AI体育SDK iOS接入文档

`此文档引导宿主App iOS端如何接入SDK`

目录

文档

一、环境、兼容性等公共配置

> 1、开发语言：Objective-C、Swift
>
> 2、三方依赖管理：CocoaPods【安装教程1、2】
>
> 3、开发工具：**Xcode请使用最新版本**

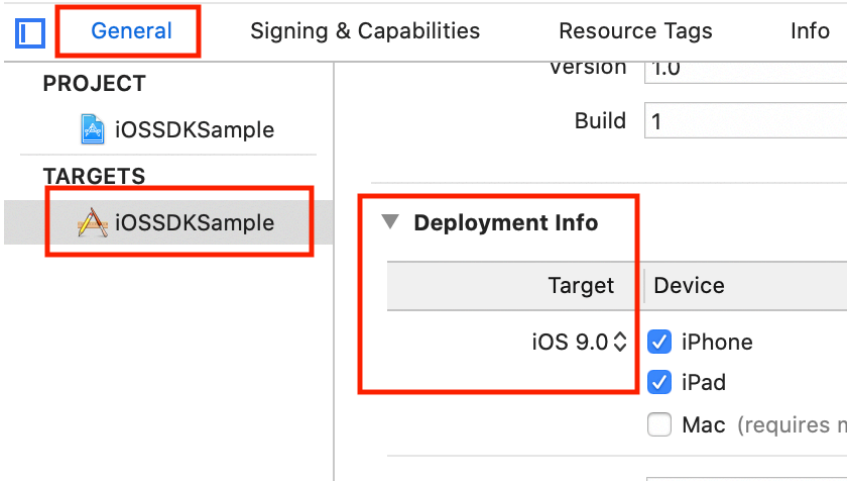
4、系统兼容性：SDK仅支持**iOS10+**系统，iPhone6或更高配手机，**现暂时仅支持真机！！！**

5、SDK引入(pods管理本地代码方式引入！！！)

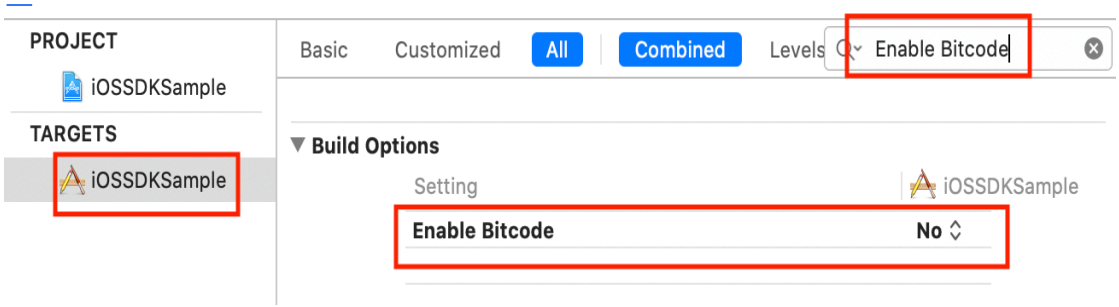
```
1、请将iOSSDK文件夹复制到工程文件夹下
2、Podfile添加:
pod 'XCSDK_iOS', :path => 'iOSSDK/iOSSDK.podspec'
pod 'AFNetworking', '~> 4.0.1' # Apple已禁止有UIWebView的IPA上传，SDK中已将版本升到4.0+
pod 'MBProgressHUD', '~> 1.1.0'
pod 'SDWebImage'
文件结构如下:
├── iOSSDKSample
│   ├── iOSSDK
│   │   ├── iOSSDK.podspec
│   │   └── frameworks
│   │         ├── App.framework
│   │       ├── XCSDK.framework
│   │       └── ...
│   ├── iOSSDKSample.xcodeproj
│   ├── iOSSDKSample.xcworkspace
│   ├── Podfile
│   ├── iOSSDKSample
│   │   └── ...
│   └── ...
└── ...
3、执行 pod install（如果失败对终端翻墙然后pod update）
```

