二〇二一年九月

**[签里眼视频面签系统Pure for iOS]**

**iOS版本对接文档**

**签里眼视频面签系统Pure**

**目录**

[一、 SDK功能 2](#_Toc28986)

[1.1 远程面签 2](#_Toc3026)

[1.2 自助面签 2](#_Toc13712)

[1.3 AI面签 3](#_Toc21495)

[1.4 现场面签 3](#_Toc19787)

[1.5 远程协助 4](#_Toc11627)

[二、开发前准备 4](#_Toc2983)

[三、配置流程 4](#_Toc3150)

[3.1 添加依赖框架、静态库、动态库 4](#_Toc12980)

[3.2 添加权限 5](#_Toc5919)

[3.3 设置 5](#_Toc12542)

[3.4 依赖的三方库 6](#_Toc25648)

[四、接口调用 6](#_Toc1431)

[4.1初始化方法 6](#_Toc30744)

[4.2 调起面签服务： 6](#_Toc17765)

[4.3 调起上传未上传视频的服务 7](#_Toc13376)

[4.4 需要实现的代理方法 7](#_Toc21466)

[ 文档更新记录 10](#_Toc7993)

# SDK功能

本SDK集成了多种视频面签的类型，您可以按照实际的需求和应用场景进行选择对接，如有问题请随时与我司技术对接。

## 1.1 远程面签

远程视频面签需要PC端坐席的支持，手机端APP作为用户端，发起视频面签，远程坐席点击接通相关呼叫，即可创建音视频房间进行面签业务处理，此种方式面签需要PC端H5SDK以坐席的身份进行接通呼叫。

实现的相关功能点如下：

* 音视频通话
* 云端录制
* 活体人脸核身（三方SDK模式）
* 拍照留存

## 1.2 自助面签

自助面签仅由用户自己即可完成整个面签的流程，用户按照指引开启自助面签后，可以自动播放一段产品简介或风险揭示类型的视频，然后客户按照指引朗读声明内容，并按照要求进行拍照留存，合同签名等，面签结束后上传视频至云端。

实现的相关功能点如下：

* 音视频通话
* 本地录制
* 播放产品简介或风险揭示视频
* 朗读声明
* 拍照留存
* 合同签名（可以录制）
* 防出框比对（待实现）

## 1.3 AI面签

AI面签通过机器人提问的形式取代真实的坐席，从后台抓取配置好的话术流程，基于客户的业务需求，通过TTS和ASR相关技术，进行问题提问和语义解析，并根据客户的回答内容进行判断以及后续流程节点的流转提问来实现整个面签过程的去人工化和自动化，并将全过程进行音视频录制留存。

实现的相关功能点如下：

* 音视频通话
* 云端录制
* 活体人脸核身（三方SDK模式）
* ASR语义解析
* TTS语音合成
* 防出框比对
* 动态话术配置（含证件托举）
* 合同签名

## 1.4 现场面签

现场面签是APP到APP的一个视频面签，基于客户经理在现场，但是又需要机械的核实一些固定问题，通过机器人播报提问来取代大部分客户经理待核实的问题，进行整个流程的录制留存。

