

## 关于发布位置信息插件 2.0 的说明

各有关单位:

为进一步贯彻落实《交通运输部办公厅关于进一步做好网络平台道路货物运输信息化监测工作的通知》(交办运函〔2020〕1520号)的部署要求,加强网络货运信息化监测,实施人流、车流、信息流闭环验证,我院在充分总结驾驶员位置信息插件 1.0 运行经验的基础上,研发了驾驶员位置信息插件(SDK)2.0 版本。请有关单位按照 2.0 版本的规范要求进行升级。驾驶员位置信息插件(SDK)2.0 版本技术规范见附件。现将主要事项说明如下:

### 一、插件获取办法

通过部交互系统网站(<http://wlhy.mot.gov.cn>)或 QQ 运维群(530051420、129197455)等渠道获取。下载文件为一个压缩包,包含文件列表如下:

表 1 压缩包文件列表

序号	文件名	说明
1	android_sdk.zip	andriod 版 sdk
2	ios_sdk.zip	ios 版 sdk
3	部网络货运信息交互系统 位置信息插件(SDK)接 口规范 V2.pdf	规范文档

## 二、插件升级内容

1、取消 1.0 中自动提取位置信息机制，改为由 APP 调用相关接口上报位置信息数据。开放了 send、pause、restart 接口，具体调用方法请参照规范（见附件）。

2、增加安全加密机制，链路接口采用 https 协议，插件包采用安全加固。

3、增加单据数据项，增加运单起始点经纬度、驾驶员姓名、车牌号等信息，具体数据项变化请参照规范（见附件）。

4、增加支持百度定位 SDK。修复 1.0 已知 bug。

## 三、插件集成事项

1、本版本（2.0）发布之后，旧版本（1.0）作废，企业需及时升级成 2.0 版本。从 2022 年 1 月 1 日起，1.0 数据接入接口将关闭。

2、已集成 1.0 插件的企业需向技术人员申请开通测试账号，重新集成 2.0 版本并测试，达到相关条件后，切换正式环境。

3、新申请开展线上能力认定的企业，需使用 2.0 开展相关测试认定，达到相关要求后，切换正式环境。

附件：

# **部网络货运信息交互系统位置信息 插件（ SDK ）接口规范 V2**

2021 年 11 月 1 日

# 目录

第 1 章 总体说明 .....	1
1.1 位置信息插件 ( SDK ) 集成流程 .....	1
1.2 位置信息插件 ( SDK ) 接口说明 .....	3
1.3 位置信息插件 ( SDK ) 使用场景说明 .....	5
第 2 章 Android 版 SDK 说明 .....	8
2.1 接口说明 .....	8
2.1.1 授权接口 ( auth ) .....	8
2.1.2 开启定位 ( start ) .....	9
2.1.3 发送定位 ( send ) .....	12
2.1.4 暂停定位 ( pause ) .....	13
2.1.5 重启定位 ( restart ) .....	15
2.1.6 结束定位 ( stop ) .....	16
2.2 配置与使用 .....	18
2.2.1 高德版本 .....	18
2.2.2 百度版本 .....	20
第 3 章 IOS 版 SDK 说明 .....	22
3.1 接口说明 .....	22

3.1.1 授权接口 ( auth ) .....	22
3.1.2 开启定位 ( start ) .....	24
3.1.3 发送定位 ( send ) .....	26
3.1.4 暂停定位 ( pause ) .....	28
3.1.5 重启定位 ( restart ) .....	30
3.1.6 结束定位 ( stop ) .....	31
3.2 配置与使用 .....	33
3.2.1 高德版本 .....	33
3.2.2 百度版本 .....	38
第 4 章 代码表 .....	42

# 第1章 总体说明

为进一步贯彻落实《交通运输部办公厅关于进一步做好网络平台道路货物运输信息化监测工作的通知》（交办运函〔2020〕1520 号）的部署要求，加强网络货运信息化监测，实施人流、车流、信息流闭环验证，我院在充分总结驾驶员位置信息插件 1.0 运行经验的基础上，研发了驾驶员位置信息插件（SDK）2.0 版本。

位置信息插件（SDK）的功能是：集成在网络货运企业司机端 APP 中，在运单起运到收货的运输过程中，抓取驾驶员位置信息数据，上传至部交互系统。

SDK 从手机设备中获取的数据列表如下：

表 1 获取数据列表

序号	数据名称	说明
1	经纬度信息	手机设备的经纬度信息
2	时间信息	手机设备的时间信息
3	运单信息	SDK 将运单信息保存于手机缓存
4	设备 ID	手机设备的唯一标识码

## 1.1 位置信息插件（SDK）集成流程

网络货运企业使用位置信息插件（SDK）流程如下：



图 1 集成流程

### 1、获取驾驶员位置信息插件

可以通过部交互系统网站（<http://wlhy.mot.gov.cn>）或 QQ 运维群（530051420、129197455）等渠道获取。下载文件为一个压缩包，包含文件列表如下：

表 2 压缩包文件列表

序号	文件名	说明
1	android_sdk.zip	andriod 版 sdk(包含 support 和 androidx 版本)
2	ios_sdk.zip	ios 版 sdk
3	部网络货运信息交互系统 位置信息插件（SDK）接 口规范 V2.pdf	规范文档

目录结构图如下：andriod 版本分为分别支持 androidx、support 的百度版本和高德版本,ios 分为百度和高德版本，app 开发端请根据实际情况选择对应的版本进行开发。

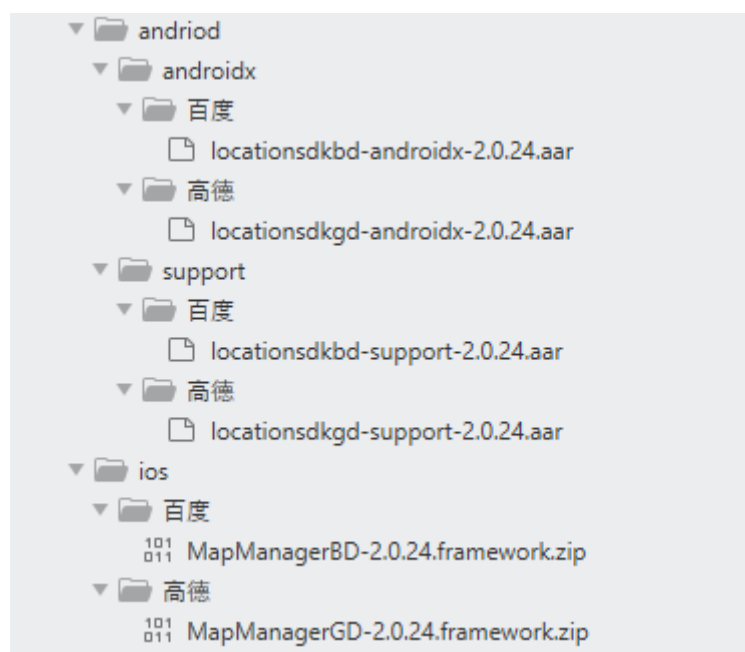


图 2 SDK 包目录结构

## 2、提交信息，获取安全码

企业在省级网络货运信息监测系统提交相关信息时，需同时提交

企业司机端 App 的唯一标识（APPID），省级监测系统将为企业分配安全码(AppSecurity)。如未获取，将无法开展集成联调测试。

### 3、司机端 APP 集成 SDK 开发

企业获取到 SDK 包以及安全码以及第三方地图网站的授权码，将 SDK 集成到司机端 APP 中，应将原有集成的 1.0 版本删除重新集成。具体步骤请参考第 2、3 章。

### 4、司机端 APP 测试

完成集成开发后，企业应使用 debug 模式进行功能测试，为避免误操作将测试数据传入 release 环境，企业可向 QQ 群内的技术人员申请测试安全码进行测试。

### 5、司机端 APP 发布

完成测试后，APP 上线发布前，应将环境参数配置成 release，将测试安全码修改成正式安全码，再进行正式发布。

## 1.2 位置信息插件（SDK）接口说明

位置信息插件（SDK）与司机端 APP 接口有 6 个，如下表所示：

表 3 SDK 接口列表

序号	接口名称	说明
1	auth	用于向 SDK 传入企业相关信息，进行企业信息认证。认证有效时间为 2 小时，如超过 2 小时无任何操作，则需调用本接口再次认证。
2	start	运单起运时，由驾驶员操作触发，传入待起运运单列表、车牌、驾驶员、预计起止地等信息，开启驾驶员定位。返回调用结果及



		下次上传的时间间隔。
3	send	运单运输过程中，在 SDK 指定的时间间隔，由 APP 自动触发，传入需触发定位的运单列表。
4	stop	运单到达后，由司机操作触发，传入待结束的运单列表、车牌、驾驶员、实际起止地等信息，结束传入运单的定位。
5	pause	运输过程中，需要换驾驶员或更换车辆，需要先调用此接口，传入待暂停的运单列表、车牌、驾驶员、本段行程实际起止地等信息，暂停驾驶员定位。
6	restart	此接口对应 pause 接口，调用此接口将更换后的运单列表、车牌、驾驶员、本段行程的预计起止地传入，重启驾驶员定位。

