AdFalcon iOS SDK 4.2.0 Developer's Guide

AdFalcon Mobile Ad Network

Product of

Noqoush Mobile Media Group

Table of Contents

1 Introduction

AdFalcon iOS SDK Integration guide contains all the information needed for iOS developers to integrate with AdFlacon Network. The guide will also show examples and code snippets on how to perform the integration.

Prerequisites

- Xcode: 8.0 or higher
- Development target: iOS 8.0 or higher
- CocoaPods to manage library dependencies with your Xcode projects (Recommended). for more information about CocoaPods visit cocoapods site

Note: *If you need to support iOS 7, you can download the old SDK via AdFalcon iOS SDK v3.5.0.*

2 Install AdFalcon SDK

There are two ways to install our SDK:

2.1 Use CocoaPods (Recommended)

To use the CocoaPods you should implement the steps below, For more info visit: https://cocoapods.org/pods/AdFalconSDK

Initialize the CocoaPods

The below steps illustrate the needed steps to initialize and prepare the CocoaPods.

- Open the terminal app from your applications
- If you haven't installed the CocoaPods before, run this command from the terminal: **sudo gem install cocoapods**
- Go to your project's directory by running this command: **cd [path of your project]**
- Create CocoaPods to your project by running this command: **pod init**
- Open your project's directory, you will find the CocoaPods project was installed and ready.

Modify the Podfile

After creating the CocoaPods project, you will find Podfile within the project's directory. Open it via TextEdit or Xcode, then insert the below pod (**pod 'AdFalconSDK'**) to your target

```
platform :ios, '8.0'

target 'YourProjectTarget' do
  pod 'AdFalconSDK'

end
```

Install the pod

Run the below command to install AdFalcon's pod to your project.

pod install

Update the pod

To be sure to get the latest iOS SDK into your project, run the below command from the terminal in the same path of Podfile' directory.

pod update

2.2 Add AdFalcon SDK framework manually

Follow the procedure below to manually add AdFalcon SDK framework to your project:

- Download AdFalcon iOS SDK.
- Unzip the downloaded zip file, you will find AdFalconSDK.framework in the Framework folder.
- Add the framework to your "Link Binary With Libraries" in "Build Phases" of your project's target.

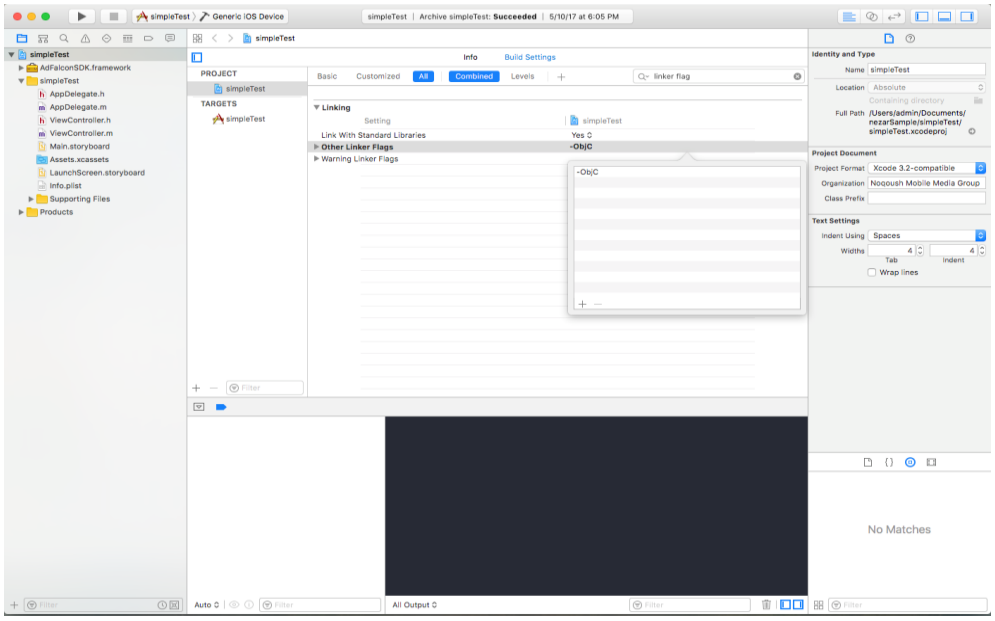
3 Configure Xcode

Add “-ObjC” to Other Linker Flags

You must add -ObjC linker flag to your project; if you forget adding the linker flag, the “Selector not recognized” exception will be raised during the runtime.

follow these steps to add -ObjC:

1. In Xcode, choose View > Navigators > Show Project Navigator, or press ⌘1.
2. Select your project under the PROJECT heading in the Project Navigator, then select the Build Settings tab.
3. Scroll down to the Other Linker Flags build setting under the Linking collection, or type "Other Linker Flags" into the search bar.
4. Set the value of the Other Linker Flags build setting to -ObjC.



