Android 101

Piya Lumyong

Topic

- Setup Tools
- Hello Android
- File System Structure
- Activity & Fragments
- Intents
- Fragmentations
- Task & Back stack
- Views & Layouts

Setup Tools

Setup Tools

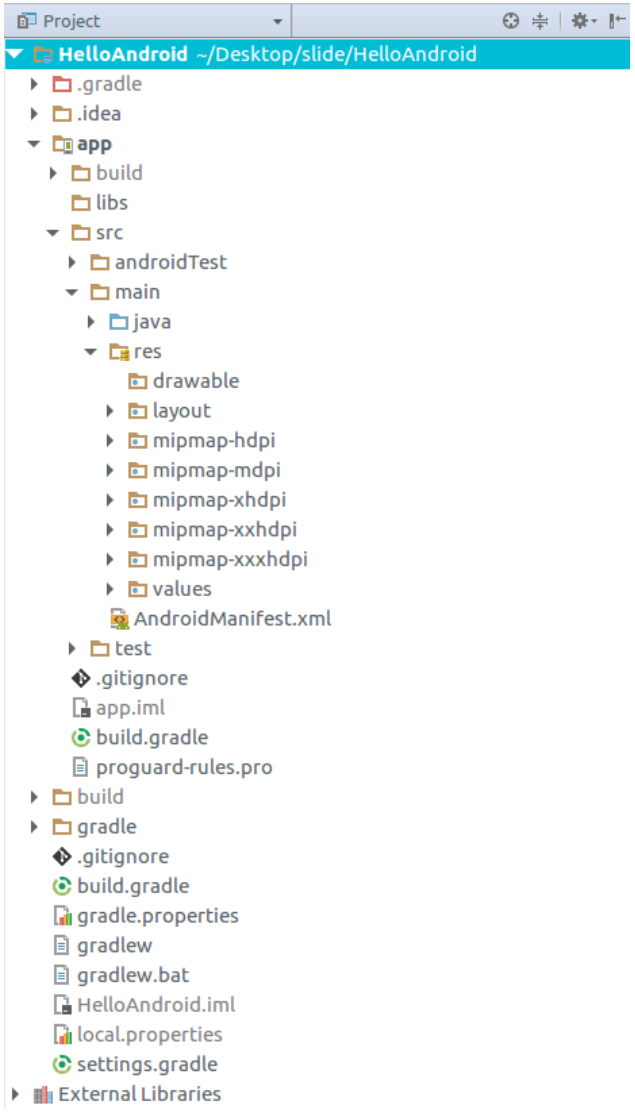
Setup Tools

- **Android SDK**
- **Genymotion (**ต้องการ **VirtualBox)**

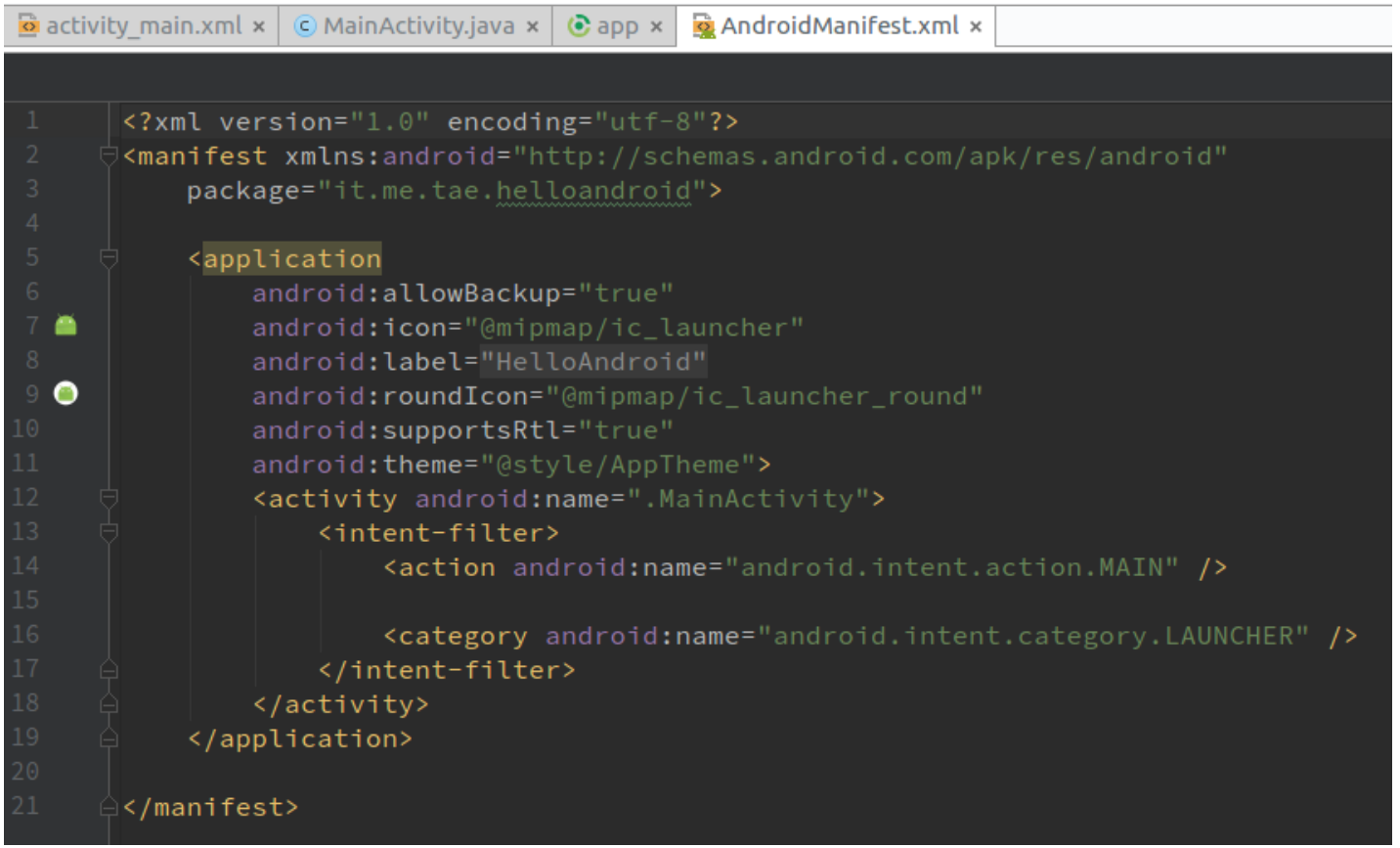
Hello Android

File System Structure

File System Structure



Android Manifest



See more: Manifest file structure

Gradle Build File

activity_main.xml × | MainActivity.java × | app ×

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

android {
    compileSdkVersion 25
    buildToolsVersion "25.0.2"
    defaultConfig {
        applicationId "it.me.tae.helloandroid"
        minSdkVersion 19
        targetSdkVersion 25
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:25.1.0'