适配不同手机操作系统的 SDK 的接口参数列表及数据项说明详见第 3 章。

# 1.3 位置信息插件（SDK）使用场景说明

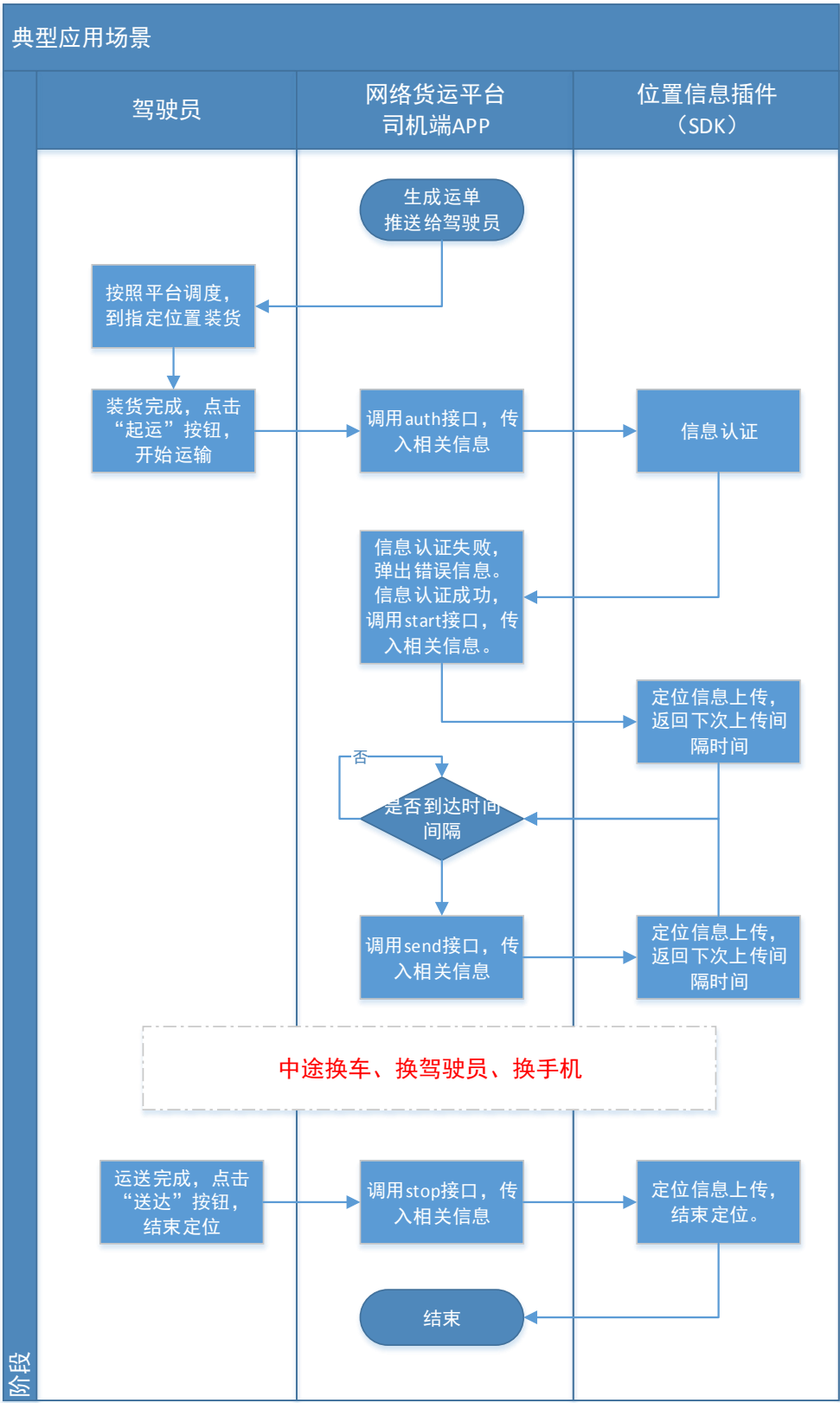


图 3 典型应用场景

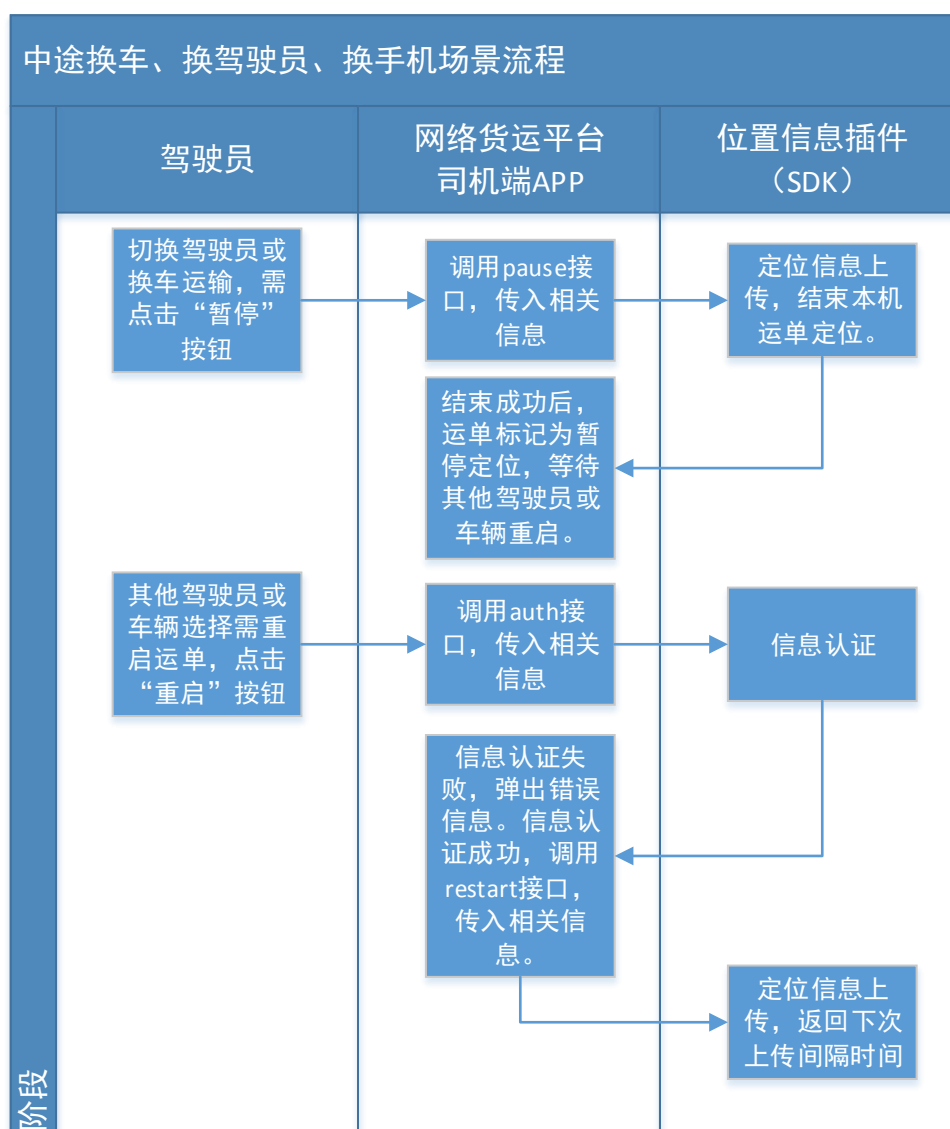


图 4 中途换车、换驾驶员、换手机场景流程

本节为企业 APP 使用 SDK 的典型使用场景，涉及对象包括驾驶员、网络货运企业司机端 APP 和位置信息插件。具体流程如下：

1、网络货运平台配货、调度车辆，确定驾驶员运送的运单列表，推送至驾驶员 APP。驾驶员装货完成后，在企业 APP 里待运输的运单点击“起运”按钮。

2、APP 接收起运指令，向 SDK 发送 auth 接口，进行企业身份

认证。认证成功后，则可调用 **start** 接口，进行起运位置信息上传。

上传成功后，将返回运单列表的下次定位上传时间。

3、APP 接收到运单列表中每个运单的下次定位上传时间，需对每个运单进行倒计时，在到达时间间隔后，调用 **send** 接口，上传该运单的位置信息。

4、如中途存在换驾驶员、换车或换手机的情况，则需驾驶员在原手机上操作，对运送运单列表执行暂停操作，调用 **pause** 接口，SDK 将运单列表中所有运单标记为暂停状态，并结束本手机的运单定位。

