AdFalcon Android Native Ad SDK 3.4.0 Developer's Guide

AdFalcon Mobile Ad Network

Product of

Noqoush Mobile Media Group

Table of Contents

1 Introduction

AdFalcon Android SDK Native Ad integration guide contains all the information needed by android developers to integrate with AdFlacon Network to implement native ads in their apps. The guide will also provide examples and code snippets on how to perform the integration with ease.

Native Ads Overview

Native ads is a form of digital advertising where the ad experience follows the natural form and function of the user experience in which it is placed.

Unlike standard ads, where AdFalcon SDK displays the ad, when loading native ads AdFalcon SDK returns a native ad object with the ad's assets and the app itself is responsible for displaying the ad. With Native Ads you control the look and feel, design, layout and position of the ad to seamlessly be an integral part of your app.

OS version support

Android SDK supports mobile phones and tablets platforms utilizing **Android version 4.0** and above.

2 Native Ad Integration Steps

AdFalcon SDK aims to make native ads implementation in your app a straightforward process where you control the look and feel, design and placement of the ad while AdFalcon SDK handles the rest of the operations needed to display the ad and process impressions and clicks successfully.

When loading native ads, AdFalcon SDK returns a native ad object with the ad's assets, and the app itself is responsible for displaying the native ad by providing a template XML layout that lays out the native ad assets in the desired design and look and feel. This process is achieved by mapping the different Views (icon, title, description…etc.) in the template to the native ad assets.

The sections below provides a step-by-step guide for native ad implementation.

Step 1: Add AdFalcon SDK

Import AdFalcon SDK jar file into your project, to be able to use it.

Step 2: Configure Android Manifest

1. **Configure Application Permissions**
   Add the below permissions to your application **manifest.xml**

   ### Required permissions
   The below permissions are required to use the SDK.

   In case you do not enable them, the SDK will throw an exception to inform you that "INTERNET/ACCESS_NETWORK_STATE permissions must be enabled in AndroidManifest.xml".

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

   ### Recommended Permissions
   The below permissions are highly recommended as they allow highly interactive rich media ads.

   If you decide not to add any of the permissions below the AdFalcon SDK will continue to work yet some of the rich media ads' functions will be disabled.

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.WRITE_CALENDAR"/>
<uses-permission android:name="android.permission.READ_CALENDAR"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

2. **Add the SDK's Activities**
   Add the below activities to your AndroidManifest.

```
<activity
        android:name="com.noqoush.adfalcon.android.sdk.ADFBrowser"
        android:configChanges="keyboard|keyboardHidden|orientation|uiMode|screenLayout|screenSize|sm
allestScreenSize"
        android:hardwareAccelerated="true">
</activity>
<activity
        android:name="com.noqoush.adfalcon.android.sdk.ADFActivity"
        android:configChanges=
"keyboard|keyboardHidden|orientation|uiMode|screenLayout|screenSize|smallestScreenSize"
        android:theme="@android:style/Theme.Translucent.NoTitleBar.Fullscreen"
        android:hardwareAccelerated="true">
</activity>
```

3. **Configure Orientation Changes Handling**
   It is highly recommended to prevent the system from restarting your activity when the screen orientation changes. Add the below attribute to your activity tag:

```
android:configChanges=
"keyboard|keyboardHidden|orientation|uiMode|screenLayout|screenSize|smallestScreenSize"
```

   For more information please refer to
   http://developer.android.com/guide/topics/resources/runtime-changes.html

4. **Hardware Acceleration**
   AdFalcon network supports HTML 5 video ads which requires enabling hardware acceleration.