    testCompile 'junit:junit:4.12'
}
```

Resources

- **animator -** XML files that define property animations.
- **anim -** XML files that define tween animations.
- **color -** XML files that define a state list of colors.
- **drawable -** Bitmap files (.png, .9.png, .jpg, .gif) or XML files that define drawable resource.
- **mipmap -** Drawable files for different launcher icon densities.
- **layout -** XML files that define a user interface layout.
- **menu -** XML files that define application menus.
- **raw -** Arbitrary files to save in their raw form.
- **values -** XML files that contain simple values, arrays, color, dimens, strings and style.
- **xml -** Arbitrary XML files.

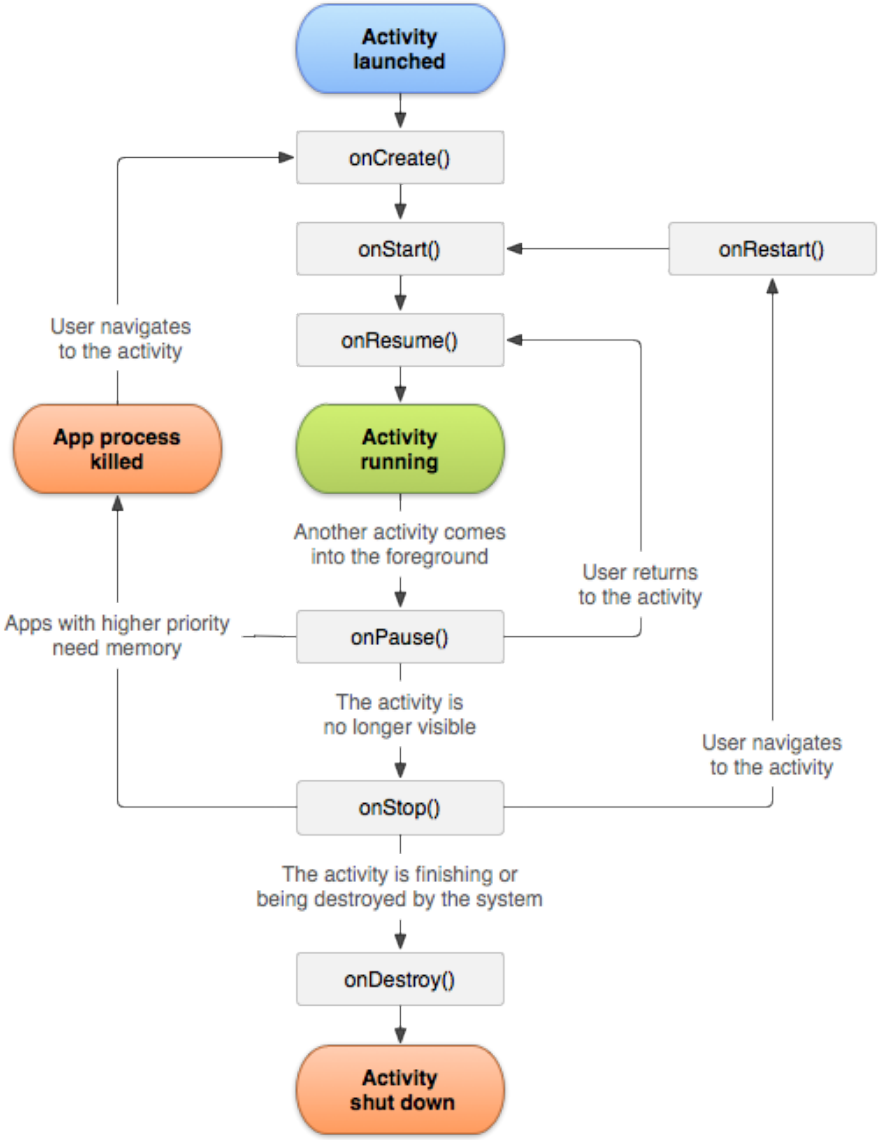
Activity & Fragments

Activity

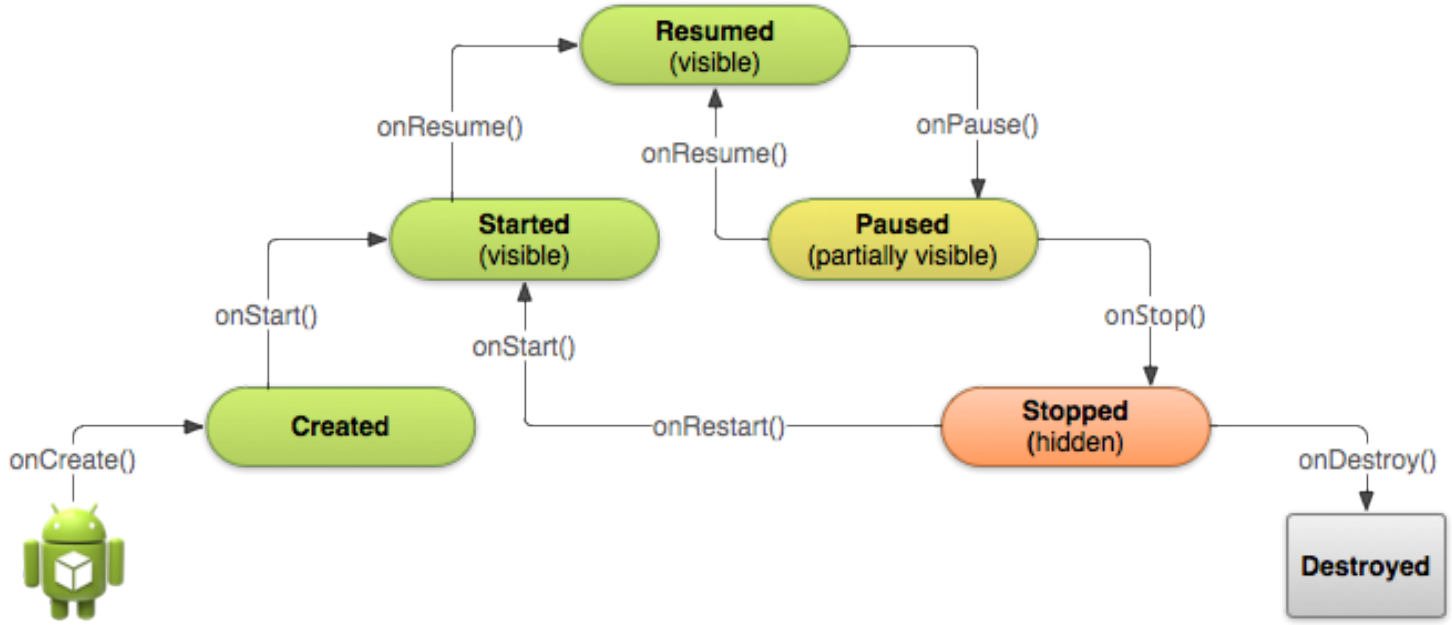
activity_main.xml × MainActivity.java ×

