
POKKT SDK v5.0 Integration Guide for Cordova/PhoneGap (Android)

Contents:

1. Overview
2. Configuration Steps
 - a. Google Play Services
 - b. Android-Support v4
 - c. Manifest Changes
3. Implementation Steps
 - a. Common
 - b. Session
 - c. Offerwall
 - d. Rewarded/Non-Rewarded Ads
 - e. Export Logs
 - f. Optional Parameters

1. Overview:

Thank you for choosing Pokkt SDK Plugin v5.0 for Cordova/PhoneGap. Pokkt SDK supports Offerwall as well as Video/Interstitial-Ad/Banner 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

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

Kindly note that these instructions are for Cordova Version 4.x and above, older versions of 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:

Extract the provided file “PokktCordovaPlugin.zip” into a directory. Execute the following command from your terminal:

\$phonegap plugin add /<path-to-plugin-directory>/PokktCordovaPlugin/

This should install the plugin with your project and you should be able to use it inside your project.

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 and MRAID ad extended features, 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" />
```

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

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|smallestScreenSize"
  android:label="@string/app_name"
  android:screenOrientation="landscape"
  android:windowSoftInputMode="stateAlwaysHidden|adjustUnspecified" />
```

```
<activity
  android:name="com.app.pokktsdk.VPAIDActivity"
  android:configChanges="keyboard|keyboardHidden|navigation|orientation|
  screenLayout|uiMode|screenSize|smallestScreenSize"
  android:label="@string/app_name"
  android:windowSoftInputMode="stateAlwaysHidden|adjustUnspecified" />
```

Added for Interstitials:

```
<activity
  android:name="com.app.pokktsdk.PokktInterstitialActivity"
  android:configChanges="keyboard|keyboardHidden|navigation| orientation|
  screenLayout|uiMode|screenSize|smallestScreenSize"
  android:label="@string/app_name"
  android:windowSoftInputMode="stateAlwaysHidden|adjustUnspecified" />
```

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>
```

```
    </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 **pokktNativeExtension.js** file using **PokktExtension** object.
2. Create a **PokktConfig** object using **creatPokktConfig()** method, then provide **applicationId** and **securityKey**, these are must for initializing Pokkt.
3. Once you have your **pokktConfig** ready, invoke **initPokkt** method before you invoke any other methods from the **PokktExtension**. This does not apply to session related methods namely **startSession** and **endSession** and few utility methods.
4. In order to know whether Pokkt is initialized or not, listen to '**PokktInitialised**' event, if the provided Boolean parameter is true only then move ahead with other operations.
5. If you are doing server to server integration with Pokkt you can also set **thirdPartyUserId** in **pokktConfig** object.
6. Apart from above mentioned parameters you can assign additional ones based on your integration type. Refer to Offerwall and Video sections below.
7. While in development please call **PokktExtension.setDebug(true)** to see Pokkt debug logs and toast messages. Make sure to change this to **false** for production build.
8. Call **PokktExtension.notifyAppInstall()** to log your application installation information with Pokkt.
9. Call **PokktExtension.trackIAP(details)** to log any in-app purchase details with Pokkt. Accepted values are (all caps): "NONE", "GOOGLE", "IOS", "AMAZON".
10. To use google analytics, please set **selectedAnalyticsType** to "GOOGLE_ANALYTICS" and **googleAnalyticsID** in **pokktConfig**.
11. To use flurry analytics please set **selectedAnalyticsType** to "FLURRY" and **flurryApplicationKey** in **pokktConfig**.
12. To use mix panel analytics please set **selectedAnalyticsType** to "MIXPANEL" and **mixPanelProjectToken** in **pokktConfig**.
13. To use mix panel analytics please set **selectedAnalyticsType** to "FABRIC".

Session

1. Invoke **PokktExtension.startSession()** at the start of his application and once only.
2. You should call **PokktExtension.endSession()** at the end of his application and once only.

