

# C++ SDK Social 使用指南

---

## [SDK概述](#)

## [SDK集成](#)

## [主要功能接入](#)

## [用户系统](#)

## [好友关系](#)

## [排行榜](#)

## [礼包兑换码](#)

## [其他功能](#)

---

文档版本	修订日期	修改内容
1.0	21-Apr-2014	创建
1.1	23-May-2014	新增用户系统，新增排行榜，新增好友
1.2	23-Jul-2014	新增网游账户登录，新增修改昵称
1.3	1-Aug-2014	更新设计KTplay入口
1.4	29-Aug-2014	优化文档结构
1.5	10-Dec-2014	优化功能点描述
1.6	29-Dec-2014	新增深度链接，强通知说明，新增上周好友排行榜，新增上周玩家排行榜
1.7	08-Apr-2015	新增礼包兑换码，新增IOS KTplay跨游戏登录

## 1 SDK概述

### 1.1 SDK功能

- 游戏内社区
- 交叉推广
- 通知
- 奖励（活动）
- 使用KTplay登录界面
- 使用网游帐号登录
- 好友
- 排行榜
- 礼包兑换码
- 分享内容到KTplay社区
- 在KTplay中使用SNS登录和分享
- 插屏通知（强通知）
- 游戏深度链接

### 1.2 平台概要

#### 1.2.1 iOS

- 开发环境
  - Mac OS X 10.10+
  - Xcode 6+
- 运行环境

- 操作系统版本: iOS 5.1.1+ (如果接入Facebook 需 iOS 6.0+)
- CPU类型: arm64,armv7,armv7s,i386(模拟器)
- 设备类型: iPhone4+, iPad2+, iTouch5+, iPad-Mini+

KTPlay在不支持的设备中将自动屏蔽, 不会影响游戏正常功能

## 1.2.2 Android

- 开发环境
  - Windows 7/Windows XP
  - Eclipse 3.6+
  - Android SDK 4.0+
- 运行环境
- Android 2.2+
- 兼容设备: Android Phone

# 2 SDK 集成

提示

请先在KTPlay开发者后台中创建App, 获得AppKey和AppSecret

## 2.1 iOS 项目集成

### 第一步 下载 ktplay\_c++\_sdk\_vx.x.x.zip并解压缩

- SDK文件夹: ktplay\_c++\_sdk\_vx.x.x

### 第二步 Xcode工程配置

- 进入ktplay\_c++\_sdk\_vx.x.x/Utils目录
  - 打开终端运行./apply2xcode.sh \$gameProjectFile
  - \$gameProjectFile 为游戏 Xcode 工程文件完整路径
- 【运行后, KTPlay 所需的 Framework 和 Other Link Flags 将自动添加到你的 Xcode 工程中】

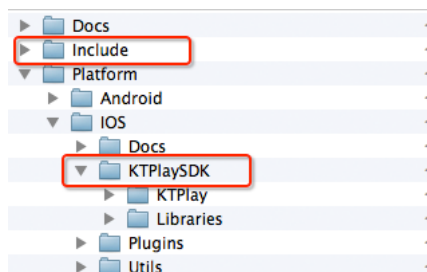
示例:

```
keiakiras-MacBook-Pro:Utils xingbin$ ./apply2xcode.sh /Volumes/Sample/kt-sample.xcodeproj
```

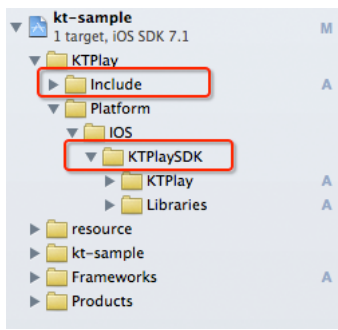
- 提示
- 通过终端控制台Log可以查看KTPlay SDK 依赖的Framework 和 Other Link Flags

### 第三步 导入KTPlay SDK

- 所需SDK文件夹



- 请在你的工程目录结构中, 右键选择Add->Existing Files..., 选择这个两个文件夹。或者将这个文件夹拖入 Xcode 工程目录结构中, 在弹出的界面中勾选Copy items into destination group's folder(if needed), 并确保Add To Targets勾选相应的target。
- 示例



#### 第四步 导入SNS SDK

KTPlay使用SNS功能用来快速登录, 分享KTPlay内容到SNS平台

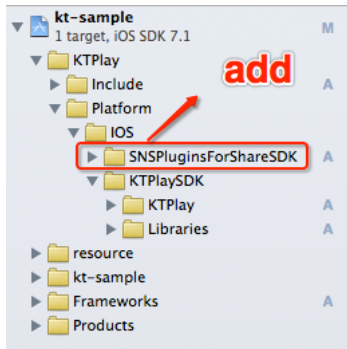
##### ShareSDK

如果你游戏已经使用ShareSDK请阅读以下内容

ShareSDK Plugins文件夹: SNSPluginsForShareSDK

- SNSPluginsForShareSDK 在 SDK文件夹ktplay\_c++\_sdk\_vx.x.x/Platform/iOS/Plugins下

示例:



- Xcode工程配置请遵循 ShareSDK 官方文档
- 代码接入请遵循 ShareSDK 官方文档

##### 提示

- 如果你确认不接入某个特定SNS, 仅需要在SNSPluginsForShareSDK中删除相对应的文件夹。

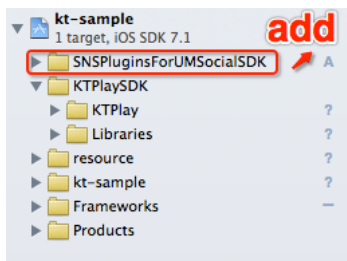
##### UMSocialSDK

如果你游戏已经使用UMSocialSDK请阅读以下内容

UMSocialSDK Plugins文件夹: SNSPluginsForUMSocialSDK

- SNSPluginsForUMSocialSDK 在SDK文件夹ktplay\_c++\_sdk\_vx.x.x/Platform/iOS/Plugins下

示例:



