**什么是百度地图Android SDK?**

百度地图 Android SDK是一套基于Android 1.5及以上版本设备的应用程序接口，不仅提供构建地图的基本接口，还提供本地搜索、路线规划、定位等服务。

您可以使用百度地图Android SDK开发适用于移动设备的地图应用，通过接口，您可以轻松访问百度服务和数据，构建功能丰富、交互性强的地图应用程序。

**面向的读者**

API是提供给那些具有一定Android编程经验和了解面向对象概念的读者使用。此外，读者还应该对地图产品有一定的了解。

您在使用中遇到任何问题，都可以通过[API贴吧](http://tieba.baidu.com/f?kw=%B0%D9%B6%C8%B5%D8%CD%BCapi)或[交流群](http://developer.baidu.com/map/jschat.htm)反馈给我们。

**获取API Key**

用户在使用API之前需要获取百度地图移动版API Key，该Key与你的百度账户相关联，您必须先有百度帐户，才能获得API KEY。并且，该KEY与您引用API的程序名称有关，具体流程请参照[获取密钥](http://api.map.baidu.com/lbsapi/cloud/android-mobile-apply-key.htm)。请妥善保存Key，地图初始化时需要用到Key。

# Hello World

**如何把API 添加到我的Android工程中？**

首先在工程里新建libs文件夹，并API开发包里的baidumapapi.jar拷贝到libs根目录下，将libBMapApiEngine.so拷贝到libs\armeabi目录下。官网demo里已有这两个文件，如果要集成到自己的工程里，就需要自己添加。然后在工程属性->Java Build Path->Libraries中选择“Add External JARs”，选定baidumapapi.jar，确定后返回，这样您就可以在您的程序中使用API了。

**在我的程序中显示地图**

在Manifest中添加使用权限

**<uses-permission** android:name="android.permission.ACCESS\_NETWORK\_STATE"**></uses-permission>**

**<uses-permission** android:name="android.permission.ACCESS\_FINE\_LOCATION"**></uses-permission>**

**<uses-permission** android:name="android.permission.INTERNET"**></uses-permission>**

**<uses-permission** android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"**></uses-permission>**

**<uses-permission** android:name="android.permission.ACCESS\_WIFI\_STATE"**></uses-permission>**

**<uses-permission** android:name="android.permission.CHANGE\_WIFI\_STATE"**></uses-permission>**

**<uses-permission** android:name="android.permission.READ\_PHONE\_STATE"**></uses-permission>**

在Manifest中添加Android版本支持

**<supports-screens** android:largeScreens="true"

android:normalScreens="true" android:smallScreens="true"

