 结果数据类型： 与请求中searchtype相同 |
| address | | string | | 结果数据地址 |
| tel | | string | | poi电话 |
| dataType | | string | | 数据分类 |
| geom | type | string | | 类型 |
| coordinates | string | | 坐标 |
| geom\_display | type | string | | 类型 |
| coordinates | string | | 坐标 |

### 根据关键字搜索所有符合条件的数据

**接口方法**：

**minemap**.service. queryAllSearchResult(adminCode, keyword, pageNumber, pageCount, callback)

**使用示例**：

**minemap**.service.queryAllSearchResult(**'110000'**,**'肯德基'**,1,10,**function** (error,results) {  
 *//console.log(results);*});

**参数说明：**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| adminCode | string | 行政区划编码 |
| keyword | string | 关键字 |
| pageNumber | number | 指定返回的页码 大于0 |
| pageCount | number | 指定返回的每页数据个数 大于0 默认值为10 |
| callback | Function | 回调函数 返回查询结果 示例function(error,result){} |

**返回数据格式：**

同上

### 根据关键字搜索poi数据

**接口方法**：

**minemap**.service. queryPoiSearchResult(adminCode, keyword, pageNumber, pageCount, callback)

**使用示例**：

**minemap**.service. queryPoiSearchResult (**'110000'**,**'肯德基'**,1,10,**function** (error,results) {  
 *//console.log(results);*});

**参数说明：**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| adminCode | string | 行政区划编码 |
| keyword | string | 关键字 |
| pageNumber | number | 指定返回的页码 大于0 |
| pageCount | number | 指定返回的每页数据个数 大于0 默认值为10 |
| callback | Function | 回调函数 返回查询结果 示例function(error,result){} |

**返回数据格式：**

同上

### 根据关键字搜索道路数据

**接口方法**：

**minemap**.service. queryRoadSearchResult (adminCode, keyword, pageNumber, pageCount, callback)

**使用示例**：

**minemap**.service. queryRoadSearchResult (**'110000'**,**'长安街'**,1,10,**function** (error,results) {  
 *//console.log(results);*});

**参数说明：**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| adminCode | string | 行政区划编码 |
| keyword | string | 关键字 |
| pageNumber | number | 指定返回的页码 大于0 |
| pageCount | number | 指定返回的每页数据个数 大于0 默认值为10 |
| callback | Function | 回调函数 返回查询结果 示例function(error,result){} |

**返回数据格式：**

同上

### 根据关键字和坐标点进行周边搜索

**接口方法**：

**minemap**.service. querySearchSurroundResult (adminCode, keyword, searchType, location, spatialFuntion, radius, seq, pageNumber, pageCount, callback)

**使用示例**：

**minemap**.service. querySearchSurroundResult('110000','肯德基','all','116.40717, 39.90469','geofilt',10000,0,1,10,**function** (error,results) {  
 *//console.log(results);*});

**参数说明：**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| adminCode | string | 行政区划编码 |
| keyword | string | 关键字 |
| searchType | string | searchType 查询的数据类型 参数说明：all-所有、busline-公交线路、busstation-公交站点、road-道路、cross-交叉路口、poi-poi数据、area-行政区划、apt-address 门址查询 |
| location | string | 中心点坐标x,y |
| spatialFuntion | string | 空间查询方式 参数说明：geofilt-周边检索、bbox-正方形检索 |
| radius | number | 搜索半径,默认500米 |
| seq | number | 排序方式 参数说明：1-按照距离排序、0-按照权值排序 |
| pageNumber | number | 指定返回的页码 大于0 |
| pageCount | number | 指定返回的每页数据个数 大于0 默认值为10 |
| callback | Function | 回调函数 返回查询结果 示例function(error,result){} |

**返回数据格式：**

同上

### 根据关键字和矩形框进行范围搜索

**接口方法**：

**minemap**.service. querySearchBboxResult(adminCode, keyword, searchType, bounds, seq, pageNumber, pageCount,callback)

**使用示例**：

**minemap**.service. querySearchBboxResult('110000','肯德基','all','116.4029557,39.9015946,116.4116457,39.9129533',0,1,10,**function** (error,results) {  
 *//console.log(results);*});

**参数说明：**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| adminCode | string | 行政区划编码 |
| keyword | string | 关键字 |
| searchType | string | searchType 查询的数据类型 参数说明：all-所有、busline-公交线路、busstation-公交站点、road-道路、cross-交叉路口、poi-poi数据、area-行政区划、apt-address 门址查询 |
| bounds | string | 矩形范围 左下角和右上角坐标串x1,y1,x2,y2 |
| seq | number | 排序方式 参数说明：1-按照距离排序、0-按照权值排序 |
| pageNumber | number | 指定返回的页码 大于0 |
| pageCount | number | 指定返回的每页数据个数 大于0 默认值为10 |
| callback | Function | 回调函数 返回查询结果 示例function(error,result){} |

**返回数据格式：**

同上

## 查找服务

### 根据坐标点查找周边信息

**接口方法**：

**minemap**.service. querySearchAroundResult (location, type, radius, count, callback)

**使用示例**：

**minemap**.service. querySearchAroundResult (**'116.4029557,39.9015946','poi'**, 1000,5,**function** (error,results) {  
 *//console.log(results);*});

