|  |
| --- |
| BRGIHT BEACON |
| 智石科技室内地图 |
| IOS API开发指南 |
|  |
| **V2.2.2** |
| **2017/08/04** |

|  |
| --- |
| 智石说（北京）科技有限公司 |

目录

[一、简介 2](#_Toc482888346)

[1.1概述 2](#_Toc482888347)

[1.2获取授权 2](#_Toc482888348)

[二、地图集成 3](#_Toc482888349)

[1 集成开发环境 3](#_Toc482888350)

[1.1 资源文件下载 3](#_Toc482888351)

[1.2 安装地图环境 3](#_Toc482888352)

[2地图显示 3](#_Toc482888353)

[2.1显示地图 3](#_Toc482888354)

[2.2.地图回调 4](#_Toc482888355)

[三、基础功能 5](#_Toc482888356)

[1、绘制形状 5](#_Toc482888357)

[2、地图弹窗 6](#_Toc482888358)

[3、POI操作 8](#_Toc482888359)

[4、POI搜索 9](#_Toc482888360)

[5、路径规划 9](#_Toc482888361)

[5.1 定义路径表现符号 10](#_Toc482888362)

[5.2开启路径规划 11](#_Toc482888363)

[5.3路径规划回调 13](#_Toc482888364)

[5.4 其他相关方法 14](#_Toc482888365)

[四、室内定位 15](#_Toc482888366)

[1.定位环境准备 15](#_Toc482888367)

[1.1 获取授权 15](#_Toc482888368)

[1.2集成定位SDK 15](#_Toc482888369)

[1.3 示例工程定位体验 16](#_Toc482888370)

[2.结合地图SDK显示定位 16](#_Toc482888371)

[3.定位初始化设置 17](#_Toc482888372)

[4.位置更新事件回调 19](#_Toc482888373)

[五、技术支持 22](#_Toc482888374)

# 一、简介

1.1概述

智石科技室内地图 iOS API，是由 Object-C 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iOS 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/)）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。开发包最低兼容IOS7及其以上系统。

**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

1.2获取授权

为保证地图服务可正常使用，您需要注册登录[*智石开放平台*](http://open.brtbeacon.com)获取appKey和buildingID用于获取操作地图的权限。获取操作流程请参考[*智石开放平台教程*](http://help.brtbeacon.com/fun/help_detail?id=13&tid=144)。

# 二、地图集成

## 1 集成开发环境

### 1.1 资源文件下载

下载所需文件<https://github.com/BrightBeacon/BrightIndoorSDK_IOS>

### 1.2 安装地图环境

① 安装GIS环境

安装[AGSRuntimeSDKiOSv10.2.5.pkg](https://github.com/BrightBeacon/BrightIndoorSDK_IOS/blob/master/AGSRuntimeSDKiOSv10.2.5.pkg)文件

设置项目引用路径

Target->Building Setting->Framework Search Paths: $HOME/Library/SDKs/ArcGIS/iOS

② 获取GEOS开发库

运行（或直接引人）geos.xcodeproj生成libgeos.a库文件，然后在target的buiding setting配置库路径：Library Search Paths：libgeos.a目录

③ 配置项目编译参数

添加Building Setting的Other Linker Flags：

 -ObjC -framework ArcGIS -lc++ -l"geos" -l"z" -l"sqlite3"

④ 引入图标资源文件

添加resource相关图标文件，到你的工程（必须）

## 2地图显示

### 2.1显示地图

请复制你的建筑资源标识kBuildingID、用户kAppKey到工程中，或使用[示例工程资源](https://github.com/BrightBeacon/BrightIndoorSDK_IOS.git)演示参数。

1） 在xib或storyBoard界面上添加一个UIView，并修改类为TYMapview，关联到IBOutlet mapView

@property (weak, nonatomic) IBOutlet TYMapView \*mapView;

或直接代码初始化MapView

@property (strong, nonatomic) TYMapView \*mapview;

self.mapview = [[TYMapView alloc]initWithFrame:self.view.frame];

[self.view addSubview:self.mapview];

2）初始化地图运行环境

[TYMapEnvironment initMapEnvironment];

3）地图加载

kBuildingID 地图显示的目标建筑

kAppKey SDK的用户appKey

self.mapView.mapDelegate = self;

[self.mapView initMapViewWithBuilding:kBuildingId AppKey:kAppKey];

4）设置显示楼层

floor 目标楼层-1，1

- (void)setFloor:(NSString)floor;

