快手广告iOS-SDK接入文档

升级iOS14注意事项

App Tracking Transparency (ATT) 适用于请求用户授权，访问与应用相关的数据已跟踪用户或者设备。访问 https://developer.apple.com/documentation/apptrackingtransparency了解更多信息

配置ATT支持

从 iOS 14 开始，若开发者设置 App Tracking Transparency 向用户申请跟踪授权，在用户授权之前IDFA 将不可用。 如果用户拒绝此请求，应用获取到的 IDFA 将是0000-0000-0000-0000，可能会导致广告收入降低
要获取 App Tracking Transparency 权限，请更新 Info.plist，添加 NSUserTrackingUsageDescription 字段和自定义文案描述。
注意：iOS 14.5之前，应用不配置ATT也能正常获取idfa，从iOS14.5之后，应用必须配置ATT，调用授权方法向用户申请ATT权限。

代码示例：

```
<key>NSUserTrackingUsageDescription</key>
<string>{{使用描述，需要开发者自行填写}}</string>
```

向用户申请权限：

```
#import <AppTrackingTransparency/AppTrackingTransparency.h>
[ATTrackingManager requestTrackingAuthorizationWithCompletionHandler:^(ATTrackingManagerAuthorizationStatus status)
{
    // Tracking authorization completed. Start loading ads here.
    // [self loadAd];
}];
```

3.3.10版本已全面支持SKAdNetwork

需要开发者在info.plist中配置skadnetworkid，访问（https://developer.apple.com/documentation/storekit/skadnetwork）了解更多信息

```
<key>SKAdNetworkItems</key>
<array>
    <dict>
        <key>SKAdNetworkIdentifier</key>
        <string>r3y5dwb26t.skadnetwork</string>
    </dict>
</array>
```

1. 接入准备

接入SDK前请您按照要求在快手手平台申请appId，申请appId时需要您填写应用名称和应用包名等信息。

2. 接入入SDK

推荐使用 CocoaPods 方式接入，CocoaPods 会协助自动处理编译问题。

2.1 CocoaPods 方式接入

在 Podfile 里指明依赖：

```
pod 'KSAdSDK', '3.0.0.2'
```

2.2 手动引入

1. 将 SDK （KSAdSDK.framework）文件加入工程（手动拖动时，选择 Copy items if needed）；
2. 在工程上链接上这些系统库:

```
'Foundation', 'UIKit', 'MobileCoreServices', 'CoreGraphics', 'Security', 'SystemConfiguration', 'CoreTelephony', 'Ad
Support', 'CoreData', 'StoreKit', 'AVFoundation', 'MediaPlayer', 'CoreMedia', 'WebKit', 'Accelerate', 'CoreLocation'
, 'AVKit', 'MessageUI', 'QuickLook', 'AddressBook', 'libz.tbd', 'resolv.9', 'sqlite3', 'c++', 'c++abi', 'CoreMotion'
```

3. 注意：为了方便模拟器开发，SDK 带有 x86_64, i386 架构。在打发布到 AppStore 的安装包时需要移除这两个架构（CocoaPods 方式接入会自动移除）。移除脚本可以参考：https://stackoverflow.com/questions/30547283/submit-to-app-store-issues-unsupported-architecture-x86

2.3 添加权限

1. 工程plist文件设置，点击右边的information Property List后边的 "+" 展开
2. 添加 App Transport Security Settings，先点击左侧展开箭头，再点右侧加号，Allow Arbitrary Loads 选项自动加入

```
<key>NSAppTransportSecurity</key>
    <dict>
```

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

2.4 SDK初始化

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

    NSString *appId = @"90010";
    [KSAdSDKManager setAppId:appId];
    // 根据需要设置日志级别
    [KSAdSDKManager setLoglevel:KSAdSDKLogLevelOff];

    // 个性化推荐开关：关闭后，看到的广告数量不变，相关度将降低。
    // 是否允许开启广告的个性化推荐（NO-关闭，YES-开启），由开发者通过SDK的接口来设置。不设置的话则默认为YES。
    [KSAdSDKManager setEnablePersonalRecommend:YES];

    // 程序化推荐开关：关闭后，看到的广告数量不变，但将不会为你推荐程序化广告。
    // 是否允许开启广告的程序化推荐（NO-关闭，YES-开启），由开发者通过SDK的接口来设置。不设置的话则默认为YES。
    [KSAdSDKManager setEnableProgrammaticRecommend:YES];
```

3. 接口使用

3.1 激励视频广告

初始化激励视频广告

目前已支持服务端回调（需要在 ssp后台配置），对应参数可通过KSRewardedVideoModel传递，具体实现如下：

