iOS SDK 开发指南

目录

[简介 5](#_Toc430268938)

[关于HDmap iOS SDK 5](#_Toc430268939)

[HDmap iOS SDK功能 5](#_Toc430268940)

[适用对象 5](#_Toc430268941)

[功能概要 5](#_Toc430268942)

[兼容性 6](#_Toc430268943)

[注意事项 6](#_Toc430268944)

[配置开发环境 6](#_Toc430268945)

[基础地图 7](#_Toc430268946)

[简介 7](#_Toc430268947)

[图源选择及加载 7](#_Toc430268948)

[地图标注 9](#_Toc430268949)

[图片 9](#_Toc430268950)

[几何图形 10](#_Toc430268951)

[折线 10](#_Toc430268952)

[多边形 10](#_Toc430268953)

[圆 10](#_Toc430268954)

[检索功能 11](#_Toc430268955)

[简介 11](#_Toc430268956)

[POI检索 11](#_Toc430268957)

[城市检索 12](#_Toc430268958)

[周边检索 12](#_Toc430268959)

[区域检索 12](#_Toc430268960)

[检索错误 13](#_Toc430268961)

[地理编码 13](#_Toc430268962)

[正向地理编码 13](#_Toc430268963)

[逆地理编码 13](#_Toc430268964)

[行政区划检索 13](#_Toc430268965)

[行政区划地理范围 13](#_Toc430268966)

[行政区划编码 13](#_Toc430268967)

[经纬度点所在行政区划搜索 14](#_Toc430268968)

[工具 14](#_Toc430268969)

[简介 14](#_Toc430268970)

[罗盘 14](#_Toc430268971)

[比例尺 15](#_Toc430268972)

[定位功能 15](#_Toc430268973)

[简介 15](#_Toc430268974)

[获取位置信息 15](#_Toc430268975)

[展示定位信息 15](#_Toc430268976)

[事件监听 15](#_Toc430268977)

[地图类 16](#_Toc430268978)

[地图单击事件 16](#_Toc430268979)

[地图双击事件 16](#_Toc430268980)

[地图长按事件 17](#_Toc430268981)

[搜索类 17](#_Toc430268982)

[POI搜索 17](#_Toc430268983)

[行政区划搜索 17](#_Toc430268984)

[地理编码 19](#_Toc430268985)

# 简介

## 关于HDmap iOS SDK

iOS SDK v1.0是一套基于armv7、armv7s、arm64处理器设备的应用程序接口，不仅提供构建地图的基本接口，还提供POI搜索、地理编码、定位、本地覆盖物绘制等接口。

您可以使用iOS SDK v1.0开发适用于iOS移动设备的地图应用，通过接口，您可以轻松访问HDmap服务和数据，构建功能丰富、交互性强的地图应用程序。

### HDmap iOS SDK功能

* 地图：提供地图展示和地图操作功能；
* POI检索：支持周边检索、区域检索和城市内兴趣点检索；
* 地理编码：提供经纬度和地址信息相互转化的功能接口；
* 覆盖物图层：支持在地图上添加覆盖物（标注、几何图形等），展示更丰富的LBS信息；
* 定位：获取当前位置信息，并在地图上展示（支持普通、跟随、罗盘三种模式）；
* LBS云检索：支持查询存储在LBS云内的自有数据；

## 适用对象

本套SDK接口是提供给那些具有一定iOS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，都可以反馈给我们。

## 功能概要

SDK按功能可分为：基础地图、检索功能、定位功能、计算工具几个部。

* 基础地图：包括基本矢量地图、卫星图、各种地图覆盖物（POI搜索结果图层、自定义图层、几何图形绘制图层、地形图图层、热力图等），此外还包括各种与地图相关的操作和事件监听；
* 检索功能：包括POI检索（周边、区域、城市内），Place详情检索，，地理编码/反地理编码等；
* 定位功能：获取当前位置信息；

