《百度地图实战 实验手册 05》

Android 课程组

版 本 1.0

文档提供：Java 课程组 丁盟

目录

第5章　百度定位

5.1 实验目的

目的一： 了解百度周边雷达的实现原理。

目的二： 掌握周围雷达的使用流程。

目的三： 能够通过周围雷达获取附近使用相同 App 的用户信息。

5.2 准备工作

准备一： 创建项目。

使用 Android Studio 创建一个新的空项目，并按照之前百度地图 SDK Android 开发环境配置对当前新项目进行配置。

准备二： 申请秘钥。

对当前项目申请百度地图开发者秘钥，并在项目中对秘钥信息进行设置。

准备三： 配置定位服务

本实验需要用到百度定位服务，所以需要在项目中 AndroidManifest.xml 中的 application 节点下配置一个定位服务。

```
<!-- 百度定位服务 -->
<service
    android:name="com.baidu.location.f"
    android:enabled="true"
    android:process=":remote" />
```

在 AndroidManifest.xml 中增加百度地图 SDK 及百度定位 SDK 所需权限。

```
<!-- 这个权限用于进行网络定位-->
<uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATIO
N" />
<!-- 这个权限用于访问 GPS 定位-->
```

```
<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION"
/>
<!-- 用于访问 wifi 网络信息，wifi 信息会用于进行网络定位-->
<uses-permission
android:name="android.permission.ACCESS_WIFI_STATE" />
<!-- 获取运营商信息，用于支持提供运营商信息相关的接口-->
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE"
/>
<!-- 这个权限用于获取 wifi 的获取权限，wifi 信息会用来进行网络定
位-->
<uses-permission
android:name="android.permission.CHANGE_WIFI_STATE" />
<!-- 用于读取手机当前的状态-->
<uses-permission
android:name="android.permission.READ_PHONE_STAT" />
<!-- 写入扩展存储，向扩展卡写入数据，用于写入离线定位数据-->
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAG
E" />
<!-- 访问网络，网络定位需要上网-->
<uses-permission
android:name="android.permission.INTERNET" />
<!-- SD 卡读取权限，用户写入离线定位数据-->
<uses-permission
android:name="android.permission.MOUNT_UNMOUNT_FILESYS
TEMS" />
```

准备四： 申请周边雷达服务

登录下面网站选择"新建周边雷达"。

http://lbsyun.baidu.com/index.php?title=radar

图 5.2.1

填写描述名称后点击下一步（描述名称主要起区分周边雷达服务用，无实际意义），然后再左侧列表中选择开启周边雷达的应用，选中复选框（可选择多个），点击下一步。

图 5.2.2

最后选择完成，完成周边雷达与应用的关联。完成后可以看到当前开启周边雷达的应用列表。

图 5.2.3

图 5.2.4

当前面准备工作完成之后，在模拟器或真机中运行项目能够显示如下效果即表示基本项目配置准备工作已完成。

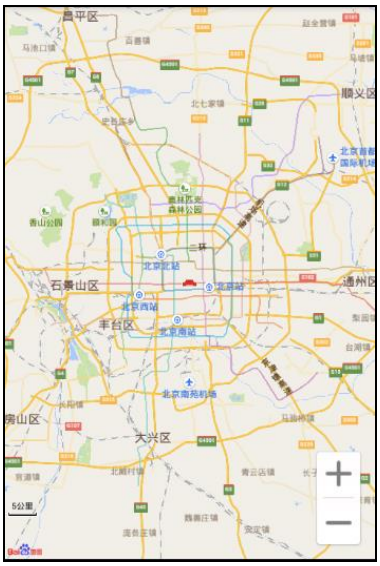


图 5.2.5

5.3 实验步骤

步骤一 在 MainActivity 中分别声明地图视图、地图控制器、定位客户端、周边雷达管理器等属性成员。

```
/* 地图控件 */
private MapView mMapView = null;
/* 地图实例 */
private BaiduMap mBaiduMap = null;
/* 定位的客户端 */
private LocationClient mLocationClient = null;
/* 周边雷达管理器 */
private RadarSearchManager mRadarSearchManager = null;
/* 定位的监听器 */
public MyLocationListener mMyLocationListener = null;
/* 周边雷达的监听器 */
MyRadarSearchListener mRadarSerchListener = null;

/* 是否是第一次定位 */
private volatile boolean isFristLocation = true;

/* 最新一次的经纬度 */
private double mCurrentLantitude;
private double mCurrentLongitude;

/* UserID */
private String mUserID = "钢铁侠";
```