```
    NSString *posId = @"90010001";
    KSRewardedVideoModel *model = [KSRewardedVideoModel new];
    //如果开启服务端回调的话，userId和extra会通过回调接口回传给接入方
    model.userId = @"123234";
    model.extra = @"test extra";
    self.rewardedVideoAd = [[KSRewardedVideoAd alloc] initWithPosId:self.posId rewardedVideoModel:model];
    self.rewardedVideoAd.delegate = self;
```

预下载广告视频：

```
[self.rewardedVideoAd loadAdData];
```

预下载会回调如下delegate方法

```
/**
 This method is called when video ad material loaded successfully.
 */
- (void)rewardedVideoAdDidLoad:(KSRewardedVideoAd *)rewardedVideoAd;

/**
 This method is called when video ad materia failed to load.
 @param error : the reason of error
 */
- (void)rewardedVideoAd:(KSRewardedVideoAd *)rewardedVideoAd didFailWithError:(NSError *_Nullable)error;

/**
 This method is called when cached successfully.
 */
- (void)rewardedVideoAdVideoDidLoad:(KSRewardedVideoAd *)rewardedVideoAd;
```

播放激励视频

```
// 默认竖屏播放
```

```
- (BOOL)showAdFromRootViewController:(UIViewController *)rootViewController;
// 默认竖屏播放
- (BOOL)showAdFromRootViewController:(UIViewController *)rootViewController showScene:(NSString *)showScene;
// 横屏/竖屏播放
- (BOOL)showAdFromRootViewController:(UIViewController *)rootViewController showScene:(nullable NSString *)showScene
 direction:(KSAdShowDirection)direction
```

播放激励视频会回调如下delegate方法

```
/**
 This method is called when video ad slot will be showing.
 */
- (void)rewardedVideoAdWillVisible:(KSRewardedVideoAd *)rewardedVideoAd;
/**
 This method is called when video ad slot has been shown.
 */
- (void)rewardedVideoAdDidVisible:(KSRewardedVideoAd *)rewardedVideoAd;
/**
 This method is called when video ad is about to close.
 */
- (void)rewardedVideoAdWillClose:(KSRewardedVideoAd *)rewardedVideoAd;
/**
 This method is called when video ad is closed.
 */
- (void)rewardedVideoAdDidClose:(KSRewardedVideoAd *)rewardedVideoAd;
/**
 This method is called when video ad is clicked.
 */
- (void)rewardedVideoAdDidClick:(KSRewardedVideoAd *)rewardedVideoAd;
/**
 This method is called when video ad play completed or an error occurred.
 @param error : the reason of error
 */
- (void)rewardedVideoAdDidPlayFinish:(KSRewardedVideoAd *)rewardedVideoAd didFailWithError:(NSError *_Nullable)error
;
/**
 This method is called when the video begin to play.
 */
- (void)rewardedVideoAdStartPlay:(KSRewardedVideoAd *)rewardedVideoAd;
```

用户相关行为回调

```
/**
 This method is called when the user clicked skip button.
 */
- (void)rewardedVideoAdDidClickSkip:(KSRewardedVideoAd *)rewardedVideoAd;
/**
 This method is called when the user close video ad.
 */
- (void)rewardedVideoAd:(KSRewardedVideoAd *)rewardedVideoAd hasReward:(BOOL)hasReward;
/**
 This method is called when the user close video ad,support staged rewards.
 */
- (void)rewardedVideoAd:(KSRewardedVideoAd *)rewardedVideoAd
              hasReward:(BOOL)hasReward
               taskType:(KSAdRewardTaskType)taskType
        currentTaskType:(KSAdRewardTaskType)currentTaskType;
```

> 具体请参考Demo中`KSRewardedVideoDemoViewController`类

3.2 全屏视频广告

初始化全屏视频广告

```
self.posId = @"90010002";
self.fullscreenVideoAd = [[KSFullscreenVideoAd alloc] initWithPosId:self.posId];
self.fullscreenVideoAd.delegate = self;
```

预下载广告视频：

