POKKT SDK v4.1 Integration Guide for Unity3d 5.2+ (Android-AAR)

Contents:

1. Overview:

Thank you for choosing Pokkt SDK Plugin v4.1 for Unity3d. Pokkt SDK supports Offerwall as well as Video/Interstitial-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 (Ad/Offerwall/Both). The current plugin supports mediation for various third party ad-networks. These are:

- AdColony
- AppLovin
- Chartboost
- Fyber
- InMobi
- SuperSonic
- UnityAds
- TapJoy
- Vungle
- AdMob
- Facebook
- MoPub

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

Kindly note that these instructions are for Unity3d Version 5.2 and above, older versions of Unity3d are not supported.

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:

Export the content of the provided unity-package file: ***PAP_AAR_WithDemo.unitypackage*** to your Unity app/game project. The contents of "Plugin/Pokkt" and the "Plugins/Android" directories are mandatory. You can safely ignore all the other stuff, they are provided here to demonstrate the integration of Pokkt-SDK, this document uses snippets from those files.

Minimum android SDK version is 14.

Include Google Play Services SDK

Please follow the instructions below to include the Google Play Services SDK:
http://developer.android.com/google/play-services/setup.html

Why is this required?
Google has now changed policy w.r.t recognizing the devices. It no longer allows the developer to read the Android_ID. Instead a new Advertisers ID is needed to be use. Please find more details below:
https://developer.android.com/google/play-services/id.html

Include "android support v4" library

In order to support Pokkt In-App Notifications, you will need to add "android support v4" library to your project.

Android-Manifest Changes

Permissions:

```
<!--REQUIRED PERMISSIONS FOR POKKT (already added)-->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"
/>

<!--RECOMMENDED PERMISSIONS FOR POKKT (already added)-->
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WAKE_LOCK" />

<!--OPTIONAL PERMISSIONS FOR POKKT (you will have to add it)-->
<uses-permission android:name="android.permission.WRITE_CALENDAR" />
```

Activities:

Added for Offerwall:

```
<activity
android:name="com.app.pokktsdk.ShowOfferwallActivity"
android:configChanges="keyboard|keyboardHidden|navigation|
orientation|screenLayout|uiMode|screenSize"
android:windowSoftInputMode="adjustPan" />
```

Added for Video-ads:

```
<activity
android:name="com.app.pokktsdk.PlayVideoCampaignActivity"
android:configChanges="keyboard|keyboardHidden|navigation|
orientation|screenLayout|uiMode|screenSize"
android:screenOrientation="landscape"
android:windowSoftInputMode="stateAlwaysHidden|adjustUnspecified" />
```

Added for Interstitials:

```
<activity
android:name="com.app.pokktsdk.InterstitialActivity"
android:configChanges="keyboard|keyboardHidden|navigation|
orientation|screenLayout|uiMode|screenSize|smallestScreenSize"/>
```

Broadcast Receivers:

Added for Offerwall:

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

Optional for Google Analytics:

```
<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>
```

Meta-Data tags:

Added for Offerwall:

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

Required for Google Play Services: Add Following meta tag for google play services (Goole play services is required and should be part of your project, if not, please refer http://developer.android.com/google/play-services/setup.html)

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

Services:

