

# iClap 批注 SDK 集成文档

## 一. 概述

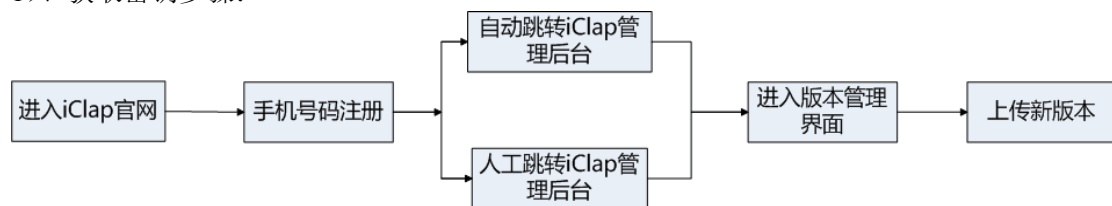
iClap SDK 是一个产品管理系统，专注于为移动互联网企业提供企业规范化解解决方案，改变传统的产品管理模式，实现产品管理场景化。可以直接在 APP 内进行批注和任务的协作，实时将你要修改的内容或者 Bug 等进行推送，团队可在 APP 上直接查看，还原当时的场景，让工作简单智能人性化！

在 iOS 集成过程中遇到问题，欢迎加入官方 QQ 群：88290667，沟通交流。

## 二. 获取密钥

集成 iclap SDK 之前，您首先需要到 iclap 官网注册，以获得唯一 appkey 和 secret，不同项目的 APP 禁止使用相同的 appkey 和 secret。

1)、获取密钥步骤：



2)、截图示意：



应用唯一性规则：

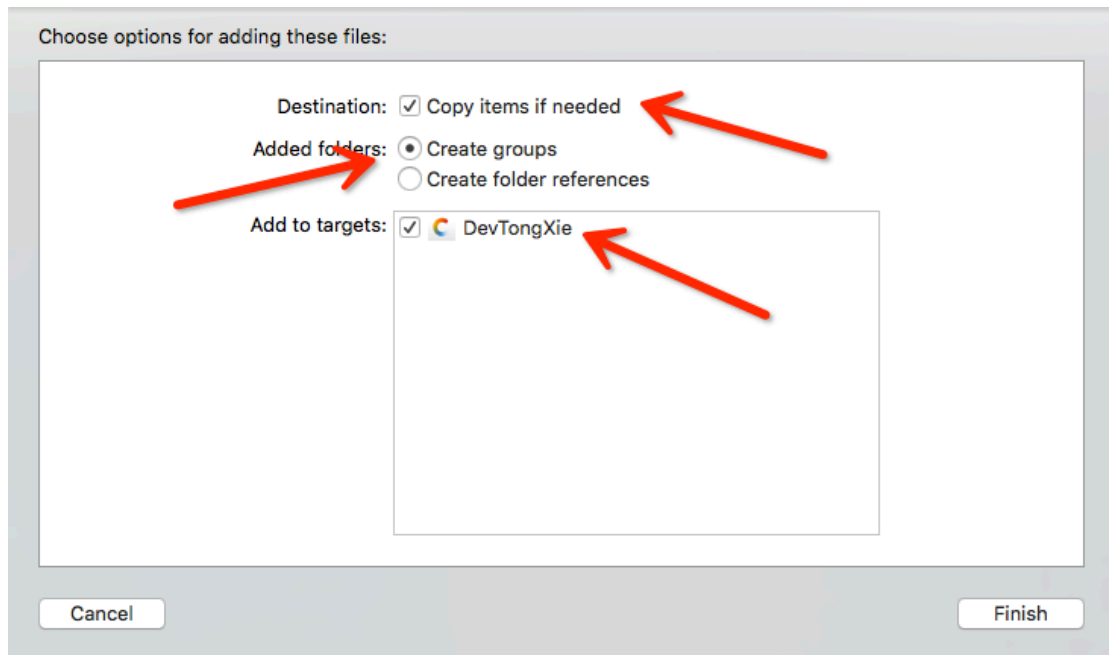
目前通过包名来区分判断应用唯一性，即会产生两种情况：

- 1、同一个应用，如使用相同包名，默认为同一个应用，上传之后，他人会无法再使用这个包名上传应用，也不能创建项目。
- 2、同一个应用，如使用不同包名，默认为不是同一个应用，上传之后，也会默认为不是同一个应用，故信息也就无法互通；