实现的相关功能点如下：

* 音视频通话
* 云端录制
* ASR语义解析
* TTS语音合成
* 防出框比对
* 动态话术配置（含证件托举）

## 1.5 远程协助

远程协助

* 音视频通话
* 云端录制

# 二、开发前准备

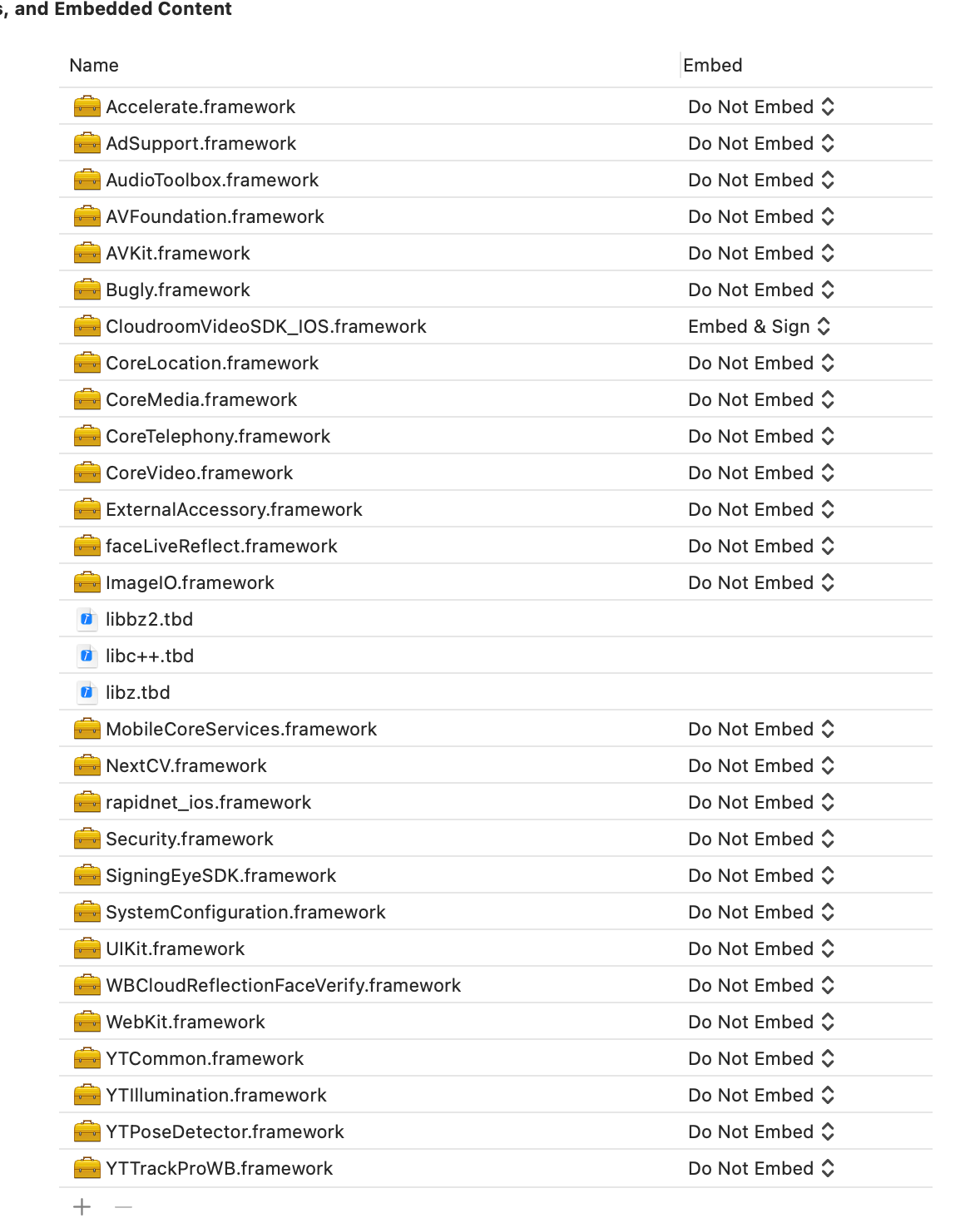
1. 请联系相关人员开通租户权限，获取【企业账号】。
2. 在【签里眼视频面签业务管理系统】对队列进行新增管理，拿到队列编号。
3. （按需提供）发起人脸识别需要姓名、证件号、大头照的base64（可有可无）。如果业务系统已经有了人脸识别。需要提供人脸识别的结果和大头照的base64。
4. 自助面签需要提供播放的视频文件地址，朗读声明的文字。

# 三、配置流程

注意： 以Xcode作为集成SDK实例工具， 集成流程如下：

## 3.1 添加依赖框架、静态库、动态库

SDK 依赖以下系统框架，需要在【Build Phases】>【Link Binary With Libraries】中添加，可以参考 Demo，具体依赖如下：



## 

## 3.2 添加权限

SDK 需要使用相机、相册和录音权限，请在 info.plist 中添加：

Privacy - Microphone Usage Description

Privacy - Camera Usage Description

Privacy - Location When In Use Usage Description

Privacy - Photo Library Usage Description

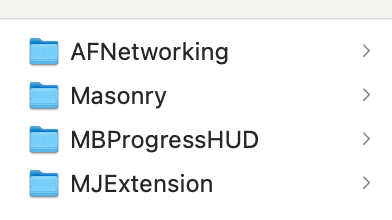
Supports opening documents in place = YES (documents 文件可访问)

