

## **POKKT SDK v4.0.0 Integration Guide for Xamarin (iOS)**

---

## Contents:

1. Overview
2. Configuration steps
3. Implementation steps
4. Functionalities
  - I.Video
5. Debugging and Logging
6. Important Points

v4.0.0

## 1. Overview

Thank you for choosing Pokkt SDK Plugin v4.0 for Xamarin. Pokkt SDK supports Video-Ad campaigns feature. This document contains all the information that is needed by you to setup the SDK with your project. The current plugin supports mediation for various third party ad-networks. These are:

- AdColony
- AppLovon
- Chartboost
- Fyber
- SuperSonic
- UnityAds
- Vungle
- Tapjoy

A separate set of documents is provided for each of these, explaining the implementation process.

There is a sample app provided with the SDK. We will be referencing this app during the course of explanation in this document. It is suggested that you should check that app to understand the following process in detail.

## 2. Configuration Steps

All we need is the file provided: *PluginExtension.zip*. This zip file contains two file one is *PokktXamarinExtension.dll* file which you need to add it in project and 2nd is *PokktSDKResource.bundle*.

**Steps need to follow:**

### Step 1:

1. Add the *PokktXamarinExtension.dll* to your project's Reference directory.
2. Extract the contents of *PokktSDKResource.bundle* folder and copy them in resource folders in your project.

### Step 2:

In order to use PokktSDK's background fetch functionality, enable "*Info.plist -> Background Modes ->Background Fetch*". Then write the following code-snippet inside "*FinishedLaunching*" method of *AppDelegate.cs* class.

```
application.SetMinimumBackgroundFetchInterval(UIApplication.BackgroundFetchIntervalMinimum);
```

### Step 3:

In order to enable local notifications for InApp Notifications, mention the following inside "*didFinishLaunchingWithOptions*" method of the app-delegate class:

```
UIUserNotificationSettings settings = UIUserNotificationSettings.GetSettingsForTypes
((UIUserNotificationType.Badge | UIUserNotificationType.Alert |
UIUserNotificationType.Sound), null);
application.RegisterUserNotificationSettings(settings);
```

### Step 4.

Further, implement/update the background-fetch delegate methods in *AppDelegate.cs* class. Invoke "*CallBackgroundTaskCompletionHandler*" method from "*PerformFetch*". Observe the following code-snippet for reference:

```
public override void PerformFetch(UIApplication application, Action<UIBackgroundFetchResult> completionHandler)
{

PokktXamarinExtension.PokktManager.CallBackgroundTaskCompletionHandler(completionHandler);
}
```

v4.0.0

**Step 5:**

Invoke “*InAppNotificationEvent*” if the user taps on local notification, do this in the “*ReceivedLocalNotification*” inside *AppDelegate.cs* class. Check the following reference:

```
public override void ReceivedLocalNotification(UIApplication application, UILocalNotification notification) {
    PokktXamarinExtension.PokktManager.InAppNotificationEvent(notification);
}
```

### 3. Implementation Steps

- Common

1. For all invocation of Pokkt SDK developer will make use of methods available in *PokktManager* class. This class only have static methods.
2. In *PokktManager* you can set *SetApplicationId*, *SetSecurityKey*, *SetIntegrationType* and *SetAutoCacheVideo*. which are must for all type of integrations.
3. Before calling any other methods from the *PokktManager* please make sure that you have called the *InitPokkt* already (This does not apply to session related methods namely *StartSession* and *EndSession*).
4. If you are doing server to server integration with pokkt you can also set *SetThirdPartyUserId* in *PokktManager*.
5. Apart from above mentioned parameters you can assign additional ones based on your integration type.(please refer to OfferWall and Video sections below).
6. While in development please call *PokktManager.SetDebug(true)*; to see pokkt debug logs and toast messages. please make sure to change this to *PokktManager.SetDebug(false)*; for production build.
7. Please call *PokktManagerHandler.SendAppInfo()* to send your application installation information to Pokkt.
8. To use google analytics, please set AnalyticsType and Analytics ID in

```
PokktManagerHandler.SetSelectedAnalyticsType
(AnalyticsName.GOOGLE_ANALYTICS)
    SetGoogleAnalyticsID("ID").
```

v4.0.0

9. To use flurry analytics please set AnalyticsType and Flurry Application Key in *PokktManagerHandler*.

```
SetSelectedAnalyticsType(AnalyticsName.FLURRY)
SetFlurryApplicationKey("key").
```

10. To use mix panel analytics please set AnalyticsType and Mix PanelProject Token in *PokktManagerHandler*

```
setSelectedAnalyticsType(AnalyticsType.MIXPANEL)
setMixPanelProjectToken().
```

- Session

1. We have option to start session for tracking for which we have *StartSession* and *EndSession* methods in *PokktManager*.
2. You should call *StartSession* at the start of his application and once only. You will need to call this method after setting required details like *SetApplicationId*, *SetSecurityKey* and *SetIntegrationType*.
3. You should call *EndSession* at the end of his application and once only.