5、如更换驾驶员，企业 APP 将该运单列表推送给新驾驶员账号并更新运单列表中的驾驶员名称，新驾驶员登录 APP 后，对该运单列表执行重启定位操作，调用 **auth** 和 **restart** 接口，对已暂停的运单列表执行重启定位操作。如更换车辆，企业 APP 应将运单进行分单标记并使用新分单号与车牌号上传，调用 **restart** 接口，对已暂停的运单列表执行重启定位操作。如为更换手机，驾驶员需登录新手机的 APP，并操作 APP 重启运单，调用 **auth** 和 **restart** 接口。之后，按钮返回的时间间隔调用 **send** 接口即可。

6、运单收货后，驾驶员点击“送达”按钮，APP 传入待结束的运单列表，调用 **stop** 接口，SDK 将上报终点位置数据，并结束该运单列表中所有运单的定位。

## 第2章 Android 版 SDK 说明

### 2.1 接口说明

#### 2.1.1 授权接口（auth）

##### 2.1.1.1 接口描述

本接口用于 SDK 对企业进行授权认证。配置企业信息、APP 信息、接入环境等。接口格式如下：

**auth** ( *Context context* , *String appId*, *String appSecurity*, *String enterpriseSenderId* , *String environment* , *OnResultListener listener*)

##### 2.1.1.2 参数说明

表 4 auth 接口参数列表

序号	参数名	必填	类型	说明
1	context	是	Context	上下文
2	appId	是	String	网络货运企业 APP 的唯一标识
3	appSecurity	是	String	网络货运企业在省平台申请的接入安全码
4	enterpriseSenderId	是	String	网络货运企业在省平台申请的企业发送代码
5	environment	是	String	环境: “debug” 接入测试环境, “release” 接入正式环境。
6	listener	是	OnResultListener	返回结果回调函数

### 2.1.1.3 返回结果

- 接口调用成功回调 `onSuccess` (`List<ShippingNoteInfo> shippingNoteInfos`)。

`ShippingNoteInfo` 参数如下：

表 5 ShippingNoteInfo 参数列表

序号	参数名	类型	说明
1	vehicleNumber	String	车牌号
2	driverName	String	司机姓名
3	shippingNoteNumber	String	运单号
4	serialNumber	String	分单号
5	interval	String	定位间隔时间(单位 ms)

- 接口失败回调 `onFailure` (`String errorCode, String errorMsg`)。

`errorCode`: 错误代码。(见第 4 章)

`errorMsg`: 错误提示信息。

## 2.1.2 开启定位 (start)

### 2.1.2.1 接口描述

企业司机端 APP 中运单列表启运时, 驾驶员点击按钮确认启运后, 调用本方法。接口格式如下:

```
start(Context context, String vehicleNumber, String driverName, String remark, final ShippingNoteInfo[] shippingNoteNumbers, @NonNull final OnResultListener listener)
```

### 2.1.2.2 参数说明

表 6 start 接口参数列表

序号	参数名	必填	类型	说明
1	context	是	Context	上下文
2	vehicleNumber	是	String	车牌号
3	driverName	是	String	司机姓名
4	remark	否	String	备注
5	shippingNoteInfos	是	ShippingNoteInfo	运单信息列表
6	listener	是	OnResultListener	结果回调函数

shippingNoteInfos 为运单信息列表，考虑到一车多单的情况，本版 SDK 一辆车运单数最大支持数为 10。此处可传入运单信息列表。参数如下：

表 7 shippingNoteInfo 参数列表

序号	参数名	必填	类型	说明
1	shippingNoteNumber	是	String	运单号
2	serialNumber	是	String	分单号
3	startCountrySubdivisionCode	待定	String	起点位置行政区划代码，调用 start/stop/pause/restart 时必填，调用 send 非必填
4	endCountrySubdivisionCode	待定	String	到达位置行政区划代码，调用 start/stop/pause/restart 时必填，调用 send 非必填
5	startLongitude	待定	Double	起点位置经度，调用 start/stop/pause/restart 时必填，调用 send 非必填
6	startLatitude	待定	Double	起点位置纬度，调用 start/stop/pause/restart 时必填，调用 send 非必填
7	endLongitude	待定	Double	到达位置经度，调用 start/stop/pause/restart 时必填，调用 send 非必填
8	endLatitude	待定	Double	到达位置纬度，调用 start/stop/pause/restart 时必

序号	参数名	必填	类型	说明
				填，调用 send 非必填
9	startLocationText	待定	String	起点地址文字描述，调用 start/stop/pause/restart 时必填，调用 send 非必填
10	endLocationText	待定	String	到达地址文字描述，调用 start/stop/pause/restart 时必填，调用 send 非必填
11	vehicleNumber	否	String	车牌号，SDK 回调返回，调用 start/stop/pause/restart/必填，send 时非必填
12	driverName	否	String	司机姓名，SDK 回调返回，调用 start/stop/pause/restart/必填，send 时非必填
13	interval	否	String	请求时间间隔，SDK 回调返回(单位 ms)

### 2.1.2.3 返回结果

- 调用成功回调 onSuccess ( *List<ShippingNoteInfo>* *shippingNoteInfos*)。

*ShippingNoteInfo* 参数如下：

表 8 ShippingNoteInfo 参数

序号	参数名	类型	说明
1	vehicleNumber	String	车牌号
2	driverName	String	司机姓名
3	shippingNoteNumber	String	运单号
4	serialNumber	String	分单号
5	interval	String	定位间隔时间(单位 ms)

- 调用失败回调 onFailure (String errorCode,String errorMsg)。  
errorCode: 错误代码。



errorMsg: 失败提示语。

## 2.1.3 发送定位 (send)

### 2.1.3.1 接口描述

运单运输过程中, APP 应当按照 SDK 返回的时间间隔调用本接口实现位置信息上报。接口格式如下:

```
send(Context context, String vehicleNumber, String driverName,String remark,  
final ShippingNoteInfo[] shippingNoteNumbers, @NonNull final OnResultListener  
listener)
```

### 2.1.3.2 参数说明

表 9 send 接口参数列表

序号	参数名	必填	类型	说明
1	context	是	Context	上下文
2	vehicleNumber	是	String	车牌号
3	driverName	是	String	司机姓名
4	remark	否	String	备注
5	shippingNoteInfos	是	ShippingNoteInfo	运单信息列表
6	listener	是	OnResultListener	结果回调函数

参数 ShippingNoteInfo 和 start 接口中的参数 ShippingNoteInfo 相同, 这里 shippingNoteInfo 信息只需传入运单号和分单号即可。

### 2.1.3.3 返回结果

- 调用成功回调 `onSuccess ( List<ShippingNoteInfo> shippingNoteInfos )`。

*ShippingNoteInfo* 参数如下：

表 10 ShippingNoteInfo 参数

序号	参数名	类型	说明
1	vehicleNumber	String	车牌号
2	driverName	String	司机姓名
3	shippingNoteNumber	String	运单号
4	serialNumber	String	分单号
5	interval	String	定位间隔时间(单位 ms)

- 调用失败回调 `onFailure (String errorCode,String errorMsg)`。

`errorCode`: 错误代码。

`errorMsg`: 失败提示语。

### 2.1.4 暂停定位 (pause)

#### 2.1.4.1 接口描述

运单运输过程中，如果出现换驾驶员、换车或换手机的情况，则需驾驶员在原手机上操作，对运送运单列表执行暂停操作，调用本接口将运单列表中所有运单标记为暂停状态，并结束本手机的运单定位。

接口参数如下：

**pause**(Context context, String vehicleNumber, String driverName,String remark, final ShippingNoteInfo[] shippingNoteNumbers, @NonNull final OnResultListener listener)

### 2.1.4.2 参数说明

表 11 pause 接口参数列表

序号	参数名	必填	类型	说明
1	context	是	Context	上下文
2	vehicleNumber	是	String	车牌号
3	driverName	是	String	司机姓名
4	remark	否	String	备注
5	shippingNoteInfos	是	ShippingNoteInfo	运单信息列表
6	listener	是	OnResultListener	结果回调函数

参数 ShippingNoteInfo 和 start 接口中的参数 ShippingNoteInfo 相同。

### 2.1.4.3 返回结果

- 调用成功回调：

**onSuccess** (*List<ShippingNoteInfo> shippingNoteInfos*)

*ShippingNoteInfo* 参数如下：

表 12 ShippingNoteInfo 参数

序号	参数名	类型	说明
1	vehicleNumber	String	车牌号
2	driverName	String	司机姓名
3	shippingNoteNumber	String	运单号
4	serialNumber	String	分单号
5	interval	String	定位间隔时间(单位 ms)

- 调用失败回调：

**onFailure** (*String errorCode,String errorMsg*)。

errorCode: 错误代码。

errorMsg: 失败提示语。

## 2.1.5 重启定位 (restart)

### 2.1.5.1 接口描述

对应 pause 接口，本接口的作用是对已暂停的运单列表执行重启定位操作，参数如下：

**restart** (*Context context, String vehicleNumber, String driverName,String remark, final ShippingNoteInfo[] shippingNoteNumbers, @NonNull final OnResultListener listener*)

