**什么是百度地图API？**

百度地图移动版API（Android）是一套基于Android 1.5及以上设备的应用程序接口，通过该接口，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。百度地图移动版API不仅包含构建 地图的基本接口，还提供了诸如地图定位、本地搜索、路线规划等数据服务，你可以根据自己的需要进行选择。

**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贴吧或交流群反馈给我们。

**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://dev.baidu.com/wiki/static/imap/key/)。请妥善保存Key，地图初始化时需要用到Key。

**兼容性**

支持Android 1.5及以上系统。

**在您的程序中显示地图**

完整的Demo例程可参考 [相关下载](http://dev.baidu.com/wiki/imap/index.php?title=Android%E5%B9%B3%E5%8F%B0/%E7%9B%B8%E5%85%B3%E4%B8%8B%E8%BD%BD)。

**如何把API添加到我的Andoid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。

**百度地图的“Hello,World”**

* 在Manifest中添加使用权限
  1. <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE"></uses-permission>
  2. <uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION"></uses-permission>
  3. <uses-permission android:name="android.permission.INTERNET"></uses-permission>
  4. <uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"></uses-permission>
  5. <uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE"></uses-permission>
  6. <uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE"></uses-permission>
  7. <uses-permission android:name="android.permission.READ\_PHONE\_STATE"></uses-permission>
* 在Manifest中添加Android版本支持
  1. <supports-screens android:largeScreens="true"
  2. android:normalScreens="true" android:smallScreens="true"
  3. android:resizeable="true" android:anyDensity="true"/>
  4. <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
  1. import java.util.ArrayList;
  2. import java.util.List;
  4. import android.content.Context;
  5. import android.graphics.Canvas;
  6. import android.graphics.Paint;
  7. import android.graphics.Point;
  8. import android.graphics.drawable.Drawable;
  9. import android.location.Location;
  10. import android.os.Bundle;
  11. import android.util.Log;
  12. import android.view.View;
  13. import android.widget.Toast;
  15. import com.baidu.mapapi.BMapManager;
  16. import com.baidu.mapapi.GeoPoint;
  17. import com.baidu.mapapi.ItemizedOverlay;
  18. import com.baidu.mapapi.LocationListener;
  19. import com.baidu.mapapi.MKAddrInfo;
  20. import com.baidu.mapapi.MKDrivingRouteResult;
  21. import com.baidu.mapapi.MKGeneralListener;
  22. import com.baidu.mapapi.MKLocationManager;
  23. import com.baidu.mapapi.MKPlanNode;
  24. import com.baidu.mapapi.MKPoiResult;
  25. import com.baidu.mapapi.MKSearch;
  26. import com.baidu.mapapi.MKSearchListener;
  27. import com.baidu.mapapi.MKTransitRouteResult;
  28. import com.baidu.mapapi.MKWalkingRouteResult;
  29. import com.baidu.mapapi.MKSuggestionResult;
  30. import com.baidu.mapapi.MapActivity;
  31. import com.baidu.mapapi.MapController;
  32. import com.baidu.mapapi.MapView;
  33. import com.baidu.mapapi.MyLocationOverlay;
  34. import com.baidu.mapapi.Overlay;
  35. import com.baidu.mapapi.OverlayItem;
  36. import com.baidu.mapapi.PoiOverlay;
  37. import com.baidu.mapapi.RouteOverlay;
  38. import com.baidu.mapapi.TransitOverlay;
  40. public class MyMapActivity extends MapActivity {
  41. @Override
  42. public void onCreate(Bundle savedInstanceState) {
  43. super.onCreate(savedInstanceState);
  44. setContentView(R.layout.main);
  45. }
  47. @Override
  48. protected boolean isRouteDisplayed() {
  49. return false;
  50. }
  51. }
* 在布局xml中添加地图控件
  1. <?xml version="1.0" encoding="utf-8"?>
  2. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  3. android:orientation="vertical" android:layout\_width="fill\_parent"
  4. android:layout\_height="fill\_parent">
  5. <TextView android:layout\_width="fill\_parent"
  6. android:layout\_height="wrap\_content" android:text="@string/hello" />
  7. <com.baidu.mapapi.MapView android:id="@+id/bmapsView"
  8. android:layout\_width="fill\_parent" android:layout\_height="fill\_parent"
  9. android:clickable="true" />
  10. </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”：

