**Android 定位SDK**

百度地图Android定位SDK为基于移动客户端开发LBS应用提供基础定位能力，它较[百度Android 地图SDK](http://developer.baidu.com/map/sdk-android.htm)而言，专注满足用户获取当前位置与获得地址信息描述功能，软件包体积小。当前已拥有1.x, 2.x,3.x三个版本，其中早先版本1.x为lib库形式，短小精简。自2.x版本开始，为Service模式，功能更多，接口更友好。各版本都会长期支持。 SDK无需申请KEY，欢迎使用！

**功能介绍**

百度Android定位SDK支持Android1.5以及以上设备，提供：

* 定位功能：通过GPS、网络定位（WIFI、基站）混合定位模式，返回当前所处的位置信息
* 反地理编码功能：解析当前所处的位置坐标，获得详细的地址描述信息。

定位SDK支持多样化服务与定位策略，用户可通过通过设置不同的定位时间间隔、选择不同的定位服务模式、定制不同的定位结果信息来满足自身需求。

**兼容性**

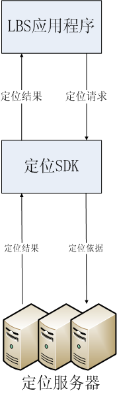
支持Android1.5及以上系统

**定位原理**

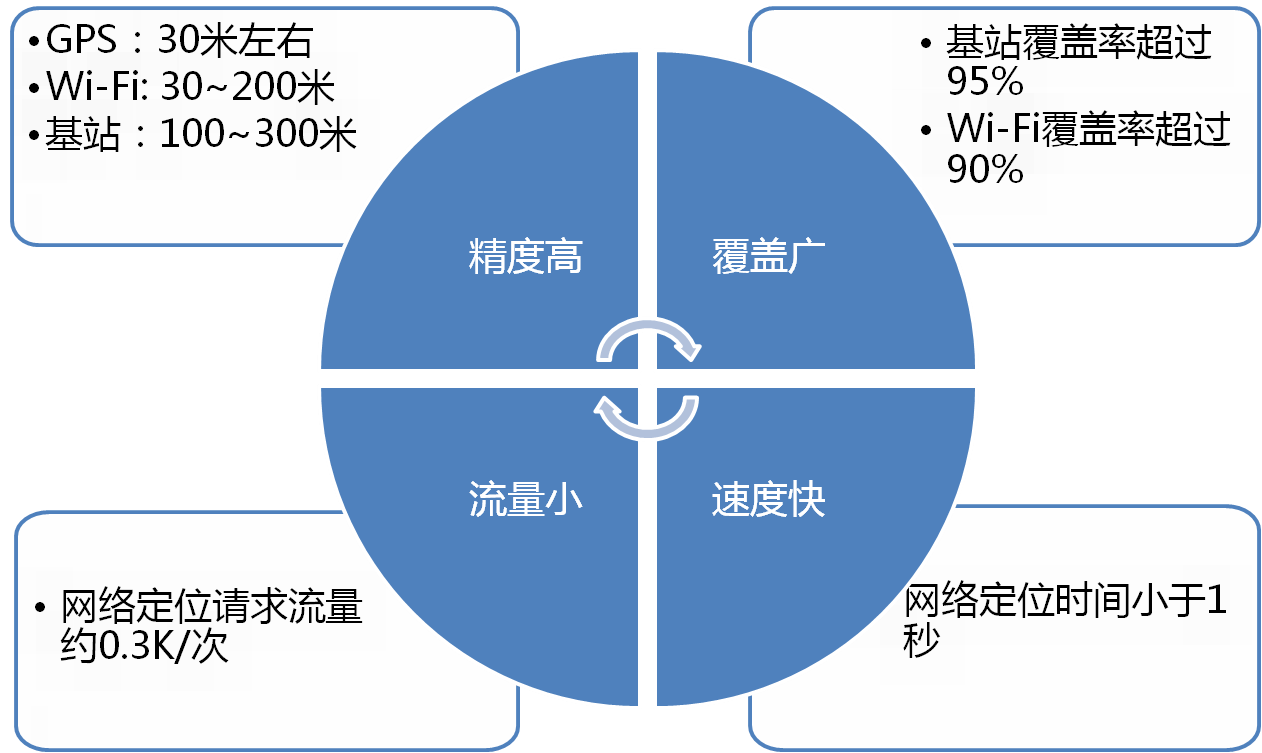
使用百度Android定位SDK必须注册GPS和网络使用权限。定位SDK采用GPS、基站、Wi-Fi信号进行定位。当应用程序向定位SDK发起定位请求时，定位SDK会根据应用的定位因素（GPS、基站、Wi-Fi信号）的实际情况（如是否开启GPS、是否连接网络、是否有信号等）来生成相应定位依据进行定位。