```java
package it.me.tae.lifecycle;

import ...

public class MainActivity extends AppCompatActivity {

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

Activity Lifecycle



Activity Lifecycle



Fragment

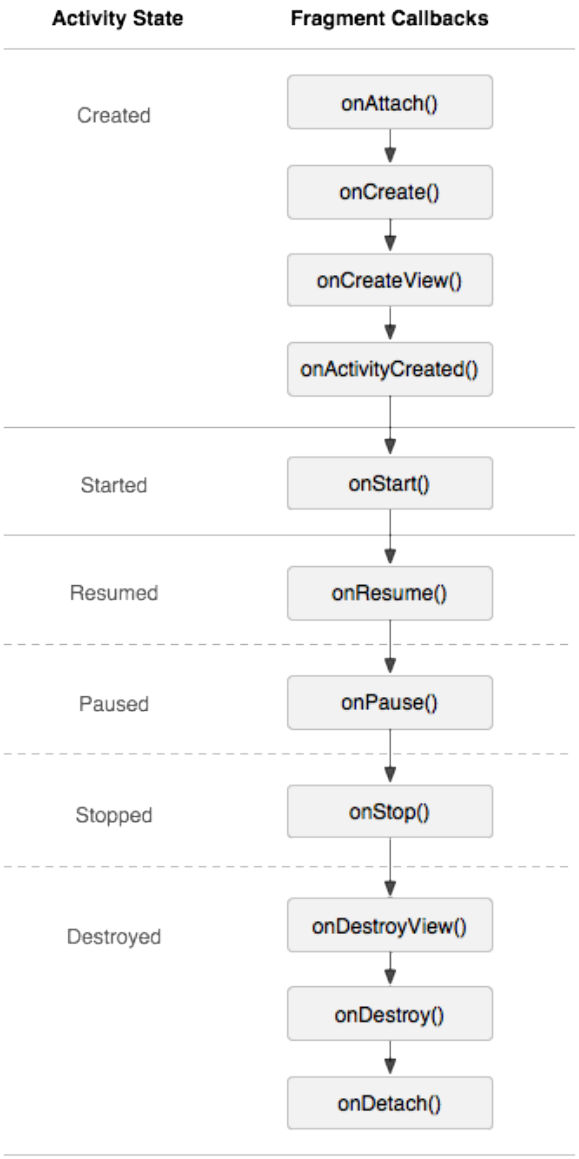
```java
public class MyFragment extends Fragment {
    private static final String ARG_PARAM1 = "param1";
    private static final String ARG_PARAM2 = "param2";

    private String mParam1;
    private String mParam2;

    public static MyFragment newInstance(String param1, String param2) {
        MyFragment fragment = new MyFragment();
        Bundle args = new Bundle();
        args.putString(ARG_PARAM1, param1);
        args.putString(ARG_PARAM2, param2);
        fragment.setArguments(args);
        return fragment;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        if (getArguments() != null) {
            mParam1 = getArguments().getString(ARG_PARAM1);
            mParam2 = getArguments().getString(ARG_PARAM2);
        }
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
        return inflater.inflate(R.layout.fragment_my, container, false);
    }

}
```