   We recommended that the developer enables hardware acceleration in his application by adding the below property to application tag or to each activity that contains AdFalcon banner.

```
android:hardwareAccelerated="true"
```

5. **Adding Google Play Services**
   The terms and conditions of Google Play requires using the Advertising ID to identify and track a user. The Advertising Id is included in Google Play services library, so you must add Google Play services library to your application to integrate with AdFalcon Android SDK. Visit the below link for instruction on how to add Google play services library:

   http://developer.android.com/google/play-services/setup.html

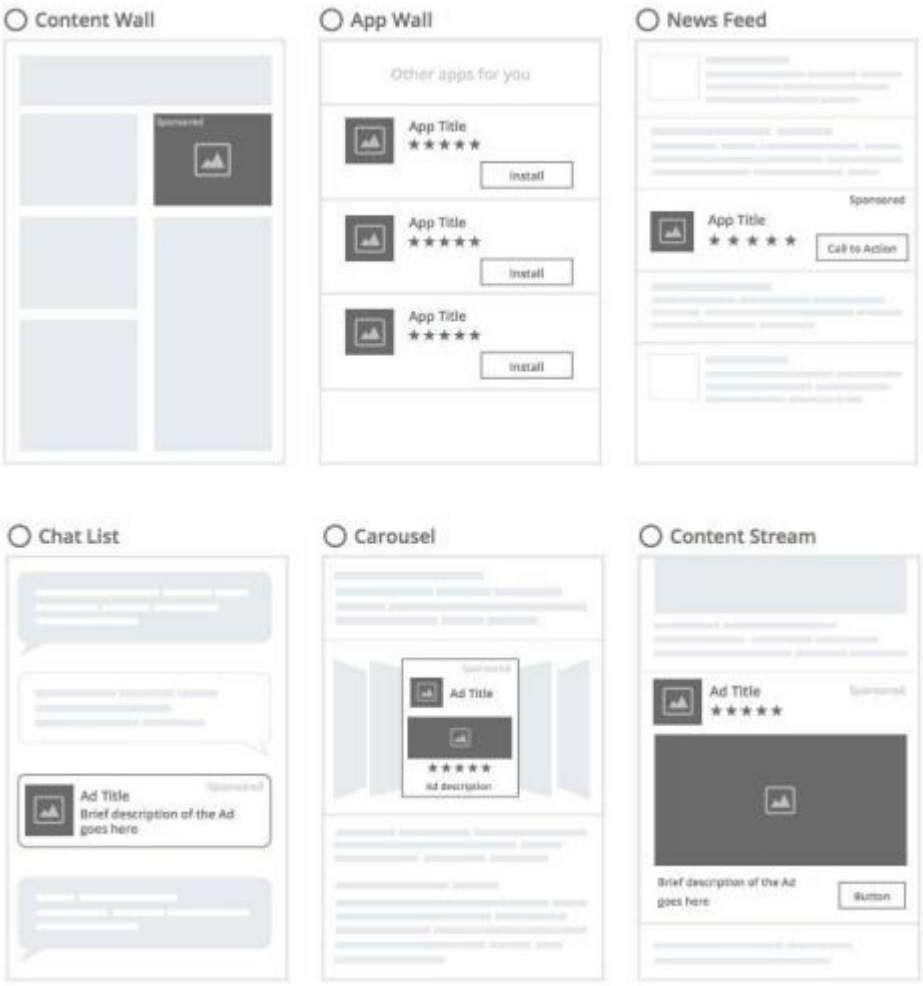
Step 3: Creating Native Ad Template

In this step, you will learn how to create your native ad XML template layout and how to bind your XML layout's resources to the native ad assets.

Step 3.1 Create Native ad XML layout

The native ad template consists of multiple Views, these Views are mapped to the native ad assets. You should include in the template only the native ad assets that are needed as per the desired design.

Some examples for native ad templates are shown below.



You can state what native ad assets are requested by including them in the native ad template layout. AdFalcon will only return native ads that include the requested assets as per the below rules:

- **Required Assets**
  Your native ad must display **AdChoices** view and at least one of the assets below otherwise it will be considered as invalid template:
  - **Icon**
  - **Title**
  - **MainAsset**

AdFalcon will only return ads which contain **all the required assets** included in the native ad template

- **Optional Assets**
  All remaining native ad asset types are optional.

  The template may include one or more of the optional assets or none at all as per the desired design.

  The returned native ad may or may not include any of the requested optional assets; i.e. the optional native ad assets will only be returned when available.

  If a native ad response does not contain any of the optional assets, the SDK will hide its associated view.

