

POKKT SDK Integration Guide (v 7.2.0)

iOS

Overview	2
Project Configuration	3
Dependencies	3
Frameworks Required	3
Info.plist	3
Other Settings	5
Implementation Steps	7
SDK Configuration	7
Ad Types	8
Video	8
Rewarded	8
Non Rewarded	8
OutStream	9
Non Rewarded	9
Interstitial	11
Rewarded	11
Non Rewarded	11
Banner	12
Ad Delegates	13
Video	13
OutStream	13
Interstitial	13
Banner	13
Pokkt ad player configuration	14
User Details	17
Debugging	18
Analytics	19
Google Analytics	19
Flurry Analytics	19

MixPanel Analytics	19
Fabric Analytics	19
IAP(In App Purchase)	20
Pokkt Dashboard	21
InApp Notifications	21
Create notifications	21

Overview

Thank you for choosing **Pokkt SDK for iOS**. This document contains all the information required to set up the SDK with your project. We also support mediation for various third party networks. To know the supported third party networks and their integration process go to mediation section.

Before implementing plugins it is mandatory to go through [project configuration](#) and [implementation steps](#), as these sections contain mandatory steps for basic SDK integration and are followed by every plugin.

You can download our SDK from pokkt.com.

In the package downloaded above you will find:

1. Docs:
Contains documentations for step wise step integration for SDK.
2. PokktAds Demo
Source code for *PokktAds Demo*(Sample app) which showcase implementation of Pokkt SDK through code for better understanding.
3. PokktSDK:
PokktSDK.framework add in to your project
4. POKKMoatMobileAppKit:
POKKMoatMobileAppKit.framework add in to your project

ScreenName: This one parameter is accepted by almost all API's of Pokkt SDK. This controls the placement of ads and can be created on Pokkt Dashboard.

We will be referencing **PokktAds Demo** app provided with SDK during the course of explanation in this document. We suggest you go through the sample app for better understanding.

Project Configuration

Dependencies

- Add *PokktSDK.framework* to your project's settings at "Build Phases -> Link Binary with Libraries and add the PokktSDK.framework
- Add *POKKMoatMobileAppKit.framework* to your project's settings at "Build Phases -> Link Binary with Libraries and add the POKKMoatMobileAppKit.framework.

Frameworks Required

```
CoreData.framework
Foundation.framework
MediaPlayer.framework
SystemConfiguration.framework
UIKit.framework
CoreTelephony.framework
EventKit.framework
AdSupport.framework
CoreGraphics.framework
CoreMotion.framework
MessageUI.framework
EventKitUI.framework
CoreLocation.framework
AVFoundation.framework
libc++.tbd
```

Info.plist

Add the below exceptions to your application's info.plist.

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

```

<key>NSExceptionDomains</key>
<dict>
  <key>pokkt.com</key>
  <dict>
    <key>NSIncludesSubdomains</key>
    <true/>
    <key>NSExceptionAllowsInsecureHTTPLoads</key>
    <true/>
    <key>NSExceptionRequiresForwardSecrecy</key>
    <false/>
    <key>NSExceptionMinimumTLSVersion</key>
    <string>TLSv1.2</string>
    <key>NSThirdPartyExceptionAllowsInsecureHTTPLoads</key>
    <false/>
    <key>NSThirdPartyExceptionRequiresForwardSecrecy</key>
    <true/>
    <key>NSThirdPartyExceptionMinimumTLSVersion</key>
    <string>TLSv1.2</string>
    <key>NSRequiresCertificateTransparency</key>
    <false/>
  </dict>
<key>cloudfront.net</key>
<dict>
  <key>NSIncludesSubdomains</key>
  <true/>
  <key>NSExceptionAllowsInsecureHTTPLoads</key>
  <true/>
  <key>NSExceptionRequiresForwardSecrecy</key>
  <false/>
  <key>NSExceptionMinimumTLSVersion</key>
  <string>TLSv1.2</string>
  <key>NSThirdPartyExceptionAllowsInsecureHTTPLoads</key>
  <false/>
  <key>NSThirdPartyExceptionRequiresForwardSecrecy</key>
  <true/>
  <key>NSThirdPartyExceptionMinimumTLSVersion</key>
  <string>TLSv1.2</string>
  <key>NSRequiresCertificateTransparency</key>

