Components of Android App

Application components are the essential building blocks of an Android application. These components are loosely coupled by the application manifest file  *AndroidManifest.xml*  that describes each component of the application and how they interact.

There are following four main components that can be used within an Android application −

|  |  |
| --- | --- |
| **Sr.No** | **Components & Description** |
| 1 | **Activities**  They dictate the UI and handle the user interaction to the smart phone screen. |
| 2 | **Services**  They handle background processing associated with an application. |
| 3 | **Broadcast Receivers**  They handle communication between Android OS and applications. |
| 4 | **Content Providers**  They handle data and database management issues. |

1. **Activities**

* An activity is the first building block of android App. It is single screen with user interface.

**Example:**

Activity for login screen, a activity shows the catalog, activity for getting our personal details.

* All activity are independent from each other. However, one activity can start another activity when required.
* There are more number of activities an application but there will be one MainActivity from which the execution starts.An activity is implemented as subclass of Activity class. You can create an activity as below

|  |  |
| --- | --- |
| public class MainActivity extends Activity  { | |
| } |

**2. Services**

* A service is component that runs in background to perform the running operations or to perform work for remote processes.
* It does not provide user interface
* Service doesn’t terminate even if the component which initiated got terminated or switched to another application
* For example, when you receive your email updates in inbox it is a service. You get the notification of new e-mail even if you are not using the e-mail app or doing something else.

We can declare the service as below

import android.app.Service;

|  |  |
| --- | --- |
| public class MyService extends Service { | |
| } |

**3.Broadcast Receivers**

* They handle communication between Android OS and applications.
* Broadcast Receivers simply respond to broadcast messages from other applications or from the system.
* Example is notify the user when battery is low.

import android.content.BroadcastReceiver;

|  |  |
| --- | --- |
| public class MyReceiver  extends  BroadcastReceiver { | |
| public void onReceive(context,intent){} |

|  |
| --- |
| } |
|  |

1. **Content Providers**

* It acts as bridge between the data and the application.
* A content provider component supplies data from one application to others on request
* The data may be stored in the file system, the database or somewhere
* We can use content providers to get a specific set of data in any application

You can implement Content Providers by subclassing ContentProvider as below –

import android.content.ContentProvider;

|  |  |
| --- | --- |
| public class MyProvider extends  ContentProvider { | |
| public void onCreate(){} |

|  |
| --- |
| } |

**Additional Android Application Components**

* Fragments
* Views
* Layouts
* Intents
* Manifest
* Resources

**Fragments**

A fragment is a part of user interface in a activity.

You can use one or more fragments in an activity.

Fragments typically contains Views and ViewGroups inside them.

You can assume fragment as a modular part of activity that has it’s own life cycle, can receive it’s own input, can be added or removed dynamically, while activity is running.

A Fragment must always be embedded in activity because it’s life cycle is affected by host activity life cycle. i.e. you can use fragment without activity.

**Views**

This is basic building block for user interface component. It is rectangular area on the screen. We can draw or handle some events in this rectangular area. Also, View is base class for widgets( used to create ui components).

Views are the individual GUI elements, like a Text View, Button.

View Groups are containers for Views. A View Group groups a collection of Views together. Views and View Groups can be nested inside an activity or inside a fragment (which is again nested inside an activity).

**Layouts**

A layout is visual structure for any user interface. For example, UI for activity, widgets, fragments etc. You can declare layouts in 2 ways.  
1. Statically: You can declare layouts in xml file. xml file is added in **res => layout** folder of any android application.  
2. Dynamically: You can declare layouts at runtime as well. To create layouts at runtime, you would need to add related code in java file.

**Intents**

Intent is an abstract information related to an operation to be performed. Intent is used to communicate between different android components in several ways. Some of them are –  
(i) Start an Activity: You can use intent to start an activity from another activity or fragment class.  
(ii) Start a Service: Same as Activity, You can use intent to start an service from a activity or fragment class.  
(iii) Broadcasting a message: You can use intent to broadcast a message so that proper action is taken against the message broadcasted.

**Manifest**

Every android application must have AndroidManifest.xml file in it’s root directory.

It contains a short description of the android application.

For example, It contains the package name of the application(that acts as unique identifier of the application), information about the components of the application(Activities, services, broadcast receivers, content providers)

**Resources**

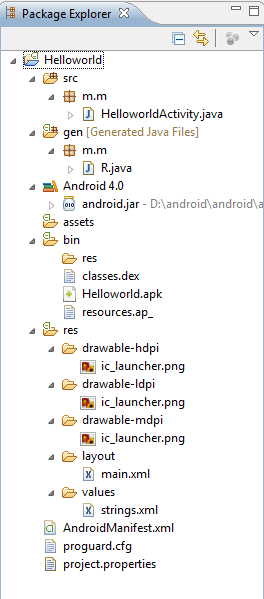
There are many more things you need while creating UI for an activity in android application. For example,

You may need images, icons, animations, audio, video etc. while creating user interface. They are the resources we need to store somewhere in the application.

Android has a separate and structured folder for these resources. They are stored in **res**folder in any android application.

Anatomy of Android App

The following picture shows the various files we need to develop Android App.



**➤ src — Contains   the   .java source files for your project. In this example, there is** **one  file, HelloworldActivity.java.  The HelloworldActivity.java file is the source file for our activity. We will write the code for your application in this file.**

**➤ Android 4.0 library — This item contains one file, android.jar, which contains all the** **class libraries needed for an Android application.**

**➤ gen — Contains the R.java file, a compiler-generated file that references all the**

**resources found in your project. We should not modify this file.**

**➤ assets — This folder contains all the assets used by our application, such as**

**HTML, text files, databases, etc.**

**➤ res — This folder contains all the resources used in your application. It also**

**contains a few other subfolders: drawable-<resolution>, layout, and values.**

**➤ AndroidManifest.xml — This is the manifest file for your Android application. Here**

**Example Prg**