### 2.2.地图回调

1）地图加载完成事件回调

- (void)TYMapViewDidLoad:(TYMapView \*)mapView withError:(NSError \*)error{

if (error) {

return;

}

//加载完成时，才能获取建筑信息、楼层信息

TYMapInfo \*info = mapView.allMapInfo.firstObject;

[self.mapView setFloor:info.floorNumber];

}

2）地图楼层加载完成回调

- (void)TYMapView:(TYMapView \*)mapView didFinishLoadingFloor:(TYMapInfo \*)mapInfo;

3) 地图点选事件回调

- (void)TYMapView:(TYMapView \*)mapView didClickAtPoint:(CGPoint)screen mapPoint:(AGSPoint \*)mappoint;

4) POI选中事件回调

- (void)TYMapView:(TYMapView \*)mapView PoiSelected:(NSArray \*)array;

# 三、基础功能

## 1、绘制形状

MapView是地图最基本的容器，它提供了一系列接口叠加不同的空间数据、漫游地图、显示信息等。 地图继承于AGSMapView，建议参考[ArcGIS相关文档](https://developers.arcgis.com/ios)。

MapView支持很多类型的图层，用于叠加在地图上展示动态、静态的Service数据。 其中AGSGraphicsLayer比较特殊，它是完全由客户端创建、更新和销毁，用来处理用户输入，展示位置标识、导航路径的利器。 AGSGraphicsLayer（图形图层）相当于一张画纸，AGSGraphics（空间要素）相当于画面上的房子、花园、马路什么的并可以用不同颜色。

AGSGraphics（空间要素）必须要有几何形状（AGSGeometry），表现符号（AGSSymbol）默认是简单样式，而属性（Attributes）是可选的。

AGSSymbol（表现符号）包括：点状、图片型、线状、面状、文字型、复合型。

以下示例代码以点状为例：

1）新建图形图层

AGSGraphicsLayer \*poiLayer = [AGSGraphicsLayer graphicsLayer];

[self.mapView addMapLayer:poiLayer];

2）创建点状符号：

AGSSimpleMarkerSymbol \*markerSymbol = [AGSSimpleMarkerSymbol simpleMarkerSymbolWithColor:[UIColor greenColor]];

markerSymbol.size = CGSizeMake(5, 5);

3）创建点状几何形状

AGSPoint \*poiCoord = [AGSPoint pointWithX:x y:y spatialReference:self.mapView.spatialReference];

（此处以可见区域中心点为例，地图加载完成之后可自动获取）

AGSPoint \*poiCoord = self.mapView.visibleAreaEnvelope.center;

4）组装要素点

AGSGraphic \*myGraphic = [AGSGraphic graphicWithGeometry:poiCoord symbol:markerSymbol attributes:nil];

5）添加要素至图形图层

[poiLayer addGraphic:myGraphic];

## 2、地图弹窗

1）默认弹窗展示

函数体：

mapView.callout.title = @"默认弹窗";

mapView.callout.detail = @"默认详情";

mapView.callout.delegate = self;

[mapView.callout showCalloutAt:mappoint screenOffset:CGPointZero animated:YES];

参数说明：mappoint  地图上的位置

             screenOffset  偏移量

               animated       是否开启动画效果

2）自定义弹窗View

- (UIView \*)customView:(TYPoi \*)poi {

UIView \*view = [[UIView alloc] initWithFrame:CGRectMake(0, 0, 200, 100)];

UILabel \*titleLabel = [[UILabel alloc] initWithFrame:CGRectMake(0, 0, 200, 25)];

titleLabel.text = @"自定义弹窗";

[view addSubview:titleLabel];

UILabel \*detailLabel = [[UILabel alloc] initWithFrame:CGRectMake(0, 25, 200, 25)];

detailLabel.text = poi.name;

[view addSubview:detailLabel];

}

//设置自定义View

mapView.callout.customView = [self customView:poi];

3）默认弹窗即将弹出回调，可对弹出框进行配置

函数体：

-(BOOL)callout:(AGSCallout\*)callout willShowForFeature:(id<AGSFeature>)feature layer:(AGSLayer<AGSHitTestable>\*)layer mapPoint:(AGSPoint \*)mapPoint;

参数说明：callout  展示的弹窗

               feature 弹窗的特征

               layer    特征所在的图层

               mapPoint 弹窗显示的坐标

4) 默认弹窗回调

-(void)calloutWillDismiss:(AGSCallout\*)callout {

}

-(void)calloutDidDismiss:(AGSCallout\*)callout {

}

## 3、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。

1）获取当前楼层下特定子层特定poiID的信息

函数体：

- (TYPoi \*)getPoiOnCurrentFloorWithPoiID:(NSString \*)pid layer:(POI\_LAYER)layer;