6、工程配置

1、Deployment Target设置成10.0或更高版本



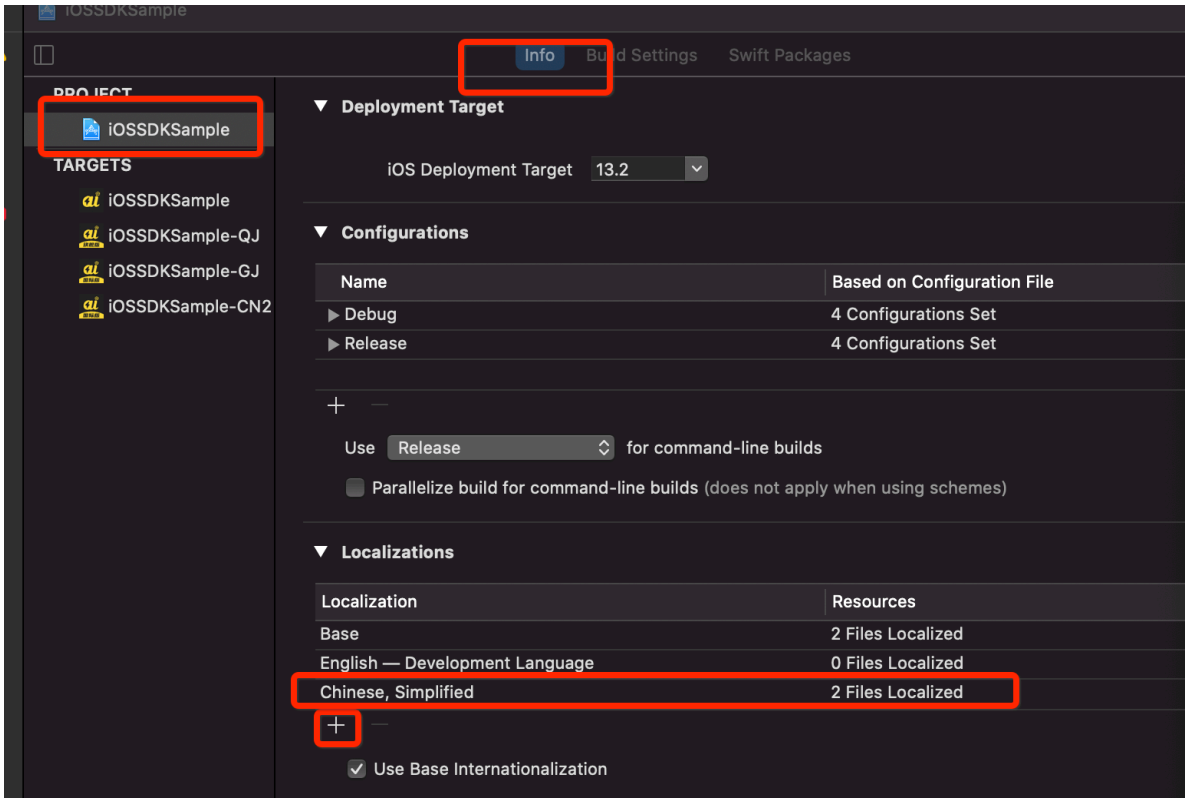
2、Enable Bitcode 设置成NO



7、info.plist(ATS与隐私设置等)

```
    <key>CFBundleDevelopmentRegion</key>
    <string>zh_CN</string>
    <key>NSAppTransportSecurity</key>
    <dict>
        <key>NSAllowsArbitraryLoads</key>
        <true/>
        <key>NSAllowsArbitraryLoadsInWebContent</key>
        <true/>
        <# 按需添加 异常域名 #>
        <key>NSExceptionDomains</key>
         <dict>
            <key>your domain eg:baidu.com</key>
            <dict>
                <key>NSExceptionAllowsInsecureHTTPLoads</key>
                <true/>
                <key>NSIncludesSubdomains</key>
                <true/>
            </dict>
         </dict>
    </dict>
    <key>NSCameraUsageDescription</key>
    <string>因更新头像等功能需要，需要使用您的相机</string>
    <key>NSMicrophoneUsageDescription</key>
    <string>因更新头像等功能需要，需要使用您的相册</string>
    <key>io.flutter.embedded_views_preview</key>
    <string>YES</string>
```

注: 如果需要App系统工具为中文（例如打开相册中的按钮及提示） 需设置CFBundleDevelopmentRegion为zh_CN 并添加中文语言包（如下图）



