Course: 305-02: Mobile Application Development – 1

Unit-5: Android Widgets (UI)

5.1 Hiding Title bar

Android Hide Title Bar and Full Screen Example

In this example, we are going to explain how to hide the title bar and how to display content in full screen mode.

The **requestWindowFeature(Window.FEATURE_NO_TITLE)** method of Activity must be called to hide the title. But, it must be coded before the setContentView method.

Code that hides title bar of activity

The getSupportActionBar() method is used to retrieve the instance of ActionBar class. Calling the hide() method of ActionBar class hides the title bar.

```
1. requestWindowFeature(Window.FEATURE_NO_TITLE);//will hide the titl
   e
2. getSupportActionBar().hide(); //hide the title bar
```

Code that enables full screen mode of activity

The **setFlags()** method of Window class is used to display content in full screen mode. You need to pass the **WindowManager.LayoutParams.FLAG_FULLSCREEN** constant in the setFlags method.

```
1. this.getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSC
   REEN,
2. WindowManager.LayoutParams.FLAG_FULLSCREEN); //show the activit
   y in full screen
```

Android Hide Title Bar and Full Screen Example

Let's see the full code to hide the title bar in android.

activity_main.xml

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="first.com.hidetitlebar.MainActivity">

    <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello World!"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintLeft_toLeftOf="parent"
            app:layout_constraintRight_toRightOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

    </android.support.constraint.ConstraintLayout>
```

Activity class

File: MainActivity.java

```
package first.com.hidetitlebar;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Window;
import android.view.WindowManager;

public class MainActivity extends AppCompatActivity {

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
```

```
13.             requestWindowFeature(Window.FEATURE_NO_TITLE); //will
    hide the title
14.             getSupportActionBar().hide(); // hide the title bar
15.             this.getWindow().setFlags(WindowManager.LayoutParams.FL
    AG_FULLSCREEN,
16.                     WindowManager.LayoutParams.FLAG_FULLSCREEN); //
    enable full screen
17.             setContentView(R.layout.activity_main);
18.
19.
20.         }
21.     }
```

Try custom settings with the title bar in MainActivity.java file:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    requestWindowFeature(Window.FEATURE_NO_TITLE);

    getSupportActionBar().setTitle("SDJ International College");

    getSupportActionBar().hide();


    this.getWindow().setFlags(WindowManager.LayoutParams.FLAG_FUL
    LSCREEN,WindowManager.LayoutParams.FLAG_FULLSCREEN);

    setContentView(R.layout.activity_main);
}
```

5.2 screen Orientation (Portrait, Landscape)

Android Screen Orientation Example

The **screenOrientation** is the attribute of activity element. The orientation of android activity can be portrait, landscape, sensor, unspecified etc. You need to define it in the AndroidManifest.xml file.

Syntax:

```
<activity android:name="package_name.Your_ActivityName"
    android:screenOrientation="orirntation_type">
</activity>
```

```
<activity android:name=" example.com.screenorientation.MainActivity"
    android:screenOrientation="portrait">
</activity>
```

```
<activity android:name=".SecondActivity"
    android:screenOrientation="landscape">
</activity>
```

The common values for screenOrientation attribute are as follows:

Value	Description
unspecified	It is the default value. In such case, system chooses the orientation.
portrait	taller not wider
landscape	wider not taller
sensor	orientation is determined by the device orientation sensor.

Android Portrait and Landscape mode screen orientation example

In this example, we will create two activities of different screen orientation. The first activity (MainActivity) will be as "portrait" orientation and second activity (SecondActivity) as "landscape" orientation type.

activity_main.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <android.support.constraint.ConstraintLayout xmlns:android="http:/
   /schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     tools:context="example.com.screenorientation.MainActivity">
8.
9.
10.         <Button
11.             android:id="@+id/button1"
12.             android:layout_width="wrap_content"
13.             android:layout_height="wrap_content"
14.             android:layout_marginBottom="8dp"
15.             android:layout_marginTop="112dp"
16.             android:onClick="onClick"
17.             android:text="Launch next activity"
18.             app:layout_constraintBottom_toBottomOf="parent"
19.             app:layout_constraintEnd_toEndOf="parent"
20.             app:layout_constraintHorizontal_bias="0.612"
21.             app:layout_constraintStart_toStartOf="parent"
22.             app:layout_constraintTop_toBottomOf="@+id/editText1"
23.             app:layout_constraintVertical_bias="0.613" />
24.
25.         <TextView
26.             android:id="@+id/editText1"
27.             android:layout_width="wrap_content"
28.             android:layout_height="wrap_content"
29.             android:layout_centerHorizontal="true"
```

