



POKKT SDK v5.0 Integration Guide for Xamarin (iOS)

Contents:

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

1. Overview

Thank you for choosing **Pokkt SDK Plugin v5.0** for **Xamarin**. Pokkt SDK supports Reward Ad, Non-Reward Ad, Banner Ad and Offerwall campaigns feature. This document contains all the information which is needed to setup the SDK with project. Please follow these steps as per your integration requirement (Reward/Non-Reward/Banner/Offerwall). The current plugin supports mediation for various third party ad-networks. These are:

1. AdColony
2. AppLovin
3. Chartboost
4. Fyber
5. Supersonic
6. UnityAds
7. Tapjoy
8. Vungle
9. AdMob
10. Facebook

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.

2. Configuration steps

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

Steps need to follow:

Step 1:

1. Add the **PokktExtension.dll** and **PokktExtension.iOS.dll** to your project's Reference directory.
2. Extract the contents of **PokktSDKResource.bundle** folder and copy them in resource folders in your project.

Step 2:

In order to use PokktSDK's background fetch functionality, enable "Info.plist -> Background Modes -> Background Fetch". Then write the following code-snippet inside "FinishedLaunching" method of AppDelegate.cs class.

```
application.SetMinimumBackgroundFetchInterval(UIApplication.BackgroundFetchIntervalMinimum);
```

Step 3:

In order to enable local notifications for InApp Notifications, mention the following inside "didFinishLaunchingWithOptions" method of the app-delegate class:

```
UIUserNotificationSettings settings = UIUserNotificationSettings.GetSettingsForTypes((UIUserNotificationType.Badge | UIUserNotificationType.Alert | UIUserNotificationType.Sound), null);  
application.RegisterUserNotificationSettings(settings);
```

Step 4.

Further, implement/update the background-fetch delegate methods in AppDelegate.cs class. Invoke "CallBackgroundTaskCompletionHandler" method from "PerformFetch". Observe the following code-snippet for reference:

```
public override void PerformFetch(UIApplication application, Action<UIBackgroundFetchResult> completionHandler) {  
    PokktXamarinExtension.PokktManager.CallBackgroundTaskCompletionHandler(completionHandler);  
}
```

Step 5:

Invoke “InAppNotificationEvent” if the user taps on local notification, do this in the “ReceivedLocalNotification” inside AppDelegate.cs class. Check the following reference:

```
public override void ReceivedLocalNotification(UIApplication application, UILocalNotification notification) {  
    PokktXamarinExtension.PokktManager.InAppNotificationEvent(notification);  
}
```

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. 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.

```
PokktManager.SetNativeExtentions(new IosExtension(this));
```

3. In **PokktConfig** you can set ApplicationId and SecurityKey which are must for all type of integrations. Please check the sample app.

```
pokktConfig.ApplicationId = "<Pokkt Application ID>";  
pokktConfig.SecurityKey = "<Pokkt Security Key>";
```

4. Before calling any other methods from the PokktManager, please make sure that you have called the **InitPokkt** with passing PokktConfig object.

```
PokktManager.InitPokkt(pokktConfig);
```

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

```
pokktConfig.ThirdPartyUserId = "<Third party user Id>";
```

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 `setDebug` method to see Pokkt logs and toast messages. Please make sure to change this to `setDebug` to false for production build.

```
PokktManager.setDebug(true/false);
```

8. To use **Google Analytics**, please set `AnalyticsType` and `Analytics ID` in

```
pokktConfig.selectedAnalyticsType = AnalyticsType.GOOGLE_ANALYTICS;  
pokktConfig.googleAnalyticsID = "Id";
```

9. To use **Flurry Analytics** please set `AnalyticsType` and `Flurry Application Key` in `PokktConfig`.

```
pokktConfig.selectedAnalyticsType = AnalyticsType.FLURRY;  
pokktConfig.flurryApplicationKey = "key";
```

10. To use **Mix Panel Analytics** please set `AnalyticsType` and `Mix PanelProject Token` in `PokktConfig`:

```
pokktConfig.selectedAnalyticsType = AnalyticsType.FLURRY;  
pokktConfig.mixPanelProjectToken = "token";
```

11. To use **Fabric Analytics** please set `AnalyticsType` and `Fabric Token` in `PokktConfig`

```
pokktConfig.selectedAnalyticsType = AnalyticsType.FABRIC
```

