MCA253 - Mobile Applications

Unit:1.1

Getting Started with Android

Dr. Siddesha S MCA, M.Sc Tech (by Research) , Ph.D,
Assistant Professor,
Dept. of Computer Applications,
JSS Science and Technology University
Mysuru – 570 006


evolution

Introduction

“Anytime – Anywhere – Anything”

What is Android?

- Android is an open source operating system and programming platform developed by Google for mobile phones and other mobile devices, such as tablets.
- It can run on many different devices from many different manufacturers.
- Android includes a software development kit (SDK) that helps you write original code and assemble software modules to create apps for Android users.
- Android also provides a marketplace to distribute apps. All together, Android represents an *ecosystem* for mobile apps.

Introduction

Why develop apps for Android?

- Developers create apps for a variety of reasons.
- They may need to address business requirements or build new services or businesses, or they may want to offer games and other types of content for users.
- Developers choose to develop for Android in order to reach the majority of mobile device users.

Most popular platform for mobile apps

- As the world's most popular mobile platform, Android powers hundreds of millions of mobile devices in more than 190 countries around the world.
- It has the largest installed base of any mobile platform and is still growing fast. Every day another million users power up their Android-powered devices for the first time and start looking for apps, games, and other digital content.

Introduction

- **Best experience for app users**
- Android provides a touch screen user interface (UI) for interacting with apps. Android's UI is mainly based on direct manipulation.
- People use touch gestures such as swiping, tapping, and pinching to manipulate on-screen objects.
- In addition to the keyboard, there's a customizable on-screen keyboard for text input.
- Android can also support game controllers and full-size physical keyboards connected by Bluetooth or USB.
- The Android home screen can contain several panes of *app icons*, which launch their associated apps.
- Home screen panes can also contain *app widgets*, which display live, auto-updating content such as the weather, the user's email inbox, or a news ticker. Android can also play multimedia content such as music, animation, and video.

Introduction

- The Android platform, based on the Linux kernel, is designed primarily for touch screen mobile devices such as mobile phones and tablets.
- Because Android-powered devices are usually battery-powered, Android is designed to manage processes to keep power consumption at a minimum, providing longer battery use.

Android Features

- It’s an open source and we can customize the OS based on our requirements.
- It support a connectivity for GSM, CDMA, WIFI, NFC, Bluetooth, etc. for telephony or data transfer.
- It will allow to make or receive a calls / SMS messages and we can send or retrieve a data across mobile networks.
- By using WIFI technology we can pair with other devices using apps
- Android have a multiple APIs to support a location-based services such as GPS
- We can perform all data storage related activities by using light weight database SQLite.
- It have a wide range of media supports like AVI, MKV, FLV, MPEG4 etc. to play or record variety of audio / video and having a different image formats like JPEG, PNG, GIF, BMP, MP3, etc.

Android Features

- It has an extensive support for multimedia hardware control to perform playback or recording using camera and microphone
- It has an integrated open source webkit layout based web browser to support HTML5, CSS3
- It supports a multi-tasking, we can move from one task window to another and multiple applications can run simultaneously .
- It will give a chance to reuse the application components and the replacement of native applications.
- We can access the hardware components like Camera, GPS, and Accelerometer
- It has a support for 2D/3D Graphics

Android History

- Initially Google launched a first version of Android platform on Nov 5, 2007.
- From that onwards Google released a lot of android versions under a codename based on desserts, such as Apple Pie, Banana Bread, Cupcake, Donut, Éclair, Froyo, Gingerbread, Jellybeans, Kitkat, Lollipop, marshmallow, etc. and made a lot of changes and additions to the android platform.