Refer to Native Ad Template Assets in Appendix for all Native Ad Template Resources which can be included in the template layout.

In the following example, we will create a template that contains icon, title, (image or xhtml), description, rating bar, sponsored and CTA for the native ad:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:padding="16dp">

   <RelativeLayout
      android:id="@+id/adchoices"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignParentRight="true" />

   <ImageView
      android:id="@+id/template_icon"
      android:layout_width="50dp"
      android:layout_height="50dp"
      android:layout_margin="10dp"
      android:src="@mipmap/ic_launcher"
      android:scaleType="fitXY" />

   <TextView
      android:id="@+id/template_title"
      style="?android:attr/textAppearanceLarge"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginTop="10dp"
      android:text="Native Ad Title"
      android:layout_toRightOf="@id/template_icon" />

   <TextView
      android:id="@+id/template_description"
```

```
        style="?android:attr/textAppearanceMedium"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/template_title"
        android:layout_marginTop="10dp"
        android:text="Native Ad Description"
        android:layout_toRightOf="@id/template_icon" />

    <RelativeLayout
      android:id="@+id/template_main_asset"
        android:layout_width="match_parent"
        android:layout_height="100dp"
        android:layout_below="@id/template_description"
        android:layout_marginTop="10dp" />

    <RatingBar
        android:id="@+id/template_rating"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/template_main_asset "
        android:layout_marginTop="10dp" />

    <TextView
        android:id="@+id/template_sponsored"
        style="?android:attr/textAppearanceMedium"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/template_rating "
        android:layout_marginTop="10dp" />

     <Button
        android:id="@+id/template_action_button"
        style="?android:attr/textAppearanceMedium"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:layout_below="@id/template_sponsored"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp"
        android:text="@string/app_name" />

</RelativeLayout>
```

Step 3.2 Bind your XML template layout's resources to native ad assets
This step shows how to connect your native ad XML template layout's resources to their corresponding native ad assets (title view, main image, icon, etc.).

You should at minimum bind one of the mandatory native ad assets (Icon, Title or MainAsset) depending on the native ad template design in-place.

The binder requires setting parameters for each native ad asset, these parameters depend on the View type and its abilities. In case of TextView, the developer should pass the maximum text length that can be rendered in the TextView perfectly. In case of ImageView, the developer should pass the minimum width and height of the image which can be rendred clearly in the ImageView.

Binding views to native ad assets is achieved using the binder object of ADFAssetsBinder class by calling the appropriate binding method (addIconImageView, addTitleImageView,

addMainAssetRelativeLayout, addDescriptionTextView, addActionButton, addStarRatingBar and addExtraDataTextView).

Please refer to Appendix 6 ADFAssetsBinder for more details.

Below is a sample bindViews method implementation:

```
ADFAssetsBinder binder = new ADFAssetsBinder.Builder(this, R.layout.template_layout).
addAdChoicesRelativeLayout(R.id.adchoices_view).
addIconImageView(R.id.icon_template, 50, 50).
addTitleTextView(R.id.title_template,25).
addDescriptionTextView(R.id. template_description, 100).
addMainAssetRelativeLayout(R.id. template_container, 50, 50).
addStarRatingBar(R.id.template _rating).
addExtraDataTextView(ADFAssetsBinder.DATA_ID_SPONSORED, R.id.template_sponsored, 20).
addActionButton(R.id.template_action_button, 10).
build();//this will build a binder which will bind an icon, title, MainAsset, sponsored extra data, and action button.
```

Step 4. Requesting and Handling Native Ad.

Step 4.1 Create an instance of ADFNativeAd

```
ADFNativeAd nativeAd = new ADFNativeAd(this); //this refers to the current activity
```

Step 4.2 Implement `ADFNativeAdListener`

ADFNativeAdListener provides ability to track native ad lifecycle events.