## 兼容性

HDmap iOS SDK支持iOS 5.0及以上操作系统，支持armv7、armv7s、arm64（自v2.5.0版本）处理器。

# 注意事项

HDmap iOS SDK v1.0是用Objective-C语言编写的，为证明该产品良好的Swift兼容性，附赠的演示程序HDMapDemo使用Swift实现。开发者可以在Objective-C和Swift中自由选择，二者对SDK接口的调用逻辑是一样的， 因此HDMapDemo中的示例代码对Objective-C开发者同样具有参考价值。

# 配置开发环境

所有必需的SDK文件都被包含在HDMap.framework中，在工程中引入SDK的步骤：

1. Build Phases->Copy Files->Destination选择Frameworks->+(Add items)->Add other->在Finder中选择HDMap.framework->Open，模拟器版本请把文件拷贝到运行目录下

2. Build Phases->Link Binary With Libraries->+(Add items)-> Add other->在Finder中选择HDMap.framework->Open

3. HDMap.framework中的HDMap.bundle拖拽到你的project navigatior中

4. Build Settings->Search Paths->Header Search Paths->+->输入(SDK所在路径)/HDMap.framework/Headers

提示：以上配置方法可以参考HDMapDemo工程

# 基础地图

## 简介

开发者可利用SDK提供的接口，使用服务端已部署的数据。目前地图等级一般为18级，根据图源的不同，可能包含不同的信息。

地图支持多点触摸、双击放大、拖拽等手势操作，支持相应的控制接口来开启/关闭这些手势操作；此外，在该版本地图对象实现了多实例特性，即开发者可以在一个页面中建立多个地图对象，并且针对这些对象分别操作且不会产生相互干扰。

地图上自定义的标注点和覆盖物我们统称为地图覆盖物。地图覆盖物的设计遵循数据与View分离的原则。

SDK支持画折线、圆、多边形（包括凹凸两种）、图片图层和自定义覆盖物。

## 图源选择及加载

要使地图视图（HDMMapView）能获取和显示地图，需要把HDMMapView和一个/多个图源（HDMSource）绑定，您需要先获得一个取图服务的已知url，以及一个备选url（如没有，可以和第一个url相同），并用它们创建并初始化一个图源对象HDMSource。图源对象和HDMMapView的绑定可以发生在MapView的初始化，也可以之后动态绑定。

以下是几个重要的使用场景：

1.创建并初始化图源，并用一个图源初始化HDMMapView：

//设置取图图层

land\_tilesource = HDMSource(mapID: "http://190.0.0.36/v1/rastertiles/14Q2\_map", reserveMapID: "http://190.0.0.36/v1/rastertiles/land\_map14Q2")

land\_tilesource!.minZoom = 1

land\_tilesource!.maxZoom = 17

land\_tilesource!.customTileSideLength(512)

mix\_tilesource = HDMSource(mapID: "http://190.0.0.36/v1/rastertiles/mix\_map", reserveMapID: "http://190.0.0.36/v1/rastertiles/mix\_map")

mix\_tilesource!.minZoom = 1

mix\_tilesource!.maxZoom = 17

mix\_tilesource!.customTileSideLength(512)

sea\_tilesource = HDMSource(mapID: "http://190.0.0.36/v1/rastertiles/sea\_map", reserveMapID: "http://190.0.0.36/v1/rastertiles/sea\_map")

sea\_tilesource!.minZoom = 1

sea\_tilesource!.maxZoom = 17

sea\_tilesource!.customTileSideLength(512)

//初始化MapView

self.mapView = HDMMapView(frame: self.view.bounds, andTilesource: mix\_tilesource)

2.图源的动态绑定，并隐藏未被选中的图源：

mapView.addTileSource(land\_tilesource)

mapView.setHidden(true, forTileSource: land\_tilesource)

mapView.addTileSource(sea\_tilesource)

mapView.setHidden(true, forTileSource: sea\_tilesource)