android:resizeable="true" android:anyDensity="true"**/>**

**<uses-sdk** android:minSdkVersion="3"**></uses-sdk>**

 让创建的地图Activity继承com.baidu.mapapi.MapActivity, 并import相关类

**import** java.util.ArrayList;

**import** java.util.List;

**import** android.content.Context;

**import** android.graphics.Canvas;

**import** android.graphics.Paint;

**import** android.graphics.Point;

**import** android.graphics.drawable.Drawable;

**import** android.location.Location;

**import** android.os.Bundle;

**import** android.util.Log;

**import** android.view.View;

**import** android.widget.Toast;

**import** com.baidu.mapapi.BMapManager;

**import** com.baidu.mapapi.GeoPoint;

**import** com.baidu.mapapi.ItemizedOverlay;

**import** com.baidu.mapapi.LocationListener;

**import** com.baidu.mapapi.MKAddrInfo;

**import** com.baidu.mapapi.MKDrivingRouteResult;

**import** com.baidu.mapapi.MKGeneralListener;

**import** com.baidu.mapapi.MKLocationManager;

**import** com.baidu.mapapi.MKPlanNode;

**import** com.baidu.mapapi.MKPoiResult;

**import** com.baidu.mapapi.MKSearch;

**import** com.baidu.mapapi.MKSearchListener;

**import** com.baidu.mapapi.MKTransitRouteResult;

**import** com.baidu.mapapi.MKWalkingRouteResult;

**import** com.baidu.mapapi.MKSuggestionResult;

**import** com.baidu.mapapi.MapActivity;

**import** com.baidu.mapapi.MapController;

**import** com.baidu.mapapi.MapView;

**import** com.baidu.mapapi.MyLocationOverlay;

**import** com.baidu.mapapi.Overlay;

**import** com.baidu.mapapi.OverlayItem;

**import** com.baidu.mapapi.PoiOverlay;

**import** com.baidu.mapapi.RouteOverlay;

**import** com.baidu.mapapi.TransitOverlay;

**public** **class** MyMapActivity **extends** MapActivity {

@Override

**public** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.main);

}

@Override

**protected** **boolean** isRouteDisplayed() {

**return** **false**;

}

}

在布局xml中添加地图控件

**<?xml** version="1.0" encoding="utf-8"**?>**

**<LinearLayout** xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical" android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"**>**

**<TextView** android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content" android:text="@string/hello" **/>**

**<com.baidu.mapapi.MapView** android:id="@+id/bmapsView"

android:layout\_width="fill\_parent" android:layout\_height="fill\_parent"

android:clickable="true" **/>**

**</LinearLayout>**

初始化地图Activity/使用key

在地图Activity中定义变量： BMapManager mBMapMan = null; 在onCreate方法中增加以下代码，并将您申请的Key替换“我的Key”：

mBMapMan = **new** BMapManager(getApplication());

mBMapMan.init("我的Key", **null**);

**super**.initMapActivity(mBMapMan);

MapView mMapView = (MapView) findViewById(R.id.bmapsView);

mMapView.setBuiltInZoomControls(**true**); *//设置启用内置的缩放控件*

MapController mMapController = mMapView.getController(); *// 得到mMapView的控制权,可以用它控制和驱动平移和缩放*

GeoPoint point = **new** GeoPoint((**int**) (39.915 \* 1E6),

(**int**) (116.404 \* 1E6)); *//用给定的经纬度构造一个GeoPoint，单位是微度 (度 \* 1E6)*

mMapController.setCenter(point); *//设置地图中心点*

mMapController.setZoom(12); *//设置地图zoom级别*

Override以下方法,管理API:

@Override

**protected** **void** onDestroy() {

**if** (mBMapMan != **null**) {

mBMapMan.destroy();

mBMapMan = **null**;

}

**super**.onDestroy();

}

@Override

**protected** **void** onPause() {

**if** (mBMapMan != **null**) {

mBMapMan.stop();

}

**super**.onPause();

}

@Override

**protected** **void** onResume() {

**if** (mBMapMan != **null**) {

mBMapMan.start();

}

**super**.onResume();

}

完成上述步骤后，运行程序，结果如下： 

# 地图图层

## 地图图层概念

地图可以包含一个或多个图层，每个图层在每个级别都是由若干张图块组成的，它们覆盖了地球的整个表面。例如您所看到包括街道、兴趣点、学校、公园等内容的地图展现就是一个图层，另外交通流量的展现也是通过图层来实现的。

## 底图

基本的地图图层，包括若干个缩放级别，显示基本的地图信息，包括道路、街道、学校、公园等内容。

## 实时交通信息

当前，全国范围内已支持多个城市实时路况查询，且会陆续开通其他城市。已开通城市列表可[查看](http://developer.baidu.com/map/question.htm#qa0033)。

在地图中显示实时交通信息示例如下：

mMapView.setTraffic(**true**);

运行程序，结果如下：

## 卫星图

mMapView.setSatellite(**true**);

## 实景图

在此版本API中暂不支持。

mMapView.setStreetView(**true**);

# 覆盖物

**地图覆盖物概述**

所有叠加或覆盖到地图的内容，我们统称为地图覆盖物。如标注、矢量图形元素(包括：折线和多边形和圆)、定位图标等。覆盖物拥有自己的地理坐标，当您拖动或缩放地图时，它们会相应的移动。

