

Saturday, 2 April 2016

## POKKT SDK v4.1 Integration Guide for Xamarin (Android)

---

## Contents:

1. Overview
2. Configuration steps
3. Implementation steps
4. Important Points

## 1. Overview

Thank you for choosing Pokkt SDK Plugin v4.1 for Xamarin. Pokkt SDK supports Offerwall as well as Video-Ad campaigns feature. This document contains all the information that is needed by you to setup the SDK with your project. Please follow these steps as per your integration requirement (Video/Offerwall/Both). The current plugin supports mediation for various third party ad-networks. These are:

- AdColony
- AppLovon
- Chartboost
- Fyber
- InMobi
- SuperSonic
- TapJoy
- Vungle
- Facebook
- AdMob

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.

**Note:** Please do not copy the code points from this PDF file as it may introduce unwanted characters and space in your code. Instead please refer to sample app source code provided with the sample app.

## 2. Configuration Steps

All we need is the file provided: *PluginExtension.zip*. This zip file contains three files one is *PokktExtension.dll*, *PokktExtension.Droid.dll* which you need to add it in project reference and 3rd is *res.zip*.

1. Add the *PokktExtension.dll* and *PokktExtension.Droid.dll* to your project's Reference directory.
2. Copy the contents of *res* folder to respective folders in your project.
3. Add following permissions in your AndroidManifest, If not already there.

```
<!-- These permissions are mandatory to run Pokkt SDK -->

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.READ_PHONE_STATE" />

<!-- These permissions are strongly recommended and will result in higher performance -->

<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

<uses-permission android:name="android.permission.WAKE_LOCK" />

<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

<!-- This permission is optional but will improve SDK feature-->

<uses-permission android:name="android.permission.WRITE_CALENDAR" />
```

4. Add the following activity in your AndroidManifest for OfferWall Integration

```
<activity

    android:name="com.app.pokktsdk.ShowOfferwallActivity"

    android:configChanges="keyboard|keyboardHidden|navigation|orientation|screenLayout|uiMode|screenSize"

    android:label="@string/app_name"

    android:windowSoftInputMode="adjustPan" >

</activity>
```

5. Add the following activity in your AndroidManifest for Video Integration

```
<activity

    android:name="com.app.pokktsdk.PlayVideoCampaignActivity"

    android:configChanges="keyboard|keyboardHidden|navigation|orientation|screenLayout|uiMode|screenSize"

    android:label="@string/app_name"

    android:screenOrientation="landscape"

    android:windowSoftInputMode="stateAlwaysHidden|adjustUnspecified" >

</activity>
```

## 6. Add Following Broadcast receiver for OfferWall Integration in AndroidManifest

```
<receiver android:name="com.app.pokktsdk.ApplInstallBroadcastReceiver" >

    <intent-filter android:priority="1000" >

        <action android:name="android.intent.action.PACKAGE_INSTALL" />

        <action android:name="android.intent.action.PACKAGE_ADDED" />

        <data android:scheme="package" />

    </intent-filter>

</receiver>
```

## 7. Add the following activity in your AndroidManifest for Pokkt Interstitial Integration

```
<activity

    android:name="com.app.pokktsdk.InterstitialActivity"

    android:configChanges="keyboard|keyboardHidden|navigation|orientation|screenLayout|uiMode|screenSize|
    smallestScreenSize"

    android:label="@string/app_name"

    android:windowSoftInputMode="stateAlwaysHidden|adjustUnspecified" >

</activity>
```

## 8. Add Following meta tag for google play services (Goole play services is required and should be part of your project, if not, please add the GooglePlay-Service component in project.)

```
<meta-data android:name="com.google.android.gms.version" android:value="@integer/google_play_services_version" />
```

## 9. Add a meta data tag for offerwall campaign. you will have to implement the *IOfferwallDelegate* interface in your project to listen for all offerwall related events. (refer to pokkt sample for example).

```
<meta-data android:name="offerwallDelegate" android:value="@com.pokkt.plugin.common.PokktOfferwallDelegate" />
```