二、Objective-C接入说明

具体代码接入

1、ICON版方式：启动SDK(示例代码如下)

```
1、导入头文件
#import <XCSDK/XCSDK.h>
//
2、启动SDK
- (IBAction)launchSDKButton:(UIButton *)sender {
    NSError *error = nil;
    __weak __typeof(self) weakSelf = self;
    NSMutableDictionary *config = [@{
        XCSDKMainUrlKey     : self.config[XCSDKMainUrlKey] ?: @"",
        XCSDKImgUrlKey      : self.config[XCSDKImgUrlKey] ?: @"",
        XCSDKImUrlKey       : self.config[XCSDKImUrlKey] ?: @"",
        XCSDKTokenKey       : self.config[XCSDKTokenKey] ?: @"",
        XCSDKAppTypeKey       : self.config[XCSDKAppTypeKey] ?: @"",
        XCSDKShowRechargeKey: @YES,
        XCSDKShowCashKey    : @YES,
        XCSDKShowTransferKey: @YES,
        XCSDKHandledSysRepair: @(XCGetUserDefaultsBool(kHandledSysRepair))
    } mutableCopy];
    NSLog(@"config=%@", config);
    // 此方法（openSDKWithConfig:naviVC:error:handler:）即将废弃
    [XCSDKManager openIconSdk:config naviVC:self.navigationController conf
```

```
        typeof(weakSelf) self = weakSelf;
        NSLog(@"method = %@, params=%@", method, params);
        NSString *showName = nil;
        if ([method isEqualToString:XCAppMethodCashOut]) {
            showName = @"宿主App-提现页面";
            [self showSampleVC:showName];
        }
        else if ([method isEqualToString:XCAppMethodRecharge]) {
            showName = @"宿主App-充值页面";
            [self showSampleVC:showName];
        }
        else if ([method isEqualToString:XCAppMethodTransfer]) {
            showName = @"宿主App-转账页面";
            [self showSampleVC:showName];
        }
        else if ([method isEqualToString:XCAppMethodNeedLogin]) {
            [self showAnonymousLogninVC:params]; // 匿名登录模式进入，触发登录
        }
        else if ([method isEqualToString:XCAppMethodExitSDK]) {
            // @"宿主App-主动退出SDK";
            [self readUserInfo];
            [self.navigationController popViewControllerAnimated:YES ];
        }
        else if ([method isEqualToString:XCAppMethodBalanceChanged]) { //
            CGFloat balance = [params[@"value"] floatValue];
            NSLog(@"balance=%f", balance);
        }
        else if ([method isEqualToString:XCAppMethodRetryAlert]) {  // 因网
            NSUInteger index = [params[@"index"] intValue];
            if (index == 0) {} // 重试
            else if (index == 1) {} // 去检查网络
        }
        else if ([method isEqualToString:XCAppMethodSystemRepair]) { // 系
            NSUInteger code = [params[@"code"] intValue];
            NSString *msg = params[@"msg"];
            NSString *whTime = params[@"whTime"];
            NSLog(@"code=%zd, msg=%@, whTime=%@", code, msg, whTime);
        }
        else if([method isEqualToString:XCAppMethodCanAutorotate]){ // 当前
            BOOL autorotate = [params[@"autorotate"] boolValue]; // YES-支
            NSLog(@"autorotate=%d", autorotate);
            [weakSelf showHUD:autorotate ? @"开启自动旋转" : @"关闭自动旋转" a
        }
    }];
    // 错误处理
    if (!!error) {
        NSLog(@"error=%@", error);
        NSString *msg = @"其他错误";
        switch (error.code) {
            case XCErrorCode_naviVCError:
            {
```

```
                msg = @"根控制器为空或类型错误";
                break;
            }
            case XCErrorCode_configError:
            {
                msg = @"配置信息错误或缺失";
                break;
            }
            default:
            {
                //...
                break;
            }
        }
    }
}
```

2、TAB版方式：配置SDK(示例代码如下)

```
1、导入头文件
#import <XCSDK/XCSDK.h>
//
#define LazyLoadingMode1 1
#define LazyLoadingMode2 2
#define TabSDKMode LazyLoadingMode1 // 强烈建议使用懒加载方式2或1接入TabSDK
- (void)viewDidLoad {
    [super viewDidLoad];
#if TabSDKMode == LazyLoadingMode1
    // TODO:懒加载方式1：初始化viewControllers->>需要使用SDK时配置SDK->>初始化ma
    // 2.1.1、配置SDK
    BOOL ret = [self configTabSDK]; if(!ret) return;
    // 2.1.2、初始化viewControllers
    [self initViewControllers];
    // 2.1.3、使用的时候再初始化betPage或mainPage
    if (self.selectedIndex == 2) [self configSDKPage:YES]; // 默认选中mainP
#elif TabSDKMode == LazyLoadingMode2
    // TODO:懒加载方式2：初始化viewControllers->>需要使用SDK时配置SDK->>初始化ma
    // 2.2.1、初始化viewControllers
    [self initViewControllers]; self.selectedIndex = 1;
#else
    // TODO:旧方式：先配置SDK->>初始化viewControllers->>初始化betPage和mainPag
    /* 如需同时初始化betPage和mainPage，请先初始化betPage然后再初始化mainPage并用
     包装示例比如：[[UINavigationVC alloc] initWithRootViewController:mainPa
    // 1.1、配置SDK
    BOOL ret = [self configTabSDK]; if(!ret) return;
    // 1.2、初始化viewControllers
    [self initViewControllers];
    // 1.3、初始化betPage和mainPage
    self.betNavVC.viewControllers = @[[XCSDKManager betPage]];
    self.mainNavVC.viewControllers = @[[XCSDKManager mainPage]];
#endif
    [self initTabBar];
    self.delegate = self;
    self.view.backgroundColor = UIColor.orangeColor;
}
- (void)tabBarController:(UITabBarController *)tabBarController didSelec
tViewController:(UIViewController *)viewController{
#if TabSDKMode == LazyLoadingMode1 || TabSDKMode == LazyLoadingMode2
    // 2.2.2、配置SDK
    if(!self.didConfigSDK) { BOOL ret = [self configTabSDK]; if(!ret) retu
    // 2.2.3、初始化mainPage或betPage
    if(self.selectedIndex == 2) [self configSDKPage:YES];
    else if(self.selectedIndex == 3) [self configSDKPage:NO];
#endif
}
```

