# GoogleAdapter

## 1产品概述

GoogleAdapter 是一款基于SuperMap iClient 6R(2012) for JavaScript 和第三方地图JavaScript开发的适配器工具，此处面对的是以Google JavaScript为基础，并且又想加入SuperMap iServer强大的功能的用户。

Google JavaScript API的简单性得到了广大普通用户的青睐，尤其是一些国外用户，但是缺少SuperMap iServer的核心GIS功能，GoogleAdapter就是以简单的方法去衔接SuperMap iServer的核心GIS功能，将其和Google JavaScript相结合，达到兼具强大的GIS功能以及易用性。

GoogleAdapter主要提供了地图叠加以及Geometry的转换，可以实现在Google地图上面叠加SuperMap iServer中合理的地图、专题图的叠加、查询功能以及分析功能。

## 2快速入门

### 2.1申请SuperMap许可

既然我们需要用到SuperMap的功能，当然需要申请SuperMap的许可，请在SupeMap的主页<http://www.supermap.com.cn/>的下载区<http://support.supermap.com.cn/ProductCenter/DownloadCenter/ProductPlatform.aspx>下载许可配置工具以及申请许可。

### 2.2启动服务器

许可安装完毕后我们还的在下载区下载服务器产品SuperMap iServer Java 6R(2012)，并且安装，然后启动服务器，服务器会自带一些基础的服务，我们的示例都是以这些服务为基础的。

### 2.3创建项目

我们的项目很简单，就是普通的html，推荐大家使用WebStorm来开发，比较快捷。

首先你需要在SuperMap iServer Java 6R(2012)的安装文件iClient\forJavaScript\libs下拷贝一份SuperMap.js文件到你的文件夹下，此js文件是SuperMap iClient 6R(2012) for JavaScript的产品包，详细信息可以在你本地<http://localhost:8090/iserver/iClient/forJavaScript/index.html>下了解或官网<http://support.supermap.com.cn:8090/iserver/iClient/forJavaScript/index.html>下，其次我们还需要适配器文件，也就是我们的GoogleAdapter.js文件。

首先我们创建一个html页面，需要添加三个引用：<script type="text/javascript" src=" https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false "></script>此为引用Google地图的JavaScript API；<script type="text/javascript" src="SuperMap.js"></script>此为引用的SuperMap.js文件；<script type="text/javascript" src=" GoogleAdapter.js"></script>此为我们的适配器。注意适配器是基于前两个扩展的，所以要最后引用。

用过Google地图API的用户都知道我们需要一个div来承载地图，所以在body里面创建一个<div id=" map-canvas" style="width: 1000px;height: 600px"></div>，注意需要赋予id。Google地图支持900913投影，所以我们叠加的图层都必须是900913的

添加核心的脚本：

<script>

var url = "http://localhost:8090/iserver/services/map-china400/rest/maps/China";

function initialize() {

var myLatlng = new google.maps.LatLng(0, 0);

var mapOptions = {

center: myLatlng,

zoom: 1,

streetViewControl: true,

mapTypeId: google.maps.MapTypeId.ROADMAP

};

var map = new google.maps.Map(document.getElementById('map-canvas'),

mapOptions);

var lat = new google.maps.LatLng(41.850033, 116);

map.overlayMapTypes.insertAt(

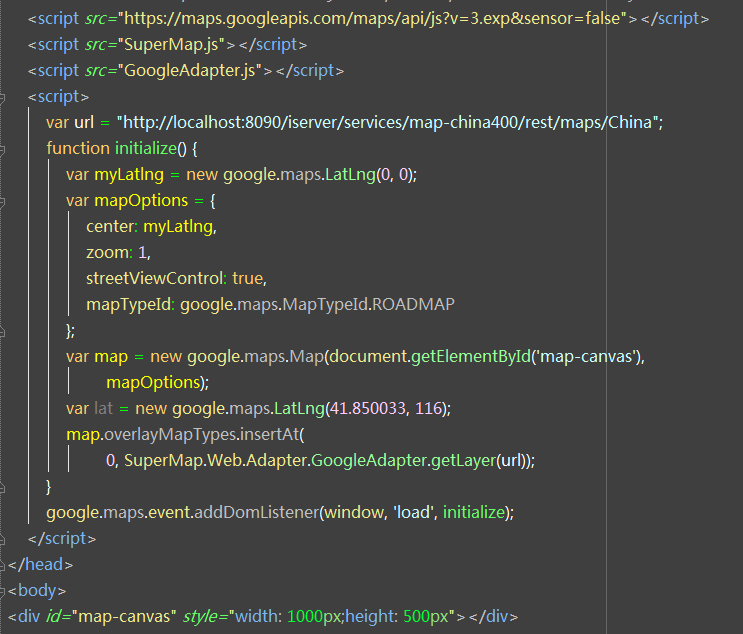
0, SuperMap.Web.Adapter.GoogleAdapter.getLayer(url));

}

google.maps.event.addDomListener(window, 'load', initialize);

</script>

如下为完整的代码截图：



然后运行网页，可以看到china地图和Google地图叠加的效果（3857）：



地图叠加效果非常不错，几乎没有偏差。

## 3API

整个GoogleAdapter所有接口都以静态方法开设，共开设了如下几个接口：

### 3.1地图叠加

地图叠加接口为SuperMap.Web.Adapter.GoogleAdapter.getLayer(url,options);此方法第一个参数url为其他地图服务的基础地址，必设参数，此地图只能为为3857投影系；第二个参数options为可选参数，options为一个object，如：options={transparent:true; cacheEnabled:false}，里面的参数包括：

