

ANDROID AD FORMATS INTEGRATION GUIDE

June 2023

Integrating the SDK into Android Studio manually.

Android Studio User:

Step 1: Extract the zip file and copy and paste the .aar file into the project for integration

Step 2: Import .aar file into project workspace and add the implementation to the application

Step 3: Follow the below code structure for integration

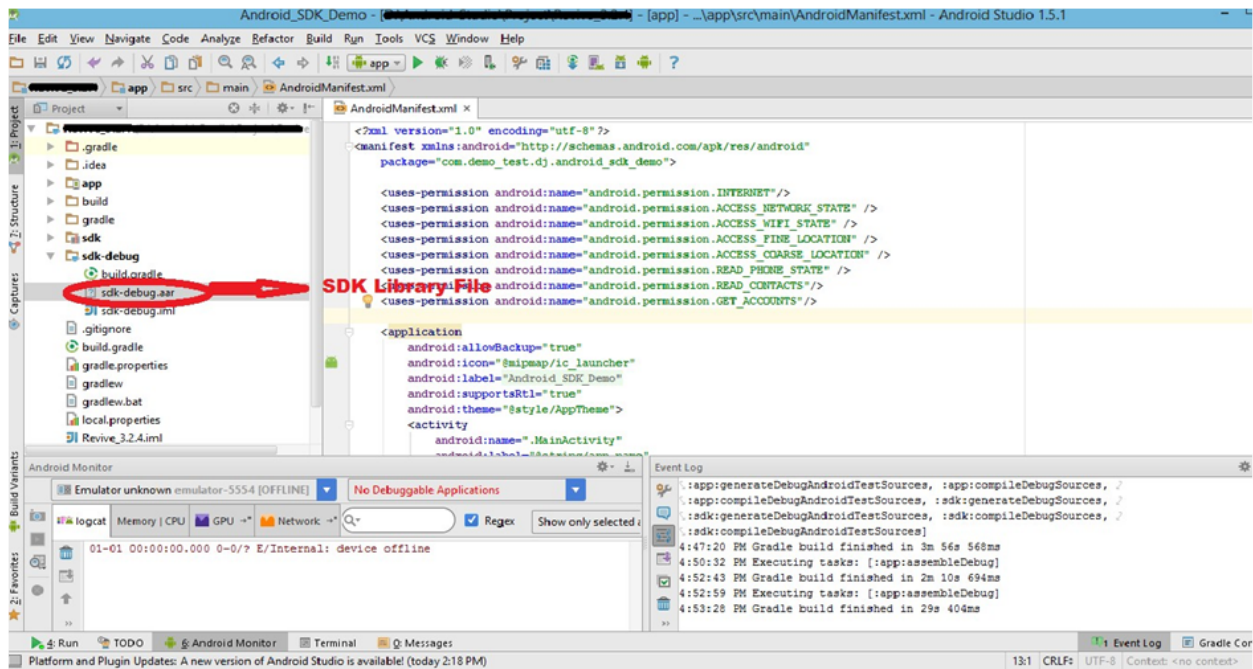


Fig: Android Studio with Library File

Minimum Requirements

Minimum OS : Android version 6

Maximum OS : Android version 12

API Levels Supported : 21 to 31

Enable App Permissions

Add the below permissions to the AndroidManifest file.

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

Permission Table:

S.NO	PERMISSION	IMPORTANT	DESCRIPTION
1	INTERNET	Required	Grants the SDK permission to access the internet.
2	ACCESS_NETWORK_STATE	Required	Grants the SDK permission to check for a live internet connection. Allows applications to access information about networks
3	ACCESS_FINE_LOCATION	Optional	Grants the SDK permission to access a more accurate location based on GPS. For GEO ads targeting
4	ACCESS_COARSE_LOCATION	Optional	Grants the SDK permission to access an approximate location based on a cell tower.

Implementations

Implement the below packages into the app level gradle file.

```
implementation 'androidx.multidex:multidex:2.0.1'
implementation 'androidx.appcompat:appcompat:1.3.1'
implementation 'com.google.android.exoplayer:exoplayer-core:2.17.1'
implementation 'com.google.android.exoplayer:exoplayer-ui:2.17.1'
implementation 'com.google.android.exoplayer:extension-ima:2.17.1'
implementation 'com.squareup.picasso:picasso:2.5.0'
implementation 'com.google.code.gson:gson:2.8.5'
implementation 'io.reactivex.rxjava2:rxandroid:2.1.1'
implementation 'io.reactivex.rxjava2:rxjava:2.1.9'
```

implementation files('sdk-release.aar')

Location permissions can help monetization

Although not technically required, the *LOCATION permissions make it possible for the SDK to send location-based data to advertisers. Sending better location data generally leads to better monetization. LOCATION is only required for GEOfencing and report targeting based ads. Instead, it will use these permissions to access the last known location of the device.

Quick Guide

This section describes some of the code written to the app in order to show ads.

