AdWhirl Open Source SDK Setup Instructions for Android

The AdWhirl Open Source SDK contains the code for your Android application to display ads from different ad networks. The instructions below assume that you are familiar with Android application development and understand how to change project settings in Eclipse. While Android does not have any specific requirements on IDE, this document uses Eclipse with Android Developer Tools (ADT) plug-in.

Setting up the AdWhirl SDK

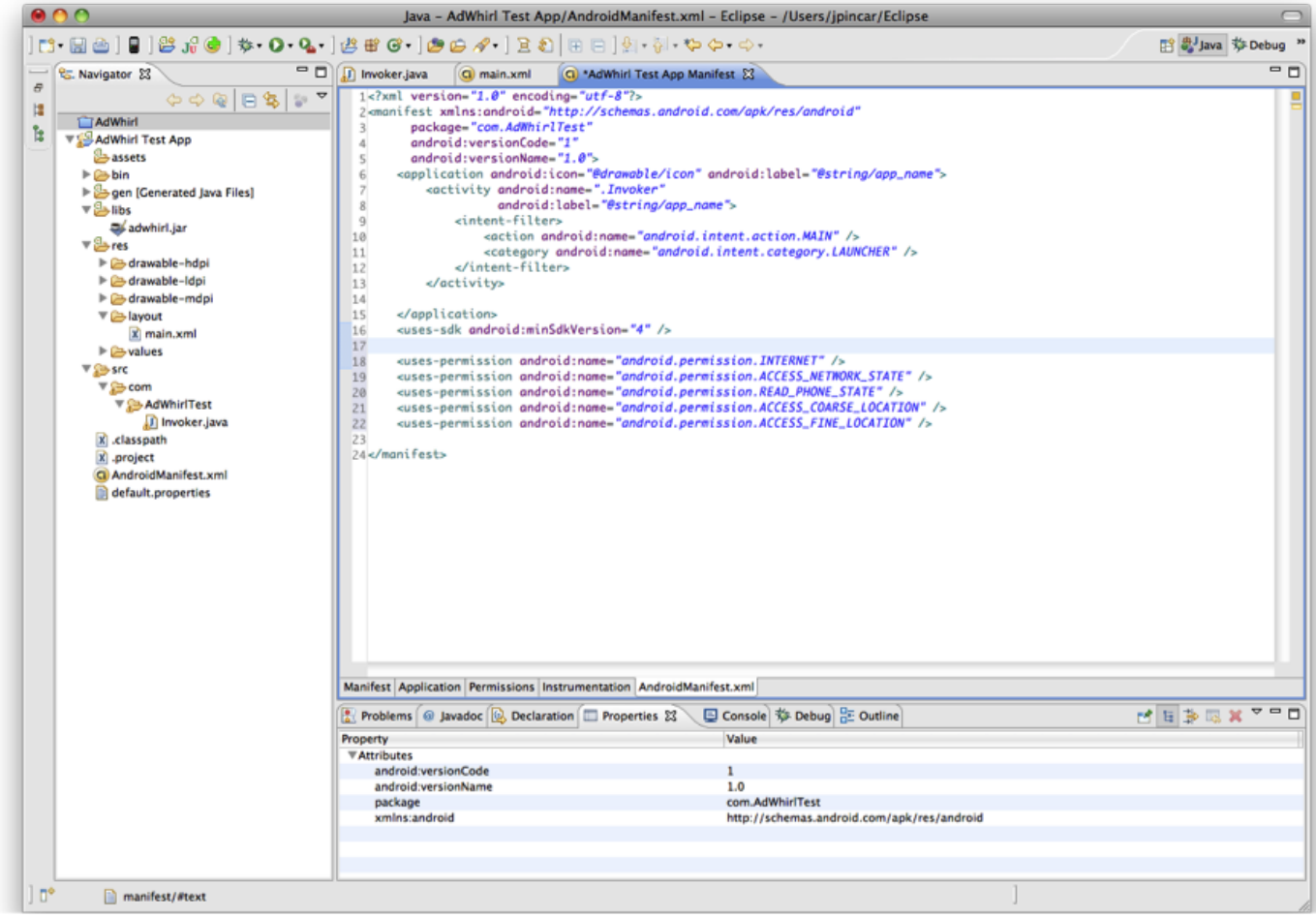
1. Add proper application permissions to AndroidManifest.xml.