参数说明：pid   poiID

               layer 目标子层

2）高亮显示POI

函数体：

- (void)highlightPoi:(TYPoi \*)poi;

参数说明：poi 目标poi

注：目标poi至少包含poiID和layer信息，当前支持ROOM和FACILITY高亮

3）根据坐标x和y提取当前楼层的ROOM POI

函数体：

- (TYPoi \*)extractRoomPoiOnCurrentFloorWithX:(double)x Y:(double)y;

参数说明：x 坐标x    y 坐标y

## 4、POI搜索

POI搜索TYSearchAdapter通过关键字、POI类别搜索PoiEntity详细信息

1）通过关键字模糊、楼层搜索POI信息

函数体：

- (NSArray \*) queryPoi:(NSString \*)key;

- (NSArray \*)queryPoi:(NSString \*)name andFloor:(int)floorNumber;

参数说明：key   名字关键字

             floorNumber目标楼层

2）通过类别id、楼层搜索POI信息

函数体：

- (NSArray \*)queryPoiByCategoryID:(NSString \*)cids;

- (NSArray \*)queryPoiByCategoryID:(NSString \*)cids andFloor:(int)floorNumber;

参数说明：cids 类别列表，多个以逗号隔开

             floorNumber目标楼层

## 5、路径规划

路径规划使用了开源库GEOS，SDK中已经集成在TYOfflineRouteManager路径管理器类里。路径规划需要自定义：路径起点、路径终点、路径楼梯出入口的表现符号（AGSSymbol）。

### 5.1 定义路径表现符号

1）设置导航线的起点符号

函数体：

- (void)setRouteStartSymbol:(AGSPictureMarkerSymbol \*)symbol;

参数说明：symbol 起点标识符号

2）设置导航线的终点符号

函数体：

- (void)setRouteEndSymbol:(AGSPictureMarkerSymbol \*)symbol;

参数说明：symbol 起点标识符号

3）设置跨层导航切换点符号

函数体：

- (void)setRouteSwitchSymbol:(AGSPictureMarkerSymbol \*)symbol;

参数说明：symbol 起点标识符号

示例代码：

AGSPictureMarkerSymbol \*startSymbol = [AGSPictureMarkerSymbol pictureMarkerSymbolWithImageNamed:@"start"];

startSymbol.offset = CGPointMake(0, 22);

//

AGSPictureMarkerSymbol \*endSymbol = [AGSPictureMarkerSymbol pictureMarkerSymbolWithImageNamed:@"end"];

endSymbol.offset = CGPointMake(0, 22);

//

AGSPictureMarkerSymbol \*switchSymbol = [AGSPictureMarkerSymbol pictureMarkerSymbolWithImageNamed:@"nav\_exit"];

//

markerSymbol = [AGSSimpleMarkerSymbol simpleMarkerSymbolWithColor:[UIColor greenColor]];

markerSymbol.size = CGSizeMake(5, 5);

//

[self.mapView setRouteStartSymbol:startSymbol];

[self.mapView setRouteEndSymbol:endSymbol];

[self.mapView setRouteSwitchSymbol:switchSymbol];

### 5.2开启路径规划

1）离线路径管理类的静态实例化方法

函数体：

+ (TYOfflineRouteManager \*)routeManagerWithBuilding:(TYBuilding \*)building MapInfos:(NSArray \*)mapInfoArray;

参数说明：building     目标建筑

               mapInfoArray 目标建筑的所有楼层信息

2）请求路径规划，在代理方法获取规划结果

函数体：

- (void)requestRouteWithStart:(TYLocalPoint \*)start End:(TYLocalPoint \*)end;

参数说明：start 路径规划起点

               end   路径规划终点

3）设置导航结果

函数体：

- (void)setRouteResult:(TYRouteResult \*)result;

参数说明：result      导航结果

4）设置导航起点

函数体：

- (void)setRouteStart:(TYLocalPoint \*)start;

参数说明：start 导航起点

5）设置导航终点

函数体：

- (void)setRouteEnd:(TYLocalPoint \*)end;

参数说明：end  导航终点

6）在地图显示当前楼层的导航路径

函数体：

- (void)showRouteResultOnCurrentFloor;

示例代码：

self.mapView.routeManager.delegate = self;

endLocalPoint = [TYLocalPoint pointWithX:274 Y:363 Floor:self.mapView.currentMapInfo.floorNumber];

