# Flickr Browser Changes For Android 1.4

## Introduction

Eventually I will re-create the Flickr videos completely for Android Marshmallow. For now, this document is what you need to do to get the app working with Android Marshmallow.

## Extra information

I have created two videos which might be of interest to you. You can watch this, or just follow the information in this guide (note that the videos don't contain everything int his document, so consider the document to be the most up to date).

## What has changed in Android Studio 1.4

https://www.udemy.com/android-marshmallow-java-app-development-course/learn/#/lecture/3736992

and

## Changes to be made to cover Android Marshmallow for the flickr app.

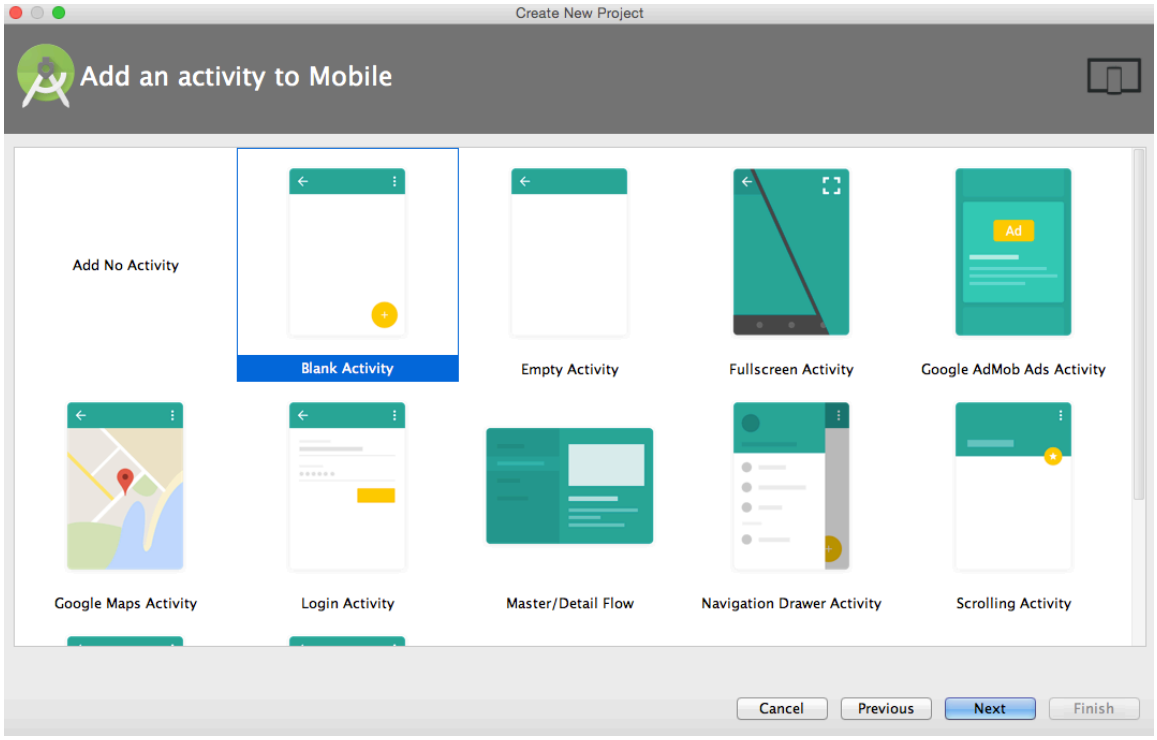
https://www.udemy.com/android-marshmallow-java-app-development-course/learn/#/lecture/3736990