* 1. mBMapMan = new BMapManager(getApplication());
  2. mBMapMan.init("我的Key", null);
  3. super.initMapActivity(mBMapMan);
  5. MapView mMapView = (MapView) findViewById(R.id.bmapsView);
  6. mMapView.setBuiltInZoomControls(true); //设置启用内置的缩放控件
  8. 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
  9. GeoPoint point = new GeoPoint((int) (39.915 \* 1E6),
  10. (int) (116.404 \* 1E6)); //用给定的经纬度构造一个GeoPoint，单位是微度 (度 \* 1E6)
  11. mMapController.setCenter(point); //设置地图中心点
  12. mMapController.setZoom(12); //设置地图zoom级别

Override以下方法,管理API:

* 1. @Override
  2. protected void onDestroy() {
  3. if (mBMapMan != null) {
  4. mBMapMan.destroy();
  5. mBMapMan = null;
  6. }
  7. super.onDestroy();
  8. }
  9. @Override
  10. protected void onPause() {
  11. if (mBMapMan != null) {
  12. mBMapMan.stop();
  13. }
  14. super.onPause();
  15. }
  16. @Override
  17. protected void onResume() {
  18. if (mBMapMan != null) {
  19. mBMapMan.start();
  20. }
  21. super.onResume();
  22. }

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**

在以下11个城市中，支持实时交通信息：北京，上海，广州，深圳，南京，南昌，成都，重庆，武汉，大连，常州。在地图中显示实时交通信息示例如下：

1. mMapView.setTraffic(true);

运行程序，结果如下： 

**卫星图**

1. mMapView.setSatellite(true);

**实景图**

在此版本API中暂不支持。

1. 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。
  1. public class MyOverlay extends Overlay {
  2. GeoPoint geoPoint = new GeoPoint((int) (39.915 \* 1E6), (int) (116.404 \* 1E6));
  3. [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)();
  5. @Override
  6. 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) {
  7. //在天安门的位置绘制一个String
  8. [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);
  9. canvas.drawText("★这里是天安门", point.x, point.y, paint);
  10. }
  11. }

添加到MapView的覆盖物中：

* 1. 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
  1. // 初始化Location模块
  2. mLocationManager = mBMapMan.getLocationManager();
  3. // 通过enableProvider和disableProvider方法，选择定位的Provider
  4. // mLocationManager.enableProvider(MKLocationManager.MK\_NETWORK\_PROVIDER);
  5. // mLocationManager.disableProvider(MKLocationManager.MK\_GPS\_PROVIDER);
  6. // 添加定位图层
  7. MyLocationOverlay mylocTest = new MyLocationOverlay(this, mMapView);
  8. mylocTest.enableMyLocation(); // 启用定位
  9. mylocTest.enableCompass(); // 启用指南针
  10. mMapView.getOverlays().add(mylocTest);

运行结果如下：

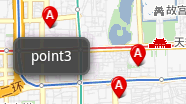
**分条目覆盖物：ItemizedOverlay**

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<OverlayItem>，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。
  1. class OverItemT extends ItemizedOverlay<OverlayItem> {
  2. private List<OverlayItem> GeoList = new ArrayList<OverlayItem>();
  3. private [Context](http://www.google.com/search?hl=en&q=allinurl%3Acontext+java.sun.com&btnI=I%27m%20Feeling%20Lucky) mContext;
  5. private double mLat1 = 39.90923;//39.9022; // point1纬度
  6. private double mLon1 = 116.397428;//116.3822; // point1经度
  8. private double mLat2 = 39.9022;
  9. private double mLon2 = 116.3922;
  11. private double mLat3 = 39.917723;
  12. private double mLon3 = 116.3722;
  14. public OverItemT(Drawable marker, [Context](http://www.google.com/search?hl=en&q=allinurl%3Acontext+java.sun.com&btnI=I%27m%20Feeling%20Lucky) context) {
  15. super(boundCenterBottom(marker));
  17. this.mContext = context;
  19. // 用给定的经纬度构造GeoPoint，单位是微度 (度 \* 1E6)
  20. GeoPoint p1 = new GeoPoint((int) (mLat1 \* 1E6), (int) (mLon1 \* 1E6));
  21. GeoPoint p2 = new GeoPoint((int) (mLat2 \* 1E6), (int) (mLon2 \* 1E6));
  22. GeoPoint p3 = new GeoPoint((int) (mLat3 \* 1E6), (int) (mLon3 \* 1E6));
  24. GeoList.add(new OverlayItem(p1, "P1", "point1"));
  25. GeoList.add(new OverlayItem(p2, "P2", "point2"));
  26. GeoList.add(new OverlayItem(p3, "P3", "point3"));
  27. 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
  28. }
  30. @Override
  31. protected OverlayItem createItem(int i) {
  32. return GeoList.get(i);
  33. }
  35. @Override
  36. public int size() {
  37. return GeoList.size();
  38. }
  40. @Override
  41. // 处理当点击事件
  42. protected boolean onTap(int i) {
  43. Toast.makeText(this.mContext, GeoList.get(i).getSnippet(),
  44. Toast.LENGTH\_SHORT).show();
  45. return true;
  46. }
  47. }