Turn Off App Transport Security (ATS)

If you build your app with iOS SDK 9.0 or later, you will need to turn off App Transport Security in your app’s plist in order to complete AdFalcon SDK integration successfully as below:

```
<key>NSAppTransportSecurity</key>
<dict>
        <key>NSAllowsArbitraryLoads</key>
         <true/>
        <key>NSAllowsArbitraryLoadsForMedia</key>
         <true/>
        <key>NSAllowsArbitraryLoadsInWebContent</key>
        <true/>
</dict>
```

4 Banner Ad Integration Steps

This section describes how to integrate with the AdFalcon's iOS SDK in order to consume the services available by the network.

In order to integrate with the SDK, please perform the following steps:

Add ADFAdView to your ViewController

1. Add ADFAdView to the view controller header file
    1. Import **AdFalconSDK** Module.
    2. declare **ADFAdView instance.**

Below is an example of how the header should look like:

```
@import "AdFalconSDK";

@interface XViewController: UIViewController{
    ADFAdView * adView;
}
```

2. Add the following code snippet to viewDidLoad method

```
//initialize ADFalcon view and set size of the view base on ad unit size
adView = [[ADFAdView alloc] initWithFrame:CGRectMake(0, 0, 320, 50)];

//Enable or disable testing mode
adView.testing = YES;

//initialize AdFalcon view and loading ads
[adView

  initializeWithAdUnit:kADFAdViewAdUnitSizeAutoBanner //set the ad unit size

            siteId:@"Your Site ID"

            params:nil //Optional set user information

            rootViewController:self //rootViewController

            enableAutorefresh:YES //Enable or disable auto refresh

            delegate:nil //Optional

];

//Add AdFalcon view
[self.view addSubview:adView];
```

Note: *In case you are overriding the [UIViewController loadView] to programmatically draw your view, ensure that AdFalcon view is initialized after completing loading the UIViewController's view.*

Note: *Ensure Test Mode Parameter is set to false before uploading your application to Apple store.*

***Note**: to know more about ad unit size go to [ADFAdViewAdUnitSize]*

Ads Options

Tracking Ad Lifecycle Events – ADFADViewDelegate
AdFalcon iOS SDK provides ability to track Ad lifecyle events, follow the procedure below to listen to the Ad lifecycle events

- Import the ADFADViewDelegate.h protocol to your header as the below:

```
@import "AdFalconSDK";

@interface XViewController: UIViewController<ADFAdViewDelegate> {
}
```

- Add the below ADFADViewDelegate's methods to your source file as the below

```
-(void)adViewWillLoadAd:(ADFAdView*) adView
{
}
-(void)adViewDidLoadAd: (ADFAdView*) adView
{
}
-(void)adView: (ADFAdView*) adView didFailWithCode:(int) code message:(NSString*) message
{
    switch (code) {
        case kADFAdViewErrorNoAdAvailabe:
            // No Ad Availabe
            break;
        case kADFAdViewErrorInvalidParam:
            // Invalid Param send to server
            break;
        case kADFAdViewErrorMissingParam:
            // Missing Param send to server
            break;
        case kADFAdViewErrorCommunication:
            //Communication with server failed or no internet
            break;
        default:
            break;
    }
}
-(void)adViewWillPresentScreen: (ADFAdView*) adView
{
}
-(void)adViewDidPresentScreen: (ADFAdView*) adView
{
}
-(void)adViewWillDismissScreen: (ADFAdView*) adView
{
}
-(void)adViewDidDismissScreen: (ADFAdView*) adView
{
}
-(void)applicationWillTerminate:(UIApplication *)application
{
}
-(void)applicationWillEnterBackground:(UIApplication *)application
{
}
```

- Assign your delegate object to ADFAdView object as the below:

```
[adView initializeWithAdUnit:kADFAdViewAdUnitSizeAutoBanner //set the ad unit size
        siteId:@"Your Site ID"
        params:nil //Optional set user information
        rootViewController:self //rootViewController
        enableAutorefresh:YES //Enable or disable auto refresh
        delegate:self  //Optional
];
OR
adView.delegate = self;
```

