

# eSDK ICP V200R001C10 开发指南 01(CC, iOS)

文档版本 01  
发布日期 2017-03-17

华为技术有限公司



**版权所有 © 华为技术有限公司 2017。 保留一切权利。**

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

## 商标声明



HUAWEI 和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

## 注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

## 华为技术有限公司

地址：                深圳市龙岗区坂田华为总部办公楼                邮编：518129

网址：                <http://www.huawei.com>

客户服务邮箱：      [support@huawei.com](mailto:support@huawei.com)

客户服务电话：      4008302118

# 目 录

<b>1 eSDK CC 是什么?</b>	<b>1</b>
<b>2 内容导航</b>	<b>2</b>
<b>3 相关资源</b>	<b>3</b>
3.1 华为开发者社区	3
3.2 Sample Codes	3
3.3 SDK 下载路径	4
3.4 接口参考	4
3.5 SDK 修订记录	4
3.6 免费申请远程实验室	5
3.7 技术支持渠道	7
<b>4 Hello World</b>	<b>8</b>
4.1 概述	8
4.2 准备环境	9
4.3 新建工程项目	10
4.4 将 SDK 导入工程	11
4.5 初始化工程配置	13
4.6 编码实现	16
4.7 调试运行	16
<b>5 典型场景开发</b>	<b>18</b>
5.1 场景一：登录与登出	18
5.2 场景二：TP 环境下的呼叫	21
5.3 场景三：MS 环境下的呼叫	24
5.4 场景四：设备控制	31
<b>6 问题定位</b>	<b>35</b>
<b>7 修订记录</b>	<b>37</b>
<b>8 附录</b>	<b>38</b>
8.1 Hello World 源码文件	38
8.1.1 ViewController.m	38

# 1 eSDK CC 是什么？

## eSDK ICP 解决方案简介

华为 ICP（Integrated Communication Platform，集成通信平台）融合指挥解决方案通过丰富的开放接口，可以与火警、匪警、急救、视频监控、无线电通讯系统、公共电话系统、CAD（计算机辅助调度）业务系统等对接，用于公众报告紧急事件和紧急求助，统一接警，统一指挥，联合行动，为市民提供相应的紧急救援服务，为城市的公共安全提供强有力的保障，大力加强了各警力单位之间的配合与协调，从而对特殊、突发、应急和重要事件做出有序、快速而高效的反应。

ICP 集成通信平台作为融合指挥的核心，实现多语音网络、多终端的语音、多视频系统的全连接，实现固定电话、移动电话、各种集群终端、智真会议等不同通信设备之间的互联互通，通过多种网络融合的语音通信，实现对应急事件处置的统一指挥调度和应急决策信息的快速传达。

## CC SDK 是什么

CC SDK 作为 eSDK ICP 解决方案中的一个子模块，提供呼叫，呼叫设备控制等功能。

我们提供给你的 SDK 包内容有以下几部分内容：

- **CC SDK 包**

二次开发使用的 SDK 包，包括软件包和接口参考文档，详情请参见 3.3 SDK 下载路径。

- **Sample Codes**

华为 SDK 提供一系列 Sample codes 演示如何调用接口，帮助您完成 eSDK CC iOS 相关业务开发。详情请参见 3.2 Sample Codes。

# 2 内容导航

本开发指南指导您如何安装和配置 eSDK CC iOS 环境、如何调用 eSDK CC iOS 标准化接口以及华为 eSDK 提供的开发者支持服务。主要内容如下：

1. 3 相关资源：二次开发过程中可能用到的软件、文档资源链接、技术支持。包括如何通过社区网站获取资料，sample codes 下载链接，如何申请远程实验室等。
2. 4 Hello World：如果你仅仅是要把 SDK 运行起来，你应该首先看这部分，它会详细说明如何下载及安装 SDK，以及如何配置你的开发环境。
3. 5 典型场景开发：介绍 eSDK CC 开放性能的典型功能场景，包括开发流程、样例代码以及注意事项等。
4. 6 问题定位：开发过程中常见问题的定位解决方法。

## 阅读建议

- 如果只是想快速入门，可仅参考 4 Hello World 章节。
- 如果想深入了解 eSDK CC iOS 核心业务的二次开发，建议您参考 5 典型场景开发章节。
- 使用 SDK 过程中遇到问题可以参考 6 问题定位章节或 3.7 技术支持渠道。

# 3 相关资源

- 3.1 [华为开发者社区](#)
- 3.2 [Sample Codes](#)
- 3.3 [SDK 下载路径](#)
- 3.4 [接口参考](#)
- 3.5 [SDK 修订记录](#)
- 3.6 [免费申请远程实验室](#)
- 3.7 [技术支持渠道](#)

## 3.1 华为开发者社区

您可通过[华为开发者社区企业云通信版块](#)，快速体验 eSDK ICP 业务功能，获取 eSDK ICP 的二次开发 SDK 工具包以及技术支持等。

## 3.2 Sample Codes

这里建议您使用 Xcode 编译执行 Sample Codes。  
如果需要下载及安装 SDK，请参考 4 Hello World。

### Sample Codes

名称	描述
eSDK_ICP_Demo_V2.1.10_iOS.zip	基于 eSDK CC iOS Sample 包括 TP 和 MS 环境下呼叫，设备控制。

## 3.3 SDK 下载路径

进入[华为开发者社区资源中心](#)，点击“**SDK > 企业云通信 > ICP > eSDK CC SDK**”，选择对应版本进行下载。

推荐使用最新的版本 **V2.1.10**。

## 3.4 接口参考

接口参考主要包含：

**概述：**明确配套版本、使用背景、场景、前提条件等内容；并指出可以获取哪些信息，完成什么样的功能。

**数据类型：**详细说明 eSDK 对外提供的自定义的数据类型（包括数据结构、枚举、类等），例如：

- 数据类型名。
- 对有继承/嵌套关系的数据结构进行说明，如结构体嵌套关系，要提供总的嵌套关系表（要包括基本数据类型）。
- 包括的数据成员以及数据成员的含义。

**接口详细描述：**

- 接口描述：详细描述该接口功能、应用场景和使用方法。
- 使用说明：描述该接口函数使用时需要注意的事项、使用的限制；功能相似的接口或者需要配合使用的接口；前提条件等。
- 方法定义：接口函数的完整声明。
- 参数描述：详细描述参数的含义、取值范围、使用限制、参数之间的依赖等。
- 返回值：说明该接口函数的返回值。
- 使用示例：举例说明该接口函数的使用，关键代码需要注释。

## 3.5 SDK 修订记录

SDK 会逐渐的升级来支持新的服务。您可以参考开发者社区网站历史版本了解不同版本有哪些更新。具体的更新信息包含：

- SDK 名称
- 产品名称：SDK 配套的产品名称。
- 更新时间：SDK 更新发布上线的时间。
- 版本号：SDK 当前的版本号。
- 下载链接：链接中包含 SDK Demo 及配套文档。
- 更新记录描述：描述变更的特性。

## 3.6 免费申请远程实验室

### 华为 eSDK 远程实验室简介

华为远程实验室为开发者提供了 7×24 小时的免费的云化实验室环境，提供真实的华为设备供开发者远程在线开发调试。借助远程实验室自助管理平台，开发者不需要购置华为产品便可基于远程实验室针对相关产品进行二次开发，并实现远程对接测试认证。目前华为远程实验室已发布云计算、SDN、大数据、企业云通信、企业移动安全等 10 个生态圈的 45 个实验室环境。

相关信息请查询[远程实验室首页](#)。

### 远程实验室有哪些优势

- 低门槛：官网注册用户即可申请使用，环境与预约时长、预约次数受限；
- 分级支持：环境按域划分，重点开发者、合作伙伴可访问特定环境并享受额外的预约环境；
- 全球资源高速互联：建设以苏州远程实验室为中心的实验室分布格局，依托 IT 全球 100ms 高性能骨干网络和 IT 全球端到端应用保障能力。

### 如何免费申请远程实验室



#### 步骤 1 注册账号。

1. 若您已有华为注册账号，则直接在登录远程实验室首页使用已注册账号登录。
2. 若您还未注册华为账号，请进入华为远程实验室网站：  
<http://developer.huawei.com/cn/ict/remotelab>，点击“**进入实验室**”，页面跳转至账号注册页，填写注册信息，并在注册邮箱中激活您的华为账号。



#### 步骤 2 预约环境。

1. 若您已成功预约环境，且环境可用时长在有效期范围内，则请忽略此步骤。



2. 华为账号注册成功后，系统会自动跳转到华为远程实验室首页，状态为已登录。使用 CC 调试环境 Contact Center，点击“预约”按钮，默认预约的环境可用时长为 2 小时，如下图所示。



3. 预约成功后，系统会自动发送 SVN 网关地址、用户名、密码等信息至您的华为账户注册邮箱中，根据邮件中的提示信息下载 SSL-VPN 客户端软件，完成本地安装。后续步骤需要使用该信息登录 SVN 客户端进行远程环境连接。