3、展示比赛详情页/主播页(示例代码如下)

```
1、导入头文件
```

```
#import <XCSDK/XCSDK.h>
//
2、启动SDK(较icon版多XCSDKGidmKey和XCSDKAnchorIdKey两个字段)
- (IBAction)queryAnchorListAndEnterSDK:(UIButton *)sender {
    if (self.config[XCSDKMainUrlKey] == nil || self.config[XCSDKTokenKey]
        [self showHUD:@"请先获取配置信息~" afterDelay:1.0f];
        return;
    }
    __weak __typeof(self) weakSelf = self;
    [XCSDKUtils queryAnchorList:self.config handler:^(NSArray<NSDictionary
        NSLog(@"resp=%@, error=%@", resp, error);
        typeof(weakSelf) self = weakSelf;
        if (resp.count == 0 || error != nil) {
            [self showHUD:resp.count == 0 ? @"主播列表为空，请稍后再试~" : err
            return;
        }
        NSMutableDictionary *config = [NSMutableDictionary dictionaryWithD
        [config setObject:resp.firstObject[XCSDKGidmKey] forKey:XCSDKGidmK
        [config setObject:resp.firstObject[XCSDKAnchorIdKey] forKey:XCSDKA
        NSError *configError = nil;
        [XCSDKManager showAnchorPage:config naviVC:self.navigationControll
            NSLog(@"method = %@, params=%@", method, params);
            NSString *showName = nil;
            if ([method isEqualToString:XCAppMethodCashOut]) {
                showName = @"宿主App-提现页面";
                [self showSampleVC:showName];
            }
            else if ([method isEqualToString:XCAppMethodRecharge]) {
                showName = @"宿主App-充值页面";
                [self showSampleVC:showName];
            }
            else if ([method isEqualToString:XCAppMethodTransfer]) {
                showName = @"宿主App-转账页面";
                [self showSampleVC:showName];
            }
            else if ([method isEqualToString:XCAppMethodNeedLogin]) {
            [self showAnonymousLogninVC:params]; // 匿名登录模式进入，触发登录
              }
            else if ([method isEqualToString:XCAppMethodExitSDK]) {
                // @"宿主App-主动退出SDK";
                [self readUserInfo];
                [self.navigationController popViewControllerAnimated:YES ]
            }
            else if ([method isEqualToString:XCAppMethodBalanceChanged]) {
                CGFloat balance = [params[@"value"] floatValue];
                NSLog(@"balance=%f", balance);
            }
            else if ([method isEqualToString:XCAppMethodRetryAlert]) {  //
                NSUInteger index = [params[@"index"] intValue];
                if (index == 0) {} // 重试
                else if (index == 1) {} // 去检查网络
```

```
            }
            else if ([method isEqualToString:XCAppMethodTokenTimeout]) {
                // token超时;
                if (self.handledTokenTimeoutSwitch.on) { // 外部处理
                    [self showHUDAndRelogin];
                }
            }
            else if ([method isEqualToString:XCAppMethodSystemRepair]) {
                NSUInteger code = [params[@"code"] intValue];
                NSString *msg = params[@"msg"];
                NSString *whTime = params[@"whTime"];
                NSLog(@"code=%zd, msg=%@, whTime=%@", code, msg, whTime);
            }
            else if([method isEqualToString:XCAppMethodCanAutorotate]){
                 BOOL autorotate = [params[@"autorotate"] boolValue]; // Y
                 NSLog(@"autorotate=%d", autorotate);
              [weakSelf showHUD:autorotate ? @"开启自动旋转" : @"关闭自动旋转"
             }
        }];
    }];
}
```

