|  |
| --- |
| BRIGHT BEACON |
| 智石科技室内定位、地图 |
| Android API开发指南 |
|  |
| **V2.1.3** |
| **2017/04/28** |

|  |
| --- |
| 智石说（北京）科技有限公司 |

目录

[一、 产品简介 4](#_Toc482293292)

[1. 概述 4](#_Toc482293293)

[2. 产品特色 4](#_Toc482293294)

[二、 地图集成 4](#_Toc482293295)

[1. 获取建筑ID、用户AppKey及定位UUID 4](#_Toc482293296)

[2. 准备工作 6](#_Toc482293297)

[2.1 资源文件下载 6](#_Toc482293298)

[2.2 创建Android项目 6](#_Toc482293299)

[2.3 配置APP权限 7](#_Toc482293300)

[2.4 配置beacon扫描服务 7](#_Toc482293301)

[三、 地图显示 7](#_Toc482293302)

[1. 地图初始化 7](#_Toc482293303)

[2. 地图回调 8](#_Toc482293304)

[四、 基础功能 9](#_Toc482293305)

[1. 地图弹窗 9](#_Toc482293306)

[1.1 弹窗样式配置 9](#_Toc482293307)

[1.2 弹窗回调 9](#_Toc482293308)

[2. POI操作 10](#_Toc482293309)

[2.1 获取单个POI 11](#_Toc482293310)

[2.2 获取POI列表 11](#_Toc482293311)

[2.3 获取POI对象类型 11](#_Toc482293312)

[2.4 高亮显示POI 11](#_Toc482293313)

[3. 线路规划 11](#_Toc482293314)

[3.1 设置起点 12](#_Toc482293315)

[3.2 设置终点和跨层点 12](#_Toc482293316)

[3.3 开启路径规划 13](#_Toc482293317)

[3.4 实现路径规划回调 13](#_Toc482293318)

[3.5 其他相关方法 13](#_Toc482293319)

[4. POI查询 14](#_Toc482293320)

[4.1 基本查询 14](#_Toc482293321)

[4.2 查询相关方法 14](#_Toc482293322)

[五、 室内定位 14](#_Toc482293323)

[1. 添加蓝牙服务及相关权限 14](#_Toc482293324)

[2. 初始化定位管理器 14](#_Toc482293325)

[3. 实现回调接口 15](#_Toc482293326)

[4. 启动定位 16](#_Toc482293327)

[六、 技术支持 16](#_Toc482293328)

# 产品简介

## 概述

智石科技室内地图 Java API，是由 Java 语言编写的应用程序接口，它能够帮助您在网站或移动端中构建功能丰富、交互性强的室内地图应用程序。除了基本地图功能的接口外，室内地图Java API还提供了诸如室内搜索、室内路线规划等数据服务，您可以根据自己的需要进行选择性使用。

室内定位开发包是基于ArcGIS框架（[关于ArcGIS](https://developers.arcgis.com/ios/" \t "_blank)）和GEOS几何计算开源库，为开发者提供了的室内地图显示、路径规划、室内定位等相关GIS功能。本开发包支持的Android版本为18或更高的系统。