Fragment

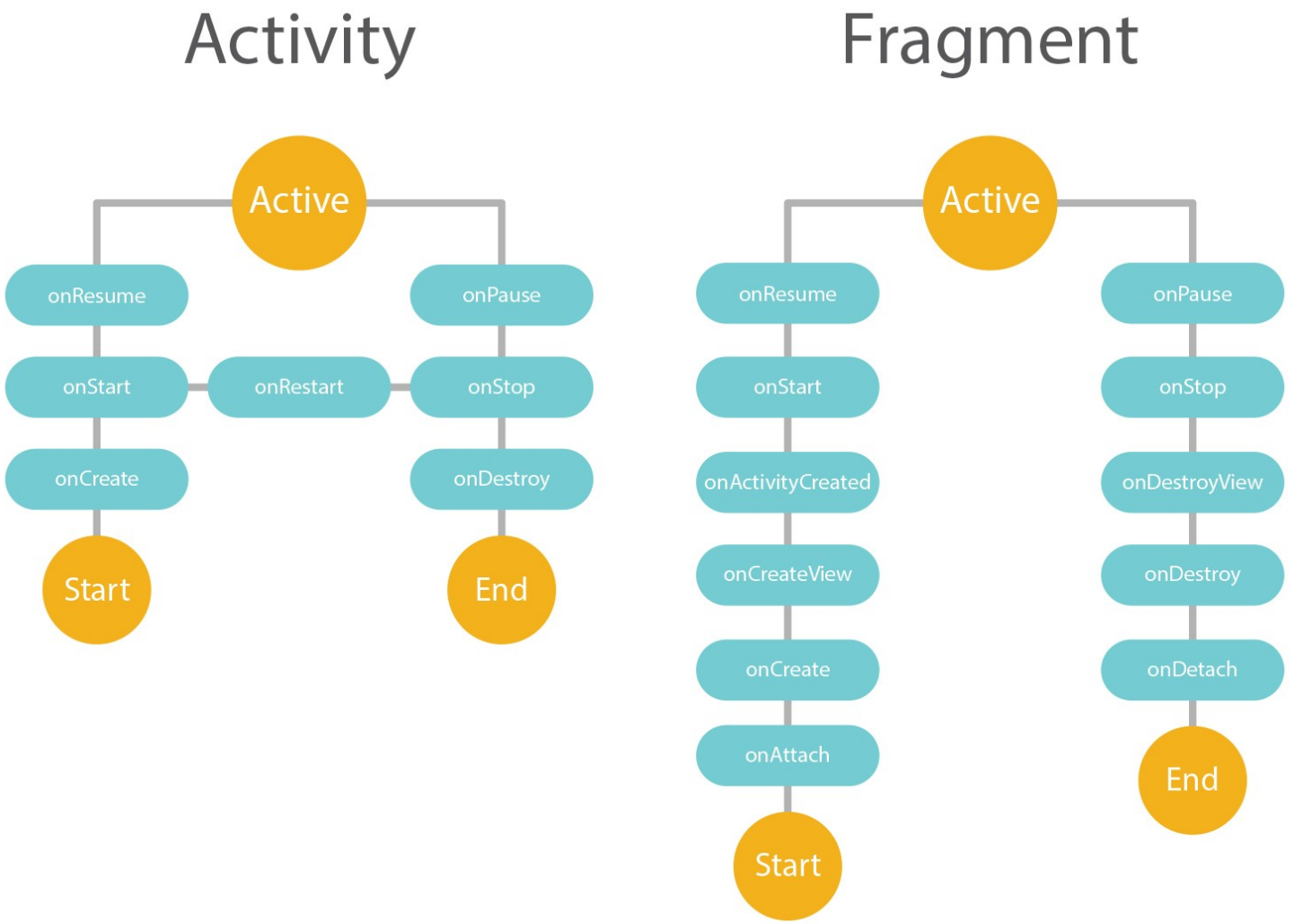
```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="it.me.tae.lifecycle.MainActivity">

    <FrameLayout
        android:id="@+id/container"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>
</RelativeLayout>
```

```java
import ...

public class MainActivity extends AppCompatActivity {

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

        getSupportFragmentManager().beginTransaction()
                .replace(R.id.container, MyFragment.newInstance("param1", "param2"))
                .commit();
    }
}
```