地图API提供了如下几种覆盖物：

* Overlay：覆盖物的抽象基类，所有的覆盖物均继承此类的方法，实现用户自定义图层显示。
* MyLocationOverlay：一个负责显示用户当前位置的Overlay。
* ItemizedOverlay<Item extends OverlayItem>：Overlay的一个基类，包含了一个OverlayItem列表，相当于一组分条的Overlay，通过继承此类，将一组兴趣点显示在地图上。
* PoiOverlay：本地搜索图层，提供某一特定地区的位置搜索服务，比如在北京市搜索“公园”，通过此图层将公园显示在地图上。
* RouteOverlay：步行、驾车导航线路图层，将步行、驾车出行方案的路线及关键点显示在地图上。
* TransitOverlay：公交换乘线路图层，将某一特定地区的公交出行方案的路线及换乘位置显示在地图上。

**覆盖物的抽象基类：Overlay**

一般来说，在MapView中添加一个Overlay需要经过以下步骤：

自定义类继承Overlay，并Override其draw()方法，如果需要点击、按键、触摸等交互操作，还需Override onTap()等方法。

**public** **class** MyOverlay **extends** Overlay {

GeoPoint geoPoint = **new** GeoPoint((**int**) (39.915 \* 1E6), (**int**) (116.404 \* 1E6));

[Paint](http://www.google.com/search?hl=en&q=allinurl%3Apaint+java.sun.com&btnI=I%27m%20Feeling%20Lucky) paint = **new** [Paint](http://www.google.com/search?hl=en&q=allinurl%3Apaint+java.sun.com&btnI=I%27m%20Feeling%20Lucky)();

@Override

**public** **void** draw([Canvas](http://www.google.com/search?hl=en&q=allinurl%3Acanvas+java.sun.com&btnI=I%27m%20Feeling%20Lucky) canvas, MapView mapView, **boolean** shadow) {

*//在天安门的位置绘制一个String*

[Point](http://www.google.com/search?hl=en&q=allinurl%3Apoint+java.sun.com&btnI=I%27m%20Feeling%20Lucky) point = mMapView.getProjection().toPixels(geoPoint, **null**);

canvas.drawText("★这里是天安门", point.x, point.y, paint);

}

}

添加到MapView的覆盖物中：

mMapView.getOverlays().add(**new** MyOverlay());

运行结果如下：app_myoverlay.png

**当前位置：MyLocationOverlay**

将MyLocationOverlay添加到覆盖物中，能够实现在地图上显示当前位置的图标以及指南针：

初始化Location模块

*// 初始化Location模块*

mLocationManager = mBMapMan.getLocationManager();

*// 通过enableProvider和disableProvider方法，选择定位的Provider*

*// mLocationManager.enableProvider(MKLocationManager.MK\_NETWORK\_PROVIDER);*

*// mLocationManager.disableProvider(MKLocationManager.MK\_GPS\_PROVIDER);*

*// 添加定位图层*

MyLocationOverlay mylocTest = **new** MyLocationOverlay(**this**, mMapView);

mylocTest.enableMyLocation(); *// 启用定位*

mylocTest.enableCompass(); *// 启用指南针*

mMapView.getOverlays().add(mylocTest);

运行结果如下：

**分条目覆盖物：ItemizedOverlay**

某个类型的覆盖物，包含多个类型相同、显示方式相同、处理方式相同的项时，使用此类：

自定义类继承ItemizedOverlay，并Override其draw()方法，如果需要点击、按键、触摸等交互操作，还需Override onTap()等方法。

**class** OverItemT **extends** ItemizedOverlay<OverlayItem> {

**private** List<OverlayItem> GeoList = **new** ArrayList<OverlayItem>();

**private** [Context](http://www.google.com/search?hl=en&q=allinurl%3Acontext+java.sun.com&btnI=I%27m%20Feeling%20Lucky) mContext;

**private** **double** mLat1 = 39.90923;*//39.9022; // point1纬度*

**private** **double** mLon1 = 116.397428;*//116.3822; // point1经度*

**private** **double** mLat2 = 39.9022;

**private** **double** mLon2 = 116.3922;

**private** **double** mLat3 = 39.917723;

**private** **double** mLon3 = 116.3722;

**public** OverItemT(Drawable marker, [Context](http://www.google.com/search?hl=en&q=allinurl%3Acontext+java.sun.com&btnI=I%27m%20Feeling%20Lucky) context) {

**super**(boundCenterBottom(marker));

**this**.mContext = context;

*// 用给定的经纬度构造GeoPoint，单位是微度 (度 \* 1E6)*

GeoPoint p1 = **new** GeoPoint((**int**) (mLat1 \* 1E6), (**int**) (mLon1 \* 1E6));

