## 4. Create New Activity in Android Studio

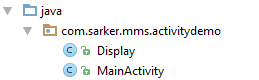
**(ActivityDemo)**

****

**1. AndoidManifests.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.sarker.mms.activitydemo"**>  
  
 <**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"**>  
 <**intent-filter**>  
 <**action android:name="android.intent.action.MAIN"** />  
  
 <**category android:name="android.intent.category.LAUNCHER"** />  
 </**intent-filter**>  
 </**activity**>  
 <**activity android:name=".Display"**></**activity**>  
 </**application**>  
  
</**manifest**>

--------------------------------------------------------------------------------------------



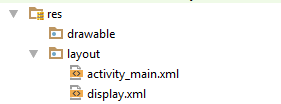
**2. Display.java**

**package** com.sarker.mms.activitydemo;  
  
**import** android.app.Activity;  
**import** android.os.Bundle;  
  
*/\*\*  
 \* Created by user on 20/03/2018.  
 \*/***public class** Display **extends** Activity{  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***display***);  
 }  
}

------------------------------------------------------

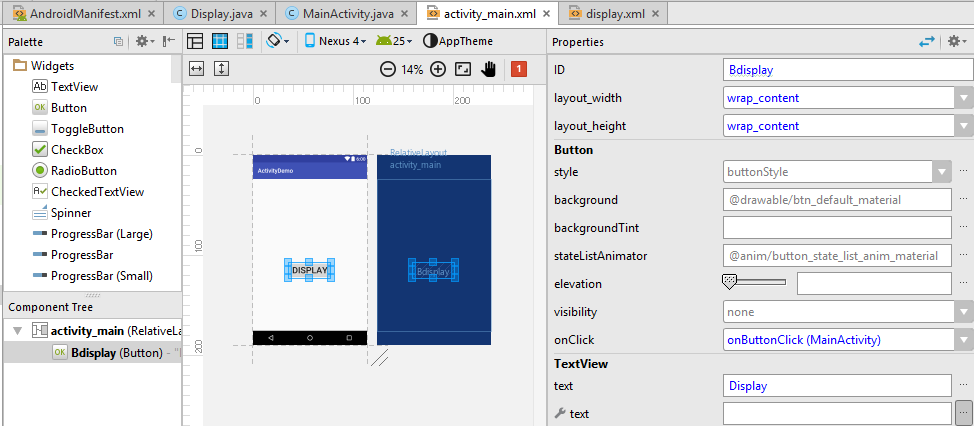
**3. MainActivity.java**

**package** com.sarker.mms.activitydemo;  
  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 }  
 **public void** onButtonClick(View v) {  
 **if** (v.getId() == R.id.***Bdisplay***) {  
 Intent i = **new** Intent(MainActivity.**this**, Display.**class**);  
 startActivity(i);  
 }  
 }}



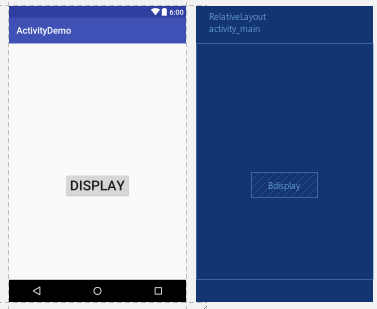
**4. activity\_main.xml**

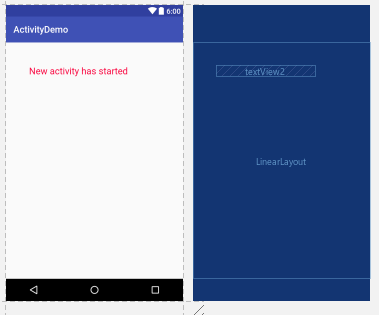
*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/activity\_main"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 tools:context="com.sarker.mms.activitydemo.MainActivity"**>  
  
 <**Button  
 android:text="Display"  
 android:textSize="30sp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentBottom="true"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginBottom="160dp"  
 android:id="@+id/Bdisplay"  
 android:onClick="onButtonClick"** />  
</**RelativeLayout**>



**5. display.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**TextView  
 android:text="New activity has started"  
 android:textColor="#f9043e"  
 android:textSize="20sp"  
 android:layout\_margin="50dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/textView2"** />  
</**LinearLayout**>





**Command explanation**

1. @**Override**

It's a Java annotation that tells the compiler that the method is **overriding** a method from the superclass.

@Override is used to change/modify the functionality of the default function. Basically if we write an override over default functions like oncreate() in our class, the function we wrote will work on creation and not the default one.

Example:

Say in Android MainActivity(sub) class is inherited from AppCompactActivity (super) class. This AppCompactActivity class has a method **onCreate**and in MainActivity class we are overriding it. Hence using the @Override annotation.

1. **AppCompatActivity**

**AppCompatActivity**is a specific type of activity that allows us to use the support library action bar features.

Actually, ActionBarActivity class has been deprecated and change the default Parent Activity class from ActionBarActivity to AppCompatActivity.

1. **android.os.Bundle**

Bundle generally use for passing data between various Activities. It depends on us what type of values we want to pass but bundle can hold all types of values and pass to the new activity

Provides **putType()** and **getType()** methods for storing and retrieving data from it.

1. **android View**

Usually View is used as arguments in methods which act as some kind of listener. For example when we have more than 1 Button in our layout and we set onClick on them, we create a method like this:

public void onClick(View view){

}

1. **Intent**

An Android Intent is an object carrying intent, i.e. a **message**from one component to another component either inside or outside of the application. Intents can communicate messages among any of the three core components of an application -- **Activities, Services, and BroadcastReceivers.**

1. **findViewById(int id)**

This is a method of the View and Activity classes. This method will take a resource Id usually in the form of R.id.mView and will return to us a View object that is a reference to that View. Note that the returned object will nearly always need to be cast to the correct type of View before we can start interacting with it. If a View by the given ID does not exist within the current activity or parent View then we will receive null.

1. **setContentView (R.layout.main).**

For Android App Development setContentView(View) mostly used for Android UserInterface to display the Layout created thorugh XML or the Dynamically created layout with data.

1. **setTextColor(Color.*parseColor*(""))**

When setting the colors for e.g. a TextView we can use the settextcolor from either a color or resource.

public void setTextColor (int color)

Sets the text color for all the states (normal, selected, focused) to be this color.

There are several ways to do this. To set the colors from your resource you will have to add getResources() like this where text is the TextView.

text.setTextColor(getResources().getColor(R.color.errorColor));

To set a general color you can do this:

text.setTextColor(Color.RED);

Parse the color from a hash:

text.setTextColor(Color.parseColor("#FFFFFF"));

You can also use a RGB color with or without alpha:

text.setTextColor(Color.rgb(200,0,0)); text.setTextColor(Color.argb(0,200,0,0));