Required:

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

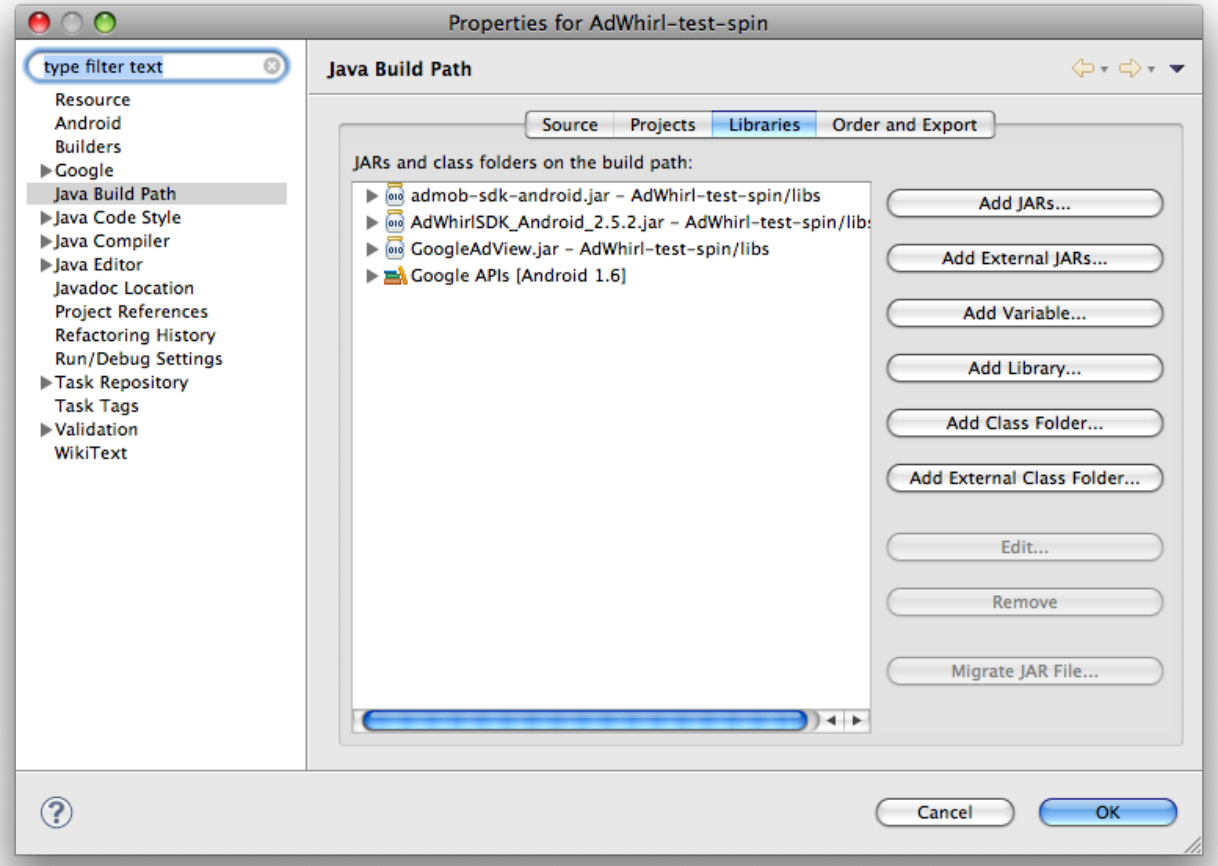
Optional:

```
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```



2. Copy the AdWhirl JAR into your `/libs` directory and add it to your Java build path.

Note: this screen shot contains configurations for both AdMob SDK and AdSense for Mobile Applications SDK, as described in Step #3.



3. Add the libraries of those ad networks you want to support. You do not have to integrate an ad network if you do not want to run their ads, or do not have access to their SDK. On Android, AdWhirl currently provides built-in support for AdMob, AFMA, Millennial Media, Quattro, and ZestADZ.

AdMob

1. Make sure the AdMob publisher id is set up on the AdWhirl network settings console.
2. Download the AdMob SDK from http://www.admob.com/.
3. Copy the AdMob SDK JAR file into your `/libs` directory and add it to your Java build path.

Google AdSense for Mobile Applications (AFMA)

1. Make sure the AFMA publisher id is set up on the AdWhirl network settings console.
2. Download the AFMA Android SDK per Google's instructions.
3. Copy the AFMA SDK JAR file into your `/libs` directory and add it to your Java build path.

If you enable and allocate a percentage to the AdSense network, you must pass in a set of parameters required by AdSense. In particular, you **must** specify a company name and app name, and you may optionally provide an AdSense channel ID and an expand direction (either "BOTTOM" or "TOP"). By providing an expand direction, you enable AFMA to return expandable ads to your app in addition to the standard text and image formats.

NOTE: If you **do not** specify both a company name and an app name, you will **not** see any ads returned from the AdSense network, so make sure you configure your app appropriately.

Parameters are set using static methods in the `AdWhirlAdapter` class, as in the following example.