## 10. Add following Service and receiver in manifest for google analytics (Optional).

```
<receiver android:name="com.google.android.gms.analytics.AnalyticsReceiver" android:enabled="true">

    <intent-filter>

        <action android:name="com.google.android.gms.analytics.ANALYTICS_DISPATCH" />

    </intent-filter>

</receiver>
```

```
<service android:name="com.google.android.gms.analytics.AnalyticsService"
```

```
android:enabled="true"  
android:exported="false"/>
```

## 11. Add following Service in manifest for in-app notification.

```
<service android:name="com.app.pokktsdk.notification.NotificationSer-  
vice" android:label="PokktNotificationService"  
android:exported="false"/>
```

### 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. You need to set extension before calling any method like below. This is mandatory to do it.  
 PokktExtension.*PokktManager*.SetNativeExtentions(new *AndroidExtension*(this));
3. In *PokktConfig* you can set *ApplicationId*, *SecurityKey*, *IntegrationType*. which are must for all type of integrations. Please check the sample app.
4. Before calling any other methods from the *PokktManager*, please make sure that you have called the *InitPokkt* with passing *PokktConfig* object.
5. If you are doing server to server integration with pokkt you can also set *ThirdPartyUserId* in *PokktConfig*.
6. Apart from above mentioned parameters you can assign additional ones based on your integration type (please refer to OfferWall and Reward sections below).
7. 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.
8. Android MinSDKVersion should be >= 14.
9. To use google analytics, please set AnalyticsType and Analytics ID in *PokktConfig*.

```
SelectedAnalyticsType = AnalyticsType.GOOGLE_ANALYTICS
GoogleAnalyticsId = "Id".
```

10. To use flurry analytics please set AnalyticsType and Flurry Application Key in *PokktConfig*.

```
SelectedAnalyticsType = AnalyticsType.FLURRY
FlurryApplicationKey = "key".
```

11. To use mix panel analytics please set AnalyticsType and Mix PanelProject Token in *PokktConfig*

```
SelectedAnalyticsType = AnalyticsType.MIXPANEL
MixPanelProjectToken = "token".
```

12. To use mix panel analytics please set AnalyticsType and Fabric Token in *PokktConfig*

```
SelectedAnalyticsType = AnalyticsType.FABRIC
```

13. Please call *PokktManager.NotifyAppInstall()* to send your application installation information to Pokkt.

14. Please call *PokktManager.TrackIAP(InAppPurchaseDetail)* to send any in-app purchase information to Pokkt. Like below you can do this.

```
InAppPurchaseDetails purchaseDetails = new InAppPurchaseDetails ();
purchaseDetails.CurrencyCode = "IN";
purchaseDetails.Description = "description";
purchaseDetails.Price = 10;
purchaseDetails.ProductId = "product id";
purchaseDetails.PurchaseData = "date";
purchaseDetails.PurchaseSignature = "signature";
purchaseDetails.PurchaseStore = IAPStoreType.AMAZON;
PokktManager.TrackIAP (purchaseDetails);
```

## Session

1. We have option to start session for tracking: *StartSession* and *EndSession* methods in *PokktManager*.
2. You should call *StartSession* at the start of his application if you want to use this but this is the optional and call it after setting application id and security key.
3. You should call *EndSession* at the end of his application.

## OfferWall

1. In *PokktConfig* for OfferWall you can set two additional parameters which are *OfferWallAssetValue* and *CloseOnSuccessFlag*. *OfferWallAssetValue* is only required if you only want to show campaign of certain value on offerwall. *CloseOnSuccessFlag* is required if you wish to close the OfferWall after user has completed one offer. It's default value is false.

2. Before calling another method for offerWall in *PokktManager*, please make sure that you have already called *InitPokkt* first.