```
[self.fullscreenVideoAd loadAdData];
```

预下载会回调如下delegate方法

```
/**
 This method is called when video ad material loaded successfully.
 */
- (void)fullscreenVideoAdDidLoad:(KSFullscreenVideoAd *)fullscreenVideoAd;
/**
 This method is called when video ad materia failed to load.
 @param error : the reason of error
 */
- (void)fullscreenVideoAd:(KSFullscreenVideoAd *)fullscreenVideoAd didFailWithError:(NSError *_Nullable)error;
/**
 This method is called when cached successfully.
 */
- (void)fullscreenVideoAdVideoDidLoad:(KSFullscreenVideoAd *)fullscreenVideoAd;
```

播放全屏视频

```
// 默认竖屏播放
- (BOOL)showAdFromRootViewController:(UIViewController *)rootViewController;
// 默认竖屏播放
- (BOOL)showAdFromRootViewController:(UIViewController *)rootViewController showScene:(NSString *)showScene;
// 横屏/竖屏播放
- (BOOL)showAdFromRootViewController:(UIViewController *)rootViewController showScene:(nullable NSString *)showScene
 direction:(KSAdShowDirection)direction;
```

播放全屏视频会回调如下delegate方法

```
/**
 This method is called when video ad slot will be showing.
 */
- (void)fullscreenVideoAdWillVisible:(KSFullscreenVideoAd *)fullscreenVideoAd;
/**
 This method is called when video ad slot has been shown.
 */
- (void)fullscreenVideoAdDidVisible:(KSFullscreenVideoAd *)fullscreenVideoAd;
/**
 This method is called when video ad is about to close.
 */
- (void)fullscreenVideoAdWillClose:(KSFullscreenVideoAd *)fullscreenVideoAd;
/**
 This method is called when video ad is closed.
 */
- (void)fullscreenVideoAdDidClose:(KSFullscreenVideoAd *)fullscreenVideoAd;

/**
 This method is called when video ad is clicked.
 */
- (void)fullscreenVideoAdDidClick:(KSFullscreenVideoAd *)fullscreenVideoAd;
/**
 This method is called when video ad play completed or an error occurred.
 @param error : the reason of error
 */
- (void)fullscreenVideoAdDidPlayFinish:(KSFullscreenVideoAd *)fullscreenVideoAd didFailWithError:(NSError *_Nullable
)error;
/**
 This method is called when the video begin to play.
 */
- (void)fullscreenVideoAdStartPlay:(KSFullscreenVideoAd *)fullscreenVideoAd;
```

用户相关行为回调

```
/**
 This method is called when the user clicked skip button.
 */
- (void)fullscreenVideoAdDidClickSkip:(KSFullscreenVideoAd *)fullscreenVideoAd;
```

> 具体请参考Demo中`KSFullscreenVideoDemoViewController`类

3.3 媒体自渲染广告

初始化

```
    self.nativeAdsManager = [[KSNativeAdsManager alloc] initWithPosId:posId];
    self.nativeAdsManager.delegate = self;
    [self.nativeAdsManager loadAdDataWithCount:5];
```

回调成功

```
#pragma mark - KSNativeAdsManagerDelegate
- (void)nativeAdsManagerSuccessToLoad:(KSNativeAdsManager *)adsManager nativeAds:(NSArray<KSNativeAd *> *_Nullable)n
ativeAdDataArray {
    [KSAdDemoLogManager logWithModule:KSAdDebugLogModuleNative format:@"nativeAdsManagerSuccessToLoad:nativeAds:"];
    NSInteger count = self.dataArray.count;
    for (KSNativeAd *nativeAd in nativeAdDataArray) {
        NSInteger index = arc4random() % count;
        [self.dataArray insertObject:nativeAd atIndex:index];
    }
    [self.tableView reloadData];
}

- (void)nativeAdsManager:(KSNativeAdsManager *)adsManager didFailWithError:(NSError *_Nullable)error {
    [KSAdDemoLogManager logWithModule:KSAdDebugLogModuleNative format:@"nativeAdsManager:didFailWithError:%@", error];
}

#pragma mark - KSNativeAdDelegate
- (void)nativeAdDidLoad:(KSNativeAd *)nativeAd {
}

- (void)nativeAd:(KSNativeAd *)nativeAd didFailWithError:(NSError *_Nullable)error {
}

- (void)nativeAdDidBecomeVisible:(KSNativeAd *)nativeAd {
}

- (void)nativeAdDidClick:(KSNativeAd *)nativeAd withView:(UIView *_Nullable)view {
}