Release Date	Version	API Level	Version Name
September 23, 2008	Android 1.0	1	Apple Pie
February 9, 2009	Android 1.1	2	Banana Bread
April 30, 2009	Android 1.5	3	Cupcake
September 15, 2009	Android 1.6	4	Donut
October 26, 2009	Android 2.0	5	Eclair
December 3, 2009	Android 2.0.1	6	
January 12, 2010	Android 2.1	7	

Android History

Release Date	Version	API Level	Version Name
May 20, 2010	Android 2.2	8	Froyo
January 18, 2011	Android 2.2.1	8	
January 22, 2011	Android 2.2.2	8	
November 21, 2011	Android 2.2.3	8	
December 6, 2010	Android 2.3	9	Gingerbread
February 9, 2011	Android 2.3.1	9	
July 25, 2011	Android 2.3.3	10	
September 2, 2011	Android 2.3.4	10	
February 22, 2011	Android 3.0.x	11	Honeycomb
May 10, 2011	Android 3.1.x	12	
July 15, 2011	Android 3.2.x	13	
October 18,2011	Android 4.0	14	Ice Cream Sandwich
October 19, 2011	Android 4.0.1	14	
November 28, 2011	Android 4.0.2	14	

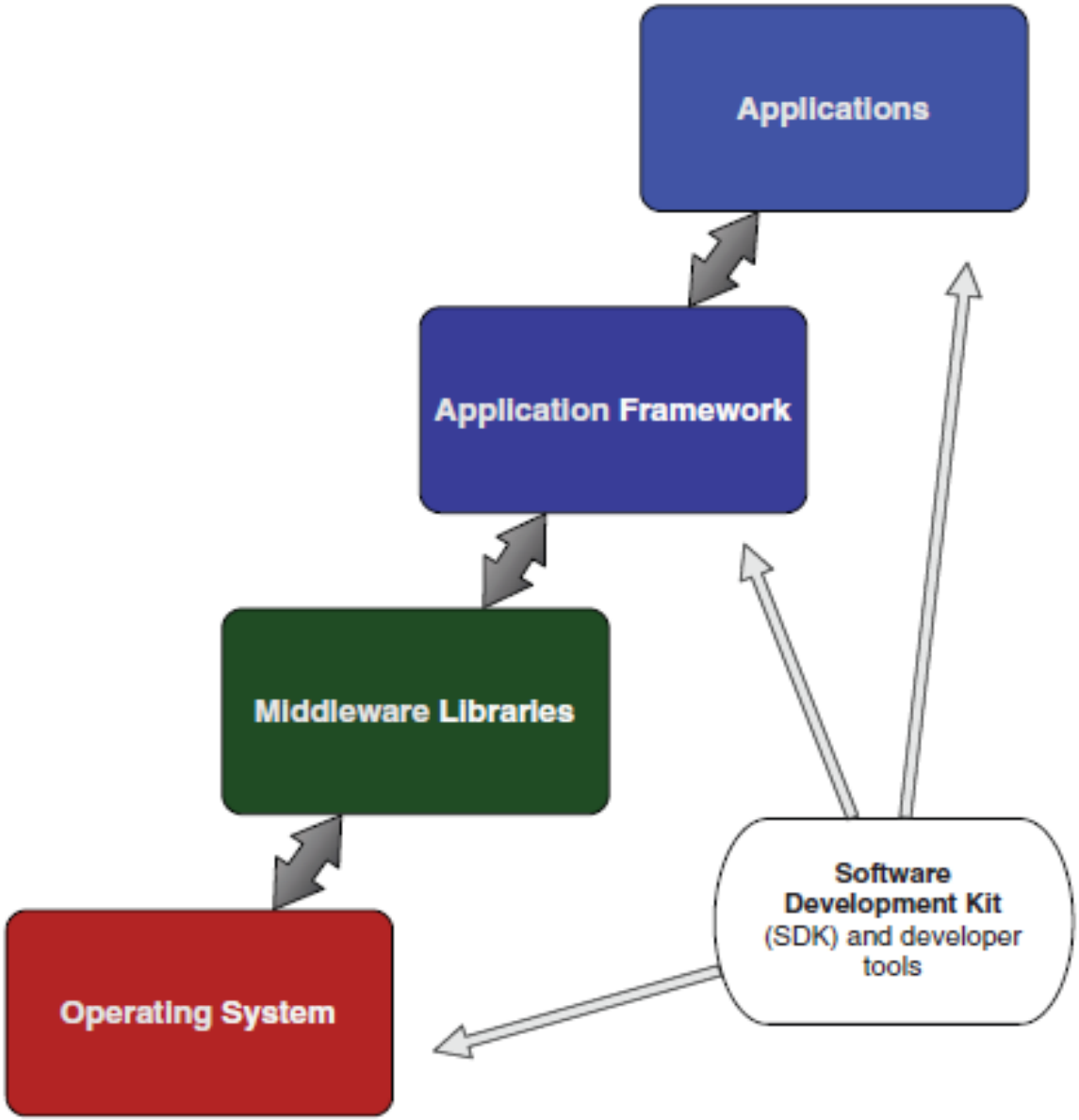
Android History

Release Date	Version	API Level	Version Name
December 16, 2011	Android 4.0.3	15	Ice Cream Sandwich
February 4, 2012	Android 4.0.4	15	
July 9, 2012	Android 4.1	16	Jelly Bean
July 23, 2012	Android 4.1.1	16	
October 9, 2012	Android 4.1.2	16	
November 13, 2012	Android 4.2	17	
November 27, 2012	Android 4.2.1	17	
February 11, 2013	Android 4.2.2	17	
July 24, 2013	Android 4.3	18	
October 31, 2013	Android 4.4	19	Kitkat
June 23, 2014	Android 4.4.1, 4.4.2, 4.4.3, 4.4.4	19	
October 17, 2014	Android 5.0	21	Lollipop
March 09, 2015	Android 5.1	22	

Android History

Release Date	Version	API Level	Version Name
October 5, 2015	Android 6.0	23	Marshmallow
December 7, 2015	Android 6.0.1	23	
August 22, 2016	Android 7.0	24	Nougat
October 4, 2016	Android 7.1	25	
August 21, 2017	Android 8.0	26	Oreo
December 5, 2017	Android 8.1	27	
August 6, 2018	Android 9.0	28	Pie
September 3, 2019	Android 10.0	29	Q -
September 8, 2020	Android 11.0	30	R – Red Velvet cake
October 4, 2021	Android 12.0	31	Snow Cone
March 7, 2022	Android 12L	32	Snow Cone v2
Q3 2022 - Expected	Android 13	33	Tiramisu

Key platform components



Key platform components

- The Android platform can be broken down into five sections:
  - Applications
  - Application framework
  - Middleware libraries
  - Operating system
  - SDK and developer tools

Applications

- Different types of applications are available on most Android devices
- Core open source applications are included as part of Android itself, such as the Browser Camera Gallery Music Phone , and more...
