

目录

iOS SDK 使用说明	1
基础信息	1
SDK用法	1
注意	1
CocoaPods 配置	1
Swift Package Manager (SPM) 配置	1
手动集成	2
Info.plist 配置	2
初始化	5
开屏广告	5
插屏广告	7
激励视频广告	8
其他方法说明	9
横幅广告	10
信息流广告	11
事件	12
标准事件 Listener	12
激励视频事件 RewardVideoListener	13
视频事件 VideoListener	13
暂停、恢复、销毁	14
FAQ	15
如何获取 IDFA?	15
如何配置 CAID?	16
广告加载失败怎么办?	16
如何判断广告是否已加载?	16
信息流广告在 UITableView 中如何手动计算布局?	17

iOS SDK 使用说明

基础信息

- 开发者：新义互联（北京）科技有限公司
- [开发者后台说明](#)
- [隐私声明 / 合规使用说明](#)
- [SDK更新日志](#)

SDK用法

注意

- 最低支持版本: iOS 12.0, Swift 5.7+

CocoaPods 配置

在 Podfile 中添加:

```
pod 'AdtalosAdKit'
```

然后运行:

```
pod install
```

Swift Package Manager (SPM) 配置

1. 在 Xcode 中, 选择 **File** -> **Add Packages Dependencies...**
2. Package URL

Package URL: <https://github.com/adtalos/AdtalosAdKit.git>

手动集成

1. 下载 `AdtalosAdKit.xcframework`
2. 将 `AdtalosAdKit.xcframework` 拖拽到 Xcode 项目中
3. 在 `General -> Frameworks, Libraries, and Embedded Content` 中添加 `AdtalosAdKit.xcframework`
4. 确保 `Embed` 选项设置为 `Embed & Sign`

Info.plist 配置

在 `Info.plist` 中添加以下配置以允许 HTTP 请求（用于接收广告资源）：

```
<key>NSAppTransportSecurity</key>
<dict>
  <key>NSAllowsArbitraryLoads</key>
  <true/>
</dict>
```