- Xcode工程配置请遵循 UMSocialSDK 官方文档
- 代码接入请遵循 UMSocialSDK 官方文档

##### 提示

- 如果你确认不接入某个特定SNS，仅需要在 SNSPluginsForUMSocialSDK中删除相对应的文件夹。

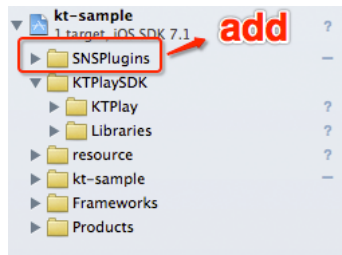
## KTSNS SDK

如果你原游戏未使用其他第三方 SNS SDK 请阅读以下内容

KTSNS Plugins文件夹：SNSPlugins

- SNSPlugins在SDK文件夹ktplay\_c++\_sdk\_vx.x.x/Platform/iOS/Plugins 下

示例：



提示

- 如果你确认不接入某个特定 SNS，仅需要在 SNSPlugins 中删除相对应的文件夹。

## 第五步 引入 KTRPlaySDK 头文件添加 KTRPlaySDK 初始化

打开AppDelegate.cpp，添加图示红色矩形框代码

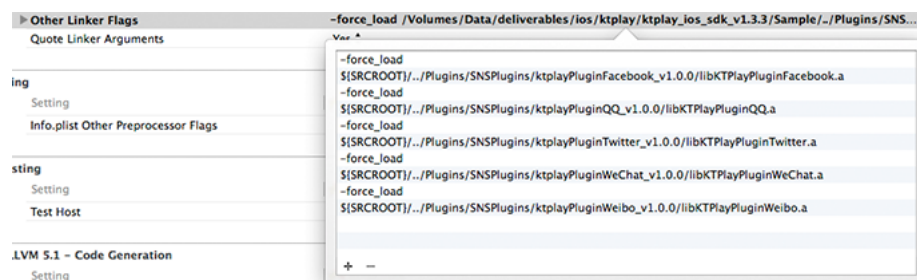
```
#import "AppDelegate.h"
#import "KTRPlay.h"

@implementation AppDelegate
@synthesize window;
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    [KTRPlay startWithAppKey:@"2na1eLK" appSecret:@"3e8bbe12d983b8d1cdc174f49d7a8448613af078"];
    //...
    return YES;
}
```

## 第六步 编译和验证接入

编译过程中可能存在的错误：

- 第三方库重复
  - 请到 KTRPlaySDK / Libraries 目录下删除重复的第三方库。
- Other Linker Flags
  - KTRPlay apply2xcode.sh脚本会追加 -ObjC 到 Other Linker Flags，如果由此导致工程Build失败，请移除-ObjC，并用以下方案中的任意一个替代：
    - 使用-all\_load
    - 使用-force\_load选项根据需求装载KTRPlay/Plugins下的类库



## 2.2 Android 项目集成

### 2.2.1 Eclipse 工程配置

#### 2.2.1.1 添加 KTRPlay 主工程

- 导入 KTRPlay 主工程到 Eclipse 游戏 Eclipse 工程中，打开菜单 File -> Import -> Android -> Existing Android Code Into Workspace,选择 KTRPlaySDK/KTRPlay 导入。

- 游戏工程中添加对 KTPlay 主工程的引用 游戏工程右键菜单，选择 Properties->Android->Library->Add,选择 KTPlay 工程添加。
- 添加 KTPlay 所需配置到游戏的 AndroidManifest 文件中
  - 打开 KTPlay 主工程的 activities.xml, 拷贝所有内容到游戏AndroidManifest 文件的 application 标签下
  - 打开 KTPlay 主工程的 permissions.xml, 拷贝所有内容到游戏AndroidManifest 文件的 manifest 标签下
  - AndroidManifest 文件中，找到名为 KTPLAY\_CHANNELID 的声明，填入 KTPlay 的 ChannelID, ChannelID 请参《KTPlay Android 渠道列表》

### 2.2.1.2 设置 JNI 环境

- 拷贝所有 KTPlaySDK/KTPlay/libs/armeabi 目录下文件到您的 Android 工程的 JNI 文件夹下
- 拷贝 Include 文件夹下的.h 文件到您的 Android 工程的 JNI 文件夹下
- 添加 KTPlay shared library 并修改jni目录下的Android.mk文件。

进入 Android 工程的 JNI 文件夹下,打开 Android.mk 文件,添加以下内容到文件中:

```
c #prebuild KTPlay library include $(CLEAR_VARS) LOCAL_MODULE := libKTPlay LOCAL_SRC_FILES := libKTPlay.so
LOCAL_MODULE_FILENAME := libKTPlay include $(PREBUILT_SHARED_LIBRARY) include $(CLEAR_VARS) LOCAL_MODULE :=
libKTAcountmanager LOCAL_SRC_FILES := libKTAcountmanager.so LOCAL_MODULE_FILENAME := libKTAcountmanager include
$(PREBUILT_SHARED_LIBRARY) include $(CLEAR_VARS) LOCAL_MODULE := libKTFriendship LOCAL_SRC_FILES := libKTFriendship.so
LOCAL_MODULE_FILENAME := libKTFriendship include $(PREBUILT_SHARED_LIBRARY) include $(CLEAR_VARS) LOCAL_MODULE :=
libKTLeaderboard LOCAL_SRC_FILES := libKTLeaderboard.so LOCAL_MODULE_FILENAME := libKTLeaderboard include
$(PREBUILT_SHARED_LIBRARY) * 游戏主模块添加依赖库
```