- There are also non–open source Google apps that are included with most official builds, including Market Gmail Maps YouTube and more....
- Third-party applications are available in the Android Market, which can be either open source or proprietary.
- Official apps from popular services like Twitter and Face book, and thousands of other choices....

Application framework

- The application framework provides a tightly integrated part of the platform SDK and APIs that allow for high-level interaction with the system from within applications.
- When your application needs access to hardware sensors, network data, the state of interface elements, or many other things, it gets to that information through the application framework

Middleware libraries

- As the name suggests, middleware is software components that sit in between—in this case between the operating system and the applications/application framework.
- The middleware includes libraries for many functions (data storage, graphics rendering, web browsing, and so on) and it also contains a special subsection called the Dalvik runtime till Android 5.0) later versions have Android Run Time (ART)
- This is Android's special nonstandard virtual machine (VM) and its core application libraries.

Operating system

- At the bottom of the Android stack is the operating system.
- Android's OS is Linux based and performs much the same tasks you'd expect from any conventional desktop computer OS.
- This includes interfacing with the hardware through a set of device drivers such as audio or video drivers), processing user input, managing application processes, handling file and network I/O, and so forth...

SDK and developer tools

- With Android's layered design, each level is an abstraction of the one beneath it.
- As a developer you won't have to deal with lower-level details directly.
- Rather, you'll always access subsystems by going through simple interfaces exposed in Android's application framework.

ANDROID – Environment Setup

You can start your Android application development on either of the following operating systems:

- ❑ Microsoft Windows 10 or later version.
- ❑ Mac OS X 10.5.8 or later version with Intel chip.
- ❑ Linux including GNU C Library 2.7 or later.

All the required tools to develop Android applications are freely available and can be downloaded from the Web.