在 `Info.plist` 中添加以下白名单（用于相关App跳转）：

```
<key>LSApplicationQueriesSchemes</key>
<array>
  <string>quark</string>
  <string>baiduboxlite</string>
  <string>taobaoLiveshare</string>
  <string>jdmobile</string>
  <string>wireless1688</string>
  <string>weixin</string>
  <string>wxwork</string>
  <string>weixin</string>
  <string>wxwork</string>
  <string>weread</string>
  <string>wehear</string>
  <string>mqq</string>
  <string>qqmusic</string>
  <string>qqreader</string>
  <string>qqmail</string>
  <string>mqqbrowser</string>
  <string>timapi</string>
  <string>weishi</string>
  <string>qqnews</string>
  <string>tenvideo</string>
  <string>comicreader</string>
  <string>weiyun</string>
  <string>wwauthab249edd27d57738000551</string>
  <string>tencentdocs</string>
  <string>qqtranslator</string>
  <string>tencentedu</string>
  <string>qqmap</string>
  <string>dwdcoco</string>
  <string>alipay</string>
  <string>dingtalk</string>
  <string>fleamarket</string>
  <string>taobao</string>
  <string>tmall</string>
  <string>koubei</string>
  <string>eleme</string>
  <string>iosamap</string>
  <string>ucbrowser</string>
  <string>etao</string>
  <string>taobaotrail</string>
```

```
<string>xiami</string>
<string>laiwang21798646</string>
<string>youku</string>
<string>cainiao</string>
<string>tudou</string>
<string>snssdk1128</string>
<string>snssdk2329</string>
<string>snssdk1112</string>
<string>tiktok</string>
<string>feishu</string>
<string>snssdk141</string>
<string>snssdk32</string>
<string>bds</string>
<string>meituan</string>
<string>meituanwaimai</string>
<string>dianping</string>
<string>iyouxuan</string>
<string>igrocery</string>
<string>homebrew</string>
<string>merchant</string>
<string>paidian</string>
<string>crowdsourcing</string>
<string>imaicai</string>
<string>openapp.jdmobile</string>
<string>openapp.jdreader</string>
<string>newsapp</string>
<string>orpheus</string>
<string>neteaseemail</string>
<string>yanxuan</string>
<string>ntesopen</string>
<string>yddict</string>
<string>baiduboxapp</string>
<string>baiduyun</string>
<string>com.baidu.tieba</string>
<string>baidumap</string>
<string>bdbook</string>
<string>baidutranslate</string>
<string>bdwenku</string>
<string>bdvipapp</string>
<string>BaiduIMShop</string>
<string>kwai</string>
<string>ksnebula</string>
<string>bilibili</string>
<string>imgotv</string>
<string>suning</string>
<string>youdaonote</string>
<string>sinaweibo</string>
<string>weibolite</string>
<string>weibointernational</string>
<string>moke</string>
<string>douban</string>
<string>zhihu</string>
<string>xhsdiscover</string>
<string>iting</string>
<string>dedaoapp</string>
<string>dewuapp</string>
<string>QDReader</string>
<string>dragon1967</string>
<string>shuqireaderapp</string>
<string>pinduoduo</string>
<string>dmall</string>
```

```

<string>blibee</string>
<string>yitongxing</string>
<string>upwallet</string>
<string>metro</string>
<string>iqiyi</string>
<string>sohuvideo</string>
<string>sohunews</string>
<string>SogouMSE</string>
<string>yykiwi</string>
<string>bixin</string>
<string>zhuanzhuan</string>
<string>yymobile</string>
<string>oasis</string>
<string>momochat</string>
<string>smzdm</string>
<string>mtxx</string>
<string>vipshop</string>
<string>changba</string>
<string>qmkege</string>
<string>kugouURL</string>
<string>csdnplus</string>
<string>duozhuayu</string>
<string>ziroom</string>
<string>ctrip</string>
<string>qunarphone</string>
<string>xmind-zen</string>
<string>evernote</string>
<string>eudic</string>
<string>oof.disk</string>
<string>camcard</string>
<string>bocmbciphone</string>
<string>wbmain</string>
<string>douyutv</string>
<string>googlechrome</string>
<string>googlegmail</string>
<string>fb</string>
<string>firefox</string>
<string>fb-messenger</string>
<string>instagram</string>
<string>sbuxcn</string>
<string>luckycoffee</string>
<string>line</string>
<string>linkedin</string>
<string>dcard</string>
<string>youtube</string>
<string>spotify</string>
<string>nflx</string>
<string>twitter</string>
<string>whatsapp</string>
</array>

```

地理位置权限获取（可选，用于广告精准投放）：

```

<key>NSLocationWhenInUseUsageDescription</key>
<string>我们需要获取您的位置信息，以便为您提供更精准的广告内容</string>

```

IDFA 追踪权限（可选，用于广告精准投放，iOS 14+ 需要）：

```

<key>NSUserTrackingUsageDescription</key>
<string>我们需要获取您的广告标识符，以便为您提供更精准的广告内容</string>

```

初始化

Objective C

```
#import <AdtalosAdKit/AdtalosAdKit.h>

- (BOOL)application:(UIApplication *)application
    didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {

    // 创建配置
    AdtalosConfiguration *config = [[AdtalosConfiguration alloc]
        initWithToken:@"媒体 Token"
        appToken:@"应用 Token"
        idfa:@""
        caids:@[
            [[AdtalosCaid alloc]
                initWithCaid:@"Caid"
                version:@"version"],
            [[AdtalosCaid alloc]
                initWithCaid:@"Caid"
                version:@"version"]
        ]];

    // 初始化SDK
    [AdtalosSDK initialize:config];

    return YES;
}
```

Swift

```
import AdtalosAdKit

func application(_ application: UIApplication,
    didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -
    > Bool {

    // 创建配置
    let config = Configuration(
        token: "媒体 Token",
        appToken: "应用 Token",
        idfa: "",
        caids: [
            Caid(caid: "Caid", version: "version"),
            Caid(caid: "Caid", version: "version")
        ]
    )

    // 初始化SDK
    SDK.initialize(config)

    return true
}
```

::: tip 注意 注意，其中的 **<媒体 Token>** 对应开发者平台 -> 账户 -> 个人信息里面的 token。 **<应用 Token>** 对应开发者平台 -> 应用信息 -> 应用列表中的您所注册应用的 token。 :::