[self.mapView.routeManager requestRouteWithStart:startLocalPoint End:endLocalPoint];

### 5.3路径规划回调

1）路径规划失败回调方法

函数体：

- (void)offlineRouteManager:(TYOfflineRouteManager \*)routeManager didFailSolveRouteWithError:(NSError \*)error;

参数说明：routeManager 离线路径管理实例

               error        错误信息

2）路径规划成功回调方法

函数体：

- (void)offlineRouteManager:(TYOfflineRouteManager \*)routeManager didSolveRouteWithResult:(TYRouteResult \*)rs;

参数说明：routeManager 离线路径管理实例

               error        错误信息

路径规划成功示例代码：

- (void)offlineRouteManager:(TYOfflineRouteManager \*)routeManager didSolveRouteWithResult:(TYRouteResult \*)rs

{

[self.mapView setRouteResult:rs];

[self.mapView setRouteStart:startLocalPoint];

[self.mapView setRouteEnd:endLocalPoint];

[self.mapView showRouteResultOnCurrentFloor];

}

### 5.4 其他相关方法

1）显示本层路径线段

- (void)TYMapView:(TYMapView \*)mapView didFinishLoadingFloor:(TYMapInfo \*)mapInfo

{

//显示当前楼层导航信息

if (isRouting) {

[self.mapView showRouteResultOnCurrentFloor];

}

}

2）显示定位状态下已经过路径

- (void)TYMapView:(TYMapView \*)mapView didClickAtPoint:(CGPoint)screen mapPoint:(AGSPoint \*)mappoint {

if (isRouting) {

TYLocalPoint \*localPoint = [TYLocalPoint pointWithX:mappoint.x Y:mappoint.y Floor:mapView.currentMapInfo.floorNumber];

//显示路过和余下线段

[mapView showPassedAndRemainingRouteResultOnCurrentFloor:localPoint];

}

3）计算路径偏航

Bool isDeviating = [routeResult isDeviatingFromRoute:localPoint WithThrehold:偏航距离]

4) 获得点位最近的线段提示信息

TYRoutePart \*part = [routeResult getNearestRoutePart:localPoint];

NSArray \*routeGuides = [routeResult getRouteDirectionalHint:part];

if (routeGuides.count) {

TYDirectionalHint \*hint = [routeResult getDirectionHintForLocation:localPoint FromHints:routeGuides];

}

# 四、室内定位

## 1.定位环境准备

### 1.1 获取授权

为保证定位服务可正常使用，您需要注册登录[*智石开放平台*](http://open.brtbeacon.com)获取appKey和buildingID用于获取定位数据的权限。

### 1.2集成定位SDK

* Building Phrases添加TYLocationEngine.framework、TYMapData.framework、libz.tbd、libsqlite3.0.tbd
* 检查Building Setting的Framework Search Paths、Header Search Paths是否包含：$(inherited)或framework所在目录
* 添加Building Setting的Other Linker Flags：-ObjC -framework "TYLocationEngine" -framework "TYMapData"

### 1.3 示例工程定位体验

室内定位需要使用与地图数据配套的iBeacon设备部署方案，以示例地图资源为例，需要准备5个iBeacon设备；配置参数列表如下：

|  |  |  |  |
| --- | --- | --- | --- |
| No. | UUID | Major | Minor |
| 区域1 | FDA50693-A4E2-4FB1-AFCF-C6EB07647825 | 10046 | 11048 |
| 区域2 | FDA50693-A4E2-4FB1-AFCF-C6EB07647825 | 10046 | 11049 |
| 区域3 | FDA50693-A4E2-4FB1-AFCF-C6EB07647825 | 10046 | 11050 |
| 区域4 | FDA50693-A4E2-4FB1-AFCF-C6EB07647825 | 10046 | 11053 |
| 区域5 | FDA50693-A4E2-4FB1-AFCF-C6EB07647825 | 10046 | 11055 |

## 2.结合地图SDK显示定位

MapView是地图最基本的容器，它提供了一系列接口叠加不同的空间数据、漫游地图、显示信息等。 地图继承于AGSMapView，你也可以自行参考[ArcGIS相关文档](https://developers.arcgis.com/ios)MapView是地图最基本的容器，它提供了一系列接口叠加不同的空间数据、漫游地图、显示信息等。 地图继承于AGSMapView，你也可以自行参考[ArcGIS相关文档](https://developers.arcgis.com/ios)

示例代码：

AGSPictureMarkerSymbol \*locSymbol = [AGSPictureMarkerSymbol pictureMarkerSymbolWithImage:[UIImage imageNamed:@"locationArrow"]];