## Things you will need to change from the original recorded videos

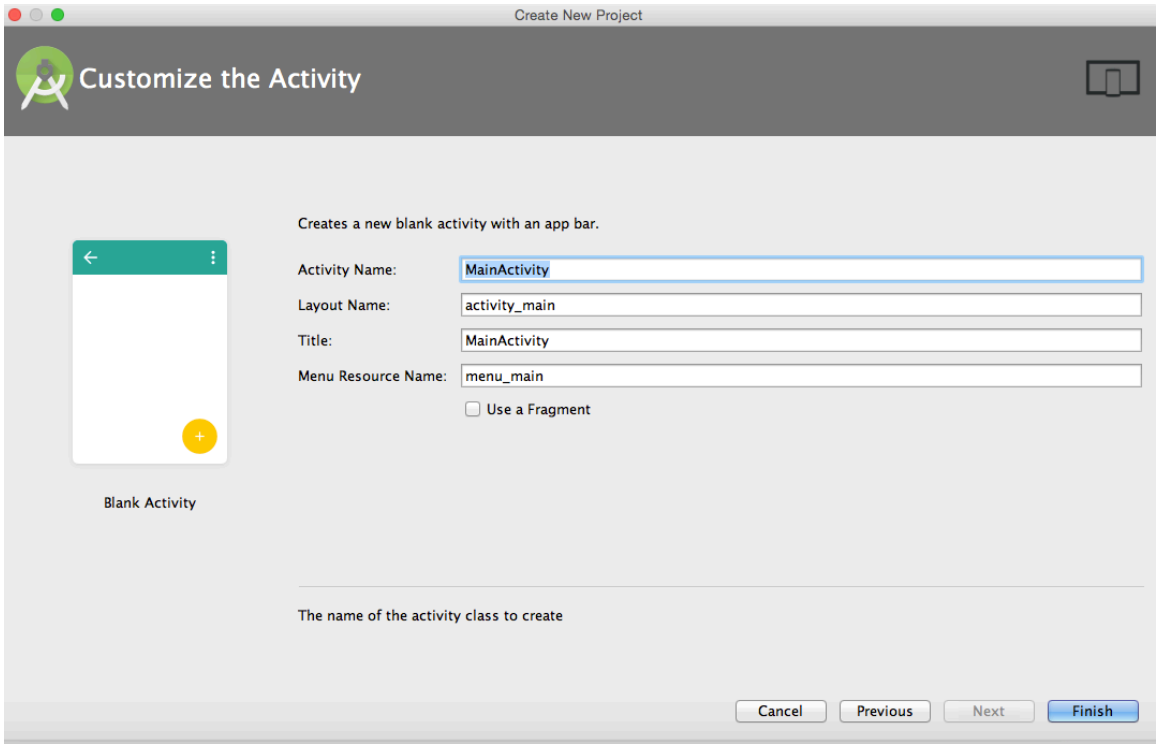
---

Here are the following items to keep in mind when you are building the app in Android Studio 1.4

When creating a new Android Studio Project for the Flickr Browser App, choose **Blank Activity** since **Blank Activity with Fragment** does not exist anymore.



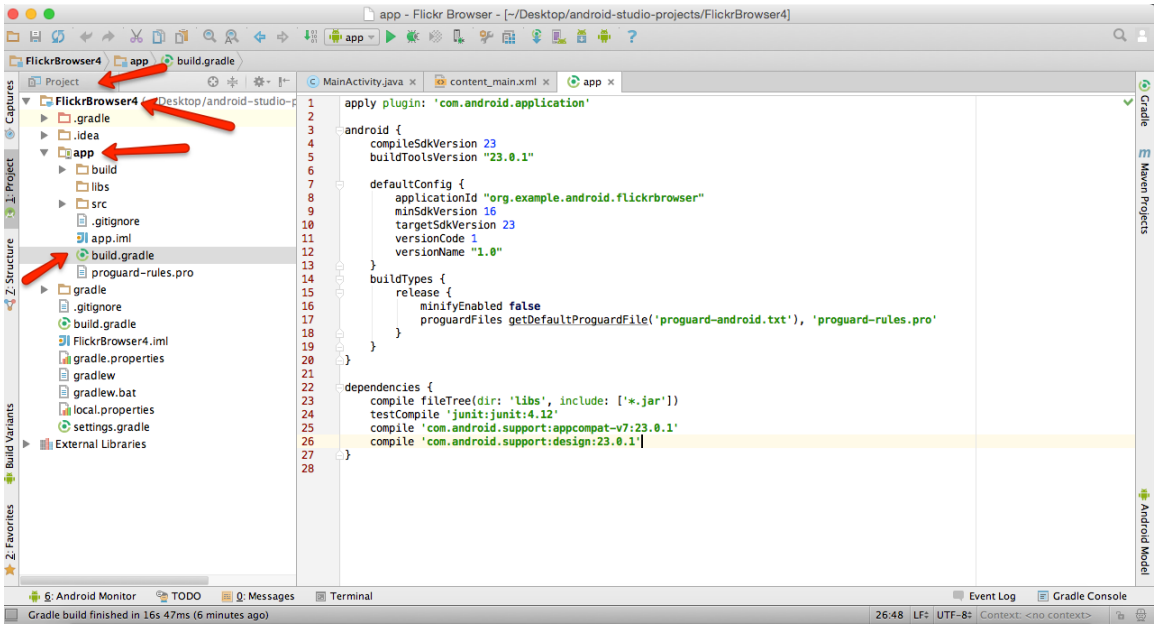

On the next screen don't bother checking the **Use a Fragment** option, leave it unchecked.



# build.gradle Dependencies

**Important Note:** *By doing the following step in this document you no longer need to download the picasso jar file and install it into the project.*

In the build gradle file found in **ProjecName>App** in the project pane window of Android Studio. Also make sure to set the view to **Project**.




Then make sure to add the following dependencies after line 26 of the build.gradle file.

```
compile 'com.android.support:recyclerview-v7:23.0.1'
compile 'com.squareup.picasso:picasso:2.5.2'
compile 'com.android.support:cardview-v7:23.0.1'
```

Then click on Sync Now link on the upper right corner to start syncing the build.gradle.

There is an alternative way to adding dependencies first you have to delete the following line from your build.gradle.

```
compile 'com.android.support:appcompat-v7:23.0.1'
```