开屏广告

Objective C

```
// 测试广告位: 5DDF6A347D99DA1D426625E847513D6F
@interface ViewController () <AdtalosListener>
```

```
@property (nonatomic, strong) AdtalosSplashAd *splashAd;
@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];

    self.splashAd = [[AdtalosSplashAd alloc] initWithUnitID:@"广告位 TOKEN"];
    self.splashAd.listener = self;
    self.splashAd.autoRetry = 0; //自动重试次数, 详情见文档末尾说明
    [self.splashAd load];
}

- (void)onRendered {
    // 显示广告
    [self.splashAd show];
}

- (void)onClosed {
    if (self.splashAd) {
        //销毁广告
        [self.splashAd destroy];
        self.splashAd = nil;
    }
}

- (void)dealloc {
    if (self.splashAd) {
        //销毁广告
        [self.splashAd destroy];
        self.splashAd = nil;
    }
}
}
```

@end

Swift

```
// 测试广告位: 5DDF6A347D99DA1D426625E847513D6F
class ViewController: UIViewController, Listener {
    var splashAd: SplashAd?

    override func viewDidLoad() {
        super.viewDidLoad()

        splashAd = SplashAd(unitID: "广告位 TOKEN")
        splashAd?.listener = self
        splashAd?.autoRetry = 0 //自动重试次数, 详情见文档末尾说明
        splashAd?.load()
    }

    func onRendered() {
        // 显示广告
        splashAd?.show()
    }

    func onClosed() {
        if let ad = splashAd {
            //销毁广告
            ad.destroy()
            splashAd = nil
        }
    }
}
```

```
    }  
  }  
  
  deinit {  
    if let ad = splashAd {  
      //销毁广告  
      ad.destroy()  
      splashAd = nil  
    }  
  }  
}
```

插屏广告

Objective C

```
// 测试广告位: 828C5CDAA5CE4772970BEB29416FA6B4  
@interface ViewController () <AdtaloListener>  
@property (nonatomic, strong) AdtaloInterstitialAd *interstitialAd;  
@end  
  
@implementation ViewController  
  
- (void)loadInterstitialAd {  
    self.interstitialAd = [[AdtaloInterstitialAd alloc] initWithUnitID:@"广告位 TOKEN"];  
    self.interstitialAd.listener = self;  
    [self.interstitialAd load];  
}  
  
- (void)onLoaded {  
    // 显示广告  
    [self.interstitialAd show];  
}  
  
- (void)onClosed {  
    if (self.interstitialAd) {  
        //销毁广告  
        [self.interstitialAd destroy];  
        self.interstitialAd = nil;  
    }  
}  
  
- (void)dealloc {  
    if (self.interstitialAd) {  
        //销毁广告  
        [self.interstitialAd destroy];  
        self.interstitialAd = nil;  
    }  
}  
  
@end
```

Swift

```
// 测试广告位: 828C5CDAA5CE4772970BEB29416FA6B4  
class ViewController: UIViewController, Listener {  
    var interstitialAd: InterstitialAd?  
  
    func loadInterstitialAd() {  
        interstitialAd = InterstitialAd(unitID: "广告位 TOKEN")  
        interstitialAd?.listener = self  
        interstitialAd?.load()  
    }  
}
```

```
    }

    func onLoaded() {
        // 显示广告
        interstitialAd?.show()
    }

    func onClosed() {
        if let ad = interstitialAd {
            //销毁广告
            ad.destroy()
            interstitialAd = nil
        }
    }

    deinit {
        if let ad = interstitialAd {
            //销毁广告
            ad.destroy()
            interstitialAd = nil
        }
    }
}
```