```

```
<false/>
</dict>
</dict>
</dict>
```

Other Settings

- Please make sure that your app project has **-ObjC** set as Other linker flag in Build Settings.
- Need to enable background fetch mode in Xcode at
Project Header -> Targets -> Capabilities -> Background Modes -> Enable Background fetch.
Now add below mentioned script at *DidFinishLaunchingWithOptions* **AppDelegate** method

```
[application
setMinimumBackgroundFetchInterval:UIApplicationBackgroundFetchIntervalMinimum];
```

- Need to implement the background fetch delegate methods in **AppDelegate** class

```
-(void)application:(UIApplication *)application
performFetchWithCompletionHandler:(void (^)
(UIBackgroundFetchResult))completionHandler
```

- Import the PokktAds Class in **AppDelegate**. Then You will have to call the

```
- (void)application:(UIApplication *)application
performFetchWithCompletionHandler:(void (^)
(UIBackgroundFetchResult))completionHandler
{
    [PokktAds callBackgroundTaskCompletionHandler:^(UIBackgroundFetchResult
    result)
    {
        completionHandler(result);
    }];
}
```

- Enable the local notification for *InApp Notifications* in **AppDelegate** class.
Add below mentioned script at *DidFinishLaunchingWithOptions* delegate method **AppDelegate** class

```

    UIApplicationSettings *settings = [UIApplicationSettings
    settingsForTypes:
        (UIRemoteNotificationTypeBadge|

    UIRemoteNotificationTypeSound|UIRemoteNotificationTypeAlert)
    categories:nil];

    [application
    registerUserNotificationSettings:settings];

```

- Implement *LocalNotification* delegate method in **AppDelegate** Class

```

- (void)application:(UIApplication *)application
    didReceiveLocalNotification:

    (UILocalNotification *)notification

```

- You will have to call the *inAppNotificationEvent* Method, When user tap on local notification.

```

- (void)application:(UIApplication *)application
    didReceiveLocalNotification:(UILocalNotification *) notification
    {
        [PokktAds inAppNotificationEvent:notification];
    }

```

- You will have to call the *notifyAppInstall* method, When application launch first.

```

+ (void) notifyAppInstall

```

Implementation Steps

SDK Configuration

1. Set **Application Id** and **Security key** in Pokkt SDK. You can get it from Pokkt dashboard from your account. We generally assign unique application Id and Security key.

```
PokktAds.setPokktConfigWithAppId:(NSString*) appId securityKey:(NSString*) securityKey
```

2. If you are using server to server integration with Pokkt, you can also set **Third Party UserId** in PokktAds.

```
PokktAds.setThirdPartyUserId:(NSString*) userId
```

3. When your application is under development and if you want to see Pokkt logs and other informatory messages, you can enable it by setting **ShouldDebug** to **true**. Make sure to disable debugging before release.

```
[PokktDebugger setDebug: YES/ NO];
```

4. Set **GDPR consent** in Pokkt SDK. **This must be called before calling any ad related API.** **Developers/Publishers must get the consent from user.** For more information on GDPR please refer <https://www.eugdpr.org/> and <https://www.eugdpr.org/gdpr-faqs.html>. This API can again be used by publishers to revoke the consent. If this API is not called or invalid data provided then SDK will access the users personal data for ad targeting.

```
PokktConsentInfo *consentInfo = [[PokktConsentInfo alloc] init];  
consentInfo.isGDPRApplicable = true;  
consentInfo.isGDPRConsentAvailable = false;  
[PokktAds setPokktConsentInfo:consentInfo];
```

