Android SDK

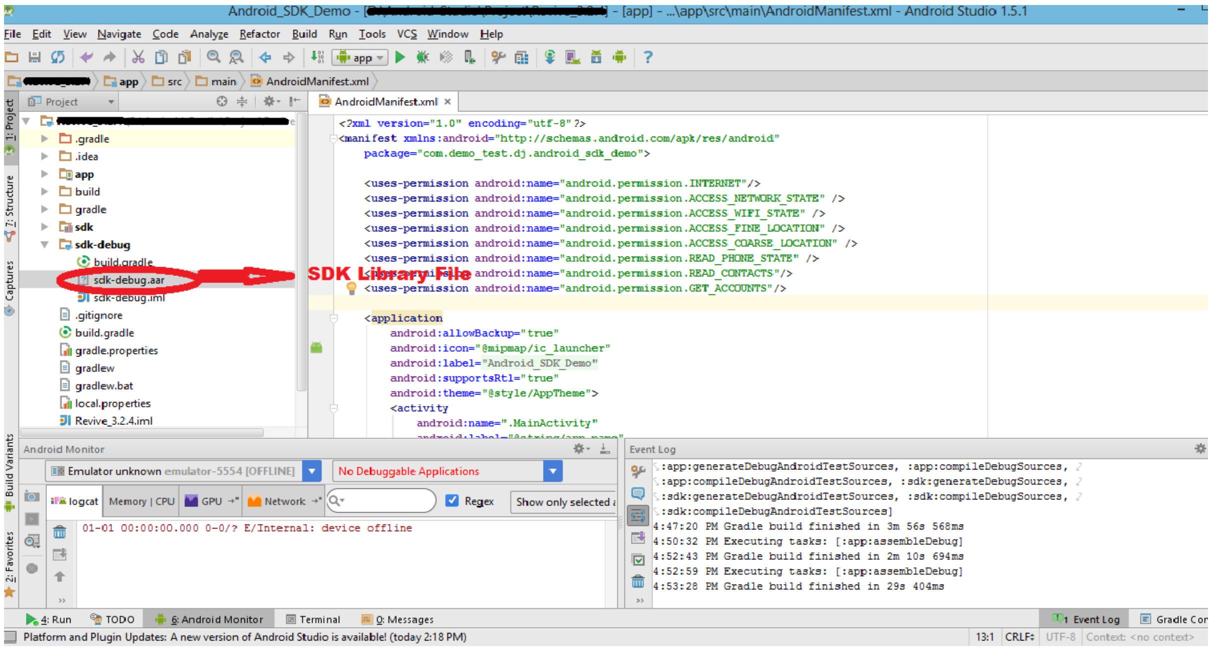
Manual includesthe integration of SDK using eclipse and android studio.

Android Studio User:

Step 1: Extract the zip file and use .aar file for integration

Step 2: Import .aar file into your workspace and add the dependency to your application

Step 3:Follow the below code structure for integration



Android_SDK_Demo - [redacted] - [app] - ...\app\src\main\AndroidManifest.xml - Android Studio 1.5.1

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.demo_test.dj.android_sdk_demo">

    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_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.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.READ_CONTACTS"/>
    <uses-permission android:name="android.permission.GET_ACCOUNTS"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Android_SDK_Demo"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
```

Android Monitor
Emulator unknown emulator-5554 [OFFLINE]
No Debuggable Applications
logcat
Memory | CPU
GPU
Network
Regex
Show only selected