Application supports iTunes file sharing = YES (documents 文件可访问)

## 3.3 设置

1. 需要在【BuildSettings】>【Other Linker Flags】中设置：-ObjC
2. 需要在【BuildSettings】> 【Enable Bitcode】中设置为 NO

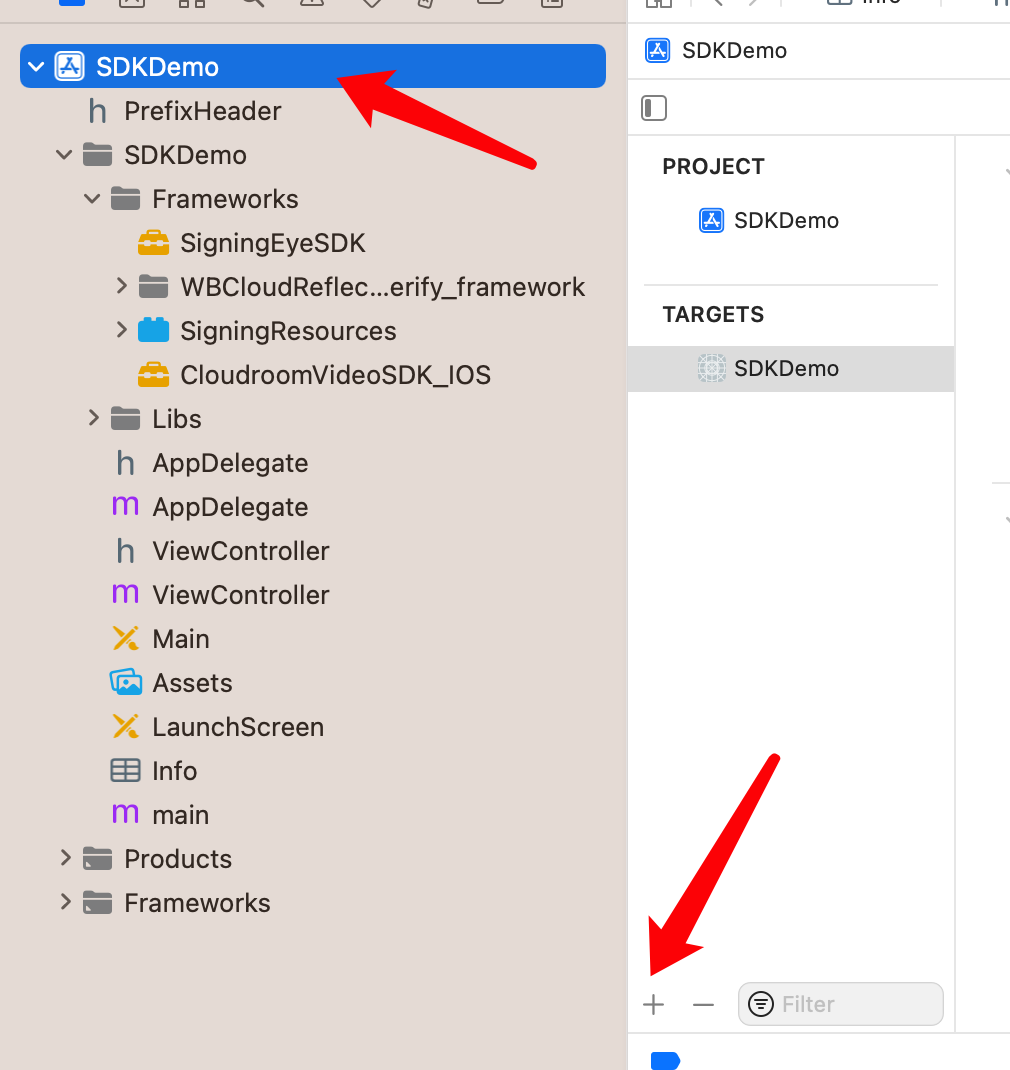
## 3.4 依赖的三方库

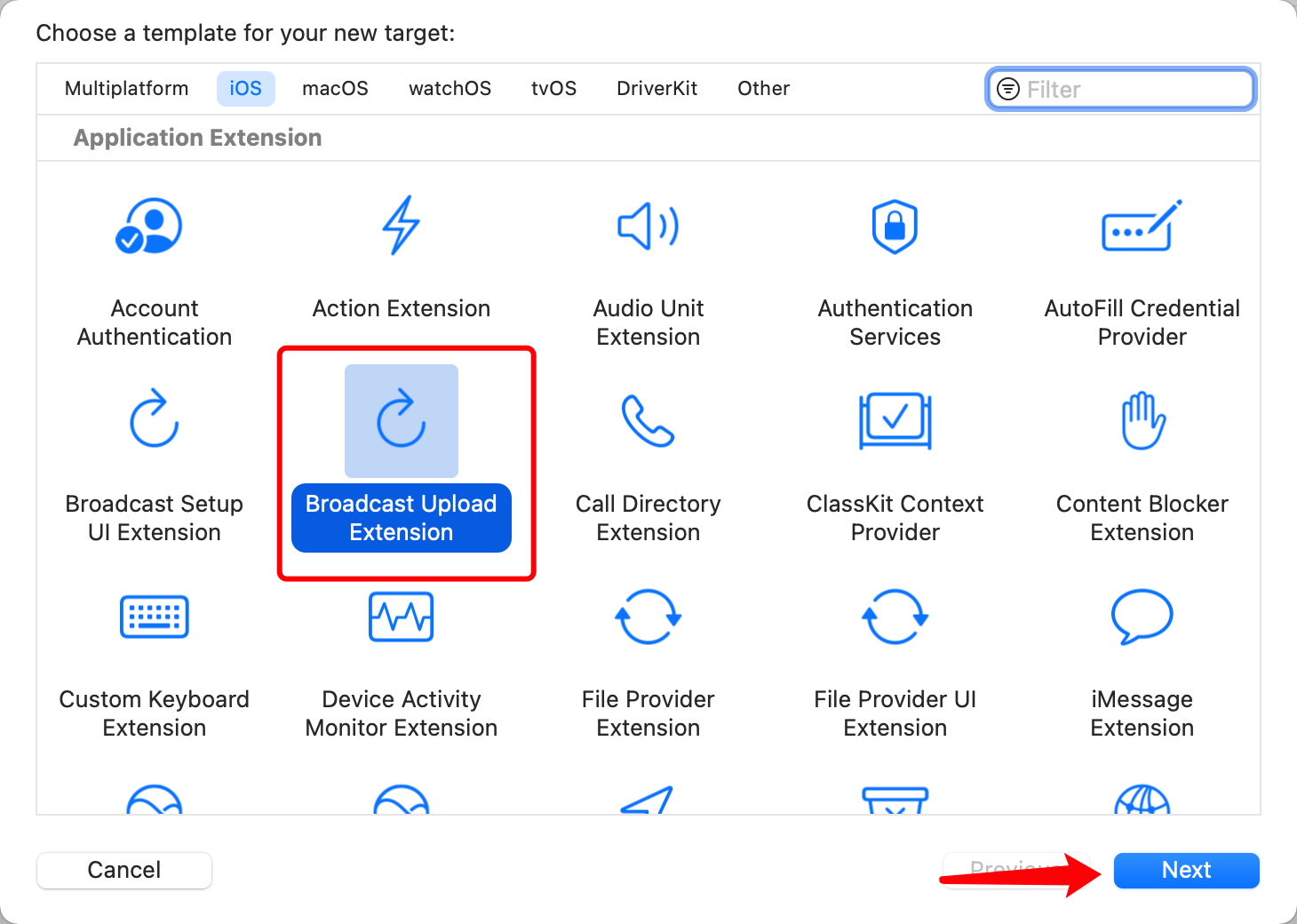