#### 步骤 3 接入环境。

1. 若您成功已接入华为远程实验室环境，请忽略此步骤。
2. 打开 SVN 客户端，将预约成功后系统发送给您的 SVN 网关地址、用户名、密码等信息，分别输入到客户端对应的输入框内，并点击“登录”。



#### 步骤 4 调测发布。

使用申请成功的 CC 平台账号、密码、IP 地址、Port 等信息，完成登录，调测您的应用程序。详见远程实验室操作引导中的“CC 平台登录信息”。

----结束

## 3.7 技术支持渠道

### 技术支持渠道

如果您在软件开发过程中碰到任何问题，开发者社区提供了以下技术支持渠道：

- 可以在[华为开发者论坛](#)发帖求助。
- 可以在 [DevCenter](#) 系统中提单跟踪。

如果您在文档使用过程中有任何疑问，可以通过以下方式联系我们。

- 华为技术支持热线电话：400-8828-000
- 华为技术支持邮箱：esdk@huawei.com

# 4 Hello World

- [4.1 概述](#)
- [4.2 准备环境](#)
- [4.3 新建工程项目](#)
- [4.4 将 SDK 导入工程](#)
- [4.5 初始化工程配置](#)
- [4.6 编码实现](#)
- [4.7 调试运行](#)

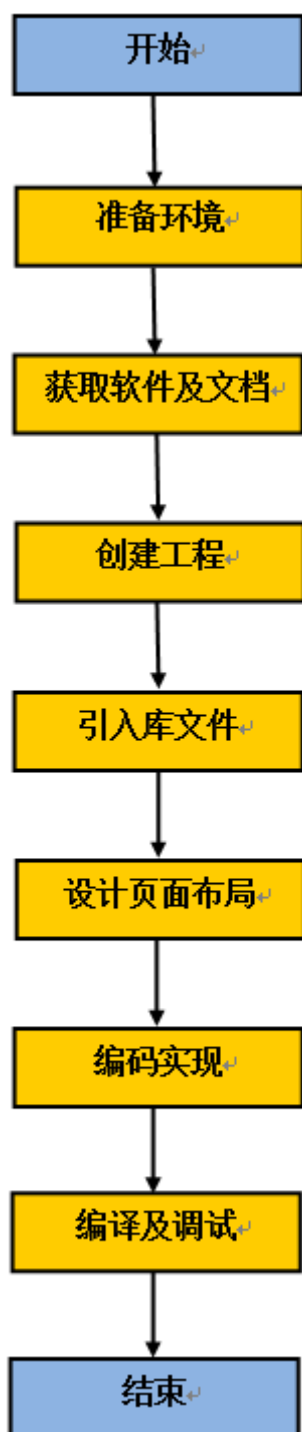
## 4.1 概述

### Hello World 开发流程

本示例以 object-c 语言简单向您展示进行 eSDK CC iOS 的二次集成开发。

开发过程中的故障处理请参考 6 问题定位

快速入门流程如下：



## 4.2 准备环境

### 开发工具

- 操作系统：MAC OS 10.9 Mavericks 及以上。

- XCode 8.0 及以上版本。
- iOS 8.0 及以上版本。

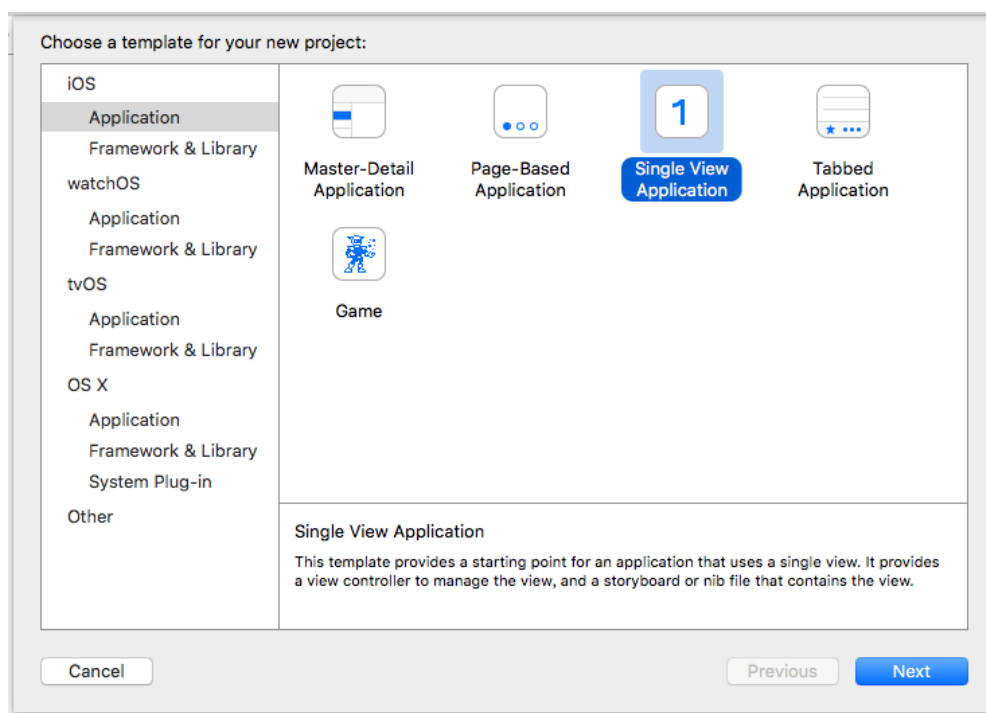
## SDK 软件包

- SDK 软件包名称：**eSDK\_ICP\_SDK\_V2.1.10\_iOS.zip**。
- SDK 软件包下载路径：参见 3.3 SDK 下载路径。

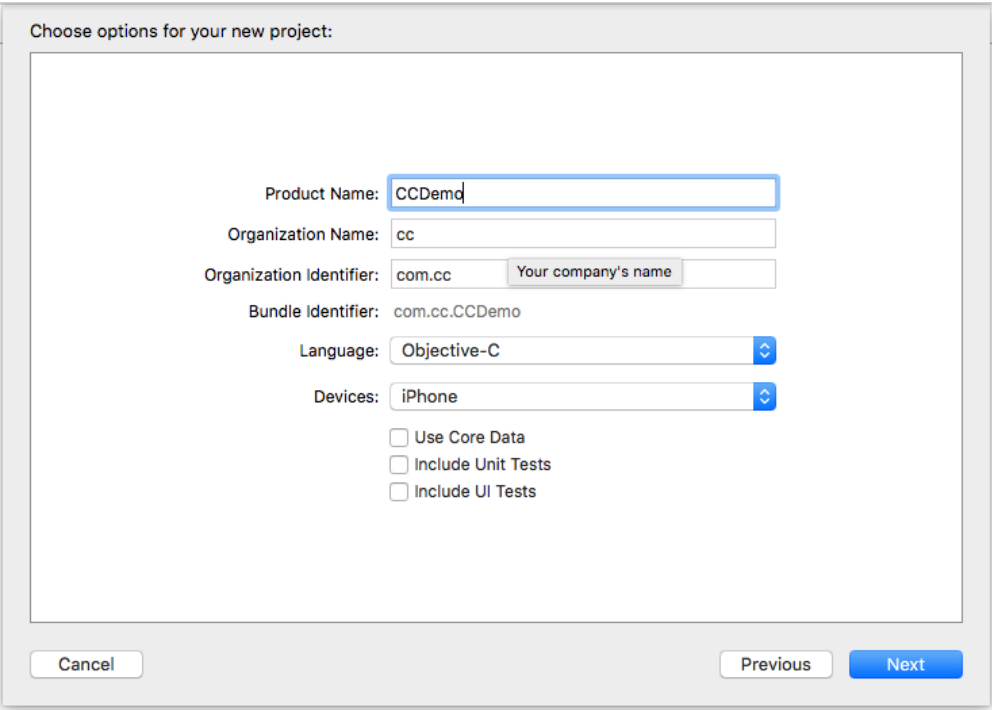
如果需要证书验证功能，需要将证书放在工程中 assets 文件夹下，例如“assets/certs/server.cer”。

## 4.3 新建工程项目

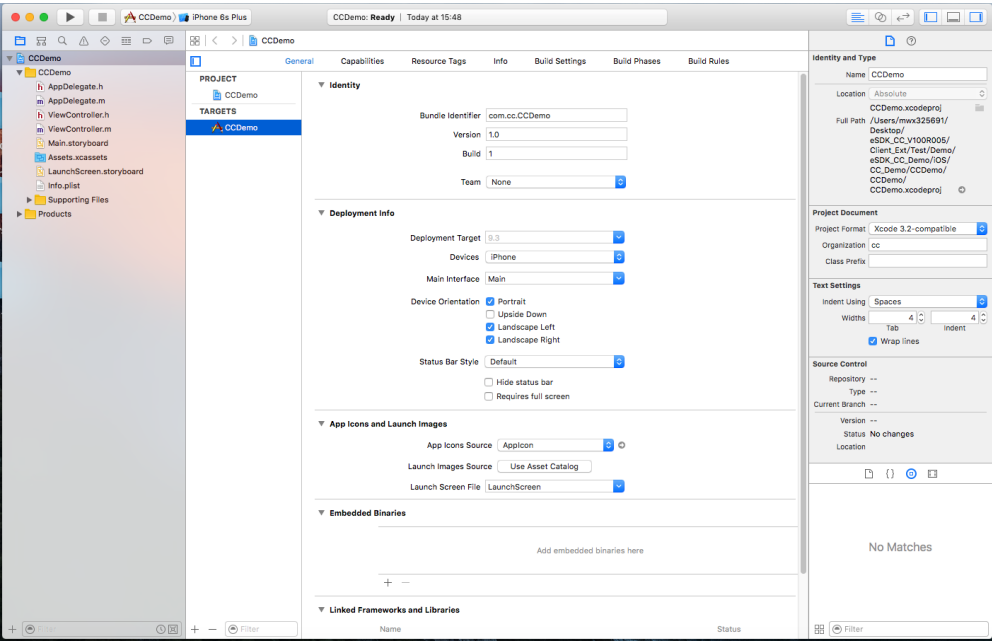
步骤 1 打开 Xcode，选择“File > New > Project”，如下图所示，点击 Next。