```
30.             android:layout_marginEnd="8dp"
31.             android:layout_marginStart="8dp"
32.             android:layout_marginTop="124dp"
33.             android:ems="10"
34.             android:textSize="22dp"
35.             android:text="This activity is portrait orientation"
36.             app:layout_constraintEnd_toEndOf="parent"
37.             app:layout_constraintHorizontal_bias="0.502"
38.             app:layout_constraintStart_toStartOf="parent"
39.             app:layout_constraintTop_toTopOf="parent" />
40.     </android.support.constraint.ConstraintLayout>
```

Activity class

File: MainActivity.java

```
1.  package example.com.screenorientation;
2.
3.  import android.content.Intent;
4.  import android.support.v7.app.AppCompatActivity;
5.  import android.os.Bundle;
6.  import android.view.View;
7.  import android.widget.Button;
8.
9.  public class MainActivity extends AppCompatActivity {
10.
11.         Button button1;
12.         @Override
13.         protected void onCreate(Bundle savedInstanceState) {
14.             super.onCreate(savedInstanceState);
15.             setContentView(R.layout.activity_main);
16.
17.             button1=(Button)findViewById(R.id.button1);
18.         }
19.         public void onClick(View v) {
```

```
20.                 Intent intent = new Intent(MainActivity.this,SecondActivity.cl
    ass);
21.                 startActivity(intent);
22.             }
23.         }
```

activity_second.xml

File: activity_second.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <android.support.constraint.ConstraintLayout xmlns:android="http:/
   /schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     tools:context="example.com.screenorientation.SecondActivity">
8.
9.     <TextView
10.             android:id="@+id/textView"
11.             android:layout_width="wrap_content"
12.             android:layout_height="wrap_content"
13.             android:layout_marginEnd="8dp"
14.             android:layout_marginStart="8dp"
15.             android:layout_marginTop="180dp"
16.             android:text="this is landscape orientation"
17.             android:textSize="22dp"
18.             app:layout_constraintEnd_toEndOf="parent"
19.             app:layout_constraintHorizontal_bias="0.502"
20.             app:layout_constraintStart_toStartOf="parent"
21.             app:layout_constraintTop_toTopOf="parent" />
22.     </android.support.constraint.ConstraintLayout>
```

SecondActivity class

File: SecondActivity.java

```
1. package example.com.screenorientation;
2. 
3. import android.support.v7.app.AppCompatActivity;
4. import android.os.Bundle;
5. 
6. public class SecondActivity extends AppCompatActivity {
7. 
8.     @Override
9.     protected void onCreate(Bundle savedInstanceState) {
10.                 super.onCreate(savedInstanceState);
11.                 setContentView(R.layout.activity_second);
12. 
13.             }
14.         }
```

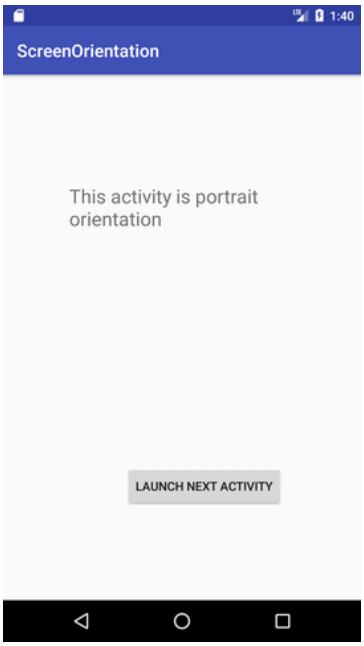
AndroidManifest.xml

File: AndroidManifest.xml

In AndroidManifest.xml file add the screenOrientation attribute in activity and provides its orientation. In this example, we provide "portrait" orientation for MainActivity and "landscape" for SecondActivity.

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <manifest xmlns:android="http://schemas.android.com/apk/res/andro
   id"
3.     package="example.com.screenorientation">
4. 
5.     <application
6.         android:allowBackup="true"
7.         android:icon="@mipmap/ic_launcher"
8.         android:label="@string/app_name"
9.         android:roundIcon="@mipmap/ic_launcher_round"
10.                 android:supportsRtl="true"
11.                 android:theme="@style/AppTheme">
12.                 <activity
```

```
13.             android:name="example.com.screenorientation.MainActivit
    y"
14.             android:screenOrientation="portrait">
15.             <intent-filter>
16.                 <action android:name="android.intent.action.MAIN" />
17.
18.                 <category android:name="android.intent.category.LAUN
    CHER" />
19.             </intent-filter>
20.         </activity>
21.         <activity android:name=".SecondActivity"
22.             android:screenOrientation="landscape">
23.         </activity>
24.     </application>
25.
26. </manifest>
```



Change Screen Orientation Programmatically.

Activity class

File: MainActivity.java

