# SMAdapter

## 1产品概述

SuperMap iClient 6R (2012) for SMAdapter 是一款基于SuperMap iClient 6R(2012) for JavaScript 和第三方地图JavaScript开发的适配器，此处面对的是以百度地图JavaScript为基础，并且又想加入SuperMap iServer强大的功能的用户。

百度地图JavaScript API的简单性得到了广大普通用户的青睐，但是缺少SuperMap iServer的核心GIS功能，SuperMap iClient 6R (2012) for SMAdapter就是以简单的方法去衔接SuperMap iServer的核心GIS功能，将其和百度地图JavaScript相结合，达到兼具强大的GIS功能以及易用性。

SuperMap iClient 6R (2012) for SMAdapter主要提供了地图叠加以及Geometry的转换，可以实现在百度地图上面叠加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申请百度密匙

既然是使用百度地图进行开发的用户必定需要申请一个自己的密匙，按照如下图方式：

在<http://developer.baidu.com/map/jshome.htm>的概述下的获取密匙里，需要申请自己的账号。申请成功的密匙为一个字符串，如：“95a21d20cfe4df8d03f267282e62d8ae”



申请了百度密匙就可以使用百度地图的API来进行开发了。

### 2.2申请SuperMap许可

既然我们需要用到SuperMap的功能，当然还的申请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。

### 2.3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。

### 2.4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SMAdapter.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="http://api.map.baidu.com/api?v=1.5&ak=95a21d20cfe4df8d03f267282e62d8ae"></script>此为引用百度的JavaScript API，并且配置了我们之前申请的密匙；<script type="text/javascript" src="SuperMap.js"></script>此为引用的SuperMap.js文件；<script type="text/javascript" src="SMAdapter.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。

用过百度地图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="allmap"></div>，注意需要赋予id，我们再为此配置一个css

<style type="text/css">

body, html,#allmap {width: 100%;height: 100%;overflow: hidden;margin:0;}

</style>

注意这里的css通过id=allmap来识别的。

添加核心的脚本：

<script type="text/javascript">

function init(){

//创建百度地图

var map = new BMap.Map('allmap');

//设置可以使用滚轮控制地图放大缩小

map.enableScrollWheelZoom(true);

//定位

map.centerAndZoom(new BMap.Point(0, 0), 1);

//通过适配器接口初始化一个可以叠加到百度地图上的图层，

//叠加图层的投影系必须是3857的

var layer = SMAdapter.getBaiduLayer(

"http://localhost:8090/iserver/services/map-china400/rest/maps/China"

);

//将地图添加到百度地图上

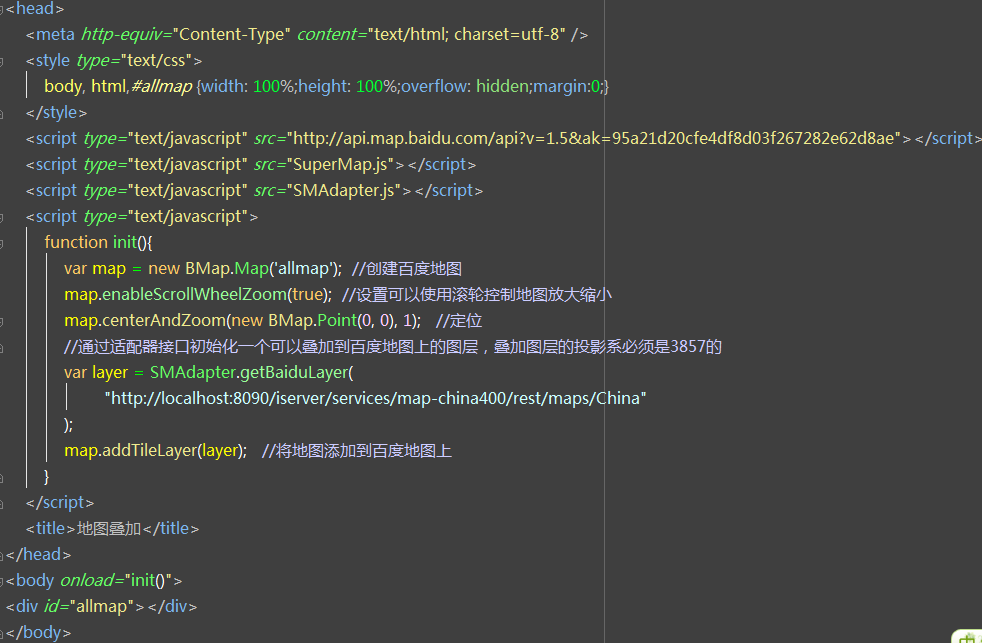
map.addTileLayer(layer);