步骤 2 填写 Product name、Organization Name、Organization Identifier 信息后，如下图所示，点击“Next”。



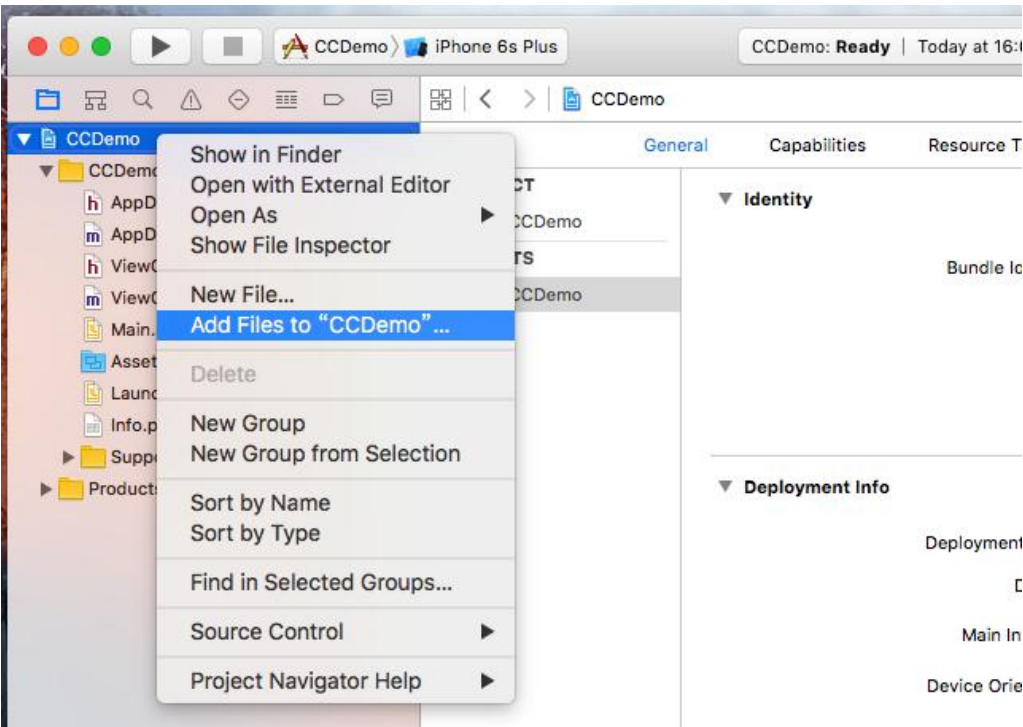
步骤 3 创建的工程如图所示。



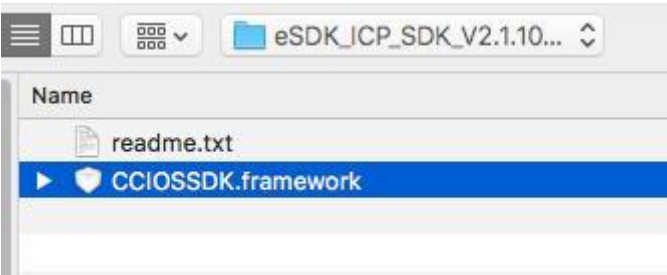
----结束

## 4.4 将 SDK 导入工程

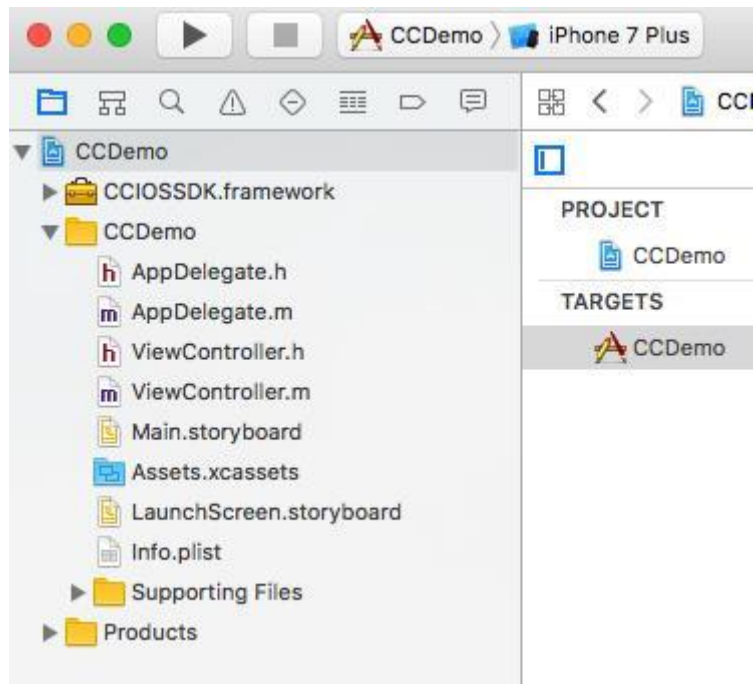
步骤 1 右击如下图所示,点击 Add Files to。



步骤 2 选中 SDK 文件夹，如下图所示。



步骤 3 选中 SDK，点击 Add，添加后如下图所示。



----结束

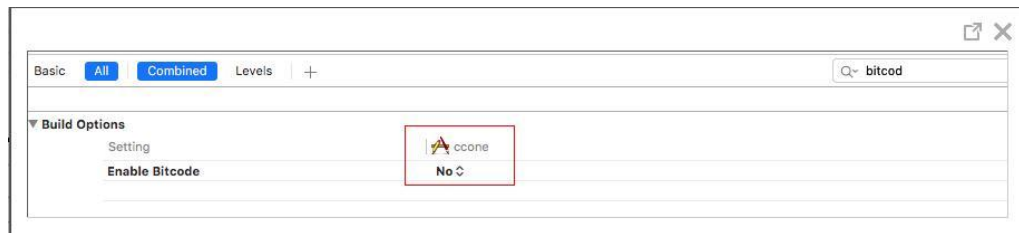
## 4.5 初始化工程配置

**步骤 1** 为了确保能登录成功，选择工程主界面 Navigator 下 “Info.plist” 文件，右击任意一行，选中 “Add Row” （可以下拉挨个查找，也可以直接输入名称 “App Transport Security Settings” ），在 item0 的 Value 处，输入 “Allow Arbitrary Loads” 并将 Boolean 值设为 YES。然后次加入设置 “Privacy - Camera Usage Description” 、 “Privacy - Contacts Usage Description” 、 “Privacy - Microphone Usage Description”

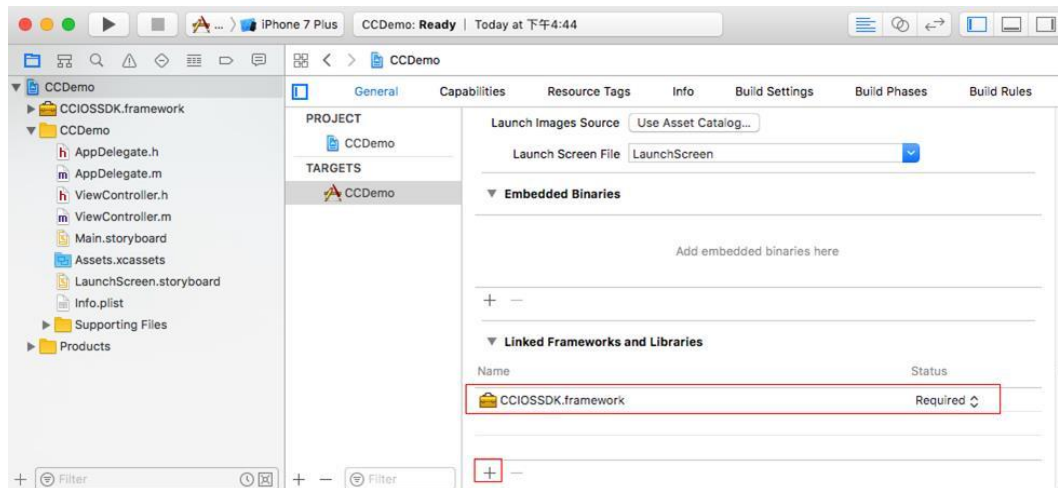
Key	Type	Value
Information Property List	Dictionary (18 items)	
Localization native development re...	String	en
Executable file	String	\$(EXECUTABLE_NAME)
Bundle identifier	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle version	String	1
Application requires iPhone enviro...	Boolean	YES
App Transport Security Settings	Dictionary (1 item)	
Allow Arbitrary Loads	Boolean	YES
Privacy - Camera Usage Description	String	cameraDescription
Privacy - Contacts Usage Descript...	String	contactsDescription
Privacy - Microphone Usage...	String	microphoneDescription
Required background modes	Array (2 items)	
Item 0	String	App plays audio or streams audio/video using AirPlay
Item 1	String	App provides Voice over IP services
Launch screen interface file base...	String	LaunchScreen
Main storyboard file base name	String	Main
Required device capabilities	Array (1 item)	
Supported interface orientations	Array (1 item)	