步骤二 在 MainActivity 中创建定位监听类。

```
/**
 * 实现定位回调监听
 */
public class MyLocationListener implements
BDLocationListener {
    @Override
    public void onReceiveLocation(BDLocation location) {

        // mapView 销毁后不在处理新接收的位置
        if (location == null || mMapView == null)
            return;

        // 构造定位数据
        MyLocationData locData = new MyLocationData.Builder()
```

```
                // 此处设置开发者获取到的方向信息，顺时针 0-360
                .latitude(location.getLatitude())
                .longitude(location.getLongitude()).build();

        // 设置 BaiduMap 的定位数据
        mBaiduMap.setMyLocationData(locData);
        // 记录位置信息
        mCurrentLantitude = location.getLatitude();
        mCurrentLongitude = location.getLongitude();

        // 第一次定位时，将地图位置移动到当前位置
        if (isFristLocation) {
            isFristLocation = false;
            center2myLoc();
        }
    }
}
```

步骤三 在 MainActivity 中创建周边雷达监听类。

```
/**
 * 设置周边雷达监听
 */
public class MyRadarSearchListener
        implements RadarSearchListener,
RadarUploadInfoCallback {
    /* 上传位置时自动调用的回调接口 */
    @Override
    public RadarUploadInfo onUploadInfoCallback() {
        // 将要上传的 Info
        RadarUploadInfo info = new RadarUploadInfo();
        // Info 的备注信息
        SimpleDateFormat simpleDateFormat = new
SimpleDateFormat("hhmmss");
        Date curDate = new Date(System.currentTimeMillis());
        String str = simpleDateFormat.format(curDate);
        info.comments = str;
        // Info 的点信息
        LatLng pt = new LatLng(mCurrentLantitude,
mCurrentLongitude);
        info.pt = pt;
        // 返回要上传的信息，即上传信息
        return info;
    }
```

```
    /* 上传状态监听 */
    @Override
    public void onGetUploadState(RadarSearchError
radarSearchError) {
        SimpleDateFormat simpleDateFormat = new
SimpleDateFormat("hh:mm:ss :");
        Date curDate = new Date(System.currentTimeMillis());
        String strTime = simpleDateFormat.format(curDate);
        if (radarSearchError ==
RadarSearchError.RADAR_NO_ERROR) {
            //上传成功
            Log.i("RadarUpload", strTime + "上传成功");
        } else {
            //上传失败
            Log.i("RadarUpload", strTime + "上传错误: " +
radarSearchError.toString());
        }
    }

    /* 查询周边的人监听 */
    @Override
    public void onGetNearbyInfoList(RadarNearbyResult
radarNearbyResult,
                                    RadarSearchError
radarSearchError) {
        if (radarSearchError ==
RadarSearchError.RADAR_NO_ERROR) {
            Log.i("RadarUpload", "查询周边成功");
            // 清理覆盖物
            mBaiduMap.clear();
            for (int i=0; i<
radarNearbyResult.infoList.size(); i++) {
                Log.i("RadarUpload", "NO." + i + " : " +

radarNearbyResult.infoList.get(i).userID + "\n" +

radarNearbyResult.infoList.get(i).comments + "\n" +

radarNearbyResult.infoList.get(i).distance + "\n" +
                        radarNearbyResult.infoList.get(i).pt +
"\n" +

radarNearbyResult.infoList.get(i).timeStamp);

addMarker(radarNearbyResult.infoList.get(i).userID,
```

```
radarNearbyResult.infoList.get(i).pt);
            }
        } else {
            Log.i("RadarUpload", "查询周边错误：" +
radarSearchError.toString());
        }
    }

    /* 清除位置信息监听 */
    @Override
    public void onGetClearInfoState(RadarSearchError
radarSearchError) {
        if (radarSearchError ==
RadarSearchError.RADAR_NO_ERROR) {
            //清除成功
            Log.i("RadarUpload", "清除位置成功");
        } else {
            //清除失败
            Log.i("RadarUpload", "清除位置失败");
        }
    }
}
```

步骤四 在 MainActivity 中创建 initBaiduMap()方法，用来完成百度地图初始化工作，并在 onCreate()中调用。

```
/**
 * 初始化百度地图
 */
private void initBaiduMap() {
    mMapView = (MapView)findViewById(R.id.bmapView);
    mBaiduMap = mMapView.getMap();
    MapStatusUpdate msu =
MapStatusUpdateFactory.zoomTo(15.0f);
    mBaiduMap.setMapStatus(msu);
}
```