}

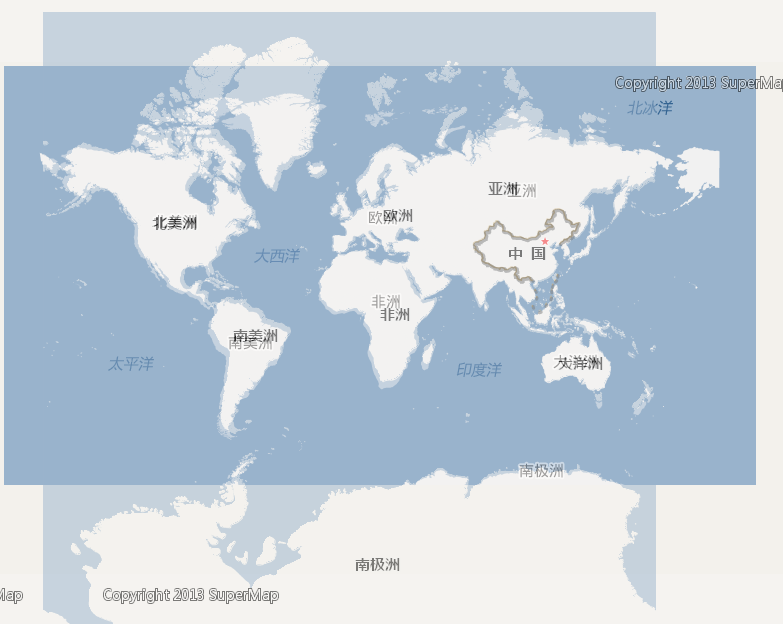
</script>

这一段脚本让其在body的onload事件里面执行。

如下为完整的代码截图：



然后运行网页，可以看到china地图和百度叠加的效果：



由于百度的地图投影系非标准，进行了二次加密，所以和标准3857的地图叠加后会出现不能完全重合的情况。

## 3API

整个SMAdapter核心只有一个类SMAdapter，SMAdapter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SMAdapter.getBaiduLayer(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，此地图必须为3857投影系才可以和百度地图进行叠加；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百度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BMap.TileLayer类型的对象，此对象可以通过百度API的方法map.addTileLayer(layer);添加到百度地图上。

### 3.2投影系转换

投影转换接口为SMAdapter.transferProjection(point, source, dest);此方法用户常见的投影系之间的转换，默认支持 EPSG:4326, CRS:84, urn:ogc:def:crs:EPSG:6.6:4326, EPSG:900913, EPSG:3857,EPSG:102113, EPSG:102100 投影间的转换。此方法第一个参数是待转换的点，可以是任意带有x,y坐标的对象，如SuperMap.Geometry.Point对象或者自定义的{x:116;y39}等；第二个参数为原坐标系统，必须是SuperMap.Projection的对象；第三个参数是目标坐标系统，也必须是SuperMap.Projection的对象。如我想将一个4326的{x:116;y:39}转换为3857下的坐标可以如下实现：

var point = SMAdapter.transferProjection(

{x:116;y:39},

new SuperMap.Projection(“EPSG:4326”),

new SuperMap.Projection(“EPSG:3857”)

);

### 3.3Geometry转换

几何图形的转换现在只支持三种：点、折线和多边形。由于百度地图的特殊性，几何图形其实最终都是分解为点后再转换成百度的坐标，再生成百度的几何对象，所以理论上我们支持3.2中的所有投影系的几何图形的转换。

#### 3.3.1点

点的转换接口为SMAdapter.transferPointToBaidu(array,projection,callback);由于百度的点坐标非标准坐标，所以进行点坐标转换的时候内部首先转换为GPS坐标，然后再通过百度服务器的接口进行转换，这样就需要向服务器发送请求，百度坐标转换的接口一次性最多支持20个点，所以用户需要拍注意当你转换的点过多时，内部是分20个点一次请求来反复请求服务器，效率上就会比较慢。

此方法的第一个参数array为点的数组，不限制数量，不过点数越多，效率越低，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. BMap.Point对象，如：

var points = [

new BMap.Point(116.38,39.9),

new BMap.Point(116.38,39.9)

];

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。第三个参数callback是一个回调函数，由于坐标转换最终需要向服务器发送请求，所有需要有一个回调函数来接收转换后的点，回调函数有一个参数，就是一个BMap.Point对象的数组。

#### 3.3.2折线

折线的转换接口为SMAdapter.transferLineToBaidu (array,projection,callback);线的转换其实是分解为点的转换，同样会在数据太多会出现效率低的问题，转换完成后再次组合成为线，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. BMap.Polyline的对象，如：

var lines = [new BMap.Polyline(

new BMap.Point(116.38,39.9),

new BMap.Point(116.38,39.9)

)];

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。第三个参数callback同样是回调函数，只是参数为BMap.Polyline对象的数组。

#### 3.3.3多边形

多边形的转换接口为SMAdapter.transferPolygonToBaidu (array,projection,callback);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，数据量太大一样也会出现效率低的问题。

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. BMap.Polygon的对象，如：

var polygons = [new BMap.Polygon(

new BMap.Point(116.37,39.90),

new BMap.Point(116.38,39.90),

new BMap.Point(116.38,39.89),

new BMap.Point(116.37,39.89)

)];

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。第三个参数callback同样是回调函数，只是参数为BMap. Polygon对象的数组。