- (void)nativeAdDidShowOtherController:(KSNativeAd *)nativeAd interactionType:(KSAdInteractionType)interactionType {
}
- (void)nativeAdDidCloseOtherController:(KSNativeAd *)nativeAd interactionType:(KSAdInteractionType)interactionType
{
}
```

开放字段

```
typedef NS_ENUM(int, KSAdSourceLogoType) {
    KSAdSourceLogoTypeWhite,
    KSAdSourceLogoTypeGray,
};

@interface KSMaterialMeta : NSObject

/// interaction types supported by ads.
@property (nonatomic, assign) KSAdInteractionType interactionType;

/// material pictures.
@property (nonatomic, strong) NSArray<KSAdImage *> *imageArray;
```

```
/// ad logo,可能为空
- (NSString *)adSourceLogoURL:(KSAdSourceLogoType)type;
/// ad source.
@property (nonatomic, copy) NSString *adSource;

@property (nonatomic, strong, nullable) KSAdImage *appIconImage;

/// 0-5
@property (nonatomic, assign) CGFloat appScore;
/// downloadCountDesc.
@property (nonatomic, copy) NSString *appDownloadCountDesc;


/// ad description.
@property (nonatomic, copy) NSString *adDescription;

/// text displayed on the creative button.
@property (nonatomic, copy) NSString *actionDescription;

/// display format of the in-feed ad, other ads ignores it.
@property (nonatomic, assign) KSAdMaterialType materialType;

// video duration
@property (nonatomic, assign) NSInteger videoDuration;

@property (nonatomic, strong) KSAdImage *videoCoverImage;
@property (nonatomic, copy) NSString *videoUrl;
// app name
@property (nonatomic, copy) NSString *appName;
// product name (for h5)
@property (nonatomic, copy) NSString *productName;
```

> 具体请参考Demo中`KSNativeAdsManagerFeedDemoViewController`类

3.4 SDK渲染广告

初始化

```
self.feedAdsManager = [[KSFeedAdsManager alloc] initWithPosId:self.posId size:CGSizeMake(proposalWidth, 0)];
    self.feedAdsManager.delegate = self;
    [self.feedAdsManager loadAdDataWithCount:3];
```

回调

```
#pragma mark - KSFeedAdsManagerDelegate
- (void)feedAdsManagerSuccessToLoad:(KSFeedAdsManager *)adsManager nativeAds:(NSArray<KSFeedAd *> *_Nullable)feedAdD
ataArray {
    self.title = @"数据加载成功";
    [self refreshWithData:adsManager];
}

- (void)feedAdsManager:(KSFeedAdsManager *)adsManager didFailWithError:(NSError *_Nullable)error {
    self.title = @"数据加载失败";
}
#pragma mark - KSFeedAdDelegate
- (void)feedAdViewWillShow:(KSFeedAd *)feedAd {
    [KSAdDemoLogManager logWithModule:KSAdDebugLogModuleOther format:@"%@", NSStringFromSelector(_cmd)];
}
- (void)feedAdDidClick:(KSFeedAd *)feedAd {
    [KSAdDemoLogManager logWithModule:KSAdDebugLogModuleOther format:@"%@", NSStringFromSelector(_cmd)];
}
- (void)feedAdDislike:(KSFeedAd *)feedAd {
    [KSAdDemoLogManager logWithModule:KSAdDebugLogModuleOther format:@"%@", NSStringFromSelector(_cmd)];
    NSMutableArray *ary = [self.dataArray mutableCopy];
    [ary removeObject:feedAd];
    self.dataArray = ary;
    [self.tableView reloadData];
}
- (void)feedAdDidShowOtherController:(KSFeedAd *)nativeAd interactionType:(KSAdInteractionType)interactionType {
```

```
    [KSAdDemoLogManager logWithModule:KSAdDebugLogModuleOther format:@"%@", NSStringFromSelector(_cmd)];
}
- (void)feedAdDidCloseOtherController:(KSFeedAd *)nativeAd interactionType:(KSAdInteractionType)interactionType {
    [KSAdDemoLogManager logWithModule:KSAdDebugLogModuleOther format:@"%@", NSStringFromSelector(_cmd)];
}
```

> 具体请参考Demo中`KSFeedAdsManagerFeedDemoViewController`类
> 注意：3.3.2版本之后的Feed 广告支持动态模板，广告布局样式可以通过后台配置，Demo中的KSFeedAdCell中不需要再手动设置间距