添加到MapView的覆盖物中：

* 1. Drawable marker = getResources().getDrawable(R.drawable.iconmark); //得到需要标在地图上的资源
  2. 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**

详见详见在[POI搜索及PoiOverlay](http://dev.baidu.com/wiki/imap/index.php?title=Android%E5%B9%B3%E5%8F%B0/%E5%BC%80%E5%8F%91%E6%8C%87%E5%8D%97#POI.E6.90.9C.E7.B4.A2.E5.8F.8APoiOverlay)。

**驾车路线覆盖物：RouteOverlay**

详见详见在[驾车路线搜索及RouteOverlay](http://dev.baidu.com/wiki/imap/index.php?title=Android%E5%B9%B3%E5%8F%B0/%E5%BC%80%E5%8F%91%E6%8C%87%E5%8D%97#.E9.A9.BE.E8.BD.A6.E8.B7.AF.E7.BA.BF.E6.90.9C.E7.B4.A2.E5.8F.8ARouteOverlay)和[步行路线搜索及RouteOverlay](http://dev.baidu.com/wiki/imap/index.php?title=Android%E5%B9%B3%E5%8F%B0/%E5%BC%80%E5%8F%91%E6%8C%87%E5%8D%97#.E6.AD.A5.E8.A1.8C.E8.B7.AF.E7.BA.BF.E6.90.9C.E7.B4.A2.E5.8F.8ARouteOverlay)。

**公交换乘路线覆盖物：TransitOverlay**

详见详见在[公交换乘路线搜索及TransitOverlay](http://dev.baidu.com/wiki/imap/index.php?title=Android%E5%B9%B3%E5%8F%B0/%E5%BC%80%E5%8F%91%E6%8C%87%E5%8D%97#.E5.85.AC.E4.BA.A4.E6.8D.A2.E4.B9.98.E8.B7.AF.E7.BA.BF.E6.90.9C.E7.B4.A2.E5.8F.8ATransitOverlay)。

**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：

1. public class MySearchListener implements MKSearchListener {
2. @Override
3. public void onGetAddrResult(MKAddrInfo result, int iError) {
4. }
6. @Override
7. public void onGetDrivingRouteResult(MKDrivingRouteResult result, int iError) {
8. }
10. @Override
11. public void onGetPoiResult(MKPoiResult result, int type, int iError) {
12. }
14. @Override
15. public void onGetTransitRouteResult(MKTransitRouteResult result, int iError) {
16. }
18. @Override
19. public void onGetWalkingRouteResult(MKWalkingRouteResult result, int iError) {
20. }
22. @Override
23. public void onGetBusDetailResult(MKBusLineResult result, int iError) {
24. }
26. @Override
27. public void onGetSuggestionResult(MKSuggestionResult result, int iError) {
28. }
29. }

然后初始化MKSearch类:

1. mMKSearch = new MKSearch();
2. 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：
  1. 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：
  1. @Override
  2. public void onGetPoiResult(MKPoiResult result, int type, int iError) {
  3. if (result == null) {
  4. return;
  5. }
  6. PoiOverlay poioverlay = new PoiOverlay(MyMapActivity.this, mMapView);
  7. poioverlay.setData(result.getAllPoi());
  8. mMapView.getOverlays().add(poioverlay);
  9. mMapView.invalidate(); //刷新地图
  10. }

