

# **ANDROID AD FORMATS INTEGRATION GUIDE**

November 2023

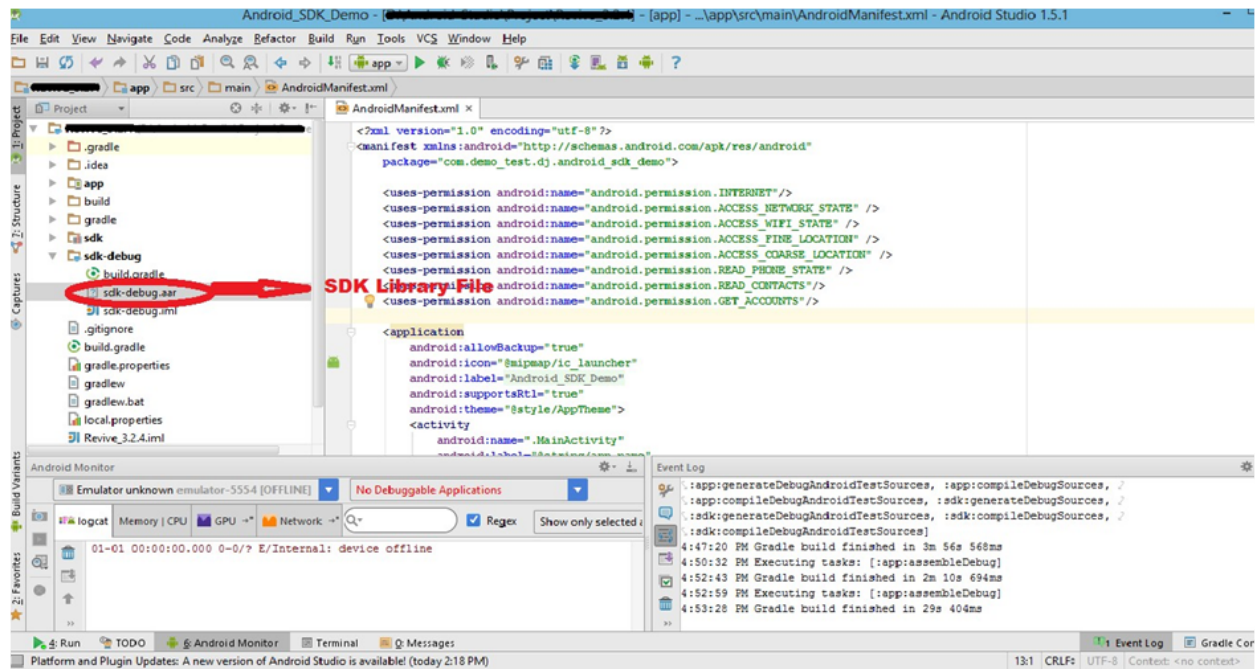
Integrating the SDK into Android Studio manually.

## 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 13

API Levels Supported : 21 to 34

## 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 'io.reactivex.rxjava2:rxandroid:2.1.1'  
implementation 'io.reactivex.rxjava2:rxjava:2.2.6'  
implementation 'com.google.android.exoplayer:exoplayer-core:2.19.1'  
implementation 'com.google.android.exoplayer:exoplayer-ui:2.19.1'  
implementation 'com.google.android.exoplayer:extension-ima:2.19.1'  
implementation 'com.google.android.gms:play-services-ads:22.4.0'  
implementation 'com.google.code.gson:gson:2.10.1'  
implementation 'com.github.bumptech.glide:glide:4.16.0'  
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 "publisher ID". A publisher ID is used by MSDK to identify a context within the app where

advertisements can be shown. App is needed to obtain a publisher ID from the MSDK representative or is needed to the ad network. Without it, app can't be able to fetch and display ads.

Ad unit Name	Fixed Size
Image ad, Top banner ad	320 x 50
Interstitial Image	320 x 480

For the Image ad and Interstitial image need to upload the image in above sizes only.