步骤五 在 MainActivity 中创建 initMyLocation ()方法，用来完成百度定位客户端的初始化工作，并在 onCreate()中调用。

```
/**
 * 初始化定位相关代码
 */
```

```
private void initMyLocation() {
    // 定位SDK初始化
    mLocationClient = new
LocationClient(getApplicationContext());

    // 设置定位的相关配置
    LocationClientOption option = new
LocationClientOption();
    option.setOpenGps(true);      // 打开gps
    option.setCoorType("bd09ll"); // 设置坐标类型
    option.setScanSpan(1000);     // 自动定位间隔

    // 定位模式
    option.setLocationMode(LocationClientOption
           .LocationMode.Hight_Accuracy);

    // 根据配置信息对定位客户端进行设置
    mLocationClient.setLocOption(option);

    // 注册定位监听
    mMyLocationListener = new MyLocationListener();

mLocationClient.registerLocationListener(mMyLocationListe
ner);

    // 构建Marker图标
    int n;
    if(mUserID.equals("钢铁侠"))
        n = R.drawable.gangtiexia;
    else if(mUserID.equals("蝙蝠侠"))
        n = R.drawable.bianfuxia;
    else if(mUserID.equals("闪电侠"))
        n = R.drawable.shandianxia;
    else
        n = R.drawable.sishen;
    BitmapDescriptor bitmap = BitmapDescriptorFactory
           .fromResource(n);

    // 设置百度地图定位图层显示模式
    MyLocationConfiguration config = new
MyLocationConfiguration(
           MyLocationConfiguration.LocationMode.NORMAL,
           true,
           bitmap);
    mBaiduMap.setMyLocationConfigeration(config);
```

```
}
```

步骤六 在 MainActivity 中创建 initRadarSearch ()方法，用来完成周边雷达的初始化工作，并在 onCreate()中调用。

```
**
 * 初始化周边雷达相关
 */
private void initRadarSearch() {
    // 获取周边雷达实例
    mRadarSearchManager = RadarSearchManager.getInstance();
    // 周边雷达设置监听，RadarSearchListener 接口实现
    mRadarSerchListener = new MyRadarSearchListener();

mRadarSearchManager.addNearbyInfoListener(mRadarSerchList
ener);
    // 周边雷达设置用户身份标识，id 为null 是设备标识，必须设置
    mRadarSearchManager.setUserID(mUserID);
}
```

步骤七 实现百度地图、百度定位客户端的生命周期相关接口。

```
@Override
protected void onStart()
{
    // 开启图层定位
    mBaiduMap.setMyLocationEnabled(true);
    if (!mLocationClient.isStarted())
    {
        mLocationClient.start();
    }

    super.onStart();
}

@Override
protected void onStop()
{
    // 关闭图层定位
    mBaiduMap.setMyLocationEnabled(false);
    mLocationClient.stop();

    super.onStop();
}
```

```
@Override
protected void onDestroy() {
    //移除监听

mRadarSearchManager.removeNearbyInfoListener(mRadarSerchL
istener);
    //释放资源
    mRadarSearchManager.destroy();
    mRadarSearchManager = null;

    super.onDestroy();
    // 在activity 执行onDestroy 时执行mMapView.onDestroy()，
实现地图生命周期管理
    mMapView.onDestroy();
}

@Override
protected void onResume() {
    super.onResume();
    // 在activity 执行onResume 时执行mMapView. onResume ()，
实现地图生命周期管理
    mMapView.onResume();
}

@Override
protected void onPause() {
    super.onPause();
    // 在activity 执行onPause 时执行mMapView. onPause ()，实现
地图生命周期管理
    mMapView.onPause();
}
```

步骤八 定义 center2myLoc 函数来实现 BaiduMap 移动到定位位置。

```
/**
 * BaiduMap 移动到我的位置
 */
private void center2myLoc() {
    LatLng ll = new LatLng(mCurrentLantitude,
mCurrentLongitude);
    // 设置当前定位位置为BaiduMap 的中心点，并移动到定位位置
    MapStatusUpdate u =
MapStatusUpdateFactory.newLatLng(ll);
```

```
    mBaiduMap.animateMapStatus(u);
}
```

步骤九 定义 addMarker 函数来实现标注覆盖物的添加。

```
/**
 * 添加标注覆盖物
 */
private void addMarker(String userID, LatLng pt) {

    int n;
    if(userID.equals("钢铁侠"))
        n = R.drawable.gangtiexia;
    else if(userID.equals("蝙蝠侠"))
        n = R.drawable.bianfuxia;
    else if(userID.equals("闪电侠"))
        n = R.drawable.shandianxia;
    else
        n = R.drawable.sishen;

    // 构建Marker 图标
    BitmapDescriptor bitmap = BitmapDescriptorFactory
            .fromResource(n);

    // 构建MarkerOption，用于在地图上添加Marker
    OverlayOptions option = new MarkerOptions()
            .position(pt)// 设置marker 的位置
            .icon(bitmap);  // 必须设置marker 图标
    //在地图上添加Marker，并显示
    Marker marker = (Marker)mBaiduMap.addOverlay(option);
}
```