5. You can set extra parameters to POKKT SDK, to be passed back to your server via POKKT server callback. These Extra parameters will be in key-value pair. The key must be alphanumeric value. See the below example.

```
NSMutableDictionary *dict = [[NSMutableDictionary alloc] initWithCapacity:0];
[dict setValue:@"value1" forKey:@"key1"];
[dict setValue:@"value2" forKey:@"key2"];
[dict setValue:@"value3" forKey:@"key3"];
[PokktAds setCallbackExtraParam:dict];
```

Ad Types

Video

- Video ad can be rewarded or non-rewarded. You can either cache the ad in advance or directly call show for it.
- We suggest you to cache the ad in advance so as to give seamless play behaviour, In other case it will stream the video which may lead to unnecessary buffering delays depending on the network connection.

Rewarded

1. To cache rewarded ad call:

```
[PokktVideoAds cacheRewardedVideoAd:(NSString*) screenName
```

2. To show rewarded ad call:

```
[PokktVideoAds showRewardedVideoAd:(NSString*) screenName
viewController:(UIViewController *)viewController];
```

You can check if ad is available or not before making *show* request.

```
[PokktVideoAds isAdCached:(NSString *)screenName isRewarded:(BOOL)isRewarded];
```

Non Rewarded

1. To cache non-rewarded ad call:

```
[PokktVideoAds cacheNonRewardedVideoAd: (NSString*) screenName];
```

2. To show non-rewarded ad call:

```
[PokktVideoAds showNonRewardedVideoAd:(NSString*) screenName  
viewController:(UIViewController *)viewController];
```

You can check if ad is available or not before making *show* request.

```
[PokktVideoAds isAdCached:(NSString *)screenName isRewarded:(BOOL)isRewarded];
```

OutStream

- OutStream ad will be non-rewarded only. OutStream ad allow only for following format:
- Ad will play only if ad is visible in bound, if it is not visible on the screen it will in pause state.
- OutStream Ad contain 3 states as Play, Pause & Replay. After completing the video Replay button will be enable.

Non Rewarded

1. ScrollView

```
+(void)loadOutstreamAd:(NSString*) screenName placeholder:(UIView *)container
scrollView:(UIScrollView *)scrollView
```

2. WebView

```
+(void)loadOutstreamAd:(NSString*) screenName inWebView:(UIWebView *)webView
```

3. TableView

OutStream Ad will display in tableview on provided indexPath cell .

```
+(void)loadOutstreamAd:(NSString*) screenName atIndexPath:(NSIndexPath
*)indexPath tableView:(UITableView *)tableView
```

4. It's compulsory to destroy OutStream Ad on exit once you come out from ViewController or If you are requesting for full screen ad .Once viewController will be disappear you have to notify by following method:

Note: If You are requesting for full screen ad then you have to destroy OutStream ad.

- ScrollView

```
+(void)disappearScrollViewOutStreamAd:(NSString *)screenName
inContainer:(UIView *)container
```

- WebView

```
+(void)disappearWebViewOutStreamAd:(NSString *)screenName
inContainer:(UIWebView *)webView
```

- TableView

```
+(void)disappearTableViewOutStreamAd:(NSString *)screenName  
inContainer:(UITableView *)tableView
```

Interstitial

Rewarded

1. To cache rewarded ad call:

```
[PokktInterstitial cacheRewardedInterstitial: (NSString*) screenName];
```

2. To show rewarded ad call:

```
[PokktInterstitial showRewardedInterstitial(NSString*) screenName  
viewController:(UIViewController *)viewController];
```

You can check if ad is available or not before making *show* request.

```
[PokktInterstitial isAdCached:(NSString *)screenName  
isRewarded:(BOOL)isRewarded];
```

Non Rewarded

1. To cache non-rewarded ad call:

```
[PokktInterstitial checkNonRewardedInterstitialAvailability: (NSString*)
screenName
```