12. Please call `trackIAP(InAppPurchaseDetail)` to send any in-app purchase information to Pokkt. Like below you can do this.

```
// create object of InAppPurchaseDetails and set properties for purchaseDetail  
s and pass object  
PokktManager.trackIAP (InAppPurchaseDetails);
```

Session

1. We have option to start session and end session for tracking in `PokktManager`.

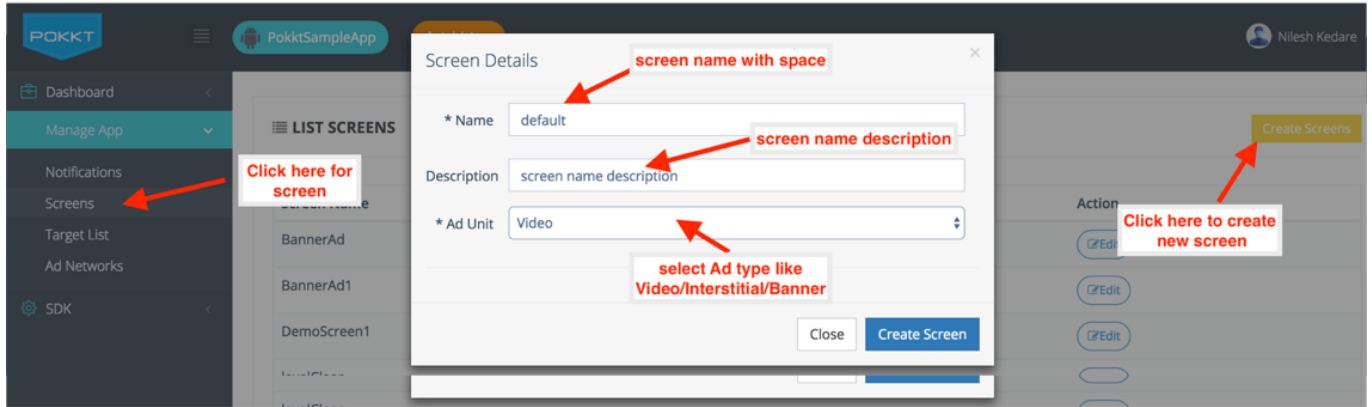
```
PokktManager.startSession();  
PokktManager.endSession();
```

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.

AdConfig (used for reward / non-reward ad)

1. In **AdConfig**, you need to provide **screenName**, **isRewarded** and **adFormat**. This screen name will be created on Pokkt dashboard.

Screen Name: For screen name, you will have to create on Pokkt dashboard. Please check below screen shot. This will help you to understand how to create new screen name on Pokkt dashboard.



IsRewarded: You need to provide ad type. Pass it "false" if you are looking for non-reward ad otherwise "true" for reward ad.

AdFormat: you can set VIDEO / BANNER / INTERSTITIAL like below:

```
adConfig->adFormat = VIDEO;  
adConfig->adFormat = BANNER;  
adConfig->adFormat = INTERSTITIAL;
```

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 "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.

Reward Ad / Non-Reward Ad

Rewarded ad and Non-Rewarded ad can be Video or Interstitial ad.

1. You need to set true/false for rewarded or non-rewarded ad like below:

```
adConfig.isRewarded = true/false;
```

2. 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 (rewarded/non-rewarded). Ad gratification will happen only for rewarded ads.

AdFormat (Required): Requested ad-format. SDK supports Video, Interstitial and Banner ad formats. Default is Video ad format.

BackButtonDisabled: Disable 'back' button press while on ad-screen. (Not available in iOS)

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

ShouldAllowSkip: An Ad is skippable or not. If set to 'false', user will be forced to watch the ad till an ad finishes.

ShouldAllowMute: Toggles "mute" button on ad-screen.

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.

3. You will have to call cache ad to start caching ads on device.

```
PokktManager.cacheAd(adConfig)
```

4. You will need to register event for getting callback for Ad related callback like below and also please check **VideoActivity.cs** class given **sample app**.

```
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;
```

5. You can call checkAdAvailability to check if the campaign are available for a particular adConfig before you try to show ad.

```
PokktManager.checkAdAvailability(adConfig)
```

6. You can call showAd method to show ad.

```
PokktManager.showAd(adConfig);
```

7. Please reward user only when **AdGratified** event gets triggered.

Banner Ad

Pokkt SDK allows to show banner ad on your screen. You can set any custom size or any position for banner. There are few fixed position already given in BannerPosition class. But you can customise that also.