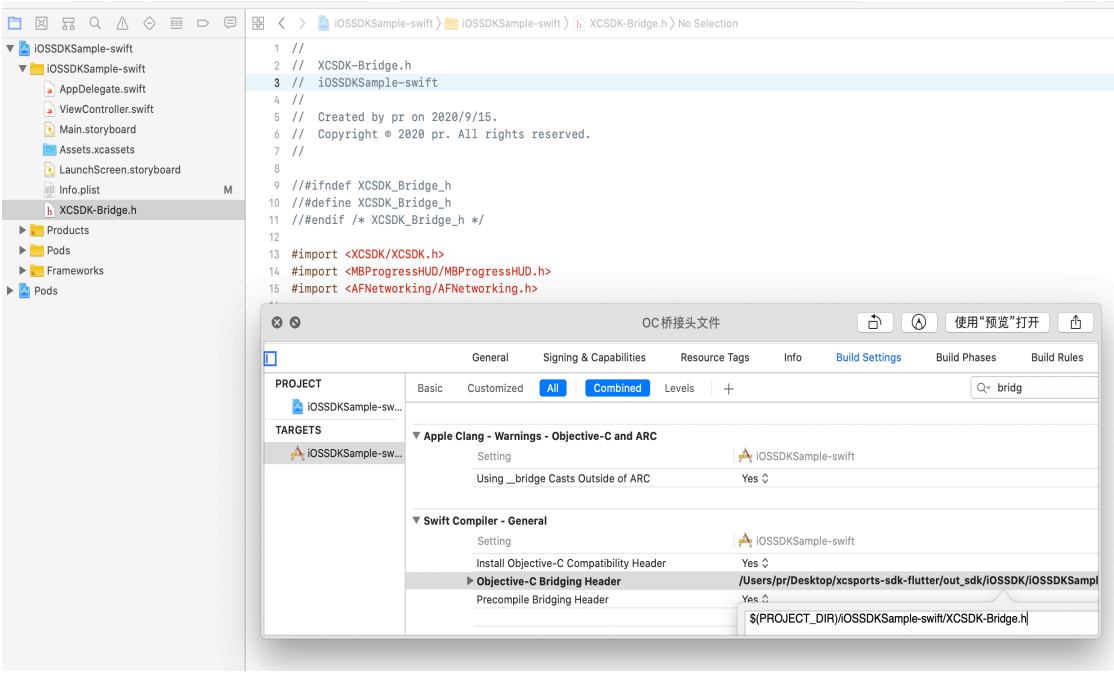
三、Swift接入说明

接入准备

1、移除多window相关配置

```
1.1、AppDelegate.swift文件中AppDelegate
  1.1.1：添加window属性：var window: UIWindow?
  1.1.2：注释或移除多屏代理方法
    //
    func application(_ application: UIApplication, configurationForCo
        return UISceneConfiguration(name: "Default Configuration", se
    }
    //
    func application(_ application: UIApplication, didDiscardSceneSes
//
1.2、SceneDelegate.swift文件解除引用
//
1.3、info.plist移除多屏配置
<key>UIApplicationSceneManifest</key>
    <dict>
        <key>UIApplicationSupportsMultipleScenes</key>
        <false/> <# ----- 此处确保为false ----- #>
        <# ----- 删除以下内容 ----- #>
        <key>UISceneConfigurations</key>
        <dict>
            <key>UIWindowSceneSessionRoleApplication</key>
            <array>
                <dict>
                    <key>UISceneConfigurationName</key>
                    <string>Default Configuration</string>
                    <key>UISceneDelegateClassName</key>
                    <string>$(PRODUCT_MODULE_NAME).SceneDelegate</str
                    <key>UISceneStoryboardFile</key>
                    <string>Main</string>
                </dict>
            </array>
        </dict>
        <# ----- 删除以上内容 ----- #>
    </dict>
```

2、添加Objective-C桥接



具体接入

```
// 获取token
func queryConfig() {
    // 模拟获取登录token
    XCSDKUtils.simulateLogin(XCSDKConfigUAT, config: ["merchantId": me
        self.data = config ?? [:];
        //NSLog("000-config=\(String(describing: config))")
    }
}
// 启动SDK
@IBAction func launchSDK(_ sender: UIButton) {
    // 组装配置
    let config = [
        XCSDKMainUrlKey: self.data[XCSDKMainUrlKey],
        XCSDKImgUrlKey: self.data[XCSDKImgUrlKey],
        XCSDKImUrlKey: self.data[XCSDKImUrlKey],
        XCSDKTokenKey: self.data[XCSDKTokenKey],
        XCSDKAppTypeKey: self.data[XCSDKAppTypeKey],
        XCSDKMerchantIdKey: merchantId
    ]
    //NSLog("111-config=\(String(describing: config))")
    // 启动SDK
    XCSDKManager.openSDK(withConfig: config as? [String : String] ?? [
        NSLog("method\(method)")
    }
}
```

四、SDK接口说明

1、配置商户信息、设置回调、启动SDK(ICON版方式)

```
/// 启动SDK(配置商户信息、设置回调)
/// @param iconSdkConfig SDK相关配置
/// @param naviVC 宿主App导航控制器，SDK导航根控制器(SDK内部仅弱引用)
/// @param configError 配置错误信息
/// @param handler SDK事件回调
+ (void)openIconSdk:(NSDictionary<NSString *, id> *_Nonnull)iconSdkConfi
g
             naviVC:(UINavigationController *_Nonnull)naviVC
        configError:(NSError *__autoreleasing *)configError
            handler:(XCSDKHandler)handler;
```

2、配置SDK信息(TAB版方式)

```
/// 配置SDK信息(返回值：NO-配置失败，YES-配置成功)
/// @param sdkConfig SDK配置配置信息
/// @param tabbarVC 宿主App tabbarVC(SDK内部仅弱引用)
/// @param configError 配置错误信息
/// @param handler  SDK回调
+ (BOOL)configTabSdk:(NSDictionary<NSString *, id> *_Nonnull)sdkConfig
            tabbarVC:(UITabBarController *_Nonnull)tabbarVC
         configError:(NSError *__autoreleasing *)configError
             handler:(XCSDKHandler)handler;
```