**参数说明：**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| location | string | 坐标点x,y |
| type | string | 查询结果类型 支持多种别，用逗号分割 参数说明：link,poi,road,cross,apt |
| radius | number | 周边搜索的半径 默认 1000米 |
| count | number | 返回记录数(类型不为link时生效)，默认为5 |
| callback | Function | 回调函数 返回查询结果 示例function(error,result){} |

**返回数据格式：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段 | | | | 类型 | 说明 |
| errcode | | | | int | 返回码 |
| errmsg | | | | string | 返回信息描述 |
| data | rows | id | | string | ID POI、LINK、CROSS |
| type | | string | 返回结果类型，值同请求参数 |
| name | | string | 名称 poi link cross |
| address | | string | 地址 |
| tel | | string | 电话 |
| distance | | int | 中心点与poi的距离 |
| direction | | int | 方向,与正北夹角,度单位 |
| geomtry | type | string | 位置形状 |
| coordinates | string |

## geocoding服务

### 根据地址查找geocoding信息

**接口方法**：

**minemap**.service. queryGeoCodingResult (adminCode, address, callback)

**使用示例**：

**minemap**.service. queryGeoCodingResult (**'** **116.37959,39.86541'**, **'** **天安门'**, **function** (error,results) {  
 *//console.log(results);*});

**参数说明：**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| adminCode | string | 行政区划编码 |
| address | string | 查找地址 |
| callback | Function | 回调函数 返回查询结果 示例function(error,result){} |

**返回数据格式：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | | | 类型 | 说明 |
| errcode | | | int | 返回代码 |
| errmsg | | | string | 返回信息描述 |
| data | rows | lon | string | 中心点经度 |
| lat | string | 中心点纬度 |
| name | string | 名称 |
| address | string | 地址 |
| adcode | string | 所在行政区划码 |
| geotype | int | 地址编码匹配类型 0 国家，仅支持中国 1 省、直辖市 2 城市、地级市 3 区、县 4 村镇、商圈 5 小区、大厦 6 公交站 7 道路 8 门牌 9 POI 10 门址插值 11 附近的门牌 |
| geolevel | int | 地址编码精度等级 0-0~25m 1-26~75m 2-76~125m 3-126~250m 4-251-500m 5-501-1000m 6-1001-2000m 9- >2000m |

### 根据地址查找地址描述信息

**接口方法**：

**minemap**.service. queryReverseGeoCodingResult (location, type, radius, roadRadius, kind, callback)

**使用示例**：

**minemap**.service. queryReverseGeoCodingResult (**'** **116.37959,39.86541'**, 1,1000,1000,0, **function** (error,results) {  
 *//console.log(results);*});

**参数说明：**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| location | string | 坐标点 x,y |
| type | number | 返回结果数据类型 默认1 参数说明：1-apt,poi data 门址或poi数据、2-nearest link data 最近link数据 |
| radius | number | 搜索周边POI/APT的半径 默认 300米 |
| roadRadius | number | 搜索周边道路的半径 默认 300米 |
| kind | number | 最近道路是否考虑路名 参数说明：0-最近的道路不考虑是否有路名、1-最近的有名称的道路 |
| callback | Function | 回调函数 返回查询结果 示例function(error,result){} |

**返回数据格式：**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | | 类型 | 说明 |
| errcode | | int | 返回码 |
| errmsg | | string | 返回信息描述 |
| data | Id | string | POI ID |
| adcode | string | 行政区划码 |
| roadname | string | 最近道路名 |
| restName | string | 名称 |
| type | string | apt,poi,cross,link |
| distance | string | 中心点与结果中点之间的距离 |

## 内容补全服务

### 根据输入内容查找智能提示信息

**接口方法**：

**minemap**.service. querySmartTipsResult (adminCode, input, searchType, pageCount, callback)

**使用示例**：

**minemap**.service. querySmartTipsResult (**'** **110000'**, **'肯德'**, **'all'** , 10,**function** (error,results) {  
 *//console.log(results);*});

**参数说明：**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| adminCode | string | 行政区划编码 |
| input | string | 输入字符串 |
| searchType | string | 查找数据类型 默认all 参数说明：（all|area|busline|busstation|cross|poi|road） |
| pageCount | number | 返回的记录数 默认为10 |
| callback | Function | 回调函数 返回查询结果 示例function(error,result){} |

**返回数据格式：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | | | 类型 | 说明 |
| errcode | | | int | 返回码 |
| errmsg | | | string | 返回信息描述 |
| data | rows | id | string | 返回信息唯一标识 |
| adcode | string | 行政区划代码 |
| name | string | 名称 poi link cross |
| area | string | 行政区划名称 |

# 常见问题

（1）地图数据无法加载？

请检查key是否获取，然后检查对应的solutionId是否正确，如果二者均正确，可能因为配图过程中某些属性配置错误。

（2）粒子图层粒子不动？

需要在地图实例化后加入以下语句：

map.**repaint** = **true**;