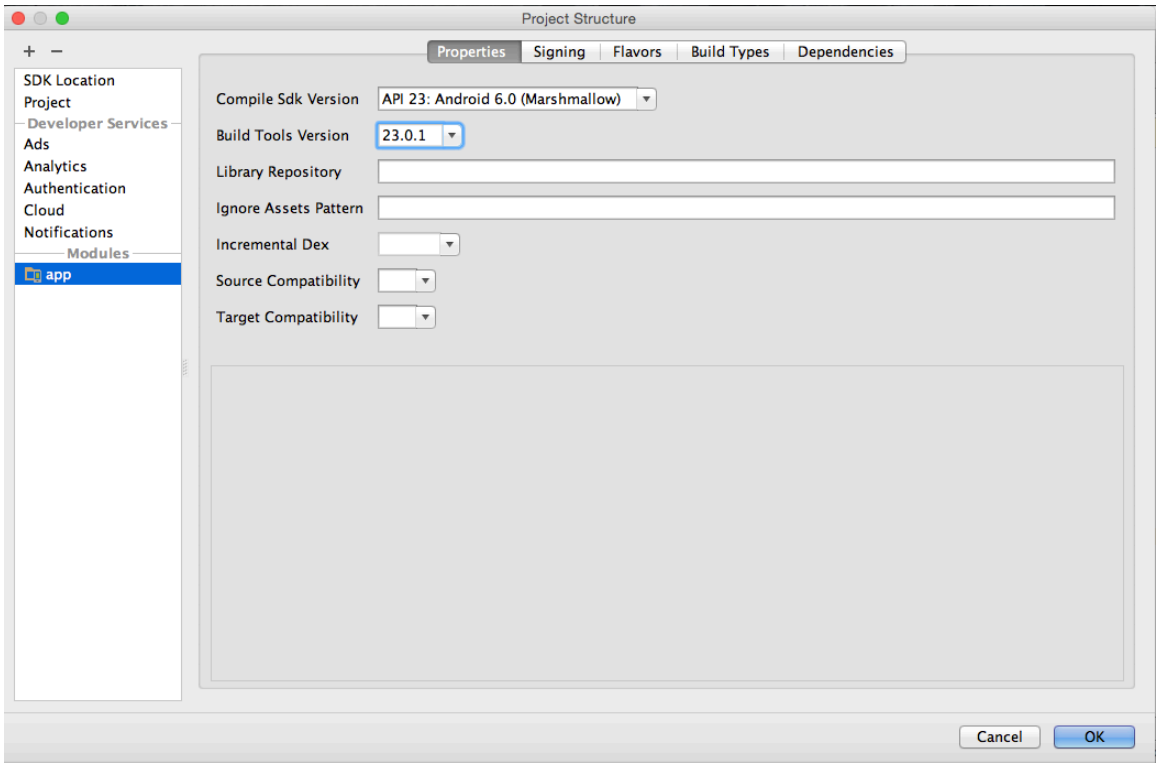
These are the only items that should be left on your dependencies area of your build.gradle file