### 2.1.5.2 参数说明

表 13 restart 接口参数列表

序号	参数名	必填	类型	说明
1	context	是	Context	上下文
2	vehicleNumber	是	String	车牌号
3	driverName	是	String	司机姓名
4	shippingNoteInfos	是	ShippingNoteInfo	运单信息列表
5	listener	是	OnResultListener	结果回调函数
6	remark	否	String	备注

参数 ShippingNoteInfo 和 start 接口中的参数 ShippingNoteInfo 相同。

### 2.1.5.3 返回结果

- 调用成功回调：

**onSuccess** (*List<ShippingNoteInfo> shippingNoteInfos*)

*ShippingNoteInfo* 参数如下：

表 14 ShippingNoteInfo 参数

序号	参数名	类型	说明
1	vehicleNumber	String	车牌号
2	driverName	String	司机姓名
3	shippingNoteNumber	String	运单号
4	serialNumber	String	分单号
5	interval	String	定位间隔时间(单位 ms)

- 调用失败回调：

**onFailure** (*String errorCode,String errorMsg*)。

errorCode： 错误代码。

errorMsg： 失败提示语。

## 2.1.6 结束定位（stop）

### 2.1.6.1 接口描述

货物送达时，驾驶员点击货物到达按钮，调用本接口结束运单定位。

接口格式如下：

**stop**(*Context context, String vehicleNumber, String driverName,String remark, final ShippingNoteInfo[] shippingNoteNumbers, @NonNull final OnResultListener listener*)

### 2.1.6.2 参数说明

表 15 stop 接口参数列表

序号	参数名	必填	类型	说明
1	context	是	Context	上下文
2	vehicleNumber	是	String	车牌号
3	driverName	是	String	司机姓名
4	remark	否	String	备注
5	shippingNoteInfos	是	ShippingNoteInfo	运单信息列表
6	listener	是	OnResultListener	结果回调函数

参数 ShippingNoteInfo 和 start 接口中的参数 ShippingNoteInfo 相同。

### 2.1.6.3 返回结果

- 调用成功回调：

**onSuccess** (*List<ShippingNoteInfo> shippingNoteInfos*)。

*ShippingNoteInfo* 参数如下：

表 16 ShippingNoteInfo 参数

序号	参数名	类型	说明
1	vehicleNumber	String	车牌号
2	driverName	String	司机姓名
3	shippingNoteNumber	String	运单号
4	serialNumber	String	分单号
5	interval	String	定位间隔时间(单位 ms)

- 调用失败回调：

**onFailure** (*String errorCode,String errorMsg*)。

errorCode: 错误代码。

errorMsg: 失败提示语。

## 2.2 配置与使用

Android 版 SDK 包括高德和百度定位开发包两种版本，企业根据自身情况决定使用哪个版本，使用前应先申请高德或百度 SDK 的 key 和 secret。

下面为需要授权的列表：

表 17 所需权限列表

序号	权限	用途
1	定位	获取当前位置并上传后台
2	网络服务	提交信息和请求数据

### 2.2.1 高德版本

SDK 集成的步骤如下：

- 1、选择合适的 aar 文件
- 2、将文件复制粘贴至 app/libs 文件夹下
- 3、在 app/build.gradle 文件中添加如下依赖，并构建项目

```
i.      implementation 'com.lzy.net:okgo:3.0.4'
ii.     implementation 'com.alibaba:fastjson:1.2.61'
iii.    implementation 'org.bouncycastle:bcprov-jdk15on:1.55'
iv.     implementation 'org.apache.commons:commons-lang3:3.5'
v.      implementation 'com.amap.api:location:latest.integration'
vi.     implementation files('libs/locationsdk-2.0.24.aar')
```

#### 4、在 AndroidManifest.配置高德定位 appkey 和添加服务

```
i.    <meta-data
ii.        android:name="com.amap.api.v2.apikey"
iii.        android:value="{高德 appKey}" />
iv.    <service
v.        android:name="com.amap.api.location.APSService"></service>
```

#### 5、在 Application 中初始化 sdk 服务

```
LocationOpenApi.init(Application application);
```

#### 6、在项目中授权 sdk

```
i.    LocationOpenApi.auth(Context context,String key,String
    secret,String enterpriseSenderCode,String
    environment,OnResultListener listener);
```

#### 7、在项目中使用 sdk 服务

```
ii.    //启用服务

i.    LocationOpenApi.start(Context context, String
    vehicleNumber,String driverName,ShippingNoteInfo[]
    shippingNoteInfos,OnResultListener listener)

ii.    //发送服务
iii.    LocationOpenApi.send(Context context, String
    vehicleNumber,String driverName,ShippingNoteInfo[]
    shippingNoteInfos,OnResultListener listener)

iv.    //停止服务
v.    LocationOpenApi.stop(Context context, String
    vehicleNumber,String driverName,ShippingNoteInfo[]
    shippingNoteInfos,OnResultListener listener)
```



## 2.2.2 百度版本

- 1、选择合适的 aar 文件。
- 2、将文件复制粘贴至 app/libs 文件夹下。
- 3、在 app/build.gradle 文件中添加如下依赖，并构建项目

```
i.    implementation 'com.lzy.net:okgo:3.0.4'  
ii.   implementation 'com.alibaba:fastjson:1.2.61'  
iii.  implementation 'org.bouncycastle:bcprov-jdk15on:1.55'  
iv.   implementation 'org.apache.commons:commons-lang3:3.5'  
v.    implementation files('libs/locationsdkbd-2.0.24.aar')
```

- 4、在 AndroidManifest.配置百度定位 appkey

```
i.    <meta-data  
ii.         android:name="com.baidu.lbsapi.API_KEY"  
iii.         android:value="{百度 key}" >  
iv.    </meta-data>
```

- 5、在 Application 中初始化 sdk 服务

```
i.    LocationOpenApi.init(Application application);
```

- 6、在项目中授权 sdk

```
i.    LocationOpenApi.auth(Context context,String key,String  
    secret,String enterpriseSenderCode,String  
    environment,OnResultListener listener);
```

- 7、在项目中使用 sdk 服务

```
i.    //启用服务
ii.   LocationOpenApi.start(Context context, String
      vehicleNumber,String driverName,ShippingNoteInfo[]
      shippingNoteInfos,OnResultListener listener)
iii.  //发送服务
iv.   LocationOpenApi.send(Context context, String
      vehicleNumber,String driverName,ShippingNoteInfo[]
      shippingNoteInfos,OnResultListener listener)
v.    //停止服务
vi.   LocationOpenApi.stop(Context context, String
      vehicleNumber,String driverName,ShippingNoteInfo[]
      shippingNoteInfos,OnResultListener listener)
```

## 第3章 IOS 版 SDK 说明

### 3.1 接口说明

#### 3.1.1 授权接口（auth）

##### 3.1.1.1 接口描述

本接口用于 SDK 对企业进行授权认证。配置企业信息、APP 信息、接入环境等。接口格式如下：

```
-(void)openServiceWithAppId:(NSString *)appId  
appSecurity:(NSString *)appSecurity  
enterpriseSenderCode:(NSString *)enterpriseSenderCode  
environment:(NSString *)environment  
listener:(void(^)(id model, NSError *error))listener;
```

##### 3.1.1.2 参数说明

表 18 openServiceWithAppId 接口参数列表

序号	参数名	必填	类型	说明
1	appId	是	NSString	网络货运企业 APP 的唯一标识
2	appSecurity	是	NSString	网络货运企业在省平台申请的接入安全码
3	enterpriseSenderCode	是	NSString	网络货运企业在省平台申请的企业发送代码
4	environment	是	NSString	环境: “debug” 接入测试环境, “release” 接入正式环境。
5	listener	是		返回结果回调函数

### 3.1.1.3 返回结果

- 接口调用成功回调 **listener** (*id model*)。

- **model** 中参数：

返回成功

**code**: 0。

**data**: 当前运行中订单。

**message**: 成功信息。

返回错误

**code**: 错误代码。（见第 4 章）

**data**: 当前运行中订单。

**fail**: 错误描述。

**message**: 错误提示信息。

下表为 **data** 中参数：

表 19 data 参数

序号	参数名	类型	说明
1	vehicleNumber	NSString	车牌号
2	driverName	NSString	司机姓名
3	shippingNoteNumber	NSString	运单号
4	serialNumber	NSString	分单号
5	interval	NSString	定位间隔时间(单位 ms)

## 3.1.2 开启定位（start）

### 3.1.2.1 接口描述

企业司机端 APP 中运单列表启运时，驾驶员点击按钮确认启运后，调用本方法。接口格式如下：

```
-(void)startLocationWithShippingNoteInfos:(NSArray *)shippingNoteInfos  
driverNameView:(NSString*)driverNameView  
vehicleNumberView:(NSString*)vehicleNumberView  
listener:(void(^)(id model, NSError *error))listener;
```

