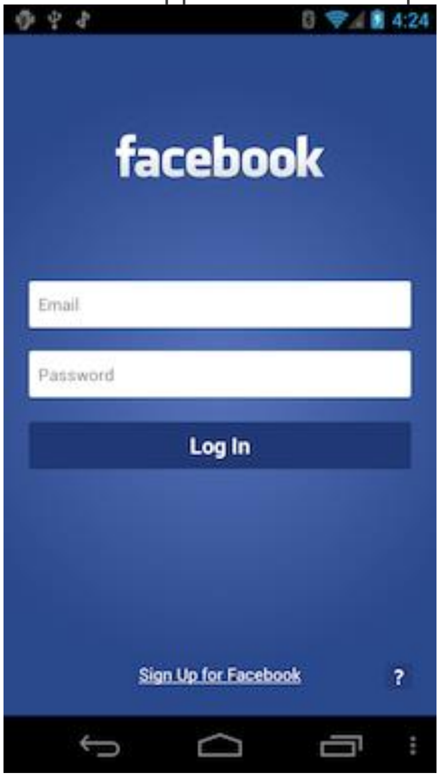
Activity

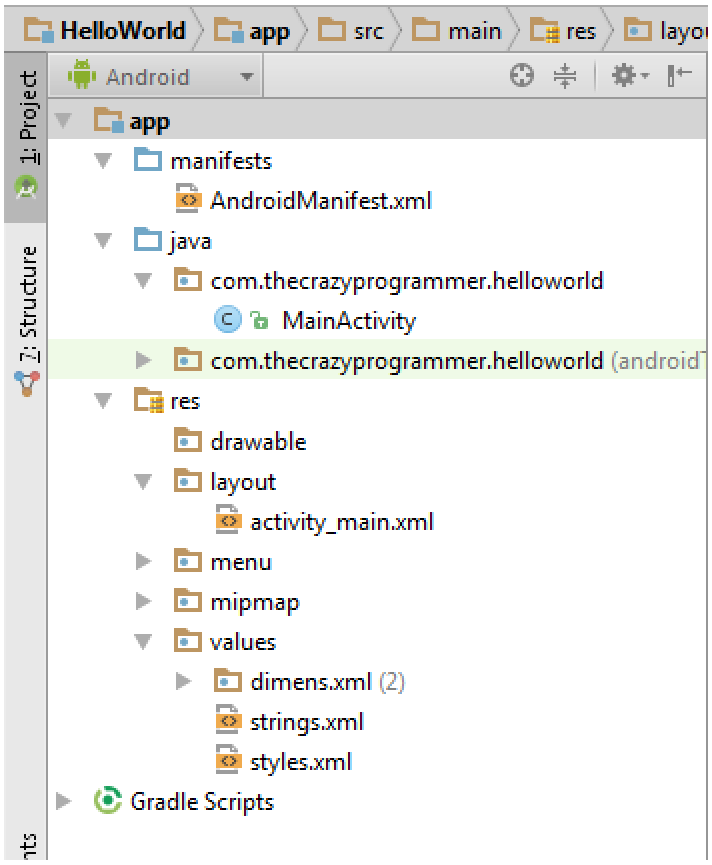
An Activity is a screen or window that interacts with the user. Activity is responsible for holding various UI components like label, button, textbox, etc. An app can have number of activities. Below I have given the login screen of Facebook app. It is an example of Activity.



Basically there are two files associated with a single activity. One is a Java file that contains working related code and another is an XML file that contains design related code.

Now I will explain about the various files and folders used in an android app.

The below image shows the project structure of an app in Android Studio.



app

It is the root folder that contains all other sub-folders.

manifests

It contains AndroidManifest.xml file that gives information about the app to the android os.

java

This folder contains packages and Java source files. By default it contains MainActivity.java, later on we can create more source files according to our need.

res

This folder contains all the extra resources required in our app.

res/drawable

All the images of different resolutions are placed in this folder.

res/layout

It contains layout xml files that define design or user interface. By default it contains activity_main.xml, later on we can create more layout files according to our need.

res/values

dimens.xml: It contains dimension files that defines various dimensions like margin, padding, etc.

strings.xml: In this file we can define different texts or strings that we use in app. For example name of activities, buttons, labels, etc.

styles.xml: Here we define different styles for the components that we use in our design. For example size and color of buttons, labels, images, etc.

Below I have shared the code for the Hello World example that you have learnt to create in last tutorial. Now I will explain it in brief.

MainActivity.java

```java
package com.thecrazyprogrammer.helloworld;

import android.app.Activity;
import android.os.Bundle;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

It is the java source file associated with the activity that we have created.

The **onCreate()** method is the first method called when the activity is created.

setContentView() method gives the information about which layout resource file to use in the activity. In this example I have used activity_main.xml.

activity_main.xml

```xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
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"
tools:context=".MainActivity">

    <TextView android:text="@string/hello_world"
android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</RelativeLayout>
```

It is the xml file associated with the activity. It contains the code related to design. You can see in above code that **<RelativeLayout>** is the outermost tag. It is the layout tag that defines how UI components like buttons, labels, etc are arranged on activity. We will discuss about all layouts in detail in upcoming tutorials.

RelativeLayout contains **<TextView>** tag. TextView is a UI component used to add a label or text on the activity.

Each tag has various attributes like **android:layout_width** and **android:layout_height** to give width and height.

@string/hello_world refers to the Hello World string defined in **strings.xml** file.

dimens.xml

```
<resources>
    <!-- Default screen margins, per the Android Design guidelines. -->
    <dimen name="activity_horizontal_margin">16dp</dimen>
    <dimen name="activity_vertical_margin">16dp</dimen>
</resources>
```

Various dimensions are defined in dimens.xml file.

strings.xml

```
<resources>
    <string name="app_name">Hello World</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>
</resources>
```

Here we define different texts or strings that we use in app.

AndroidManifest.xml

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        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>
```

Every application must have an **AndroidManifest.xml** file. It gives essential information about app to the android os.

R.java

It is an auto generated file and we should not modify its content. It contains the ids of all the resources present in res folder. Whenever we add new component in our layout xml file, its id is automatically generated in R.java file. We use these ids in our Java source file to perform any action on the components.