2. To show non-rewarded ad call:

```
[PokktInterstitial showNonRewardedInterstitial(NSString*) screenName
viewController:(UIViewController *)viewController];
```

You can check if ad is available or not before making *show* request.

```
[PokktInterstitial isAdCached:(NSString *)screenName
isRewarded:(BOOL)isRewarded];
```

Banner

1. Load banner using following :-

```
PokktBannerView *banner = [PokktBanner initWithBannerAdSize:CGSizeMake(0, 0,
[UIScreen mainScreen].bounds.size.width, 50)];
[self.view addSubview:banner];

[PokktBanner loadBanner:banner withScreenName:@"default"
rootViewController:self];
```

2. Refresh banner using: -

You can set banner refresh rate on pokkt dashboard. Refresh rate should be in range of 30 -100.

3. Destroy banner using: -

```
[PokktBanner destroyBanner:<PokktBannerView>];
```

It's compulsory to destroy banner on exit once you come out from ViewController

Ad Delegates

Ad actions are optional, but we suggest to implement them as it will help you to keep track of the status of your ad request.

Video

```
[PokktVideoAdsDelegate setPokktVideoAdsDelegate:self];
```

OutStream

```
[PokktOutStremAds setPokktOutStreamAdsDelegate:self];
```

Interstitial

```
[PokktInterstitial setPokktInterstitialDelegate:self];
```

Banner

```
[PokktBanner setPokktBannerDelegate:self];
```

Pokkt ad player configuration

Pokkt Ad player works the way App is configured at Pokkt dashboard, but we provide a way to override those settings using **PokktAdPlayerViewConfig**.

Application should prefer configuration provided through code by developer or what's configured for the app in dashboard, can be controlled any time through the dashboard itself. If you want to make changes to this configuration after your app distribution, you can contact **Pokkt Team** to do the same for your app through admin console.

```
PokktAdPlayerViewConfig * adPlayerViewConfig = [[PokktAdPlayerViewConfig alloc] init];  
// set properties values to adPlayerViewConfig  
PokktAds.setAdPlayerViewConfig(adPlayerViewConfig );
```

Various properties that can be managed through this are:

1. Default skip time

Defines the time after which user can skip the Ad.

Property name: DefaultSkipTime

Values:

Any Integer value.

Default value is 10 seconds .

2. Should allow skip

Defines if user is allowed to skip the Ad or not.

Property name: ShouldAllowSkip

Values:

True = User can skip Ad.

False = User can't skip Ad.

3. Should allow mute

Defines if user is allowed to mute the Video Ad or not.

Property name: ShouldAllowMute

Values:

True = User can mute video Ad.
False = User can't mute video Ad.

4. Should confirm skip

Defines if confirmation dialog is to be shown before skipping the Ad.

Property name: ShouldConfirmSkip

Values:

True = Confirmation dialog will be shown before skipping the video.
False = Confirmation dialog will not be shown before skipping the video.

5. Skip confirmation message

Defines what confirmation message to be shown in skip dialog.

Property name: SkipConfirmMessage

Values:

Any String message.
Default value is "Skipping this video will earn you NO rewards. Are you sure?".

6. Affirmative label for skip dialog

Defines what should be the label for affirmative button in skip dialog.

Property name: SkipConfirmYesLabel

Values:

Any String message.
Default value is "Yes".

7. Negative label for skip dialog

Defines what should be the label for affirmative button in skip dialog.

Property name: SkipConfirmNoLabel

Values:

Any String message.
Default value is "No".

8. Skip timer message

Defines message to be shown before enabling skip button. Don't forget to add placeholder "##" in your custom message.

This placeholder is replaced by property "Default skip time" assigned above.

Property name: SkipTimerMessage

Values:

Any String message.

Default value is "You can skip video in ## seconds"

9. Incentive message

Defines message to be shown during video progress, that after what time user will be incentivised.

Property name: IncentiveMessage

Values:

Any String message

Default value is "more seconds only for your reward !"