### 3.1.2.2 参数说明

表 20 startLocationWithShippingNoteInfos 接口参数列表

序号	参数名	必填	类型	说明
1	vehicleNumber	是	NSString	车牌号
2	driverName	是	NSString	司机姓名
3	shippingNoteInfos	是	NSArray	运单信息列表
4	listener	是		结果回调函数

shippingNoteInfos 为运单信息列表，考虑到一车多单的情况，本版 SDK 一辆车运单数最大支持数为 10。此处可传入运单信息列表。参数如下：

表 21 shippingNoteInfo 参数列表

序号	参数名	必填	类型	说明
1	shippingNoteNumber	是	NSString	运单号
2	serialNumber	是	NSString	分单号
3	startCountrySubdivisionCode	待定	NSString	起点位置行政区划代码，调用 start/stop/pause/restart 时

序号	参数名	必填	类型	说明
				必填，调用 send 非必填
4	endCountrySubdivisionCode	待定	NSString	到达位置行政区划代码，调用 start/stop/pause/restart 时必填，调用 send 非必填
5	startLongitude	待定	NSString	起点位置经度，调用 start/stop/pause/restart 时必填，调用 send 非必填
6	startLatitude	待定	NSString	起点位置纬度，调用 start/stop/pause/restart 时必填，调用 send 非必填
7	endLongitude	待定	NSString	到达位置经度，调用 start/stop/pause/restart 时必填，调用 send 非必填
8	endLatitude	待定	NSString	到达位置纬度，调用 start/stop/pause/restart 时必填，调用 send 非必填
9	startLocationText	待定	NSString	起点地址文字描述，调用 start/stop/pause/restart 时必填，调用 send 非必填
10	endLocationText	待定	NSString	到达地址文字描述，调用 start/stop/pause/restart 时必填，调用 send 非必填

### 3.1.2.3 返回结果

- 接口调用成功回调 **listener** (*id model*)。

- model 中参数：

返回成功：

code:0。

data: 当前运行中订单。

message: 成功信息

返回错误：

code: 错误代码。（见第 4 章）

data: 当前运行中订单。

fail: 错误描述。

message: 错误提示信息。

下表为 data 中参数：

表 22 data 参数

序号	参数名	类型	说明
1	vehicleNumber	NSString	车牌号
2	driverName	NSString	司机姓名
3	shippingNoteNumber	NSString	运单号
4	serialNumber	NSString	分单号
5	interval	NSString	定位间隔时间(单位 ms)

### 3.1.3 发送定位（send）

#### 3.1.3.1 接口描述

运单运输过程中，APP 应当按照 SDK 返回的时间间隔调用本接口实现位置信息上报。接口格式如下：

```
-(void)sendLocationWithShippingNoteInfos:(NSArray *)shippingNoteInfos  
driverNameView:(NSString*)driverNameView  
vehicleNumberView:(NSString*)vehicleNumberView  
listener:(void(^)(id model, NSError *error))listener;
```

#### 3.1.3.2 参数说明

表 23 sendLocationWithShippingNoteInfos 接口参数列表

序号	参数名	必填	类型	说明
1	vehicleNumber	是	NSString	车牌号
2	driverName	是	NSString	司机姓名

序号	参数名	必填	类型	说明
3	shippingNoteInfos	是	NSArray	运单信息列表
4	listener	是		结果回调函数

参数 ShippingNoteInfo 和 start 接口中的参数 ShippingNoteInfo 相同，这里 shippingNoteInfo 信息只需传入运单号和分单号即可。

### 3.1.3.3 返回结果

- 接口调用成功回调 **listener** (*id model*)。
- model 中参数：

返回成功：

code: 0。

message: 成功信息。

返回错误

code: 错误代码。（见第 4 章）

data: 当前运行中订单。

fail: 错误描述。

message: 错误提示信息。

下表为 data 中参数：

表 24 data 参数

序号	参数名	类型	说明
1	vehicleNumber	NSString	车牌号
2	driverName	NSString	司机姓名
3	shippingNoteNumber	NSString	运单号
4	serialNumber	NSString	分单号



序号	参数名	类型	说明
5	interval	NSString	定位间隔时间(单位 ms)

### 3.1.4 暂停定位（pause）

#### 3.1.4.1 接口描述

运单运输过程中，如果出现换驾驶员、换车或换手机的情况，则需驾驶员在原手机上操作，对运送运单列表执行暂停操作，调用本接口将运单列表中所有运单标记为暂停状态，并结束本手机的运单定位。接口参数如下：

```

-(void)pauseLocationWithShippingNoteInfos:(NSArray *)shippingNoteInfos
driverNameView:(NSString*)driverNameView
vehicleNumberView:(NSString*)vehicleNumberView
remark:(NSString*)remark
listener:(void(^)(id model, NSError *error))listener;

```

#### 3.1.4.2 参数说明

表 25 pauseLocationWithShippingNoteInfos 接口参数列表

序号	参数名	必填	类型	说明
1	vehicleNumber	是	NSString	车牌号
2	driverName	是	NSString	司机姓名
3	remark	否	NSString	备注
4	shippingNoteInfos	是	NSArray	运单信息列表
5	listener	是		结果回调函数

参数 ShippingNoteInfo 和 start 接口中的参数 ShippingNoteInfo 相同。

### 3.1.4.3 返回结果

- 接口调用成功回调 **listener** (*id model*)。

- model 中参数:

返回成功:

code: 0。

message: 成功信息。

返回错误:

code: 错误代码。(见第 4 章)

data: 当前运行中订单。

fail: 错误描述。

message: 错误提示信息。

下表为 data 中参数:

表 26 data 参数

序号	参数名	类型	说明
1	vehicleNumber	NSString	车牌号
2	driverName	NSString	司机姓名
3	shippingNoteNumber	NSString	运单号
4	serialNumber	NSString	分单号
5	interval	NSString	定位间隔时间(单位 ms)

## 3.1.5 重启定位（restart）

### 3.1.5.1 接口描述

对应 `pause` 接口，本接口的作用是对已暂停的运单列表执行重启定位操作，参数如下：

```
-(void)reStartLocationWithShippingNoteInfos:(NSArray *)shippingNoteInfos  
driverNameView:(NSString*)driverNameView  
vehicleNumberView:(NSString*)vehicleNumberView  
remark:(NSString*)remark  
listener:(void(^)(id model, NSError *error))listener;
```

### 3.1.5.2 参数说明

表 27 reStartLocationWithShippingNoteInfos 接口参数列表

序号	参数名	必填	类型	说明
1	vehicleNumber	是	NSString	车牌号
2	driverName	是	NSString	司机姓名
3	remark	否	NSString	备注
4	shippingNoteInfos	是	NSArray	运单信息列表
5	listener	是		结果回调函数

参数 `ShippingNoteInfo` 和 `start` 接口中的参数 `ShippingNoteInfo` 相同。

### 3.1.5.3 返回结果

- 接口调用成功回调 **listener** (*id model*)。
- `model` 中参数：

返回成功：

code: 0。

data: 当前运行中订单。

message: 成功信息。

返回错误

code: 错误代码。(见第 4 章)

fail: 错误描述。

message: 错误提示信息。

下表为 data 中参数:

表 28 data 参数

序号	参数名	类型	说明
1	vehicleNumber	NSString	车牌号
2	driverName	NSString	司机姓名
3	shippingNoteNumber	NSString	运单号
4	serialNumber	NSString	分单号
5	interval	NSString	定位间隔时间(单位 ms)

### 3.1.6 结束定位 (stop)

#### 3.1.6.1 接口描述

货物送达时, 驾驶员点击货物到达按钮, 调用本接口结束运单定位。

接口格式如下:

```

-(void)stopLocationWithShippingNoteInfos:(NSArray *)shippingNoteInfos
driverNameView:(NSString*)driverNameView
vehicleNumberView:(NSString*)vehicleNumberView
listener:(void(^)(id model, NSError *error))listener;

```

### 3.1.6.2 参数说明

表 29 stopLocationWithShippingNoteInfos 接口参数列表

序号	参数名	必填	类型	说明
1	vehicleNumber	是	NSString	车牌号
2	driverName	是	NSString	司机姓名
3	shippingNoteInfos	是	NSArray	运单信息列表
4	listener	是		结果回调函数

参数 ShippingNoteInfo 和 start 接口中的参数 ShippingNoteInfo 相同。

### 3.1.6.3 返回结果

- 接口调用成功回调 **listener** (*id model*)。
- model 中参数：

返回成功：

code: 0。

message: 成功信息。

返回错误：

code: 错误代码。（见第 4 章）

data: 当前运行中订单。

fail: 错误描述。

message: 错误提示信息。

下表为 data 中参数:

表 30 data 参数

序号	参数名	类型	说明
1	vehicleNumber	NSString	车牌号
2	driverName	NSString	司机姓名
3	shippingNoteNumber	NSString	运单号
4	serialNumber	NSString	分单号
5	interval	NSString	定位间隔时间(单位 ms)