**步骤 2** 设置 Bitcode，点击 Navigator 下的工程名称，“Build Setting ->Enable Bitcode”，改为 NO。

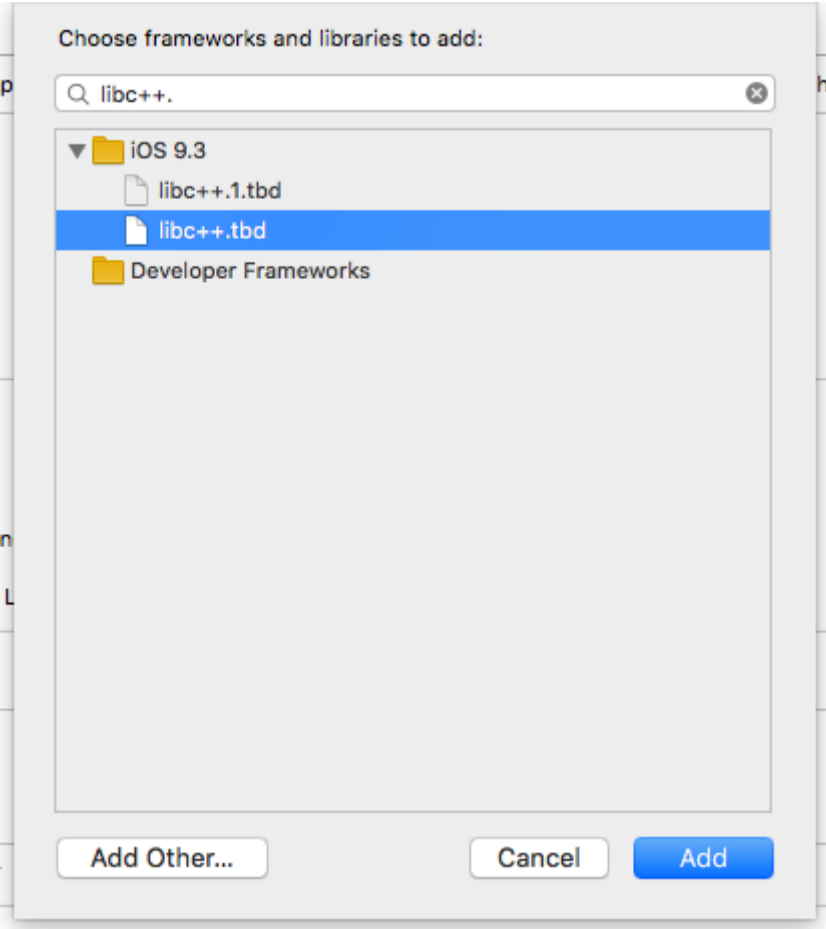




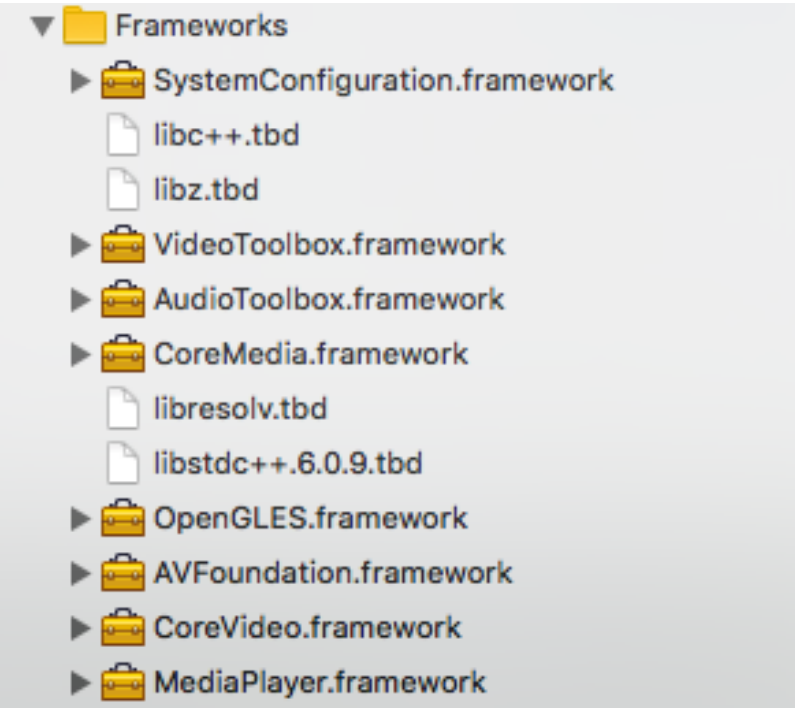
步骤 3 点击 Navigator 下的工程名称，“General->Linked Frameworks and Libraries”依次点击图中“+”弹出添加依赖库文件的对话框。



步骤 4 在搜索框中查找“libc++.tbd”，选中并单击“Add”。



步骤 5 重复以上步骤，分别添加下图 依赖框架所示的所有依赖框架。



----结束

## 4.6 编码实现

### 总体结构

```
CCDemo
----CCDemo
-----ViewController.h
-----ViewController.m
-----Main.storyboard
-----AppDelegate.h
-----AppDelegate.m
----CCIOSSDK.framework
```

源码链接：[8.1 Hello World 源码文件](#)。

### 代码参考

点击上述结构中的文件名获取文件源码，并向各文件填充代码。部分代码片段参考如下：

#### 注册登录与登出通知

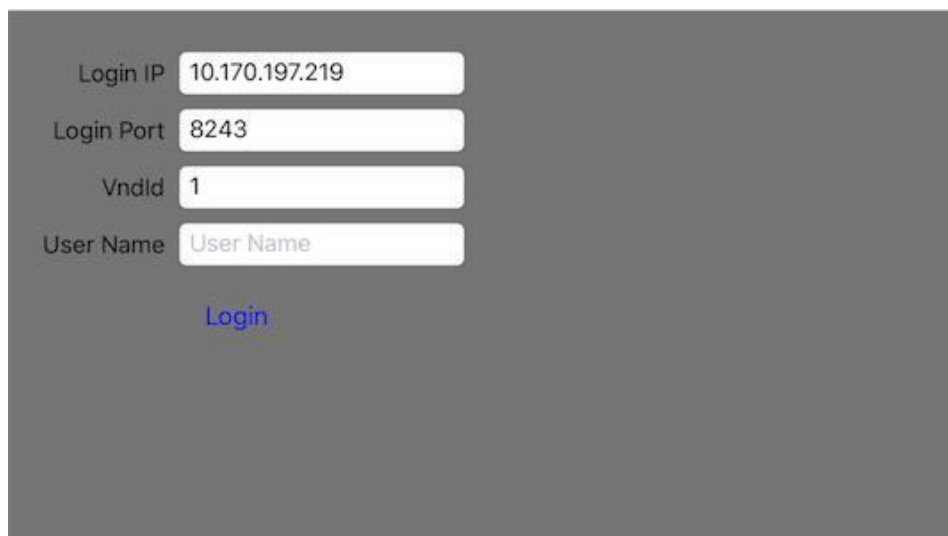
```
//objc code
- (void)addNotifions{
[[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(loginResult:)
name:AUTH MSG ON LOGIN object:nil];
[[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(logoutResult:)
name:AUTH MSG ON LOGOUT object:nil];
}
```

#### 设接入网关地址以及登录

```
//objc code
NSInteger setRes = [[CCSDK sharedInstance] setHostAddress:self.ipText.text
port:self.portText.text transSecurity:NO sipServerType:enType];
if (setRes != RET_OK){
[self showMessage:[NSString stringWithFormat:@"%d",NSLocalizedString(@"Host",
""), (long)setRes]];
}
else{
NSInteger ret = [[CCSDK sharedInstance] login:@"1" userName:self.userNameText.text];
```