在上传项目的时候请参考实际情况正确使用。

### 三. 开发指南

首先：下载 SDK 库文件解压，Release-iphoneos 仅用于真机使用；Release-iphonesimulator 仅用于模拟器使用；Products 支持真机和模拟器。将解压后的 iClapSDK.framework 导入您的工程中，拖到工程中后弹出以下对话框，勾选“Copy items into destination group's folder(if needed)”，并点击“Finish”按钮，如图



更新配置：

a、iOS9 引入了新特性 App Transport Security (ATS)，要求 App 内访问的网络必须使用 HTTPS 协议，配置项目支持 HTTP 协议如下：

1、在 Info.plist 中添加 NSAppTransportSecurity 类型 Dictionary。

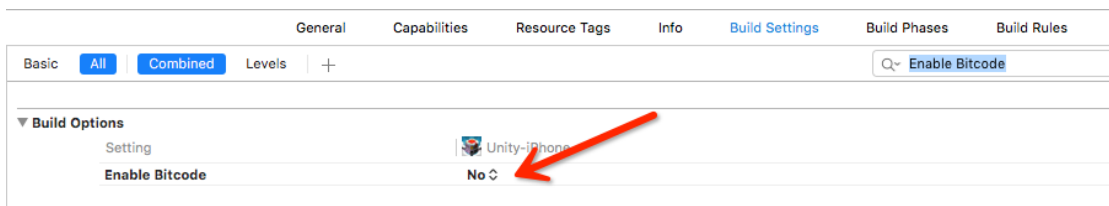
2、在 NSAppTransportSecurity 下添加 NSAllowsArbitraryLoads 类型 Boolean, 值设为 YES。

▼ NSAppTransportSecurity	Dictionary	(1 item)
NSAllowsArbitraryLoads	Boolean	YES


















b、Xcode7 默认会开启 Bitcode，关闭方法如下：

1、打开 Build Settings，切换到 All 选项。

2、搜索 Enable Bitcode，把 Yes 改为 NO。



项目中需要导入 libc++.dylib 及以下框架

▼ Linked Frameworks and Libraries		
Name		Status
 libstdc++.dylib		Required ⇅
 CoreTelephony.framework		Required ⇅
 CoreMedia.framework		Required ⇅
 libc++.dylib		Required ⇅
 SystemConfiguration.framework		Required ⇅
 CoreLocation.framework		Required ⇅
 AVFoundation.framework		Required ⇅
 AudioToolbox.framework		Required ⇅
 libsqlite3.dylib		Required ⇅
 CoreText.framework		Required ⇅
 AddressBook.framework		Required ⇅
 AddressBookUI.framework		Required ⇅
 AssetsLibrary.framework		Required ⇅
 MobileCoreServices.framework		Required ⇅
 MessageUI.framework		Required ⇅
 ImageIO.framework		Required ⇅
 iClapSDK.framework		Required ⇅

然后,需要在项目 AppDelegate 方法中导入  
`#import <iClapSDK/DevTongXieConfig.h>`

添加初始化方法, 如下

```

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    // Override point for customization after application launch.
    // 初始化Dev批注
    [[DevTongXieConfig sharedInstance] initWithAppkey:Appkey secret:SecretKey];
    [[DevTongXieConfig sharedInstance] devRegisterForRemoteNotifications];

    [self.window makeKeyAndVisible];

    [[DevTongXieConfig sharedInstance] devDidFinishLaunchingWithOptions:launchOptions];

    return YES;
}

```

// 初始化 Dev 批注(Appkey 和 SecretKey 是获取的密钥)

```

//MARK: 批注平台类型
typedef NS_ENUM (NSInteger, DevPlatformType)
{
    Platform_Type_Default      = 1 << 1, // 普通应用App
    Platform_Type_Application = Platform_Type_Default, // 普通应用App
    Platform_Type_Unity3d      = 1 << 2, // 基于Unity3d开发的App
    Platform_Type_Cocos2d      = 1 << 3, // 基于Cocos2d开发的App
    Platform_Type_Cocos2dx     = 1 << 4, // 基于Cocos2dx开发的App
};