```
nativeAd.setListener(new ADFNativeAdListener(){

        public void onLoadAd(ADFNativeAd nativeAd) {
        //this method will be called once the ad has been loaded and ready to be shown.
        }

        public void onFail(ADFNativeAd ad, ADFErrorCode errorCode, String message) {
        //this method will be called once an error occurred while loading or showing ad.
        }

        public void renderExtraData(View view, Hashtable<Integer, String> hashtable) {
        /* This method is used to allow the app to draw any extra data assets that were not bound
        explicitly in the ADFAssetsBinder. This is used if the app is willing to entirely take over the
        extra data display and presentation.*/
        }

        public void onPresentAdScreen(ADFNativeAd ad){
        //this method will be called once the ad action has been viewed.
        }

        public void onDismissAdScreen (ADFNativeAd ad){
        //this method will be called once the ad action view has been dismissed.
        }

        public void onLeaveApplication (){
        //this method will be called once the ad action has been viewed in another application.
        }

});
```

Step 4.3 Load native ad

Call loadAd method to load a new native ad by passing your site ID and binder.

```
nativeAd.loadAd("your site ID", null, binder);
```

Note: *It is highly recommend that you provide all available targeting information such as gender, location, content category to your ad request in order to receive the best ad that meets the criteria of your audience and maximize your return. This can be done by filling the* ***params*** *object in* ***loadAd*** *method. Please refer to Appendix 6* ***ADFTargetingParams*** *for all the available targeting parameters.*

Step 4.4 Rendering the native ad

After loading the native ad successfully, onLoadAd method will be fired immediately. Hence this method will be used to add the native ad to your ViewGroup.

```
public void onLoadAd(ADFNativeAd nativeAd) {
        //this method will be called once the ad has been loaded and ready to be shown.
        LinearLayout layout = (LinearLayout) findViewById(R.id.layout);
        layout.addView(nativeAd);
}
```

Step 4.5 Destroy the native ad

You should destroy the native ad once **onDestroy** lifecycle method is called.

```
protected void onDestroy() {
        nativeAd.destroy();
        super.onDestroy();
}
```

Step 4.6 Test native ad code

Enable testing parameter to ensure your code and template during development to ensure everything is working as expected.

Add the below line of code before calling **loadAd** method.

```
nativeAd.setTesting(true);
```

Note: *Ensure testing parameter is set to false before uploading your app to the market.*

Advance Native Ad Options

ADFNativeAdStatus getStatus()

This method returns the current lifecycle status of the native ad. The below table illustrates the possible values of **ADFNativeAdStatus**

Enum	Description
loading	Native ad is loading currently.
loaded	Native ad has finished loading.
shown	Native ad has been shown.
failed	An error occurred while loading or showing the native ad. You can call **getErrorMessage()** to obtain more information.
clicked	User has clicked on the native ad.

setLogging(boolean logging)

This method is used to enable or disable the logger of the SDK.

3 Adding Native Ads to List Views via ADFNativeAdAdapter

AdFalcon helps the developers to add native ads to a ListView easily by using ADFNativeAdAdapter which embeds the native ads in the listView.

The default behavior is to insert 3 native ads starting from position 5, and will leave 20 rows between the native ads. These default settings can be overridden by the developer to use the desired configuration.

This adapter is not part of AdFalcon SDK jar, you will find it with the needed classes in the package named “adfhelper” within the sample project of the SDK bundle.

Note: *ADFNativeAdAdapter supports GridView as well, the same procedure below can be used to display Native Ads to GridView.*

Prerequirements

a. ListView
b. BaseAdapter
c. adfhelper package

Step 1 Create the XML template layout

for more details go back to Step 3.1

Step 2 Create the ADFAssetsBinder

for more details go back to Step 3.2

Step 3 Create ADFNativeAdAdapterModel class

This class is the main class responsible for requesting native ads and rendering them in the ListView.

Create ADFNativeAdAdapterModel with the four parameters as the below

a. ListView listView
b. BaseAdapter developerAdapter: which is the adapter the developer is currently using to draw the data in the listview.
c. String siteID
d. ADFAssetsBinder binder

```
ADFNativeAdAdapterModel adapterModel = new ADFNativeAdAdapterModel(listView,
developerAdapter,
        "your site ID", binder);
```

Step 4 Create ADFNativeAdAdapter

```
ADFNativeAdAdapter nativeAdAdapter = new ADFNativeAdAdapter(adapterModel);
```

Step 5 Pass the ADFNativeAdAdapter to the ListView

Once the ADFNativeAdAdapter is passed to the ListView, the ads will begin loading and will be displayed immediatly.