```
@implementation KSFeedAdCell
- (void)refreshWithFeedAd:(KSFeedAd *)feedAd soundEnable:(BOOL)soundEnable {

    UIView *view = [self.contentView viewWithTag:100];
    [view removeFromSuperview];
    feedAd.feedView.tag = 100;
    [feedAd setVideoSoundEnable:soundEnable];
    [self.contentView addSubview:feedAd.feedView];
    [feedAd.feedView mas_remakeConstraints:^(MASConstraintMaker *make) {
        make.edges.equalTo(@0);
    }];
}
@end
```

3.5 Draw竖版信息流广告

初始化

```
self.drawAdsManager = [[KSDrawAdsManager alloc] initWithPosId:posId];
self.drawAdsManager.delegate = self;
[self.drawAdsManager loadAdDataWithCount:1];
```

回调

```
#pragma mark - KSDrawAdsManagerDelegate
- (void)drawAdsManagerSuccessToLoad:(KSDrawAdsManager *)adsManager drawAds:(NSArray<KSDrawAd *> *_Nullable)drawAdDat
aArray {
    NSInteger index = MIN(self.dataArray.count / 2, 2);
    NSMutableArray *mutAry = [self.dataArray mutableCopy];
    for (NSInteger i = 0; i < adsManager.data.count; i++) {
        KSDrawAd *drawAd = adsManager.data[i];
        [mutAry insertObject:drawAd atIndex:(index + i)];
    }
    self.dataArray = mutAry;
    [self.tableView reloadData];
}
- (void)drawAdsManager:(KSDrawAdsManager *)adsManager didFailWithError:(NSError *_Nullable)error {
}
#pragma mark - KSDrawAdDelegate
- (void)drawAdViewWillShow:(KSDrawAd *)drawAd {
}
- (void)drawAdDidClick:(KSDrawAd *)drawAd {
}
- (void)drawAdDidShowOtherController:(KSDrawAd *)drawAd interactionType:(KSAdInteractionType)interactionType {
}
- (void)drawAdDidCloseOtherController:(KSDrawAd *)drawAd interactionType:(KSAdInteractionType)interactionType {
}
```

> 具体请参考Demo中`KSDrawAdDemoViewController`类

3.6 开屏广告

使用KSSplashAdView实现开屏（KSAdSplashManager已废弃， 3.3.10版本开始支持ssp设置跳过按钮出现时间和是否显示倒计时，需要使用KSSplashAdView实现）示例如下：

```
//初始化开屏广告
self.splashAdView = [[KSSplashAdView alloc] initWithPosId:posId];
```

```
self.splashAdView.delegate = self;
/*
 * warning: 开屏广告必须设置rootViewController
 */
self.splashAdView.rootViewController = root;
//视频播放5s后进入小窗
self.splashAdView.needShowMiniWindow = YES;
//加载开屏广告
[self.splashAdView loadAdData];

#pragma -mark KSSplashAdDelegate
- (void)ksad_splashAdDidLoad:(KSSplashAdView *)splashAdView {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
}

- (void)ksad_splashAdContentDidLoad:(KSSplashAdView *)splashAdView {
    //展示开屏广告
    self.splashAdView.frame = [UIScreen mainScreen].bounds;
    [self.splashAdView showInView:self.window];
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
}

- (void)ksad_splashAdDidVisible:(KSSplashAdView *)splashAdView {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
}

- (void)ksad_splashAdVideoDidBeginPlay:(KSSplashAdView *)splashAdView {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
}

- (void)ksad_splashAd:(KSSplashAdView *)splashAdView didFailWithError:(nonnull NSError *)error {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
    [self removeSplash];
}

- (void)ksad_splashAd:(KSSplashAdView *)splashAdView didSkip:(NSTimeInterval)playDuration {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
    if (!splashAdView.showingMiniWindow) {
        [self removeSplash];
    }
}

- (BOOL)ksad_splashAd:(KSSplashAdView *)splashAdView willZoomTo:(inout CGRect *)frame {
    //开屏广告即将展示小窗，可以在这里修改小窗位置
    return YES;
}

- (void)ksad_splashAd:(KSSplashAdView *)splashAdView didClick:(BOOL)onZoomOutState {
    //点击跳转了
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
}

- (void)ksad_splashAdDidAutoDismiss:(KSSplashAdView *)splashAdView {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
    [self removeSplash];
}

- (void)ksad_splashAdDidClose:(KSSplashAdView *)splashAdView {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
    [self removeSplash];
}

