
POKKT SDK v2.1.3 Integration Guide **for Cordova/PhoneGap (iOS)**

Contents:

1. Introduction
2. Installation
3. SDK Setup on Cordova/PhoneGap
4. Functionalities: Video
5. 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. 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();
```

4. Functionalities:

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);
```

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.