激励视频广告

Objective C

// 测试广告位: C5DC50CE250B45F11BB5A70AE8BC557E

```
@interface ViewController () <AdtaloListener, AdtaloRewardVideoListener>
@property (nonatomic, strong) AdtaloRewardVideoAd *rewardVideoAd;
@end
```

```
@implementation ViewController
```

```
- (void)loadRewardVideoAd {
    self.rewardVideoAd = [[AdtaloRewardVideoAd alloc] initWithUnitID:@"广告位 TOKEN"];
    self.rewardVideoAd.listener = self;
    self.rewardVideoAd.videoListener = self;
    [self.rewardVideoAd load];
}

- (void)onLoaded {
    // 显示广告
    [self.rewardVideoAd show];
}

- (void)onRewarded:(NSString *)data {
    // 处理激励事件
    NSLog(@"激励数据: %@", data);
}

- (void)onClosed {
    if (self.rewardVideoAd) {
        //销毁广告
        [self.rewardVideoAd destroy];
        self.rewardVideoAd = nil;
    }
}

- (void)dealloc {
```



```

        if (self.rewardVideoAd) {
            //销毁广告
            [self.rewardVideoAd destroy];
            self.rewardVideoAd = nil;
        }
    }

@end

Swift

// 测试广告位: C5DC50CE250B45F11BB5A70AE8BC557E
class ViewController: UIViewController, Listener, RewardVideoListener {
    var rewardVideoAd: RewardVideoAd?

    func loadRewardVideoAd() {
        rewardVideoAd = RewardVideoAd(unitID: "广告位 TOKEN")
        rewardVideoAd?.listener = self
        rewardVideoAd?.videoListener = self
        rewardVideoAd?.load()
    }

    func onLoaded() {
        // 显示广告
        rewardVideoAd?.show()
    }

    func onRewarded(_ data: String) {
        // 处理激励事件
        print("激励数据: \(data)")
    }

    func onClosed() {
        if let ad = rewardVideoAd {
            //销毁广告
            ad.destroy()
            rewardVideoAd = nil
        }
    }

    deinit {
        if let ad = rewardVideoAd {
            //销毁广告
            ad.destroy()
            rewardVideoAd = nil
        }
    }
}

```

其他方法说明

```

/**
 * 广告是否已加载且有效
 */
public var isLoading: Bool

/**
 * 设置重试次数
 * @param times, 小于等于 0 时禁用自动重试
 */
public var autoRetry: Int

```

横幅广告

Objective C

```
// 测试广告位: 65F7C6A0A01358281D08D5781C5CB718
@interface ViewController () <AdtaloListener>
@property (nonatomic, strong) AdtaloBannerAd *bannerAd;
@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];

    // 创建横幅广告, 指定位置
    self.bannerAd = [[AdtaloBannerAd alloc] initWithCGRect:CGRectMake(0, 50)
                                                                    unitID:@"广告位 TOKEN"];

    self.bannerAd.listener = self;
    [self.bannerAd load];
}

// 广告加载完成
- (void)onLoaded {
}

// 广告渲染完成
- (void)onRendered {
    // 显示广告
    UIView *adView = self.bannerAd.view;
    if (adView) {
        [self.view addSubview:adView];
    }
}

- (void)dealloc {
    if (self.bannerAd) {
        [self.bannerAd destroy];
        self.bannerAd = nil;
    }
}

@end
```

Swift

```
// 测试广告位: 65F7C6A0A01358281D08D5781C5CB718
class ViewController: UIViewController, Listener {
    var bannerAd: BannerAd?

    override func viewDidLoad() {
        super.viewDidLoad()

        // 创建横幅广告, 指定位置
        bannerAd = BannerAd(CGPoint(x: 0, y: 50), unitID: "广告位 TOKEN")
        bannerAd?.listener = self
        bannerAd?.load()
    }

    // 广告加载完成
    func onLoaded() {
    }
}
```

```

// 广告渲染完成
func onRendered() {
    // 显示广告
    if let adView = bannerAd?.view {
        view.addSubview(adView)
    }
}

deinit {
    if let ad = bannerAd {
        ad.destroy()
        bannerAd = nil
    }
}
}

```

::: tip 注意 注意， `view` 在 `onLoaded` 事件就可以调用，但是在插入到视图中显示时，最好在 `onRendered` 事件之后再插入，可以避免界面刷新的过程。 :::

信息流广告

Objective C

```

// 测试广告位: 7E6DF6872DD50BF85A8C07700E9C4C5B
@interface FeedViewController () <AdtaloListener>
@property (nonatomic, strong) AdtaloFeedAd *feedAd;
@end

@implementation FeedViewController

- (void)viewDidLoad {
    [super viewDidLoad];

    self.feedAd = [[AdtaloFeedAd alloc] initWithCGPointMake(0, 0)
                                                            unitID:@"广告位 TOKEN"];
    self.feedAd.listener = self;
    [self.feedAd load];
}

- (void)onLoaded {
    // 显示广告
    UIView *adView = self.feedAd.view;
    if (adView) {
        [self.containerView addSubview:adView];
    }
}

- (void)onRendered {
    // 广告渲染完成
}

- (void)dealloc {
    if (self.feedAd) {
        [self.feedAd destroy];
        self.feedAd = nil;
    }
}

@end