[self.mapView setLocationSymbol:locSymbol];

## 3.定位初始化设置

1）使用TYLocationEngine实现室内地图定位，导入头文件

#import <TYLocationEngine/TYLocationEngine.h>

2）设置定位文件根目录（注：rootDir为定位数据目录）

[TYBLEEnvironment setRootDirectoryForFiles:rootDir]

3）初始化定位引擎

函数体：

- (id)initWithBuilding:(NSString \*)buildingID appKey:(NSString \*)aKey;

参数说明：building 目标建筑信息

4）设置beacon信号阈值

函数体：

- (void)setRssiThreshold:(int)threshold;

参数说明：threshold 低于此信号阈值将忽略beacon信号

5）设置用于定位的Beacon参数(可选)

函数体：

- (void)setBeaconRegion:(CLBeaconRegion \*)region;

参数说明：region Beacon Region参数

注：region初始化示例：

[[CLBeaconRegion alloc] initWithProximityUUID:[[NSUUID alloc] initWithUUIDString:@"FDA50693-A4E2-4FB1-AFCF-C6EB07647825"] identifier:@"identifier"]

6）开启定位引擎

- (void)startUpdateLocation;

7）关闭定位引擎

- (void)stopUpdateLocation;

8）在地图上显示定位结果在地图上显示定位结果

函数体：

- (void)showLocation:(TYLocalPoint \*)location;

参数说明：location 定位结果坐标点

注：该方法通常在‘位置更新并返回新的位置结果’回调中调用

9）在地图上移除定位结果在地图上移除定位结果

- (void)removeLocation;

示例代码：

if (!self.locationManager) {

self.locationManager = [[TYLocationManager alloc] initWithBuilding:kBuildingId appKey:kAppKey];

NSUUID \*uuid = [[NSUUID alloc] initWithUUIDString:kUUID];

CLBeaconRegion \*region = [[CLBeaconRegion alloc] initWithProximityUUID:uuid identifier:@"id"];

[self.locationManager setBeaconRegion:region];

self.locationManager.delegate = self;

}

[self.locationManager startUpdateLocation];

## 4.位置更新事件回调

1）位置更新并返回新的位置结果

函数体：

- (void)TYLocationManager:(TYLocationManager \*)manager didUpdateLocation:(TYLocalPoint \*)newLocation;

参数说明：manager     定位引擎实例

               newLocation 新的位置结果

2）实时返回位置结果

/\*\*

\* 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

\*

\* @param manager 定位引擎实例

\* @param newLocation 新的位置结果

\*/

- (void)TYLocationManager:(TYLocationManager \*)manager didUpdateImmediateLocation:(TYLocalPoint \*)newImmediateLocation {

//NSLog(@"您的位置：%@",newImmediateLocation);

//[self.mapView showLocation:newImmediateLocation];

}

3）位置更新失败事件回调位置更新失败事件回调

函数体：

- (void)TYLocationManager:(TYLocationManager \*)manager didFailUpdateLocation:(NSError \*)error {

NSLog(@"检查设置");

}

参数说明：manager 定位引擎实例

4）设备方向回调

/\*\*

\* 设备方向改变事件回调。结合地图Api可以处理地图自动旋转，以及方向箭头等功能。

\*

\* @param manager 定位引擎实例

\* @param newHeading 新的设备方向结果

\*/

- (void)TYLocationManager:(TYLocationManager \*)manager didUpdateDeviceHeading:(double)newHeading {

NSLog(@"地图初始北偏角：%f，当前设备北偏角：%f",self.mapView.building.initAngle,newHeading);

//初始北偏角内部已经处理

[self.mapView processDeviceRotation:newHeading];

}

5）扫描到iBeacon设备回调

/\*\*

\* Beacon扫描结果事件回调，返回符合扫描参数的所有Beacon

\*

\* @param manager 定位引擎实例

\* @param beacons Beacon数组，[TYBeacon]

\*/

- (void)TYLocationManager:(TYLocationManager \*)manager didRangedBeacons:(NSArray \*)beacons {

// NSLog(@"all beacons find:%@",beacons);

}

/\*\*

\* 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。

\*

\* @param manager 定位引擎实例

\* @param beacons 定位Beacon数组，[TYPublicBeacon]

\*/

- (void)TYLocationManager:(TYLocationManager \*)manager didRangedLocationBeacons:(NSArray \*)beacons {

NSLog(@"location beacons find:%d",(int)beacons.count);

}

# 五、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