步骤十 在 res 下的 drawable 下添加标注覆盖物所需的图片资源。

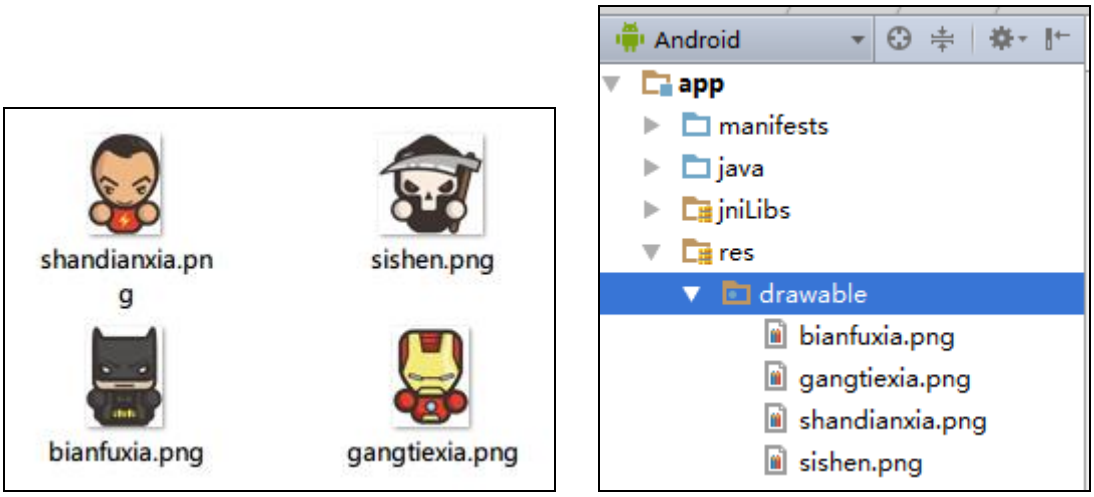


图 5.3.1

步骤十一 在 res 目录下创建 menu 目录，并添加 main.xml 文件。

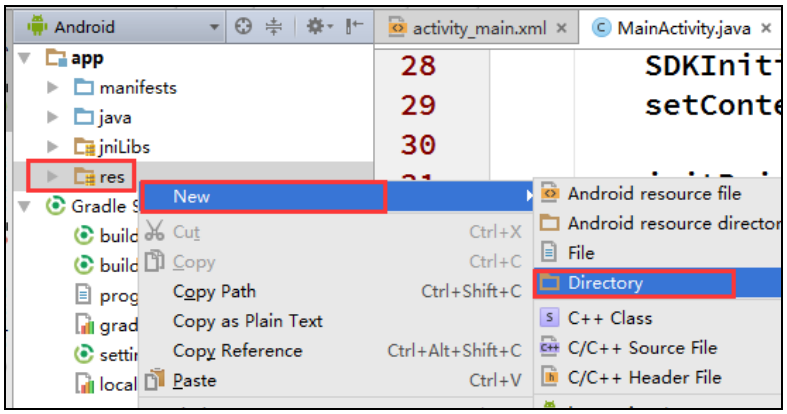


图 5.3.2

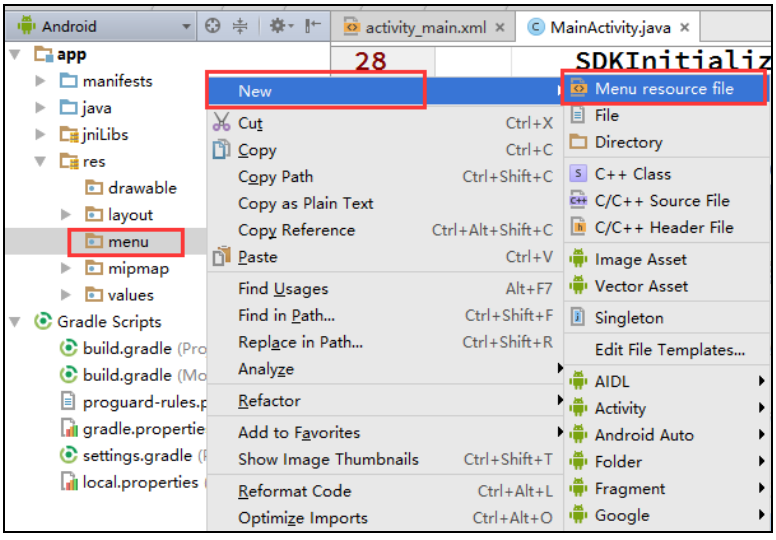


图 5.3.3

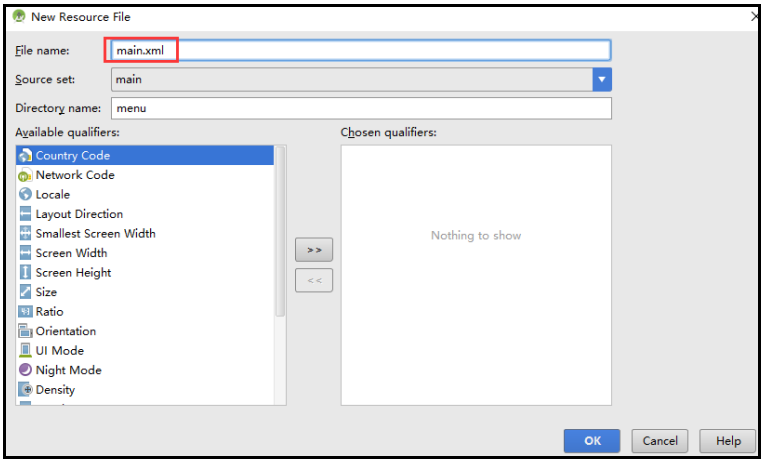


图 5.3.4

步骤十二 当 main.xml 文件创建完成之后，在其中添加如下 item 项。

(main.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<menu
xmlns:android="http://schemas.android.com/apk/res/android
"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <item
        android:id="@+id/id_menu_map_myloc"
        android:title="我的位置"
        app:showAsAction="never" />

    <item
        android:id="@+id/id_menu_map_upload"
        android:title="上传位置"
        app:showAsAction="never" />

    <item
        android:id="@+id/id_menu_map_destory"
        android:title="销毁位置"
        app:showAsAction="never" />

    <item
        android:id="@+id/id_menu_map_search"
        android:title="查询周边"
        app:showAsAction="never" />

</menu>
```

步骤十三　在 MainActivity 类中实现菜单创建接口。

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu, menu);
    return true;
}
```

步骤十四　在 MainActivity 类中实现菜单点击相应接口。

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.id_menu_map_myloc:    // 我的位置
            center2myLoc();
            break;

        case R.id.id_menu_map_upload:    // 上传位置

mRadarSearchManager.startUploadAuto(mRadarSerchListener,
                    5000);
            break;

        case R.id.id_menu_map_destory:   // 销毁位置
            // 停止上传位置信息
            mRadarSearchManager.stopUploadAuto();
            // 清除用户信息
            mRadarSearchManager.clearUserInfo();
            break;

        case R.id.id_menu_map_search:   // 查询周边
            // 构造请求参数，其中 centerPt 是自己的位置坐标
            LatLng ll = new LatLng(mCurrentLantitude,
mCurrentLongitude);
            RadarNearbySearchOption option