打开 Android.mk, 找到游戏主模块的脚本, 如果 **LOCAL\_SHARED\_LIBRARIES**已定义, 添加以下到您的脚本上: **LOCAL\_SHARED\_LIBRARIES+= libKTPlay libKTAcountmanager libKTFriendship libKTLeaderboard**

如果 **LOCAL\_SHARED\_LIBRARIES** 没有定义, 添加以下到您的脚本上: **LOCAL\_SHARED\_LIBRARIES:= libKTPlay libKTAcountmanager libKTFriendship libKTLeaderboard**

**注意事项** KTPlay动态链接库提供对以下指令集的支持

- armeabi
- armeabi-v7a
- x86
- mips

如果你的游戏不提供对某种指令集的支持，请删除KTPlay工程下对应文件夹

### 2.2.1.3 初始化

在游戏主 Activity 类的 onCreate 方法中，调用 KTPlay 类的 startWithAppKey 方法初始化KTPlay。

```
c KTPlay.startWithAppKey(Activity 对象,你的AppKey,你的AppSecret);
```

**注意事项**

- KTPlay的.so文件会在调用KTPlay.startWithAppKey时装载，请确保游戏在装载自己的.so文件之前已经调用了KTPlay.startWithAppKey方法，因为游戏的.so文件引用到KTPlay的.so文件。

### 2.2.1.4 添加基本回调

**注意：**此节所列的步骤不可跳过，否则 **KTPlay** 功能无法正常使用。

- onPause 在游戏主 Activity 类的 onPause 方法中，调用 KTPlay 类的 onPause 方法。

```
c KTPlay.onPause(Context context);
```

- onResume 在游戏主 Activity 类的 onResume 方法中，调用 KTPlay 类的 onResume 方法。

```
c KTPlay.onResume (Context context);
```

## 3 主要功能接入

### 3.1 KTPlay主窗口

#### 3.1.1 设计KTPlay主窗口入口

接入 KTPlay 前，先设计一个或多个入口来触发 KTPlay 功能，如：在游戏界面放置一个按钮，玩家点击后显示KTPlay主窗口。

**注意**

- 为保障 KTPlay在线开关功能正常，作为 KTPlay入口的按钮，只有在 KTPlay 可用时才显示
- KTPlay 的 isEnabled 方法用来判断KTPlay是否可用
- KTPlay 在不可用时所有涉及 KTPlay 功能接口将失效
- isEnabled 为实时状态。请每次实时调用，不要用变量缓存

- 当 isEnabled 状态变更时提供[设置监听者](#)、[监听KTPlay SDK的可用状态变更](#)
- KTPlay 在以下情况中不可用：
  - SDK 同步 Portal 的配置信息未完成
  - 设备不支持
  - 开发者后台关闭了 KTPlay 开关

示例:

```
c //每帧刷新游戏视图，直接使用KTPlayC::isEnabled方法来判断入口的按钮是否需要绘制 if(KTPlayC::isEnabled()){ Game::drawKtButton(); }
```

### 3.1.2 打开KTPlay主窗口

```
c KTPlayC::show();
```

提示

查看如何替换[主窗口皮肤](#)

## 3.2 设置监听者，监听奖励发放事件

注意



- 奖励功能需要先在KTPlay开发者网站配置货币和道具
- 在同一设备只能领取1次

示例:

开发者在KTPlay网站配置ID为"Gold","Diamond"的货币或道具

## 货币和道具

当游戏发布后，请谨慎修改道具和货币的ID，避免影响相关功能的使用。

Icon	奖励类型 ?	ID ?	
	金币	Gold	编辑 移除
	钻石	Diamond	编辑 移除

+ 创建

- 新建dispatchRewards方法实现奖励发放回调

```
c void dispatchRewards (KTRewardItemC * pRewardArray, int length){ for ( int i = 0; i < length; i ++ ) { if (strcmp(pRewardArray[i].typeId, "Gold") == 0) { Game.appendGold( pRewardArray[i].typeId.value) } else if (strcmp(pRewardArray[i].typeId, "Diamond") == 0) { Game.appendDiamond( pRewardArray[i].typeId.value) } } }
```

- 设置 DidDispatchRewardsCallback

```
c KTPlayC::setDidDispatchRewardsCallback( dispatchRewards );
```

### 3.3 设置监听者，监听用户新动态

KTPlay会不定时的收到消息，如一些最新回复，新通知，新奖励等，KTPlay把此类消息称之为最新动态，此时KTPlay会把它以回调的形式反馈给游戏中。

游戏接到回调后建议使用下面两种方案：

- 实现KTPlay入口按钮动画（如闪烁）
- 实现在KTPlay入口按钮上合适的位置添加小红点

示例:

- 新建 activityStatusChanged方法实现新动态提示回调。

```
c void activityStatusChanged(bool hasNewActivities) { if(hasNewActivities) { Game::showKtTip(); //显示新动态提示 } else { Game::hideKtTip(); //隐藏新动态提示 } }
```

- 设置 KTRewardItemCCallback

```
c KTPlayC::setActivityStatusChangedCallback(activityStatusChanged);
```