1. transparent：{Boolean}设置每一张图片是否背景透明，默认为true
2. cacheEnabled：{Boolean}设置访问服务器图片的时候是否使用缓存，默认为false
3. layersID：{String}当需要将服务器出的专题图叠加到Google地图上时需要将专题的id号设置到这个属性上，这样在访问图片的时候才能正确的找到专题图的图片

此接口最终返回一个Google的图层扩展类型的对象，此对象可以通过Google地图API的方法map.overlayMapTypes.insertAt (index,layer);添加到Google地图上。

### 3.2Geometry转换

几何图形的转换现在只支持三种：点、折线和多边形。由于地图的特殊性，几何图形其实最终都是分解为点后再转换成Google地图的坐标，再生成Google地图的几何对象，所以理论上我们支持大部分常用投影系的几何图形的转换。

#### 3.2.1点

点的转换接口为SuperMap.Web.Adapter.GoogleAdapter.transferPoint (array,projection);需要将点坐标转换成为4326的标准店再生成Google的点（其实Google的点做了一点偏移，误差在500米左右，直接使用Google的纠偏数据将不存在偏差，而使用标准的常用投影系下的数据都将有偏差）。

此方法的第一个参数array为点的数组，不限制数量，array里面支持的点的形式有四种：

1. 自定义对象，必须有x,y，如：

var points = [

{x:116.1,y:38.9},

{x:114.1,y:34.1}

];

1. SuperMap.Geometry.Point对象，如：

var points = [

new SuperMap.Geometry.Point(116.1,38.9),

new SuperMap.Geometry.Point(116.1,38.9)

];

1. SuperMap.LonLat对象，如：

var points = [

new SuperMap.LonLat(116.1,38.9),

new SuperMap.LonLat(116.1,38.4)

];

1. google.maps.LatLng对象，如：

var points = [

new google.maps.LatLng (39.9,116.38),

new google.maps.LatLng (39.9,116.35)

];

第二个参数projection为SuperMap.Projection对象。代表第一个参数里面的点是什么坐标系下的，这样内部才方便进行转换。

最后返回google.maps.LatLng对象的数组。

#### 3.2.2折线

折线的转换接口为SuperMap.Web.Adapter.GoogleAdapter.transferLine (array,projection);此方法同样支持任意多条折线的转换。

第一个参数array也是数组，不过是折线数组，里面的折线支持两种形式：

1. SuperMap.Geometry.LineString的对象，如：

var lines = [new SuperMap.Geometry.LineString(

new SuperMap.Geometry.Point(116.1,38.9),

new SuperMap.Geometry.Point(116.1,38.9)

)];

1. google.maps.Polyline的对象，如：

var lines = [new google.maps.Polyline (

{path:[

new google.maps.LatLng(39.9,116.38),

new google.maps.LatLng(39.8,116.38)

]})];

第二个参数projection为SuperMap.Projection对象。代表第一个参数里面的线是什么坐标系下的。最后返回google.maps.Polyline对象的数组

#### 3.2.3多边形

多边形的转换接口为SuperMap.Web.Adapter.GoogleAdapter.transferPolygon (array,projection);多边形的转换最终也是分解为点转换完成后再组合成为多边形。

第一个参数array也是数组，这里是多边形数组，里面的多边形支持两种形式：

1. SuperMap.Geometry.Polygon的对象，如：

var polygons = [new SuperMap.Geometry.Polygon(

[new SuperMap.Geometry.LinearRing(

new SuperMap.Geometry.Point(116.37,39.90),

new SuperMap.Geometry.Point(116.38,39.90),

new SuperMap.Geometry.Point(116.38,39.89),

new SuperMap.Geometry.Point(116.37,39.89)

)

]

)];

1. google.maps.Polygon的对象，如：

var polygons = [new google.maps.Polygon (

{path:[

new google.maps.LatLng(39.90762965106183,116.3786889372559),

new google.maps.LatLng(39.90795884517671,116.38632786853032),

new google.maps.LatLng(39.897432133833574,116.38534009082035),

new google.maps.LatLng(39.89789300648029,116.37624058825688)

]})];

第二个参数projection为SuperMap.Projection对象。代表第一个参数里面的多边形是什么坐标系下的。最后返回google.maps.Polygon对象的数组。

#### 3.2.4数据纠偏

接口名称为SuperMap.Web.Adapter. GoogleAdapter.transfer(lng,lat)。

由于用户的底图和数据都存在标准和偏移的情况，当用户的底图和数据都是标准或者偏移的，那不需要实现此方法，如果不一致需要用户实现两者之间的转换。

当用户需要纠偏时，则需要覆盖此方法，内部每次转换前会调用此方法，将待转换的经度坐标和纬度坐标传进来，通过用户的方式实现纠偏后返回。

第一个参数lng为Number类型，表示需要纠偏的经度；第一个参数lat为Number类型，表示需要纠偏的纬度；返回值为Object类型，按照形如:

{

lng:116.3,

lat:39.5

}

的格式进行返回。

比如用户的地图为中国范围的Google地图（是做了偏移的），如果没有买Google的纠偏数据，用自己的真实数据叠加上去就会出现位置错误，此时就需要实现此方法，将每一个坐标进行纠偏。