
POKKT SDK v2.1.4 Integration Guide for Cordova/PhoneGap (Android)

Contents:

1. Introduction
2. Installation
3. AndroidManifest.xml Setup
4. SDK Setup on Cordova/PhoneGap
5. Functionalities:
 1. Offerwall
 2. Video
6. Debugging and Logging

1. Introduction:

Thank you for choosing Pokkt SDK for Cordova/PhoneGap. This document contains all the information that is needed by you to setup the SDK with your project. Kindly note that these instructions are for PhoneGap Version 4.x and above.

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. Installation:

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.

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.

3. AndroidManifest.xml Setup:

AndroidManifest.xml is pre-setup and should there is nothing else you are required to do yourself. Just make sure that in your config.xml, "android-minSdkVersion" is set to 14 or above as mentioned below:

```
<preference name="android-minSdkVersion" value="14" />
```

Overview of the items added to the manifest:

Added required permission:

Common:

```
READ_PHONE_STATE
INTERNET
ACCESS_NETWORK_STATE
```

Video specific:

```
WRITE_EXTERNAL_STORAGE
WAKE_LOCK
```

Added Activity definitions:

Offerwall:

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

Video:

```
<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="adjustPan" />
```

Added BroadcastReceiver (required only 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>
```

Added meta-data information

```
<meta-data
    android:name="offerwallDelegate"
    android:value="com.pokkt.cordova.OfferwallEventsHandler" />
<meta-data
    android:name="videoDelegate"
    android:value="com.pokkt.cordova.VideoEventsHandler" />
```

Step 6: 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>

4. SDK Setup on Cordova/PhoneGap:

Initialize PokktManager:

You start with setting up the following values.

- Security Key
- Application Id
- Integration Type
- Auto Cache Video

You set these values as the first most step. See the following for reference:

```
var pe = window.plugins.pokktExtension;  
  
// set required config params  
pe.setSecurityKey('<security_key>');  
pe.setApplicationId('application_id');  
pe.setIntegrationType(0); // 1: Offerwall 2: Video 0: Both
```

After setting these values, make sure to set the auto-caching option after you set the params (this is mandatory). Ref.:

```
pe.setAutoCaching(true/false);
```

Now you are ready to initialize the SDK, you do that by calling the following method:

```
pe.initPokkt();
```

5. Functionalities:

5.1 Offerwall

There are five events to listen too. These are:

- CoinResponse
- CoinResponseWithTransId
- CoinResponseFailed
- CampaignAvailability
- OfferwallClosed

(Ideally)Add handlers to these events in the Awake() method of your MonoBehaviour class.
Below are the references on how to use them:

Reference on how to consume them:

```
document.addEventListener('CoinResponse',
    this.onCoinResponse, false);

document.addEventListener('CoinResponseWithTransId',
    this.onCoinResponseWithTransId, false);

document.addEventListener('CoinResponseFailed',
    this.onCoinResponseFailed, false);

document.addEventListener('CampaignAvailability',
    this.onCampaignAvailability, false);

document.addEventListener('OfferwallClosed',
    this.onOfferwallClosed, false);

...

onCoinResponse: function(params) {
    console.log('onCoinResponse: ' + params.param);
    var labelPointsEarned =
        document.getElementById('labelPointsEarned');
    labelPointsEarned.innerHTML = params.param;
},

onCoinResponseWithTransId: function(params) {
    console.log('onCoinResponseWithTransId: ' + params.param);
    var labelPointsEarned =
        document.getElementById('labelPointsEarned');
    labelPointsEarned.innerHTML = params.param;
},

onCoinResponseFailed: function(params) {
    console.log('onCoinResponseFailed: ' + params.param);
},
```



```
onCampaignAvailability: function(params) {  
    console.log('onCampaignAvailability: ' + params.param);  
},  
  
onOfferwallClosed: function(params) {  
    console.log('onOfferwallClosed: ' + params.param);  
},
```

There are two ways to invoke offerwall.

Open Asset Value: In this case, POKKT platform provides all offers with any asset value.

Sample code snippet:

```
var pe = window.plugins.pokktExtension;  
pe.getCoins(0);
```

Fixed Asset Value: In this case, POKKT platform provides all offers with fixed asset value.

Sample code snippet:

```
var pe = window.plugins.pokktExtension;  
pe.getCoins(<asset_value>);
```

Pending Coins: In case after completing activity, if status of transaction is pending, then call getPendingCoins() method.