运行结果如下：

**驾车路线搜索及RouteOverlay**

* 检索从天安门到百度大厦的驾车路线：
  1. MKPlanNode start = new MKPlanNode();
  2. start.pt = new GeoPoint((int) (39.915 \* 1E6), (int) (116.404 \* 1E6));
  3. MKPlanNode end = new MKPlanNode();
  4. end.pt = new GeoPoint(40057031, 116307852);
  5. // 设置驾车路线搜索策略，时间优先、费用最少或距离最短
  6. mMKSearch.setDrivingPolicy(MKSearch.ECAR\_TIME\_FIRST);
  7. mMKSearch.drivingSearch(null, start, null, end);
* 实现MySearchListener的onGetDrivingRouteResult，并展示检索结果：
  1. @Override
  2. public void onGetDrivingRouteResult(MKDrivingRouteResult result, int iError) {
  3. if (result == null) {
  4. return;
  5. }
  6. RouteOverlay routeOverlay = new RouteOverlay(MyMapActivity.this, mMapView);
  7. // 此处仅展示一个方案作为示例
  8. routeOverlay.setData(result.getPlan(0).getRoute(0));
  9. mMapView.getOverlays().add(routeOverlay);
  10. }

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：
  1. MKPlanNode start = new MKPlanNode();
  2. start.pt = new GeoPoint((int) (39.915 \* 1E6), (int) (116.404 \* 1E6));
  3. MKPlanNode end = new MKPlanNode();
  4. end.pt = new GeoPoint(40057031, 116307852);
  5. // 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
  6. mMKSearch.setTransitPolicy(MKSearch.EBUS\_TRANSFER\_FIRST);
  7. 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：
  1. @Override
  2. public void onGetTransitRouteResult(MKTransitRouteResult result, int iError) {
  3. if (result == null) {
  4. return;
  5. }
  6. TransitOverlay transitOverlay = new TransitOverlay(MyMapActivity.this, mMapView);
  7. // 此处仅展示一个方案作为示例
  8. transitOverlay.setData(result.getPlan(0));
  9. mMapView.getOverlays().add(transitOverlay);
  10. mMapView.invalidate(); //刷新地图
  11. }

**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：
  1. 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：
  1. @Override
  2. public void onGetPoiResult(MKPoiResult res, int type, int error) {
  3. // 错误号可参考MKEvent中的定义
  4. if (error != 0 || res == null) {
  5. Toast.makeText(BusLineSearch.this, "抱歉，未找到结果", Toast.LENGTH\_LONG).show();
  6. return;
  7. }
  9. // 找到公交路线poi node
  10. MKPoiInfo curPoi = null;
  11. int totalPoiNum = res.getNumPois();
  12. for( int idx = 0; idx < totalPoiNum; idx++ ) {
  13. curPoi = res.getPoi(idx);
  14. if ( 2 == curPoi.ePoiType ) {
  15. break;
  16. }
  17. }
  19. mSearch.busLineSearch(mCityName, curPoi.uid);
  20. }
* 实现MySearchListener的onGetBusDetailResult(MKBusLineResult result, int iError)，并展示搜索结果：
  1. public void onGetBusDetailResult(MKBusLineResult result, int iError) {
  2. if (iError != 0 || result == null) {
  3. Toast.makeText(BusLineSearch.this, "抱歉，未找到结果", Toast.LENGTH\_LONG).show();
  4. return;
  5. }
  7. RouteOverlay routeOverlay = new RouteOverlay(BusLineSearch.this, mMapView);
  8. // 此处仅展示一个方案作为示例
  9. routeOverlay.setData(result.getBusRoute());
  10. mMapView.getOverlays().clear();
  11. mMapView.getOverlays().add(routeOverlay);
  12. mMapView.invalidate();
  14. mMapView.getController().animateTo(result.getBusRoute().getStart());
  15. }

运行结果如下：

**地址信息查询**

根据地理坐标查询地址信息：

1. mMKSearch.reverseGeocode(new GeoPoint(40057031, 116307852)); //逆地址解析
2. 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onGetAddrResult方法。