```
public class MainActivity extends AppCompatActivity {

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

    public void onPotraitButtonClick(View view) {

setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT
);
    }

    public void onLandscapeButtonClick(View view) {

setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAP
E);
    }

    public void onSensorButtonClick(View view) {

setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_SENSOR);
    }
}
```

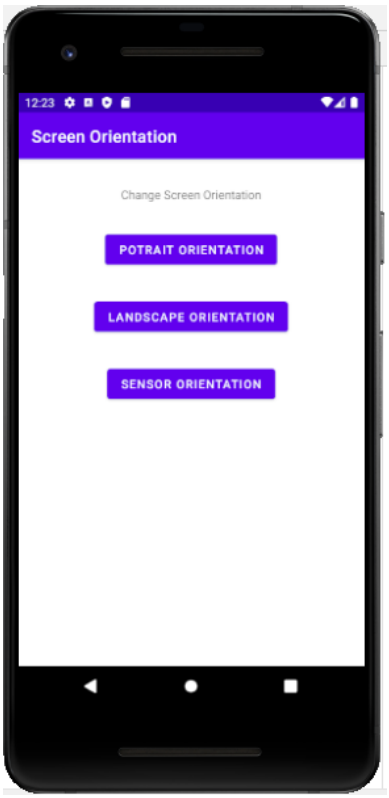
Layout File

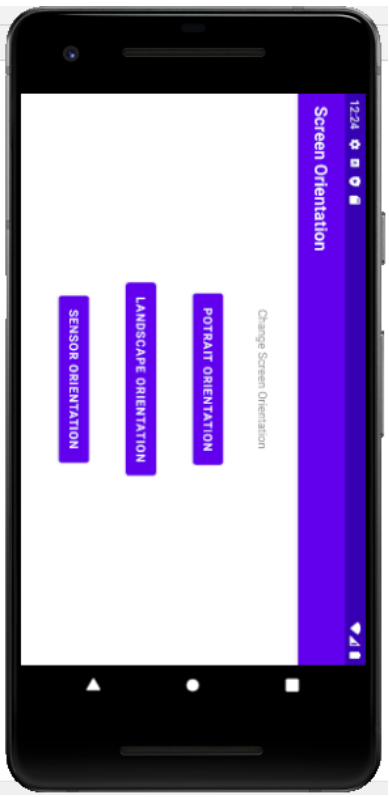
File: activity_main.xml

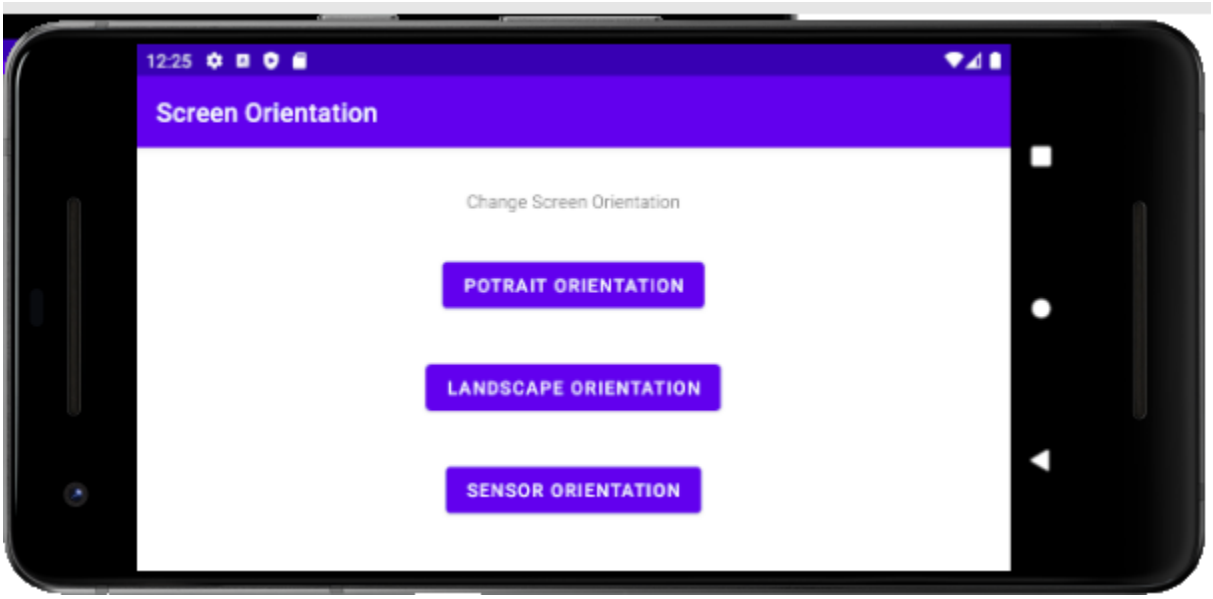
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
```

```
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:text="Change Screen Orientation"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:onClick="onPotraitButtonClick"
        android:text="Potrait Orientation"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView2" />

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:onClick="onLandscapeButtonClick"
        android:text="Landscape Orientation"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/button" />

    <Button
        android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:onClick="onSensorButtonClick"
        android:text="Sensor Orientation"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/button2" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Output:


12:23
Screen Orientation
Change Screen Orientation
POTRAIT ORIENTATION
LANDSCAPE ORIENTATION
SENSOR ORIENTATION

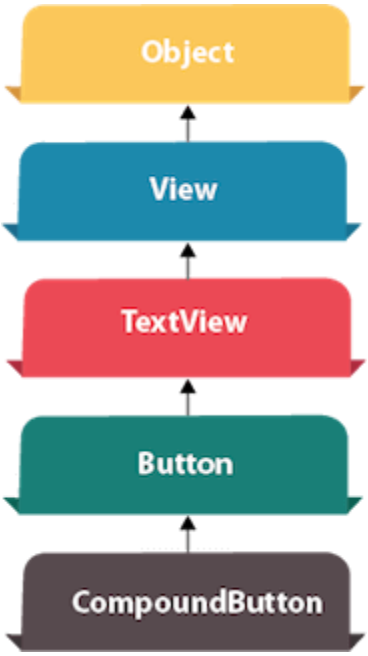

12:24
Screen Orientation
SENSOR ORIENTATION
LANDSCAPE ORIENTATION
POTRAIT ORIENTATION
Change Screen Orientation


12:25
Screen Orientation
Change Screen Orientation
POTRAIT ORIENTATION
LANDSCAPE ORIENTATION
SENSOR ORIENTATION

5.3 Form Widget Palette

5.3.1 Placing text fields and Button

5.3.2 Button onClick event



Android Button Example

Android Button represents a push-button. The android.widget.Button is subclass of TextView class and CompoundButton is the subclass of Button class.

There are different types of buttons in android such as RadioButton, ToggleButton, CompoundButton etc.

Android Button Example with Listener

Here, we are going to create two textfields and one button for sum of two numbers. If user clicks button, sum of two input values is displayed on the Toast.

We can perform action on button using different types such as calling listener on button or adding onClick property of button in activity's xml file.