3.当前显示图源的切换：

注意：您可能需要在切换图源后清除瓦片缓存，为了节省流量，请实际测试后决定是否需要这样做。

IBAction func OnLandClick(sender: UIButton) {

mapView.setHidden(false, forTileSource: land\_tilesource)

mapView.setHidden(true, forTileSource: mix\_tilesource)

mapView.setHidden(true, forTileSource: sea\_tilesource)

mapView.removeAllCachedImages()

tilesource\_change\_view.hidden = true

}

4.删除不再使用的图源：

使用HDMMapView的removeTileSource即可。

## 地图标注

SDK提供四种类型的地图标注：自定义图片，圆形，折现，多边形。您可以使用它们在地图上添加您的自定义图形显示。HDMapDemo中提供了全部类型标注的使用示例，您可以使用“anno”关键字搜索（anno即annotation的简写）。

## 图片

图片标注的添加步骤：

1. 用mapview，地理坐标，标题初始化。地理坐标可以用屏幕坐标通过mapview的计算方法得到
2. 根据需要添加用户信息和子标题，决定是否显示弹出气泡
3. 指定图片图层
4. 调用addAnnotation绑定到MapView

func singleTapOnMap(map: HDMMapView!, at point: CGPoint) {

if (Parameters.AnnotationState.AddMarker == Parameters.Instance().annostate) {

var anno = HDMAnnotation(mapView: viewcontroller.mapView, coordinate: viewcontroller.mapView.pixelToCoordinate(point), andTitle: "图片标注")

anno.userInfo = "图片标注用户信息"

anno.subtitle = "子标题"

anno.layer = HDMMarker(UIImage: UIImage(named: "TrackingDot.png"))

anno.layer.canShowCallout = true

viewcontroller.mapView.addAnnotation(anno)

可以使用removeAnnotation/removeAnnotations/removeAllAnnotations接口删除标注。

## 几何图形

### 折线

折线标注的添加方法如下图，注意初始化标注使用的points参数为CLLocation数组类型

var point\_list:[CGPoint] = Parameters.Instance().points

var hdm\_point\_list:[CLLocation] = []

for p in point\_list {

hdm\_point\_list.append(CLLocation(latitude: Double(p.y.native), longitude: Double(p.x.native)))

}

pl\_annotation = HDMPolylineAnnotation(mapView: mapView, points: hdm\_point\_list)

mapView.addAnnotation(pl\_annotation)

### 多边形

和折线的添加方法基本相同，例外是您可以自定义多边形标注的填充色和透明度

pg\_annotation!.fillColor = UIColor(red: 0, green: 0.2, blue: 0.5, alpha: 0.3)

### 圆

图片标注的添加步骤：

1.用mapview，圆心地理坐标，半径初始化。地理坐标可以用屏幕坐标通过mapview的计算方法得到，半径单位为米

2.调用addAnnotation绑定到MapView

var anno = HDMCircleAnnotation(mapView: viewcontroller.mapView, centerCoordinate: viewcontroller.mapView.pixelToCoordinate(point), radiusInMeters: 1000)

viewcontroller.mapView.addAnnotation(anno)

Parameters.Instance().annostate = Parameters.AnnotationState.None

# 检索功能

## 简介

SDK提供的检索服务包括以下功能模块：POI检索，地理编码，反地理编码，行政区划检索。

相关的代码示例可以在SearchViewController.swift中找到。

首先，要获得检索结果，需要定义您的回调代理：

class SearchViewController: UIViewController, HDMSearchDelegate, HDMDistrictSearchDelegate,HDMGeoCodeSearchDelegate, UITableViewDataSource, UITableViewDelegate{

## POI检索

POI（Point of Interest），中文可以翻译为“兴趣点”。在地理信息系统中，一个POI可以是一栋房子、一个商铺、一个邮筒、一个公交站等。

注意：POI搜索参数含有type属性，0为海事地图数据搜索，1为陆地地图数据搜索。请您根据自己的业务需要配置该属性。

POI检索结果回调处理：

func onGetSearchResult(searcher : HDMSearch!, result:HDMPoiSearchResult!, errorCode:NSNumber!) {

if (nil == result) {

NSLog("onGetSearchResult failed:\(errorCode)")

return

}

NSLog("onGetSearchResult, count:\(result.totalPoiNum)")

var log = "\(result.status);\(result.message);"

if (0 == result.status) {

for v in result.poiInfoList {

log += "\(v.name),\(v.telephone)," + v.address

log += ",\(v.location.latitude),\(v.location.longitude);"