用户可以设置满足自身需求的定位依据：

若用户设置GPS优先，则优先使用GPS进行定位，如果GPS定位未打开或者没有可用位置信息，且网络连接正常，定位SDK则会返回网络定位（即Wi-Fi与基站）的最优结果。为了使获得的网络定位结果更加精确，请打开手机的Wi-Fi开关。



**定位精度**



# 开发指南v3.1

## 导入库文件

在[相关下载](http://developer.baidu.com/map/geosdk-android-download.htm)最新的库文件。将liblocSDK3.so文件拷贝到libs/armeabi目录下。将locSDK3.1.jar文件拷贝到工程根目录下，并在工程属性->Java Build Path->Libraries中选择“Add JARs”，选定locSDK3.1.jar，确定后返回。这样您就可以在程序中使用百度定位SDK了。

## 设置AndroidManifest.xml

在application标签中声明service组件,为了避免各个app公用1个service出现的权限问题，新版本各个app单独拥有自己的定位service.

**<service** android:name="com.baidu.location.f" android:enabled="true" android:process=":remote"**>**

**</service>**

声明使用权限

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

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

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

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

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

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

**<uses-permission** android:name="android.permission.MOUNT\_UNMOUNT\_FILESYSTEMS"**></uses-permission>**

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

## import相关类

**import** com.baidu.location.BDLocation;

**import** com.baidu.location.BDLocationListener;

**import** com.baidu.location.LocationClient;

**import** com.baidu.location.LocationClientOption;

**import** com.baidu.location.BDNotifyListener;*//假如用到位置提醒功能，需要import该类*

## 功能类的使用

#### 初始化LocationClient类

此处需要注意：LocationClient类必须在主线程中声明。需要Context类型的参数。

Context需要时全进程有效的context,推荐用getApplicationConext获取全进程有效的context

**public** LocationClient mLocationClient = **null**;

**public** BDLocationListener myListener = **new** MyLocationListener();

**public** **void** onCreate() {

mLocationClient = **new** LocationClient(**getApplicationContext()**); *//声明LocationClient类*

mLocationClient.registerLocationListener( myListener ); *//注册监听函数*

}

#### 实现BDLocationListener接口

BDLocationListener接口有2个方法需要实现： 1.接收异步返回的定位结果，参数是BDLocation类型参数。 2.接收异步返回的POI查询结果，参数是BDLocation类型参数。

**public** **class** MyLocationListenner **implements** BDLocationListener {

@Override

**public** **void** onReceiveLocation(BDLocation location) {

**if** (location == **null**)

**return** ;

[StringBuffer](http://www.google.com/search?hl=en&q=allinurl%3Astringbuffer+java.sun.com&btnI=I%27m%20Feeling%20Lucky) sb = **new** [StringBuffer](http://www.google.com/search?hl=en&q=allinurl%3Astringbuffer+java.sun.com&btnI=I%27m%20Feeling%20Lucky)(256);

sb.append("time : ");

sb.append(location.getTime());

sb.append("**\n**error code : ");

sb.append(location.getLocType());

sb.append("**\n**latitude : ");

sb.append(location.getLatitude());

sb.append("**\n**lontitude : ");

sb.append(location.getLongitude());

sb.append("**\n**radius : ");

sb.append(location.getRadius());