GeoPoint p2 = **new** GeoPoint((**int**) (mLat2 \* 1E6), (**int**) (mLon2 \* 1E6));

GeoPoint p3 = **new** GeoPoint((**int**) (mLat3 \* 1E6), (**int**) (mLon3 \* 1E6));

  GeoList.add(**new** OverlayItem(p1, "P1", "point1"));

GeoList.add(**new** OverlayItem(p2, "P2", "point2"));

GeoList.add(**new** OverlayItem(p3, "P3", "point3"));

populate(); *//createItem(int)方法构造item。一旦有了数据，在调用其它方法前，首先调用这个方法*

}

  @Override

**protected** OverlayItem createItem(**int** i) {

**return** GeoList.get(i);

}

  @Override

**public** **int** size() {

**return** GeoList.size();

}

  @Override

*// 处理当点击事件*

**protected** **boolean** onTap(**int** i) {

Toast.makeText(**this**.mContext, GeoList.get(i).getSnippet(),

Toast.LENGTH\_SHORT).show();

**return** **true**;

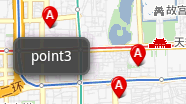
}

}

添加到MapView的覆盖物中：

Drawable marker = getResources().getDrawable(R.drawable.iconmark); *//得到需要标在地图上的资源*

mMapView.getOverlays().add(**new** OverItemT(marker, **this**)); *//添加ItemizedOverlay实例到mMapView*

点击其中一个图标，运行结果如下：

**本地搜索覆盖物：PoiOverlay**

详见在POI搜索及PoiOverlay。

**驾车路线覆盖物：RouteOverlay**

详见在驾车路线搜索及RouteOverlay和步行路线搜索及RouteOverlay。

**公交换乘路线覆盖物：TransitOverlay**

详见在公交换乘路线搜索及TransitOverlay。

**如何删除覆盖物？**

清除指定覆盖物： 参考demo里面的ItemizedOverlayDemo.java，重载protected OverlayItem createItem(int i) 接口，如果要移除覆盖物，就把自己维护的mGeoList列表删除。

清除所有覆盖物：mapView.getOverlays().clear();

# 服务类

**搜索服务**

百度地图移动版API集成搜索服务包括：位置检索、周边检索、范围检索、公交检索、驾乘检索、步行检索，通过初始化MKSearch类，注册搜索结果的监听对象MKSearchListener，实现异步搜索服务。首先自定义MySearchListener实现MKSearchListener接口，通过不同的回调方法，获得搜索结果：

**public** **class** MySearchListener **implements** MKSearchListener {

@Override

**public** **void** onGetAddrResult(MKAddrInfo result, **int** iError) {

}

@Override

**public** **void** onGetDrivingRouteResult(MKDrivingRouteResult result, **int** iError) {

}

@Override

**public** **void** onGetPoiResult(MKPoiResult result, **int** type, **int** iError) {

}

@Override

**public** **void** onGetTransitRouteResult(MKTransitRouteResult result, **int** iError) {

}

@Override

**public** **void** onGetWalkingRouteResult(MKWalkingRouteResult result, **int** iError) {

}

@Override

**public** **void** onGetBusDetailResult(MKBusLineResult result, **int** iError) {

}

@Override

**public** **void** onGetSuggestionResult(MKSuggestionResult result, **int** iError) {

}

}

然后初始化MKSearch类:

mMKSearch = **new** MKSearch();

mMKSearch.init(mBMapMan, **new** MySearchListener());*//注意，MKSearchListener只支持一个，以最后一次设置为准*

**POI搜索及PoiOverlay**

POI搜索有三种方式，根据范围和检索词发起范围检索poiSearchInbounds，城市poi检索poiSearchInCity，周边检索poiSearchNearBy，以下以周边检索为例介绍如何进行检索并显示覆盖物PoiOverlay：

检索天安门周边5000米之内的KFC餐厅：

mMKSearch.poiSearchNearBy("KFC", **new** GeoPoint((**int**) (39.915 \* 1E6), (**int**) (116.404 \* 1E6)), 5000);