- (void)removeSplash {
    if (self.splashAdView.showingMiniWindow) {
        //可以自己指定任意动画方式
        [UIView animateWithDuration:.25f animations:^{
            self.splashAdView.transform = CGAffineTransformMakeScale(0.1, 0.1);
            self.splashAdView.alpha = 0;
        } completion:^(BOOL finished) {
            [self.splashAdView removeFromSuperview];
            self.splashAdView = nil;
        }];
```

```
    } else {
        [self.splashAdView removeFromSuperview];
        self.splashAdView = nil;
    }
}
```

3.7 插屏广告

插屏广告支持图片，视频，实现方式如下（注意：插屏广告展示时不支持屏幕旋转，需要接入方限制）：

```
    // 获取插屏视频广告
    self.interstitialAd = [[KSInterstitialAd alloc] initWithPosId:self.posId containerSize:self.navigationController.v
    self.interstitialAd.delegate = self;
    [self.interstitialAd loadAdData];
```

```
    #pragma mark - KSInterstitialAdDelegate
- (void)ksad_interstitialAdDidLoad:(KSInterstitialAd *)interstitialAd {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
}

- (void)ksad_interstitialAdDidClick:(KSInterstitialAd *)interstitialAd {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
}

- (void)ksad_interstitialAdDidClose:(KSInterstitialAd *)interstitialAd {
    self.showingInterstitalAd = NO;
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
    [self resetInterstitialAd];
}

- (void)ksad_interstitialAdWillVisible:(KSInterstitialAd *)interstitialAd {
    self.showingInterstitalAd = YES;
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
}

- (void)ksad_interstitialAdDidVisible:(KSInterstitialAd *)interstitialAd {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
}

- (void)ksad_interstitialAdRenderSuccess:(KSInterstitialAd *)interstitialAd {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
    self.playBtn.enabled = YES;
}

- (void)ksad_interstitialAdRenderFail:(KSInterstitialAd *)interstitialAd error:(NSError *)error {
    [[KSBulletScreenManager sharedInstance] showWithText:error.description];
}

- (void)ksad_interstitialAdDidCloseOtherController:(KSInterstitialAd *)interstitialAd interactionType:(KSAdInteracti
onType)interactionType {
    [[KSBulletScreenManager sharedInstance] showWithText:NSStringFromSelector(_cmd)];
    [self resetInterstitialAd];
}
```

3.8 媒体二次议价

在获取到广告 model（FeedAd、nativeAd等）后媒体可在 winnotice 时设置二次议价，接口如下:

```
@interface KSAd : NSObject

// 单位:分，只有视频资源下载成功后，这个才可能有值
@property (nonatomic, readonly) NSInteger ecpm;
/// 媒体二次议价，单位分
- (void)setBidEcpm:(NSInteger)ecpm;

@end
```

3.9 广告曝光失败后上报失败原因

```
@interface KSAd : NSObject

/// 广告曝光失败后上报失败原因
/// @param failureCode 曝光失败原因类型
/// @param reportParam 曝光失败原因描述 reportParam.winEcpm 胜出者的ecpm报价（单位：分）
- (void)reportAdExposureFailed:(KSAdExposureFailureCode)failureCode reportParam:(KSAdExposureReportParam *)reportPar
am;

@end
```

4. 错误码

code	说明
40001	没有网络
40002	数据解析失败
40003	广告数据为空
40004	缓存视频资源失
100001	参数有误
100002	服务器错误
100003	不允许的操作
100004	服务不可用
310001	appId未注册
310002	appId无效
310003	appId已封禁
310004	packageName与注册的packageName不一致
310005	操作系统与注册的不一致
320002	appId对应账号无效
320003	appId对应账号已封禁
330001	posId未注册
330002	posId无效
330003	posId已封禁
330004	posid与注册的appId信息不一致

5. FQA

1. [MediaRemote] OutputDeviceUID is nil 扬声器: (null)

https://stackoverflow.com/questions/56243550/new-outputdeviceuid-is-nil-msg-when-instantiating-mpvolumeview

2. 请求视频广告失败：

   1.请核对广告SDK初始化的AppID是否正确；

   2.请核对请求广告时传入的广告场景参数posId是否正确。

   3.请根据返回的错误吗，参考错误码对照表知晓问题

3. SDK广告缓存清理机制

激励视频和全屏视频广告请求成功后，会对需要资源提前进行缓存，缓存资源超过10个的时候会自动清理一半(LRU)