The document refers to the "zone ID". A zone ID is used by MSDK to identify a context within the app where advertisements can be shown. App is needed to obtain a zone ID from the MSDK representative or is needed to the ad network. Without it, app can't be able to fetch and display ads.

XML	Call the Function	Description	Example
msdk:zone_id	ad.setZoneid()	The zone ID associated with the app's inventory. Must include a zone ID or an error will be thrown.	436
msdk:auto_refresh_time	ad.set_Auto_refresh_time ()	The interval, in milliseconds, at which the AdView will Request new ads, if auto refresh is enabled. The minimum period is 15 seconds. The default period is 30 seconds. Set this to 0 to disable auto refresh.	30000

In the XML file add the below line for auto refresh the UI layout
xmlns:msdk="http://schemas.android.com/apk/res-auto"

Main activity or Main method is needed to integrate these steps to call the SDK.

```
AdView adv1 = new AdView(ZoneActivity.this);
adv1.setZoneid("Enter zone id here");
adv1.LoadAd(adv1);
```

Image Ad

Configure the banner ad using Java, XML, or a mixture of the two.

Code Snippets

To show the Banner ads inside the application, by implementing the listener/interface in the java class. **BannerListener** is the listener for banner ads. It contains two methods,

```
public class BannerActivity extends AppCompatActivity implements BannerListener{
```

Implementing Listener :

BannerListener is common for all banner ads. If it is needed

```
@Override
public void AdLoaded() {
}
@Override
public void AdFailed() {
}
```

Xml Format:

To show the banner ads center of the content OR middle of the content means must include the below XML code for showing the banner ads.

```
<WebView
    android:id="@+id/banner1"
    android:layout_width="match_parent"
    android:layout_height="60dp"
    tools:ignore="WebViewLayout" />
```

Code Format :

Below the code set for integration of Banner ads inside the content.

```
WebView webView = findViewById(R.id.banner1);
```

Explanation about the integration steps:

```
new BannerAds().loadBannerAd(this,webView);
```

The **AdView()** function is the entry point of the Android SDK. When calling this function it will create a new object for the AdView class and access the methods of AdView class. Also, Pass the **appcontext** and **webview** in the AdView class.

Top Banner Ad

Configure the Top Banner ad using Java, XML, or a mixture of the two.

Code Snippets

To show the Top Banner ads inside the application, by implementing the listener/interface in the java class. BannerListener is the listener for banner ads. It contains two methods,

```
public class TopBanner extends AppCompatActivity implements BannerListener
```

Xml Format:

To show the direct link ads center of the content OR middle of the content means must include the below XML code for showing the banner ads.

```
<WebView  
    android:id="@+id/banner1"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    tools:ignore="WebViewLayout" />
```

Code Format :

Below the code set for integration of Top Banner ads inside the content.

```
WebView webView = findViewById(R.id.banner1);
```

Explanation about the integration steps:

```
new TopBannerAD().loadImageBanner(TopBanner.this,webView);
```

The **AdView()** function is the entry point of the Android SDK. When calling this function it will create a new object for the AdView class and access the methods of AdView class. Also, Pass the app **context** and **webview** in the AdView class.

Banner Slider Ad

Configure the BannerSlider ad using Java, XML, or a mixture of the two.

Code Snippets

To show the BannerSlider ads inside the application, by implementing the listener/interface in the java classBannerListener is the listener for banner ads. It contains two methods,

```
public class BannerSlider extends AppCompatActivity
```

Xml Format:

To show the banner slider ads bottom of the content showing the banner slider ads.

```
<WebView  
    android:id="@+id/banner1"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    tools:ignore="WebViewLayout" />
```

Code Format :

Below the code set for integration of Banner Slider ads inside the content.

```
WebView webView = findViewById(R.id.banner1);
```

Explanation about the integration steps:

```
new BannerSliderAd().loadBannerSlider(BannerSlider.this,webView);
```

The **AdView()** function is the entry point of the Android SDK. When calling this function it will create a new object for the AdView class and access the methods of AdView class. Also, Pass the app **context** and **webview** in the AdView class.

Direct Link Ad

Configure the direct link ad using Java, XML, or a mixture of the two.

Code Snippets

To show the Direct Link ads inside the application, by implementing the listener/interface in the java class. **BannerListener** is the listener for direct link ads. It contains two methods,

```
public class DirectLinkAd extends AppCompatActivity implements BannerListener
```

Xml Format:

To show the direct link ads center of the content OR middle of the content means must include the below XML code for showing the banner ads.

```
<TextView  
    android:id="@+id/banner2"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    tools:ignore="WebViewLayout" />
```

Code Format :

Below the code set for integration of Direct Link ads inside the content.

```
TextView wb1 = findViewById(R.id.banner1);
```


Explanation about the integration steps:

```
wb1.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        Intent i = new Intent(Intent.ACTION_VIEW);  
        i.setData(Uri.parse(htmlCode));  
        startActivity(i);  
    }  
});
```

The **AdView()** function is the entry point of the Android SDK. When calling this function it will create a new object for the AdView class and access the methods of AdView class. Also, Pass the app **context** and **webview** in the AdView class.