```
// Required Google AdSense network parameters
AdWhirlAdapter.setGoogleAdSenseCompanyName(COMPANY_NAME);
AdWhirlAdapter.setGoogleAdSenseAppName(APP_NAME);

// Optional Google AdSense network parameters
AdWhirlAdapter.setGoogleAdSenseChannel(CHANNEL_ID);
AdWhirlAdapter.setGoogleAdSenseExpandDirection("BOTTOM");
```

This snippet should directly precede the code that instantiates the AdWhirlLayout object and adds it to your Android app's layout as discussed in the next section.

Millennial Media

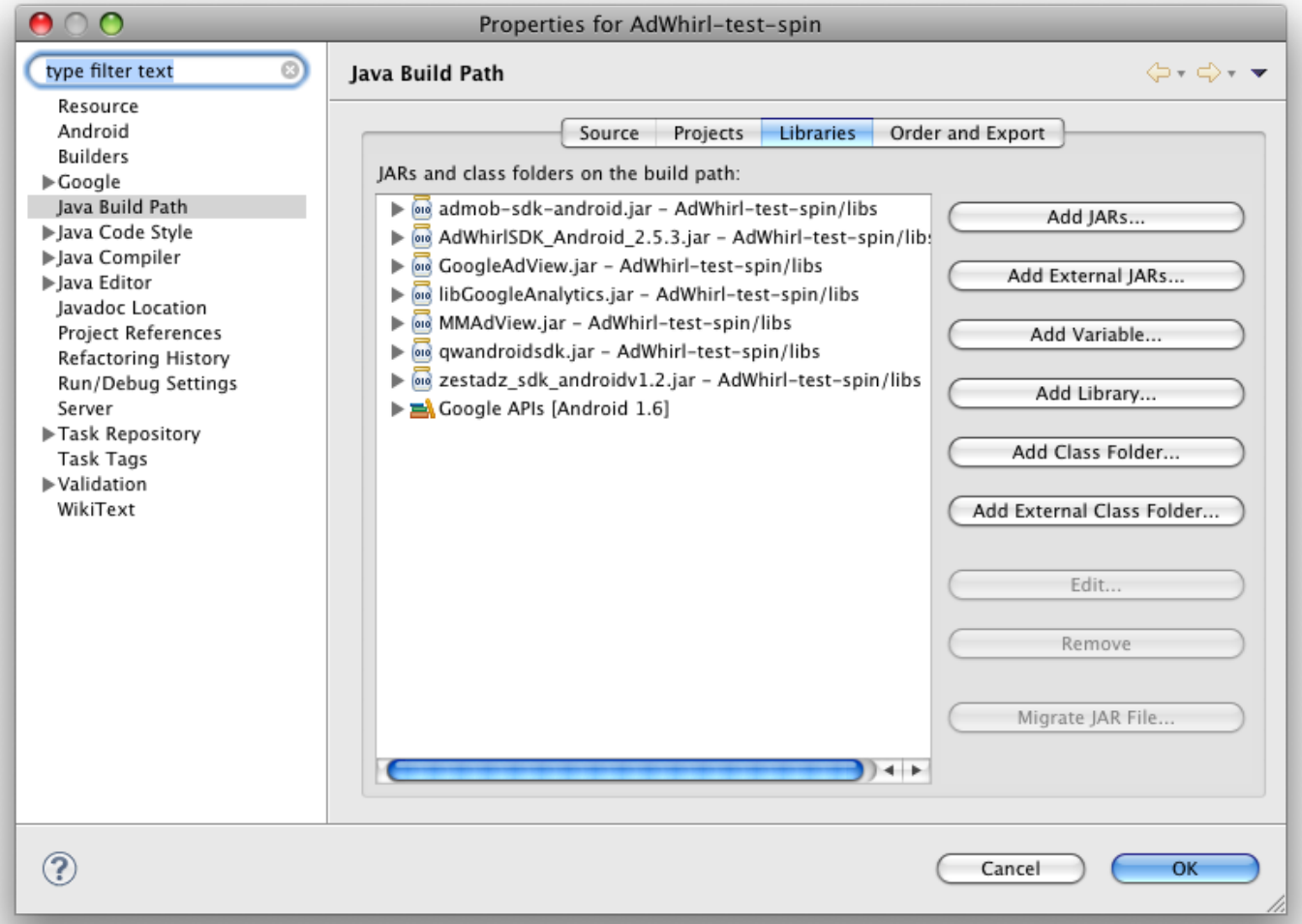
1. Make sure the Millennial Media key is set up on the AdWhirl network settings console.
2. Download the Millennial Media SDK from http://developer.millennialmedia.com/
3. Copy the Millennial Media SDK JAR file into your `/libs` directory and add it to your Java build path.

Quattro

1. Make sure the Quattro site id and publisher id are set up on the AdWhirl network settings console.
2. Download the Quattro SDK from http://www.quattrowireless.com/
3. Copy the Quattro SDK JAR file into your `/libs` directory and add it to your Java build path.

ZestADZ

1. Make sure the ZestADZ site id is set up on the AdWhirl network settings console.
2. Download the ZestADZ SDK from http://www.zestadz.com/
3. Copy the ZestADZ SDK JAR file into your `/libs` directory and add it to your Java build path.