10. AudioEnabled

AudioEnabled set YES or NO to mute ad,if 'YES' ad will be mute. , default is NO.

Property name: isAudioEnabled

Values:

True = If you want to mute Ad.

False = If you don't want to mute Ad.

11. Should collect feedback

Defines message to be shown during video progress, that after what time user will be incentivised.

Property name: setShouldCollectFeedback

Values:

True = If you want to collect feedback from the user for the Ad.

False = If you don't want to collect feedback from the user for the Ad.

User Details

For better targeting of ads you can also provide user details to our SDK using.

```
PokktUserDetails *pokktUserDetails = [PokktUserDetails alloc] init];
pokktUserDetails.Name = "";
pokktUserDetails.Age = "";
pokktUserDetails.Sex = "";
pokktUserDetails.MobileNo = "";
pokktUserDetails.EmailAddress = "";
pokktUserDetails.Location = "";
pokktUserDetails.Birthday = "";
pokktUserDetails.MaritalStatus = "";
pokktUserDetails.FacebookId = "";
pokktUserDetails.TwitterHandle = "";
pokktUserDetails.Education = "";
pokktUserDetails.Nationality = "";
pokktUserDetails.Employment = "";
pokktUserDetails.MaturityRating = "";

[PokktAds setPokktUserDetails: pokktUserDetails]
```

Debugging

Other than enabling debugging for Pokkt SDK, it can also be used to:

Export log

Export your log to your desired location, we generally have it in root directory of SD card, if permission for external storage is provided and in cache folder otherwise.

```
[PokktDebugger exportLog: (UIViewController *)controller
```

Analytics

We support various analytics in Pokkt SDK.

Below is mentioned how to enable various analytics with Pokkt SDK.

Google Analytics

Google analytics Id can be obtained from Google dashboard.

```
PokktAnalyticsDetails *analyticsDetail = [[PokktAnalyticsDetails alloc] init];  
analyticsDetail.eventType = GOOGLE_ANALYTICS;  
analyticsDetail.googleTrackerID = @"xyz";  
[PokktAds setPokktAnalyticsDetail:analyticsDetail];
```

Flurry Analytics

Flurry application key can be obtained from Flurry dashboard.

```
PokktAnalyticsDetails *analyticsDetail = [[PokktAnalyticsDetails alloc] init];  
analyticsDetail.eventType = FLURRY_ANALYTICS;  
analyticsDetail.flurryTrackerID = @"xyz";  
[PokktAds setPokktAnalyticsDetail:analyticsDetail];
```

MixPanel Analytics

MixPanel project token can be obtained from MixPanel dashboard.

```
PokktAnalyticsDetails *analyticsDetail = [[PokktAnalyticsDetails alloc] init];  
analyticsDetail.eventType = MIXPANNEL_ANALYTICS;  
analyticsDetail.mixPanelTrackerID = @"xyz";  
[PokktAds setPokktAnalyticsDetail:analyticsDetail];
```

Fabric Analytics

Analytics Id is not required in case of Fabric.

```
PokktAnalyticsDetails *analyticsDetail = [[PokktAnalyticsDetails alloc] init];
analyticsDetail.eventType = FABRIC_ANALYTICS;
analyticsDetail.fabricTrackerID = @"xyz";
[PokktAds setPokktAnalyticsDetail:analyticsDetail];
```

IAP(In App Purchase)

Call trackIAP to send any In App purchase information to Pokkt.

```
InAppPurchaseDetails * inAppPurchaseDetails = [ InAppPurchaseDetails alloc]init];
inAppPurchaseDetails.ProductId = "<productId>";
inAppPurchaseDetails.PurchaseData = "<purchaseData>";
inAppPurchaseDetails.PurchaseSignature = "<purchaseSignature>";
inAppPurchaseDetails.PurchaseStore = IAPStoreType.GOOGLE;
inAppPurchaseDetails.Price = <100.00>;

[PokktAds trackIAP: inAppPurchaseDetails]
```