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

```
new AdsInitialize(MainActivity.this, "Enter the publisher id", new AdViewListener() {  
    @Override  
    public void onInitializationComplete() {  
  
    }  
  
    @Override  
    public void onInitializationFailure() {  
  
    }  
});
```

## Banner Image Ad

Configure the banner ad using Java.

### 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.

### Implementing Listener :

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

```
@Override  
public void onAdsAdLoaded() {  
}  
  
@Override  
public void onAdsAdFailed() {  
}
```

### Code Format :

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

```
BannerImageAD.show(BannerImageAd.this, BannerPosition.BOTTOM, new BannerListener() {  
    @Override  
    public void onAdsAdLoaded() {  
  
    }  
  
    @Override  
    public void onAdsAdFailed() {  
  
    }  
});
```

## Top Banner Ad

Configure the top banner ad using Java.

### 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.

### Implementing Listener :

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

```
@Override  
public void onAdsAdLoaded() {  
}  
  
@Override  
public void onAdsAdFailed() {  
}
```

### Code Format :

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

```
TopBannerAD.show(TopBanner.this, new BannerListener() {  
    @Override  
    public void onAdsAdLoaded() {  
  
    }  
  
    @Override  
    public void onAdsAdFailed() {  
  
    }  
});
```

## Direct Link Ad

Configure the direct link ad using Java.

### 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 banner ads. It contains two methods.

### Implementing Listener :

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

```
@Override
public void onAdsAdLoaded() {
}
@Override
public void onAdsAdFailed() {
}
```

### Code Format :

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

```
DirectLinkAd.show(Direct_link_ad.this, "Go To Browser", new BannerListener() {
    @Override
    public void onAdsAdLoaded() {

    }

    @Override
    public void onAdsAdFailed() {

    }
});
```

## HTML Ad

Configure the HTML ad using Java.

### Code Snippets

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

### Implementing Listener :

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

```
@Override
public void onAdsAdLoaded() {
}

@Override
public void onAdsAdFailed() {
}
```

### Code Format :

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

```
HTMLAD.show(HTML_Ads.this, new BannerListener() {
    @Override
    public void onAdsAdLoaded() {

    }

    @Override
    public void onAdsAdFailed() {

    }
});
```

## HTML5 Ad

Configure the HTML5 ad using Java.

### Code Snippets

To show the html 5 ads inside the application, by implementing the listener/interface in the java class. **BannerListener** is the listener for banner ads. It contains two methods.

### Implementing Listener :

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

```
@Override
public void onAdsAdLoaded() {
}

@Override
public void onAdsAdFailed() {
}
```

### Code Format :

Below the code set for integration of HTML 5 ads inside the content.

```
HTML_5_Ad.show(HTML_5_Ads.this, new BannerListener() {  
    @Override  
    public void onAdsAdLoaded() {  
  
    }  
  
    @Override  
    public void onAdsAdFailed() {  
  
    }  
});
```

## PopUp Ad

Configure the PopUp ad using Java.

### Code Snippets

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

### Implementing Listener :

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

```
@Override  
public void onAdsShowFailure() {  
}  
  
@Override  
public void onAdsShowStart() {  
}  
  
@Override  
public void onAdsShowClick() {  
}  
  
@Override  
public void onAdsShowComplete() {
```



```
}  
  
@Override  
public void onAdsDismissed() {  
}
```

### Code Format :

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

```
PopupAd.load(PopUpAd.this, new PopupAdListener() {  
    @Override  
    public void onAdsShowFailure() {  
  
    }  
  
    @Override  
    public void onAdsShowStart() {  
  
    }  
  
    @Override  
    public void onAdsShowClick() {  
  
    }  
  
    @Override  
    public void onAdsShowComplete() {  
  
    }  
  
    @Override  
    public void onAdsDismissed() {  
  
    }  
});
```

## Interstitial Image Ad & Interstitial Video ad

Configure the interstitial image and video 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. There are 2 steps to show the Interstitial ads :