```
01-01 00:00:00.000 0-0/? E/Internal: device offline
```

```
:app:generateDebugAndroidTestSources, :app:compileDebugSources,
:app:compileDebugAndroidTestSources, :sdk:generateDebugSources,
:sdk:generateDebugAndroidTestSources, :sdk:compileDebugSources,
:sdk:compileDebugAndroidTestSources]
4:47:20 PM Gradle build finished in 3m 56s 568ms
4:50:32 PM Executing tasks: [:app:assembleDebug]
4:52:43 PM Gradle build finished in 2m 10s 694ms
4:52:59 PM Executing tasks: [:app:assembleDebug]
4:53:28 PM Gradle build finished in 29s 404ms
```

Platform and Plugin Updates: A new version of Android Studio is available! (today 2:18 PM)

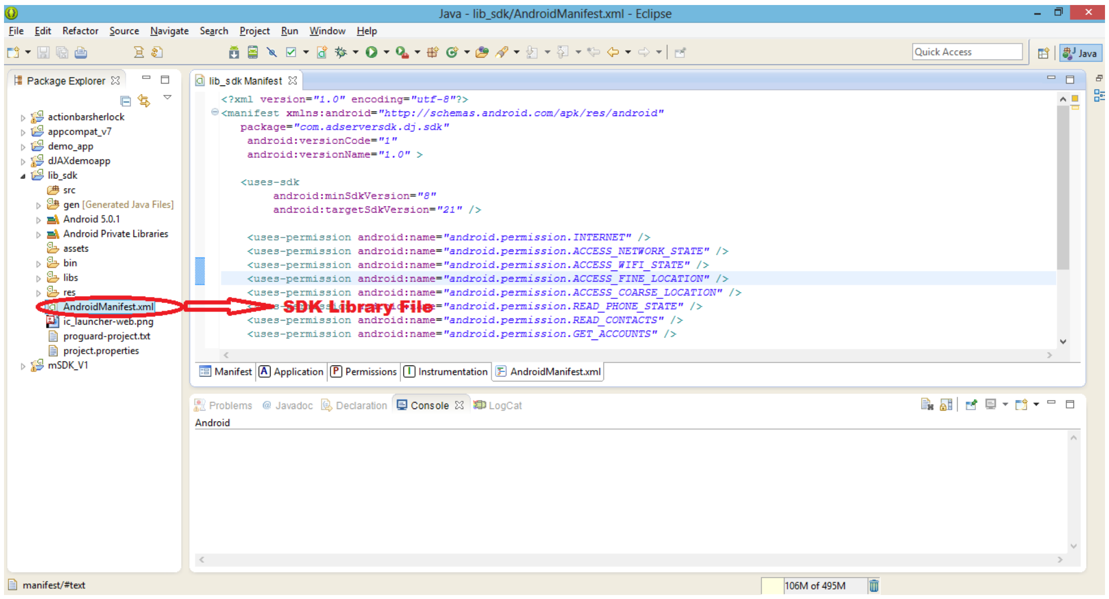
Fig: Android Studio With Library File

Eclipse User:

Step 1: Extract the zip file and use dJAX_Lib library file for integration

Step 2: Import dJAX_Lib library file into your workspace and use this library file foryour application

Step 3: Follow the below code structure for integration



```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.adserversdk.dj.sdk"
    android:versionCode="1"
    android:versionName="1.0" >

   <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="21" />

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_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.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.READ_CONTACTS" />
    <uses-permission android:name="android.permission.GET_ACCOUNTS" />
```

Fig: EclipseWith Library File

Code Structure:

Edit App Permissions

Edit your Android manifest to include the permissions needed by this app:

```xml
<uses-permission  android:name="android.permission.INTERNET"/>
 <uses-permission  android:name="android.permission.ACCESS_NETWORK_STATE"/>
 <uses-permission  android:name="android.permission.ACCESS_WIFI_STATE" />
 <uses-permission  android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
 <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
 <uses-permission android:name="android.permission.READ_PHONE_STATE" />
 <uses-permission android:name="android.permission.READ_CONTACTS"/>
 <uses-permission android:name="android.permission.GET_ACCOUNTS"/>
```

- INTERNET (required) - Grants the SDK permission to access the internet.
- ACCESS_NETWORK_STATE (required) - Grants the SDK permission to check fora live internet connection.
- ACCESS_WIFI_STATE (required) - Grants the SDK permission to accessinformation about Wi-Fi networks.
- ACCESS_FINE_LOCATION (Optional) - Grants the SDK permission toaccess a more accurate location based on GPS.
- ACCESS_COARSE_LOCATION (Optional) - Grants the SDK permissionto access approximate location based on cell tower.
- READ_PHONE_STATE (Optional) - Grants the SDK permission to readonly access to phone state.
- READ_CONTACTS (Optional) - Grants the SDK permission to read theuser's contacts data.
- GET_ACCOUNTS (Optional) - Grants the SDK permission to the list ofaccounts in the Accounts Service.

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. Please note that the SDK will never wake up the phone to request the location to be updated; this would take time and battery. Instead, it will use these permissions to access the last known location of the device.

Show Ads

This section describes some of the code you'll write in order to show ads.

This document refers to something called a "zone ID". A zone ID is just a numeric ID used by MSDK to identify a context within an app where advertisements can be shown. You'll need to obtain a zone ID from your MSDK representative or your ad network. Without it, you won't be able to fetch and display ads.

Banners

You can configure your banner ad view using Java, XML, or a mixture of the two. The table below lists the XML and Java equivalents.

XML	**Java Equivalent**	**Description**	**Example**
msdk:zone_id	ad.setZoneid()	The zone ID associated with your app's inventory. **You 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 autorefresh is enabled. The minimum period is 15 seconds. The default period is 30 seconds. Set this to 0 to disable autorefresh.	**"30000"**

If you're using XML, you'll need to add the xmlns:msdk namespace attribute describing your application to your layout tag; for example this might be a RelativeLayout, LinearLayout, or FrameLayout and ScrollView.