3、获取SDK主页面

```
/// 单独获取SDK主页面
+ (UIViewController *_Nullable)mainPage;
```

4、获取投注记录页面

```
/// 单独获取SDK投注记录页面
+ (UIViewController * _Nullable)betPage;
```

5、获取页面ID列表

```
/// 获取页面ID列表
+ (NSArray *_Nullable)pageIds;
```

6、页面释放

```
/// 页面释放
/// @param pageIds nil-释放所有页面，非nil-释放指定页面如：@"mainPage"/@"memb
erBettingPage"
+ (void)destroyByPageIds:(NSArray<NSString *> *_Nullable)pageIds;
```

7、查看SDK版本

```
+ (NSString *_Nonnull)sdkVersion;
```

8、展示详情页

```
/// 展示详情页(传主播id展示主播直播)
/// @param sdkConfig SDK与主播配置信息
/// @param naviVC 宿主App导航控制器，做为SDK导航根控制器(SDK内部仅弱引用)
/// @param configError 配置错误信息
/// @param handler  SDK回调
+ (void)showDetailPage:(NSDictionary<NSString *, id> *_Nonnull)sdkConfig
                naviVC:(UINavigationController *_Nonnull)naviVC
           configError:(NSError *__autoreleasing *)configError
               handler:(XCSDKHandler)handler;
```

9、视频播放/暂停

```
/// 暂停视频
+ (void)pauseVideo;
/// 播放视频
+ (void)startVideo;
```

```
- (void)tabBarController:(UITabBarController *)tabBarController didSelectViewController:(UIViewController *)viewController{
    NSLog(@"Native-%@", self.selectedIndex == 2 ? @"主页" : @"其他页面");
    if(self.selectedIndex == 2) {
        [XCSDKManager startVideo]; // 切回主页播放视频
    }
    else {
        [XCSDKManager pauseVideo]; // 切非主页暂停视频
    }

#if TabSDKMode == LazyLoadingMode1 || TabSDKMode == LazyLoadingMode2
    // 2.2.2、配置SDK
    if(!self.didConfigSDK) { BOOL ret = [self configTabSDK]; if(!ret) return; }
    // 2.2.3、初始化mainPage或betPage
    if(self.selectedIndex == 2) [self configSDKPage:YES];
    else if(self.selectedIndex == 3) [self configSDKPage:NO];
#endif
}
```

五、动画播放对环境要求

(1) 支持js
(2) 支持http 及 https
(3) 支持websocket
(4) 当打开关闭遮罩通知功能时，WebView 需拦截 hybrid:协议的请求。否则会报错。若没进行拦截，请勿填写参数 hybrid 或设 hybrid 为 0。
(5) Iframe，当前没使用 iframe，但为了将来的扩展，建议在 webview 中支持该特性。
(6) 是部分 android 机型设置的字体过大会导致版面错乱的情况，android 客户端应设置 webview 的默认字体大小为 100。实现代码： webView.getSettings().setTextZoom(100)

六、如何获取config配置信息与文档

> 1、在开放平台申请接入并审核通过后方获取config配置
>
> 2、接入文档/SDK/Demo下载链接
>
> 备注：如文档或SDK下载链接失效请联系客服人员

七、config配置参数说明:

key	说明	是否必传	备注
XCSDKMainUrlKey	基础url	必传	默认值：无
XCSDKImgUrlKey	资源url	必传	默认值：无
XCSDKImUrlKey	push服务url	必传	默认值：无
XCSDKTokenKey	登录token	非必传	默认值：无，不传或传空即为匿名登录
XCSDKTypeKey	SDK类型(TAB/ICON版)	必传	默认值：iOS-内部已隐式传入无需再传，安卓-需显式传入
XCSDKAppTypeKey	App页面展示类型(ai体育/Gbet/虚拟体育页面样式)	非必传	ai/Gbet/vmSports默认值：ai
XCSDKGidmKey	赛事Id	跳主播页时必传	默认值：无
		跳主播页	