**3.5 需要屏幕共享功能相关配置(**远程协助模式**)**

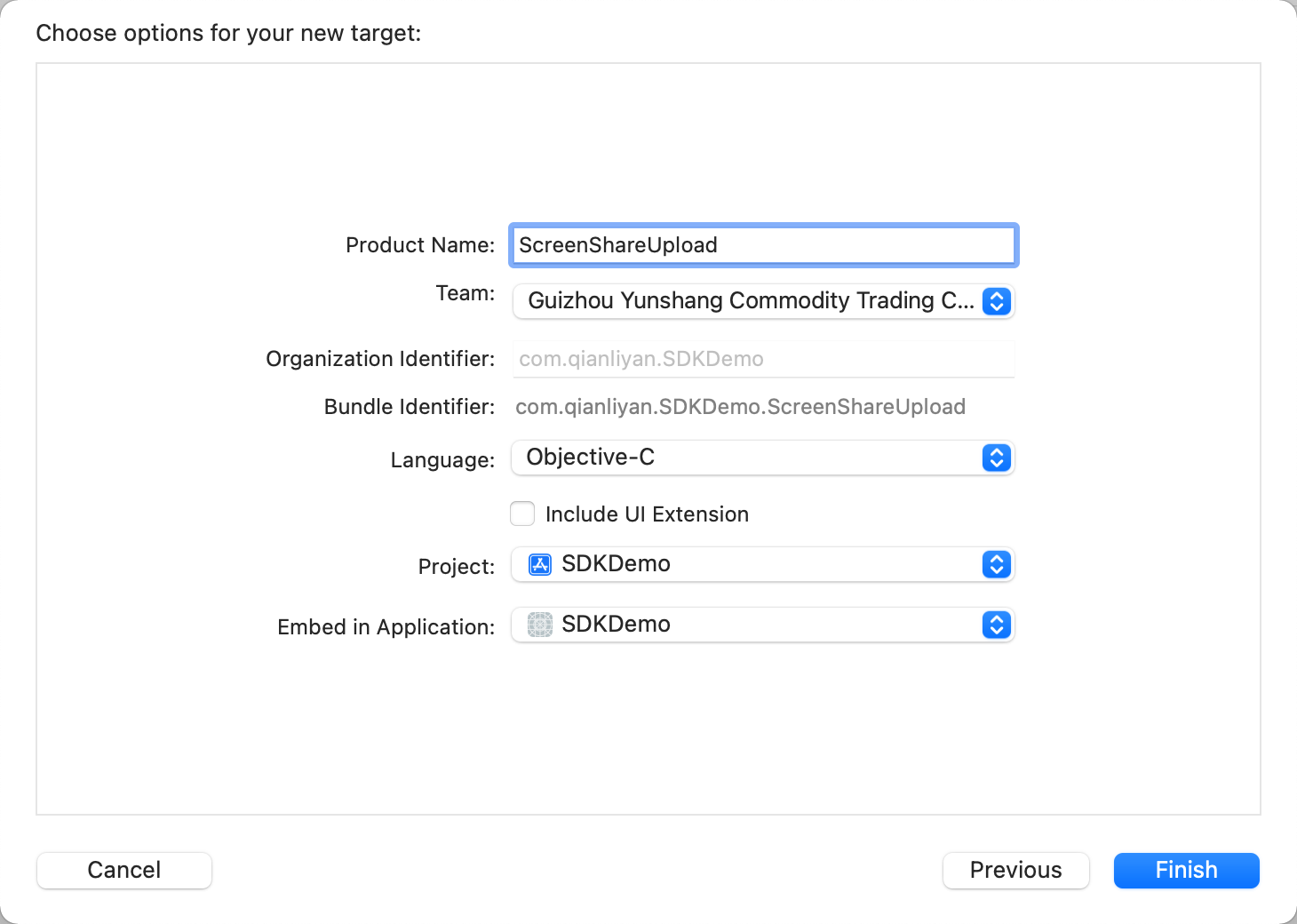
注意:设备系统在iOS12.0以上才支持本功能

1.创建屏幕录制子进程，进程名字为ScreenShareUpload



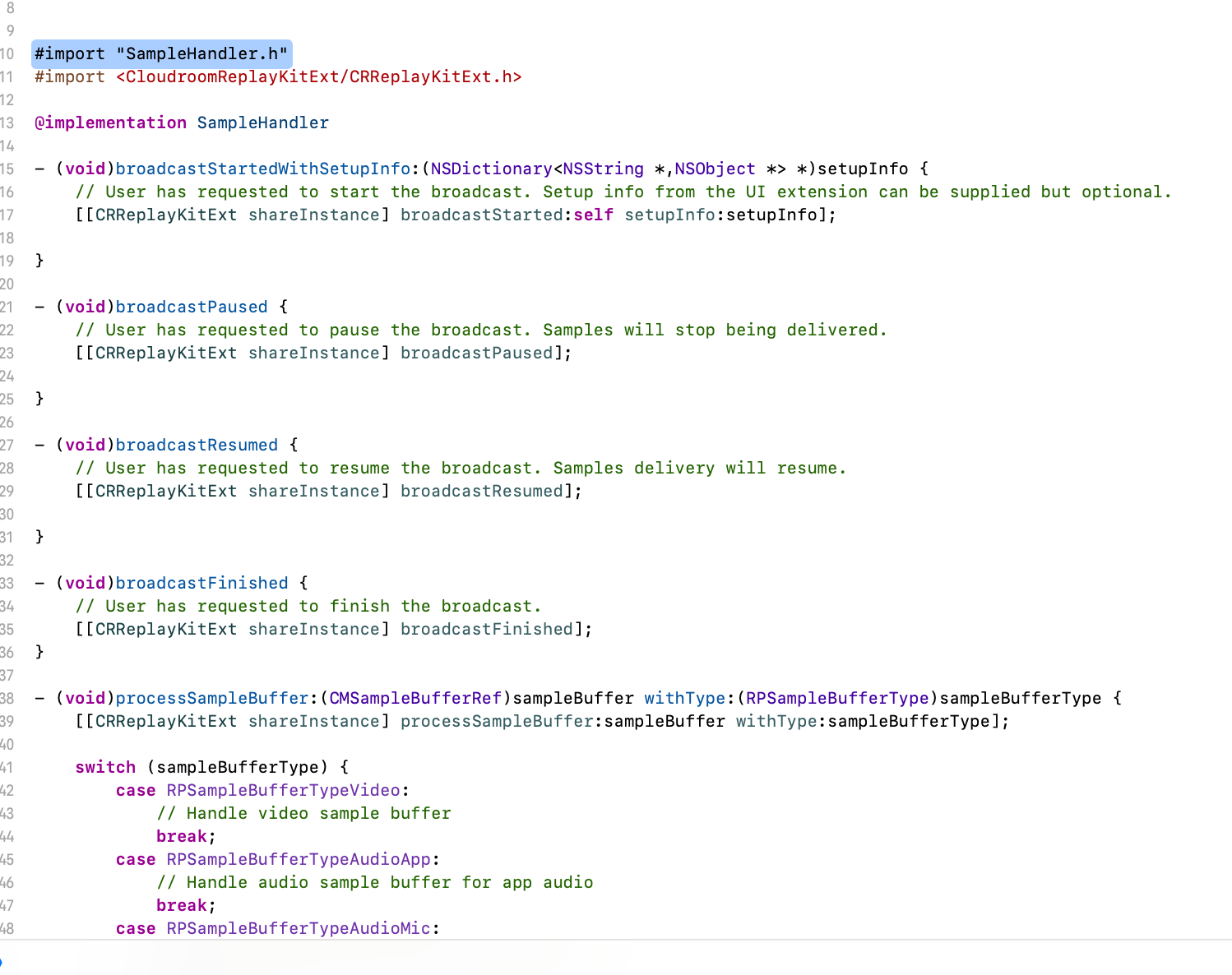


命名为ScreenShareUpload

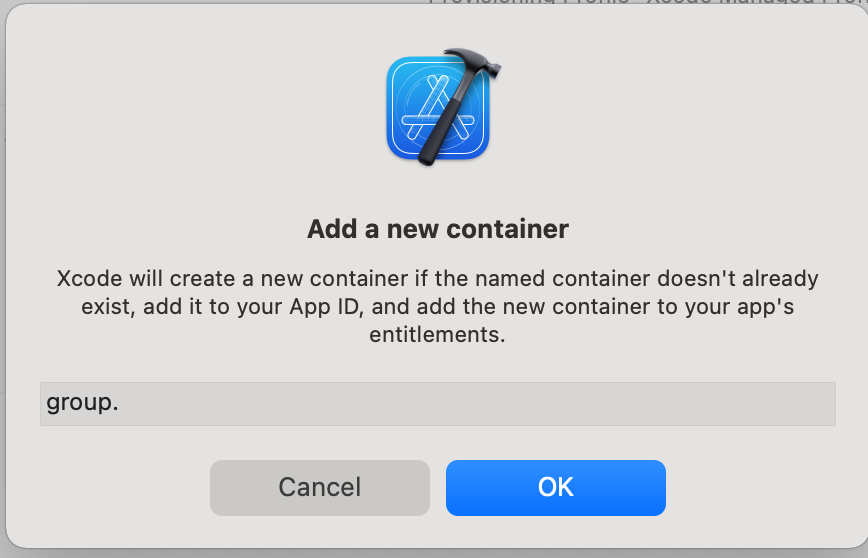


在屏幕录制子进程中引用CloudroomReplayKitExt.framework , ReplayKit.framework

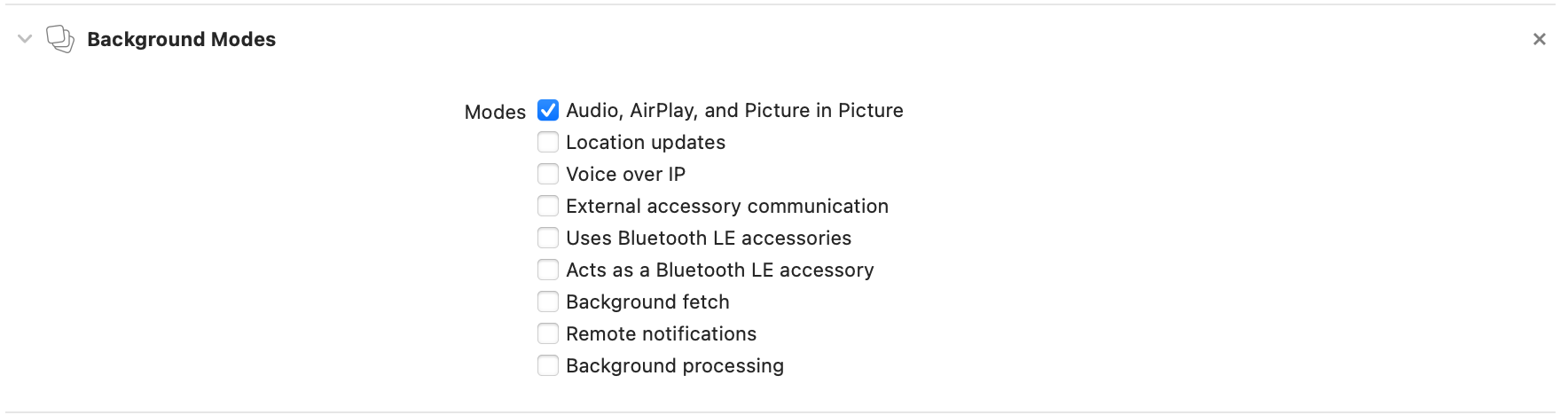