```
xmlns:msdk="http://schemas.android.com/apk/res-auto"
```

Sample code :

Xml Format:

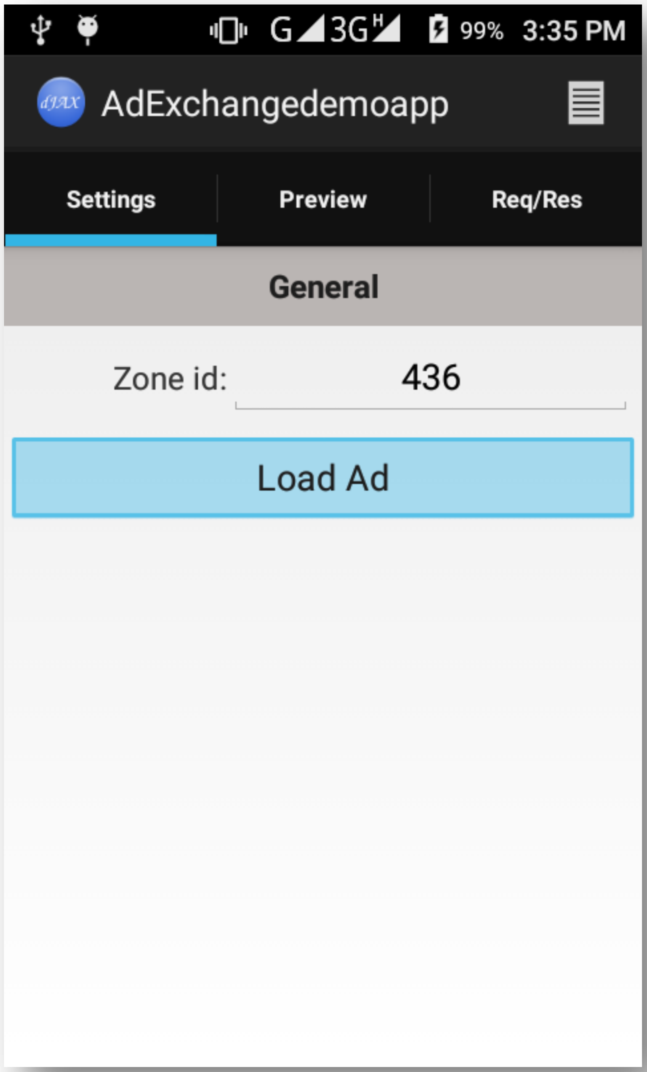
```
<LinearLayout
android:orientation="vertical"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_centerHorizontal="true"
android:id="@+id/preview_flag">
</LinearLayout>
```

Code Format:

```
LinearLayout.LayoutParamsparams = newLinearLayout.LayoutParams(LinearLayout.
LayoutParams.MATCH_PARENT,LinearLayout.LayoutParams.WRAP_CONTENT);
 LinearLayoutadFrame = (LinearLayout) findViewById(R.id.preview_flag);
AdViewad = new AdView(this);
ad.setZoneid("436");//Place Your Zone id
ad.setAuto_refresh_time(30000);//Interval in Milliseconds
/*Loads an Ad*/
ad.LoadAd();
ad.setLayoutParams(params);
adFrame.addView(ad);
adFrame.bringToFront();
```

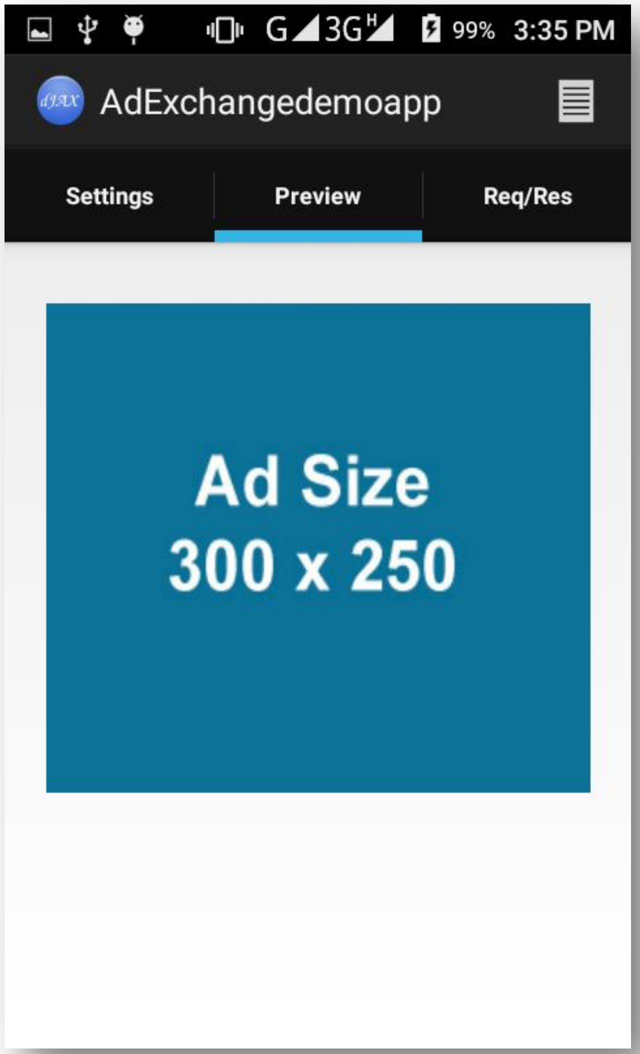
dJAXdemoapp Procedure:

1. Enter the zone id then click load ad button.

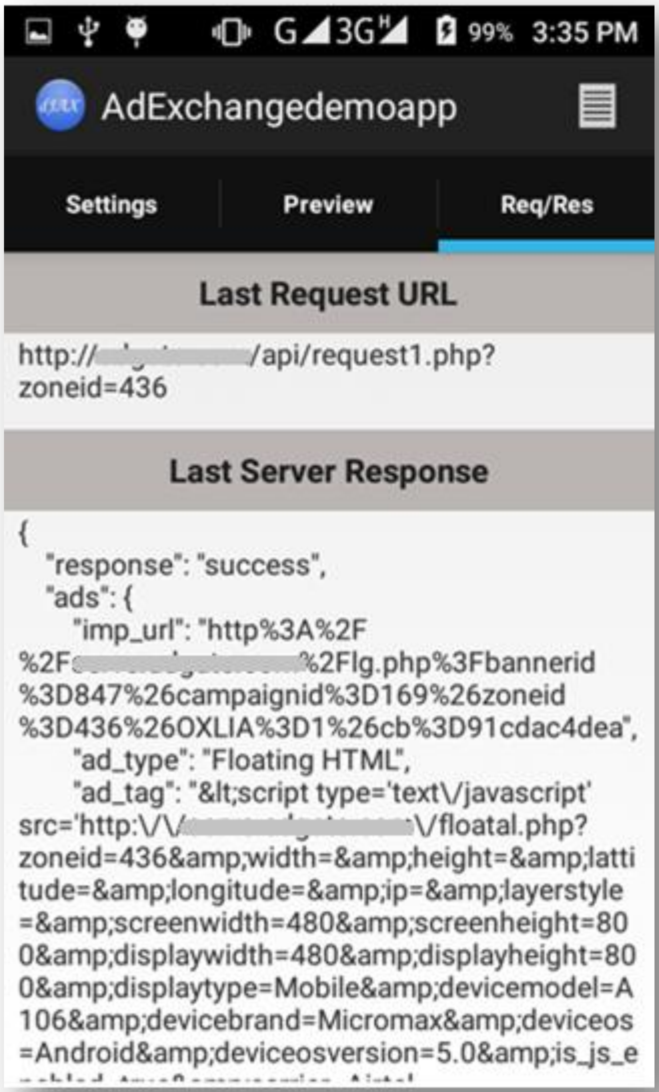


Zone id: 436

2. The requested banner is loaded in the preview tab as shown below.



3. You can view the request and response in the request tab.



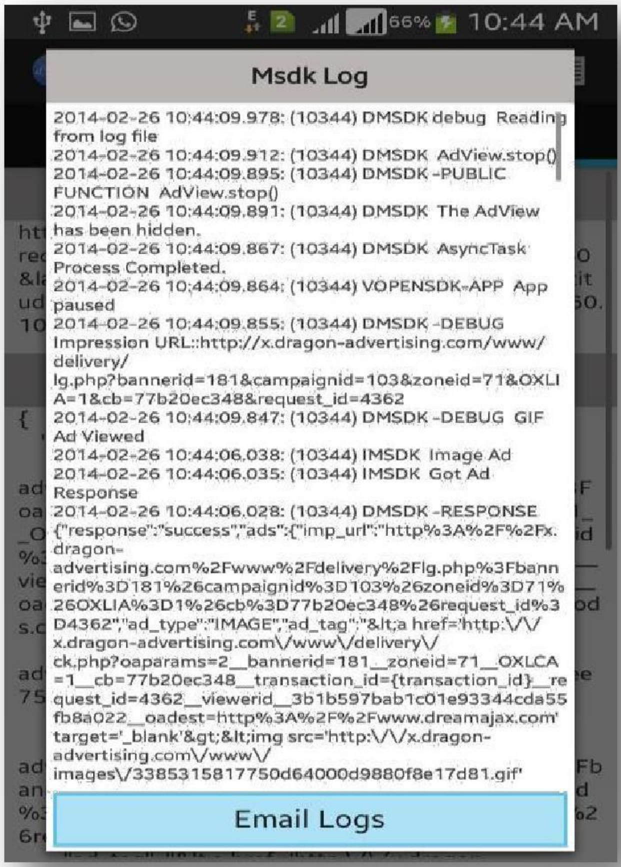
Last Request URL

http://[redacted]/api/request1.php?zoneid=436

Last Server Response