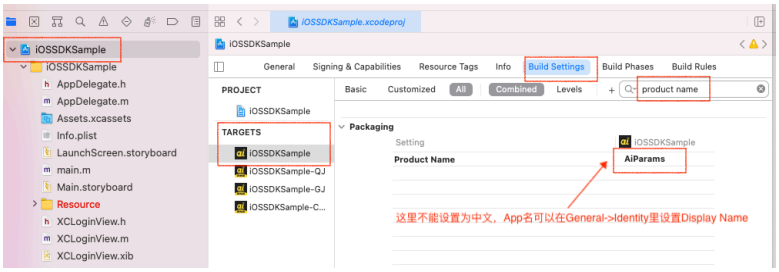
XCSDKAnchorIdKey	主播Id	时必传	默认值：无
XCSDKLaunchTitleKey	SDK启动标题	必传	默认值：不传显示"AI体育"
XCSDKLaunchSubTitlesKey	SDK启动子标题列表	非必传	默认值：见"XCSDKManager.h"
XCSDKHandledSysRepairKey	系统维护是否由商户处理	非必传	默认：NO/0-内部处理
XCSDKHandledTimeoutKey	token超时是否由商户处理	非必传	默认：NO/0-内部处理，TAB版商户必传重新获取回传给SDK
XCSDKShowRechargeKey	存款入口展示配置	非必传	默认：NO/0-不展示
XCSDKShowCashKey	取款入口展示配置	非必传	默认：NO/0-不展示
XCSDKShowTransferKey	转账入口展示配置	非必传	默认：NO/0-不展示
XCSDKMerchantIdKey	商户ID	非必传	默认值：无
XCSDKShowTabbarOnLoadingKey	加载资源或启动是否展示底部tabbar	非必传	默认值：NO/0-不展示Tabbar，仅TAB版有效开启后仅支持接入主页面
XCSDKHidesBottomBarWhenPushedKey	功能同hidesBottomBarWhenPushed	非必传	默认值：NO/0
XCSDKUmAppKey	友盟AppKey	非必传	默认值：无

XCSDKUmChannel	友盟channel	非必传	默认值：无
XCSDKUseInnerUmAppKey	是否启用SDK内部友盟	非必传	默认值：NO/0
XCSDKListStyleKey	老鸟，新手，清爽进入设置。1-小白，2-老鸟，2-小白。传0或不传，进入时用户手动设置	非必传	默认值：0

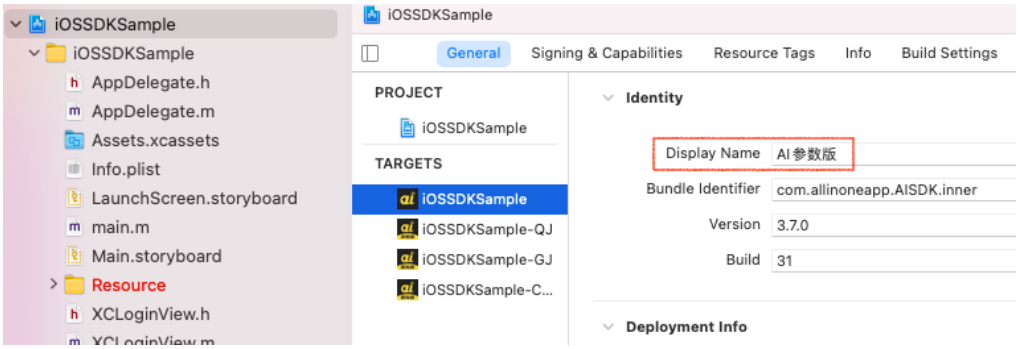
八、常见问题

1、iOS14兼容性：因SDK内部使用FlutterSDK的版本是1.22.6，此版本Product Name不支持中文(非UTF8)，配置中文(非UTF8)会导致crash，如宿主App需要设置中文(非UTF8)App名请使用Display Name来替代

Product Name配置：Product Name配置中文(非UTF8)



Display Name配置



九、版本更新说明

更改日期	版本	更改人	版本说明	审核人

2019-11-01	1.0.0V1	eric	创建	hank
2019-11-15	1.1.0V1	eric	1.1需求相关	hank
2019-11-22	1.1.1V1	eric	修改接入方式、Bug fixed	hank
2019-12-06	1.1.2V1	eric	新增配置信息、Bug fixed	abraham
2019-12-13	1.1.3V1	eric	优化接入流程、Bug fixed	abraham
2019-12-20	1.2.0V1	eric	1.2需求相关、Bug fixed	alice/tab/abraham
2019-12-27	1.2.1V1	eric	1.2需求相关、Bug fixed	isco/abraham
2020-01-03	1.3.0V1	eric	新增1.3版本功能与相关配置 Bug fixed	isco/abraham
2020-01-04	1.3.0V2	eric	新增示例代码	isco/abraham
2020-01-09	1.3.0V3	eric	去掉相关配置，优化流程	isco/abraham
2020-01-16	1.3.4V1	eric	快捷投注	isco/abraham
2020-03-01	1.5.1V1	eric	1.5.1新需求	isco/abraham
2020-03-09	1.5.1V2	eric	边沿侧滑返回宿主App可配置	isco/abraham
2020-04-03	1.6.4V1	eric	1.6.4需求	isco/abraham
2020-05-06	2.0.0V1	eric	2.0.0需求	isco/abraham
			【新增】新增了5款体育游戏，随时随地畅玩，大奖等你拿！ 【新增】新增了足球及棒球新增了多达40多种新玩法 【新增】联赛中新增了冠军玩法	