Select Ad Unit Size

Use the below enum table to select the needed ad unit size

Enum	Description	Size	Devices
kADFAdViewAdUnitSizeAutoBanner	The server will return the best banner ad based on the screen size and device type		All
kADFAdViewAdUnitSize320x50	Standard	320 x 50	All
kADFAdViewAdUnitSize300x250	Medium Rectangle	300 x 250	All
kADFAdViewAdUnitSize468x60	Full Banner	468 x 60	Tablet
kADFAdViewAdUnitSize728x90	Leaderboard	728 x 90	Tablet
kADFAdViewAdUnitSize120x600	Skyscraper	120 x 600	Tablet

Refresh Ad

Use the below methods to enable, pause, resume or force refresh Ad.

Method	Description
-(void) initializeWith AdUnit:(ADFAdViewAdUnitSize) adUnit siteId: (NSString*) siteId userInfo: (ADFUserInfo*) userInfo rootViewController: (UIViewController*) rootViewController enableAutorefresh: (BOOL) enableAutorefresh delegate: (NSObject<ADFAdViewDelegate> *) delegate;	Enable or disable auto refresh ad
-(void) pauseAutoRefresh;	Will pause auto refresh timer which is responsible for getting new ad the ad after a certain period has elapsed.
-(void) resumeAutoRefresh;	Will resume the auto refresh timer which responsible for getting new ad after a certain period has elapsed.
-(void) refreshAd;	Will get a new ad from AdFalcon.

5 Interstitial Ad Integration Steps

This section describes how to integrate with the AdFalcon's iOS SDK in order to consume the services available by the network.

in order to integrate with the SDK, please perform the following steps:

Add ADFInterstitialAd to your ViewController header file

1. Import the module of AdFalcon SDK "**AdFalconSDK"**.
2. implement **ADFInterstitialAdDelegate protocol** and declare **ADFInterstitialAd** instance**.**

Below is an example of how the header should look like:

```
@import "AdFalconSDK";

@interface ViewController: UIViewController<ADFInterstitialAdDelegate> {
    ADFInterstitialAd * adfInterstitial;
}
```

Add ADFInterstitialAd to your ViewController implementation

1. Add the following code in viewDidLoad method

```
//initialize ADFInterstitialAd
adfInterstitial = [[ADFInterstitialAd alloc] init];

//If you want to receive test ads
adView.testing = YES;

//load new interstitial
[adfInterstitial loadInterstitialWithSiteID:@"Your site ID" delegate:self];
```

Note: *If more information are available about the user, then it is recommended that you call the method below instead and pass the user information in* *ADFTargetingParams* *parameter in the same way this is handled in the Standard Banner Ad.*

```
-(void) loadInterstitialWithSiteID:(NSString*) siteID testing:(BOOL) testing
delegate:(NSObject<ADFInterstitialAdDelegate> *) delegate
params:(ADFTargetingParams*) params;
```

2. Add the following required delegate methods for the ADFInterstitialAdDelegate protocol.

```
-(void)adfInterstitialDidLoadAd: (ADFInterstitialAd*) adfInterstitial{
        [adfInterstitial presentFromRootViewController:self]
}

-(void)adfInterstitialDidDismissScreen: (ADFInterstitialAd*) adfInterstitial{
}

-(void)adfInterstitial: (ADFInterstitialAd*) adfInterstitial didFailWithErrorCode:(int) code
message:(NSString*)message{
switch (code) {
      case kADFAdViewErrorNoAdAvailabe:
        // No Ad Availabe
```

```
                break;
            case kADFAdViewErrorInvalidParam:
                // Invalid Param send to server
                break;
            case kADFAdViewErrorMissingParam:
                // Missing Param send to server
                break;
            case kADFAdViewErrorCommunication:
                //Communication with server failed or no internet
                break;
            default:
                break;
        }
    }
```

6 Appendix

ADFAdView Class

This is the main class and it extends UIView. This Class is responsible for gathering all the needed information to get an ad from AdFalcon network and rendering it on the device.

The properties of the ADFADView Class are:

Property	Description
int refreshDuration	Refresh duration in seconds
NSObject<ADFAdViewDelegate> * delegate	Delegate of AdFalcon view
BOOL testing	The property is used to inform AdFalcon network that the application is under the testing mode rather than production mode.
BOOL logging	Enable logging
ADFTargetingParams targetingParams	This property is used to hold the extra information about the current user of App.