```
1. button.setOnClickListener(new View.OnClickListener() {
2.             @Override
3.             public void onClick(View view) {
4.                //code
5.             }
6. });
```

```
1. <Button
2.         android:onClick="methodName"
3. />
```

Drag the component or write the code for UI in activity_main.xml

First of all, drag 2 textfields from the Text Fields palette and one button from the Form Widgets palette as shown in the following figure.

The generated code for the ui components will be like this:

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
```

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example..com.sumoftwonumber.MainActivity">

    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="61dp"
        android:ems="10"
        android:inputType="number"
        tools:layout_editor_absoluteX="84dp"
        tools:layout_editor_absoluteY="53dp" />

    <EditText
        android:id="@+id/editText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editText1"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="32dp"
        android:ems="10"
        android:inputType="number"
        tools:layout_editor_absoluteX="84dp"
        tools:layout_editor_absoluteY="127dp" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editText2"
        android:layout_centerHorizontal="true"
```

```
            android:layout_marginTop="109dp"
            android:text="ADD"
            tools:layout_editor_absoluteX="148dp"
            tools:layout_editor_absoluteY="266dp" />
    </RelativeLayout>
```

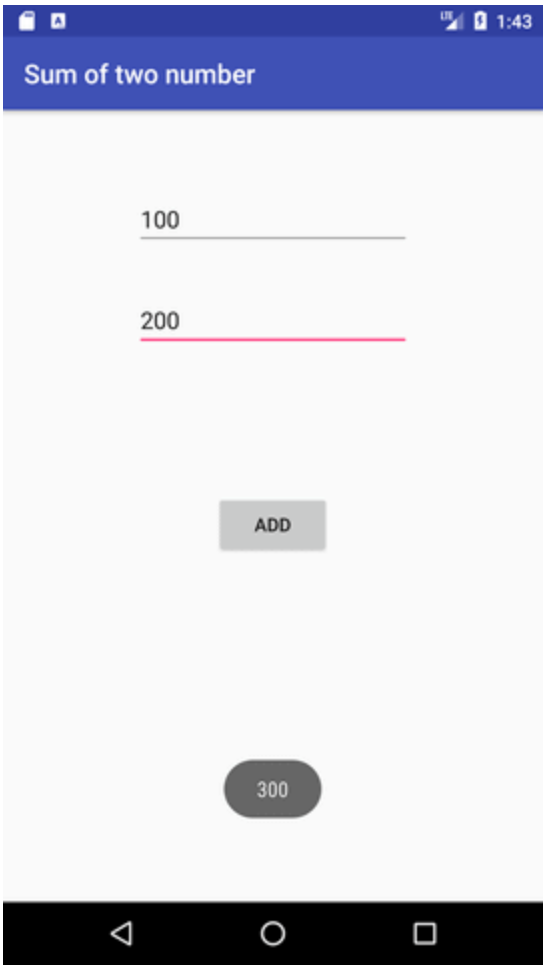
Activity class

Now write the code to display the sum of two numbers.

File: MainActivity.java