```

Swift

```
// 测试广告位: 7E6DF6872DD50BF85A8C07700E9C4C5B
class FeedViewController: UIViewController, Listener {
    var feedAd: FeedAd?

    override func viewDidLoad() {
        super.viewDidLoad()

        feedAd = FeedAd(CGPoint(x: 0, y: 0), unitID: "广告位 TOKEN")
        feedAd?.listener = self
        feedAd?.load()
    }

    func onLoaded() {
        // 显示广告
        if let adView = feedAd?.view {
            containerView.addSubview(adView)
        }
    }

    func onRendered() {
        // 广告渲染完成
    }

    deinit {
        if let ad = feedAd {
            ad.destroy()
            feedAd = nil
        }
    }
}
```

::: tip 注意 注意，请及时销毁超出屏幕的广告：在滚动过程中，监听可见 cell 的变化，及时销毁超出屏幕的广告实例，以减少资源消耗。 :::

事件

标准事件 Listener Objective C

```
@protocol AdtaloListener <NSObject>
```

```
@optional
```

```
// 请求广告前触发。
```

```
- (void)onBeforeRequest;
```

```
// 当广告加载完毕后触发。
```

```
- (void)onLoaded;
```

```
// 当广告加载失败后触发。
```

```
- (void)onFailedToLoad:(NSError *)error;
```

```
// 当广告渲染完毕后触发。
```

```
- (void)onRendered;
```

```
// 当广告展示完毕后触发。
```

```
- (void)onShown;
```

```
// 当广告中的有效连接被点击时触发。
```

```
- (void)onClicked;
```

```
// 当用户点击打开其他应用（如外部浏览器、Deeplink等），从而当前应用在后台运行时触发。
```

```
- (void)onLeftApplication;
```

```
// 当广告关闭时触发。
```

```
- (void)onClosed;
```

```
@end
```

```
Swift
```

```
@objc public protocol Listener: NSObjectProtocol {
    @objc optional func onBeforeRequest()
    @objc optional func onLoaded()
    @objc optional func onFailedToLoad(_ error: Error)
    @objc optional func onRendered()
    @objc optional func onShown()
    @objc optional func onClicked()
    @objc optional func onLeftApplication()
    @objc optional func onClosed()
}
```

激励视频事件 **RewardVideoListener** Objective C

```
@protocol AdtalosRewardVideoListener <AdtalosListener>
```

```
@optional
```

```
// 当激励发生时触发。
```

```
- (void)onRewarded:(NSString *)data;
```

```
@end
```

```
Swift
```

```
@objc public protocol RewardVideoListener: Listener {
    @objc optional func onRewarded(_ data: String)
}
```

视频事件 **VideoListener** Objective C

```
@protocol AdtalosVideoListener <NSObject>
```

```
@optional
```

```
// 视频加载时触发
```

```
- (void)onVideoLoad:(AdtalosVideoMetadata *)metadata;
```

```
// 开始播放视频
```

```
- (void)onVideoStart;
```

```
// 视频播放中
```

```
- (void)onVideoPlay;
```

```
// 视频暂停
```

```
- (void)onVideoPause;
```

```
// 视频播放结束
```

```
- (void)onVideoEnd;
```

```
// 视频音量变化
```

```
// @param volume, 音量值 (0.0 - 1.0)
```

```
// @param muted, 是否静音
```

```
- (void)onVideoVolumeChange:(double)volume muted:(BOOL)muted;
```

```
// 视频播放进度更新
```

```
// @param currentTime, 当前播放时间 (秒)
```

```
// @param duration, 视频总时长 (秒)
```

```
- (void)onVideoTimeUpdate:(double)currentTime duration:(double)duration;
```

```

// 视频播放错误
- (void)onVideoError;

// 视频播放中断
- (void)onVideoBreak;

@end

Swift

@objc public protocol VideoListener: NSObjectProtocol {
    /// 视频加载时触发
    @objc optional func onVideoLoad(_ metadata: VideoMetadata)

    /// 开始播放视频
    @objc optional func onVideoStart()

    /// 视频播放中
    @objc optional func onVideoPlay()

    /// 视频暂停
    @objc optional func onVideoPause()

    /// 视频播放结束
    @objc optional func onVideoEnd()

    /// 视频音量变化
    /// - Parameters:
    ///   - volume: 音量值 (0.0 - 1.0)
    ///   - muted: 是否静音
    @objc optional func onVideoVolumeChange(_ volume: Double, muted: Bool)

    /// 视频播放进度更新
    /// - Parameters:
    ///   - currentTime: 当前播放时间 (秒)
    ///   - duration: 视频总时长 (秒)
    @objc optional func onVideoTimeUpdate(_ currentTime: Double, duration: Double)

    /// 视频播放错误
    @objc optional func onVideoError()

    /// 视频播放中断
    @objc optional func onVideoBreak()
}