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:design:23.0.1'
}
```

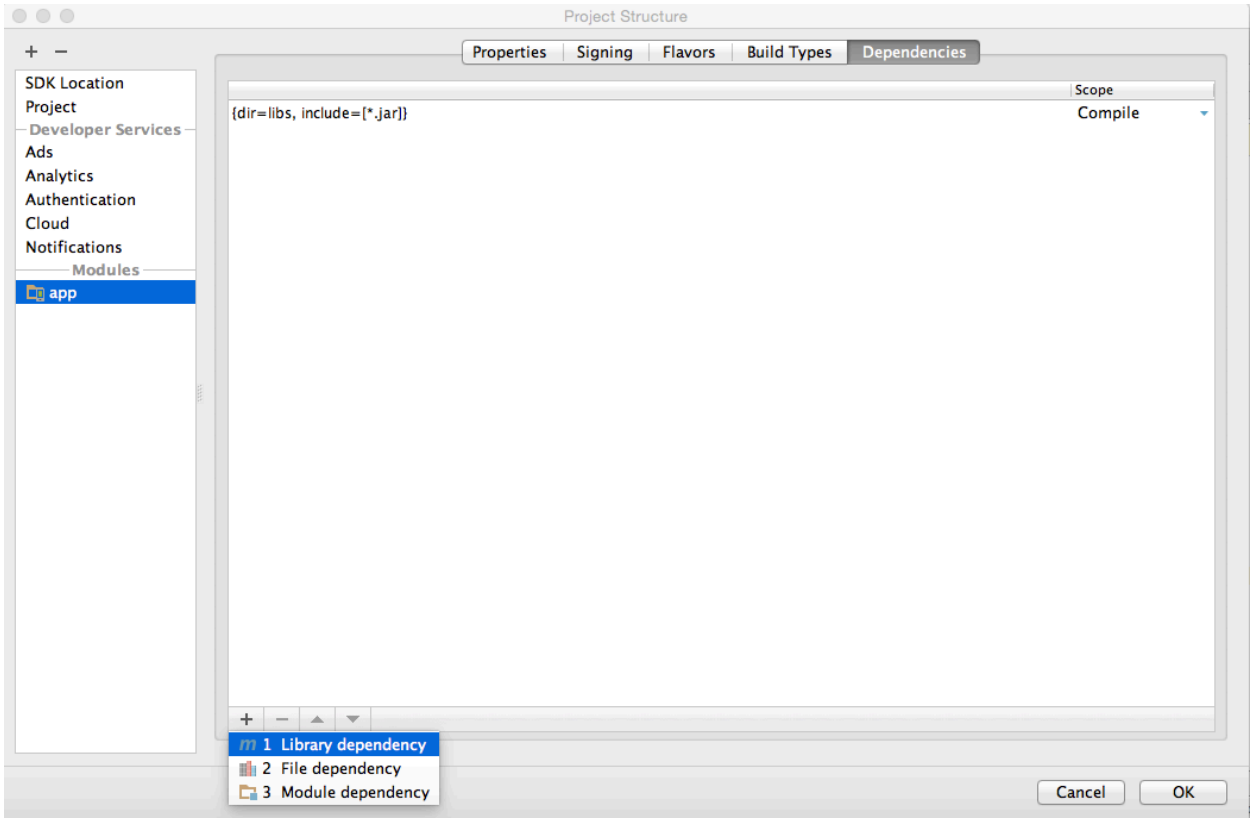
Next, go to your project pane, right click on the top most folder in the labeled **FlickrBrowser**.

Right click on it and select **Open Module Settings**.

In the open module settings window, select **app** on the **left menu**. Make sure that the latest Build Tools Version is selected, which in this current time of documentation is **23.0.1**.

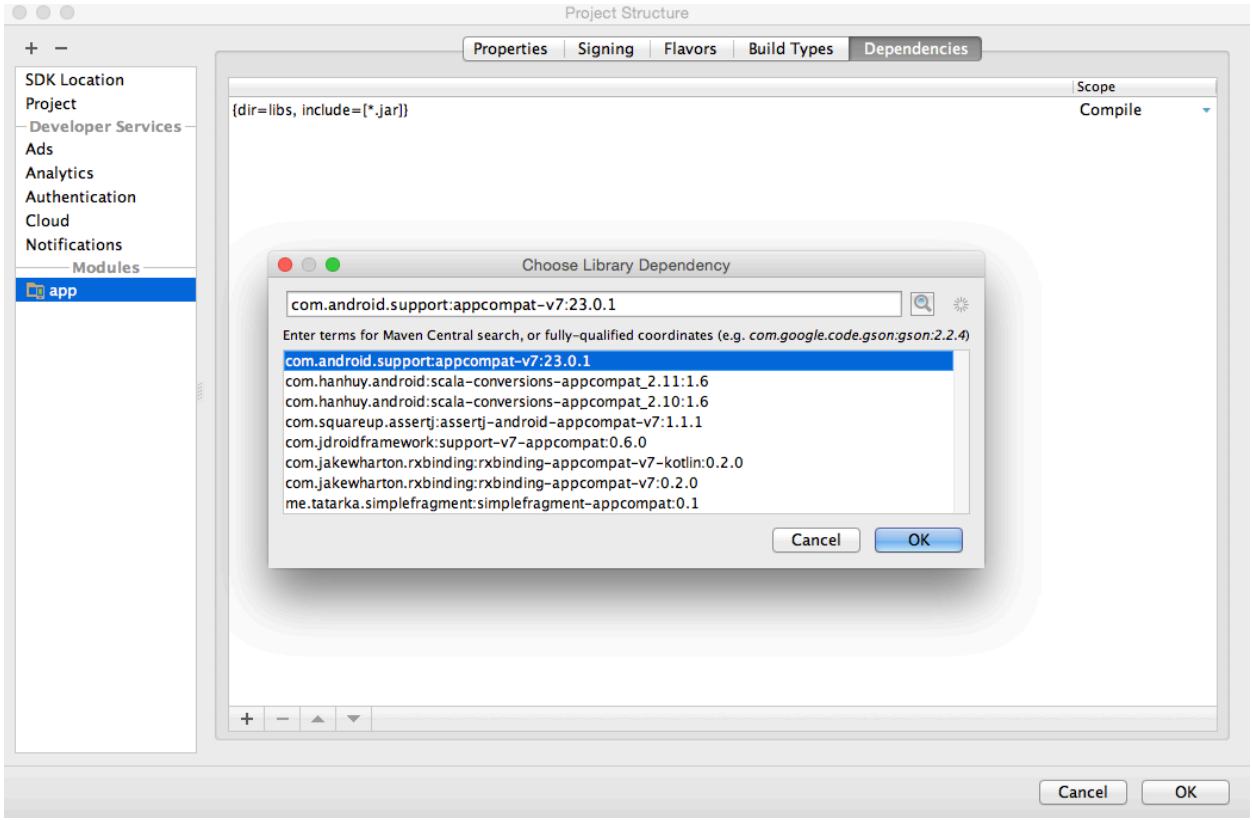


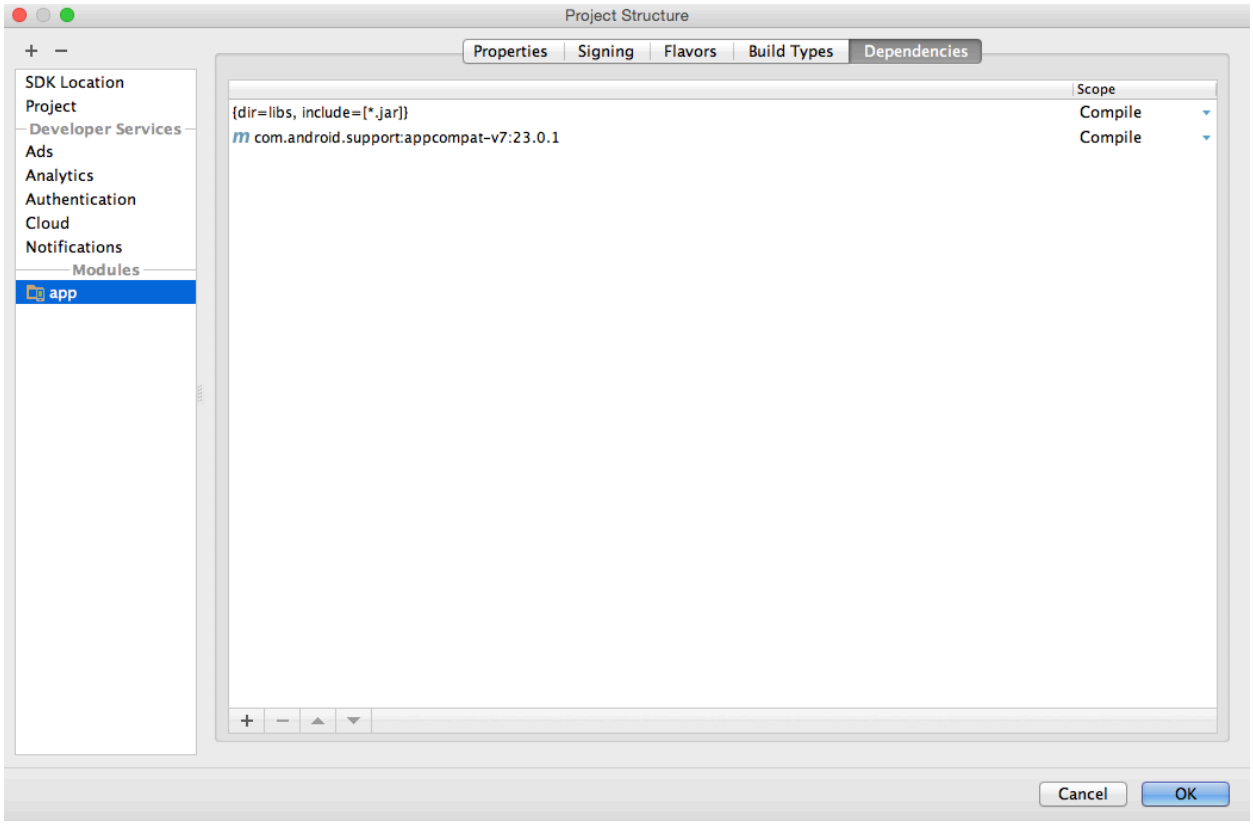

Click the **Dependencies** tab, which is where you we will add the dependencies with correct latest versions. First click the plus sign in the lower left, and choose **Library** dependency.



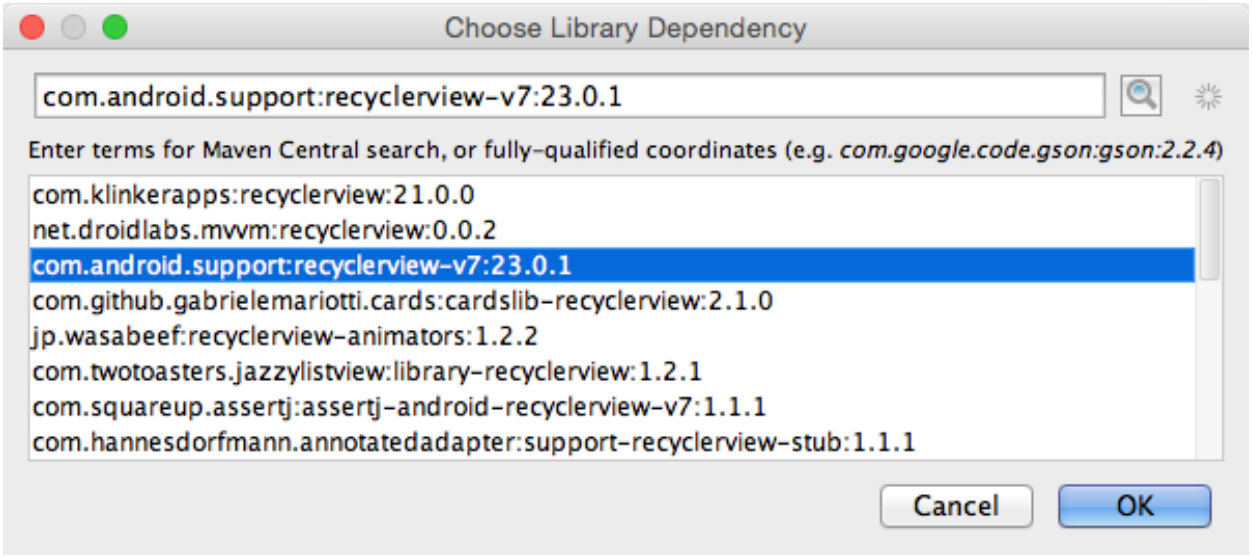

A small window will pop out for searching the Library depdencies we want to add. First seach for **appcompat** by typing **app compat** in the search field and press enter. Wait for it to finish searching. It will then bring back a list of all the appcompats it could find.

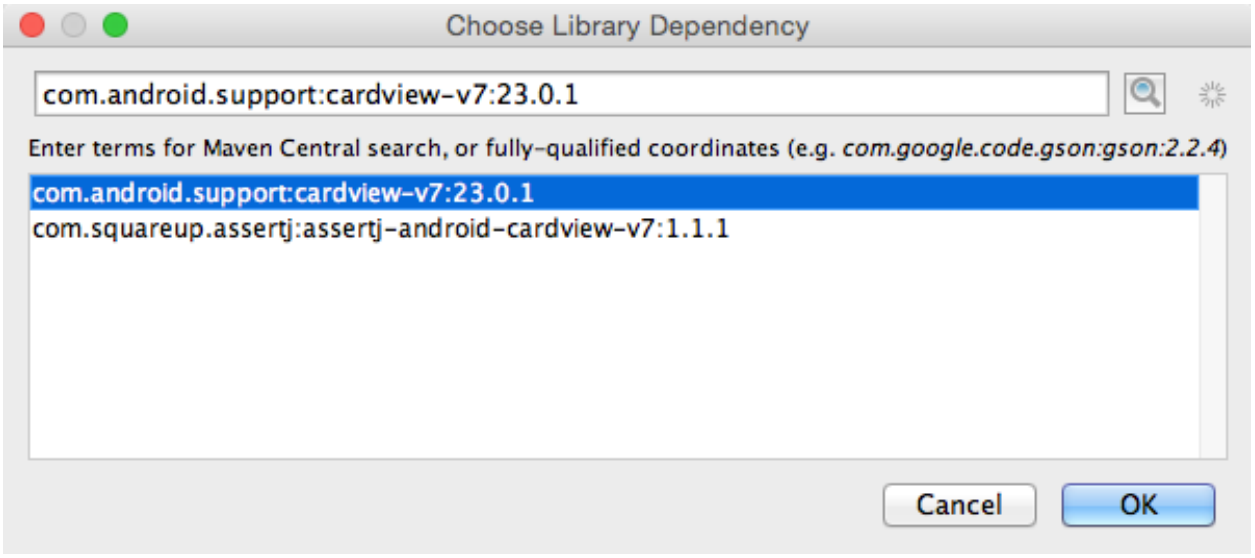