Fragment Lifecycle

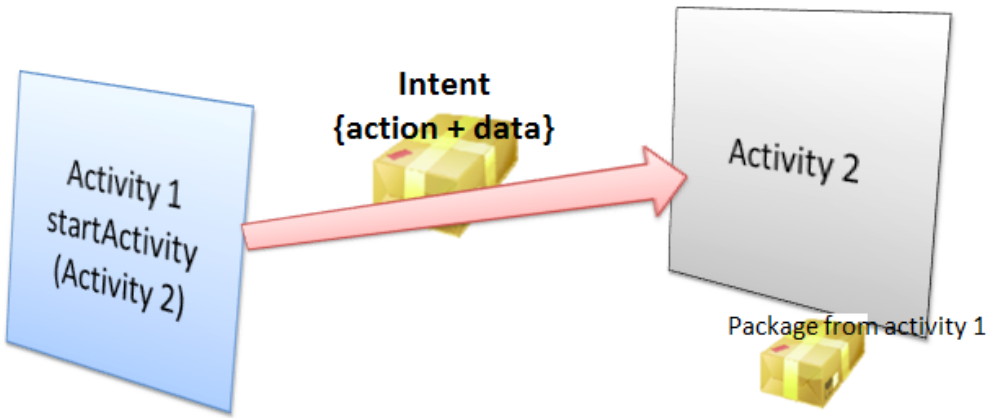


Activity vs Fragment Lifecycle



Intents

Intent


Intent
{action + data}
Activity 1
startActivity
(Activity 2)
Activity 2
Package from activity 1

Intent

- **The primary information contained in an Intent**
  - Component name
  - Action
  - Data
  - Category
  - Extras
  - Flags

Fragmentations

Fragmentations

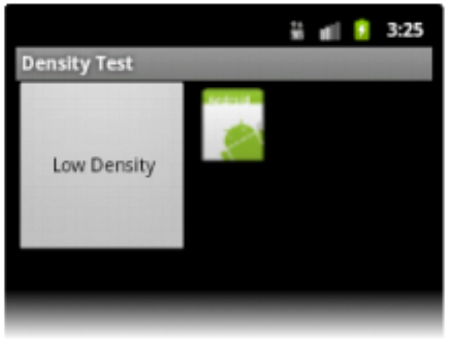
- **Resolution & Display size**
- **Android Versions**

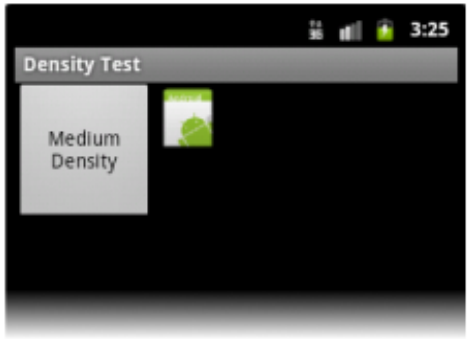
Target version & sizes

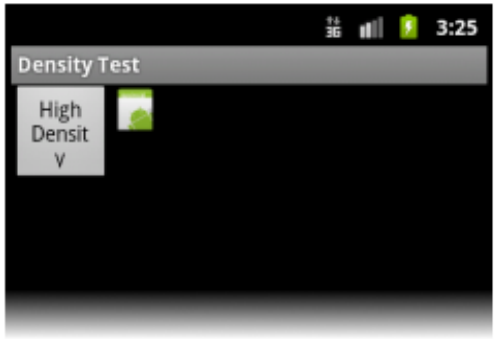
- **drawable-hdpi-v11 – used for android 11 and up**
- **layout-w600dp-land – using in landscape wide at least 600 dp width**