```
listView.setAdapter(nativeAdAdapter);
```

Step 6 Destroy the native ad

You should destroy the native ad once **onDestroy** lifecycle method is called.

```
protected void onDestroy() {
        nativeAdAdapter.destroy();
        super.onDestroy();
}
```

Step 7 Call the below Methods from ADFNativeAdAdapter instead of ListView

a. setSelection(int originalPosition)
b. smoothScrollToPosition(int originalPosition)
c. getChildAt(int index)
d. getFirstVisiblePosition()
e. getLastVisiblePosition()
f. getPositionForView(View view)
g. getItemIdAtPosition(int position)
h. getPositionForView(View view)

Step 8 Set the below listeners to ADFNativeAdAdapter instead of ListView

a. setOnItemClickListener(OnItemClickListener listener)
b. setOnItemLongClickListener(OnItemLongClickListener listener)
c. setOnItemSelectedListener(OnItemSelectedListener listener)

Step 9 Test native ad code

Enable testing parameter to ensure your code and template during development to enure everything is working as expected.

Add the below line of code before passing the ADFNativeAdAdapterModel object to ADFNativeAdAdapter.

```
adapterModel.getAdsInfo().setTesting(true);
```

Note: *Ensure testing parameter is set to false before uploading your app to the market.*

4 Appendix

Native Ad Template Assets

Asset	Required	View Type	Description
AdChoices	Yes	RelativeLayout	An AdChoices icon must be displayed in your ad to enable the user to learn about interest-based advertising. The SDK will decide when it is needed to display the icon and will also handle the tap event on the view by opening an opt-out page in the browser. **Parameters**: The size of the view: **20x20.** You must place the view on any of the four corners of the ad template; upper right hand corner is recommended.
Title	At least one of Title, Icon or MainAsset must be included	TextView	String representation of the native ad, which could be the name of the product or the service being advertised. **Parameters**: Maximum title length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***25 characters***. Only native ads with title assets whose length is less than the requested maximum title length will be returned.
Icon	At least one of Title, Icon or MainAsset must be us included	ImageView	A squared image that shows the icon of the product, service, brand, … etc. **Parameters** : Minimum width and height. The Developer should explicitly set these values while binding the template view to the native ad asset. Recommended minimum width and height of the Icon ImageView is ***50x50***. Only native ads with icon assets whose width and height is less than the requested minimum icon width and height will be returned.

MainAsset	At least one of Title, Icon or MainAsset must be included	RelativeLayout	The Native Ad main asset can be one of the below types: • Image • XHTML (HTML+JavaScript) • Video The MainAsset is a RelativeLayout which is used as a container view for the actual native ad main asset. The SDK will automatically create the adequate RelativeLayout type that match the native ad main asset. **Parameters** : Minimum width and height. The Developer should explicitly set these values while binding the template view to the native ad asset. The main asset RelativeLayout width and height aspect ratio should always be W/H=1.91. It is recommended to use 320x167 points. Only native ads with main assets whose width and height is greater than the requested minimum main asset width and height will be returned.
Sponsored	No	TextView	"Sponsored By" message where the response should contain the brand name of the sponsor. **Parameters**: Maximum sponsored length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***10 characters***.
Description	No	TextView	Descriptive text associated with the product or the service being advertised. **Parameters**: Maximum description length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***100 characters***.
Star Rating	No	RatingBar	Consists of five stars, which represents the rating of an app. **Parameters** : Minimum width and height. The

			Developer should explicitly set these values while binding the template view to the native ad asset. Recommended minimum width and height for the Star Rating ImageView is to follow aspect ratio of W/H=5 such as ***100x20***.
Likes	No	TextView	Number of social ratings or "likes" of the product being offered to the user. **Parameters**: Maximum likes length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***20 characters***.
Downloads	No	TextView	Number of downloads of the product being offered to the user. **Parameters**: Maximum downloads length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***25 characters***.