```
package example..com.sumoftwonumber;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

    public class MainActivity extends AppCompatActivity {
        private EditText edittext1, edittext2;
        private Button buttonSum;

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

            addListenerOnButton();
        }

        public void addListenerOnButton() {
```

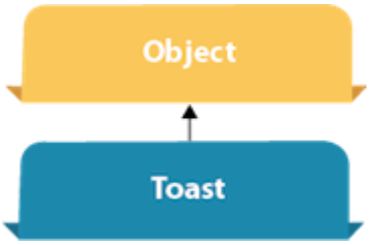
```
23.             edittext1 = (EditText) findViewById(R.id.editText1);
24.             edittext2 = (EditText) findViewById(R.id.editText2);
25.             buttonSum = (Button) findViewById(R.id.button);
26.
27.             buttonSum.setOnClickListener(new View.OnClickListener() {
28.                 @Override
29.                 public void onClick(View view) {
30.                     String value1=edittext1.getText().toString();
31.                     String value2=edittext2.getText().toString();
32.                     int a=Integer.parseInt(value1);
33.                     int b=Integer.parseInt(value2);
34.                     int sum=a+b;
35.                     Toast.makeText(getApplicationContext(),String.valueOf(s
    um), Toast.LENGTH_LONG).show();
36.                 }
37.             });
38.         }
39.     }
```


1:43
Sum of two number
100
200
ADD
300

5.4 Displaying Notification:

5.4.1 Toast Class

5.4.2 Displaying message on Toast



Android Toast Example

Andorid Toast can be used to display information for the short period of time. A toast contains message to be displayed quickly and disappears after sometime.

The android.widget.Toast class is the subclass of java.lang.Object class.

You can also create custom toast as well for example toast displaying image. You can visit next page to see the code for custom toast.

Toast class

Toast class is used to show notification for a particular interval of time. After sometime it disappears. It doesn't block the user interaction.

Constants of Toast class

There are only 2 constants of Toast class which are given below.

Constant	Description
public static final int LENGTH_LONG	displays view for the long duration of time.
public static final int LENGTH_SHORT	displays view for the short duration of time.

Methods of Toast class

The widely used methods of Toast class are given below.

Method	Description
public static Toast makeText(Context context, CharSequence text, int duration)	makes the toast containing text and duration.
public void show()	displays toast.
public void setMargin (float horizontalMargin, float verticalMargin)	changes the horizontal and vertical margin difference.

Android Toast Example

```
1. Toast.makeText(getApplicationContext(),"Hello Javatpoint",Toast.LENGT
   H_SHORT).show();
```

Another code:

```
1. Toast toast=Toast.makeText(getApplicationContext(),"Hello Javatpoint",To
   ast.LENGTH_SHORT);
2. toast.setMargin(50,50);
3. toast.show();
```

Here, getApplicationContext() method returns the instance of Context.

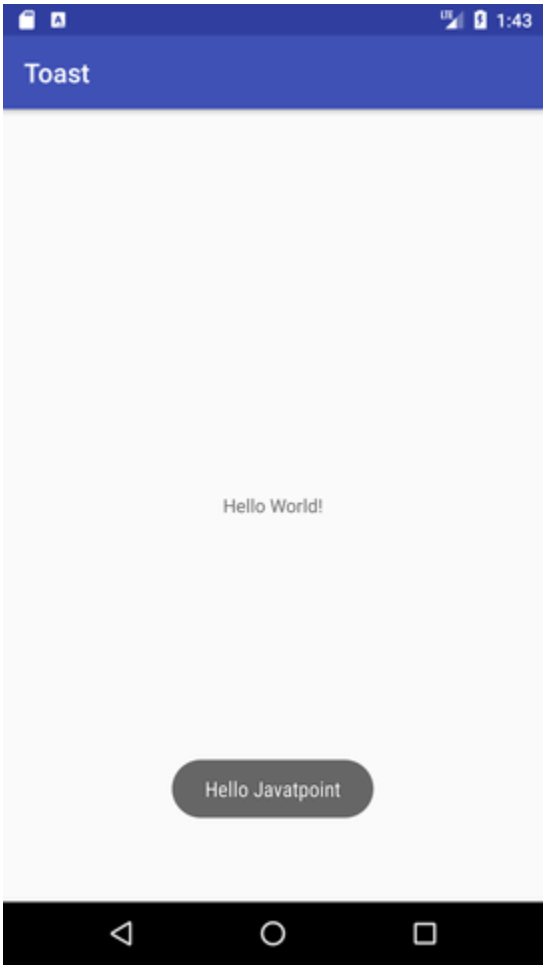
Full code of activity class displaying Toast

Let's see the code to display the toast.

File: MainActivity.java