The methods of the ADFADView Class are:

Method	Description
-(void) initializeWith AdUnit:(ADFAdViewAdUnitSize) adUnit siteId: (NSString*) siteId userInfo: (ADFUserInfo*) userInfo rootViewController: (UIViewController*) rootViewController enableAutorefresh: (BOOL) enableAutorefresh delegate: (NSObject<ADFAdViewDelegate> *) delegate	Will initialize the AdFalcon view and load the ad. All the parameters are required except “delegate” which can be null.
-(void) pauseAutoRefresh;	Will pause auto refresh timer which is responsible for getting new ad the ad after a certain period has elapsed.
-(void) resumeAutoRefresh	Will resume the auto refresh timer which responsible for getting new ad after a certain period has elapsed.
-(void) refreshAd	Will get a new ad from AdFalcon.

ADFAdViewDelegate Protocol

AdFalcon SDK provides a protocol called ADFAdViewDelegate that is responsibles for providing feedback from AdFalcon SDK. This protocol contains nine optional delegate methods which are called upon the following events:

- When an error occurs
- Before and After an ad is loaded
- Before and After ad click action screen is displayed
- Before and After ad click action screen is closed
- Application will go to background mode and
- Application will terminate.

The methods of the ADFADViewDelegate Class are:

Method	Description
-(void)adViewWillLoadAd:(ADFAdView*) adView;	Will be called before the ad is being loaded.
-(void)adViewDidLoadAd: (ADFAdView*) adView;	Will be called after the ad has been loaded
-(void)adView: (ADFAdView*) adView didFailWithCode:(int) code message:(NSString*) message;	Will be called when an error has occured during loading an ad.
-(void)adViewWillPresentScreen: (ADFAdView*) adView;	Will be called before the ad click screen is displayed.
-(void)adViewDidPresentScreen: (ADFAdView*) adView;	Will be called after the ad click screen is displayed.
-(void)adViewWillDismissScreen: (ADFAdView*) adView;	Will be called before the ad click screen is dismissed.
-(void)adViewDidDismissScreen: (ADFAdView*) adView;	Will be called after the ad click screen is dismissed.
-(void)applicationWillTerminate:(UIApplication *)application;	Will be called before the Application is terminated.
-(void)applicationWillEnterBackground:(UIApplication *)application;	Will be called before the Application enters into background mode.

ADFInterstitialAd Class

This Class is responsible for gathering all the needed information to get an interstitial ad from AdFalcon network and presenting it on the device.

The properties of the ADFInterstital Class are:

Property	Description
NSObject<ADFInterstitialAdDelegate> * delegate	This property is required. It is used to pass a reference to the class implementing AdFalcon

	interstitial delegate.
NSString * siteId	This property is required. Represents the Id of App.
BOOL testing	The property is used to inform AdFalcon network that the application is under the testing mode rather than production mode. Before releasing your application ensure this is set to false.
BOOL isReadyToPresent	The value is true when the ad is loaded successfully and ready to be presented otherwise will be false.
ADFTargetingParams * targetingParams	This property is used to hold the extra information about the current user of App.
BOOL logging	Enable logging

The methods of the ADFADView Class are:

Method	Description
-(void) loadInterstitial;	Used to load new interstitial Ad but before using this method you must fill the required properties.
-(void) loadInterstitialWithSiteID:(NSString*) siteID delegate:(NSObject<ADFInterstitialAdDelegate> *) delegate;	Used to load new interstitial Ad with required parameters.
-(void) loadInterstitialWithSiteID:(NSString*) siteID testing:(BOOL) testing delegate:(NSObject<ADFInterstitialAdDelegate> *) delegate params:(ADFTargetingParams*) params;	Used to load new interstitial Ad with all possible parameters including additional user information.
-(void) presentFromRootViewController:(UIViewController*) rootViewController;	Used to present the loaded interstitial from specified root view controller. You should not call this method before the interstitial is loaded successfully. So we recommended to call this method when The Interstitial fires *adfInterstitialDidLoadAd* delegate's method.

ADFInterstitialAdDelegate Protocol

AdFalcon SDK provides a protocol called ADFInterstitialAdDelegate that is responsible for providing feedback from interstitial view. This protocol contains delegate methods which are called upon the following events:

- When an error occurs
- Before and After an ad is loaded
- Before and After interstitial is displayed
- Application will go to background mode and will be terminated.