**if** (location.getLocType() == BDLocation.TypeGpsLocation){

sb.append("**\n**speed : ");

sb.append(location.getSpeed());

sb.append("**\n**satellite : ");

sb.append(location.getSatelliteNumber());

} **else** **if** (location.getLocType() == BDLocation.TypeNetWorkLocation){

sb.append("**\n**addr : ");

sb.append(location.getAddrStr());

}

logMsg(sb.toString());

}

**public** **void** onReceivePoi(BDLocation poiLocation) {

**if** (poiLocation == **null**){

**return** ;

}

[StringBuffer](http://www.google.com/search?hl=en&q=allinurl%3Astringbuffer+java.sun.com&btnI=I%27m%20Feeling%20Lucky) sb = **new** [StringBuffer](http://www.google.com/search?hl=en&q=allinurl%3Astringbuffer+java.sun.com&btnI=I%27m%20Feeling%20Lucky)(256);

sb.append("Poi time : ");

sb.append(poiLocation.getTime());

sb.append("**\n**error code : ");

sb.append(poiLocation.getLocType());

sb.append("**\n**latitude : ");

sb.append(poiLocation.getLatitude());

sb.append("**\n**lontitude : ");

sb.append(poiLocation.getLongitude());

sb.append("**\n**radius : ");

sb.append(poiLocation.getRadius());

**if** (poiLocation.getLocType() == BDLocation.TypeNetWorkLocation){

sb.append("**\n**addr : ");

sb.append(poiLocation.getAddrStr());

}

**if**(poiLocation.hasPoi()){

sb.append("**\n**Poi:");

sb.append(poiLocation.getPoi());

}**else**{

sb.append("noPoi information");

}

logMsg(sb.toString());

}

}

#### 设置参数

设置定位参数包括：定位模式（单次定位，定时定位），返回坐标类型，是否打开GPS等等。eg：

LocationClientOption option = **new** LocationClientOption();

option.setOpenGps(**true**);

option.setAddrType("all");*//返回的定位结果包含地址信息*

option.setCoorType("bd09ll");*//返回的定位结果是百度经纬度,默认值gcj02*

option.setScanSpan(5000);*//设置发起定位请求的间隔时间为5000ms*

option.disableCache(**true**);*//禁止启用缓存定位*

option.setPoiNumber(5); *//最多返回POI个数*

option.setPoiDistance(1000); *//poi查询距离*

option.setPoiExtraInfo(**true**); *//是否需要POI的电话和地址等详细信息*

mLocClient.setLocOption(option);

#### 发起定位请求

发起定位请求。请求过程是异步的，定位结果在上面的监听函数onReceiveLocation中获取。

**if** (mLocClient != **null** && mLocClient.isStarted())

mLocClient.requestLocation();

**else**

Log.d("LocSDK3", "locClient is null or not started");

#### 发起POI查询请求

发起POI查询请求。请求过程是异步的，定位结果在上面的监听函数onReceivePoi中获取。

**if** (mLocClient != **null** && mLocClient.isStarted())

mLocClient.requestPoi();

#### 发起离线定位请求

发起离线定位请求。请求过程是异步的，定位结果在上面的监听函数onReceiveLocation中获取。

getLocTypte = BDLocation.TypteOfflineLocation || BDLocation.TypeOfflineLocationFail

表示是离线定位请求返回的定位结果

**if** (mLocClient != **null** && mLocClient.isStarted())

mLocClient.requestOfflineLocation();

#### 位置提醒使用

位置提醒最多提醒3次，3次过后将不再提醒。 假如需要再次提醒，或者要修改提醒点坐标，都可通过函数SetNotifyLocation()来实现。

*//位置提醒相关代码*

mNotifyer = **new** NotifyLister();

mNotifyer.SetNotifyLocation(42.03249652949337,113.3129895882556,3000,"gps");*//4个参数代表要位置提醒的点的坐标，具体含义依次为：纬度，经度，距离范围，坐标系类型(gcj02,gps,bd09,bd09ll)*

mLocationClient.registerNotify(mNotifyer);