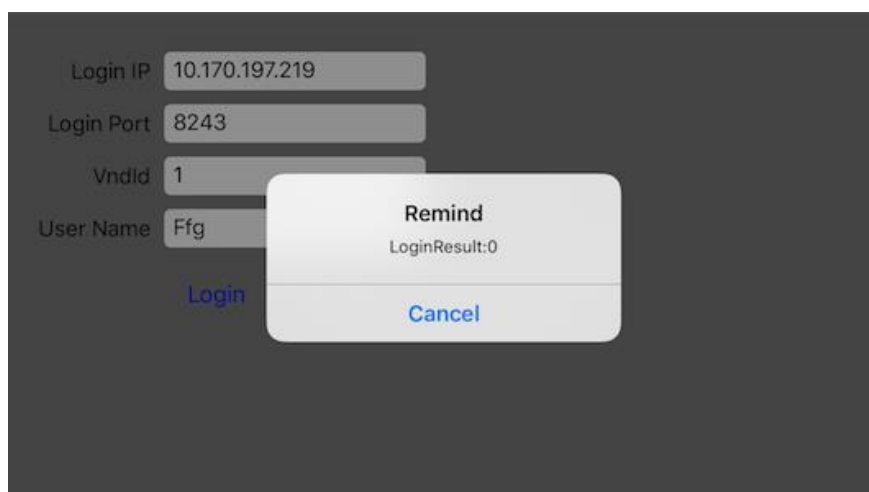
## 4.7 调试运行

步骤 1 点击“Product > Run”，手机显示如下图所示。



A login form interface with a dark gray background. It contains four input fields: 'Login IP' with the value '10.170.197.219', 'Login Port' with '8243', 'VndId' with '1', and 'User Name' with 'User Name'. Below the fields is a blue 'Login' button.

步骤 2 信息输入完成后，单击“Login”按钮，等待页面显示对话框“LoginResult:0”时，表示调用服务成功，如下图所示。



----结束

# 5 典型场景开发

- 5.1 场景一：登录与登出
- 5.2 场景二：TP 环境下的呼叫
- 5.3 场景三：MS 环境下的呼叫
- 5.4 场景四：设备控制

## 5.1 场景一：登录与登出

### 功能介绍

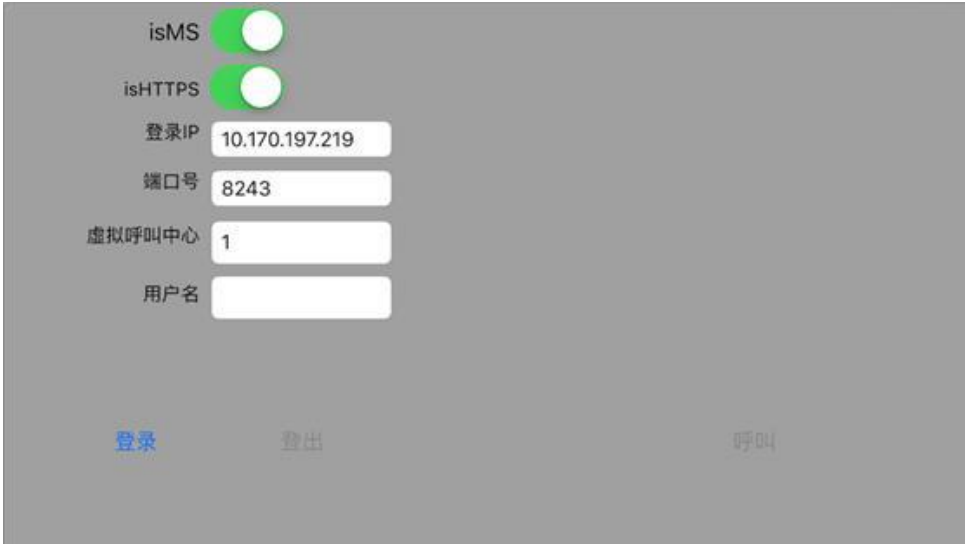
用于用户登录与登出。

对于登录不稳定的情况，我们需要在服务器操作系统中将 `net.ipv4.tcp.timeamps` 设置为 0

### 示例代码：

Demo 中 `LoginViewController.h` 和 `LoginViewController.m` 主要是对登录与登出的处理。

登录界面如下图所示。



登录成功页面：



关键代码如下所示，具体代码参考 eSDK\_ICP\_Demo\_V2.1.10\_iOS 中 LoginViewController.m 文件：

```
//引入 eSDK 头文件
#import <CCIOSSDK/CCIOSSDK.h>

//注册通知
- (void)addNotifions{
    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(loginResult:)
                                             name:AUTH MSG ON LOGIN object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(logoutResult:)
                                             name:AUTH MSG ON LOGOUT object:nil];
}

//登录结果
- (void)loginResult:(NSNotification *)notification{
    NSString *loginResult = (NSString *)notification.object;
```

```
        if ([loginResult isEqualToString:@"0"]) {
            dispatch_async(dispatch_get_main_queue(), ^{
                [self loginSuccess];
            });
        }else{
            dispatch_async(dispatch_get_main_queue(), ^{
                [self showMessage:[NSString stringWithFormat:@"登录失败:%@",loginResult]];
            });
        }
    }

//登出结果
- (void)logoutResult:(NSNotification *)notification{
    NSString *logoutResult = (NSString *)notification.object;
    if ([logoutResult isEqualToString:@"0"]) {
        dispatch_async(dispatch_get_main_queue(), ^{
            [self logoutSuccess];
        });
    }else{
        dispatch_async(dispatch_get_main_queue(), ^{
            [self showMessage:[NSString stringWithFormat:@"登出失败:%@",logoutResult]];
        });
    }
}

//登录操作
- (IBAction)loginClick:(id)sender {
    NSInteger setRes = [[CCSDK sharedInstance] setHostAddress:self.ipText.text
port:self.portText.text transSecurity:NO];
    if (setRes != RET_OK){
        [self showMessage:[NSString stringWithFormat:@"设置错误:%ld", (long) setRes]];
    }
    else{
        NSInteger ret = [[CCSDK sharedInstance] login:@"1"
userName:self.userNameText.text];
        if (ret == RET_OK) {
        }else{
            [self showMessage:[NSString stringWithFormat:@"登录接口调用错
误:%ld", (long)ret]];
        };
    }
}

//登出操作
- (IBAction)logoutClick:(id)sender {
    [self disableBtnStatus];
    [[CCSDK sharedInstance] logout];
}
}
```

涉及的主要接口如下表所示：

接口名	接口描述
- (NSInteger)setHostAddress:(NSString *)ip port:(NSString *)port transSecurity:(BOOL)transSec	设置登录的网关地址，http 与 https 选 择，环境的选择

接口名	接口描述
sipServerType:(int)serverType;	
- (NSInteger)login:(NSString *)vndid userName:(NSString *)userName;	登录
- (void)logout;	登出