## 1. Interstitial Load Code Format:

```
Interstitial.load(InterstitialActivity.this, new InterstitialLoadAdListener() {  
    @Override  
    public void onAdsAdLoaded() {  
  
    }  
  
    @Override  
    public void onAdsAdFailed() {  
  
    }  
});
```

## 2. Interstitial Show Code Format:

```
Interstitial.show(InterstitialActivity.this, new InterstitialAdShowListener() {  
    @Override  
    public void onAdsShowFailure() {  
  
    }  
  
    @Override  
    public void onAdsShowStart() {  
  
    }  
  
    @Override  
    public void onAdsShowClick() {  
  
    }  
  
    @Override  
    public void onAdsShowComplete() {  
  
    }  
  
    @Override  
    public void onAdsDismissed() {  
  
    }  
});
```

## 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. There are 2 steps to show the Interstitial ads :

### 1. Rewarded Load Code Format:

```
RewardedVideo.load(RewardActivity.this, new RewardedAdLoadListener() {  
    @Override  
    public void onAdsAdLoaded() {  
  
    }  
  
    @Override  
    public void onAdsAdFailed() {  
  
    }  
});
```

### 2. Rewarded Show Code Format:

```
RewardedVideo.show(RewardActivity.this, new RewardedAdShowListener() {  
    @Override  
    public void onAdsShowFailure() {  
  
    }  
  
    @Override  
    public void onAdsShowStart() {  
  
    }  
  
    @Override  
    public void onAdsShowClicked() {  
  
    }  
  
    @Override  
    public void onAdsShowComplete() {  
  
    }  
  
    @Override  
    public void Rewarded(String rewardItem, int rewarded) {  
  
    }  
});
```

## InArticle Ads

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

### 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" />
```

### Code Format:

```
public StyledPlayerView playerView;
ExoPlayer player;
InArticleVideoAds inArticleVideoAds;

@Override
protected void onCreate(Bundle savedInstanceState) {
    playerView = findViewById(R.id.player_view);
    playerView.setPlayer(player);
    inArticleVideoAds = new InArticleVideoAds();
    inArticleVideoAds.loadRadiosAds(playerView, InArticleVideo.this);
}

@Override
public void onStart() {
    super.onStart();
    if (Util.SDK_INT > 23) {
        playerView.onResume();
    }
}

@Override
public void onResume() {
    super.onResume();
    if (Util.SDK_INT <= 23 || player == null) {
        inArticleVideoAds.resumePlayer();
    }
}
```

```

@Override
public void onPause() {
    super.onPause();
    if (Util.SDK_INT <= 23) {
        inArticleVideoAds.pausePlayer();
    }
}

@Override
public void onStop() {
    super.onStop();
    if (Util.SDK_INT > 23) {
        inArticleVideoAds.pausePlayer();
    }
}

@Override
protected void onDestroy() {
    Log.e("onDestroy", "AppSide Called.");
    inArticleVideoAds.stopPlayer();
    super.onDestroy();
}

```

## Bottom Slider Ad

Configure the Bottom Slider ad using Java.

### Code Snippets

To show the bottom slider ads inside the application, by implementing the listener/interface in the java class. **BottomSlider** is the listener for bottom slider ads. It contains two methods.

### Implementing Listener :

```

@Override
public void onAdsAdLoaded() {

}

@Override
public void onAdsAdFailed() {

```

```
}

@Override
public void onAdsAdShown() {

}

@Override
public void onAdsAdClicked() {

}

@Override
public void onAdsAdDismissed() {

}
```

### Code Format :

Below the code set for integration of Bottom slider ads inside the content.

```
BottomSliderAd.show(BottomSlider.this, new BottomSliderAdListener() {
    @Override
    public void onAdsAdLoaded() {

    }

    @Override
    public void onAdsAdFailed() {

    }

    @Override
    public void onAdsAdShown() {

    }

    @Override
    public void onAdsAdClicked() {

    }

    @Override
    public void onAdsAdDismissed() {

    }
});
```