Android UI Control [Switch]

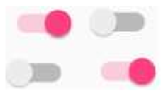
Android – UI Control

Switch (ON / OFF) Button

- **Switch** is a two-state user interface element that is used to display **ON** (**Checked**) or **OFF** (**Unchecked**) states as a button with thumb slider. By using **thumb**, the user may drag back and forth to choose an option either **ON** or **OFF**.

- The **Switch** element is useful for the users to change the settings between two states either **ON** or **OFF**. We can add a **Switch** to our application layout by using **Switch** object.

- Following is the pictorial representation of using **Switch** in android applications.



- By default, the android **Switch** will be in the **OFF** (**Unchecked**) state. We can change the default state of **Switch** by using **android:checked** attribute.In case, if we want to change the state of Switch to **ON** (**Checked**), then we need to set **android:checked = “true”** in our XML layout file.
- In android, we can create **Switch** control in two ways either in the XML layout file or create it in the Activity file programmatically.

Android – UI Control

Create Switch in XML Layout File

- Following is the sample way to define **Switch** control in XML layout file in the android application.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
android:layout_height="match_parent">
<Switch
   android:id="@+id/switch1"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:switchMinWidth="56dp"
   android:layout_marginLeft="100dp"
   android:layout_marginTop="120dp"
   android:text="Switch1:"
   android:checked="true"
   android:textOff="OFF"
   android:textOn="ON"/>
</RelativeLayout>
```

Android – UI Control

Create Switch Control in Activity File

- We can create **Switch** control programmatically in **activity** file based on our requirements.

- Following is the example of creating **Switch** control dynamically in an activity file.

```
RelativeLayout layout = (RelativeLayout)findViewById(R.id.r_layout);
Switch sb = new Switch(this);
sb.setTextOff("OFF");
sb.setTextOn("ON");
sb.setChecked(true);
layout.addView(sb);
```

Android – UI Control

Handle Switch Click Events

- Whenever the user clicks on **Switch**, we can detect whether the Switch is in **ON** or **OFF** state and we can handle the Switch click event in activity file using **setOnCheckedChangeListener** like as shown below.

```
Switch sw = (Switch) findViewById(R.id.switch1);
sw.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
   public void onCheckedChanged(CompoundButton buttonView, boolean
isChecked) {
      if (isChecked) {
         // The toggle is enabled
      } else {
         // The toggle is disabled
      }
   }
});
```

Android – UI Control

Android Switch Control Attributes

Attribute	Description
android:id	It is used to uniquely identify the control
android:checked	It is used to specify the current state of switch control
android:gravity	It is used to specify how to align the text like left, right, center, top, etc.
android:text	It is used to set the text.
android:textOn	It is used to set the text when the toggle button is in the ON / Checked state.
android:textOff	It is used to set the text when toggle button is in OFF / Unchecked state.
android:textColor	It is used to change the color of the text.
android:textSize	It is used to specify the size of the text.

Android – UI Control

Android Switch Control Attributes

Attribute	Description
android:textStyle	It is used to change the style (bold, italic, bolditalic) of text.
android:background	It is used to set the background color for toggle button control.
android:padding	It is used to set the padding from left, right, top and bottom.
android:drawableBottom	It's a drawable to be drawn to the below of text.
android:drawableRight	It's a drawable to be drawn to the right of the text.
android:drawableLeft	It's drawable to be drawn to the left of the text.

Android – UI Control

Android Switch Control Example

- In this example we define two **Switch** controls, one **Button** control in **RelativeLayout** to get the state of **Switch** controls when we click on button control in the android application.

- Create a new android application using android studio and give names as **SwitchExample**.

- Now open an **activity_main.xml** file from **\res\layout** path and write the code like as shown below

Android – UI Control

Android Switch Control Example

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.co
m/apk/res/android"
   android:layout_width="match_parent"
android:layout_height="match_parent">
   <Switch
      android:id="@+id/switch1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:switchMinWidth="56dp"
      android:layout_marginLeft="100dp"
      android:layout_marginTop="120dp"
      android:text="Switch1:"
      android:checked="true"
      android:textOff="OFF"
      android:textOn="ON"/>
   <Switch
      android:id="@+id/switch2"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:switchMinWidth="56dp"
      android:layout_below="@+id/switch1"
      android:layout_alignLeft="@+id/switch1"
      android:text="Switch2:"
      android:textOff="OFF"
      android:textOn ="ON"/>
   <Button
      android:id="@+id/getBtn"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginLeft="150dp"
      android:layout_marginTop="200dp"
      android:text="Get" />
</RelativeLayout>
```

Android – UI Control

Android Switch Control Example

MainActivity.java

```
package com.tutlane.switchexample;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Switch;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
private Switch sw1,sw2;
   private  Button btnGet;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      sw1 = (Switch)findViewById(R.id.switch1);
      sw2 = (Switch)findViewById(R.id.switch2);
      btnGet = (Button)findViewById(R.id.getBtn);
```

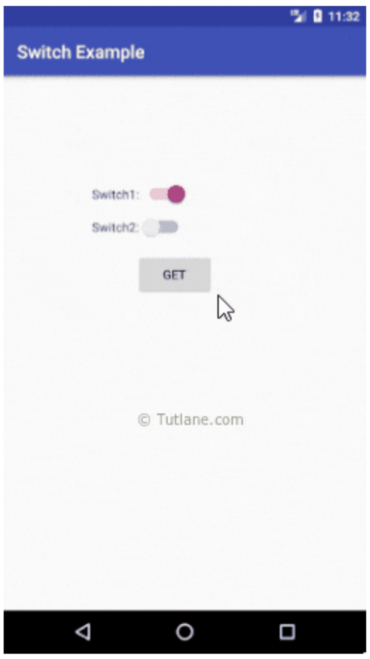
Android – UI Control

Android Switch Control Example

```
btnGet.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String str1, str2;
                if (sw1.isChecked())
                    str1 = sw1.getTextOn().toString();
                else
                    str1 = sw1.getTextOff().toString();
                if (sw2.isChecked())
                    str2 = sw2.getTextOn().toString();
                else
                    str2 = sw2.getTextOff().toString();
                Toast.makeText(getApplicationContext(), "Switch1 -  " + str1 + " \n" + "Switch2
- " + str2,Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

Android – UI Control

Output of Android Switch Example



THANKS