4. In the Activity where you intend to add the ad:
    - Import the `com.adwhirl.AdWhirlLayout` package
    - Add an AdWhirl layout with a code snippet similar to below.

```
LinearLayout layout = (LinearLayout) findViewById(R.id.layout_main);
AdWhirlLayout adWhirlLayout = new AdWhirlLayout(this, "AdWhirl SDK Key");
RelativeLayout.LayoutParams adWhirlLayoutParams =
    new RelativeLayout.LayoutParams(
        LayoutParams.FILL_PARENT,
        LayoutParams.WRAP_CONTENT);
layout.addView(adWhirlLayout, adWhirlLayoutParams);
layout.invalidate();
```

- Per Android best practice, avoid using absolute sizes and layouts in order to optimize for devices with different screen sizes and densities. For more information on this subject, visit: http://developer.android.com/guide/practices/screens_support.html
  If developers must specify layout dimensions, they should leverage density-independent pixel (dip). Code snippet below illustrates how to scale dip units to physical pixels:

```
...
final int DIP_WIDTH = 320;
final int DIP_HEIGHT = 52;
final float DENSITY = getResources().getDisplayMetrics().density;
int scaledWidth = (int) (DENSITY * DIP_WIDTH + 0.5f);
int scaledHeight = (int) (DENSITY * DIP_HEIGHT + 0.5f);
RelativeLayout.LayoutParams adWhirlLayoutParams =
    new RelativeLayout.LayoutParams(scaledWidth, scaledHeight);
...
```