## 5.2 场景二：TP 环境下的呼叫

### 功能介绍

主要介绍在 TP 环境下的视频呼叫。

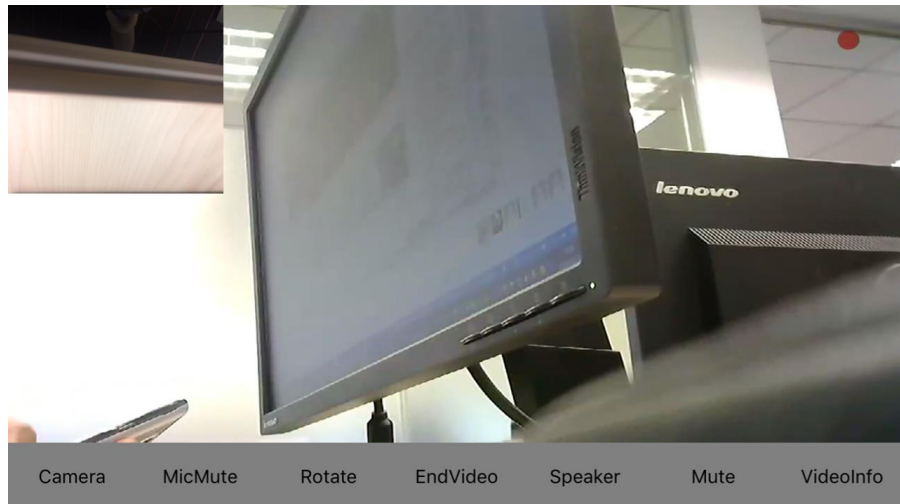
### 示例代码：

呼叫排队如下图所示。



呼叫成功如下图所示。





关键代码如下所示，具体代码参考 TPViewController.m 文件。

```
//引入头文件
#import <CCIOSSDK/CCIOSSDK.h>

//注册通知
- (void)addNotifications{
    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(showVerifyCode:)
                                             name:CALL GET VERIFY CODE
                                             object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(TPcallSuccess:)
                                             name:CALL MSG ON CONNECTED object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(TPcallEnd:)
                                             name:CALL MSG ON DISCONNECTED object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(TPcallFail:)
                                             name:CALL MSG ON FAIL object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(TPcallIsQueuing:)
                                             name:CALL MSG ON QUEUING object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(TPcallQueueTimeOut:)
                                             name:CALL MSG ON QUEUE TIMEOUT
                                             object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(TPreceiveQueueInfo:)
                                             name:CALL MSG ON QUEUE INFO object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(TPqueueIsCancel:)
                                             name:CALL MSG ON CANCEL QUEUE object:nil];
}

//处理获取验证码通知
- (void)showVerifyCode:(NSNotification *)notify
```

```

{
    if ([notify.object intValue] == 0) {
        NSString *encodedImageStr = [notify.userInfo objectForKey:@"verifyCode"];
        NSData *decodedImageData = [[NSData alloc]
initWithBase64EncodedString:encodedImageStr options:0];

        UIImage *decodedImage= [UIImage imageWithData:decodedImageData];
        dispatch_async(dispatch_get_main_queue(), ^{
            //抛回主线程进行显示
        });
    }
    else{
        dispatch_async(dispatch_get_main_queue(), ^{
            //抛回主线程进行提示，获取失败
        });
    }
}

//先获取验证码
- (IBAction)refreshBtnClick:(id)sender {
    [[CCSDK sharedInstance] getVerifyCode];
}

//呼叫
- (void)callClick{
    if ( callSuccess) {
        [[CCSDK sharedInstance] releaseCall];
    }else{
        NSInteger callRet = [[CCSDK sharedInstance] makeCall:self.aCode
callType:VIDEO_CALL callData:self.callData verifyCode: @"5679"];
        if (callRet != RET_OK) {
            [self showMessage:[NSString stringWithFormat:@"呼叫接口调用错
误:%ld", (long)callRet]];
        }else{
            dispatch_async(dispatch_get_main_queue(), ^{
                self.remindLabel.text = @"呼叫中";
            });
        }
    }
}

//取消排队
- (void)cancelQueueClick
{
    [[CCSDK sharedInstance] cancelQueue];
    dispatch_async(dispatch_get_main_queue(), ^{
        [self.cancelQueueBtn removeFromSuperview];
    });
}

//获取排队信息
- (void)getQueue{
    [[CCSDK sharedInstance] getCallQueueInfo];
}

```

涉及的主要接口如下表所示。

接口名	接口描述
- (BOOL)getVerifyCode	用于获取验证码
- (NSInteger)makeCall:(NSString *)accessCode callType:(NSString *)callType callData:(NSString *)callData verifyCode: (NSString *)verifyCode;	用于语音视频呼叫
- (void)releaseCall;	结束通话
- (void)getCallQueueInfo;	获取排队信息
- (void)cancelQueue;	取消排队
- (void)setVideoContainer:(id)localView remoteView:(id)remoteView;	设置本地和远端视频显示容器

### 5.3 场景三：MS 环境下的呼叫

#### 功能介绍

主要介绍在 MS 环境下的文字交谈、语音呼叫、会议，桌面共享等功能。NAT 穿越已经在 sdk 中做好，只要配置相应的参数即可。发送呼叫前的获取验证码可以在服务器上配置是否需要验证码，若不需要验证码，可以在 IcsGateway 服务器中打开 `home/prometheus/tomcat7/webapps/icsgateway/WEB-INF/config/verifycode.properties` 这个文件，修改 `VERIFYCODE_ISUSERFORCALL = false`，然后重启 IcsGateway 服务器即可。然后直接调用呼叫接口，验证码参数传 `nil` 即可。

如果出现语音连上就立即断开的情况，请到路由器的配置中，在“WAN 设置”中把 SIP ALG 开启。

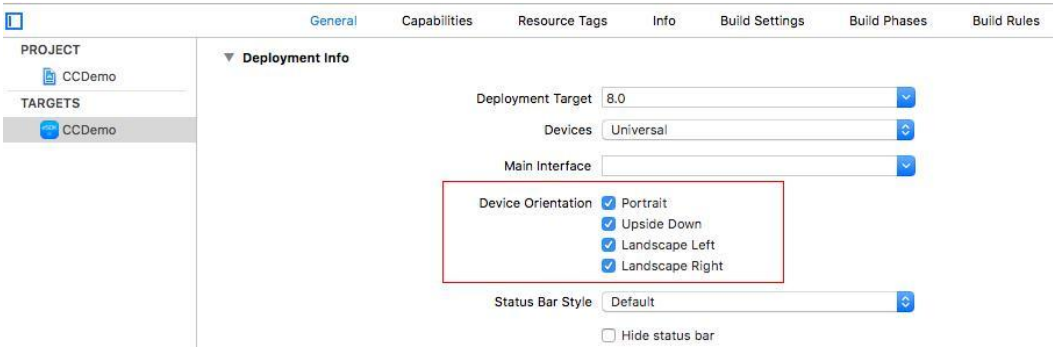
#### 示例代码：

通话界面如下图所示。

文字呼叫：



视频呼叫设置（本地视频界面的横竖屏设置同 app 的横竖屏设置保持一致，若 app 所有方向均支持，则视频界面在手机没有锁定横屏或竖屏的情况会根据重力感应调整）：



成功视频呼叫：



共享桌面如下图：



关键代码如下所示，具体代码参考 eSDK\_ICP\_Demo\_V2.1.10\_iOS 中的 MSViewController.m 文件。

```
//引入头文件
#import <CCIOSSDK/CCIOSSDK.h>

//注册通知- (void)addNotifications{

    [[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(callSuccess:)
        name:CALL MSG ON CONNECTED object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(callEnd:)
        name:CALL MSG ON DISCONNECTED object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(userLeave:)
        name:CALL MSG ON USER LEAVE object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(callFail:)
        name:CALL MSG ON FAIL object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(sendMsgSuccess)
        name:CHAT MSG ON SUCCESS object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(sendMsgFail:)
        name:CHAT MSG ON FAIL object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(receiveMessage:)
        name:CHAT MSG ON RECEIVE object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(callIsQueuing:)
```

```

        name:CALL_MSG_ON_QUEUEING object:nil];

[[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(queueIsCancel:)
        name:CALL_MSG_ON_CANCEL_QUEUE object:nil];

[[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(queueTimeOut:)
        name:CALL_MSG_ON_QUEUE_TIMEOUT object:nil];

[[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(receiveQueueInfo:)
        name:CALL_MSG_ON_QUEUE_INFO object:nil];

[[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(receiveDeskShare:)
        name:CALL_MSG_ON_SCREEN_DATA_RECEIVE object:nil];

[[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(deskShareStop:)
        name:CALL_MSG_ON_SCREEN_SHARE_STOP object:nil];

[[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(showVerifyCode:)
        name:CALL_GET_VERIFY_CODE object:nil];
}
//获取验证码

- (void)refreshBtnClick{
    [self.refreshBtn setBackgroundColor:[UIColor clearColor]];
    [[CCSDK shareInstance] getVerifyCode];
}

- (void)showVerifyCode:(NSNotification *)notify
{
    if ([notify.object intValue] == 0) {
        NSString *encodedImageStr = [notify.userInfo objectForKey:@"verifyCode"];
        NSData *decodedImageData = [[NSData alloc]
initWithBase64EncodedString:encodedImageStr options:0];

        UIImage *decodedImage= [UIImage imageWithData:decodedImageData];

        NSLog(@"==Decoded image size: %@", NSStringFromCGSize(decodedImage.size));
        dispatch async(dispatch get main queue(), ^{
            [self.verifyCodeImg setImage:decodedImage];
            [self.refreshBtn setTitle:@"" forState:UIControlStateNormal];
        });
    }
    else{
        dispatch async(dispatch get main queue(), ^{
            [CCDemoUtil showAlertWithTitle:NSLocalizedString(@"Remind", "")
            content:NSLocalizedString(@"GetCodeError", "")];
        });
    }
}

```

```

//发起文字交谈
- (void)doWebChatCall
{
    [[CCSDK sharedInstance] webChatCall:[LoginInfo sharedInstance].MSChatACode
callData:[LoginInfo sharedInstance].MScallData verifyCode:[LoginInfo
sharedInstance].VCode];
    _isWebCall = YES;
    /*_callTimer = [NSTimer scheduledTimerWithTimeInterval:1
                                                    target:self
                                                    selector:@selector(countTimer)
                                                    userInfo:nil repeats:YES];*/

    [self startCallTimer];
    self.remindLabel.text = [NSString
stringWithFormat:@"%@@ %@",NSLocalizedString(@"Chat",
""),NSLocalizedString(@"Calling", "")];
    _isCalling = YES;
}
//发送文字消息
- (void)sendMessageClick
{
    if (_isCalling)
    {
        [CCDemoUtil showAlertWithTitle:NSLocalizedString(@"Remind", "")
content:NSLocalizedString(@"Current", "")];
        return;
    }
    if (self.chatText.text.length == 0) {
        [CCDemoUtil showAlertWithTitle:NSLocalizedString(@"Remind", "")
content:NSLocalizedString(@"NoMessage", "")];
        return;
    }

    NSInteger ret = [[CCSDK sharedInstance] sendMsg:self.chatText.text];
    if (ret != RET_OK)
    {
        [CCDemoUtil showAlertWithTitle:NSLocalizedString(@"Remind", "")
content:[NSString
stringWithFormat:@"%@@:ld",NSLocalizedString(@"SendInterface", ""),(long)ret]];
        return;
    }
    NSString *message = [NSString
stringWithFormat:@"%@@:ld",NSLocalizedString(@"Send", ""),self.chatText.text];
    [self.dataArray addObject:message];
    self.chatText.text = @"";
    [self.chatTableView reloadData];
    [self.chatTableView scrollToRowAtIndexPath:[NSIndexPath
indexPathForRow:self.dataArray.count - 1 inSection:0]
atScrollPosition:UITableViewScrollPositionBottom animated:YES];
}

//视频呼叫
- (void)videoCallClick
{