### 3.4 设置监听者，监听KTPlay SDK的可用状态变更

KTSDK Enable状态是可以在Portal进行控制的，因此KTPlay提供当Enable状态发生变更时触发的回调。

由[设计KTPlay主窗口入口](#) 阐述 isEnabled状态是可能发生改变。所以KTPlay提供isEnabled方法的同时，也提供当isEnabled状态发生变更时的回调。

示例:

- 新建availabilityChangedCallback方法

```
c void availabilityChangedCallback(bool isEnabled){ //isEnabled 为当前KTPlay实时状态 }
```

- 设置 KTPlayC::setAvailabilityChangedCallback

```
c KTPlayC::setAvailabilityChangedCallback(availabilityChangedCallback);
```

### 3.5 显示强通知（插屏通知）

KTPlay通知功能中提供了强通知类型，强通知在游戏中以插屏的形式展示，需要主动调用KTPlayC::showInterstitialNotification方法展示强通知。 `c KTPlayC::showInterstitialNotification();`

### 3.6 设置监听者，监听深度链接发放事件

注意，深度链接功能需要先在开发者网站配置相关内容，具体配置请遵循网站的提示

KTPlay中的深度链接功能，主要负责从服务器接收自定义字符串，当游戏开发收到该自定义字符串后根据自身逻辑处理相应的游戏流程。

- 新建KTDeepLinkBlock实现深度链接接受功能。

例如当收服务器配置"GoToLeaderboard" 字符串，并发送深度链接通知。

```
c void deepLinkCallback (const char *linkSchemas){ Game::processDeepLink(linkSchemas) //linkSchemas内容为 "GoToLeaderboard"，游戏接收到此字符串后，跳转到游戏排行榜界面 }
```

- 调用KTPlayC::setDeepLinkCallback方法设置深度链接监听。

```
c KTPlayC::setDeepLinkCallback(deepLinkCallback );
```

### 3.7 启用定位功能[IOS]

KTPlay使用了LBS相关功能，需要在Info.plist追加如下字段

NSLocationAlwaysUsageDescription

```
NSLocationAlwaysUsageDescri... String
```

## 4 用户系统

涉及用户头像图片大小请参考[缩略图](#) 章节

### 4.1 使用网游帐户登录KTPlay平台

注意

- 该接口未完全开放，调用该接口需要KTPlay运营团队提供支持，请联系support@ktplay.com

当游戏登录成功时调用该接口后，进入KTPlay后，KTPlay已经为登录状态，且KTPlay无登出功能。

示例:

- 新建loginWithGameUserCallback方法

```
c static void loginWithGameUserCallback(bool isSuccess ,const char * gameId, KTUserC * user,KTErrorC *error) { if (isSuccess) { // 操作成功 user 是当前用户信息 } else { // 操作失败 } }
```

- 调用 KTAccountManagerC::loginWithGameUser

```
KTAccountManagerC::loginWithGameUser("100010", loginWithGameUserCallback);
```

提示 + 如果希望网游昵称和KTPlay社区昵称一致，需要调用[修改当前登录帐户昵称](#)接口同步昵称

### 4.2 修改当前登录帐户昵称

注意

- KTPlay昵称可重复
- KTPlay昵称仅限制了最大长度128位
- KTPlay平台昵称会把敏感词替换成星号
- 昵称的规则限制需要游戏开发者控制后再调用该接口
- 新建setNickNameCallBack方法实现设置昵称回调

```
c static void setNickNameCallBack(bool isSuccess ,const char * nickName,KTUserC * user, KTErrorC *error) { if (isSuccess) { // user 是当前用户信息 } else { // error 详细错误信息 } }
```

- 调用 KTAccountManagerC::setNickName

```
c KTAccountManagerC::setNickName("opop", setNickNameCallBack);
```

### 4.3 打开KTPlay登录界面

- 新建showLoginViewCallback方法实现KTPlay登录界面回调

```
c static void showLoginViewCallback(bool isSuccess ,KTUserC * user,KTErrC *error) { if (isSuccess) { //登录成功 } else {  
//登录失败 } }
```

- 调用 KTAcountManagerC::showLoginView

```
c KTAcountManagerC::showLoginView(false, showLoginViewCallback);
```

### 4.4 设置监听者，监听KTPlay登录状态变更

当游戏登录状态发生变更时会以回调的方式通知游戏

示例:

- 新建loginStatusChangedCallback方法实现登录状态回调

```
c //创建回调 void loginStatusChangedCallback(bool isLoggedIn, KTUserC * user) { if(isLoggedIn) { //登录 } else { //登出 } }
```

- 调用 KTAcountManagerC::setLoginStatusChangedCallback

```
c KTAcountManagerC::setLoginStatusChangedCallback(loginStatusChangedCallback);
```

### 4.5 判断是否已有用户登录

示例:

```
c bool isLoggedIn = KTAcountManagerC::isLoggedIn(); if(isLoggedIn) { //已登录 } else { //未登录 }
```

### 4.6 获取任意KTPlay用户信息

示例:

- 新建userProfileCallback方法实现获取玩家信息回调

```
c static void userProfileCallback(bool isSuccess ,const char *userId ,KTUserC * user,KTErrC *error) { if (isSuccess) {  
// 操作成功 user即玩家信息 } else { //操作失败 } }
```

- 调用KTAcountManagerC::userProfile

```
c KTAcountManagerC::userProfile("1234" ,userProfileCallback);
```