HTML Ad

Configure the HTML ad using Java, XML, or a mixture of the two.

Code Snippets

To show the HTML ads inside the application, by implementing the interface in the java class. It contains two methods,

```
public class HTML extends AppCompatActivity
```

Xml Format:

To show the HTML ads center of the content OR middle of the content means must include the below XML code for showing the banner ads.

```
<WebView  
    android:id="@+id/banner2"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    tools:ignore="WebViewLayout" />
```

Code Format :

Below the code set for integration of HTML ads inside the content.

```
WebView webView = findViewById(R.id.banner1);
```

Explanation about the integration steps:

```
new HTML_ADS().loadHTMLAd(this,webView);
```

The **AdView()** function is the entry point of the Android SDK. When calling this function it will create a new object for the AdView class and access the methods of AdView class. Also, Pass the app **context** and **webview** in the AdView class.

HTML5 Ad

Configure the HTML5 ad using Java, XML, or a mixture of the two.

Code Snippets

To show the HTML5 ads inside the application, by implementing the interface in the java class. It contains two methods,

```
public class HTML5 extends AppCompatActivity
```

Xml Format:

To show the html5 ads inside the XML wherever we the content means must include the XML code for showing the banner ads.

```
<WebView  
    android:id="@+id/banner2"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    tools:ignore="WebViewLayout" />
```

Code Format :

Below the code set for integration of HTML5 ads inside the content.

```
WebView webView = findViewById(R.id.banner1);
```

Explanation about the integration steps:

```
new HTML_5_ADS().loadHTML5Ads(this,webView);
```

The **AdView()** function is the entry point of the Android SDK. When calling this function it will create a new object for the AdView class and access the methods of AdView class. Also, Pass the app **context** and **webview** in the AdView class.

PopUp Ad

Configure the PopUp ad using Java, XML, or a mixture of the two.

Code Snippets

To show the PopUp ads inside the application, by implementing the listener/interface in the java class. It contains two methods,

```
public class GIFPopUpActivity extends AppCompatActivity implements GIFAdListener
```

Code Format :

Below the code set for integration of PopUp inside the content.

```
WebView webView = findViewById(R.id.banner2);
```

Explanation about the integration steps:

```
new LoadGIFAdview().loadGIFAd(GIFPopUpActivity.this,GIFPopUpActivity.this);
```

The **AdView()** function is the entry point of the Android SDK. When calling this function it will create a new object for the AdView class and access the methods of AdView class. Also, Pass the app **context** and **webview** in the AdView class.

Interstitial Image Ad & Interstitial Video ad

Configure the interstitial image using Java. It is used to bring in the Interstitial ads to the app. It can be placed anywhere in the app like a normal Interstitial ad. This ad will appear on the whole screen with a close button.

Interstitial Image Code Format:

```
new Interstitial_image().loadInterstitial(InterstitialActivity.this,InterstitialActivity.this);
```

Interstitial Video Code Format:

```
new InterstitialVideo().loadAdShow(HomeActivity.this);
```

Interface is common for Interstitial Video and Interstitial Image ads :
Implement the Interface in the Class

```
public class HomeActivity extends AppCompatActivity implements InterstitialImageAdListener
```

Implement methods:

```
public interface InterstitialImageAdListener {  
    void onInterstitialAdLoaded();  
    void onInterstitialAdFailed();  
    void onInterstitialAdShown();  
    void onInterstitialAdClicked();  
    void onInterstitialAdDismissed();  
}
```

Rewarded Video Ad

Rewarded video advertising is a format that gives the reward to the user for spending a time, for viewing a full-screen ad. Rewarded videos are 15-30 seconds length and cannot be skipped.

Rewarded ads are a great way to keep the users engaged in the app while earning rewards. The reward generally comes in the form of game currency (gold, coins, power-ups, etc.) and it is given to the users after a successful ad completion.

Code Format:

```
new Rewardedvideo().loadRewardedAd(RewardActivity.this,RewardActivity.this);
```

Implementing Listener

```

public class "ActivityName" implements RewardedAdListener{
// This Listener needs to be added in the class where you want to call the interstitial Ad to get
// various callbacks as mentioned below
@Override
public void Rewarded(String rewardItem, int rewardvalue) { //we got the rewarded value }
@Override
public void AdLoaded() { //called once ad loaded }
@Override
public void AdFailed() { //called once ad loading failed }
@Override
public void AdClosed() { //called once ad closed by user }
@Override
public void AdClicked() { //called once ad clicked by user}

```

InArticle Ads

Configure the InArticle Video ad view using Java. This ad show the ad in between the content.

Code Format:

```

new InArticleVideoAds().loadRadioAds(playerView, InArticleVideo.this);

```

Xml Format:

```

<com.google.android.exoplayer2.ui.StyledPlayerView
    android:id="@+id/player_view"
    android:layout_width="match_parent"
    android:layout_height="200dp"
    android:visibility="gone"
    app:use_controller="false" />

```