List of software needed:

- ➢ Java JDK12 or 14 or later version
- ➢ Android Studio (current version is 4.0.1)

There are so many sophisticated Technologies are available to develop android applications, the familiar technology used is Android Studio

ANDROID – Architecture

ANDROID – Architecture

- Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram.

Linux kernel

- At the bottom of the layers is Linux - Linux 3.6 with approximately 115 patches.
- This provides a level of abstraction between the device hardware and it contains all the essential hardware drivers like camera, keypad, display etc.
- Also, the kernel handles all the things that Linux is really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.

ANDROID – Architecture

Libraries

- On top of Linux kernel there is a set of libraries including open-source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc.

Android Libraries

- This category encompasses those Java-based libraries that are specific to Android development.
- Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access.

ANDROID – Architecture

- A summary of some key core Android libraries available to the Android developer is as follows-
  - android.app − Provides access to the application model and is the cornerstone of all Android applications.
  - android.content − Facilitates content access, publishing and messaging between applications and application components.
  - android.database − Used to access data published by content providers and includes SQLite database management classes.
  - android.opengl − A Java interface to the OpenGL ES 3D graphics rendering API.
  - android.os − Provides applications with access to standard operating system services including messages, system services and inter-process communication.

ANDROID – Architecture

- android.text − Used to render and manipulate text on a device display.
- android.view − The fundamental building blocks of application user interfaces.
- android.widget − A rich collection of pre-built user interface components such as buttons, labels, list views, layout managers, radio buttons etc.
- android.webkit − A set of classes intended to allow web-browsing capabilities to be built into applications.
- Having covered the Java-based core libraries in the Android runtime, it is now time to turn our attention to the C/C++ based libraries contained in this layer of the Android software stack.

ANDROID – Architecture

Android Runtime

- Third section of the architecture and available on the second layer from the bottom. This section provides a key component called **Dalvik Virtual Machine(till 5.0) later is ART** which is a kind of Java Virtual Machine specially designed and optimized for Android.
- The Dalvik VM / ATR makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik / ART VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine / ART.
- The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

ANDROID – Architecture

Application Framework

- The Application Framework layer provides many higher-level services to applications in the form of Java classes.
- Application developers are allowed to make use of these services in their applications.
- The Android framework includes the following key services −
  - ❑ Activity Manager − Controls all aspects of the application lifecycle and activity stack.
  - ❑ Content Providers − Allows applications to publish and share data with other applications.
  - ❑ Resource Manager − Provides access to non-code embedded resources such as strings, color settings and user interface layouts.

ANDROID – Architecture

- Notifications Manager − Allows applications to display alerts and notifications to the user.
- View System − An extensible set of views used to create application user interfaces.

Applications

- You will find all the Android application at the top layer. You will write your application to be installed on this layer only.

Android - Application Components

- 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.

Components	Description
Activities	They dictate the UI and handle the user interaction to the Smartphone screen
Services	They handle background processing associated with an application.
Broadcast Receivers	They handle communication between Android OS and applications.
Content Providers	They handle data and database management issues.

Android Application Anatomy / Components

Activities

1. Provides User Interface
2. Usually represents a Single Screen
3. Can contain one/more Views
4. Extends the Activity Base class

Services

1. No User Interface
2. Runs in Background
3. Extends the Service Base Class

Application= Set of Android Components

Intent/Broadcast Receiver

1. Receives and Reacts to broadcast Intents
2. No UI but can start an Activity
3. Extends the BroadcastReceiver Base Class