## 3.2 配置与使用

IOS 版 SDK 包括高德和百度定位开发包两种版本,企业根据自身情况决定使用哪个版本,使用前应先申请高德或百度 SDK 的 key 和 secret。

下面为需要授权的列表:

表 31 IOS 需获取配置权限列表

序号	权限	用途
1	定位	获取当前位置并上传后台
2	网络服务	提交信息和请求数据

### 3.2.1 高德版本

#### 3.2.1.1 导入本 SDK

1、将本 SDK 直接拖入项目中 :

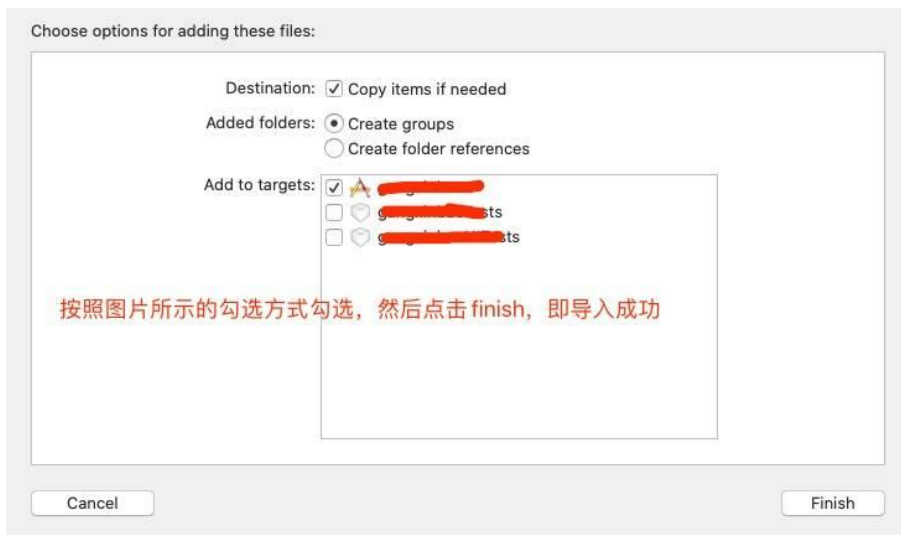


图 5 拖入项目

2、SDK 拖入项目后，在项目中显示如下图

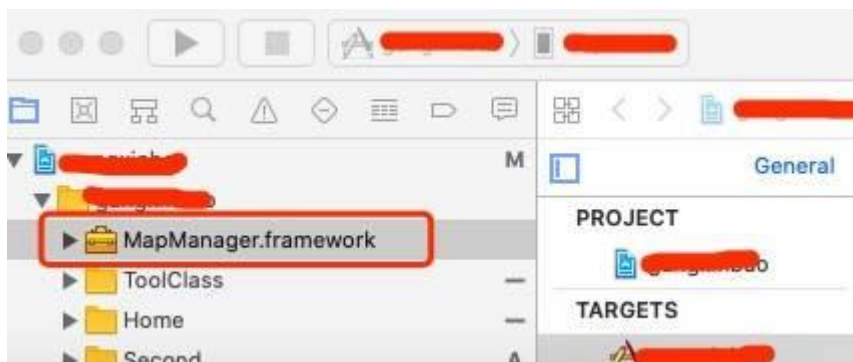


图 6 项目显示

3、进入下图页面位置，查看静态库是否导入进来，正常情况下会自动导入，如果没有，自己手动添加即可。

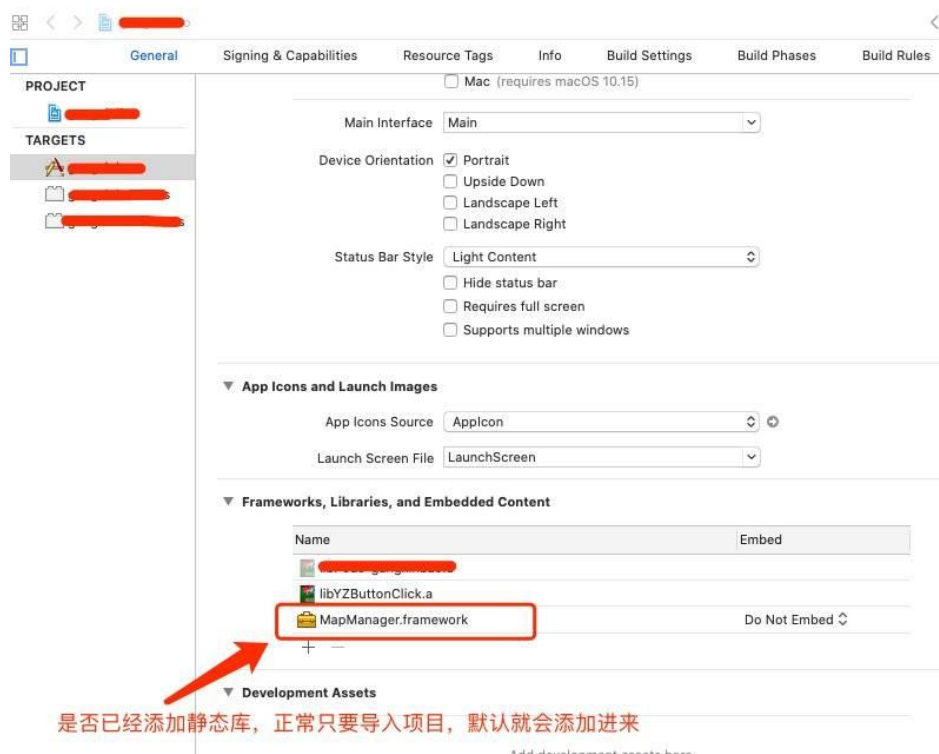


图 7 查看静态库

4、 SDK 导入成功，可以正常使用其功能。

在使用的页面，一定要记得引用 SDK。或者在 pch 文件中引用一次即可（整个项目都可以使用）。

1. `//导入 SDK`
2. `#import <MapManager/MapManager.h>`

### 3.2.1.2 通过 pods 导入高德地图定位和国密算法

在 Podfile 文件中配置安装：

1. `pod 'AMapLocation'`
2. `pod 'AMapSearch'`
3. `pod 'GMObjC'`