The one you need to choose is the most latest version for the **com.android.support** which in this case is 23.0.1. So select that one and press Ok, it will then get added to the list in the dependencies tab.


Project Structure
Properties
Signing
Flavors
Build Types
Dependencies
SDK Location
Project
Developer Services
Ads
Analytics
Authentication
Cloud
Notifications
Modules
app
Scope
{dir=libs, include=[*.jar]}
Compile
Choose Library Dependency
com.android.support:appcompat-v7:23.0.1
Enter terms for Maven Central search, or fully-qualified coordinates (e.g. com.google.code.gson:gson:2.2.4)
com.android.support:appcompat-v7:23.0.1
com.hanhuy.android:scala-conversions-appcompat_2.11:1.6
com.hanhuy.android:scala-conversions-appcompat_2.10:1.6
com.squareup.assertj:assertj-android-appcompat-v7:1.1.1
com.jdroidframework:support-v7-appcompat:0.6.0
com.jakewharton.rxbinding:rxbinding-appcompat-v7-kotlin:0.2.0
com.jakewharton.rxbinding:rxbinding-appcompat-v7:0.2.0
me.tatarka.simplefragment:simplefragment-appcompat:0.1
Cancel
OK
Cancel
OK

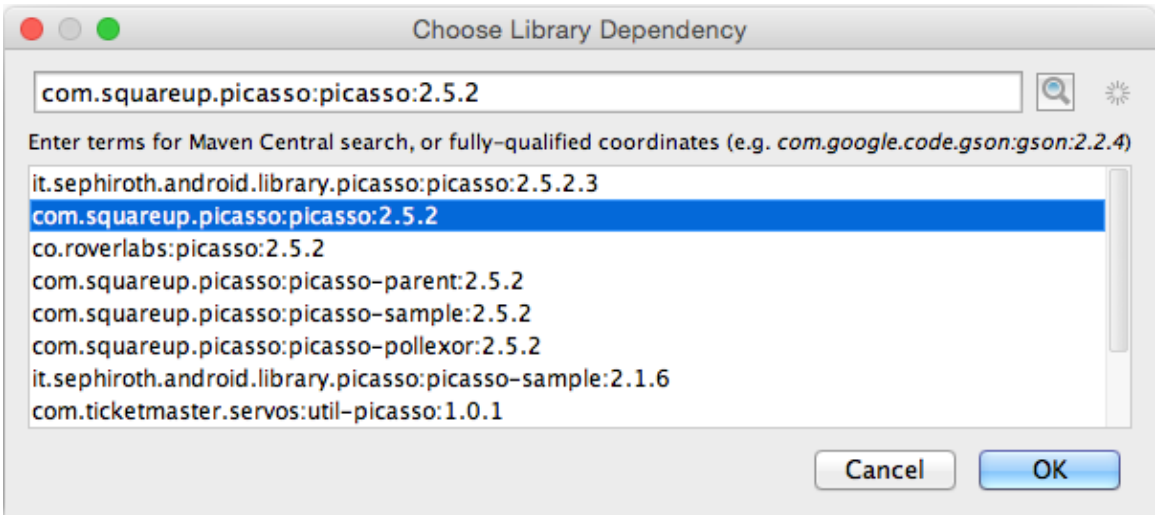

Project Structure