## 产品特色

轻捷的架构

1. API文件小，简单易用
2. 模块化地图功能，各模块按需加载

丰富的可视化效果

1. 地图拖拽、缩放平滑流畅
2. 点标记、信息窗体效果绚丽

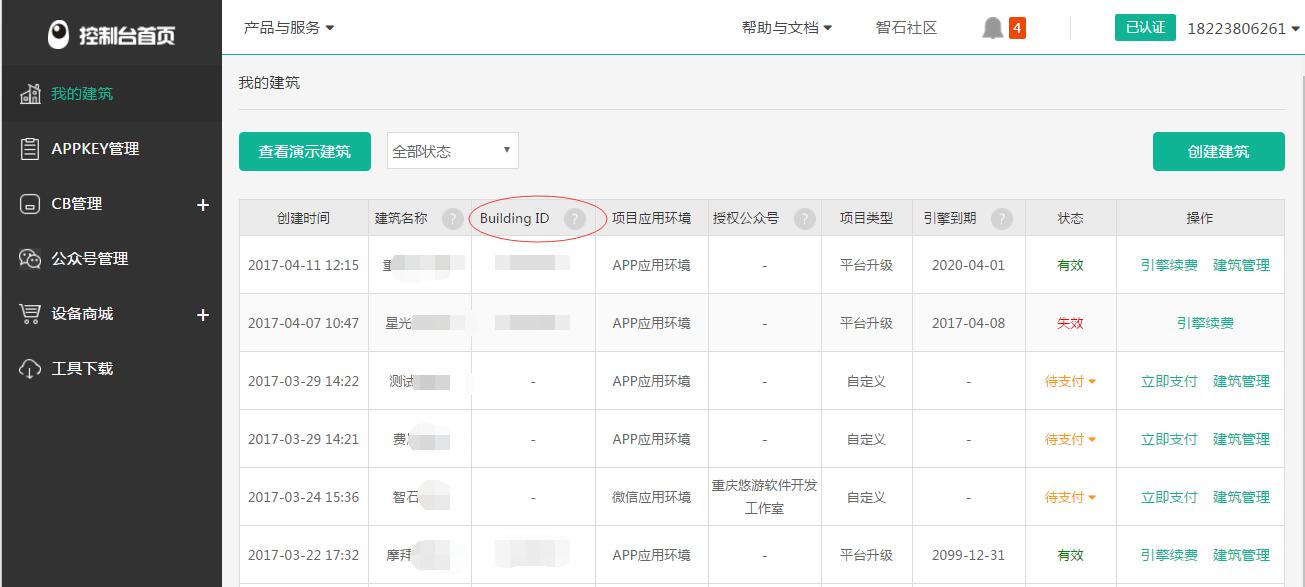
灵活开放的接口设计

1. 支持用户自定义点标记、信息窗口等的样式
2. 支持用户自定义插件工具，灵活的进行地图操作

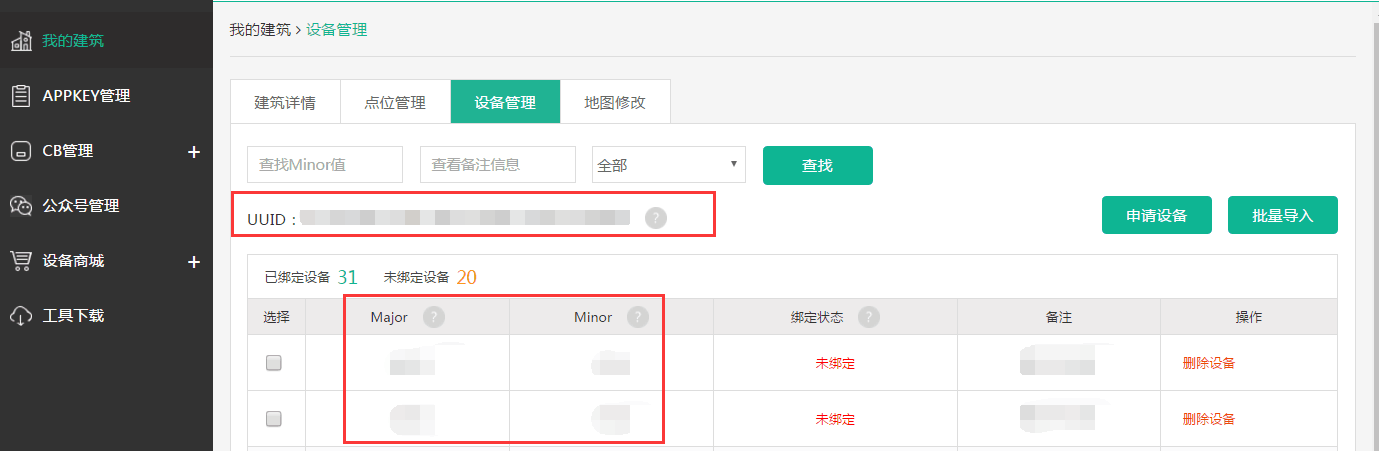
# 地图集成

## 获取建筑ID、用户AppKey及定位UUID

1. 您需要成为智石的开发者用户（[注册开发者账户](http://open.brtbeacon.com/)）；
2. 在[开放平台](http://open.brtbeacon.com/)创建您的appkey（可以参阅[开发者中心使用手册](http://help.brtbeacon.com/help.html?id=1)）；
3. 获取UUID、建筑ID、AppKey