### 3.2.1.3 定位相关配置

#### 1、设置以上步骤需要勾选服务

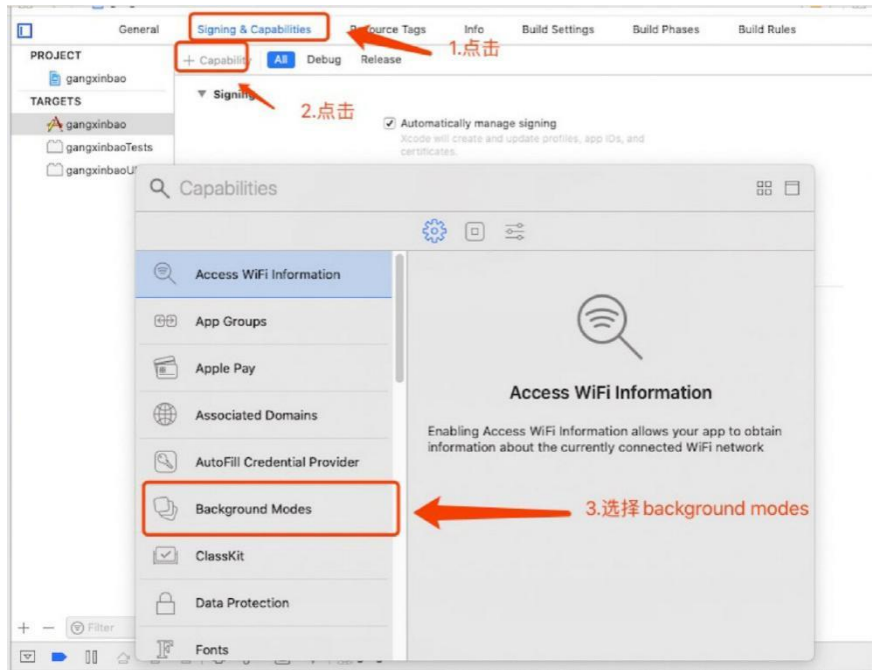


图 8 选择后台模式

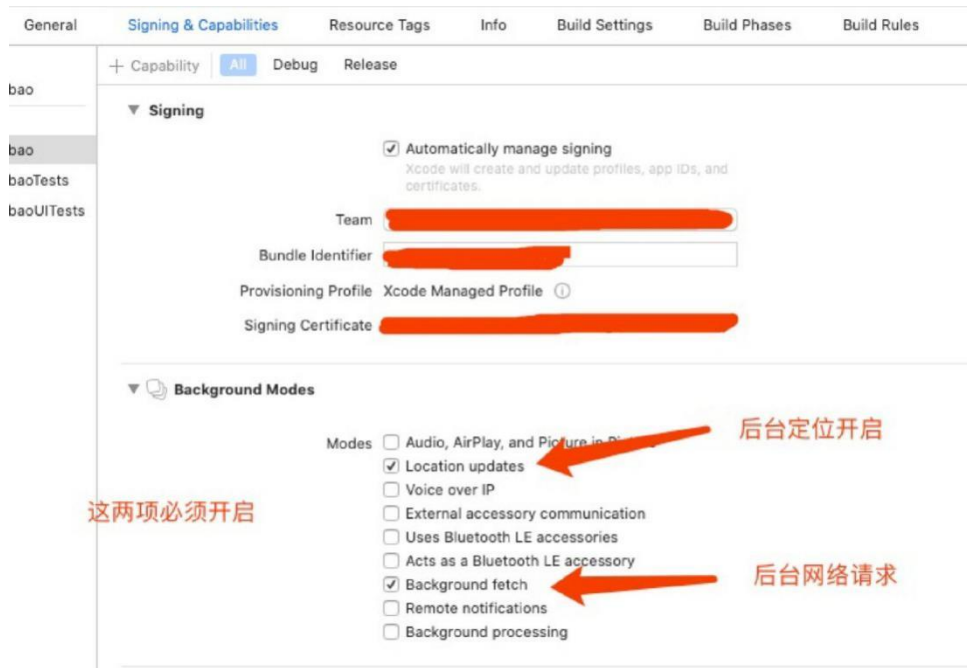


图 9 勾选定位与网络

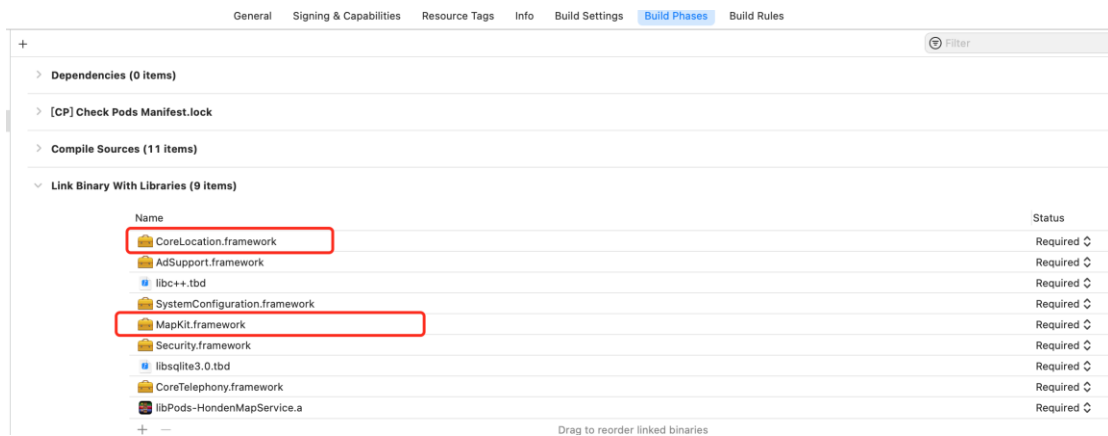


图 10 编译选项

## 3.2.2 百度版本

### 3.2.2.1 导入本 SDK

1、将百度版 SDK 直接拖入项目中：

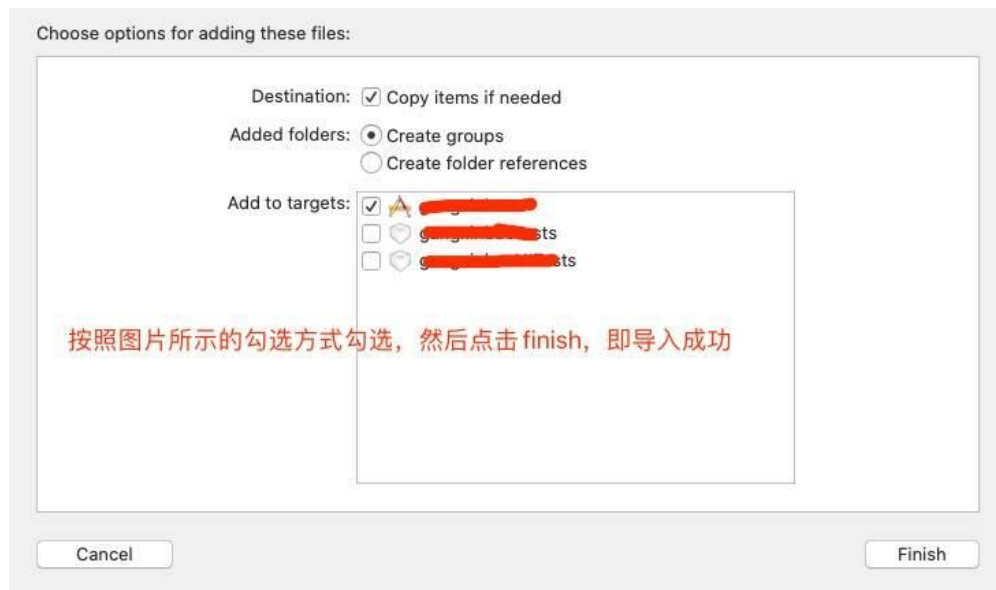


图 11 拖入项目

2、SDK 拖入项目后，在项目中显示如下图：

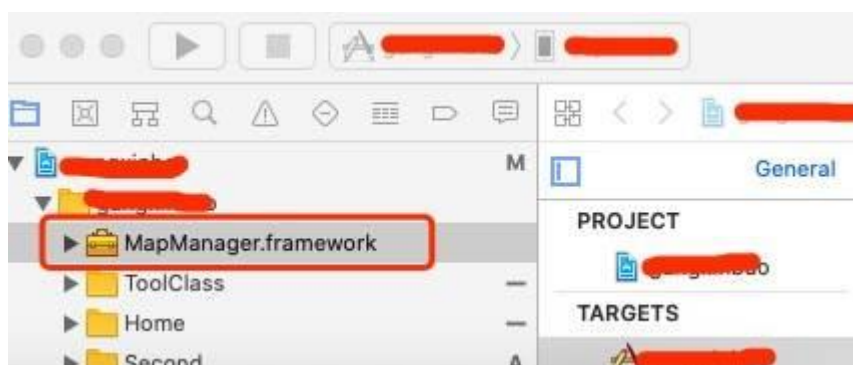


图 12 项目显示

3.进入下图页面位置，查看静态库是否导入进来，正常情况下会自动导入，如果没有，自己手动添加即可。

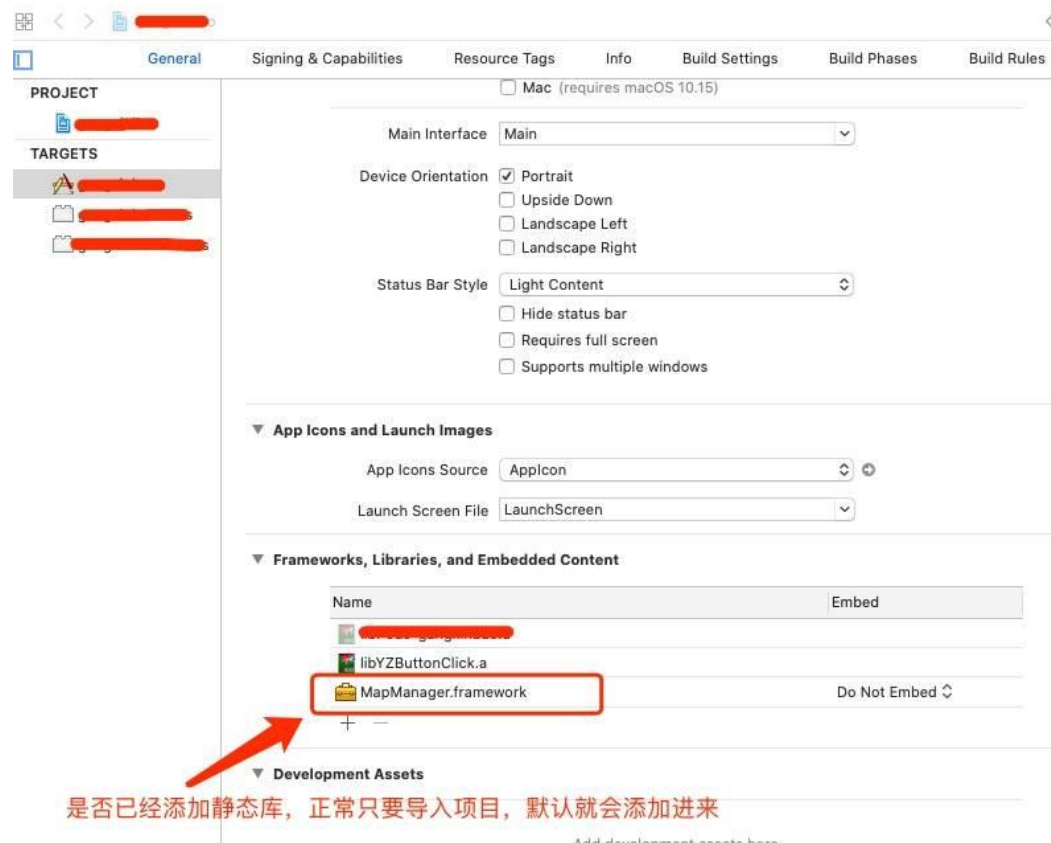


图 13 查看静态库

4、SDK 导入成功，可以正常使用其功能。

在使用的页面，一定要记得引用 SDK。或者在 pch 文件中引用一次即可（整个项目都可以使用）。

1. `//导入 SDK`
2. `#import <MapManagerBD/MapManagerBD.h>`

### 3.2.2.2 通过 pods 导入百度地图定位和国密算法

在 Podfile 文件中配置安装：

1. pod 'BaiduMapKit'
2. pod 'BMKLocationKit'
3. pod 'GMObjC'