```

```

        if (_isCalling)
        {
            [CCDemoUtil showAlertWithTitle:NSLocalizedString(@"Remind", "")
            content:NSLocalizedString(@"Current", "")];
            return;
        }

        if (_videoSuccess)
        {
            [[CCSDK sharedInstance] releaseCall];
            _videoSuccess = NO;
            if (_screenShare) {
                _screenShare = NO;
            }
            if (_videoViewOpen)
            {
                _videoViewOpen = NO;
                dispatch_async(dispatch_get_main_queue(), ^{
                    [self closeView];
                });
            }
            if (_shareViewOpen)
            {
                shareViewOpen = NO;
                dispatch_async(dispatch_get_main_queue(), ^{
                    [self.screenshareView removeFromSuperview];
                });
            }
            dispatch_async(dispatch_get_main_queue(), ^{
                [self.remindLabel removeFromSuperview];
                self.remindLabel.text = [NSString
                stringWithFormat:@"%@@ %@",NSLocalizedString(@"Video",
                ""),NSLocalizedString(@"CallEnd", "")];
                [self.view addSubview:self.remindLabel];
                [self.videoBtn setTitle:NSLocalizedString(@"Video", "")
                forState:UIControlStateNormal];
            });
        }
        else
        {
            {
                isCalling = YES;
                if( audioSuccess)
                {
                    [self startCallTimer];
                    self.remindLabel.text = [NSString
                    stringWithFormat:@"%@@ %@",NSLocalizedString(@"Video",
                    ""),NSLocalizedString(@"Calling", "")];
                    [[CCSDK sharedInstance] updateToVideo];
                }
                else
                {
                    self.remindLabel.text = [NSString
                    stringWithFormat:@"%@@ %@",NSLocalizedString(@"Video",
                    ""),NSLocalizedString(@"Calling", "")];

```



```

        [[CCSDK sharedInstance] makeCall:[LoginInfo sharedInstance].MSAudioACode
callType:VIDEO_CALL callData:[LoginInfo sharedInstance].MSCallData
verifyCode:self.verifyText.text];
    }
}

//语音呼叫
-(void)audioCall
{
    _isCalling = YES;

    self.remindLabel.text = [NSString
stringWithFormat:@"%@" %@"",NSLocalizedString(@"Audio",
""),NSLocalizedString(@"Calling", "")];
    if (!_audioSuccess)
    {
        [[CCSDK sharedInstance] makeCall:[LoginInfo sharedInstance].MSAudioACode
callType:AUDIO_CALL callData:[LoginInfo sharedInstance].MSCallData
verifyCode:self.verifyText.text];
    }
    else
    {
        [[CCSDK sharedInstance] releaseCall];
        if ( videoSuccess)
        {
            videoSuccess = NO;

            if ( screenShare)
            {
                screenShare = NO;
            }
            if ( videoViewOpen)
            {
                videoViewOpen = NO;
                dispatch async(dispatch get main queue(), ^{
                    [self closeView];
                });
            }

            if ( shareViewOpen)
            {
                shareViewOpen = NO;
                dispatch async(dispatch get main queue(), ^{
                    [self.screenshareView removeFromSuperview];
                });
            }
            dispatch async(dispatch get main queue(), ^{
                [self.remindLabel removeFromSuperview];
                self.remindLabel.text = NSLocalizedString(@"CallEnd", "");
                [self.view addSubview:self.remindLabel];
                [self.videoBtn setTitle:NSLocalizedString(@"Video", "")
forState:UIControlStateNormal];
            });
        }
    }
}

```

```
    }  
}  
//设置桌面共享容器  
[[CCSDK shareInstance] setDesktopShareContainer:self.screenshareView];
```

涉及的主要接口如下表所示：

接口名	接口描述
- (NSInteger)setSIPServerAddress:(NSString *)ip port:(NSString *)port ;	设置 SIP 服务器地址（呼叫地址）
- (NSInteger)webChatCall:(NSString *)accessCode callData:(NSString *)callData verifyCode:(NSString *)verifyCode;	发起文字交谈
- (NSInteger)sendMsg:(NSString *)message;	发送文字消息
- (NSInteger)makeCall:(NSString *)accessCode callType:(NSString *)callType callData:(NSString *)callData verifyCode:(NSString *)verifyCode;	语音/视频呼叫
- (NSInteger)updateToVideo;	语音升级为视频
- (void)releaseCall;	结束呼叫
- (void)getCallQueueInfo;	获取排队信息
- (void)cancelQueue;	取消排队
- (void)setVideoContainer:(id)localView remoteView:(id)remoteView;	设置本地和远端视频显示容器
- (void)setDesktopShareContainer:(UIImage View *)shareView;	设置桌面共享容器

## 5.4 场景四：设备控制

### 功能介绍

通话过程中前置/后置摄像头切换，扬声器/听筒模式切换，麦克风静音。

## 示例代码：

以 MS 环境为例(TP 接口同样适用)，扬声器/听筒切换、摄像头旋转、前/后摄像头切换、画质模式切换界面如下图所示。



关键代码如下所示，具体代码参考 eSDK\_ICP\_Demo\_V2.1.10\_iOS 中的 MSViewController.m 文件：

```
//扬声器/听筒切换
- (void)speakMuteClick
{
    if (!_audioSuccess){
        [CCDemoUtil showAlertWithTitle:NSLocalizedString(@"Remind", "")
        content:NSLocalizedString(@"NoAudio", "")];
        return;
    }

    [[self class] cancelPreviousPerformRequestsWithTarget:self
    selector:@selector(speakMute) object:nil];
    [self performSelector:@selector(speakMute) withObject:nil afterDelay:1.0];
}
- (void)speakMute
{
    if (_speakIsMute)
    {
        [[CCSDK sharedInstance] changeAudioRoute:1];
        [self.speakMuteBtn setTitle:NSLocalizedString(@"Receiver", "")
        forState:UIControlStateNormal];
        _speakIsMute = NO;
        return;
    }
    [[CCSDK sharedInstance] changeAudioRoute:0];
    [self.speakMuteBtn setTitle:NSLocalizedString(@"Speaker", "")
    forState:UIControlStateNormal];
    speakIsMute = YES;
}
//摄像头旋转
```

```
- (void)rotateClick
{
    if (!_videoSuccess)
    {
        [CCDemoUtil showAlertWithTitle:NSLocalizedString(@"Remind", "")
content:NSLocalizedString(@"NoVideo", "")];
        return;
    }
    _rotate += 90;
    if (_rotate > 270)
    {
        _rotate = 0;
    }
    [[CCSDK sharedInstance] setVideoRotate:_rotate];
}

//切换摄像头
- (void)cameraSwitchClick
{
    if (!_videoSuccess)
    {
        [CCDemoUtil showAlertWithTitle:NSLocalizedString(@"Remind", "")
content:NSLocalizedString(@"NoVideo", "")];
        return;
    }
    [[self class] cancelPreviousPerformRequestsWithTarget:self
selector:@selector(cameraSwitch) object:nil];
    [self performSelector:@selector(cameraSwitch) withObject:nil afterDelay:1.0];
}
- (void)cameraSwitch
{
    rotate = 0;
    if ( isBackCamera)
    {
        [[CCSDK sharedInstance] switchCamera:1];
        isBackCamera = NO;
        [self.cameraSwitchBtn setTitle:NSLocalizedString(@"FrontCamera", "FrontCam")
forState:UIControlStateNormal];
        return;
    }
    [[CCSDK sharedInstance] switchCamera:0];
    isBackCamera = YES;
    [self.cameraSwitchBtn setTitle:NSLocalizedString(@"BackCamera", "BackCam")
forState:UIControlStateNormal];
}

//画质模式切换
- (void)videoModeClick
{
    if (! videoSuccess)
    {
        [CCDemoUtil showAlertWithTitle:NSLocalizedString(@"Remind", "")
content:NSLocalizedString(@"NoVideo", "")];
        return;
    }
    [[self class] cancelPreviousPerformRequestsWithTarget:self
```

```
selector:@selector(videoMode) object:nil];
    [self performSelector:@selector(videoMode) withObject:nil afterDelay:1.0];
}
- (void)videoMode
{
    if (_isHResolution)
    {
        [[CCSDK sharedInstance] setVideoMode:1];
        _isHResolution = NO;
        return;
    }
    [[CCSDK sharedInstance] setVideoMode:0];
    _isHResolution = YES;
}
```

涉及的主要接口如下表所示。

接口名	接口描述
- (BOOL)changeAudioRoute:(int)route	扬声器/听筒模式切换
- (BOOL)setVideoRotate:(VIDEO_ROTATE)rotate	设置视频旋转角度
- (BOOL)switchCamera:(int)index	前置/后置摄像头切换
- (BOOL)setVideoMode:(int)videoMode	设置视频显示模式