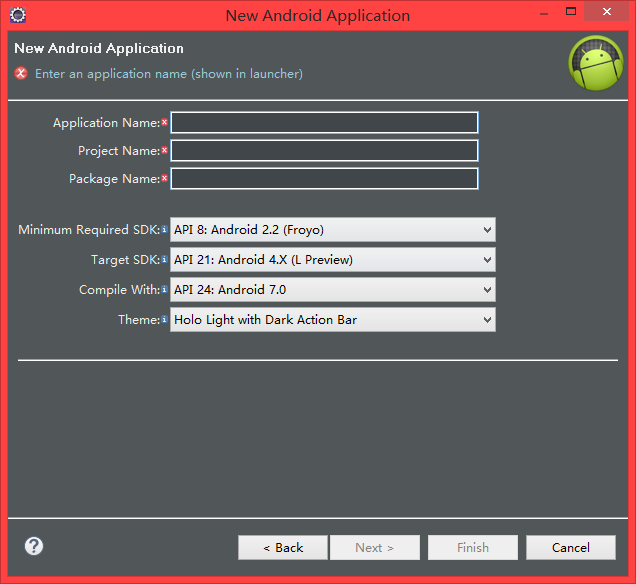
## 准备工作

### 资源文件下载

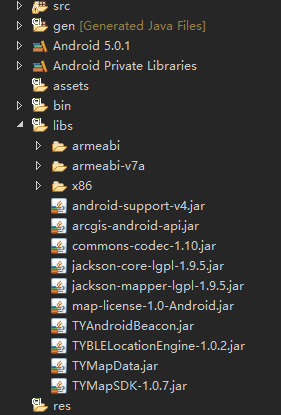
下载所需文件[<https://github.com/BrightBeacon/BrightIndoorSDK_Android>]

### 创建Android项目

1. 在Android Studio中创建项目，如下图所示：



1. 把libs和res目录下的资源文件（设施图标必要）拷贝到项目对应目录，如下图所示：



### 配置APP权限

在项目的AndroidManifest.xml文件中添加如下配置：

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

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

<uses-permission android:name="android.permission.BLUETOOTH\_ADMIN"/>

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

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

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

<uses-featureandroid:name="android.hardware.bluetooth\_le"android:required="true"/>

安卓6.0及后续的版本,需要添加以下权限,且权限还需要在使用时动态申请:

<uses-permission-sdk-23 android:name="android.permission.ACCESS\_COARSE\_LOCATION"/>

声明权限请求的标识;

public final static int APP\_PERMISSIONS\_REQUEST\_ACCESS\_COARSE\_LOCATION = 789;

权限申请代码;

if (ContextCompat.checkSelfPermission(this,

Manifest.permission.ACCESS\_COARSE\_LOCATION) != PackageManager.PERMISSION\_GRANTED) {

//请求权限

ActivityCompat.requestPermissions(this, new String[]{ Manifest.permission.ACCESS\_COARSE\_LOCATION },

APP\_PERMISSIONS\_REQUEST\_ACCESS\_COARSE\_LOCATION);

//判断是否需要说明申请该权限的原因

if(ActivityCompat.shouldShowRequestPermissionRationale(this,

Manifest.permission.ACCESS\_COARSE\_LOCATION)) {

Toast.makeText(this, "蓝牙扫描定位功能需要此权限!", Toast.LENGTH\_SHORT).show();

}

}

权限申请处理代码;

@Override

public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,

@NonNull int[] grantResults) {

super.onRequestPermissionsResult(requestCode, permissions, grantResults);