SampleHandler中添加调用SDK接口



在屏幕录制子进程与主进程中添加APP Group，名字为：group.+主进程BundleIdentifier



主进程中添加后台运行申请



# 四、接口调用

## 4.1初始化方法

在 appdelegate 的didFinishLaunchingWithOptions 方法中调用SDK初始化的方法。

[[SigningESDK sharedInstance]initSDK];

[SigningESDK sharedInstance].delegate = self;

可以在代理方法中查看初始化结果：

#pragma mark - SigningESDKDelegate

-(void)SigningESDKinitResult:(SEInitResult \*)result{

}

## 4.2 调起面签服务：

SEVideoManager \*manager = [SEVideoManager sharedInstance];

manager.delegate = self;

/\*\*

调起服务

@param tenantID 企业账号，需要在后台申请

@param contractID辅助编号

@param businessID 业务编号

\*/

[manager videoWithTenantID:tenantID contractID:contractID businessID:businessID];

## 4.3 调起上传未上传视频的服务

/\*\*

上传视频

@param videoName 视频名称

@param tenantID 企业账号

@param contractId 辅助编号

@param businessID 业务编号

\*/

[[SEVideoManager sharedInstance]uploadVideoWithVideoName:videoName tenantID:tenantID contractId:businessID businessID:businessID];