### 4.7 获取当前登录用户信息

示例:

```
c KTUserC *userC = KTAcountManagerC::currentAccount();
```

### 4.8 登出当前帐户

示例:

```
c KTAcountManagerC::logout();
```

## 5 好友关系

### 5.1 打开添加好友界面

示例:

```
c KTFriendshipC::showFriendRequestsView();
```

### 5.2 发送好友邀请

示例:

- 新建friendRequestCallback方法实现交友邀请回调

```
c static void friendRequestCallback(bool isSuccess, int successCount, KTErrC *error) { if (isSuccess) { //操作成功  
successCount, 返回成功邀请的好友个数。 } else { //操作失败 } }
```

- 调用KTFriendshipC::sendFriendRequests

```
c char *p[] = {"123", "245"}; KTFriendshipC::sendFriendRequests(p,2,friendRequestCallback);
```

### 5.3 获取我的好友列表数据

- 新建friendListCallback方法实现KTPlay好友列表回调



```
c static void friendListCallback(bool isSuccess,KTUserC * userArray,int userArrayCount,KTErrC *error) { if (isSuccess)
{ //操作成功 userArray 获取的好友数组 } else { //操作失败 } }
```

- 调用KTFriendshipC::friendList

```
c KTFriendshipC::friendList(friendListCallback);
```

## 6 排行榜

注意

排行榜相关数据需要在开发者网站进行配置

### 6.1 获取好友排行榜数据

注意

- 如果startIndex 参数传-1 时, 返回当前登录用户排名所处位置的排行榜数据
- 排行榜分数返回的结果格式需要开发者网站配置
- 新建friendLeaderboardCallback方法实现好友排行榜数据回调

```
""c static void friendLeaderboardCallback(bool isSuccess,const char leaderboardId ,KTLeaderboardPaginatorC leaderboard,KTErrC *error) { if
(isSuccess) { //操作成功 } else { //操作失败 }
} ""
```

- 调用KTLeaderboardC::friendsLeaderboard

```
c KTLeaderboardC::friendsLeaderboard("10020", 0, 20, friendLeaderboardCallback);
```

### 6.2 获取玩家排行榜数据

注意

- 排行榜分数返回的结果格式需要开发者网站配置

示例:

- 新建globalLeaderboardCallback方法实现游戏排行榜数据回调

```
""c static void globalLeaderboardCallback(bool isSuccess,const char leaderboardId ,KTLeaderboardPaginatorC leaderboard,KTErrC *error) { if
(isSuccess) { //操作成功 }else{ //操作失败 }
} ""
```

- 调用KTLeaderboardC::globalLeaderboard

```
c KTLeaderboardC::globalLeaderboard("10020", 0, 20, globalLeaderboardCallback);
```

### 6.3 获取上周好友排行榜数据

注意

- 如果startIndex 参数传-1 时, 返回当前登录用户排名所处位置的排行榜数据
- 排行榜分数返回的结果格式需要开发者网站配置
- 新建friendLeaderboardCallback方法实现好友排行榜数据回调

```
""c static void friendLeaderboardCallback(bool isSuccess,const char leaderboardId ,KTLeaderboardPaginatorC leaderboard,KTErrC *error) { if
(isSuccess) { //操作成功 } else { //操作失败 }
} ""
```

- 调用KTLeaderboardC::lastFriendsLeaderboard

```
c KTLeaderboardC::lastFriendsLeaderboard("10020", 0, 20, friendLeaderboardCallback);
```

### 6.4 获取上周玩家排行榜数据

注意

- 排行榜分数返回的结果格式需要开发者网站配置

示例:

- 新建globalLeaderboardCallback方法实现游戏排行榜数据回调

```
""c static void globalLeaderboardCallback(bool isSuccess,const char leaderboardId ,KTLeaderboardPaginatorC leaderboard,KTErrC *error) { if
(isSuccess) { //操作成功 }else{ //操作失败 }
} ""
```

- 调用KTLeaderboardC::lastGlobalLeaderboard

```
c KTLeaderboardC::lastGlobalLeaderboard("10020", 0, 20, globalLeaderboardCallback);
```

## 6.5 上传积分数据

示例:

- 新建reportScoreCallback方法实现上传分数回调

```
c static void reportScoreCallback(bool isSuccess, const char *leaderboardId, long long score, KErrorC *error) { if (isSuccess) { //操作成功, } else { //操作失败。leaderboardId , score 为请求信息 } }
```

- 调用 KTLeaderboardC::reportScore

```
c KTLeaderboardC::reportScore("10020",100010 ,reportScoreCallback);
```

## 7 礼包兑换码

开发者可以通过KTplay开发者网站为自己的应用创建礼包并生成兑换码，通过不同的分发渠道分发给玩家。玩家获取到兑换码后进入游戏可以通过兑换码窗口输入兑换指定礼包，并领取礼包中的奖励。

### 7.1 呼出礼包兑换码窗口

示例:

```
c [KTPlayC:: showRedemptionView();
```

### 7.2 设置监听者，监听礼包发放事件

用户领取礼包成功时，SDK会触发相应的回调方法，你需要先设置setDidDispatchRewardsCallback以监听礼包发放事件，关于setDidDispatchRewardsCallback的用法请参考[3.2 设置监听者、监听奖励发放事件](#)

## 8 其他功能

### 8.1 C++其他接口

#### 8.1.1 设置监听者，监听打开KTPlay主窗口事件

示例:

- 新建viewDidAppearCallback方法实现窗口展示回调

```
c //创建回调 void viewDidAppearCallback(){ Game::pause(); }
```

- 调用 KTPlay::setViewDidAppearCallback

```
c KTPlayC::setViewDidAppearCallback(viewDidAppearCallback);
```