Pokkt Dashboard

InApp Notifications

Open developer dashboard -> Manage App -> Notifications

Create notifications

1. Basic notification information required.

Add Notification

Basic

Name

App

videodemo

Platform

☐ iOS
 ☐ Android
 ☒ All

Filters

Countries

App Version

Last Seen

- a. Name:

A friendly name for notification. It will help you to distinguish between different notifications.
- b. App:

Select your app for which you want to assign notifications.
- c. Platform:

Select OS platform for which you want to target notifications.
- d. Countries:

Select countries where this notifications will be shown to users. Let's say if you have users in multiple countries, you can selectively target notifications to them.
- e. App version:

Enter your app version for which you want to show notifications. Let's say you have multiple version installed among users and you want to send different notifications to different users based on their versions.

f. Last seen:

Set minimum and maximum limit in days for which user can remain away from the app.

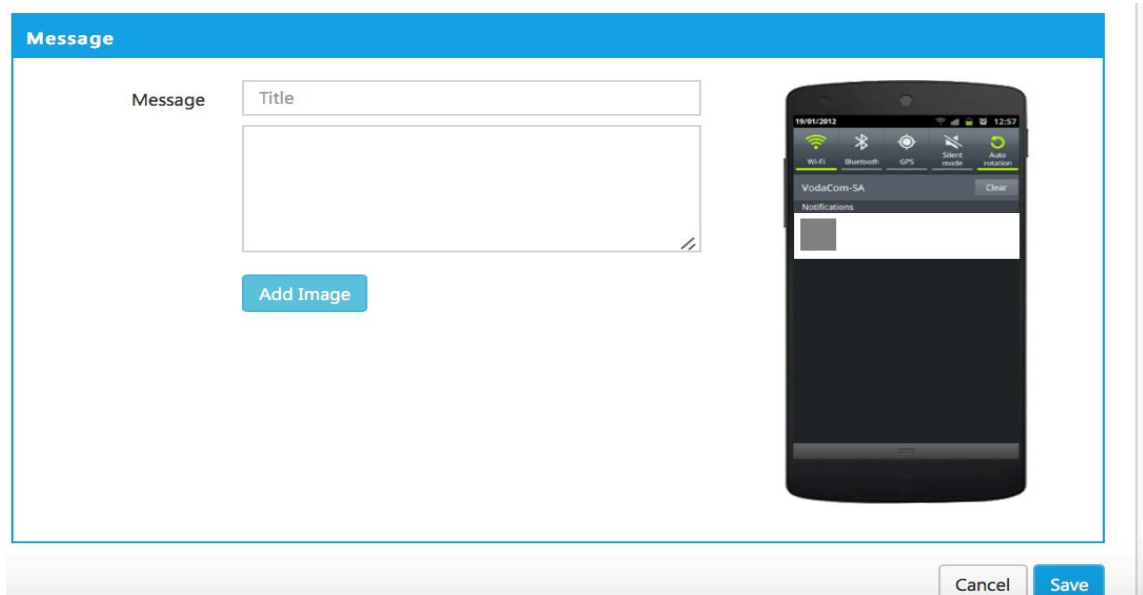
Let's say if min = 2 and max = 4, and user hasn't open your app for atleast 2 days, you can remind by showing notification, but if 4 days have passed app will not show any more notifications to user.

g. Message:

Message you want to show in notifications bar.

h. Title:

Title for the notification to be shown in notification bar.



2. Schedule notification

a. Daily

Schedule

Repeat
Daily

Every
1
Day(s)

Time
12
O'clock

b. Weekly

Schedule

Repeat
Weekly

Every
1
Week(s)

Mon	Tue	Wed	Thu	Fri	Sat
Sun					

Time
12
O'clock

c. Monthly

Repeat

Every
Month(s)

Dates

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31					

Time
O'clock