                    = new RadarNearbySearchOption()
                    .centerPt(ll)       // 搜索中心点
                    .pageNum(0)         // 分页编号
                    .pageCapacity(50)   // 每页容量
                    .radius(2000);      // 检索半径
            // 发起查询请求
            mRadarSearchManager.nearbyInfoRequest(option);
            break;
    }
```

```
    return super.onOptionsItemSelected(item);
}
```

步骤十五　最终 MainActivity 类完整代码如下。(MainActivity.java)

```
public class MainActivity extends AppCompatActivity {
    /* 地图控件 */
    private MapView mMapView = null;
    /* 地图实例 */
    private BaiduMap mBaiduMap = null;
    /* 定位的客户端 */
    private LocationClient mLocationClient = null;
    /* 周边雷达管理器 */
    private RadarSearchManager mRadarSearchManager = null;
    /* 定位的监听器 */
    public MyLocationListener mMyLocationListener = null;
    /* 周边雷达的监听器 */
    MyRadarSearchListener mRadarSerchListener = null;

    /* 是否是第一次定位 */
    private volatile boolean isFristLocation = true;

    /* 最新一次的经纬度 */
    private double mCurrentLantitude;
    private double mCurrentLongitude;

    /* UserID */
    private String mUserID = "钢铁侠";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        SDKInitializer.initialize(getApplicationContext());
        setContentView(R.layout.activity_main);

        // 初始化 BaiduMap 相关
        initBaiduMap();

        // 初始化百度定位客户端
        initMyLocation();

        // 初始化周边雷达
        initRadarSearch();
```

```
    }

    /**
     * 初始化周边雷达相关
     */
    private void initRadarSearch() {
        // 获取周边雷达实例
        mRadarSearchManager =
RadarSearchManager.getInstance();
        // 周边雷达设置监听，RadarSearchListener 接口实现
        mRadarSerchListener = new MyRadarSearchListener();

mRadarSearchManager.addNearbyInfoListener(mRadarSerchList
ener);
        // 周边雷达设置用户身份标识，id 为null 是设备标识，必须设置
        mRadarSearchManager.setUserID(mUserID);
    }