注意

建议在此回调方法中将游戏暂停

#### 8.1.2 设置监听者，监听关闭KTPlay主窗口事件

示例:

- 新建viewDidDisappearCallback方法实现窗口消失回调

```
c //创建回调 void onKTDisappear() { Game::resume(); }
```

- 调用 KTPlay::setViewDidDisappearCallback

```
c KTPlayC::setViewDidDisappearCallback(onKTDisappear);
```

#### 8.1.3 关闭KTPlay主窗口

示例:

```
c KTPlayC::dismiss();
```

#### 8.1.4 判断KTPlay主窗口是否处于打开状态

示例:

```
c bool isKTShowing = KTPlayC::isShowing();
```

#### 8.1.5 设置截图旋转角度

某些游戏绘制方式下，KTPlay 获取到得截图，方向可能不正确，需要开发者主动调用截图旋转方法来调整。

调用 KTPlayC 类的 setScreenshotRotation 方法来旋转截图

注意:

- 参数传的是角度而不是弧度，如 90,180,270

```
c KPlayC:: setScreenshotRotation(90)
```

### 8.1.6 截取游戏当前画面并分享到KPlay社区

```
c KPlayC::shareScreenshotToKT("大家来看看图片哈~~~");
```

### 8.1.7 分享图片/文本到KPlay社区

注意: `imagePath` 为图片的绝对路径

```
c KPlayC::shareImageToKT(imgPath , "大家来看看图片哈~~~");
```

### 8.1.8 启用/禁用通知功能

游戏中会出现 KPlay 提供的通知 UI。但在某些情况下游戏进行中并不需要显示它。可调用以下方法进行屏蔽。

注意:

首次设置 `NotificationEnabled` 为 `NO` 方法需要设置在 `startWithAppKey` 之前

- iOS

```
""c - (BOOL)application:(UIApplication)application didFinishLaunchingWithOptions:(NSDictionary)launchOptions {
KPlayC::setNotificationEnabled(false);
```

```
[KPlay startWithAppKey:@"2na1e1K"
appSecret:@"3e8bbe12d983b8d1cdc174f49d7a8448613af078"];
//...
return YES;
```

```
} ""
```

启用通知功能

```
c KPlayC::setNotificationEnabled(true);
```

## 8.2 iOS 平台相关接口

### 8.2.1 设置用于显示KPlay窗口的父容器

默认情况下，KPlay会使用 `[[[UIApplication sharedApplication] keyWindow] rootViewController] view]` 作为父view，如果遇到异常，请调用KPlay类的 `setKTParentView` 方法设置自己的view。

注意 请在 `AppDelegate` 类的 `didFinishLaunchingWithOptions` 方法中调用此方法

示例:

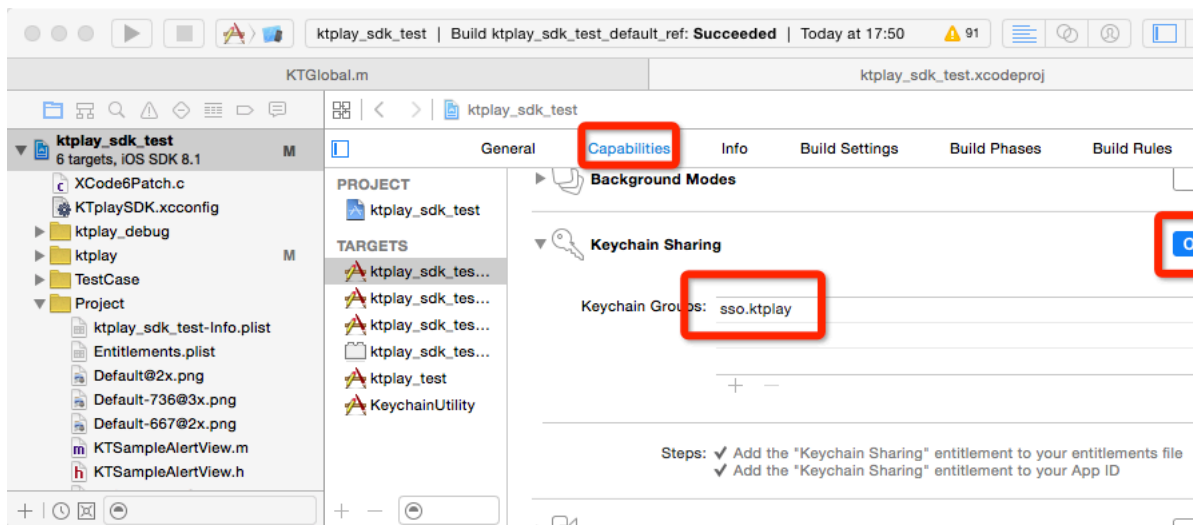
```
c [KPlay setKTParentView:myView];
```