1. **Load Banner:** Use loadBanner to show banner ad like below:

```
PokktManager.LoadBanner(<ScreenName>, (int)BannerPosition.TopCenter, <activity>);
```

There is predefined positions are already given which can be used but there is separate method if you want to customise banner ad.

Predefined position: TOP_LEFT, TOP_CENTER, TOP_RIGHT, MIDDLE_LEFT, MIDDLE_CENTER, MIDDLE_RIGHT, BOTTOM_LEFT, BOTTOM_CENTER, BOTTOM_RIGHT

2. **Remove Banner:** Call removeBanner method to remove banner ad from screen like below:

```
PokktManager.removeBanner(ScreenName);
```

3. **Auto Refresh Banner:** Use setBannerAutoRefresh method to disable or enable auto refresh. Default it is true and it will refresh automatically based on given time on Pokkt dashboard for particular screen name.

```
PokktManager.setBannerAutoRefresh(false/true);
```

4. **Banner Position:** Use BannerPosition class properties for banner position.

5. **Custom banner:** There is also option given to customise banner size and position by using below method:

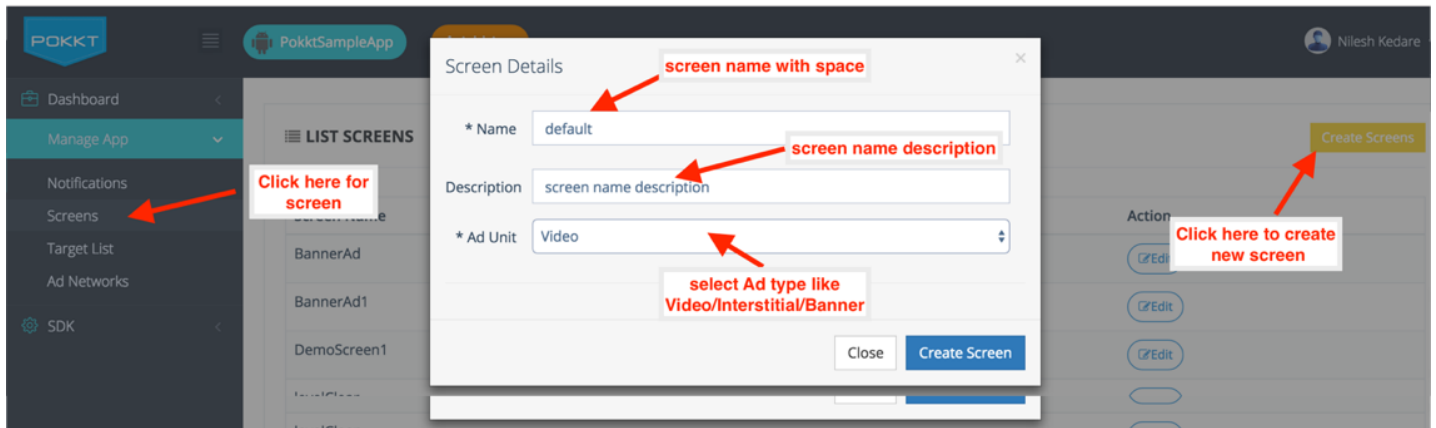
```
PokktManager.loadBannerWithRect(ScreenName, width, height, x, y);
```

6. **Banner Event:** Register event to get banner related callback like below:

```
//Register event
PokktManager.Dispatcher.BannerLoadedEvent += onBannerLoaded;
PokktManager.Dispatcher.BannerLoadFailedEvent += onBannerLoadFailed;

// Handler method
public void onBannerLoaded(String screenName) { }
public void onBannerLoadFailed(String screenName, String errorMessage) { }
```

7. **Screen Name:** For screen name, you will have to create in Pokkt dashboard. Please check below screen. This will help you to understand how to create new screen name in Pokkt dashboard.



Mediation Info

1. Pokkt SDK supports 10 ad networks which you can integrate in your application for better monetization.
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.
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' **PlacementID / ZoneID / AdUnit** etc in the dashboard.

Export Logs

1. Developer should call **exportLog** to export the Pokkt SDK logs to folder of your choice.

```
PokktManager.exportLog( )
```

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, setEducation, setNationality, setEmployment and setMaturityRating*.

4. 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.