*//注册位置提醒监听事件后，可以通过SetNotifyLocation 来修改位置提醒设置，修改后立刻生效。*

*//BDNotifyListner实现*

**public** **class** NotifyLister **extends** BDNotifyListener{

**public** **void** onNotify(BDLocation mlocation, **float** distance){

mVibrator01.vibrate(1000);*//振动提醒已到设定位置附近*

}

}

*//取消位置提醒*

mLocationClient.removeNotifyEvent(mNotifyer);

## 需要注意的问题

* 定位SDK必须注册GPS和网络的使用权限。
* 使用定位SDK请保证网络连接通畅（GPS定位方式不需要连网）。
* 我们强烈建议您设置自己的prodName，并保管好，这样方便我们为您提供更好的定位服务。
* 若需要返回的定位结果里包含地址信息，请保证网络连接。
* 定位SDK可以返回bd09、bd09ll、gcj02三种类型坐标，若需要将定位点的位置通过百度Android地图 SDK进行地图展示，请返回bd09ll，将无偏差的叠加在百度地图上。
* 有的移动设备锁屏后为了省电会自动关闭网络连接，此时网络定位模式的定位失效。此外，锁屏后移动设备若进入cpu休眠，定时定位功能也失效。若您需要实现在cpu休眠状态仍需定时定位，可以用alarmManager 实现1个cpu可叫醒的timer，定时请求定位。

参考类

## LocationClient类

功能：定位SDK的核心类。方法如下：

#### 构造类：

**public** LocationClient ( [Context](http://www.google.com/search?hl=en&q=allinurl%3Acontext+java.sun.com&btnI=I%27m%20Feeling%20Lucky) ) *//须在主线程中声明*

#### 设置参数：

**public** **void** setLocOption ( LocationClientOption )

参数：配置定位SDK，详见LocationClientOption类 说明：2.x版本以后的定位给API支持在API运行过程中，热切换配置参数。

#### 注册监听函数：

**public** **void** registerLocationListener ( BDLocationListener )

参数：详见BDLocationListener类 说明：当没有注册监听函数时，无法发起网络请求。

#### 开启/关闭：

**public** **void** start();

**public** **void** stop();

start：启动定位SDK。 stop：关闭定位SDK。调用stop之后，设置的参数LocationClientOption仍然保留。

#### 发起定位：

**public** **int** requestLocation()

发起定位，异步获取当前位置。因为是异步的，所以立即返回，不会引起阻塞。定位结果在ReceiveListener的方法OnReceive方法的参数中返回。

需要注意：当定位SDK从定位依据判定，位置和上一次没发生变化，而且上一次定位结果可用时，则不会发起网络请求，而是返回上一次的定位结果。 返回值：

* 0：正常发起了定位。
* 1：服务没有启动。
* 2：没有监听函数。
* 6：请求间隔过短。 前后两次请求定位时间间隔不能小于1000ms。

#### 请求离线定位：

**public** **int** requestOfflineLocation()

发起离线定位，异步获取当前位置。因为是异步的，所以立即返回，不会引起阻塞。定位结果在ReceiveListener的方法OnReceive方法的参数中返回。

返回值：

* 0：正常发起了定位。
* 1：服务没有启动。
* 2：没有监听函数。

#### 取消监听函数：

**public** **void** unRegisterLocationListener(BDLocationListener listener)

取消监听函数。

#### 位置提醒相关函数：

*//注册位置提醒监听事件*

**public** **void** registerNotify(BDNotifyListener mNotify)

*//取消位置提醒监听事件*

**public** **void** removeNotifyEvent(BDNotifyListener mNotify)

## LocationClientOption类

该类用来设置定位SDK的定位方式。

#### 设置是否启用缓存定位

*//true表示禁用缓存定位，false表示启用缓存定位。*

**public** **void** disableCache(**boolean**)

#### 设置打开GPS

setOpenGps( **boolean** )