### 8.2.2 启用KPlay跨游戏登录

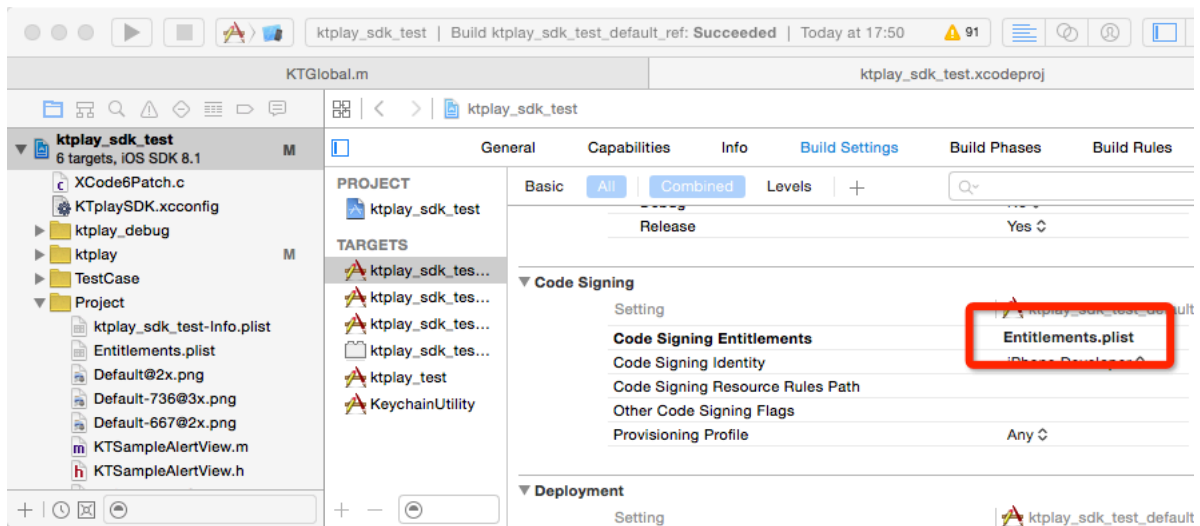
注意: KPlay跨游戏登录是指同一开发者多个游戏接入KPlay SDK后，玩家可直接使用已经注册的KPlay帐号登录新的游戏。跨游戏登录功能使用了 `Keychain Sharing` 相关技术，仅在同一苹果开发者帐号下的游戏可以实现信息共享。`Keychain` 的存储位置是由系统控制的，在删除应用之后，存储在 `Keychain` 中的数据也不会消失。

Step1 打开 Xcode -> Capabilities中Keychain Sharing 功能（打开该功能会自动生成Entitlements.plist文件）

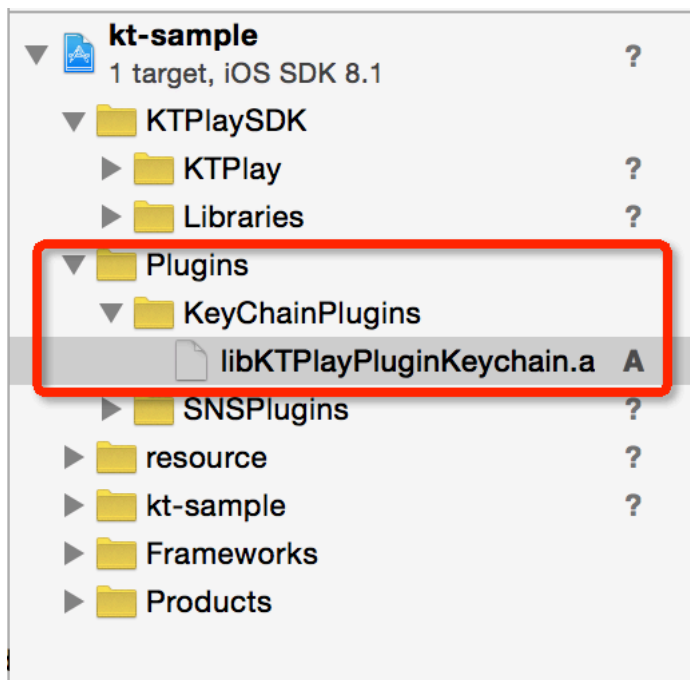
Step2 在Keychain Sharing中新增sso.kplay项



Step3 在Xcode -> Build Settings中找到Code Signing Entitlements项, 输入Entitlements.plist



Step4 确认libKTPlayPluginKeychain.a 已经被添加进工程



## 8.3 配置功能

### 8.3.1 iOS

除提供 SDK 接口之外，KTPlay 还可通过配置文件实现一些功能的定制或开关。KTPlay 配置内容统一放在游戏 Xcode 项目 Info -> Custom iOS Target Properties -> ktplay\_config 中。

#### 8.3.1.1 锁定KTPlay界面方向

默认情况下，KTPlay会根据设备的屏幕方向展示相应的内容，如果游戏的屏幕方向与设备的屏幕方向不一致，可以通过加入下列配置来锁定KTPlay的界面方向：

名称：lock\_orientation

可用值	备注
portrait	竖屏
landscape	横屏

#### Custom iOS Target Properties

Key	Type	Value
Required device capabilities	Array	(1 item)
Bundle identifier	String	com.ktplay.\${PRODUCT_NAME:rfc1034identifier}
InfoDictionary version	String	6.0
Icon files (iOS 5)	Dictionary	(1 item)
Bundle version	String	1.2
Executable file	String	\$(EXECUTABLE_NAME)
Application requires iPhone environment	Boolean	YES
Bundle name	String	\$(PRODUCT_NAME)
Supported interface orientations	Array	(3 items)
Bundle display name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle creator OS Type code	String	????
ktplay_config	Dictionary	(1 item)
lock_orientation	String	landscape
Localization native development region	String	en
Supported interface orientations (iPad)	Array	(4 items)
Bundle versions string, short	String	1.2

#### 8.3.1.2 区域切换 (不填写此字段默认为china)

名称：region

可用值	备注
china	中国, SNS默认支持新浪微博, 腾讯QQ, 微信
international	国外, SNS默认支持Facebook, Twitter
global	全球, 根据首次启动时的系统语言自动选择区域, 简体中文为中国区, 其他语言为国际区。