The methods of the ADFADViewDelegate Class are:

Method	Description
-(void)adfInterstitialDidLoadAd: (ADFInterstitialAd*) adfInterstitial;	Required delegate method and will be called after the ad has been loaded.
-(void)adfInterstitialDidDismissScreen: (ADFInterstitialAd*) adfInterstitial;	Required delegate method and will be called after the interstitial screen has been dismissed.
-(void)adfInterstitial: (ADFInterstitialAd*) adfInterstitial didFailWithErrorCode:(int) code message:(NSString*)message;	Required delegate method and will be called when an error has occured during loading an ad.
-(void)adfInterstitialWillPresentScreen: (ADFInterstitialAd*) adfInterstitial;	Optional. Will be called before the interstitial screen is presented.
-(void)adfInterstitialWillDismissScreen: (ADFInterstitialAd*) adfInterstitial;	Optional. Will be called before the interstitial screen is dismissed.
-(void)adfInterstitialAppWillTerminate:(ADFInterstitialAd*) adfInterstitial;	Optional. Will be called before the Application is terminated.
-(void)adfInterstitialAppWillEnterBackground:(ADFInterstitialAd*) adfInterstitial;	Optional. Will be called before the Application enters into background mode.

ADFUserInfo Class

The ADFUserInfo Class contains all the needed parameters about the user of application.

The parameters of the ADFUserInfo Class are:

Parameter	Required	Description	Values
Language	No	The language of the requested ad in ISO 639-1 language codes format (Two Letters code);	ar, en
Postal code	No	A parameter containing the user's postal/ZIP code	11121
Area code	No	A parameter containing the user's area code	06
Age	No	A parameter containing the user's age	27
Gender	No	A parameter containing the user's	kADFUserInfoGenderNone kADFUserInfoGenderMale kADFUserInfoGenderFemale

		gender	
	Country code	No	County code of the end user in ISO 3166-1 alpha-2 format code (two-letter code)	JO, SA …etc.
	Birthdate	No	Birthdate of application user in format dd.MM.yyyy	21.11.1984
	Location: Latitude, Longitude	No	The geolocation information of the device. The location information is divided into two double values; latitude and longitude.	35.658, 34.641

ADFTargetingParams Class

The ADFTargetingParams Class contains all the needed parameters about any given user and application in order to help adFalcon network to send most related and targeted ads to the user and application. All parameters are optional.

The parameters of the ADFTargetingParams Class are:

	Parameter	Required	Description	Values
	userInfo	No	A class containing information about the user of application	ADFUserInfo
	Keywords	No	A list conataining keywords in comma separated format. AdFalcon's ads selector engine will search for ads containing these keywords.	ex. sport, news, lifestyle, …etc.
	Additional Info	No	A map of keys and values to add	

	additional parameters.

ADFAdViewAdUnitSize Enum

ADSAdViewAdUnitSize Enum defines the ad units supported by the AdFalcon iOS SDK

Enum	Description	Size	Devices
kADFAdViewAdUnitSizeAutoBanner	The server will return the best banner ad based on the screen size and device type		All
kADFAdViewAdUnitSize320x50	Standard	320 x 50	All
kADFAdViewAdUnitSize300x250	Medium Rectangle	300 x 250	All
kADFAdViewAdUnitSize468x60	Full Banner	468 x 60	Tablet
kADFAdViewAdUnitSize728x90	Leaderboard	728 x 90	Tablet
kADFAdViewAdUnitSize120x600	Skyscraper	120 x 600	Tablet

ADFAdViewError Enum

Enum	Description
kADFAdViewErrorInternalServer	is an error that's happened within the web server attempting to get you an ad. It's typically a server-side problem out of your control
kADFAdViewErrorNoAdAvailabe	no ad available in AdFalcon stores
kADFAdViewErrorInvalidParam	There is a parameter has invalid value
kADFAdViewErrorMissingParam	You have missed to fill required parameter
kADFAdViewErrorGenericSDK	An error happened within the SDK during attempting to load or render an ad.
kADFAdViewErrorCommunication	No connection available to the internet.
kADFAdViewErrorInterstitialAlreadyUsed	An error happened when application try to reuse Interstitial

7 More Information:

You can find more information in the sample project within the downloaded zip file.

For any SDK integration queries, please send us an email to support@adfalcon.com along with your login id.