    /**
     * 初始化百度地图相关
     */
    private void initBaiduMap() {
        mMapView = (MapView)findViewById(R.id.bmapView);
        mBaiduMap = mMapView.getMap();
        MapStatusUpdate msu =
MapStatusUpdateFactory.zoomTo(15.0f);
        mBaiduMap.setMapStatus(msu);
    }

    /**
     * 初始化定位相关代码
     */
    private void initMyLocation() {
        // 定位 SDK 初始化
        mLocationClient = new
LocationClient(getApplicationContext());

        // 设置定位的相关配置
        LocationClientOption option = new
LocationClientOption();
        option.setOpenGps(true);      // 打开 gps
        option.setCoorType("bd09ll"); // 设置坐标类型
        option.setScanSpan(1000);     // 自动定位间隔
```

```
        // 定位模式
        option.setLocationMode(LocationClientOption
                .LocationMode.Hight_Accuracy);

        // 根据配置信息对定位客户端进行设置
        mLocationClient.setLocOption(option);

        // 注册定位监听
        mMyLocationListener = new MyLocationListener();

mLocationClient.registerLocationListener(mMyLocationListe
ner);

        // 构建 Marker 图标
        int n;
        if(mUserID.equals("钢铁侠"))
            n = R.drawable.gangtiexia;
        else if(mUserID.equals("蝙蝠侠"))
            n = R.drawable.bianfuxia;
        else if(mUserID.equals("闪电侠"))
            n = R.drawable.shandianxia;
        else
            n = R.drawable.sishen;
        BitmapDescriptor bitmap = BitmapDescriptorFactory
                .fromResource(n);

        // 设置百度地图定位图层显示模式
        MyLocationConfiguration config = new
MyLocationConfiguration(
                MyLocationConfiguration.LocationMode.NORMAL,
                true,
                bitmap);
        mBaiduMap.setMyLocationConfigeration(config);
    }

    /**
     * BaiduMap 移动到我的位置
     */
    private void center2myLoc() {
        LatLng ll = new LatLng(mCurrentLantitude,
mCurrentLongitude);
        // 设置当前定位位置为 BaiduMap 的中心点，并移动到定位位置
        MapStatusUpdate u =
MapStatusUpdateFactory.newLatLng(ll);
```

```
        mBaiduMap.animateMapStatus(u);
    }

    /**
     * 设置周边雷达监听
     */
    public class MyRadarSearchListener
            implements RadarSearchListener,
RadarUploadInfoCallback {
        /* 上传位置时自动调用的回调接口 */
        @Override
        public RadarUploadInfo onUploadInfoCallback() {
            // 将要上传的 Info
            RadarUploadInfo info = new RadarUploadInfo();
            // Info 的备注信息
            SimpleDateFormat simpleDateFormat = new
SimpleDateFormat("hhmmss");
            Date curDate = new
Date(System.currentTimeMillis());
            String str = simpleDateFormat.format(curDate);
            info.comments = str;
            // Info 的点信息
            LatLng pt = new LatLng(mCurrentLantitude,
mCurrentLongitude);
            info.pt = pt;
            // 返回要上传的信息，即上传信息
            return info;
        }

        /* 上传状态监听 */
        @Override
        public void onGetUploadState(RadarSearchError
radarSearchError) {
            SimpleDateFormat simpleDateFormat = new
SimpleDateFormat("hh:mm:ss :");
            Date curDate = new
Date(System.currentTimeMillis());
            String strTime =
simpleDateFormat.format(curDate);
            if (radarSearchError ==
RadarSearchError.RADAR_NO_ERROR) {
                //上传成功
                Log.i("RadarUpload", strTime + "上传成功");
            } else {
```

```
                //上传失败
                Log.i("RadarUpload", strTime + "上传错误：" +
radarSearchError.toString());
            }
        }

        /* 查询周边的人监听 */
        @Override
        public void onGetNearbyInfoList(RadarNearbyResult
radarNearbyResult,
                                        RadarSearchError
radarSearchError) {
            if (radarSearchError ==
RadarSearchError.RADAR_NO_ERROR) {
                Log.i("RadarUpload", "查询周边成功");
                // 清理覆盖物
                mBaiduMap.clear();
                for (int i=0; i<
radarNearbyResult.infoList.size(); i++) {
                    Log.i("RadarUpload", "NO." + i + " : " +

radarNearbyResult.infoList.get(i).userID + "\n" +

radarNearbyResult.infoList.get(i).comments + "\n" +

radarNearbyResult.infoList.get(i).distance + "\n" +

radarNearbyResult.infoList.get(i).pt + "\n" +

radarNearbyResult.infoList.get(i).timeStamp);

addMarker(radarNearbyResult.infoList.get(i).userID,

radarNearbyResult.infoList.get(i).pt);
                }
            } else {
                Log.i("RadarUpload", "查询周边错误：" +
radarSearchError.toString());
            }
        }