```

如果需要使用推送功能, 需要添加:

```

//注册远程通知
#ifdef __IPHONE_8_0
- (void)application:(UIApplication *)application didRegisterUserNotificationSettings:(UIUserNotificationSettings *)
notificationSettings
{
    [application registerForRemoteNotifications];
}
#endif

- (void)application:(UIApplication *)application didRegisterForRemoteNotificationsWithDeviceToken:(NSData *)deviceToken
{
    [[DevTongXieConfig sharedInstance] devDidRegisterForRemoteNotificationsWithDeviceToken:deviceToken];
}

- (void)application:(UIApplication *)application didReceiveRemoteNotification:(NSDictionary *)userInfo
{
    [[DevTongXieConfig sharedInstance] devDidReceiveRemoteNotification:userInfo];
}

#ifdef __IPHONE_8_0
- (void)application:(UIApplication *)application handleActionWithIdentifier:(NSString *)identifier forRemoteNotification:
(NSDictionary *)userInfo completionHandler:(void (^)(void))completionHandler
{
}
#endif

```

## 四. 类参考

主要是批注功能扩展（批注样式、插件功能的开启与关闭）

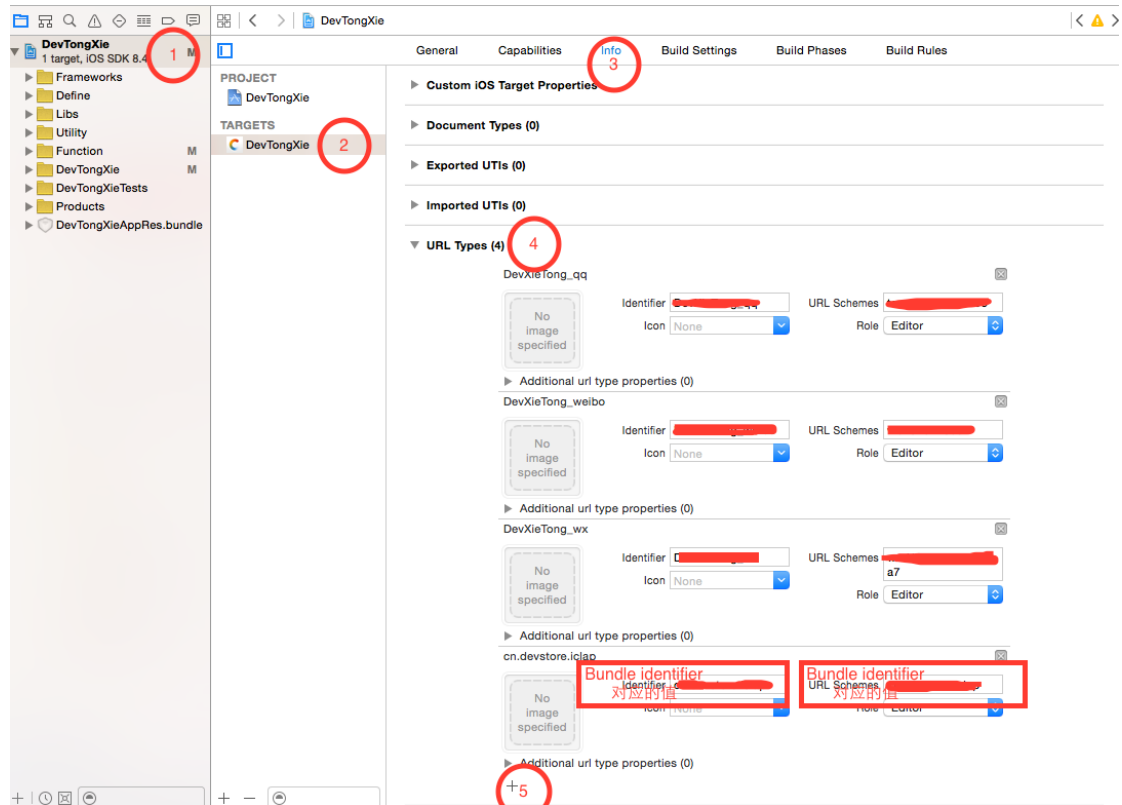
```

/**
 * 根据申请的身份令牌信息
 *
 * @param appKeyStr 身份令牌
 * @param secretStr 加密字符
 */
- (void)initIClapSDKWithAppkey:(NSString *)appkeyStr
secret:(NSString *)secretStr;

/**
 * 校验申请的身份令牌信息
 *
 * @param appkey 身份令牌
 * @param secret 加密字符
 * @param platformType 批注类型
 */
- (void)initIClapSDKWithAppkey:( NSString *)appkeyStr
secret:( NSString *)secretStr
hookType:(DevPlatformType)platformType;