Properties
Signing
Flavors
Build Types
Dependencies
SDK Location
Project
Developer Services
Ads
Analytics
Authentication
Cloud
Notifications
Modules
app
Scope
{dir=libs, include=[*.jar]}
Compile
com.android.support:appcompat-v7:23.0.1
Compile
Cancel
OK

Next, we need to add dependencies for **cardview** and **Recyclerview**. So search for those again and add them. Be careful thought because you might pick the wrong ones, so make sure the ones you have selected have the **com.android.support** on it.

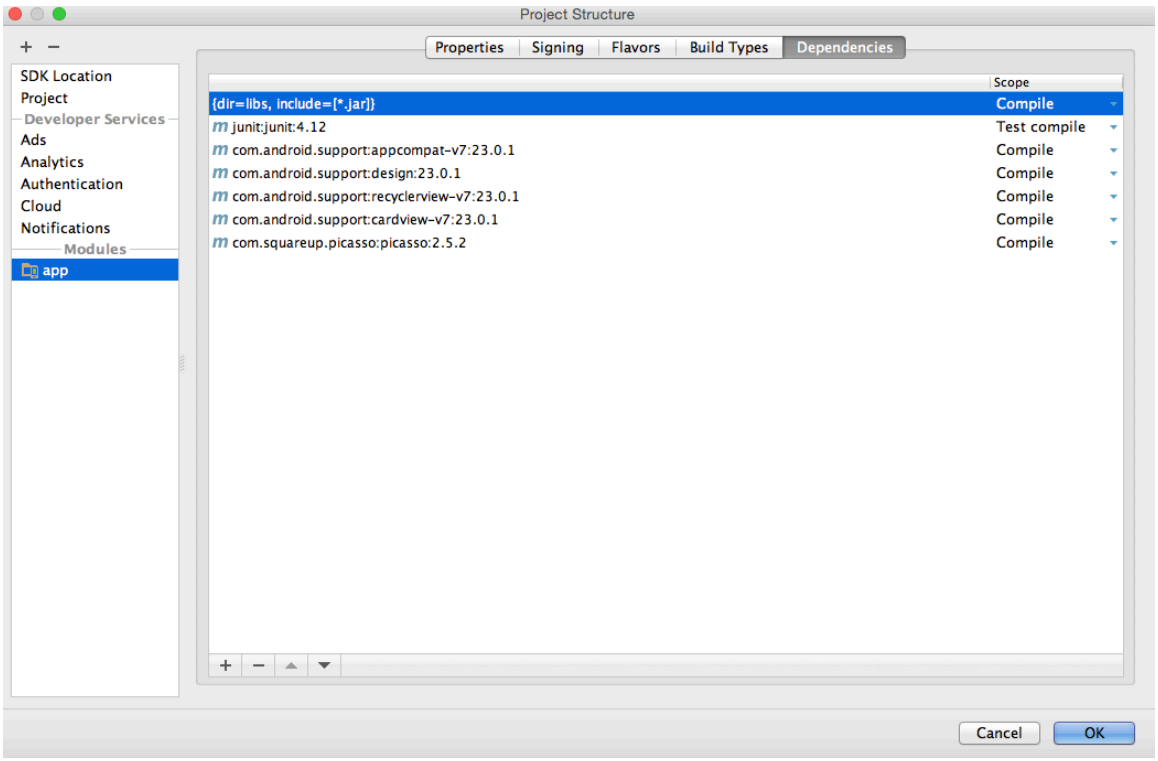



And lastly we need to add the dependency for **picasso** so type that into the search box and add it. **Be sure to pick the one that starts with com.squareup.picasso and of course pick the latest version.**




Once you have all 4 depdencies showing up the in the window press Ok to add them to the build.gradle file.




The build.gradle file will add the dependencies and it will automatically start syncing them. Wait for it to finish the syncing process and your done.

Your build.gradle contents should now look like the following below.