```
1. package example..com.toast;
2.
3. import android.support.v7.app.AppCompatActivity;
4. import android.os.Bundle;
5. import android.widget.Toast;
6.
```

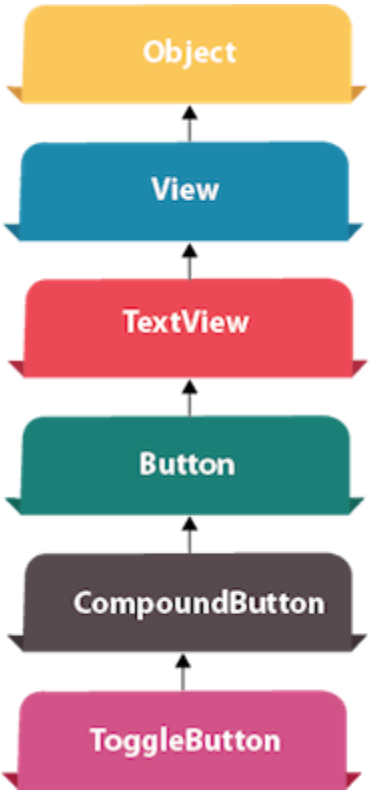
```
7.  public class MainActivity extends AppCompatActivity {
8.
9.      @Override
10.         protected void onCreate(Bundle savedInstanceState) {
11.             super.onCreate(savedInstanceState);
12.             setContentView(R.layout.activity_main);
13.
14.             //Displaying Toast with Hello Javatpoint message
15.             Toast.makeText(getApplicationContext(),"Hello Javatpoint",To
    ast.LENGTH_SHORT).show();
16.         }
17.     }
```



5.5 ToggleButton:

5.5.1 ToggleButton Attributes:(textOff, textOn)

5.5.2 Event methods : getTextOff(), getTextOn(), setChecked()



Android ToggleButton Example

Android Toggle Button can be used to display checked/unchecked (On/Off) state on the button.

It is beneficial if user have to change the setting between two states. It can be used to On/Off Sound, Wifi, Bluetooth etc.

Since Android 4.0, there is another type of toggle button called *switch* that provides slider control.

Android ToggleButton and Switch both are the subclasses of CompoundButton class.

Android ToggleButton class

ToggleButton class provides the facility of creating the toggle button.

XML Attributes of ToggleButton class

The 3 XML attributes of ToggleButton class.

XML Attribute	Description
android:disabledAlpha	The alpha to apply to the indicator when disabled.
android:textOff	The text for the button when it is not checked.
android:textOn	The text for the button when it is checked.

Methods of ToggleButton class

The widely used methods of ToggleButton class are given below.

Method	Description
CharSequence getTextOff()	Returns the text when button is not in the checked state.
CharSequence getTextOn()	Returns the text for when button is in the checked state.
void setChecked(boolean checked)	Changes the checked state of this button.

Android ToggleButton Example

activity_main.xml

Drag two toggle button and one button for the layout. Now the activity_main.xml file will look like this:

File: activity_main.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <android.support.constraint.ConstraintLayout xmlns:android="http:/
   /schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     tools:context="example..com.togglebutton.MainActivity">
8.
9.     <ToggleButton
10.            android:id="@+id/toggleButton"
11.            android:layout_width="wrap_content"
12.            android:layout_height="wrap_content"
13.            android:layout_marginLeft="8dp"
```

```
            android:layout_marginTop="80dp"
            android:text="ToggleButton"
            android:textOff="Off"
            android:textOn="On"
            app:layout_constraintEnd_toStartOf="@+id/toggleButton2"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <ToggleButton
            android:id="@+id/toggleButton2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginRight="60dp"
            android:layout_marginTop="80dp"
            android:text="ToggleButton"
            android:textOff="Off"
            android:textOn="On"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <Button
            android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginBottom="144dp"
            android:layout_marginLeft="148dp"
            android:text="Submit"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintStart_toStartOf="parent" />
    </android.support.constraint.ConstraintLayout>
```