中国区/国际区是两个独立的服务器, 是由两个不同的portal支持, 如果想要全球统一发现, 请联系support@ktplay.com

#### ▼ Custom iOS Target Properties

Key	Type	Value
► Required device capabilities	Array	(1 item)
Bundle identifier	String	com.ktplay.\${PRODUCT_NAME:rfc1034identifier}
InfoDictionary version	String	6.0
► Icon files (iOS 5)	Dictionary	(1 item)
Bundle version	String	1.2
Executable file	String	\$(EXECUTABLE_NAME)
Application requires iPhone environment	Boolean	YES
Bundle name	String	\$(PRODUCT_NAME)
► Supported interface orientations	Array	(3 items)
Bundle display name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle creator OS Type code	String	????
▼ ktplay_config	Dictionary	(1 item)
region	String	global
Localization native development region	String	en
► Supported interface orientations (iPad)	Array	(4 items)
Bundle versions string, short	String	1.2

### 8.3.2 Android

#### 8.3.2.1 区域切换 (不填写此字段默认为china)

通过在AndroidManifest.xml中设置KTPLAY\_REGION来配置KTPlay所支持的区域

可用值	备注
China	中国, SNS默认支持新浪微博, 腾讯QQ, 微信
International	国外, SNS默认支持Facebook, Twitter
Global	全球, 根据首次启动时的系统语言自动选择区域, 简体中文为中国区, 其他语言为国际区。

中国区/国际区是两个独立的服务器, 是由两个不同的portal支持, 如果想要全球统一发现, 请联系support@ktplay.com

### 8.4 缩略图

获取图片缩略图KTPlay统一遵循任意图片url路径\_size size范围: 50x50 80x80 120x120 200x 200

例子

size	url
原图	http://dw.img.ktplay.cn/group1/M00/62/5D/CgIAEVPL7AGAltQaAAHic5V3UQU0010831
50x50	http://dw.img.ktplay.cn/group1/M00/62/5D/CgIAEVPL7AGAltQaAAHic5V3UQU0010831_50x50
80x80	http://dw.img.ktplay.cn/group1/M00/62/5D/CgIAEVPL7AGAltQaAAHic5V3UQU0010831_80x80
120x120	http://dw.img.ktplay.cn/group1/M00/62/5D/CgIAEVPL7AGAltQaAAHic5V3UQU0010831_120x120
200x200	http://dw.img.ktplay.cn/group1/M00/62/5D/CgIAEVPL7AGAltQaAAHic5V3UQU0010831_200x200

### 8.5 自定义皮肤

#### 8.5.1 iOS

为适配游戏风格, SDK提供针对KTPlay主窗口及KTPlay登录窗口换肤功能

- 主窗口皮肤
  - 下载并解压皮肤资源包ktplay\_ios\_sdk\_skin\_x.x.x.zip
  - 编辑皮肤
    - 进入ktplay\_ios\_sdk\_skin\_x.x.x/CommunityUI/RawResources编辑美术资源
    - portrait文件夹内是竖屏资源/landscape文件夹内是横屏资源
  - 运行脚本生产新的资源。
    - 进入ktplay\_ios\_sdk\_skin\_x.x.x/CommunityUI目录
    - 打开终端运行./apply\_skin.sh \$ktplaybundle
      - \$ktplaybundle 是完整路径。ktplaybundle指存放在KTPlaySDK/KTPlay/Core文件夹中ktplay.bundle
      - 新ktplay.bundle会生成到ktplay\_ios\_sdk\_skin\_x.x.x/CommunityUI目录内。
      - 使用新ktplay.bundle手动替换KTPlaySDK/KTPlay/Core/ktplay.bundle

- Clear XcodeProject 重新编译
- 登录界面皮肤（仅使用KTPlay登录界面功能）
  - 下载并解压KTPlay登录界面皮肤资源包ktplay\_ios\_sdk\_skin\_x.x.x.zip
  - 编辑皮肤
    - 进入ktplay\_ios\_sdk\_skin\_x.x.x/LoginUI/RawResources/image编辑美术资源
    - default文件夹为iphone, iphonehd, ipad公共资源,
      - ipad-hd文件夹未ipadhd资源
    - 打开ktplay\_ios\_sdk\_skin\_x.x.x/LoginUI/RawResources/color/color\_property.plist 编辑颜色
  - 运行脚本生产新的资源
    - 进入ktplay\_ios\_sdk\_skin\_x.x.x/LoginUI目录
    - 打开终端运行 ./apply\_login\_skin.sh \$ktloginbundle
      - \$ktloginbundle 是完整路径。ktloginbundle指存放在KTPlaySDK/KTPlay/AccountManager文件夹中ktlogin.bundle
      - 新ktlogin.bundle会生成到ktplay\_ios\_sdk\_skin\_x.x.x/LoginUI目录内。
        - 使用新ktlogin.bundle手动替换KTPlaySDK/KTPlay/AccountManager/ktlogin.bundle
  - Clear XcodeProject 重新编译

### 8.5.2 Android

- 在开发者网站下载皮肤资源
- 更换图片资源
  - 打开 PSD 文件，按游戏风格进行调整，保存为同名PNG文件
  - 拷贝所有PNG文件到**KTPlaySDK\KTPlay\res\drawable-xhdpi**文件夹下，替换原有文件
- 更换颜色值
  - 打开各颜色值文件(.xml),按注释说明修应颜色值以适配游戏风格
  - 拷贝所有.xml文件到**KTPlaySDK\KTPlay\res\values**文件夹下，替换原有文件