2020-05-25	2.1.0V1	eric	【新增】新增了比分栏目，随时随刻掌握最新赛事信息 【优化】全新的2.1界面升级，加入了动效，优化了交互细节，等你来细细发现 【修复】修复了横屏锁屏时无法投注的Bug 【修复】修复了关注无法更新动态的Bug	isco/abraham
2020-05-25	2.2.0V1	eric	2.2.0需求	isco/abraham
2020-06-12	2.2.0V2	eric	SDK新增启动图配置项，优化内存	isco/abraham
2020-06-23	2.2.0V3	eric	SDK减包	isco/abraham
2020-07-03	2.2.1V1	eric	2.2.1 new feature	isco/abraham
2020-07-17	2.2.2V1	eric	2.2.2 new feature	isco/abraham
2020-07-24	2.2.3V1	eric	2.2.3 new feature	isco/abraham
2020-08-27	3.0.0V1	eric	多数据源，新增页面方式集成方式	isco/abraham
2020-09-15	2.2.3V2 3.0.0V1	eric	支持Swift	isco/abraham
2020-09-15	3.4.0V1	eric	1.TabSdk启动方式API变更, 简化接入流程 2.新增余额变化通知 3.新增赛事详情页主播发红包 4.新增赛事详情注单分享 5.新增首屏推荐弹框 6.主播页面改版 7.修复已知bug	isco/abraham
2020-11-16	3.5.0V1	eric	新增TAB版token失效回调 新增系统维护回调 新增视图控制器生命周期方法回调	isco/abraham
			1.新增串关联赛筛选 2.今日页新增查看全部选项 3.移除修改密码功能 4.优化投注错误提示 5.修改串关投注最低限额为2元	

2020-12-02	3.6.0V1	eric	6.优化小视频盘口显示 7.比赛详情页新增更多功能 8.修复主播简介显示乱码问题 9.优化投注时危险球的交互与投注受理 10.修复热投人数显示错误问题 11.优化SDK登录失效回调事件(技术查看最新文档) 12.调整加载资源背景图取后台配置图片	isco/abraham
2021-01-13	3.7.0V1	eric	1.新增直接进入比赛主播直播间调用方法 2.新增横屏旋转回调事件 3.新增红包倒计时效果 4.新增提前结算功能 5.新增用户标签与主播标签 6.新增全部体育项栏目 7.优化比赛详情页性能 8.优化赔率点水交互效果 9.优化注单分享功能与跟投退单交互 10.优化列表盘口开关盘及时性 11.优化主播心愿单交互效果 12.优化比分、注单受理、主播开播、注单结算、红包等推送效果与性能 13.修复已知bug	isco/abraham
2021-02-05	3.8.0V1	eric	1.新增清除缓存 2.资源下载支持断点续传 3.Tab版新增配置 4.优化下载资源包，下载资源减少18M 5.修复已知bug，增强系统稳定性	isco/abraham
2021-03-10	3.8.5V1	eric	1.优化提前结算功能，提升结算成功率。2.新增足球注单记录查看赛果和比赛详情。 3.优化聊天室进入交互效果。 4.修复比分遮罩层无法关闭bug。 5.修复比赛直播横屏投注金额输入错误问题。 6.修复系统维护时，页面显示错误问题。	isco/abraham
2021-			1.新增主播送礼物功能 2.新增主播排行榜活动 3.搜索页新增联赛筛选功能，快速筛选想要的联赛比赛 4.所有优联赛筛选功能优化，方便快速找到指定联赛 5.优化视频播放速度，修复视频偶现变形	

04-09	3.9.0V1	eric	问题 6.新增注单延期结算标记 7.优化直播视频播放，新增视频源切换 8.新增保留上一笔投注注单 9.取消安卓震动效果 10.修复ios内存未释放问题 11.新增篮球注单查看比赛赛果	isco/abraham
2021-04-25	3.10.0	eric	3.10.0需求	ricky/abraham
2021-05-27	3.10.2	eric	3.10.2需求	ricky/abraham
2021-06-09	3.12.0	eric	3.12.0需求	ricky/abraham
2021-06-10	3.12.1	eric	3.12.1需求	ricky/abraham
2021-06-12	3.12.2	eric	3.12.2需求	ricky/abraham
2021-06-13	3.12.3	eric	3.12.3需求	ricky/abraham
2021-06-23	3.12.4	eric	3.12.4需求	ricky/abraham
2021-08-18	3.13.0	eric	3.13.0需求	ricky/abraham
2021-09-07	3.13.1	eric	3.13.1需求	ricky/abraham
2021-10-18	3.14.6	eric	3.14.6需求	ricky/abraham
2021-11-20	3.14.6	eric	1.添加老鸟，新手，清爽版设置	ricky/abraham
2021-11-29	3.14.6	eric	1.添加匿名登陆	ricky/abraham