```
{
   "response": "success",
   "ads": {
      "imp_url": "http%3A%2F
%2F[redacted]%2Flg.php%3Fbannerid
%3D847%26campaignid%3D169%26zoneid
%3D436%26OXLIA%3D1%26cb%3D91cdac4dea",
      "ad_type": "Floating HTML",
      "ad_tag": "&lt;script type='text\/javascript'
src='http:\/\/[redacted]\/floatal.php?
zoneid=436&amp;width=&amp;height=&amp;latti
tude=&amp;longitude=&amp;ip=&amp;layerstyle
=&amp;screenwidth=480&amp;screenheight=80
0&amp;displaywidth=480&amp;displayheight=80
0&amp;displaytype=Mobile&amp;devicemodel=A
106&amp;devicebrand=Micromax&amp;deviceos
=Android&amp;deviceosversion=5.0&amp;is_js_e
```

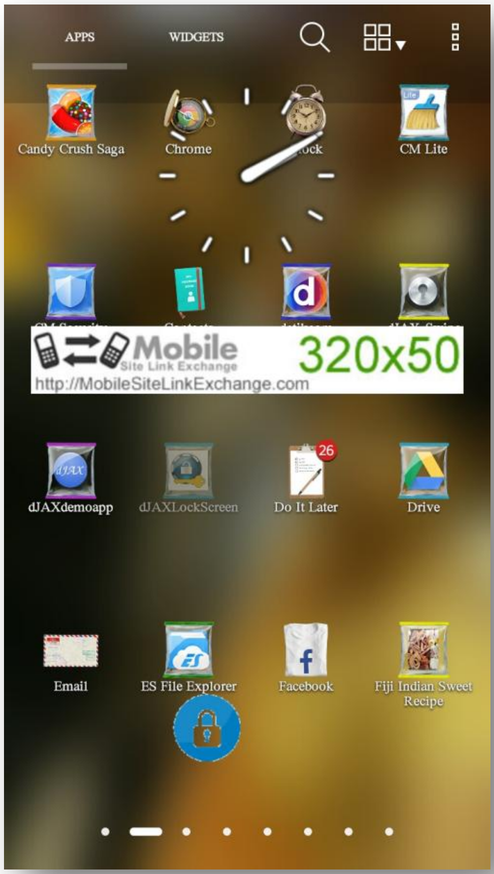
4. The Log of the app can be viewed by clicking on log icon in menu.



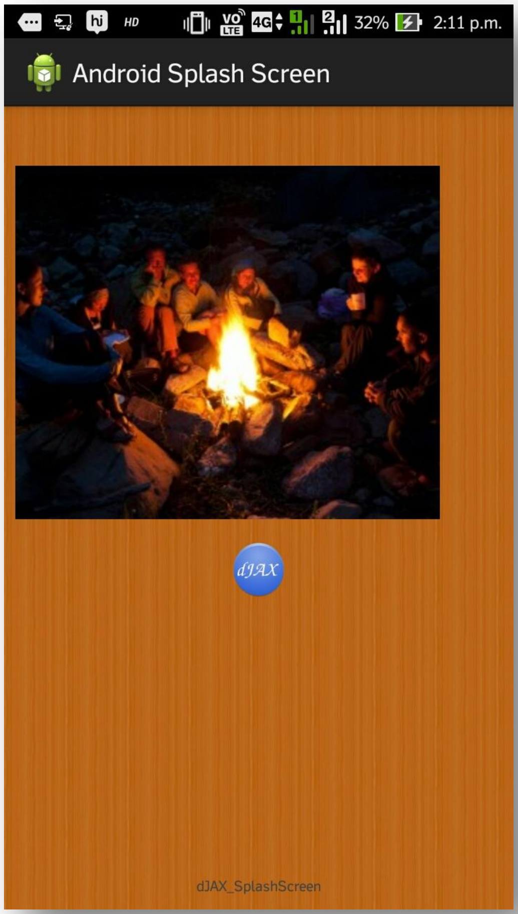
```
2014-02-26 10:44:09.978: (10344) DMSDK debug  Reading
from log file
2014-02-26 10:44:09.912: (10344) DMSDK  AdView.stop()
2014-02-26 10:44:09.895: (10344) DMSDK -PUBLIC
FUNCTION  AdView.stop()
2014-02-26 10:44:09.891: (10344) DMSDK  The AdView
has been hidden.
2014-02-26 10:44:09.867: (10344) DMSDK  AsyncTask
Process Completed.
2014-02-26 10:44:09.864: (10344) VOPENSDK-APP  App
paused
2014-02-26 10:44:09.855: (10344) DMSDK -DEBUG
Impression URL::http://x.dragon-advertising.com/www/
delivery/
lg.php?bannerid=181&campaignid=103&zoneid=71&OXLI
A=1&cb=77b20ec348&request_id=4362
2014-02-26 10:44:09.847: (10344) DMSDK -DEBUG  GIF
Ad Viewed
2014-02-26 10:44:06.038: (10344) IMSDK  Image Ad
2014-02-26 10:44:06.035: (10344) IMSDK  Got Ad
Response
2014-02-26 10:44:06.028: (10344) DMSDK -RESPONSE
{"response":"success","ads":{"imp_url":"http%3A%2F%2Fx.
dragon-
advertising.com%2Fwww%2Fdelivery%2Flg.php%3Fbann
erid%3D181%26campaignid%3D103%26zoneid%3D71%
26OXLIA%3D1%26cb%3D77b20ec348%26request_id%3
D4362","ad_type":"IMAGE","ad_tag":"&lt;a href='http:\/\/
x.dragon-advertising.com\/www\/delivery\/