```

Key	Type	Value
▼ Information Property List	Dictionary	(20 items)
Bundle display name	String	iClap
Executable file	String	\$(EXECUTABLE_NAME)
<b>Bundle identifier</b>	String	<b>[Redacted]</b>
InfoDictionary version	String	6.0
▶ Localizations	Array	(2 items)
Bundle name	String	看看



## 五. 可选功能

说明：按照如下步骤配置后可在 iClap 直接启动 APP

1. 找到 APP 的 plist 文件，复制 “Bundle identifier” 对应的 value 值
2. 增加一个 URL Types，如下图：
3. 第 5 步会添加一个 URL Types，在 “identifier” 和 “URL Schemes” 对应值输入 “Bundle identifier” 对应的值，其中 “URL Schemes” 对应的值是必填项。

## 六. 常见问题

- 1、64 位支持 (启用对 64 位支持)
- 2、引用路径 (请按照开发文档配置 SDK 路径)
- 3、C++编译器 (Apple LLVM...Language - Modules 使用 C++11 support)
- 4、批注功能未能开启 (请检查获取的密钥 TokenId 和 SecretKey 值是否正确)
- 5、C++错误, 如下图 (检查导入 libc++.dylib 库)

```
"std::__1::basic_string<char, std::__1::char_traits<char>, std::__1::allocator<char>>::replace(unsigned long, unsigned long, char const*, unsigned long)",  
referenced from:  
  StrPacket::Replace(std::__1::basic_string<char, std::__1::char_traits<char>, std::__1::allocator<char>> &, std::__1::basic_string<char,  
std::__1::char_traits<char>, std::__1::allocator<char>> const&, std::__1::basic_string<char, std::__1::char_traits<char>, std::__1::allocator<char>> const&  
iClapSDK(StrPacket.o)  
"std::__1::basic_string<char, std::__1::char_traits<char>, std::__1::allocator<char>>::compare(char const*) const", referenced from:
```

- 6、选择真机运行时使用 Release-iphonios 下的 iClapSDK.framework, 模拟器选择 Release-iphonesimulator 下的 iClapSDK.framework

- 7、静态库中包含了 Category (分类)

如果静态库中包含了 Category, 有时候在使用静态库的工程中会报“方法找不到”的错误 (unrecognized selector sent to instance)

解决方案: 在使用静态库的工程中配置 Other Linker Flags 为-ObjC

- 8、如果项目中有 CCX 在解决问题 7 时, 使用静态库的工程中配置 Other Linker Flags 为-ObjC 后报错, 导入 MediaPlayer.framework 和 GameController.framework(需要 iOS7 以上版本, Status 设为 Optional)

```
Undefined symbols for architecture armv7:  
  "_MPMoviePlayerPlaybackStateDidChangeNotification", referenced from:  
    -[UIViewWrapperIOS dealloc] in libcocos2d iOS.a(UIVideoPlayer-ios.o)  
    -[UIViewWrapperIOS setURL:] in libcocos2d iOS.a(UIVideoPlayer-ios.o)  
  "_MPMoviePlayerPlaybackDidFinishNotification", referenced from:  
    -[UIViewWrapperIOS dealloc] in libcocos2d iOS.a(UIVideoPlayer-ios.o)  
    -[UIViewWrapperIOS setURL:] in libcocos2d iOS.a(UIVideoPlayer-ios.o)  
  "_OBJC_CLASS_$_GCController", referenced from:  
    objc-class-ref in libcocos2d iOS.a(CCController-iOS.o)  
    (maybe you meant: _OBJC_CLASS_$_GCControllerConnectionEventHandler)  
  "_OBJC_CLASS_$_MPMoviePlayerController", referenced from:  
    objc-class-ref in libcocos2d iOS.a(UIVideoPlayer-ios.o)  
  "_GCControllerDidDisconnectNotification", referenced from:  
    -[GCControllerConnectionEventHandler observerConnection:disconnection:] in libcocos2d iOS.a(CCController-iOS.o)  
  "_GCControllerDidConnectNotification", referenced from:  
    -[GCControllerConnectionEventHandler observerConnection:disconnection:] in libcocos2d iOS.a(CCController-iOS.o)  
ld: symbol(s) not found for architecture armv7  
clang: error: linker command failed with exit code 1 (use -v to see invocation)
```

```
1. "_MPMoviePlayerPlaybackStateDidChangeNotification", referenced from:  
-[UIViewWrapperIOS dealloc] in libcocos2d iOS.a(UIVideoPlayer-ios.o)  
-[UIViewWrapperIOS setURL:] in libcocos2d iOS.a(UIVideoPlayer-ios.o)  
2. "_MPMoviePlayerPlaybackDidFinishNotification", referenced from:  
-[UIViewWrapperIOS dealloc] in libcocos2d iOS.a(UIVideoPlayer-ios.o)  
-[UIViewWrapperIOS setURL:] in libcocos2d iOS.a(UIVideoPlayer-ios.o)  
3. "_OBJC_CLASS_$_GCController", referenced from:  
Objc-class-ref in libcocos2d iOS.a(CCController-iOS.o)  
(maybe you meant: _OBJC_CLASS_$_GCControllerConnectionEventHandler)  
4. "_OBJC_CLASS_$_MPMoviePlayerController", referenced from:  
Objc-class-ref in libcocos2d iOS.a(UIVideoPlayer-ios.o)  
5. "_GCControllerDidDisconnectNotification", referenced from:  
-[GCControllerConnectionEventHandler observerConnection:disconnection:] in libcocos2d iOS.a(CCController-iOS.o)  
6. "_GCControllerDidConnectNotification", referenced from:  
-[GCControllerConnectionEventHandler observerConnection:disconnection:] in libcocos2d iOS.a(CCController-iOS.o)  
Symbol(s) not found for architecture armv7  
7. Linker command failed with exit code 1 (use -v to see invocation)
```