```
apply plugin: 'com.android.application'
```

```
android {
    compileSdkVersion 23
    buildToolsVersion "23.0.1"

    defaultConfig {
        applicationId "org.example.android.flickrbrowser"
        minSdkVersion 16
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-
android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:23.0.1'
    compile 'com.android.support:design:23.0.1'
    compile 'com.android.support:recyclerview-v7:23.0.1'
    compile 'com.squareup.picasso:picasso:2.5.2'
    compile 'com.android.support:cardview-v7:23.0.1'

}
```

In the end you should have Appcompat, RecyclerView, CardView, and Picasso in your build.gradle dependencies.

---

## Commenting out the Floating Action Button and Creating Blank Activity.

For every Blank Activity you create throughout the entire project you need to comment out the following lines of code. Also when creating **blank activity with fragment** please choose instead a Blank Activity and don't check the checkbox for **Use a Fragment** when you come to the screen when your adding the Name

Example if you have a created a Blank Activity called MainActivity you need to comment out the following lines in the MainActivity.java file. Just select the lines

below and press CMD+/ or CTRL+/ if your in a Windows. Java files are stored in the **App>Src>Main>Java>Packagename** of the project.

```
FloatingActionButton fab = (FloatingActionButton)
findViewById(R.id.fab);
fab.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Snackbar.make(view, "Replace with your own action",
Snackbar.LENGTH_LONG)
                .setAction("Action", null).show();
    }
});
```

After commenting the following lines of code the current inside of your MainActivity.java should now look like the following below.

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar)
findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

//        FloatingActionButton fab = (FloatingActionButton)
findViewById(R.id.fab);
//        fab.setOnClickListener(new View.OnClickListener()
{
//            @Override
//            public void onClick(View view) {
//                Snackbar.make(view, "Replace with your
own action", Snackbar.LENGTH_LONG)
//                        .setAction("Action",
null).show();
//            }
//        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action
bar if it is present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
```

```
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action
bar will
        // automatically handle clicks on the Home/Up
button, so long
        // as you specify a parent activity in
AndroidManifest.xml.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
}
```

Then you need to open up the activity's corresponding activity layout xml file which in this example is activity_main.xml and comment out these lines of code. Xml layout files are always stored in **App>Src>Main>Res>layout** of the project.

```
<android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.AppBarOverlay">

    <android.support.v7.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:background="?attr/colorPrimary"
        app:popupTheme="@style/AppTheme.PopupOverlay"/>

</android.support.design.widget.AppBarLayout>

<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom|end"
    android:layout_margin="@dimen/fab_margin"
    android:src="@android:drawable/ic_dialog_email"/>
```

You can comment them by selecting the lines of code and press CMD+/ or CTRL+/ for windows.