## 4.4 需要实现的代理方法

### 4.4.1是否继续视频面签的服务

-(BOOL)SEVideoServiceContinue;

### 4.4.2人脸识别的数据从这个代理方法中取

/\*\*

1、如果业务系统不需要人脸识别的功能，

姓名 name 和 证件号 非必传

2、如果需要人脸识别的功能，姓名 name 和证件号 idNo 必传，

人脸识别结果faceVerifyResultType，sourcePhotoStr 为非必传，

sourcePhotoStr参数有值：使用合作伙伴提供的比对源照片进行比对，必须注照片是正脸可信照片，照片质量由合作方保证。参数为空 ：根据身份证号 + 姓名使用权威数据源比对

SEFaceVerifyData \*faceData = [[SEFaceVerifyData alloc]init];

faceData.idNo = @"130422199201111111”;//证件号

faceData.name = @"张三”;//姓名

faceData.faceVerifyResultType = SEFaceVerifyResultTypeSuccess;

faceData.sourcePhotoStr = @"/9j/4AAQSkZJRgAB……”;// BASE64String 比对源照片，注意：原始图片不能超过 500k，且必须为 JPG 或 PNG 格式。

\*/

-(SEFaceVerifyData \*)SEVideoServiceWithFaceVerifyData;

### 4.4.3 返回人脸识别的结果

/// @param faceVerifyResult 人脸识别的结果

-(void)SEVideoServiceWithFaceVerifyResult:(SEFaceVerifyResult \*)faceVerifyResult;

### 4.4.4 视频面签的数据

-(SEInterviewData \*)SEVideoServiceWithInterviewData;

1. 公共数据

SEInterviewData \*interViewData = [[SEInterviewData alloc]init];

interViewData.userName = 用户唯一标识(必传)

interViewData.tenantName = @”公司名称”//(必传)

1. 远程面签参数

interViewData.queueID = 队列编号

interViewData.remotelyRiskVoideUrl =远程风险视频地址

1. 自助面签参数

  interViewData.statementStr =朗读文字

  interViewData.videoArray = @[@”风险视频地址”];

interViewData.contractAddress = 合同地址

interViewData.faceComparedImage = 人脸比对图base串(人脸识别>默认比对图>人脸比对图片)

interViewData.interviewType = SEInterviewTypeAuto //面签方式(必传)

1. AI自助面签

interViewData.wordCode = 话术id

interViewData.contractAddress = 合同地址

interViewData.pramas = 话术id需要参数(key value,json串)

interViewData.isLiftWord = @”1” //1是有证件托举  0是没有证件托举

interViewData.interviewType = SEInterviewTypeAiAuto //面签方式(必传)

1. 现场面签坐席端

interViewData.queueID = 服务队列编号

interViewData.interviewType = SEInterviewTypeoSeat //面签方式(必传)

1. 现场面签用户端

interViewData.queueID = 队列编号

interViewData.wordCode = 话术id

interViewData.pramas = 话术id需要参数(key value,json串)

interViewData.interviewType = SEInterviewTypeoClient

1. 远程协助坐席端

interViewData.queueID = 队列编号

interViewData.interviewType = SEInterviewTypeoAssistanceService

//面签方式(必传)

1. 远程协助用户端

interViewData.queueID = 队列编号

interViewData.shareWindow = app.window2;//副窗口

interViewData.rootWindow = app.window;//主窗口

interViewData.interviewType = SEInterviewTypeoAssist //面签方式(必传)

AppDelegate \* app = (AppDelegate \*)[[UIApplication sharedApplication] delegate];

interViewData.shareWindow = app.window2;

interViewData.rootWindow = app.window;

在AppDelegate的中创建一个UIWindow 窗口

@property (nonatomic, strong)UIWindow \*window2;

在didFinishLaunchingWithOptions初始化UIWindow 窗口

self.window2 =[[UIWindow alloc] initWithFrame:[UIScreen mainScreen].bounds];

### 4.4.5视频服务结束的方法

-(void)SEVideoServiceDidFinishedWithResult:(SEVideoResult \*)videoResult;