}

### 配置beacon扫描服务

在项目的AndroidManifest.xml文件中添加如下配置：

<service android:name="com.ty.locationengine.ibeacon.BeaconService"/>

# 地图显示

## 地图初始化

新建actvity类，按需引人地图、定位支持库:

static {

System.loadLibrary("TYMapSDK");

System.loadLibrary("TYLocationEngine");

}

在Activity布局文件中加入地图控件：

<com.ty.mapsdk.TYMapView

        android:id="@+id/map"

        android:layout\_width="match\_parent"

        android:layout\_height="match\_parent"/>

设置地图数据保存在SD卡的位置，默认TYData

TYMapEnvironment.initMapEnvironment();

TYMapEnvironment.setRootDirectoryForMapFiles(dir);

初始化地图监听、并加载地图数据,加载地图需要用到建筑标识buildingID，并传人授权appKey以验证地图使用权限

mapView = (TYMapView) findViewById(R.id.map);

//设置地图回调

mapView.addMapListener(this);

//预设楼层

mapView.setFloor("F1");

//加载地图资源

mapView.init(Constants.BUILDING\_ID,Constants.APP\_KEY);

//申请appKey:(http://open.brtbeacon.com)

## 地图回调

以上步骤完成后，地图资源准备就绪后回调如下方法；此处也可设置、获取楼层信息

@Override

public void mapViewDidLoad(TYMapView mapView, Error error) {

if(error == null){

List<TYMapInfo>mapInfos = mapView.allMapInfo();

//mapView.setFloor(mapInfos.get(0));

}else {

Utils.showToast(this, error.toString());

}

}

# 基础功能

## 地图弹窗

### 弹窗样式配置

新建文件callout\_style.xml到res文件夹下的xml文件夹

<resources>

<calloutViewStyle

backgroundColor="#ffffff"//弹窗背景颜色

backgroundAlpha="255"//弹窗背景透明度

frameColor="#66FFCC"//弹窗边框颜色

cornerCurve="10"//弹窗边框拐角度数

anchor="5"/> //弹窗锚 指的方向 【1：上，2:右上，3：右，4：右下, ....】

</resources>

### 弹窗回调

Activity实现TYMapView.TYMapViewListener接口 [[代码详见demo](https://github.com/BrightBeacon/BrightIndoorSDK_Android/blob/master/app/src/main/java/com/brtbeacon/indoor/ui/PopviewActivity.java)]

@Override

public void onClickAtPoint(TYMapView tyMapView, Point point) {

//点击地图上的点回调方法

}

@Override

public void onPoiSelected(TYMapView tyMapView,List<TYPoi> list) {

//选择poi回调方法

popview.setStyle(R.xml.callout\_style);

if(popview != null && popview.isShowing()) {

popview.hide();

}

if(list != null && list.size()>0) {

TYPoi poi = list.get(0);

Point position;

if(poi.getGeometry().getClass() == Polygon.class) {

position = GeometryEngine.getLabelPointForPolygon((Polygon)

poi.getGeometry(),TYMapEnvironment.defaultSpatialReference());

} else {

position = (Point)poi.getGeometry();

}

String title = poi.getName();

String detail = poi.getPoiID();

//设置弹出view的大小

popview.setMaxWidth(dip2px(PoiActivity.this,160));

popview.setMaxHeight(dip2px(PoiActivity.this,120));

popview.setContent(poiView(title, detail));

popview.show(position);//显示弹窗

}

}

@Override

public void onFinishLoadingFloor(TYMapView tyMapView, TYMapInfo tyMapInfo) {

//楼层加载完毕回调方法

}

@Override

public void mapViewDidZoomed(TYMapView tyMapView) {

//地图缩放回调方法

}

## POI操作

POI操作主要是操作地图上所有图标。比如获取地图所有POI点位，高亮展示POI点位。

### 获取单个POI

根据POI的对象ID获取单个POI信息：

TYPoipoi = map.searchPoiById("POI ID")