设置是否打开gps，使用gps前提是用户硬件打开gps。默认是不打开gps的。

#### 设置是否需要地址信息

**public** **void** setAddrType ( **boolean** )

String 值为 all时，表示返回地址信息。

其他值都表示不返回地址信息。

#### 设置坐标类型：

设置返回值的坐标类型。

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

我们支持返回若干种坐标系，包括国测局坐标系、百度坐标系，需要更多坐标系请联系我们，需要深度合作。目前这些参数的代码为。因此需要在请求时指定类型，如果不指定，默认返回百度坐标系。注意当仅输入IP时，不会返回坐标。目前这些参数的代码为

* 返回国测局经纬度坐标系 coor=gcj02
* 返回百度墨卡托坐标系 coor=bd09
* 返回百度经纬度坐标系 coor=bd09ll

百度手机地图对外接口中的坐标系默认是bd09ll，如果配合百度地图产品的话，需要注意坐标系对应问题。

有关坐标系的更多问题，请看[常见问题](http://developer.baidu.com/map/geosdk-android-qa.htm#.E7.99.BE.E5.BA.A6.E5.9C.B0.E5.9B.BE.E5.AE.9A.E4.BD.8DAPI.E4.BD.BF.E7.94.A8.E4.BB.80.E4.B9.88.E5.9D.90.E6.A0.87.E7.B3.BB.EF.BC.9F)

#### 设置产品线名称：

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

设置产品线名称。强烈建议您使用自定义的产品线名称，方便我们以后为您提供更高效准确的定位服务。

#### 设置定位时间间隔：

**public** **void** setScanSpan ( **int** ) *//设置定时定位的时间间隔。单位ms*

说明：

* 当所设的整数值大于等于1000（ms）时，定位SDK内部使用定时定位模式。调用requestLocation( )后，每隔设定的时间，定位SDK就会进行一次定位。如果定位SDK根据定位依据发现位置没有发生变化，就不会发起网络请求，返回上一次定位的结果；如果发现位置改变，就进行网络请求进行定位，得到新的定位结果。定时定位时，调用一次requestLocation，会定时监听到定位结果。
* 当不设此项，或者所设的整数值小于1000（ms）时，采用一次定位模式。每调用一次requestLocation( )，定位SDK会发起一次定位。请求定位与监听结果一一对应。
* 设定了定时定位后，可以热切换成一次定位，需要重新设置时间间隔小于1000（ms）即可。locationClient对象stop后，将不再进行定位。如果设定了定时定位模式后，多次调用requestLocation（），则是每隔一段时间进行一次定位，同时额外的定位请求也会进行定位，但频率不会超过1秒一次。

#### 设置优先级：

**public** **void** setPriority( **int** )

设置定位方式的优先级。目前定位SDK的定位方式有两类：一是使用GPS进行定位。优点是定位准确，精度在几十米，缺点是第一次定位速度较慢，甚至需要2、3分钟。二是使用网络定位。优点是定位速度快，服务端只需8ms，考虑到网速的话，一般客户端3秒左右即可定位，缺点是没有gps准确 ，精度在几十到几百米。为了方便用户，我们提供了有两个整型的项：LocationClientOption.GpsFirst 以及 LocationClientOption.NetWorkFirst：

* GpsFirst：当gps可用，而且获取了定位结果时，不再发起网络请求，直接返回给用户坐标。这个选项适合希望得到准确坐标位置的用户。如果gps不可用，再发起网络请求，进行定位。
* NetWorkFirst：即时有gps，而且可用，也仍旧会发起网络请求。这个选项适合对精确坐标不是特别敏感，但是希望得到位置描述的用户。