```

所有广告类型都可通过 `listener` / `videoListener` 属性来设置和获取。

暂停、恢复、销毁

每种广告都有一个 `destroy` 方法，用于销毁对象，防止内存泄漏。如果可能的话，尽量在销毁对象时调用一下该方法。

另外，对于横幅和信息流广告还提供了 `pause` 和 `resume` 两个方法，用于手动暂停和恢复广告的播放。需要注意的是，SDK 内部已实现当广告不在可见区域时自动暂停和播放的功能，因此如无特殊需求，通常无需手动调用这两个方法。

Objective C

```

// 销毁广告
- (void)destroy;

// 暂停广告（仅横幅和信息流广告）

```

```
- (void)pause;
```

```
// 恢复广告（仅横幅和信息流广告）
```

```
- (void)resume;
```

```
Swift
```

```
// 销毁广告
```

```
@MainActor
```

```
public func destroy()
```

```
// 暂停广告（仅横幅和信息流广告）
```

```
@MainActor
```

```
public func pause()
```

```
// 恢复广告（仅横幅和信息流广告）
```

```
@MainActor
```

```
public func resume()
```

FAQ

如何获取 IDFA？ 注意： IDFA 的获取需要开发者在自己的 APP 中实现，SDK 不会自动获取。

在 iOS 14.5+ 中，需要用户授权才能获取 IDFA。开发者需要在 APP 中通过 `AppTrackingTransparency` 框架请求授权，获取到 IDFA 后传入 SDK 的 `Configuration`。

Swift 示例：

```
import AppTrackingTransparency
```

```
import AdSupport
```

```
func requestIDFA() {
    if #available(iOS 14, *) {
        ATTrackingManager.requestTrackingAuthorization { status in
            if status == .authorized {
                let idfa = ASIdentifierManager.shared().advertisingIdentifier.uuidString
                // 使用 idfa 初始化 SDK
                let config = Configuration(
                    token: "媒体 Token",
                    appToken: "应用 Token",
                    idfa: idfa, // 传入获取到的 IDFA
                    caids: []
                )
                SDK.initialize(config)
            } else {
                // 用户拒绝授权，使用空字符串初始化
                let config = Configuration(
                    token: "媒体 Token",
                    appToken: "应用 Token",
                    idfa: "",
                    caids: []
                )
                SDK.initialize(config)
            }
        }
    } else {
        // iOS 14 以下版本直接获取
        let idfa = ASIdentifierManager.shared().advertisingIdentifier.uuidString
        let config = Configuration(
            token: "媒体 Token",
            appToken: "应用 Token",
            idfa: idfa,

```

```

        caids: []
    )
    SDK.initialize(config)
}

```

Objective-C 示例:

```

#import <AppTrackingTransparency/AppTrackingTransparency.h>
#import <AdSupport/AdSupport.h>

```

```

- (void)requestIDFA {
    if (@available(iOS 14, *)) {
        [ATTrackingManager requestTrackingAuthorizationWithCompletionHandler:^(ATTrackingManagerAuthorizationStatus status) {
            NSString *idfa = @"";
            if (status == ATTrackingManagerAuthorizationStatusAuthorized) {
                idfa = [[[ASIdentifierManager sharedManager] advertisingIdentifier] UUIDString];
            }
            // 使用 idfa 初始化 SDK
            AdtalosConfiguration *config = [[AdtalosConfiguration alloc]
                initWithToken:@"媒体 Token"
                appToken:@"应用 Token"
                idfa:idfa
                caids:@[]];
            [AdtalosSDK initialize:config];
        }];
    } else {
        // iOS 14 以下版本直接获取
        NSString *idfa = [[[ASIdentifierManager sharedManager] advertisingIdentifier] UUIDString];
        AdtalosConfiguration *config = [[AdtalosConfiguration alloc]
            initWithToken:@"媒体 Token"
            appToken:@"应用 Token"
            idfa:idfa
            caids:@[]];
        [AdtalosSDK initialize:config];
    }
}