根据POI的对象名称获取单个POI信息：

TYPoi poi = map.searchPoiByName("POI 名称")

### 获取POI列表

根据POI对象的类型获取POI对象的列表：

List<TYPoi> pois = map.searchPoiByType(type);

根据当前位置以及范围距离获取POI对象的列表：

List<TYPoi> pois = map.searchNearPoiByLocation(location, distance);

获取指定楼层所有POI对象：

List<TYPoi> pois = map.getAllPoiByFloor("楼层号");

### 获取POI对象类型

获取当前楼层的所有POI对象的类型：

List<Integer> poiIds = map.getAllPoiTypeByCurrentFloor();

### 高亮显示POI

高亮显示指定过的POI对象：

map.highlightPoi("POI 对象");

map.highlightPois("POI对象集合");

//清理地图上所有高亮的POI状态：

map.clearSelection();

## 线路规划

### 设置起点

//清空起点图标

mapView.setStartSymbol(null);

pt1 =newTYLocalPoint(startPoint.getX(),startPoint.getY(),currentMapInfo.getFloorNumber());

TYPictureMarkerSymbol start = new TYPictureMarkerSymbol(getResources().getDrawable(R.drawable.qidian));

//设置起点图标的宽、高

start.setWidth(dip2px(PoiActivity.this,12));

start.setHeight(dip2px(PoiActivity.this,16));

//设置起点 位置

mapView.setRouteStart(pt1);

//设置起点位置的图标

mapView.setStartSymbol(start);

//显示起点图标在当前楼层地图上

mapView.showRouteStartSymbolOnCurrentFloor(pt1);

### 设置终点和跨层点

pt2 = newTYLocalPoint(endPoint.getX(),endPoint.getY(),currentMapInfo.getFloorNumber());

if(pt1.getFloor()!= pt2.getFloor()){

TYPictureMarkerSymbol pic\_floor = newTYPictureMarkerSymbol( getResources().getDrawable(R.drawable.green\_pushpin) );

pic\_floor.setWidth(dip2px(PoiActivity.this,12));

pic\_floor.setHeight(dip2px(PoiActivity.this,12));

mapView.setSwitchSymbol(pic\_floor);

}

//

routeManager.requestRoute(pt1, pt2);

TYPictureMarkerSymbol pic = newTYPictureMarkerSymbol(getResources().getDrawable(R.drawable.zhongdian));

mapView.setRouteEnd(pt2);

mapView.setEndSymbol(pic);

//显示终点图标在当前楼层地图上

mapView.showRouteEndSymbolOnCurrentFloor(pt2);

### 开启路径规划

//开始规划，并添加回调listenser

mapView.routeManager().addRouteManagerListener(this);

mapView.resetRouteLayer();

routeManager.requestRoute(startPoint, endPoint);

### 实现路径规划回调

@Override

public void didSolveRouteWithResult(TYOfflineRouteManagerty OfflineRouteManager, TYRouteResult tyRouteResult){

//设置线路

mapView.setRouteResult(tyRouteResult);//设置线路结果

mapView.showRouteResultOnCurrentFloor();//地图上显示线路

}

@Override

public void didFailSolveRouteWithError(TYOfflineRouteManagerty OfflineRouteManager,Exception e){

//线路规划错误

}

### 其他相关方法

1）楼层切换后，重新显示本层路段

mapView.showRouteResultOnCurrentFloor();

2）显示经过的路径

mapView.showPassedAndRemainingRouteResultOnCurrentFloor(localPoint);

3）计算路径偏航

routeResult.isDeviatingFromRoute(localPoint,偏航米数)

4）获得线段方向提示

TYRoutePart part = routeResult.getNearestRoutePart(localPoint);

if (part != null) {

List<TYDirectionalHint> hints = routeResult.getRouteDirectionalHint(part);

TYDirectionalHint hint = routeResult.getDirectionalHintForLocationFromHints( localPoint, hints );

}

## POI查询

### 基本查询

TYSearchAdapter searchAdapter = new TYSearchAdapter(buildingId);