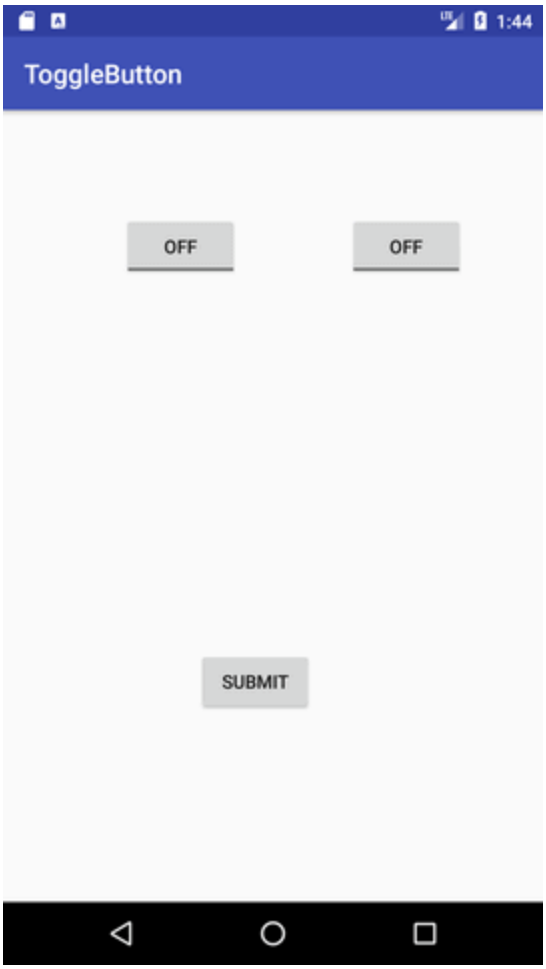
Activity class

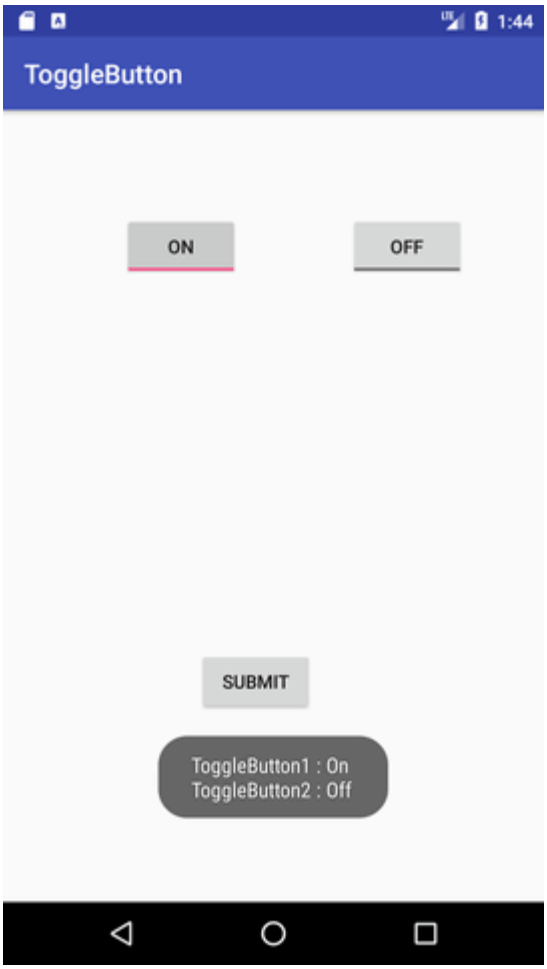
Let's write the code to check which toggle button is ON/OFF.

File: MainActivity.java

```
1.  package example..com.togglebutton;
2.
3.  import android.support.v7.app.AppCompatActivity;
4.  import android.os.Bundle;
5.  import android.view.View;
6.  import android.widget.Button;
7.  import android.widget.Toast;
8.  import android.widget.ToggleButton;
9.
10.         public class MainActivity extends AppCompatActivity {
11.             private ToggleButton toggleButton1, toggleButton2;
12.             private Button buttonSubmit;
13.             @Override
14.             protected void onCreate(Bundle savedInstanceState) {
15.                 super.onCreate(savedInstanceState);
16.                 setContentView(R.layout.activity_main);
17.
18.                 addListenerOnButtonClick();
19.             }
20.
21.             public void addListenerOnButtonClick(){
22.                 //Getting the ToggleButton and Button instance from the lay
    out xml file
23.                 toggleButton1=(ToggleButton)findViewById(R.id.toggleButton);

24.                 toggleButton2=(ToggleButton)findViewById(R.id.toggleButton2
    );
25.                 buttonSubmit=(Button)findViewById(R.id.button);
26.
27.                 //Performing action on button click
28.                 buttonSubmit.setOnClickListener(new View.OnClickListener()
    {
29.
```

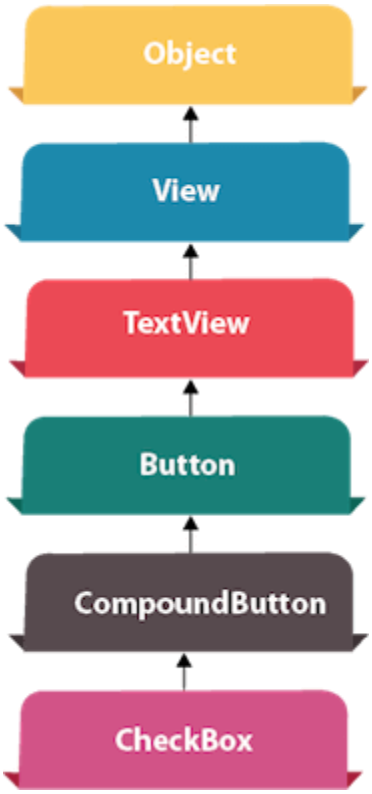
```
30.                 @Override
31.                 public void onClick(View view) {
32.                     StringBuilder result = new StringBuilder();
33.                     result.append("ToggleButton1 : ").append(toggleButton1.
    getText());
34.                     result.append("\nToggleButton2 : ").append(toggleButto
    n2.getText());
35.                     //Displaying the message in toast
36.                     Toast.makeText(getApplicationContext(), result.toString()
    ,Toast.LENGTH_LONG).show();
37.                 }
38.
39.             });
40.
41.         }
42.     }
```


1:44
ToggleButton
OFF
OFF
SUBMIT


1:44
ToggleButton
ON
OFF
SUBMIT
ToggleButton1 : On
ToggleButton2 : Off

5.6 CheckBox:

5.6.1 Event methods: isChecked(), setChecked()



Android CheckBox Example

Android CheckBox is a type of two state button either checked or unchecked.