```

重要提示: - 需要在 `Info.plist` 中添加 `NSUserTrackingUsageDescription` 权限说明, 否则无法弹出授权弹窗 - 建议在合适的时机 (如用户同意隐私政策后) 调用授权请求 - 如果用户拒绝授权, 可以传入空字符串 `""` 初始化 SDK

如何配置 CAID? CAID (中国广告标识符) 用于中国市场的广告追踪。在初始化 SDK 时, 可以通过 `Configuration` 的 `caids` 参数传入:

```

let caids = [
    Caid(caid: "XXXXXXXXXXXXXXXXXXXX", version: "XXXXX"),
    Caid(caid: "XXXXXXXXXXXXXXXXXXXX", version: "XXXXX")
]
let config = Configuration(token: "媒体 Token", appToken: "应用 Token", idfa: "", caids: caids)

```

广告加载失败怎么办? SDK 提供了自动重试机制, 可以通过设置 `autoRetry` 属性来控制重试次数:

```

ad.autoRetry = 5 // 默认重试 5 次
ad.autoRetry = 0 // 禁用自动重试

```

如何判断广告是否已加载? 可以通过 `isLoading` 属性来判断:

```

if ad.isLoading {
    // 广告已加载, 可以显示
    ad.show()
}

```



```
}
```

信息流广告在 UITableView 中如何手动计算布局？ 在 UITableView 中使用信息流广告时，如果需要对 cell 高度进行手动计算，可以参考以下实现方式：

核心要点：

1. **使用字典管理多个广告实例：** 由于信息流广告通常会插入到列表的多个位置，需要使用字典来管理不同位置的广告实例。
2. **从广告视图获取实时高度：** 在 `heightForRowAtIndexPath` 方法中，直接从 `feedAd.view.frame.size.height` 获取广告视图的实时高度。
3. **监听广告视图尺寸变化：** 使用 KVO 监听广告视图的 `frame` 变化，当广告内容加载完成后，通知 tableView 更新高度。
4. **及时销毁超出屏幕的广告：** 在滚动过程中，监听可见 cell 的变化，及时销毁超出屏幕的广告实例，以减少资源消耗。

Objective-C 实现示例：

```
@interface FeedViewController () <UITableViewDelegate, UITableViewDataSource>
@property (nonatomic, strong) NSMutableDictionary<NSNumber *, AdtalosFeedAd *> *feedAds;
@property (nonatomic, strong) NSMutableSet<NSNumber *> *visibleAdPositions;
@end

@implementation FeedViewController

- (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(NSIndexPath *)indexPath {
    if ([self isAdPosition:indexPath.row]) {
        NSInteger adPosition = [self adPositionIndexForRow:indexPath.row];
        NSNumber *positionKey = @(adPosition);

        // 直接从广告视图获取实时高度
        AdtalosFeedAd *feedAd = self.feedAds[positionKey];
        if (feedAd && feedAd.isLoaded && feedAd.view) {
            CGFloat height = feedAd.view.frame.size.height;
            if (height > 0) {
                return height;
            }
        }
        return 0.0; // 广告未加载时返回 0
    }
    return 100.0; // 普通 cell 高度
}

// 监听广告视图尺寸变化，更新 cell 高度
- (void)feedAdCell:(FeedAdCell *)cell didUpdateAdViewSize:(CGSize)size {
    NSIndexPath *indexPath = [self.tableView indexPathForCell:cell];
    if (indexPath) {
        // 使用 beginUpdates 和 endUpdates 来平滑更新高度
        [self.tableView beginUpdates];
        [self.tableView endUpdates];
    }
}

// 滚动结束时，更新可见广告列表，销毁超出屏幕的广告
- (void)scrollViewDidEndDecelerating:(UIScrollView *)scrollView {
    [self updateVisibleAds];
}

- (void)updateVisibleAds {
    NSArray *visibleIndexPaths = [self.tableView indexPathsForVisibleRows];
```

```

NSMutableSet *newVisibleAds = [NSMutableSet set];

// 收集当前可见的广告位置
for (NSIndexPath *indexPath in visibleIndexPaths) {
    if ([self isAdPosition:indexPath.row]) {
        NSInteger pos = [self adPositionIndexForRow:indexPath.row];
        [newVisibleAds addObject:@(pos)];
    }
}

// 销毁离开屏幕的广告
for (NSNumber *pos in self.visibleAdPositions) {
    if (![newVisibleAds containsObject:pos]) {
        [self destroyAdAtPosition:pos.integerValue];
    }
}

self.visibleAdPositions = newVisibleAds;
}

- (void)destroyAdAtPosition:(NSInteger)position {
    NSNumber *positionKey = @(position);
    AdtalosFeedAd *feedAd = self.feedAds[positionKey];

    if (feedAd) {
        // 先清理 listener，避免回调访问已销毁的实例
        feedAd.listener = nil;
        feedAd.videoListener = nil;
        [feedAd destroy];
        [self.feedAds removeObjectForKey:positionKey];
    }
}

@end