Price	No	TextView	Price for product / app / in-app purchase. Value should include currency symbol in localized format. **Parameters**: Maximum price length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***20 characters***.
Sale price	No	TextView	Sale price that can be used together with price to indicate a discounted price compared to a regular price. Value should include currency symbol in localized format. **Parameters**: Maximum sale price length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***15 characters***.

Phone	No	TextView	Formatted string that represents the phone number. **Parameters**: Maximum phone length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***20 characters***.
Address	No	TextView	Address data. **Parameters**: Maximum address length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***100 characters***.
Description 2	No	TextView	Additional descriptive text associated with the product or service being advertised. **Parameters**: Maximum Address 2 length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***100 characters***.
Display URL	No	TextView	Display URL data. **Parameters**: Maximum display URL length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***50 characters***.
Call to action “CTA”	No	Button	Descriptive text describing a ‘call to action’ button for the destination URL such as open, register, install. **Parameters**: Maximum CTA length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the label of button is laid out to display at least ***20 characters***.

Views	No	TextView	Number of times the ad has been viewed. **Parameters**: Maximum Views length. The Developer should explicitly set this value while binding the template view to the native ad asset. It is recommended that the TextView is laid out to display at least ***20 characters***.

ADFNativeAdListener interface

Method	Description
public void onLoadAd(ADFNativeAd ad)	Fired when the ad is loaded successfully.
public void onFail(ADFNativeAd ad, ADFErrorCode errorCode, String message)	Fired when failed to load an ad with failure message. **Parameters** ad: AdFalcon native ad error: type of error message: description for error
public void renderExtraData (View view, Hashtable<Integer, String> hashtable)	This method is used to allow the app to draw any extra data assets that were not bound explicitly in the ADFAssetsBinder. This is used if the app is willing to entirely take over the extra data display and presentation.
public void onPresentAdScreen(ADFNativeAd ad)	Ad action has been viewed.
public void onDismissAdScreen(ADFNativeAd ad)	Ad action has been closed.
public void onLeaveApplication()	Ad action has been viewed in another application

ADFTargetingParams class

Variable	Required	Description	Values
Language	No	A two-characters string parameter indicating Ad language	ar, en
Postal code	No	A string parameter containing the user's postal/ZIP code	11121
Area code	No	A string parameter containing the user's area code	06
Age	No	An integer parameter containing the user's age	27

Keywords	No	List of keywords in comma separated format. AdFalcon's ads selector engine will search for ads containing these keywords.	ex. sport, news, lifestyle, …etc.
Gender	No	A parameter containing the user's gender	NONE GENDER_MALE GENDER_FEMALE
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	
Additional Info	No	A Map of keys and values to add additional parameters.	
Location: Latitude, Longitude	No	The geolocation information of the device. The location information is divided into two double values; latitude and longitude.	35.654, 34.6598

ADFNativeAd class

Method	Description
public void setTesting(**boolean** test)	Used to inform AdFalcon network if your application is working in test or production mode.
public void setListener(ADFNativeAdListener listener)	Set ADFNativeAdListener.
public ADFNativeAdListener getListener()	Get ADFNativeAdListener.
public void setLogging(**boolean** logging)	Enable/Disable logging while loading native ad.
public boolean getLogging()	Check if logging is enabled.
public ADFNativeAdStatus getStatus()	Get status of native ad. such as loading, loaded, shown, clicked and failed.
public String getErrorMessage()	Get error message when the status is failed
public void destroy()	This method should be called either in onDestroy() method to finalize the ADFNativeAd or when you are done with the ADFNativeAd object. After calling this method the SDK will dispose the native ad.
public void loadAd(String siteID, ADFTargetingParams params, ADFAssetsBinder binder)	This method is used to load new ad.

ADFAssetsBinder.Builder class

Method	Description
public void addIconImageView (**int** resID, **int** minWidth, **int** minHeight)	Binds the native ad icon asset to its corresponding ImageView in the template. **Params** • resID: resource id of the image view in the layout. • minWidth: minimum width of icon. • minHeight: minimum height of icon.