```
var pe = window.plugins.pokktExtension;  
pe.getPendingCoins();
```

Check for campaign availability: If you are using optional meta-tag as mentioned in Step 5, you can call the following to check for campaign availability:

```
var pe = window.plugins.pokktExtension;  
pe.checkOfferWallCampaign();
```

The result will be noticed with the event:

CampaignAvailability

Moreover, you can listen to the following event to know whether offerwall has been closed or not:

OfferwallClosed

5.2 Video:

There are 7 events to manage the video caching and its playback, these are:

- VideoClosedEvent
- VideoDisplayedEvent
- VideoSippedEvent
- VideoCompletedEvent
- VideoGratifiedEvent
- DownloadCompletedEvent
- DownloadFailedEvent

(Ideally)Add handlers to these events in the Awake() method of your MonoBehaviour class.
Below are the references on how to use them:

Reference on how to consume them:

```
document.addEventListener('DownloadCompleted',
    this.onDownloadCompleted, false);

document.addEventListener('DownloadFailed',
    this.onDownloadFailed, false);

document.addEventListener('VideoDisplayed',
    this.onVideoDisplayed, false);

document.addEventListener('VideoClosed',
    this.onVideoClosed, false);

document.addEventListener('VideoSkipped',
    this.onVideoSkipped, false);

document.addEventListener('VideoCompleted',
    this.onVideoCompleted, false);

document.addEventListener('VideoGratified',
    this.onVideoGratified, false);

onDownloadCompleted: function(params) {
    console.log('onDownloadCompleted: ' + params.param);

    // set button state
    videoController.toggleButtons(true);
},

onDownloadFailed: function(params) {
    console.log('onDownloadFailed: ' + params.param);

    // set button state
    videoController.toggleButtons(false);
},
```

```

onVideoDisplayed: function(params) {
    console.log('onVideoDisplayed: ' + params.param);

    // set button state
    videoController.toggleButtons(false);
},

onVideoClosed: function(params) {
    console.log('onVideoClosed: ' + params.param);

    var pe = window.plugins.pokktExtension;
    if (pe.getAutoCaching()) {

        // set button state
        videoController.toggleButtons(true);
    } else {
        var buttonStartCaching =
            document.getElementById('buttonStartCaching');
        buttonStartCaching.disabled = false;
    }
},

onVideoSkipped: function(params) {
    console.log('onVideoSkipped: ' + params.param);
},

onVideoCompleted: function(params) {
    console.log('onVideoCompleted: ' + params.param);
},

onVideoGratified: function(params) {
    console.log('onVideoGratified: ' + params.param);
    var labelPointsEarned =
        document.getElementById('labelPointsEarned');
    labelPointsEarned.innerHTML = params.param;
},

```

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:

```

var pe = window.plugins.pokktExtension;
pe.cacheVideoCampaign();

```

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

```

var pe = window.plugins.pokktExtension;
pe.checkIsVideoAvailable(function(isAvailable) {
    console.log('checkIsVideoAvailable result: ' + isAvailable);
});

```

You should listen to "DownloadCompleted" to check whether download is completed or not, you can show the play buttons once you receive this event.

Furthermore, Application can decide to play video as incent (user will be gratified after watching complete video) or non-incent (user will not be gratified after watching complete video). You must provide the screen-name parameter for it. Followings are the method calls to me made:

```
var pe = window.plugins.pokktExtension;  
if (incentivised)  
    pe.getVideo('sampleApp');  
else  
    pe.getVideoNonIncent('sampleApp');
```

Next, you can listen to VideoGratified to get the coins earned, if at all, by watching the last video.

6. Debugging and Logging

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

```
pe.setDebug(true);
```

You can use the following command to log some debug messages:

```
pe.showLog('sampleApp');
```

You can use the following command to display a message as Toast on android device:

```
pe.showToast('sampleApp');
```

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.