实现MySearchListener的onGetPoiResult，并展示检索结果：

@Override

**public** **void** onGetPoiResult(MKPoiResult result, **int** type, **int** iError) {

**if** (result == **null**) {

**return**;

}

PoiOverlay poioverlay = **new** PoiOverlay(MyMapActivity.**this**, mMapView);

poioverlay.setData(result.getAllPoi());

mMapView.getOverlays().add(poioverlay);

mMapView.invalidate(); *//刷新地图*

}

运行结果如下：

**place详情页面使用方法**

当执行完poi检索后,我们会得到一个poi的列表,每个poi节点都有个uid属性,我们可以根据这个uid获取关于这个poi的一些更详细的信息.比如：评论、图片、商户描述等。

在AndroidManifest.xml中设置如下信息：

<activity android:name="com.baidu.mapapi.PlaceCaterActivity"  
 android:configChanges="orientation|keyboardHidden"  
 android:theme="@android:style/Theme.NoTitleBar"> </activity>

发起请求：

mSearch.poiDetailSearch(poi.uid);

详情页运行结果：

具体实现详见[相关下载](http://developer.baidu.com/map/sdkandev-download.htm)提供的Demo（PoiSearch.java）

**驾车路线搜索及RouteOverlay**

 检索从天安门到百度大厦的驾车路线：

MKPlanNode start = **new** MKPlanNode();

start.pt = **new** GeoPoint((**int**) (39.915 \* 1E6), (**int**) (116.404 \* 1E6));

MKPlanNode end = **new** MKPlanNode();

end.pt = **new** GeoPoint(40057031, 116307852);

*// 设置驾车路线搜索策略，时间优先、费用最少或距离最短*

mMKSearch.setDrivingPolicy(MKSearch.ECAR\_TIME\_FIRST);

mMKSearch.drivingSearch(**null**, start, **null**, end);

实现MySearchListener的onGetDrivingRouteResult，并展示检索结果：

@Override

**public** **void** onGetDrivingRouteResult(MKDrivingRouteResult result, **int** iError) {

**if** (result == **null**) {

**return**;

}

RouteOverlay routeOverlay = **new** RouteOverlay(MyMapActivity.**this**, mMapView);

*// 此处仅展示一个方案作为示例*

routeOverlay.setData(result.getPlan(0).getRoute(0));

mMapView.getOverlays().add(routeOverlay);

}

运行结果如下：

**步行路线搜索及RouteOverlay**

方式与驾车路线搜索类似，只需将mMKSearch.drivingSearch(null, start, null, end)修改为mMKSearch.walkingSearch(null, start, null, end),实现的方法改为onGetWalkingRouteResult即可，不再赘述。

**公交换乘路线搜索及TransitOverlay**

检索从天安门到百度大厦的公交换乘路线：

MKPlanNode start = **new** MKPlanNode();

start.pt = **new** GeoPoint((**int**) (39.915 \* 1E6), (**int**) (116.404 \* 1E6));

MKPlanNode end = **new** MKPlanNode();

end.pt = **new** GeoPoint(40057031, 116307852);

*// 设置乘车路线搜索策略，时间优先、最少换乘、最少步行距离或不含地铁*

mMKSearch.setTransitPolicy(MKSearch.EBUS\_TRANSFER\_FIRST);

mMKSearch.transitSearch("北京", start, end); *// 必须设置城市名*

实现MySearchListener的onGetTransitRouteResult(MKTransitRouteResult, int)，并展示检索结果：

@Override

**public** **void** onGetTransitRouteResult(MKTransitRouteResult result, **int** iError) {

**if** (result == **null**) {

**return**;

}

TransitOverlay transitOverlay = **new** TransitOverlay(MyMapActivity.**this**, mMapView);

*// 此处仅展示一个方案作为示例*

transitOverlay.setData(result.getPlan(0));

mMapView.getOverlays().add(transitOverlay);

mMapView.invalidate(); *//刷新地图*

}

**公交路线详情搜索**

检索北京市公交路线717的poi，获取公交路线的uid：

mSearch.poiSearchInCity("北京", "717");

实现MySearchListener的onGetPoiResult (MKPoiResult res, int type, int error)，获得公交线路poi的uid，并根据此uid发起公交线路详情的检索：