Content Provider

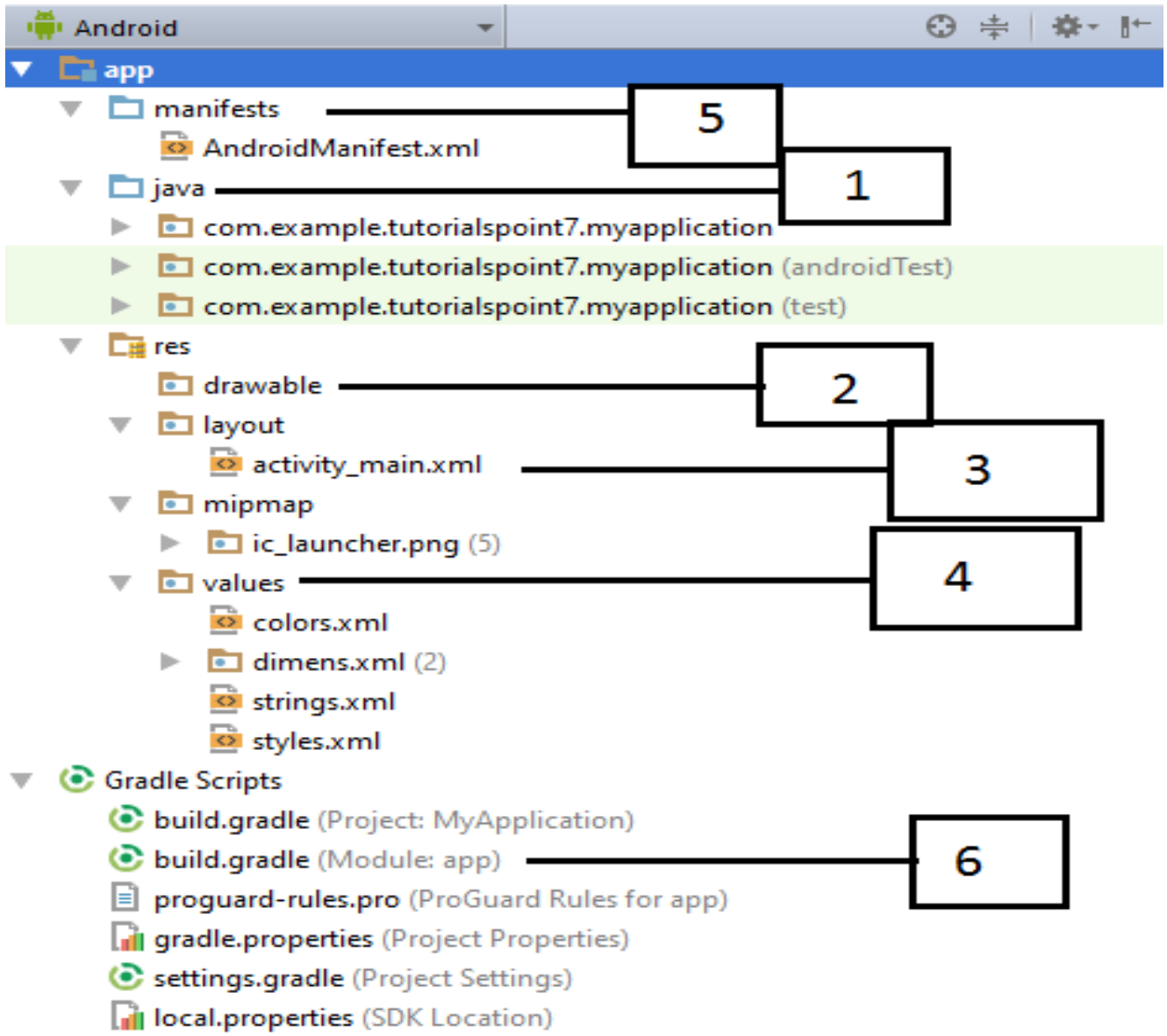
1. Makes application data available to other apps
2. Data stored in SQLite database
3. Extends the ContentProvider Base class

Additional Components

- There are additional components which will be used in the construction of above mentioned entities, their logic, and wiring between them.
- These components are –

Components	Description
Fragments	Represents a portion of user interface in an Activity.
Views	UI elements that are drawn on-screen including buttons, lists forms etc.
Layouts	View hierarchies that control screen format and appearance of the views.
Intents	Messages wiring components together.
Resources	External elements, such as strings, constants and drawable pictures.
Manifest	Configuration file for the application.