Offerwall

1. In **pokktConfig** for Offerwall you can set two additional parameters which are **setOfferWallAssetValue** and **setCloseOnSuccessFlag**. Setting of **setOfferWallAssetValue** is only required if you only want to show campaign of certain value on the Offerwall. **setCloseOnSuccessFlag** 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 **PokktExtension**.
3. To show Offerwall you can call **PokktExtension.getCoins(pokktConfig)**.
4. In the screen or activity where you have button to show Offerwall, in that activity **onResume** you should call **PokktExtension.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 call in your **IOfferwallDelegate** implementation class in method **EarnedCoins** or **CoinResponseFailed**.
5. You can call **PokktExtension.checkCampaignAvailable(pokktConfig)** to check whether the campaigns are available before showing Offerwall button to user.
6. Following are offerwall-related events, you can refer to the provided sample-code to understand the ideal implementation on how to consume these:
 - 'CoinResponse'
 - 'CoinResponseWithTransId'
 - 'CoinResponseFailed'
 - 'CampaignAvailability'
 - 'OfferwallClosed'

Rewarded/Non-Rewarded Ads

1. You will have to configure an **adConfig** object to request for any ad to be displayed. Get one using **createAdConfig()** method. 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.
 - **adFormat** (Required): Requested ad format. SDK supports Video, Interstitial and banner adFormats. Default is Video ad format.
 - **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 **PokktExtension.cacheAd(adConfig)** to cache the ad on device. Cached-ads provide better user experience than streaming-ads
 3. You can call **PokktExtension.checkAdAvailability(adConfig)** to check if the ad-campaigns are available or not. The result will be notified via '**AdAvailability**'.
 4. Invoke **PokktExtension.showAd(adConfig)** to show ad.

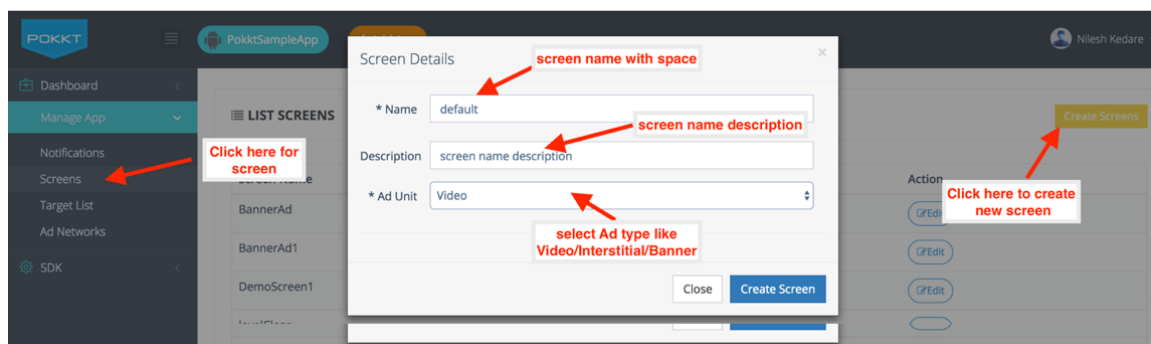
- Following are ad-related events, you can refer to the provided sample-code to understand the ideal implementation on how to consume these:

- 'AdCachingCompleted'
- 'AdCachingFailed'
- 'AdClosed'
- 'AdCompleted'
- 'AdDisplayed'
- 'AdSkipped'
- 'AdGratified'
- 'AdAvailability'

- Reward user ONLY from the '**AdGratified**' event.

Banner Ads

- You will have to call **PokktManager.loadBanner(screenName, position)** to show banner on device.
- Screen name** : Screen name has to be created in Pokkt dashboard. Please check below screen. This will help you to understand how to create new screen name in Pokkt dashboard.



- Position** : Position can be a value from **BannerPosition**. **BannerPosition** has values: **TopLeft**, **TopCenter**, **TopRight**, **MiddleLeft**, **MiddleRight**, **BottomLeft**, **BottomCenter** and **BottomRight**. Please choose any of these value depending on your requirement.
- Invoke **PokktManager.loadBannerWithRect(screenName, height, width, x, y)** to customize your banner size/position.
- Invoke **PokktManager.bannerAutoRefresh(bool refresh)** to start/stop automatic refresh of banner.
- Invoke **PokktManager.removeBanner()** to remove banner.

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.