@Override

**public** **void** onGetPoiResult(MKPoiResult res, **int** type, **int** error) {

*// 错误号可参考MKEvent中的定义*

**if** (error != 0 || res == **null**) {

Toast.makeText(BusLineSearch.**this**, "抱歉，未找到结果", Toast.LENGTH\_LONG).show();

**return**;

}

*// 找到公交路线poi node*

MKPoiInfo curPoi = **null**;

**int** totalPoiNum = res.getNumPois();

**for**( **int** idx = 0; idx < totalPoiNum; idx++ ) {

curPoi = res.getPoi(idx);

**if** ( 2 == curPoi.ePoiType ) {

**break**;

}

}

mSearch.busLineSearch(mCityName, curPoi.uid);

}

实现MySearchListener的onGetBusDetailResult(MKBusLineResult result, int iError)，并展示搜索结果：

**public** **void** onGetBusDetailResult(MKBusLineResult result, **int** iError) {

**if** (iError != 0 || result == **null**) {

Toast.makeText(BusLineSearch.**this**, "抱歉，未找到结果", Toast.LENGTH\_LONG).show();

**return**;

}

RouteOverlay routeOverlay = **new** RouteOverlay(BusLineSearch.**this**, mMapView);

*// 此处仅展示一个方案作为示例*

routeOverlay.setData(result.getBusRoute());

mMapView.getOverlays().clear();

mMapView.getOverlays().add(routeOverlay);

mMapView.invalidate();

mMapView.getController().animateTo(result.getBusRoute().getStart());

}

运行结果如下：

**地址信息查询**

根据地理坐标查询地址信息：

mMKSearch.reverseGeocode(**new** GeoPoint(40057031, 116307852)); *//逆地址解析*

mMKSearch.geocode(key, city);*//地址解析*

geocode返回结果在MKSearchListener里的onGetAddrResult方法。

reverseGeocode返回结果在MKSearchListener里的onGetPoiResult方法。

**public** **void** onGetAddrResult(MKAddrInfo res, **int** error) {

**if** (error != 0) {

[String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky) str = [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky).format("错误号：%d", error);

Toast.makeText(GeoCoder.**this**, str, Toast.LENGTH\_LONG).show();

**return**;

}

mMapView.getController().animateTo(res.geoPt);

[String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky) strInfo = [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky).format("纬度：%f 经度：%f**\r\n**", res.geoPt.getLatitudeE6()/1e6,res.geoPt.getLongitudeE6()/1e6);

Toast.makeText(GeoCoder.**this**, strInfo, Toast.LENGTH\_LONG).show();

Drawable marker = getResources().getDrawable(R.drawable.iconmarka); *//得到需要标在地图上的资源*

marker.setBounds(0, 0, marker.getIntrinsicWidth(), marker.getIntrinsicHeight()); *//为maker定义位置和边界*

mMapView.getOverlays().clear();

mMapView.getOverlays().add(**new** OverItemT(marker, GeoCoder.**this**, res.geoPt, res.strAddr));

}

**public** **void** onGetPoiResult(MKPoiResult res, **int** type, **int** error) {

**if** (error != 0 || res == **null**) {

Toast.makeText(GeoCoder.**this**, "解析失败", Toast.LENGTH\_LONG).show();

**return**;

}

**if** (res != **null** && res.getCurrentNumPois() > 0) {

GeoPoint ptGeo = res.getAllPoi().get(0).pt;

*// 移动地图到该点：*

mMapView.getController().animateTo(ptGeo);

[String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky) strInfo = [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky).format("纬度：%f 经度：%f**\r\n**", ptGeo.getLatitudeE6()/1e6,ptGeo.getLongitudeE6()/1e6);

strInfo += "**\r\n**附近有：";

**for** (**int** i = 0; i < res.getAllPoi().size(); i++) {

strInfo += (res.getAllPoi().get(i).name + ";");

}

Toast.makeText(GeoCoder.**this**, strInfo, Toast.LENGTH\_LONG).show();