Anatomy of Android Application



Anatomy of Android Application

Folder / File	Description
Java	This contains the .java source files for your project. By default, it includes an *MainActivity.java* source file having an activity class that runs when your app is launched using the app icon.
res/drawable-hdpi	This is a directory for drawable objects that are designed for high-density screens.
res/layout	This is a directory for files that define your app's user interface.
res/values	This is a directory for other various XML files that contain a collection of resources, such as strings and colours definitions.
AndroidManifest.xml	This is the manifest file which describes the fundamental characteristics of the app and defines each of its components.
Build.gradle	This is an auto generated file which contains compileSdkVersion, buildToolsVersion, applicationId, minSdkVersion, targetSdkVersion, versionCode and versionName

The Main Activity File

- file which ultimately gets converted to a ART Android Run Time – The main activity code is a Java file MainActivity.java. This is the actual application successor of Dalvik VM) and runs your application. Following is the default code generated by the application wizard for *Hello World!* application −

```
package com.example.helloworld;
 import android.support.v7.app.AppCompatActivity;
 import android.os.Bundle;
 public class MainActivity extends AppCompatActivity {
     protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_main);
     }
  }
```

- Here, R.layout.activity_main refers to the activity_main.xml file located in the res/layout folder.
- The onCreate() method is one of many methods that are figured when an activity is loaded.

The Manifest File

- Whatever component you develop as a part of your application, you must declare all its components in a *manifest.xml* which resides at the root of the application project directory.
- This file works as an interface between Android OS and your application, so if you do not declare your component in this file, then it will not be considered by the OS.
- For example, a default manifest file will look like as following file −

The Manifest File

Following is the list of tags which you will use in your manifest file to specify different Android application components −

- *<activity>elements for activities*
- *<service> elements for services*
- *<receiver> elements for broadcast receivers*
- *<provider> elements for content providers*

The Strings File

- The strings.xml file is located in the *res/values* folder and it contains all the text that your application uses. For example, the names of buttons, labels, default text, and similar types of strings go into this file.
- This file is responsible for their textual content. For example, a default strings file will look like as following file −

```
<resources>
    <string name="app_name">HelloWorld</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">MainActivity</string>
</resources>
```

The Layout File

- The activity_main.xml is a layout file available in *res/layout* directory, that is referenced by your application when building its interface and is modified very frequently to change the layout of your application.
- For your "Hello World!" application, this file will have following content related to default layout –

```
<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" >
    <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerHorizontal="true"
            android:layout_centerVertical="true"
            android:padding="@dimen/padding_medium"
            android:text="@string/hello_world"
            tools:context=".MainActivity" />
</RelativeLayout>
```

Organize resource in Android Studio

```
MyProject/
    app/
        manifest/
            AndroidManifest.xml
    java/
        MyActivity.java
        res/
            drawable/
                icon.png
            layout/
                activity_main.xml
                info.xml
            values/
                strings.xml
```

Directory	Resource Type
anim/	XML files that define property animations. They are saved in res/anim/ folder and accessed from the **R.anim** class.
color/	XML files that define a state list of colors. They are saved in res/color/ and accessed from the **R.color** class
drawable/	Image files like .png, .jpg, .gif or XML files that are compiled into bitmaps, state lists, shapes, animation drawable. They are saved in res/drawable/ and accessed from the **R.drawable** class
layout/	XML files that define a user interface layout. They are saved in res/layout/ and accessed from the **R.layout** class
menu/	XML files that define application menus, such as an Options Menu, Context Menu, or Sub Menu. They are saved in res/menu/ and accessed from the **R.menu** class.

Organize resource in Android Studio

Directory	Resource Type
raw/	Arbitrary files to save in their raw form. You need to call *Resources.openRawResource()* with the resource ID, which is *R.raw.filename* to open such raw files.
values/	❑ arrays.xml for resource arrays, and accessed from the **R.array** class. ❑ integers.xml for resource integers, and accessed from the **R.integer** class. ❑ bools.xml for resource boolean, and accessed from the **R.bool** class. ❑ colors.xml for color values, and accessed from the **R.color** class. ❑ dimens.xml for dimension values, and accessed from the **R.dimen** class. ❑ strings.xml for string values, and accessed from the **R.string** class. ❑ styles.xml for styles, and accessed from the **R.style** class.
xml/	Arbitrary XML files that can be read at runtime by calling *Resources.getXML()*. You can save various configuration files here which will be used at run time.