ck.php?oaparams=2__bannerid=181__zoneid=71__OXLCA
=1__cb=77b20ec348__transaction_id={transaction_id}__re
quest_id=4362__viewerid__3b1b597bab1c01e93344cda55
fb8a022__oadest=http%3A%2F%2Fwww.dreamajax.com'
target='_blank'&gt;&lt;img src='http:\/\/x.dragon-
advertising.com\/www\/
images\/3385315817750d64000d9880f8e17d81.gif'
```

5. You can email the logs by clicking on the Email Logs button.

6. Delivering ad in Swipe Screen (Lock Screen), below is the appearance



7. Delivering ad in Splash Screen (Prestitial), below is the appearance



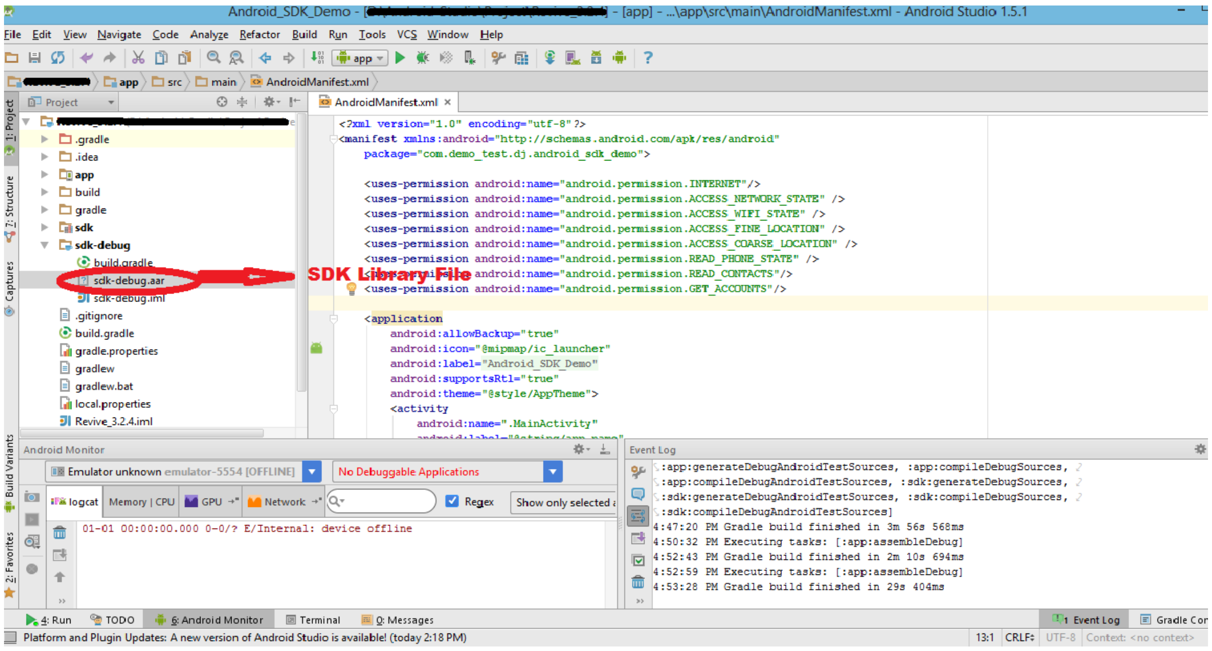
Interstitial Video Ad Format

Android Studio User:

Step 1: Extract the zip file and use .aar file for integration

Step 2: Import .aar file into your workspace and add the dependency to your application

Step 3: Follow the below code structure for integration



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.demo_test.dj.android_sdk_demo">

    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_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.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.READ_CONTACTS"/>
    <uses-permission android:name="android.permission.GET_ACCOUNTS"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Android_SDK_Demo"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
```

```
01-01 00:00:00.000 0-0/? E/Internal: device offline
```

```
:app:generateDebugAndroidTestSources, :app:compileDebugSources,
:app:compileDebugAndroidTestSources, :sdk:generateDebugSources,
:sdk:generateDebugAndroidTestSources, :sdk:compileDebugSources,
:sdk:compileDebugAndroidTestSources]
4:47:20 PM Gradle build finished in 3m 56s 568ms
4:50:32 PM Executing tasks: [:app:assembleDebug]
4:52:43 PM Gradle build finished in 2m 10s 694ms
4:52:59 PM Executing tasks: [:app:assembleDebug]
4:53:28 PM Gradle build finished in 29s 404ms
```