public void addTitleTextView (**int** resID, **int** maxLen)	Binds the native ad title asset to its corresponding TextView in the template. **Params** • resID: resource id of the text view in your created layout. • maxLen:maximum length of title text.
public void addMainAssetRelativeLayout (**int** resID, **int** minWidth, **int** minHeight)	Binds the native ad main asset to its corresponding RelativeLayout in the template. The MainAsset is a RelativeLayout which is used as a container view for the actual native ad main asset which can be: Image, or XHTML (HTML+JavaScript), or Video The SDK will automatically create the adequate View type that match the native ad main asset. **Params** • resID: resource id of the RelativeLayout in the template. • minWidth: minimum width of the content of RelativeLayout. • minHeight: minimum height of the content of RelativeLayout.
public void addDescriptionTextView (**int** resID, **int** maxLen)	Binds the native ad description asset to its corresponding TextView in the template. **Params** • resID: resource id of the view in your created layout. • maxLen:maximum length of description text.
public void addStarRatingBar (**int** resID)	Binds the native ad rating asset to its corresponding RatingBar in the template.

	Params • resID: resource id of the view in your created layout.
public void addActionButton (**int** resID, **int** maxLen)	Binds the native ad button asset to its corresponding Button in the template. **Params** • resID: resource id of the view in your created layout. • maxLen:maximum length of action text.
public void addExtraDataTextView (**int** dataID, **int** resID, **int** maxLen)	Binds the native ad data asset to its corresponding TextView in the template. All extra data views are optional which means native ad can be returned even if they don't include the requested extra data assets. Additionally, the SDK will hide any extra data view that is not included in the returned native ad. **Params** • dataID: data id that you want to show based on enum in ADFAssetsBinder class • resID: resource id of the view in your created layout. • maxLen:maximum length of data content.
public void addAdChoicesRelativeLayout (**int** resID)	An AdChoices icon must be displayed in your ad to enable the user to learn about interest-based advertising. The SDK will decide when it is needed to display the icon and will also handle the tap event on the view by opening an opt-out page in the browser. **Parameters**: The size of the view: **20x20.** You must place the view in any one of the four corners of the ad template; upper right hand corner is recommended.
public ADFAssetsBinder build()	This method is used to build the native ad.

ADFAssetsBinder data id enum

ADFAssetsBinder data id enum defines the extra data that you want to add in your native ad.

Enum	Description
DATA_ID_SPONSORED	Sponsored By message where response should contain the brand name of the sponsor.
DATA_ID_LIKES	Number of social ratings or "likes" of the product being offered to the user.
DATA_ID_DOWNLOADS	Number of downloads of the product being offered to the user.

DATA_ID_PRICE	Price for product / app / in-app purchase. Value should include currency symbol in localised format.
DATA_ID_SALEPRICE	Sale price that can be used together with price to indicate a discounted price compared to a regular price. Value should include currency symbol in localised format.
DATA_ID_PHONE	Phone number
DATA_ID_ADDRESS	Additional address text associated with the product or service being advertised
DATA_ID_DESC2	Additional descriptive text associated with the product or service being advertised
DATA_ID_DISPLAYURL	Display URL for the text ad
DATA_ID_VIEWS	Number of views of this ad

ADFNativeAdStatus enum

ADFNativeAdStatus enum defines the status of the native ad.

Enum	Description
Loading	Native ad is loading currently.
Loaded	Native ad has finished loading.
Shown	Native ad has been shown.
Failed	An error occurred while loading or showing the native ad.
Clicked	User has clicked on the native ad.

ADFErrorCode enum

ADFErrorCode Enum defines AdFalcon SDK and Ad Server error codes.

Enum	Description
GENERIC_SDK_ERROR	Error occurred in the SDK.
INTERNAL_SERVER_ERROR	Error occurred in AdFalcon server.
COMMUNICATION_ERROR	SDK can not reach the AdFalcon server; mostly for issues related to the phone internet connection.
NO_AD_AVAILABLE	No ad is available in the AdFalcon store that meets the ad request parameters.
INVALID_PARAM	An invalid ad request parameter has been set.
MISSING_PARAM	A required ad request parameter has not been set.

5 More Information:

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

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