}

}

NSLog(log)

}

### 城市检索

var searcher = HDMSearch(baseUrl: "http://190.0.0.36:8080")

searcher.delegate = self

var option = HDMCitySearchOption()

option.cityName = “大连”

option.keyword = “海事”

option.pageSize = 10

option.pageIndex = 0

option.searchType = 0

searcher.poiSearchInCity(option)

### 周边检索

var searcher = HDMSearch(baseUrl: "http://190.0.0.36:8080")

searcher.delegate = self

var option = HDMNearbySearchOption()

option.keyword = “大连港”

option.location.latitude = 38.9123445

option.location.longitude = 121.3234223

option.radius = 500000

option.pageSize = 10

option.pageIndex = 0

option.searchType = 1

searcher.poiSearchNearby(option)

### 区域检索

var searcher = HDMSearch(baseUrl: "http://190.0.0.36:8080")

searcher.delegate = self

var option = HDMBoundSearchOption()

option.keyword = “电影”

option.rightTop.latitude = (param[2] as NSString).doubleValue

option.rightTop.longitude = (param[3] as NSString).doubleValue

option.leftBottom.latitude = (param[4] as NSString).doubleValue

option.leftBottom.longitude = (param[5] as NSString).doubleValue

option.pageSize = 10

option.pageIndex = 0

option.searchType = 0

searcher.poiSearchInBounds(option)

### 检索错误

检索错误时，检索结果的status字段值不为0，message字段携带错误提示

## 地理编码

### 正向地理编码

var searcher = HDMGeoCodeSearch(baseUrl: "http://190.0.0.36:8080")

searcher.delegate = self

var option = HDMGeoCodeLocationSearchOption()

option.address = "中山路"

option.cityName = "大连"

searcher.geoCode(option)

### 逆地理编码

var searcher = HDMGeoCodeSearch(baseUrl: "http://190.0.0.36:8080")

searcher.delegate = self

var option = HDMReverseGeoCodeLocationSearchOption()

option.location.latitude = 38.234223

option.location.longitude = 121.987456

searcher.reverseGeoCode(option)

## 行政区划检索

### 行政区划地理范围

var searcher = HDMDistrictSearch(baseUrl: "http://190.0.0.36:8080")

searcher.delegate = self

var option = HDMDistrictRegionSearchOption()

option.districtRegion = 210200

searcher.districtSearchByRegion(option)

### 行政区划编码

var searcher = HDMDistrictSearch(baseUrl: "http://190.0.0.36:8080")

searcher.delegate = self

var option = HDMDistrictNameSearchOption()

option.districtName = "大连"

searcher.districtSearchByName(option)

### 经纬度点所在行政区划搜索

var searcher = HDMDistrictSearch(baseUrl: "http://190.0.0.36:8080")

searcher.delegate = self

var option = HDMDistrictLocationSearchOption()

option.districtLocation.latitude = 38.9273482

option.districtLocation.longitude = 121.2342385

searcher.districtSearchByLocation(option)

# 工具

## 简介

地图SDK目前提供的工具有：罗盘，比例尺功能，可以开启和关闭。

## 罗盘

罗盘的开启关闭不由用户直接控制，用户在界面上添加HDMUserTrackingBarButtonItem后，当定位跟随模式为follow bearing时，罗盘自动显示，其他时候关闭。

HDMUserTrackingBarButtonItem的添加方法：

1. 修改AppDelegate，设置navigation controller，设置rootViewController为HDMMapView所属的视图控制器对象，本例中为ViewController类的静态实例：

func application(application: UIApplication, didFinishLaunchingWithOptions launchOptions: [NSObject: AnyObject]?) -> Bool {

// Override point for customization after application launch.

self.window = UIWindow(frame: UIScreen.mainScreen().bounds)

self.window?.rootViewController = UINavigationController(rootViewController: ViewController.instance!)