Density independence

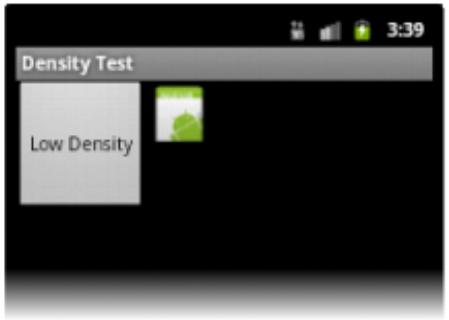
Bad

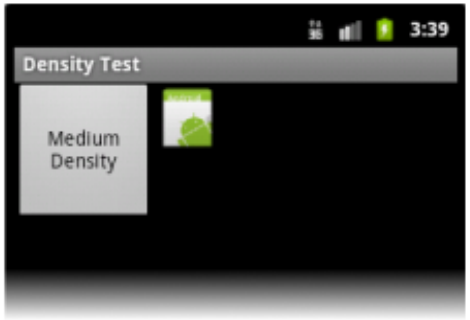


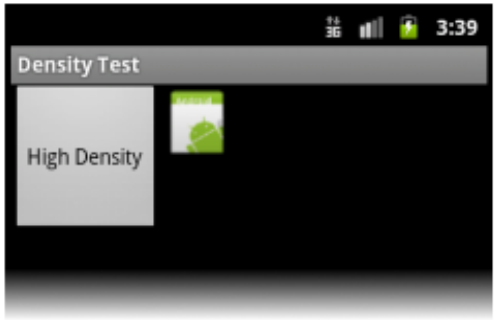




Good

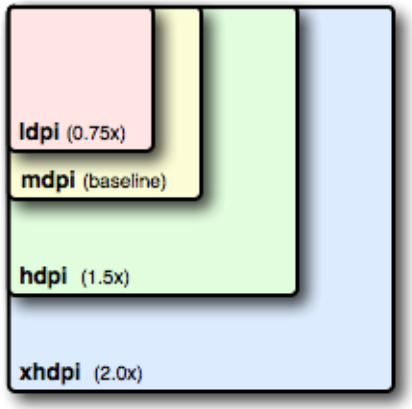






Density independence

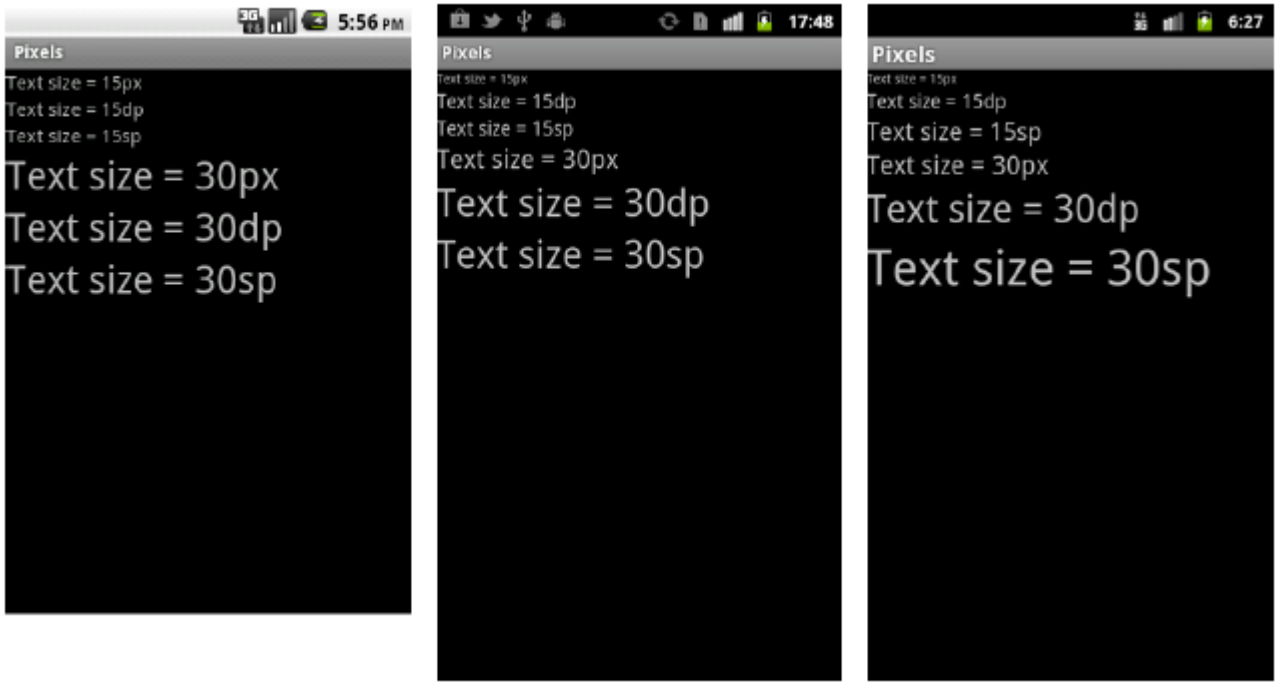
- **density equals**
  - .75 on ldpi (low) (120 dpi)
  - 1.0 on mdpi (medium) (160 dpi; baseline)
  - 1.5 on hdpi (high) (240 dpi)
  - 2.0 on xhdpi (extra-high) (320 dpi)
  - 3.0 on xxhdpi (extra-extra-high) (480 dpi)
  - 4.0 on xxxhdpi (extra-extra-extra-high) (640 dpi)
- **screen sizes** Device Metrics
  - xlarge screens are at least 960dp x 720dp
  - large screens are at least 640dp x 480dp
  - normal screens are at least 470dp x 320dp
  - small screens are at least 426dp x 320dp



Density independence

- **Typical screen widths:**
  - 320dp: a typical phone screen (240x320 ldpi, 320x480 mdpi, 480x800 hdpi, etc).
  - 480dp: a tweener tablet like the Streak (480x800 mdpi).
  - 600dp: a 7” tablet (600x1024 mdpi).
  - 720dp: a 10” tablet (720x1280 mdpi, 800x1280 mdpi, etc).