This is what your activity_main.xml contents should look like after commenting.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout

xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:context=".MainActivity">

    <!--<android.support.design.widget.AppBarLayout-->
        <!--android:layout_height="wrap_content"-->
        <!--android:layout_width="match_parent"-->
        <!--android:theme="@style/AppTheme.AppBarOverlay">-
->

        <!--<android.support.v7.widget.Toolbar-->
            <!--android:id="@+id/toolbar"-->
            <!--android:layout_width="match_parent"-->
            <!--
android:layout_height="?attr/actionBarSize"-->
            <!--android:background="?attr/colorPrimary"-->
            <!--
app:popupTheme="@style/AppTheme.PopupOverlay"/>-->

    <!--</android.support.design.widget.AppBarLayout>-->

    <include layout="@layout/content_main"/>

    <!--
<android.support.design.widget.FloatingActionButton-->
        <!--android:id="@+id/fab"-->
        <!--android:layout_width="wrap_content"-->
        <!--android:layout_height="wrap_content"-->
        <!--android:layout_gravity="bottom|end"-->
        <!--android:layout_margin="@dimen/fab_margin"-->
        <!--
android:src="@android:drawable/ic_dialog_email"/>-->

</android.support.design.widget.CoordinatorLayout>
```

**Apply this every time you create a new Blank Activity. You must comment out the following lines of code in both the java file and the corresponding activity xml file of it.**

---

## Removing Some Line of Code in the AndroidManifest.xml

Open up the AndroidManifest.xml file in **App>Src>Main** and look for the following line of code. It should be inside the first <activity> tag for .MainActivity.

```
<activity
    android:name=".MainActivity"
    android:label="@string/app_name"
    android:theme="@style/AppTheme.NoActionBar" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN"
/>

        <category
android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

**Delete** the line of code highlighted in **red** and your current AndroidManifest.xml file should only look like this for now.

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme" >
        <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>
    </application>

</manifest>
```

---

## Editing content_main.xml and content xml files in general

When the lecture shows editing **activity_main.xml** you should instead edit the **content_main.xml** file. The same applies to any new Blank Activities you create, instead of editing the activity xml file for that activity; you should instead edit the content xml file.

When editing the **content_main.xml file or any content xml file**. Please do not delete the following highlighted in **red**.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout

xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"

android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"

app:layout_behavior="@string/appbar_scrolling_view_behavior
"
    tools:showIn="@layout/activity_main"
    tools:context=".MainActivity">

    <TextView
        android:text="Hello World!"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
</RelativeLayout>
```

Another example if I created a new Blank Activity called SearchActivity. It would generate me the java file, and two xml files activity_search.xml and content_activity.xml

---

When editing the content xml of the SearchActivity which is content_activity.xml I would not remove the following lines.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout

xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"

android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"

app:layout_behavior="@string/appbar_scrolling_view_behavior
"
    tools:showIn="@layout/activity_search"

tools:context="org.example.android.flickrbrowser.SearchActi
vity">

</RelativeLayout>
```

Please keep this in mind.

---

## Additional Method for the RecyclerItemClickListener Class

When you finish going through the lecture on creating the **RecyclerItemClickListener** Class which is **Lecture 79** as of the time of this documentation. You need to add this empty method in before the closing bracket of the entire class.

```
@Override
public void onRequestDisallowInterceptTouchEvent(boolean
disallowIntercept) {

}
```

Your RecyclerItemClickListener class should now look like this in then.

```
public class RecyclerItemClickListener implements
RecyclerView.OnItemTouchListener {

    public static interface OnItemClickListener {
        public void onItemClick(View view, int position);

        public void onItemLongClick(View view, int
position);
    }

    private OnItemClickListener mListener;
    private GestureDetector mGestureDetector;

    public RecyclerItemClickListener(Context context, final
RecyclerView recyclerView, OnItemClickListener listener) {
        mListener = listener;
        mGestureDetector = new GestureDetector(context, new
GestureDetector.SimpleOnGestureListener() {
            public boolean onSingleTapUp(MotionEvent e) {
                return true;
            }

            public void onLongPress(MotionEvent e) {
                View childView =
recyclerView.findChildViewUnder(e.getX(), e.getY());
                if (childView != null && mListener != null)
{
                    mListener.onItemLongClick(childView,
recyclerView.getChildPosition(childView));
                }
            }
        });
    }

    public boolean onInterceptTouchEvent(RecyclerView view,
MotionEvent e) {
        View childView = view.findChildViewUnder(e.getX(),
e.getY());
        if (childView != null && mListener != null &&
mGestureDetector.onTouchEvent(e)) {
            mListener.onItemClick(childView,
view.getChildPosition(childView));
        }
        return false;
    }

    public void onTouchEvent(RecyclerView view, MotionEvent
```

```
motionEvent) {
    }

    @Override
    public void
onRequestDisallowInterceptTouchEvent(boolean
disallowIntercept) {

    }

}
```