1. public void onGetAddrResult(MKAddrInfo res, int error) {
2. if (error != 0) {
3. [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);
4. Toast.makeText(GeoCoder.this, str, Toast.LENGTH\_LONG).show();
5. return;
6. }
7. mMapView.getController().animateTo(res.geoPt);
8. [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);
9. Toast.makeText(GeoCoder.this, strInfo, Toast.LENGTH\_LONG).show();
10. Drawable marker = getResources().getDrawable(R.drawable.iconmarka); //得到需要标在地图上的资源
11. marker.setBounds(0, 0, marker.getIntrinsicWidth(), marker.getIntrinsicHeight()); //为maker定义位置和边界
12. mMapView.getOverlays().clear();
13. mMapView.getOverlays().add(new OverItemT(marker, GeoCoder.this, res.geoPt, res.strAddr));
14. }
15. public void onGetPoiResult(MKPoiResult res, int type, int error) {
16. if (error != 0 || res == null) {
17. Toast.makeText(GeoCoder.this, "解析失败", Toast.LENGTH\_LONG).show();
18. return;
19. }
20. if (res != null && res.getCurrentNumPois() > 0) {
21. GeoPoint ptGeo = res.getAllPoi().get(0).pt;
22. // 移动地图到该点：
23. mMapView.getController().animateTo(ptGeo);
24. [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);
25. strInfo += "\r\n附近有：";
26. for (int i = 0; i < res.getAllPoi().size(); i++) {
27. strInfo += (res.getAllPoi().get(i).name + ";");
28. }
29. Toast.makeText(GeoCoder.this, strInfo, Toast.LENGTH\_LONG).show();
30. }
31. }

**在线建议查询**

根据关键词查询在线建议词：

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

实现MySearchListener的onGetSuggestionResult，得到查询结果：

1. [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);
2. @Override
3. public void onGetSuggestionResult(MKSuggestionResult res, int iError)
4. {
5. if (iError!= 0 || res == null) {
6. Toast.makeText(PoiSearch.this, "抱歉，未找到结果", Toast.LENGTH\_LONG).show();
7. return;
8. }
9. int nSize = res.getSuggestionNum();
10. [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];
11. for (int i = 0; i < nSize; i++)
12. {
13. mStrSuggestions[i] = res.getSuggestion(i).city + res.getSuggestion(i).key;
14. }
15. ArrayAdapter<String> suggestionString = new ArrayAdapter<String>(PoiSearch.this, android.R.layout.simple\_list\_item\_1,mStrSuggestions);
16. mSuggestionList.setAdapter(suggestionString);
18. }

**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：

1. mLocationManager = mBMapMan.getLocationManager();
2. LocationListener listener = new LocationListener() {
3. @Override
4. public void onLocationChanged(Location location) {
5. // TODO 在此处处理位置变化
6. }
7. };
8. // 注册监听
9. mLocationManager.requestLocationUpdates(listener);
10. // 不需要时移除监听
11. 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：

1. mBMapMan.init("我的key", new MKGeneralListener() {
2. @Override
3. public void onGetPermissionState(int iError) {
4. // TODO 返回授权验证错误，通过错误代码判断原因，MKEvent中常量值。
5. }
6. @Override
7. public void onGetNetworkState(int iError) {
8. // TODO 返回网络错误，通过错误代码判断原因，MKEvent中常量值。
9. }
10. });

**离线地图**

**初始化**

1. //写在继承MapActivity里
2. MKOfflineMap mOffline = null; //申明变量

5. //写在onCreate函数里
6. mOffline = new MKOfflineMap();
7. mOffline.init(mBMapMan, new MKOfflineMapListener() {
8. @Override
9. public void onGetOfflineMapState(int type, int state) {
10. switch (type) {
11. case MKOfflineMap.TYPE\_DOWNLOAD\_UPDATE:
12. {
13. MKOLUpdateElement update = mOffline.getUpdateInfo(state);
14. //mText.setText(String.format("%s : %d%%", update.cityName, update.ratio));
15. }
16. break;
17. case MKOfflineMap.TYPE\_NEW\_OFFLINE:
18. Log.d("OfflineDemo", [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky).format("add offlinemap num:%d", state));
19. break;
20. case MKOfflineMap.TYPE\_VER\_UPDATE:
21. Log.d("OfflineDemo", [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky).format("new offlinemap ver"));
22. break;
23. }
24. }
25. }
26. );

**导入离线包**

SDK支持导入离线包，将从官方渠道下载的离线包(栅格版,老版地图)解压，把其中的Mapdata文件夹拷入SD卡根目录下的BaiduMapSdk文件夹内。

离线地图下载地址：<http://shouji.baidu.com/map/map.html?from=3052>

1. int num = mOffline.scan();
2. //if (num != 0)
3. //mText.setText(String.format("已安装%d个离线包", num));

**WIFI下载离线包**

SDK支持在WIFI网络情况下，下载离线包。提供如下功能：

1. 返回热门城市列表。

2. 城市名搜索离线地图信息。

3. 启动下载。

4. 暂停下载。

5. 删除离线地图。

6. 多个APP共享一份离线地图数据。

详见官网Demo中OfflineDemo.java文件。