9、错误提示：/libSDKnetec.a(NETEC\_Core.o)' does not contain bitcode. You must rebuild it with bitcode enabled (Xcode setting ENABLE\_BITCODE), obtain an updated library from the vendor, or disable bitcode for this target. for architecture armv7

解决方案：在 Build Settings 里面搜索 bitcode,把 Enable Bitcode 设置为 NO



10、网络验证授权信息失败

首先，检查网络是否畅通。

然后，检查是否更新了 Info.plist 中项目配置，以支持 HTTP 协议，如下图：

▼ NSAppTransportSecurity	Dictionary	(1 item)
NSAllowsArbitraryLoads	Boolean	YES

## 七. 更新日志

### Ver1.9.0

- 1.1、开放项目云盘模块
- 1.2、添加工具拆分支持
- 1.3、开放任务模块
- 1.4、批注功能优化
- 1.5、IM群聊优化

### 其它历史版本

### Ver1.8.1

- 1、添加备忘录功能：用户可将IM聊天内容以及批注内容转入备忘录，并设定提醒人以及提醒时间，到期后提醒用户。
- 2、开放我的邮箱功能：用户可绑定常用邮箱或使用iClap默认邮箱，支持写邮件、存草稿、发送邮件、以及标星邮件等功能。
- 3、个人云盘改版：个人云盘中添加对云盘文件的管理操作，如：重命名、移动文件夹/文件、将文件夹/文件转存入项目云盘等常用功能。
- 4、优化IM消息收发：优化IM消息收发功能，使之更符合用户操作习惯，更方便阅读。



5、更新Dev信息：更新最新的Dev信息，使内容更加丰富，更方便用户阅读和使用。

6、修复已知Bug：修复部分已知BUG，使产品更加稳定。

## **Ver 1.7.0**

- 1、兼容iPhone6s及iPhone6s Plus批注，
- 2、优化附件上传、下载
- 3、优化智能语音功能
- 4、修复已知 Bug

## **Ver 1.6.0**

- 1、支持动态界面批注，
- 2、优化游戏批注
- 3、优化截屏代码
- 4、修复已知 Bug

## **Ver 1.5.0**

- 1、更新用户角色权限管理
- 2、批注详情页面优化
- 3、增加横竖屏支持
- 4、修复已知 Bug

## **Ver 1.4.0**

- 1、适配最新 iOS 系统
- 2、优化批注点截屏功能
- 3、完善批注转为任务流程
- 4、更新不同角色用户批注权限功能

## **Ver 1.3.0**

1. 增加了批注点增量更新，节省流量，提高程序运行效率。

2. 修改库文件，进一步提高了程序兼容性和稳定性。
3. 添加了对基于 `cocos2d` 及 `cocos2d-x` 引擎开发的应用、游戏的支持。
4. 更新了批注功能授权信息的验证机制，使之更加方便使用。