return true

}

1. 在视图控制器中绑定HDMUserTrackingBarButtonItem，绑定时机用户可以自行安排，例子中为viewDidAppear：

self.navigationItem.rightBarButtonItem = HDMUserTrackingBarButtonItem(mapView: mapView)

self.navigationItem.rightBarButtonItem!.tintColor = tint\_color

## 比例尺

比例尺的开关由用户直接控制，可以设置线段颜色，字体颜色，线段粗细，字体大小，锚点

//锚点为相对于左下角的位置

var anchorpoint:CGPoint = CGPoint(x:10, y:50)

mapView?.ShowScaleBar(UIKit.UIColor.blackColor(), fontColor: UIKit.UIColor.blackColor(), lineWidth: 2, fontSize: 12, anchorPoint: anchorpoint)

# 定位功能

## 简介

由于系统原因，iOS不允许使用第三方定位，因此地图SDK中的定位方法，本质上是对原生定位的二次封装。通过封装，开发者可更便捷的使用。此外，地图SDK中还提供了相应的定位图层（支持定位三态效果），帮助开发者显示当前位置信息。

## 获取位置信息

可以通过HDMMapView类的centerCoordinate属性获取地图中心点坐标，或userLocation属性访问用户定位标注

## 展示定位信息

可以通过HDMMapView类的showUserlocation属性开关定位显示，或setUserLocation方法调整定位显示模式

# 事件监听

Android SDK提供了丰富的回调接口，供用户对于不同的事件进行处理。

## 地图类

创建HDMMapViewDelegate协议的子类实体，根据业务需要实现其接口。在监听事件之前用该实体为HDMMapView的delegate属性赋值。

下面列举三种事件的监听，其他的事件监听请参考HDMMapViewDelegate.h文件和HDMapDemo程序示例代码

### 地图单击事件

下面的单击事件处理中，判断了业务数据的标注操作状态，以决定在单击的位置添加图片/圆形标注/不添加标注在屏幕上显示坐标信息

func singleTapOnMap(map: HDMMapView!, at point: CGPoint) {

if (Parameters.AnnotationState.AddMarker == Parameters.Instance().annostate) {

var anno = HDMAnnotation(mapView: viewcontroller.mapView, coordinate: viewcontroller.mapView.pixelToCoordinate(point), andTitle: "图片标注")

anno.userInfo = "图片标注用户信息"

anno.subtitle = "子标题"

anno.layer = HDMMarker(UIImage: UIImage(named: "TrackingDot.png"))

anno.layer.canShowCallout = true

viewcontroller.mapView.addAnnotation(anno)

Parameters.Instance().annostate = Parameters.AnnotationState.None

return

} else if (Parameters.AnnotationState.AddCircle == Parameters.Instance().annostate){

var anno = HDMCircleAnnotation(mapView: viewcontroller.mapView, centerCoordinate: viewcontroller.mapView.pixelToCoordinate(point), radiusInMeters: 1000)

viewcontroller.mapView.addAnnotation(anno)

Parameters.Instance().annostate = Parameters.AnnotationState.None

}

if (Parameters.Instance().single\_click\_userconfig == 0) {

viewcontroller.bt\_single\_click.setTitle("地图\(PointString(point))被点击\(TimeString())", forState: UIControlState.Normal)

viewcontroller.bt\_single\_click.hidden = false

}

}

### 地图双击事件

func doubleTapOnMap(map: HDMMapView!, at point: CGPoint) {

if (Parameters.Instance().double\_click\_userconfig == 0) {

viewcontroller.bt\_double\_click.setTitle("地图\(PointString(point))被双击\(TimeString())", forState: UIControlState.Normal)

viewcontroller.bt\_double\_click.hidden = false

}

}

### 地图长按事件

func longPressOnMap(map: HDMMapView!, at point: CGPoint) {

if (Parameters.Instance().double\_click\_userconfig == 0) {

viewcontroller.bt\_long\_press.setTitle("地图\(PointString(point))被长按\(TimeString())", forState: UIControlState.Normal)