optional for In-App Notifications:

```
<service
    android:name="com.google.android.gms.analytics.AnalyticsService"
    android:enabled="true"
    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. Setup your **PokktConfig,** provide **ApplicationId** and **SecurityKey,** these are must for initializing Pokkt.

3. Once you have you **PokktConfig** ready, invoke **InitPokkt** method before you invoke any other methods from the **PokktManager.** This does not apply to session related methods namely **StartSession** and **EndSession** and few utility methods.

4. If you are doing server to server integration with Pokkt you can also set **ThirdPartyUserId** in **PokktConfig**.

5. Apart from above mentioned parameters you can assign additional ones based on your integration type. Refer to **OfferWall** and Video sections below.

6. While in development, you can call **PokktManager.SetDebug(true)** to see Pokkt debug logs and toast messages. please make sure to change this to **false** for production build.

7. Call **PokktManager.NotifyAppInstall()** to log your application installation information with Pokkt.

8. Call **PokktManager.TrackIAP(details)** to log any in-app purchase details with Pokkt.

9. To use google analytics, please set **SelectedAnalyticsType** and **GoogleAnalyticsID** in **PokktConfig**.

10. To use flurry analytics please set **SelectedAnalyticsType** and **FlurryApplicationKey** in **PokktConfig**.

11. To use mix panel analytics please set **SelectedAnalyticsType** and **MixPanelProjectToken** in **PokktConfig**.

Session

1. Invoke **PokktManager.StartSession()** at the start of his application and once only.

2. You should call **PokktManager.EndSession()** at the end of his application and once only.

Offerwall

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

2. Make sure to call **InitPokkt** before calling another method for Offerwall in **PokktManager**.

3. To show Offerwall you can call **PokktManager.GetCoins()**.

4. Maintain a method **OnApplicationPause** in one of your **MonoBehavior**, you should call **PokktManager.GetPendingCoins()** here so that you get a callback to award points to the user after he has come back to your game/app after finishing with Offerwall. You will get a callback for this call in your **IOfferwallDelegate** implementation class in method **EarnedCoins** or **CoinResponseFailed**.

5. You can call **PokktManager.CheckCampaignAvailable()** to check whether the campaigns are available before showing Offerwall button to user. You will get a callback for this call in your **IOfferwallDelegate** implementation class in method **OnOfferwallCampaignCheck**.

6. Following are offerwall-related events, you can refer to the provided sample-code to understand the ideal implementation on how to consume these:

    - CoinResponseEvent
    - CoinResposnseWithTransIdEvent
    - CoinResponseFailedEvent
    - CampaignAvailabilityEvent
    - OfferwallClosedEvent

Rewarded/Non-Rewarded Ads

1. You will have to configure an **AdConfig** object to request for any ad to be displayed. It also provides options for customizing your ad-screen. It is recommended to have different **AdConfig** objects for each screens. Followings are the values that you can set with **AdConfig**:
    - **ScreenName** (Required): This controls the placement of ads and can be created on Pokkt Dashboard.

    - **IsRewarded** (Required): Requested ad type. Ad gratification will happen only for rewarded ads.

    - **BackButtonDisabled**: Disable 'back' button press while on ad-screen.

- **DefaultSkipTime**: If ad-skipping is allowed, this provides the seconds it will wait before the skip button appears.

- **ShouldAllowSkip**: Whether skipping-ad is allowed or not. If set to 'false', user will be forced to watch the ad till it finishes.

- **ShouldAllowMute**: Whether to allow sound-mute while on ad-screen, it controls the 'mute' button. Cannot contains whitespaces and only special characters allowed are hyphens (-) and underscores (_).

- **ShouldConfirmSkip**: Controls whether to show the skip-confirmation dialog box. If set to 'false', the ad will be silently closed without prompting for confirmation

- **SkipConfirmMessage**: The message that will appear on skip-confirmation dialog box.

- **SkipConfirmYesLabel**: 'Yes' Label of skip-confirmation dialog box.

- **SkipConfirmNoLabel**: 'No' Label of skip-confirmation dialog box.

- **SkipTimerMessage**: The message on countdown-timer before the skip button appears. The message must contain a '##'-placeholder to show timer value.

- **IncentiveMessage**: If set, the message will be displayed while prompting user to watch the ad for certain time before it can be rewarded.

2. Invoke **PokktManager.CacheAd(adConfig)** to cache the ad on device. Cached-ads provide better user experience than streaming-ads

3. You can call **PokktManager.CheckAdAvailability(adConfig)** to check if the ad-campaigns are available or not. The result will be notified via **AdAvailabilityEvent** event.

4. Invoke **PokktManager.ShowAd(adConfig)** to show ad.

5. Following are ad-related events, you can refer to the provided sample-code to understand the ideal implementation on how to consume these:

    - AdCachingCompletedEvent
    - AdCachingFailedEvent
    - AdDisplayedEvent
    - AdSkippedEvent
    - AdCompletedEvent

- AdClosedEvent
- AdGratifiedEvent
- AdAvailabilityEvent

6. Reward user ONLY from the **AdGratifiedEvent**.

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: **Name**, **Age**, **Sex**, **MobileNo**, **EmailAddress**, **Location**, **Birthday**, **MaritalStatus**, **FacebookId**, **TwitterHandle**, **Education**, **Nationality**, **Employment** and **MaturityRating**.

This concludes the integration documentation. It is highly suggested that you should check the sample app that is provided to you to understand it better.