Scale Independent



Credit: The Big Nerd Ranch Guide

Target version & sizes

```
if (android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
    // call new api
} else {
    // call old api
}
```

```
DisplayMetrics metrics = getResources().getDisplayMetrics();
int dpi = metrics.densityDpi;
float widthDp = metrics.widthPixels / metrics.density;
float heightDp = metrics.heightPixels / metrics.density;
float smallestWidth = Math.min(widthDp, heightDp);
boolean isTablet = smallestWidth >= 600;
```

Android Support Library

- **Support Library Packages**
- **Support Library Features Guide**

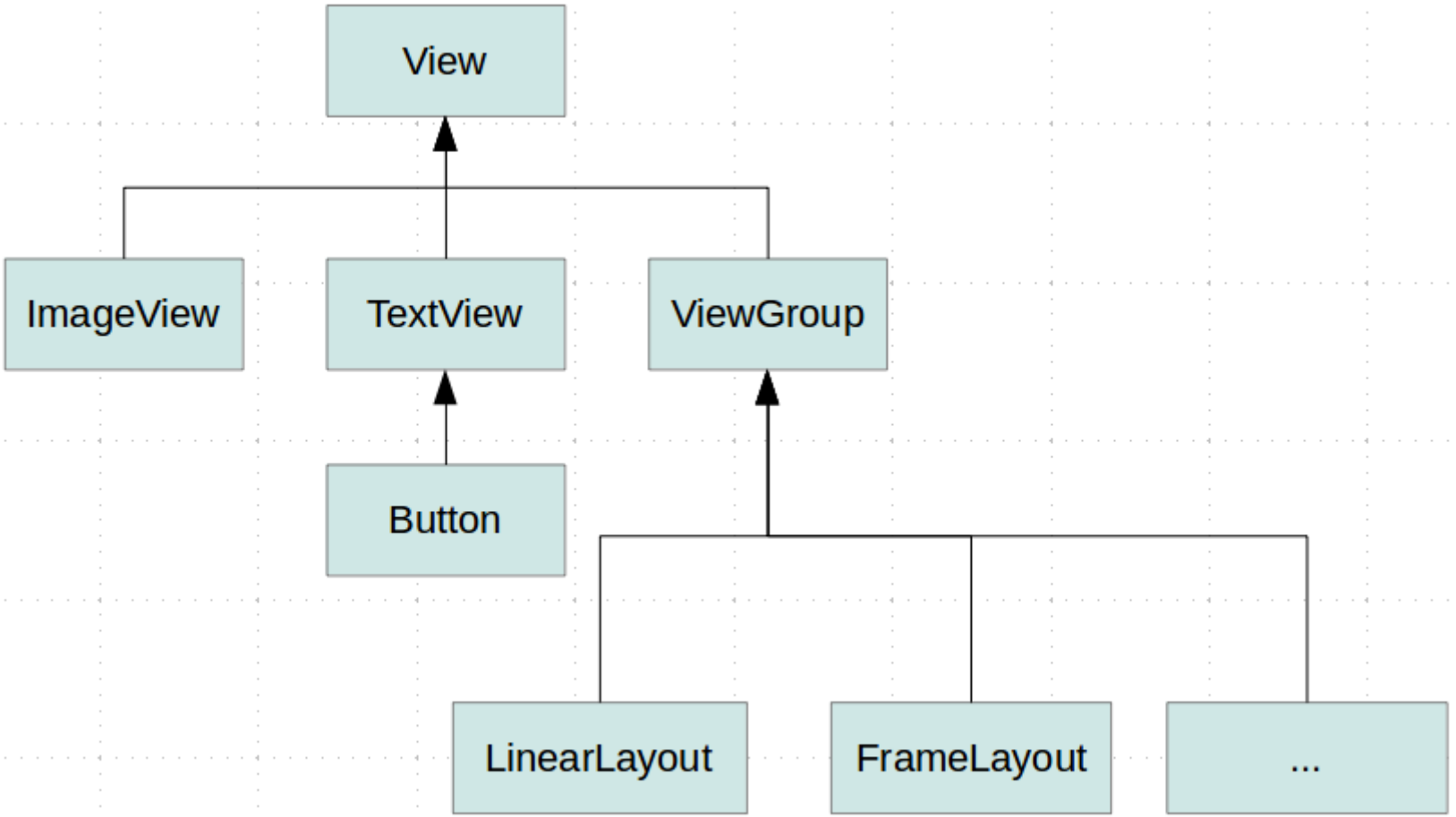
Task & Back Stack

Task & Back Stack

- **Launch mode**
  - standard
  - singleTop
  - singleTask
  - singleInstance

Views & Layouts

Views & Layouts



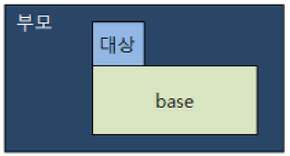
Linear Layout

Example: Linear Layout

Relative Layout

Example: Relative Layout

Relative Layout: position

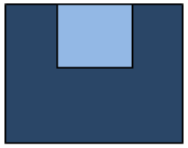



android:layout_alignParentTop

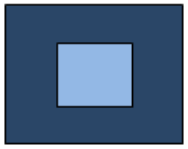

android:layout_alignParentBottom

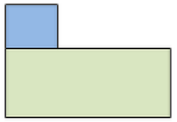

android:layout_alignParentLeft

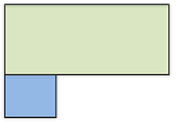

android:layout_alignParentRight


android:layout_centerHorizontal


android:layout_centerVertical


android:layout_centerInParent

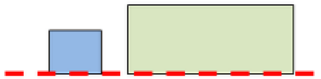

android:layout_above="base"

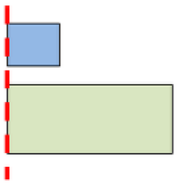

android:layout_below="base"

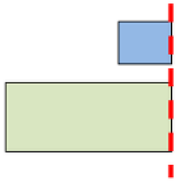

android:layout_alignTop="base"


android:layout_toLeftOf="base"


android:layout_toRightOf="base"