3. You need to add event listener as below or also please check the OfferwallActivity class in sample app.

```
PokktManager.Dispatcher.PokktInitialisedEvent += PokktInitialisedEvent;
PokktManager.Dispatcher.CampaignAvailabilityEvent += CampaignAvailabilityEvent;
PokktManager.Dispatcher.CoinResponseEvent += CoinResponseEvent;
PokktManager.Dispatcher.CoinResponseFailedEvent += CoinResponseFailedEvent;
PokktManager.Dispatcher.CoinResponseWithTransIdEvent += CoinResponseWithTransIdEvent;
PokktManager.Dispatcher.OfferwallClosedEvent += OfferwallClosedEvent;
```

4. To show OfferWall you can call [\*PokktManager.GetCoins\(pokktConfig\)\*](#).
5. In the screen or activity where you have button to show offer wall, in that activity onResume you should call [\*PokktManager.GetPendingCoins\(pokktConfig\)\*](#); so that you get a callback to award points to the user after he has come back to your game after finishing with OfferWall. You will get a callback for this where you add the listener.
6. You can call [\*PokktManager.CheckOfferWallCampaign\(pokktConfig\)\*](#); to check whether the campaigns are available before showing OfferWall button to user. You will get a callback for this call.

## AdConfig

1. In [\*AdConfig\*](#) you should set [\*ScreenName\*](#) and [\*IsRewarded\*](#). This screen name will be created on pokkt dashboard.
2. In [\*AdConfig\*](#) , developer can also set [\*ShouldAllowSkip\*](#), [\*DefaultSkipTime\*](#), [\*SkipConfirmMessage\*](#), [\*BackButtonDisabled\*](#), [\*ShouldAllowMute\*](#), [\*ShouldSkipConfirm\*](#), [\*SkipConfirmYesLabel\*](#), [\*SkipConfirmNoLabel\*](#), [\*SkipTimerMessage\*](#) and [\*IncentiveMessage\*](#) . These values can be used to configure the behaviour of ad.
3. If you want to enable/disable the skip button on video screen please set [\*ShouldAllowSkip\*](#) as true/false. The default value for [\*ShouldAllowSkip\*](#) is true.
4. If you have enabled skipped button by setting [\*ShouldAllowSkip\*](#) as true then you can control after how many seconds the skip button will be visible in video by setting [\*DefaultSkipTime\*](#) to appropriate value. Since most videos will be 30 sec or less please set [\*DefaultSkipTime\*](#) as 10 or less. You can also give your own skip message by setting [\*SkipConfirmMessage\*](#) on [\*AdConfig\*](#)
5. The [\*screenName\*](#) 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 ads. You will control ad targeting based on these screen names which should match exactly with screen names defined in dashboard. ScreenName can not contain white spaces and only special characters allowed are hyphen and underscore.
6. You can choose to show ad with or without incentive to user by setting [\*IsRewarded\*](#) as true or false. Ad gratification will only happen for incentivised playback.
7. You can disable the back button while video is playing by setting [\*BackButtonDisabled\*](#) on [\*AdConfig\*](#).
8. You can configure the ad skip dialog yes/no labels by setting [\*SkipConfirmYesLabel\*](#) and [\*SkipConfirmNoLabel\*](#).

9. You can configure the ad incentive message by setting *IncentiveMessage*.
10. You can configure the ad skip timer message by setting *SkipTimerMessage*. The message must contain a ## placeholder to show skip time value, which will keep changing as per the time.

### Rewarded Ad/Non-Rewarded Ad

1. You need to set true/false for rewarded or non-rewarded ad in `adConfig.IsRewarded = true/false`;
2. You will have to call *PokktManager.CacheAd(adConfig)*; to start caching ads on device.
3. You will need to register event for getting callback for Ad related callback like below and also please check given sample app VideoActivity.cs class.

```
PokktManager.Dispatcher.PokktInitialisedEvent += PokktInitialised;  
PokktManager.Dispatcher.AdCachingCompletedEvent += AdCachingCompleted;  
PokktManager.Dispatcher.AdCachingFailedEvent += AdCachingFailed;  
PokktManager.Dispatcher.AdAvailabilityEvent += AdAvailability;  
PokktManager.Dispatcher.AdDisplayedEvent += AdDisplayed;  
PokktManager.Dispatcher.AdCompletedEvent += AdCompleted;  
PokktManager.Dispatcher.AdClosedEvent += AdClosed;  
PokktManager.Dispatcher.AdSkippedEvent += AdSkipped;  
PokktManager.Dispatcher.AdGratifiedEvent += AdGratified;
```