- Video

1. In *PokktManager* for Video you can set five additional parameters which are *SetAutoCacheVideo*, *SetSkipEnabled*, *SetDefaultSkipTime*, *SetScreenName* and *SetIncentivised*.
2. *SetAutoCacheVideo* is required if you want to automatically cache video on user device. It has default value as true. if you set it as false then video will not be automatically cached and you will have to call *PokktManager.CacheVideoCampaign()*; to start caching videos on device.
3. If you want to enable/disable the skip button on video screen please set *SetSkipEnabled* as true/false. The default value for *SetSkipEnabled* is false.
4. If you have enabled skipped button by setting *SetSkipEnabled* as true then you can control after how many seconds the skip button will be visible in video by setting *SetDefaultSkipTime* to appropriate value. Since most videos will be 30 sec or less please set *SetDefaultSkipTime* as 10 or less. You can also give your own skip message by setting *SetCustomSkipMessage* on *PokktManager*.
5. *SetScreenName* has default value as *default* and can be used by you to give different screen name for different places in your app where you are showing video ads. You will control ad targeting based on these screen names which should match exactly with screen names defined in

dashboard. *SetScreenName* can not contain white spaces and only special characters allowed are hyphen and underscore.

6. You can choose to show video with or without incentive to user by setting *SetIncentivised* as true or false. Video gratification will only happen for incentivised playback.
7. You will need to listen video related events which you can see *PokktXamarinExtension* and also check the sample App in details.
8. You can call *PokktManager.IsVideoAvailable()* to check if the campaign are available before you try to play video.
9. You can call *PokktManager.GetVideo()*; to play video.
10. You will get different callbacks event for video playback from *PokktXamarinExtension* class but for that you need to register the event. Plesse check the sampe App for this.
12. Please reward user only from the *VideoGratifiedEvent* method gets triggered.
13. In *PokktManagerHandler*, you can also set the video caching retry duration when all the networks caching fails. Video caching will start again after this duration. Please call *PokktManagerHandler.SetRetryDuration*. Pass the time value in string.  
*PokktManager. RetryDuration("50"); // 50 is in seconds;*

## Export Logs

1. Developer should call *PokktManagerHandler.ExportLog()* to export the Pokkt SDK logs to folder of your choice.
2. This API shows a folder chooser dialog where user can choose a particular folder.
3. User can also create a new folder where user wants to export the logs

## Optional Parameters

- *PokktManager* also has provision for developers to provide extra user data available with them to pokkt. We currently support following data points: *SetName, SetAge, SetSex, SetMobileNo, SetEmailAddress, SetLocation, SetBirthday, SetMaritalStatus, SetFacebookId, SetTwitterHandle, SetEducation, SetNationality, SetEmployment and SetMaturityRating.*

## 4. Functionalities:

### 4.2 Video:

There are seven events which needs to register for Video campaign.

1. OnDownloadCompleted.
2. OnDownloadFailed.
3. OnVideoClosed.
4. OnVideoCompleted.
5. OnVideoDisplayed.
6. OnVideoGratified.
7. OnVideoSkipped.

These are the events which needs to register and needs to write handle events. Here is the process how to do that:

```
PokktXamarinExtension.PokktManager.DownloadCompletedEvent += downloadCompletedEvent;
PokktXamarinExtension.PokktManager.DownloadFailedEvent += downloadFailedEvent;
PokktXamarinExtension.PokktManager.VideoClosedEvent += videoClosedEvent;
PokktXamarinExtension.PokktManager.VideoCompletedEvent += videoCompletedEvent;
PokktXamarinExtension.PokktManager.VideoDisplayedEvent += videoDisplayedEvent;
PokktXamarinExtension.PokktManager.VideoSkippedEvent += videoSkippedEvent;
PokktXamarinExtension.PokktManager.VideoGratifiedEvent += videoGratifiedEvent;
```

```
private void downloadFailedEvent()    {    }

private void videoClosedEvent()      {    }

private void videoDisplayedEvent()    {    }

private void videoSkippedEvent()      {    }

private void videoCompletedEvent()    {    }

private void videoGratifiedEvent(string earnedPoints) {    }
```

A video file is cached on user's device. You can set the auto-caching option in the beginning, as mentioned earlier in this document. In case of manual caching, call the following to start video caching:

```
PokktManager.CacheVideoCampaign();
```

Before playing video or showing button to play video, Application should check whether video is cached or not by calling the following:

```
PokktManager.IsVideoAvailable();
```

**DownloadCompleted** event will get triggered once video gets cached locally on device.

Furthermore, Application can decide to play video as incentivised (user will be gratified after watching complete video) or non-incentivised (user will not be gratified after watching complete video). You must provide the



v4.0.0

screen-name parameter for it and set the incentive as true or false.  
Followings are the method calls to me made:

```
PokktManager.SetScreenName("screenname");  
PokktManager.SetIncentivised(<true/false>);  
PokktManager.GetVideo();
```

Next, [VideoGratifiedEvent](#) method will get called once you earn the points after completing the video.

v4.0.0

## 5. Debugging and Logging

You can enable the SDK logs by setting the debugging option to true anytime.

*PokktManager.SetDebug(<true/false>);*

## 6. Important Points

- Please do not copy the code points from this pdf as it may introduce unwanted characters and space in your code. Instead please refer to sample app source code in pokkt bundle.
- Please also refer to sample app source code for better understanding of implementation.