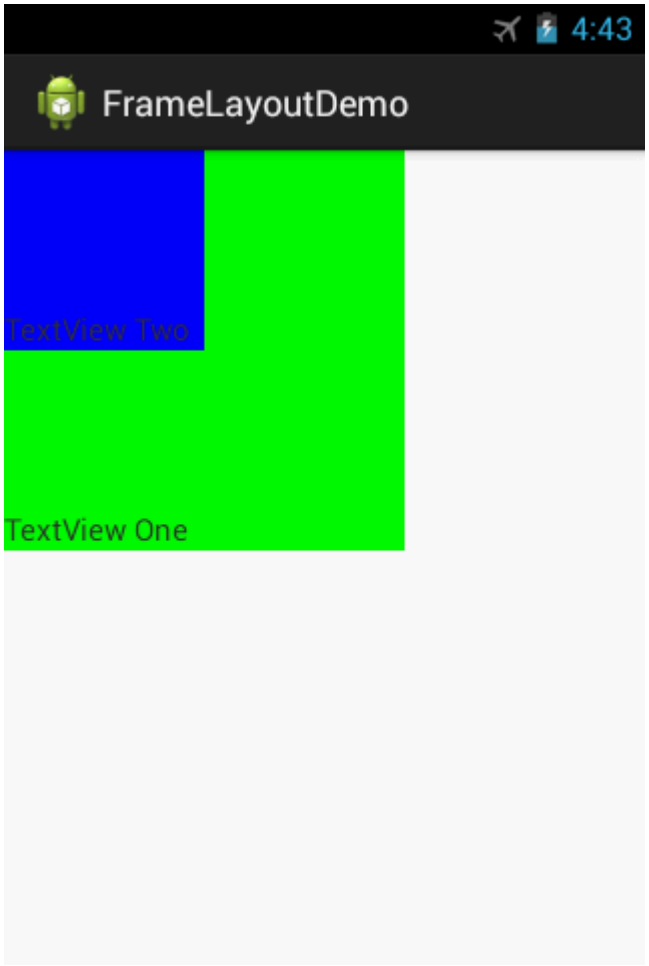

android:layout_alignBottom="base"

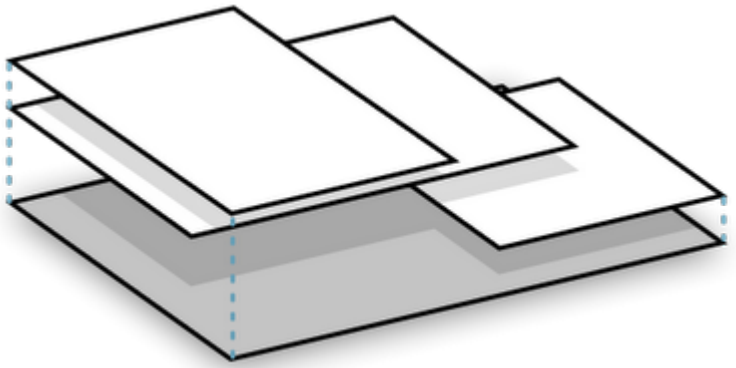

android:layout_alignLeft="base"


android:layout_alignRight="base"

[Credit: Be away]

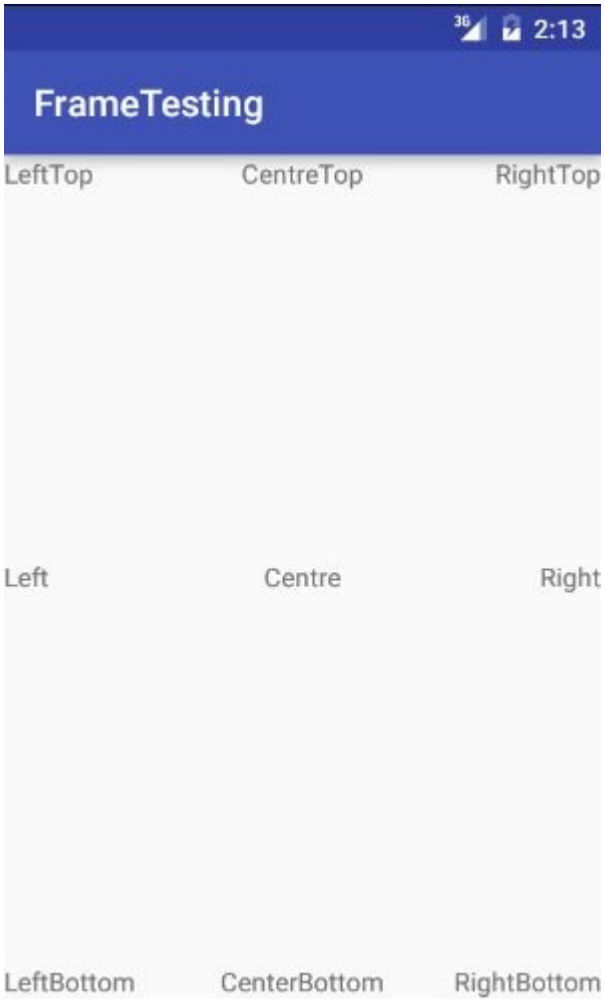
Frame Layout





Credit: androidsubway

Frame Layout: layout_gravity



Credit: AbhiAndroid