List<PoiEntity> poiEntityList = searchAdapter.queryPoi("POI 名称关键字");

### 查询相关方法

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| queryPoi(String keyName) | keyName: 名称关键字 | 根据名称关键字查询 |
| queryPoi(String keyName, integer floorNum) | keyName: 名称关键字  floorNum: 楼层索引 | 根据名称关键字和楼层信息查询 |
| queryPoiByCategoryID(String cateID) | cateID: 类型ID | 根据建筑类型查询 |
| queryPoiByCategoryID(String cateID, Integer floorNum) | cateID: 类型ID  floorNum: 楼层索引 | 根据建筑类型和楼层信息查询 |

# 室内定位

## 添加蓝牙服务及相关权限

请参考地图集成,“配置APP权限”和“配置Beacon扫描服务”。

## 初始化定位管理器

定位管理器主要负责搜寻附近beacon设备计算出当前手机设备在地图上的位置,所以初始化它的时候可以通过BeaconRegion对象配置对beacon设备的过滤条件。(比如：UUID,MAJOR,MINOR的限制)

locationManager = new TYLocationManager(this, Constants.BUILDING\_ID, Constants.APP\_KEY);

locationManager.addLocationEngineListener(this);

BeaconRegion region = new BeaconRegion("demo",Constants.UUID,null,null);

locationManager.setBeaconRegion(Arrays.asList(new BeaconRegion[]{region}));

## 实现回调接口

@Override

public void didRangedBeacons(TYLocationManager arg0, List<TYBeacon> arg1) {

// Beacon扫描结果事件回调，返回符合扫描参数的所有Beacon

}

@Override

public void didRangedLocationBeacons(TYLocationManager arg0, List<TYPublicBeacon> arg1) {

// 定位Beacon扫描结果事件回调，返回符合扫描参数的定位Beacon，定位Beacon包含坐标信息。此方法可用于辅助巡检，以及基于定位beacon的相关触发事件。

}

@Override

public void didUpdateDeviceHeading(TYLocationManager arg0, double newHeading) {

// 设备方向改变事件回调。结合地图MapMode可以处理地图自动旋转，或箭头方向功能。

//mapView.setMapMode(TYMapViewMode.TYMapViewModeDefault);

//mapView.setMapMode(TYMapViewMode.TYMapViewModeFollowing);

Log.i(TAG,"地图初始北偏角："+mapView.building.getInitAngle()+"；当前设备北偏角：" + newHeading);

mapView.processDeviceRotation(newHeading);

}

@Override

public void didUpdateImmediateLocation(TYLocationManager arg0, TYLocalPoint arg1) {

// \* 位置更新事件回调，位置更新并返回新的位置结果。

// 与[TYLocationManager:didUpdateLocatin:]方法相近，此方法回调结果未融合计步器信息，灵敏度较高，适合用于行车场景下

}

@Override

public void didFailUpdateLocation(TYLocationManagertyLocationManager, Error error) {

if (error != null) Log.i(TAG,error.toString());

}

@Override

public void didUpdateLocation(TYLocationManager arg0, TYLocalPointnewLocalPoint) {

// 位置更新事件回调，位置更新并返回新的位置结果。

// 与[TYLocationManager:didUpdateImmediationLocation:]方法相近，此方法回调结果融合计步器信息，稳定性较好，适合用于步行场景下。

Log.i(TAG, newLocalPoint.getX()+" "+newLocalPoint.getY());

mapView.showLocation(newLocalPoint);

//mapView.showRemainingRouteResultOnCurrentFloor(newLocalPoint);

}

## 启动定位

开始定位:调用TYLocationManager对象的startUpdateLocation()。

TYLocationManager.startUpdateLocation();

停止定位:调用TYLocationManager对象的stopUpdateLocation()。

TYLocationManager.stopUpdateLocation();

# 技术支持

技术交流论坛：[bbs.brtbeacon.com](http://bbs.brtbeacon.com/)

官方网站：[www.brtbeacon.com](http://www.brtbeacon.com)

QQ技术群：127596680