        /* 清除位置信息监听 */
        @Override
        public void onGetClearInfoState(RadarSearchError
```

```
radarSearchError) {
          if (radarSearchError ==
RadarSearchError.RADAR_NO_ERROR) {
              //清除成功
              Log.i("RadarUpload", "清除位置成功");
          } else {
              //清除失败
              Log.i("RadarUpload", "清除位置失败");
          }
       }
    }

    /**
     * 添加标注覆盖物
     */
    private void addMarker(String userID, LatLng pt) {

       int n;
       if(userID.equals("钢铁侠"))
          n = R.drawable.gangtiexia;
       else if(userID.equals("蝙蝠侠"))
          n = R.drawable.bianfuxia;
       else if(userID.equals("闪电侠"))
          n = R.drawable.shandianxia;
       else
          n = R.drawable.sishen;

       // 构建Marker图标
       BitmapDescriptor bitmap = BitmapDescriptorFactory
             .fromResource(n);

       // 构建MarkerOption，用于在地图上添加Marker
       OverlayOptions option = new MarkerOptions()
             .position(pt)// 设置marker的位置
             .icon(bitmap);  // 必须设置marker图标
       //在地图上添加Marker，并显示
       Marker marker =
(Marker)mBaiduMap.addOverlay(option);
    }

    /**
     * 实现定位回调监听
     */
    public class MyLocationListener implements
```

```
BDLocationListener {
        @Override
        public void onReceiveLocation(BDLocation location) {

            // mapView 销毁后不在处理新接收的位置
            if (location == null || mMapView == null)
                return;

            // 构造定位数据
            MyLocationData locData = new
MyLocationData.Builder()
                    // 此处设置开发者获取到的方向信息，顺时针 0-360
                    .latitude(location.getLatitude())
                    .longitude(location.getLongitude()).build
();

            // 设置 BaiduMap 的定位数据
            mBaiduMap.setMyLocationData(locData);
            // 记录位置信息
            mCurrentLantitude = location.getLatitude();
            mCurrentLongitude = location.getLongitude();

            // 第一次定位时，将地图位置移动到当前位置
            if (isFristLocation) {
                isFristLocation = false;
                center2myLoc();
            }
        }
    }

    @Override
    protected void onStart()
    {
        // 开启图层定位
        mBaiduMap.setMyLocationEnabled(true);
        if (!mLocationClient.isStarted())
        {
            mLocationClient.start();
        }

        super.onStart();
    }

    @Override
```

```
    protected void onStop()
    {
        // 关闭图层定位
        mBaiduMap.setMyLocationEnabled(false);
        mLocationClient.stop();

        super.onStop();
    }

    @Override
    protected void onDestroy() {
        //移除监听

mRadarSearchManager.removeNearbyInfoListener(mRadarSerchL
istener);
        //释放资源
        mRadarSearchManager.destroy();
        mRadarSearchManager = null;

        super.onDestroy();
        // 在activity执行onDestroy时执行
mMapView.onDestroy()，实现地图生命周期管理
        mMapView.onDestroy();
    }
    @Override
    protected void onResume() {
        super.onResume();
        // 在activity执行onResume时执行mMapView. onResume ()，
实现地图生命周期管理
        mMapView.onResume();
    }
    @Override
    protected void onPause() {
        super.onPause();
        // 在activity执行onPause时执行mMapView. onPause ()，
实现地图生命周期管理
        mMapView.onPause();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu)
    {
        getMenuInflater().inflate(R.menu.main, menu);
        return super.onCreateOptionsMenu(menu);
```

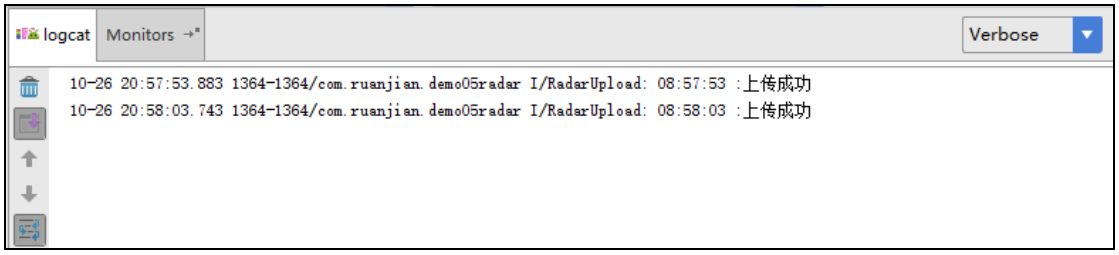
```
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.id_menu_map_myloc:    // 我的位置
                center2myLoc();
                break;