Platform and Plugin Updates: A new version of Android Studio is available! (today 2:18 PM)

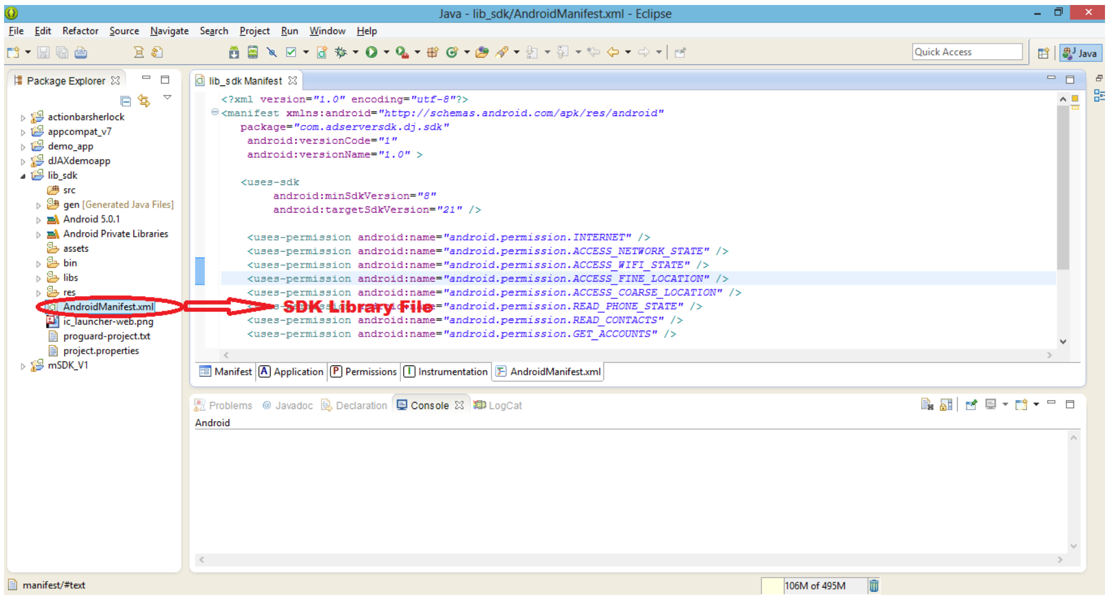
Fig: Android Studio With Library File

Eclipse User:

Step 1: Extract the zip file and use dJAX_Lib library file for integration

Step 2: Import dJAX_Lib library file into your workspace and use this library file for your application

Step 3: Follow the below code structure for integration



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.adserversdk.dj.sdk"
    android:versionCode="1"
    android:versionName="1.0" >

   <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="21" />

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_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.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.READ_CONTACTS" />
    <uses-permission android:name="android.permission.GET_ACCOUNTS" />
```

Fig: EclipseWith Library File

Code Structure:

Edit App Permissions

Edit your Android manifest to include the permissions needed by this app:

```
<uses-permission  android:name="android.permission.INTERNET"/>
 <uses-permission  android:name="android.permission.ACCESS_NETWORK_STATE"/>
 <uses-permission  android:name="android.permission.ACCESS_WIFI_STATE" />
 <uses-permission  android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
 <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
 <uses-permission android:name="android.permission.READ_PHONE_STATE" />
 <uses-permission android:name="android.permission.READ_CONTACTS"/>
 <uses-permission android:name="android.permission.GET_ACCOUNTS"/>
```

```
  <application android:label="@string/app_name"
    android:icon="@drawable/ic_launcher_djax"
     android:theme="@style/Theme.Example">
    <activity android:name=".MainActivity" android:label="@string/app_name">
       <intent-filter>
                    <action android:name="android.intent.action.MAIN" />
                  <category android:name="android.intent.category.LAUNCHER" />
       </intent-filter>
    </activity>
     <activity   android:name=" com.adserversdk.dj.sdk.adserver.AdScreenLayout" >
   </activity>
  </application>
</manifest>
```

- ACTIVITY (required) – To access the activity of interstitial video ad format
- ANDROID:NAME="COM.ADSERVERSDK.DJ.SDK.ADSERVER.ADSCREENLAYOUT" (required) – This activity will invokes the interstitial ad file for app interstitial video ad function

Show Ads

This section describes some of the code you'll write in order to show ads. The code format which is given below must be included in your application MainActivity.java file.

The code must be used inside any Main Function () and the zone_id is the keyword, you have to replace the zone_id value instead "52" you can include as you needed. This ad will get displayed at application exit page

Sample code:

Code Format:

```
@Override
    public void onBackPressed() {
      Intent i = new Intent(getApplicationContext(),AdScreenLayout.class);
      i.putExtra("zone_id", "52");//Enter zone_id value replace "52"
      startActivity(i);
      finish();
    }
```

After Installation:

When you Press the interstitial view ad button of dJAXdemoapp or when you attempt to deliver your application you can view the interstitial video ad. If you need the background to be changed as black you can add it in ad_interstitial file for the best appearance.