There can be a lot of usage of checkboxes. For example, it can be used to know the hobby of the user, activate/deactivate the specific action etc.

Android CheckBox class is the subclass of CompoundButton class.

Android CheckBox class

The android.widget.CheckBox class provides the facility of creating the CheckBoxes.

Methods of CheckBox class

There are many inherited methods of View, TextView, and Button classes in the CheckBox class. Some of them are as follows:

Method	Description
public boolean isChecked()	Returns true if it is checked otherwise false.
public void setChecked(boolean status)	Changes the state of the CheckBox.

Android CheckBox Example

activity_main.xml

Drag the three checkboxes and one button for the layout. Now the activity_main.xml file will look like this:

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://
/schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example..com.checkbox.MainActivity">


        <CheckBox
            android:id="@+id/checkBox"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="144dp"
            android:layout_marginTop="68dp"
            android:text="Pizza"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <CheckBox
            android:id="@+id/checkBox2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="144dp"
            android:layout_marginTop="28dp"
            android:text="Coffee"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/checkBox" />
```

```
29.
30.         <CheckBox
31.             android:id="@+id/checkBox3"
32.             android:layout_width="wrap_content"
33.             android:layout_height="wrap_content"
34.             android:layout_marginLeft="144dp"
35.             android:layout_marginTop="28dp"
36.             android:text="Burger"
37.             app:layout_constraintStart_toStartOf="parent"
38.             app:layout_constraintTop_toBottomOf="@+id/checkBox2" />
39.
40.         <Button
41.             android:id="@+id/button"
42.             android:layout_width="wrap_content"
43.             android:layout_height="wrap_content"
44.             android:layout_marginLeft="144dp"
45.             android:layout_marginTop="184dp"
46.             android:text="Order"
47.             app:layout_constraintStart_toStartOf="parent"
48.             app:layout_constraintTop_toBottomOf="@+id/checkBox3" />
49.
50.     </android.support.constraint.ConstraintLayout>
```

Activity class

Let's write the code to check which toggle button is ON/OFF.

File: MainActivity.java

```
1. package example..com.checkbox;
2.
3. import android.support.v7.app.AppCompatActivity;
4. import android.os.Bundle;
5. import android.view.View;
6. import android.widget.Button;
7. import android.widget.CheckBox;
```

```
8.  import android.widget.Toast;
9.
10.         public class MainActivity extends AppCompatActivity {
11.             CheckBox pizza,coffe,burger;
12.             Button buttonOrder;
13.             @Override
14.             protected void onCreate(Bundle savedInstanceState) {
15.                 super.onCreate(savedInstanceState);
16.                 setContentView(R.layout.activity_main);
17.                 addListenerOnButtonClick();
18.             }
19.             public void addListenerOnButtonClick(){
20.                 //Getting instance of CheckBoxes and Button from the activt
    y_main.xml file
21.                 pizza=(CheckBox)findViewById(R.id.checkBox);
22.                 coffe=(CheckBox)findViewById(R.id.checkBox2);
23.                 burger=(CheckBox)findViewById(R.id.checkBox3);
24.                 buttonOrder=(Button)findViewById(R.id.button);
25.
26.                 //Applying the Listener on the Button click
27.                 buttonOrder.setOnClickListener(new View.OnClickListener(){
28.
29.                     @Override
30.                     public void onClick(View view) {
31.                         int totalamount=0;
32.                         StringBuilder result=new StringBuilder();
33.                         result.append("Selected Items:");
34.                         if(pizza.isChecked()){
35.                             result.append("\nPizza 100Rs");
36.                             totalamount+=100;
37.                         }
38.                         if(coffe.isChecked()){
39.                             result.append("\nCoffe 50Rs");
40.                             totalamount+=50;
```

```
41.                     }
42.                     if(burger.isChecked()){
43.                         result.append("\nBurger 120Rs");
44.                         totalamount+=120;
45.                     }
46.                     result.append("\nTotal: "+totalamount+"Rs");
47.                     //Displaying the message on the toast
48.                     Toast.makeText(getApplicationContext(), result.toString()
    , Toast.LENGTH_LONG).show();
49.                 }
50.
51.             });
52.         }
53.     }
```