            case R.id.id_menu_map_upload:    // 上传位置

mRadarSearchManager.startUploadAuto(mRadarSerchListener,
                        5000);
                break;

            case R.id.id_menu_map_destory:   // 销毁位置
                // 停止上传位置信息
                mRadarSearchManager.stopUploadAuto();
                // 清除用户信息
                mRadarSearchManager.clearUserInfo();
                break;

            case R.id.id_menu_map_search:   // 查询周边
                // 构造请求参数，其中centerPt是自己的位置坐标
                LatLng ll = new LatLng(mCurrentLantitude,
mCurrentLongitude);
                RadarNearbySearchOption option
                        = new RadarNearbySearchOption()
                        .centerPt(ll)       // 搜索中心点
                        .pageNum(0)         // 分页编号
                        .pageCapacity(50)   // 每页容量
                        .radius(2000);      // 检索半径
                // 发起查询请求

mRadarSearchManager.nearbyInfoRequest(option);
                break;
        }

        return super.onOptionsItemSelected(item);
    }
}
```

5.4 实验结论

当编码工作完成后在模拟器或真机中运行项目。

步骤一：

运行项目后在菜单中选择上传位置，会打印上传成功的 Log 信息，此时 UserID 为"钢铁侠"的位置信息已经上传到百度的 LBS 服务中。



步骤二：

关闭程序，项目中修改 mUserId 的数值为"闪电侠"后运行项目，并在改变位置后选择"上传位置"将新 UserID 的位置信息进行上传，然后关闭项目。

步骤三：

在项目中修改 mUserId 的数值为"蝙蝠侠"后运行项目，先进行"上传位置"操作，上传位置操作成功后，在菜单中选择"查询周边"，对周边使用相同 App 的用户进行查询，此时程序界面会显示如下效果，UserID 为"钢铁侠"，"闪电侠"的两个位置信息会在地图上显现出来，并且会打印如下 Log 信息。


河北师范大学-数信学院
师大新校区-移动物联网研究院
师大新校区-物流系
南门
师大新校区-软件学院
河北师范大学(新校区)
河北师大新校区-行政办公楼B座
河北师范大学新校区-研究生院
10米
河北师范大学(新校区)

图 5.4.1

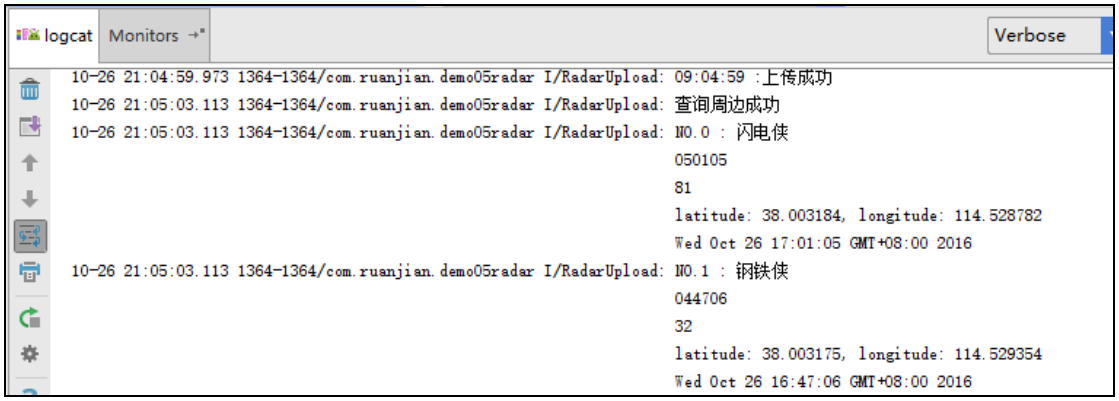

logcat
Monitors
Verbose
10-26 21:04:59.973 1364-1364/com.ruanjian.demo05radar I/RadarUpload: 09:04:59 :上传成功
10-26 21:05:03.113 1364-1364/com.ruanjian.demo05radar I/RadarUpload: 查询周边成功
10-26 21:05:03.113 1364-1364/com.ruanjian.demo05radar I/RadarUpload: NO.0 : 闪电侠
050105
81
latitude: 38.003184, longitude: 114.528782
Wed Oct 26 17:01:05 GMT+08:00 2016
10-26 21:05:03.113 1364-1364/com.ruanjian.demo05radar I/RadarUpload: NO.1 : 钢铁侠
044706
32
latitude: 38.003175, longitude: 114.529354
Wed Oct 26 16:47:06 GMT+08:00 2016

图 5.4.2