### 3.2.2.3 定位相关配置

1、设置以上步骤需要勾选服务

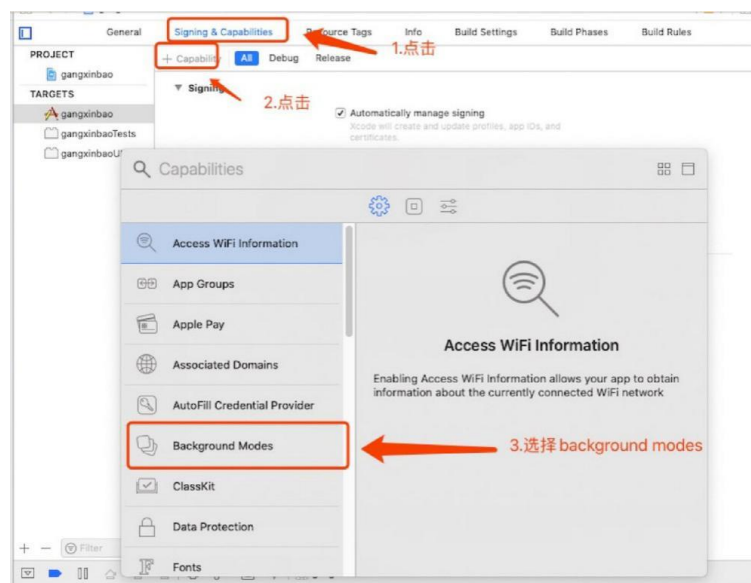


图 14 选择后台模式

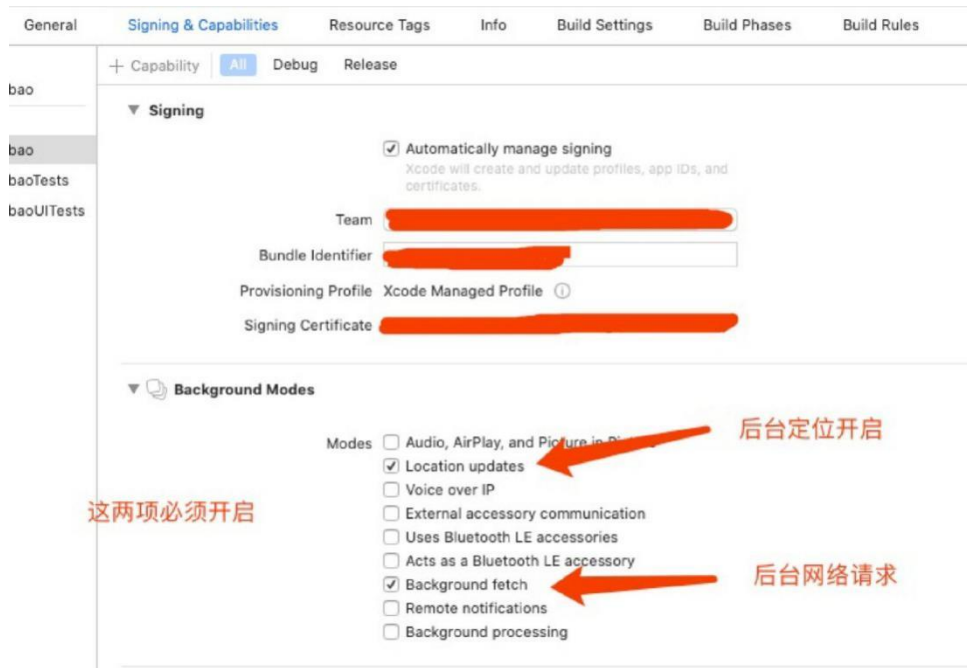


图 15 勾选定位与网络

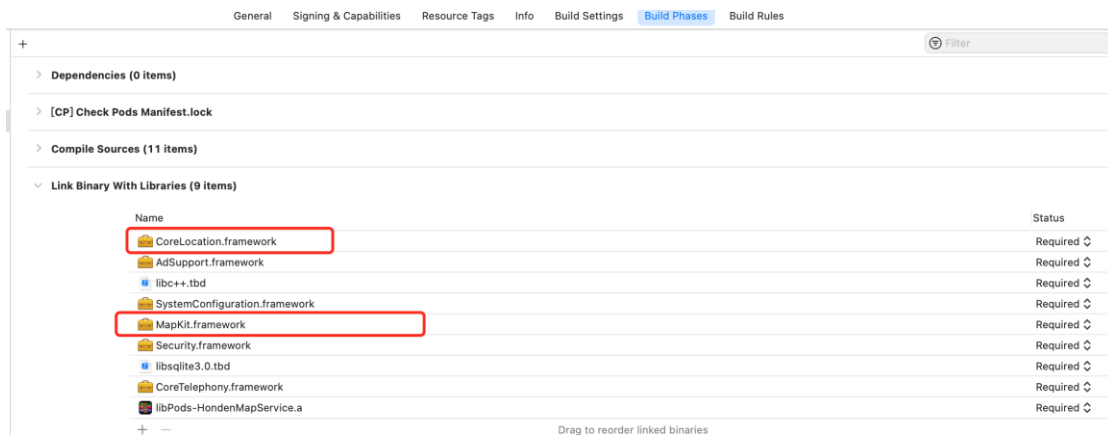


图 16 编译选项

## 第4章 代码表

表 32 SDK 返回代码说明列表

序号	代码	信息	说明
1	999999	未知错误	手机网络无信号、没打开网络或者发生未知错误
2	100000	上下文为空	context 参数为空
3	100001	企业 AppId 为空	网络货运企业 APP 的唯一标识参数为空
4	100002	企业 AppSecurity 为空	接入安全码参数为空
5	100003	企业发送代码为空	企业发送代码参数为空
6	100004	分单号为空	分单号参数为空
7	100005	环境参数为空	授权接口 environment 字段为空，环境：“debug”接入测试环境，“release”接入正式环境。
8	100006	运单号为空	运单号参数为空
9	100009	定位权限未开启	手机未开启定位权限
10	100010	未开启持续定位	定位的时候有选项，定位需选择始终允许
11	100011	起点行政区划代码为空	起点行政区划代码为空
12	100012	终点行政区划代码为空	终点行政区划代码为空
13	100021	起运地经纬度为空	start/restart/pause/stop 等方法

序号	代码	信息	说明
			调用的时候需要传入起运地经纬度
14	100022	收货地经纬度为空	start/restart/pause/stop 等方法调用的时候需要传入起点经纬度
15	100023	起运地详情地址为空	起运地详情地址为空
16	100024	收货地详情地址为空	收货地详情地址为空
17	100025	无法获得设备 ID	sdk 无法从该设备获取设备 ID，请更换设备
18	100026	调用发送位置接口频繁	发生在 app 调用 sdk send 接口太过频繁，请按前面文档中所述要求调用 send 接口
19	100027	车牌信息为空	车牌参数为空
20	100028	驾驶员姓名为空	驾驶员姓名参数为空
21	100029	车牌信息超长	车牌参数长度超过长度限制
22	100030	分单号超长	分单号参数长度超过长度限制
23	100031	行政区划代码超长	行政区划代码参数超过长度限制
24	100032	经纬度超长	经纬度参数超过长度限制
25	100043	经纬度小数点位数超长	经纬度参数小数点后位数超过长度限制



序号	代码	信息	说明
26	110000	定位超时	检查定位权限，重新调用此接口
27	110001	定位失败	检查定位权限，重新调用此接口，或者检查是否超过高德百度平台允许的调用次数。
28	888882	App 传递运单信息为空	调用 start 等接口传入的 ShippingNoteInfo 数组为空
29	888883	运单数超过限制	累计传入运单数目超过最大允许运单数量
30	888884	有未开始的运单	在没有调用 start 接口情况下调用了 send,stop 等接口。
31	888888	未授权错误	在没有调用 auth 接口的情况下调用 start 等接口。