4. You can call *PokktManager.CheckAdAvailability(adConfig)* to check if the campaign are available for a particular adConfig before you try to show ad.
5. You can call *PokktManager.ShowAd(adConfig)*; to show ad.
6. You will get different callbacks as given in *AdCampaignDelegate* implementation class for ad display.
7. Please reward user only from the *AdGratified* method.

### Mediation Info

1. Pokkt SDK now supports 10 ad networks which you can integrate in your application for better monetisation.
2. To integrate these networks through Pokkt, please visit the mediation menu on downloads page and download *xamarin mediation zip* and *documentation zip* files.
3. Please follow the mediation integration documents shipped for each network.

Saturday, 2 April 2016

4. You will need to create account on these networks and add the network details in your Pokkt dashboard after login into your account on pokkt website.
5. You will also need to do the mapping of Pokkt screens with the corresponding ad networks' placement id/zone id/ad unit etc in the dashboard.

### Export Logs

1. Developer should call [\*PokktManager.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

[\*PokktConfig\*](#) 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\*](#).

## In-App Notifications

Developer can add In-App notifications in their dashboard.

Add Notification

Basic

Name

App

Platform ☐ iOS ☐ Android ☒ All

Filters

Countries

App Version

Last Seen

Schedule

Repeat

Repeat

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

Message

Message

Title

Add Image

Cancel Save

Repeat schedule can be daily, weekly monthly.

Daily Repeat can have options like frequency of repeat and time in hours of notification.

The screenshot shows two forms. The top form, titled 'Schedule', has a 'Repeat' dropdown set to 'Daily'. Below it, 'Every' is set to '1' with a unit of 'Day(s)'. The 'Time' is set to '12' O'clock. The bottom form, titled 'Message', has a 'Title' field and a larger text area. An 'Add Image' button is below the text area. To the right of the text area is a preview of a smartphone screen showing a notification for 'VivoCom SA' with the title 'Notifications'. At the bottom right of the forms are 'Cancel' and 'Save' buttons.

Weekly repeat can have options like frequency of repeat in weeks, days of repeat and time in hours of notification.

The screenshot shows two forms. The top form, titled 'Schedule', has a 'Repeat' dropdown set to 'Weekly'. Below it, 'Every' is set to '1' with a unit of 'Week(s)'. There is a row of buttons for days of the week: 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', and 'Sun'. The 'Time' is set to '12' O'clock. The bottom form, titled 'Message', is identical to the one in the previous screenshot, with a 'Title' field, a text area, an 'Add Image' button, and a smartphone preview showing a notification for 'VivoCom SA'. At the bottom right of the forms are 'Cancel' and 'Save' buttons.

Saturday, 2 April 2016

Monthly repeat can have options like frequency of repeat in months dates of repeat and time in hours for notification .

The screenshot shows a configuration window for a notification. At the top, the 'Repeat' dropdown is set to 'Monthly'. Below it, the frequency is set to 'Every 1 Month(s)'. A 'Dates' section contains a calendar grid with dates from 1 to 31. The 'Time' is set to '12 O'clock'. Below the calendar is a 'Message' section with a 'Title' input field, a large text area, and an 'Add Image' button. To the right of the text area is a preview of a smartphone displaying the notification. At the bottom right are 'Cancel' and 'Save' buttons.

For don't repeat case, there are options like dates and time in hour for notification.

The notifications are listed and can be edited. Notifications can also be deactivated/activated.

#### List Notifications

Id	Name	App Id	Header	Status	Action
1	push	1000125	Hi There	ACTIVE	<a href="#">Edit</a> <a href="#">Deactivate</a>
2	in app	1000125	Hi There	ACTIVE	<a href="#">Edit</a> <a href="#">Deactivate</a>

## 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.