viewcontroller.bt\_long\_press.hidden = false

}

}

## 搜索类

### POI搜索

#### 按城市搜索

见[POI检索](#_POI检索)一节

#### 按区域搜索

见[POI检索](#_POI检索)一节

#### 按范围搜索

见[POI检索](#_POI检索)一节

### 行政区划搜索

行政区划地理范围的多边形数据处理较为特殊，请重点理解。

#### 经纬度点所在行政区划搜索

func onGetDistrictInfoSearchResult(searcher: HDMDistrictSearch!, result searchResult: HDMDistrictInfoSearchResult!, errorCode: HDMDistrictSearchErrorCode) {

if (nil == searchResult) {

NSLog("onGetDistrictGeoSearchResult failed:\(errorCode)")

return

}

var upload\_text = "[\(search\_list[searcher.hashValue]!.req!)]\r\n"

upload\_text += "\(searchResult.status);\(searchResult.message);"

if (0 == searchResult.status) {

for district\_list in searchResult.alldistrictInfoList {

for district in district\_list as! [HDMDistrictInfo] {

upload\_text += "\(district.district\_name),\(district.district\_code),\(district.district\_class);"

}

}

}

}

#### 行政区划编码搜索

[同经纬度点所在行政区划搜索](#_经纬度点所在行政区划搜索)

#### 行政区划地理范围搜索

geometry\_coord属性的类型是三维double数组，第三维是两个double值的定长数组，代表一个地理点

func onGetDistrictGeoSearchResult(searcher: HDMDistrictSearch!, result searchResult: HDMDistrictGeoSearchResult!, errorCode: HDMDistrictSearchErrorCode) {

if (nil == searchResult) {

NSLog("onGetDistrictGeoSearchResult failed:\(errorCode)")

return

}

var upload\_text = "[\(search\_list[searcher.hashValue]!.req!)]\r\n"

upload\_text += "\(searchResult.status);\(searchResult.message);"

if (0 == searchResult.status) {

upload\_text += "\(searchResult.districtProperties.district\_name),\(searchResult.districtProperties.district\_code),\(searchResult.districtType),\(searchResult.districtGeometry.geometry\_type):"

for coord\_list2 in searchResult.districtGeometry.geometry\_coord {

for coord\_list1 in coord\_list2 as! [AnyObject] {

for coord in coord\_list1 as! [AnyObject] {

upload\_text += "\(coord[0]),\(coord[1]);"

}

}

}

}

}

### 地理编码

#### 正向地理编码

func onGetGeoCodeResult(searcher: HDMGeoCodeSearch!, result searchResult: HDMGeoCodeSearchResult!, errorCode: HDMGeoCodeSearchErrorCode) {

if (nil == searchResult) {

NSLog("onGetDistrictGeoSearchResult failed:\(errorCode)")

return

}

var upload\_text = "[\(search\_list[searcher.hashValue]!.req!)]\r\n"

upload\_text += "\(searchResult.status);\(searchResult.message);"

if (0 == searchResult.status) {

for info in searchResult.allgeoCodeInfoList as! [HDMGeoCodeInfo] {

upload\_text += "\(info.address),\(info.location.longitude),\(info.location.latitude);"

}

}

}

#### 逆地理编码

func onGetReverseGeoCodeResult(searcher: HDMGeoCodeSearch!, result searchResult: HDMReverseGeoCodeSearchResult!, errorCode: HDMGeoCodeSearchErrorCode) {

if (nil == searchResult) {

NSLog("onGetDistrictGeoSearchResult failed:\(errorCode)")

return

}

var upload\_text = "[\(search\_list[searcher.hashValue]!.req!)]\r\n"

upload\_text += "\(searchResult.status);\(searchResult.message);"

if (0 == searchResult.status) {

upload\_text += "\(searchResult.reverseGeoCodeInfo.address),\(searchResult.reverseGeoCodeInfo.location.longitude),\(searchResult.reverseGeoCodeInfo.location.latitude),\(searchResult.reverseGeoCodeInfo.telephone),\(searchResult.reverseGeoCodeInfo.district\_code),\(searchResult.reverseGeoCodeInfo.poi\_name);"

}

}