}

}

**在线建议查询**

根据关键词查询在线建议词：

mMKSearch.suggestionSearch("天安门"); *//输入关键词*

实现MySearchListener的onGetSuggestionResult，得到查询结果：

[ListView](http://www.google.com/search?hl=en&q=allinurl%3Alistview+java.sun.com&btnI=I%27m%20Feeling%20Lucky) mSuggestionList = ([ListView](http://www.google.com/search?hl=en&q=allinurl%3Alistview+java.sun.com&btnI=I%27m%20Feeling%20Lucky)) findViewById(R.id.listView1);

@Override

**public** **void** onGetSuggestionResult(MKSuggestionResult res, **int** iError)

{

**if** (iError!= 0 || res == **null**) {

Toast.makeText(PoiSearch.**this**, "抱歉，未找到结果", Toast.LENGTH\_LONG).show();

**return**;

}

**int** nSize = res.getSuggestionNum();

[String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky)[] mStrSuggestions = **new** [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky)[nSize];

**for** (**int** i = 0; i < nSize; i++)

{

mStrSuggestions[i] = res.getSuggestion(i).city + res.getSuggestion(i).key;

}

ArrayAdapter<String> suggestionString = **new** ArrayAdapter<String>(PoiSearch.**this**, android.R.layout.simple\_list\_item\_1,mStrSuggestions);

mSuggestionList.setAdapter(suggestionString);

}

**云检索模块的使用**

首先实例化GeoSearchListener接口：

@Override

**public** **void** onGetGeoDetailsResult(DetailResult result, **int** type, **int** iError)

{

**if** (result != **null**) {

**if** (result.content != **null**) {

Toast.makeText(CloudSearchDemo.**this**, result.content.name, Toast.LENGTH\_LONG).show();

}

**else** {

Toast.makeText(CloudSearchDemo.**this**, "status:" + result.status, Toast.LENGTH\_SHORT).show();

}

}

}

@Override

**public** **void** onGetGeoResult(GeoSearchResult result, **int** type, **int** iError)

{

**if** (result != **null** && result.poiList != **null** && result.poiList.size() > **0**) {

CloudOverlay poiOverlay = **new** CloudOverlay(**this**);

poiOverlay.setData(result.poiList);

mMapView.getOverlays().clear();

mMapView.getOverlays().add(poiOverlay);

mMapView.invalidate();

mMapView.getController().animateTo(new GeoPoint((int)(result.poiList.get(0).latitude \* 1e6), (int)(result.poiList.get(0).longitude \* 1e6)));

}

}

发起云检索请求：

BoundsSearchInfo r = **new** BoundsSearchInfo();

r.queryWords = "五中";

r.ak = "输入你的ak";

r.bounds = **new** Bounds(39843895, 116402214, 40956948, 116431457);

r.filter.put("databox", 848);

r.scope = 2;

GeoSearchManager.getInstance().searchBounds(r);

*// 当请求成功时，会回调onGetGeoResult函数，执行打点操作。*

CloudOverlay poiOverlay = **new** CloudOverlay(**this**);

poiOverlay.setData(result.poiList);

mMapView.getOverlays().clear();

mMapView.getOverlays().add(poiOverlay);

mMapView.invalidate();

运行结果如下：

# 事件

## 定位监听

实现方式与系统的定位监听类似，通过MKLocationManager注册或者移除定位监听器：

mLocationManager = mBMapMan.getLocationManager();

LocationListener listener = **new** LocationListener() {

@Override

**public** **void** onLocationChanged(Location location) {

*// TODO 在此处处理位置变化*

}

};

*// 注册监听*

mLocationManager.requestLocationUpdates(listener);

*// 不需要时移除监听*

mLocationManager.removeUpdates(listener);

## 一般事件监听

在初始化地图Activity时，注册一般事件监听，并实现MKGeneralListener的接口处理相应事件，将mBMapMan.init("我的Key", null)替换为下面的代码：

mBMapMan.init("我的key", **new** MKGeneralListener() {

@Override

**public** **void** onGetPermissionState(**int** iError) {

*// TODO 返回授权验证错误，通过错误代码判断原因，MKEvent中常量值。*

}

@Override

**public** **void** onGetNetworkState(**int** iError) {

*// TODO 返回网络错误，通过错误代码判断原因，MKEvent中常量值。*

}

});