# 6 问题定位

## 错误码获取方法

### 接口返回

有些接口调用后，无论调用成功还是失败，会有一个返回值或者事件通知。如果返回值为 0，表示成功；否则表示失败，该返回值即为错误码。

### 日志获取

- 日志路径  
根据设置日志路径接口中传入的日志路径进行查找。
- 获取方法

接口日志文件里记录了调用的每个接口调用的时间、接口入参以及调用结果。

以设置网关接口为例：

setHostAddress：搜索接口关键字，找到该行记录，可以看到接口的具体调用信息。

```
CCiOSSDK [Debug] 17/2/23 14:31:01:CCSDK.m line:189 | -[CCSDK  
setHostAddress:port:transSecurity:sipServerType:] |  
ip=10.170.197.219,port=8243,transSecurity=1,sipServerType=2
```

## 错误信息查询方法

在接口参考文档里，列出了所有错误码信息。

根据接口返回的错误码，查询相对应的错误描述。

错误码	
错误码	说明
0	成功。
-1	参数错误。
-2	上下文尚未建立。
-3	连接未连接。
-4	无人应答。
-5	网络发生错误。
100-100-002	服务端platform.properties中的SERVER_IP地址配置错误。
100-100-003	没有连接到CCS，Wan服务不可用。

## 日志分析

以设置网关接口的调用方法为例，全局搜索关键字“setHostAddress”，搜索到对应的记录。

```
CCiOSSDK [Debug] 17/2/23 14:31:01:CCSDK.m line:189 | -[CCSDK  
setHostAddress:port:transSecurity:sipServerType:] |  
ip=10.170.197.219,port=8243,transSecurity=1,sipServerType=2
```

# 7 修订记录

发布日期	文档版本	描述
2017-3-15	V200R001C10	文档 V200R001C10 版本发布。 全量适配 TP&MS 功能。
2017-3-8	1.1.T1	文档 1.1.T1 版本发布。 增加： <ul style="list-style-type: none"><li>5.3 场景三：MS 环境下的呼叫</li><li>添加 MS NAT 功能</li></ul> 修改：5.4 场景四：设备控制
2016-12-31	V2.1.00	版本第一次正式发布。



# 8 附录

## 8.1 Hello World 源码文件

## 8.1 Hello World 源码文件

### 8.1.1 ViewController.m

```
//objc code
#import "ViewController.h"
#import <CCIOSSDK/CCIOSSDK.h>

@interface ViewController ()

@property (strong, nonatomic) UITextField *ipText;
@property (strong, nonatomic) UITextField *portText;
@property (strong, nonatomic) UITextField *userNameText;
@property (strong, nonatomic) UITextField *vndIdText;

@property (strong, nonatomic) UIButton *loginBtn;

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.
    [self addNotifications];
    [self initView];
}

- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}
```

```
- (void)addNotifications
{
    [[NSNotificationCenter defaultCenter] addObserver:self
    selector:@selector(onKeyBoardWillShow:)
    name:UIKeyboardWillShowNotification
    object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
    selector:@selector(onKeyBoardWillHide:)
    name:UIKeyboardWillHideNotification
    object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
    selector:@selector(loginResult:)
    name:AUTH_MSG_ON_LOGIN
    object:nil];
}

-(void)onKeyBoardWillShow:(NSNotification *)notify
{
    NSDictionary *userInfo = [notify userInfo];
    NSNumber *duration = userInfo[UIKeyboardAnimationDurationUserInfoKey];
    [UIView animateWithDuration:[duration doubleValue] animations:^(
    self.view.frame = CGRectMake(0, -50, [UIScreen mainScreen].bounds.size.width,
    [UIScreen mainScreen].bounds.size.height);
    )];
}

-(void)onKeyBoardWillHide:(NSNotification *)notify
{
    NSDictionary *userInfo = [notify userInfo];
    NSNumber *duration = userInfo[UIKeyboardAnimationDurationUserInfoKey];
    [UIView animateWithDuration:[duration doubleValue] animations:^(
    self.view.frame = CGRectMake(0, 0, [UIScreen mainScreen].bounds.size.width,
    [UIScreen mainScreen].bounds.size.height);
    )];
}

- (void)touchesBegan:(NSSet<UITouch *> *)touches withEvent:(UIEvent *)event{
[self.view endEditing:YES];
}

-(void)initView{
self.view.backgroundColor = [UIColor grayColor];

UILabel *loginIp = [[UILabel alloc] initWithFrame:CGRectMake(10, 30, 100, 30)];
loginIp.text=@"Login IP";
loginIp.textAlignment = NSTextAlignmentRight;
[self.view addSubview:loginIp];

self.ipText = [[UITextField alloc] initWithFrame:CGRectMake(120,30, 200, 30)];
self.ipText.placeholder = @"Login IP";
self.ipText.text = @"10.170.197.219";
self.ipText.borderStyle = UITextBorderStyleRoundedRect;
```

```
[self.view addSubview:self.ipText];

UILabel *loginPort = [[UILabel alloc] initWithFrame:CGRectMake(10, 70, 100, 30)];
loginPort.text=@"Login Port";
loginPort.textAlignment = NSTextAlignmentRight;
[self.view addSubview:loginPort];

self.portText = [[UITextField alloc] initWithFrame:CGRectMake(120,70, 200, 30)];
self.portText.placeholder = @"Login Port";
self.portText.text = @"8243";
self.portText.borderStyle = UITextBorderStyleRoundedRect;
[self.view addSubview:self.portText];

UILabel *loginVndId = [[UILabel alloc] initWithFrame:CGRectMake(10, 110, 100, 30)];
loginVndId.text=@"VndId";
loginVndId.textAlignment = NSTextAlignmentRight;
[self.view addSubview:loginVndId];

self.vndIdText = [[UITextField alloc] initWithFrame:CGRectMake(120,110, 200, 30)];
self.vndIdText.placeholder = @"VndId";
self.vndIdText.text = @"1";
self.vndIdText.borderStyle = UITextBorderStyleRoundedRect;
[self.view addSubview:self.vndIdText];

UILabel *userName = [[UILabel alloc] initWithFrame:CGRectMake(10, 150, 100, 30)];
userName.text=@"User Name";
userName.textAlignment = NSTextAlignmentRight;
[self.view addSubview:userName];

self.userNameText = [[UITextField alloc] initWithFrame:CGRectMake(120,150, 200,
30)];
self.userNameText.placeholder = @"User Name";
self.userNameText.borderStyle = UITextBorderStyleRoundedRect;
[self.view addSubview:self.userNameText];

self.loginBtn = [[UIButton alloc] initWithFrame:CGRectMake(120,200, 80, 30)];
[self.loginBtn setTitle:@"Login" forState:UIControlStateNormal];
[self.loginBtn setTitleColor:[UIColor blueColor] forState:UIControlStateNormal];
[self.loginBtn addTarget:self action:@selector(loginClick)
forControlEvents:UIControlEventTouchUpInside];
[self.view addSubview:self.loginBtn];

}

- (void)loginClick {
if ([self.userNameText.text length] == 0)
{
[self showAlertWithTitle:NSLocalizedString(@"Remind", "") content:@"userName is
nil"];
return;
}

int serverType = SERVER_TYPE_MS;
BOOL _isHTTPS = YES;
```

```

if (_isHTTPS)
{
//若使用 https, 导入并设置证书设置如下
//      NSString * cerPath = [[NSBundle mainBundle] pathForResource:@"server"
ofType:@"der"];
//      NSData *dataCA = [NSData dataWithContentsOfFile:cerPath];
//      [[CCSDK sharedInstance] setNeedValidate:YES needValidateDomain:NO
certificateData:dataCA];

// 启用 https, 不验证证书与域名设置如下
[[CCSDK sharedInstance] setNeedValidate:NO needValidateDomain:NO
certificateData:nil];
}

[[CCSDK sharedInstance] setHostAddress:self.ipText.text
port:self.portText.text
transSecurity:YES//若不使用 https, 设置为 NO
sipServerType:serverType];

NSInteger result = [[CCSDK sharedInstance] login:self.vndIdText.text
userName:self.userNameText.text];
if (result != RET_OK)
{
[self showAlertWithTitle:NSLocalizedString(@"Remind", "") content:@"param is
invalid"];
}
}

- (void)loginResult:(NSNotification *)notify
{
NSString *loginResult = (NSString *)notify.object;
dispatch_async(dispatch_get_main_queue(), ^{
[self showAlertWithTitle:NSLocalizedString(@"Remind", "")
content:[NSString stringWithFormat:@"%s:%s",NSLocalizedString(@"LoginResult",
""),loginResult]];
});
}

- (void)showAlertWithTitle:(NSString*)title content:(NSString*)content
{
UIAlertView *alertView = [[UIAlertView alloc]initWithTitle:title message:content
delegate:nil cancelButtonTitle:@"Cancel" otherButtonTitles:nil];
[alertView show];
}
@end

```