#### 设置POI个数：

**public** **void** setPoiNumber( **int** )

设置最多可返回的POI个数，默认值为3。由于POI查询比较耗费流量，设置最多返回的POI个数，以便节省流量。

#### 设置POI查询距离：

**public** **void** setPoiDistance( **float** )

设置查询范围，默认值为500，即以当前定位位置为中心的半径大小。

#### 设置是否需要POI的电话地址等详细信息：

**public** **void** setPoiExtraInfo( **boolean** )

设置是否返回POI的电话和地址等详细信息。默认值为false，即不返回POI的电话和地址信息。

#### 设置位置提醒接口：

**public** **void** registerNotify( BDNotifyListener mNotify )

LocationClient注册位置提醒监听事件

**public** **void** removeNotifyEvent( BDNotifyListener mNotify )

LocationClient取消位置提醒监听事件

#### 设置是否启用缓存设定说明：

**public** **void** disableCache( **boolean** )

LocationClientOption;true表示禁用缓存定位，false表示启用缓存定位

## BDLocation类

封装了定位SDK的定位结果，在BDLocationListener的onReceive方法中获取。通过该类用户可以获取error code，位置的坐标，精度半径等信息

#### 获取error code：

**public** **int** getLocType ( )

返回值：

* 61 ： GPS定位结果
* 62 ： 扫描整合定位依据失败。此时定位结果无效。
* 63 ： 网络异常，没有成功向服务器发起请求。此时定位结果无效。
* 65 ： 定位缓存的结果。
* 66 ： 离线定位结果。通过requestOfflineLocaiton调用时对应的返回结果
* 67 ： 离线定位失败。通过requestOfflineLocaiton调用时对应的返回结果
* 68 ： 网络连接失败时，查找本地离线定位时对应的返回结果
* 161： 表示网络定位结果
* 162~167： 服务端定位失败。

**如果不能定位，请记住这个返回值，并到我们的hi群或者贴吧中交流。若返回值是162~167，请发送邮件至mapapi@baidu.com反馈。**

#### 获取经纬度坐标：

**public** **double** getLatitude ( ) *//获取维度*

**public** **double** getLongitude ( ) *//获取经度*

获取定位的坐标。坐标的类型在setLocationClientOption方法中设定。

#### 获取定位精度：

**public** **boolean** hasRadius ( ) *//判断是否有定位精度半径*

**public** **float** getRadius ( ) *//获取定位精度半径，单位是米*

#### 获取文字描述的地址(反地理编码)：

**public** [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky) getAddrStr ( ) *//获取反地理编码*

只有使用网络定位的情况下，才能获取当前位置的反地理编码描述。注意：在定位SDK2.0版本中，不允许用户自定义反地理编码的格式了。返回的都是“detail”类型。

#### 获取POI信息：

**public** **boolean** hasPoi()

**public** [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky) getPoi() *//获取poi信息*

hasPoi()为false，则表明附近没有查询到相应的poi信息。poi字串JSON格式的字符串。 字串示例： {"p":[{"addr":"xx路xx号"，"y":"40.2234","dis":"324.3","x":"116.3424","name":"xx酒店","tel":""},{"y":"40.2234","dis":"324.3","x":"116.3424","name":"xx酒店"}]}

## BDLocationListener接口类

作用：获取定位结果，获取POI信息。该类有2个方法：

#### 获取定位结果：

**public** **interface** BDLocationListener {

**public** **void** onReceiveLocation ( BDLocation location );

**public** **void** onReceivePoi(BDLocation poiLocation);

}

获取定位结果的接口。用户自己实现这个接口后，监听定位结果。 获取POI信息的接口。用户自己实现这个接口后，监听poi的相关结果。

## BDNotifyListener 类

作用：位置提醒接口类，用于设定位置提醒点，以及实现监听函数。

#### 设定位置提醒点

*//用于设定位置提醒的点的具体信息，4个参数依次为：纬度，经度，半径，坐标类型*

**public** **void** SetNotifyLocation(**double** latitude, **double** longitude ,**float** radius, [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky) coorType)

#### 监听位置提醒事件

**public** **void** onNotify(BDLocation mlocation, **float** distance){

*//位置提醒监听事件具体实现.*

*//mlocation表示当前位置，distance是当前坐标中心点与设定位置提醒的坐标点之间的距离值。*

}