```

Swift 实现示例：

```

class FeedViewController: UIViewController {
    var feedAds: [Int: FeedAd] = [:]
    var visibleAdPositions: Set<Int> = []

    func tableView(_ tableView: UITableView, heightForRowAt indexPath: IndexPath) -> CGFloat {
        if isAdPosition(indexPath.row) {
            let adPosition = adPositionIndexForRow(indexPath.row)

            // 直接从广告视图获取实时高度
            if let feedAd = feedAds[adPosition],
               feedAd.isLoaded,
               let adView = feedAd.view,
               adView.frame.height > 0 {
                return adView.frame.height
            }
            return 0.0 // 广告未加载时返回 0
        }
        return 100.0 // 普通 cell 高度
    }

    // 监听广告视图尺寸变化
    func feedAdCell(_ cell: FeedAdCell, didUpdateAdViewSize size: CGSize) {
        if let indexPath = tableView.indexPath(for: cell) {
            // 使用 beginUpdates 和 endUpdates 来平滑更新高度
            tableView.beginUpdates()

```

```

        tableView.endUpdates()
    }
}

// 滚动结束时，更新可见广告列表
func scrollViewDidEndDecelerating(_ scrollView: UIScrollView) {
    updateVisibleAds()
}

func updateVisibleAds() {
    guard let visibleIndexPaths = tableView.indexPathsForVisibleRows else { return }

    var newVisibleAds: Set<Int> = []

    // 收集当前可见的广告位置
    for indexPath in visibleIndexPaths {
        if isAdPosition(indexPath.row) {
            let pos = adPositionIndexForRow(indexPath.row)
            newVisibleAds.insert(pos)
        }
    }

    // 销毁离开屏幕的广告
    for pos in visibleAdPositions {
        if !newVisibleAds.contains(pos) {
            destroyAdAtPosition(pos)
        }
    }

    visibleAdPositions = newVisibleAds
}

func destroyAdAtPosition(_ position: Int) {
    guard let feedAd = feedAds[position] else { return }

    // 先清理 listener，避免回调访问已销毁的实例
    feedAd.listener = nil
    feedAd.videoListener = nil
    feedAd.destroy()
    feedAds.removeValue(forKey: position)
}
}

```

重要提示：

- **及时销毁超出屏幕的广告：**当广告滚动出屏幕时，务必调用 `destroy()` 方法销毁广告实例，以减少内存占用和资源消耗。建议在 `scrollViewDidEndDecelerating` 或 `scrollViewDidEndDragging` 中更新可见广告列表并销毁不可见的广告。
- **监听广告视图尺寸变化：**广告内容可能在加载完成后才确定最终高度，因此需要监听广告视图的 `frame` 变化，并通过 `tableView.beginUpdates()` 和 `tableView.endUpdates()` 来更新 `cell` 高度。
- **使用字典管理多个广告：**由于信息流中可能有多个广告位置，使用字典（`key` 为位置索引）来管理不同位置的广告实例，方便查找和销毁。
- **避免重复加载：**在加载广告前，检查是否已有实例或正在加载中，避免重复创建和加载。