# Android GUI Widgets

# TextView

- In Android, TextView displays text to the user and optionally allows them to edit it programmatically. TextView is a complete text editor, however basic class is configured to not allow editing but we can edit it.
- View is the parent class of TextView. Being a subclass of view the text view component can be used in your app's GUI inside a ViewGroup, or as the content view of an activity.

# TextView code in XML:

- <TextView android:id="@+id/simpleTextView"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="“Hello" />
- **TextView code in JAVA:**

  TextView textView = (TextView)
  findViewById(R.id.textView);
  textView.setText("AbhiAndroid"); //set text for
  text view

# Attributes of TextView:

- **1. id:** id is an attribute used to uniquely identify a text view. Below is the example code in which we set the id of a text view.

```
<TextView android:id="@+id/simpleTextView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"/>
```

- **2. gravity:** The gravity attribute is an optional attribute which is used to control the alignment of the text like left, right, center, top, bottom, center_vertical, center_horizontal etc.

  Below is the example code with explanation included in which we set the center_horizontal gravity for text of a TextView.

```
<TextView android:id="@+id/simpleTextView"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text=“Hello"
android:textSize="20sp"
android:gravity="center_horizontal"/>
<!--center horizontal gravity-->
```

**3. text:** text attribute is used to set the text in a text view. We can set the text in xml as well as in the java class.

```
<TextView android:id="@+id/simpleTextView"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:layout_centerInParent="true"
 android:textSize="25sp"
 android:text="Hello "/>
```

**In Java class:**

```
TextView textView =
 (TextView)findViewById(R.id.textView);
 textView.setText("Hello"); //set text for text view
```

- **4. textColor:** textColor attribute is used to set the text color of a text view.
- <TextView android:id="@+id/simpleTextView" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text=“HEllo" android:layout_centerInParent="true" android:textSize="25sp" android:textColor="#f00"/><!--red color for text view-->
- **In Java class:**

TextView textView = (TextView)findViewById(R.id.textView);

textView.setTextColor(Color.***RED***); //set red color for text vie

5. **textSize:** textSize attribute is used to set the size of text of a text view. We can set the text size in sp(scale independent pixel) or dp(density pixel).

```
<TextView android:id="@+id/simpleTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=“Hello"
    android:layout_centerInParent="true"
    android:textSize="40sp" /><!--Set size-->
```

**In Java class:**

```
TextView textView =
    (TextView)findViewById(R.id.textView);
    textView.setTextSize(20); //set 20sp size of text
```

**6. textStyle:** textStyle attribute is used to set the text style of a text view. The possible text styles are bold, italic and normal. If we need to use two or more styles for a text view then “|” operator is used for that.

```
<TextView android:id="@+id/simpleTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=“HEllo"
    android:layout_centerInParent="true"
    android:textSize="40sp"
    android:textStyle="bold|italic"/><!--bold and
    italic text style of text-->
```

- **7. background:** background attribute is used to set the background of a text view. We can set a color or a drawable in the background of a text view.
- **8. padding:** padding attribute is used to set the padding from left, right, top or bottom. In above example code of background we also set the 10dp padding from all the side's of text view.
- <TextView android:id="@+id/simpleTextView" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="AbhiAndroid" android:layout_centerInParent="true" android:textSize="40sp" android:padding="10dp" android:textColor="#fff" android:background="#000"/> <!--red color for background of text view-->

- **In Java class:**
- TextView textView = (TextView)findViewById(R.id.textView);
  textView.setBackgroundColor(Color.***BLACK***);//set background color

# EditText

- In Android, EditText is a standard entry widget in android apps. It is an overlay over TextView that configures itself to be editable. EditText is a subclass of TextView with text editing operations. **We often use EditText in our applications in order to provide an input or text field, especially in forms.**
- <EditText android:id="@+id/simpleEditText" android:layout_height="wrap_content" android:layout_width="match_parent"/>

- **Retrieving / Getting the Value From EditText In Java Class:**
- EditText simpleEditText = (EditText) findViewById(R.id.simpleEditText);
  String editTextValue = simpleEditText.getText().toString();

# Attributes of EditText:

- **1. id:** id is an attribute used to uniquely identify a text EditText. Below is the example code in which we set the id of a edit text.
- **2. gravity:** The gravity attribute is an optional attribute which is used to control the alignment of the text like left, right, center, top, bottom, center_vertical, center_horizontal etc.
- **3. text:** text attribute is used to set the text in a EditText. We can set the text in xml as well as in the java class.
- <EditText android:id="@+id/simpleEditText" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:text="Username"/><!--set text in edit text-->
- **Setting text in EditText In Java class:**
- EditText editText = (EditText)findViewById(R.id.simpleEditText); editText.setText("Username");//set the text in edit text

- **4. hint:** hint is an attribute used to set the hint i.e. what you want user to enter in this edit text. Whenever user start to type in edit text the hint will automatically disappear.
- <EditText android:id="@+id/simpleEditText" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true“

  android:hint="Enter Your Name Here" /><!--display the hint-->
- **Setting hint in EditText In Java class:**
- EditText editText = (EditText)findViewById(R.id.simpleEditText); editText.setHint("Enter Your Name Here");//display the hint

- **5. textColor:** textColor attribute is used to set the text color of a text edit text.
- <EditText android:id="@+id/simpleEditText" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:text="Password" android:textColor="#f00"/><!--set the red text color-->
- **Setting textColor in EditText In Java class:**
- EditText simpleEditText=(EditText)findViewById(R.id.*simpleEditText*); simpleEditText.setTextColor(Color.*RED*);//set the red text color

- **6. textColorHint:** textColorHint is an attribute used to set the color of displayed hint.
- Below is the example code with explanation included in which we set the green color for displayed hint of a edit text.
- <EditText android:id="@+id/simpleEditText" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:hint="Enter Your Name Here" android:textColorHint="#0f0"/><!--set the hint color green-->
- **Setting textColorHint in EditText In Java class:**
- EditText simpleEditText=(EditText)findViewById(R.id.*simpleEditText*); simpleEditText.setHintTextColor(Color.*green*(0));//set the green hint color

- **7. textSize:** textSize attribute is used to set the size of text of a edit text. We can set the text size in sp(scale independent pixel) or dp(density pixel).
- <EditText android:id="@+id/simpleEditText" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:text="AbhiAndroid" android:textSize="25sp" /><!--set 25sp text size-->
- **Setting textSize in EditText in Java class:**
- EditText simpleEditText=(EditText)findViewById(R.id.*simpleEditText*); simpleEditText.setTextSize(25);//set size of text

- **8. textStyle:** textStyle attribute is used to set the text style of a edit text. The possible text styles are bold, italic and normal. If we need to use two or more styles for a edit text then “|” operator is used for that.
- <EditText android:id="@+id/simpleEditText" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:text="Email" android:textSize="25sp" android:textStyle="bold|italic"/><!--set bold and italic text style-->

- **9. background:** background attribute is used to set the background of a edit text. We can set a color or a drawable in the background of a edit text.
- <EditText android:id="@+id/simpleEditText" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:hint="Enter Your Name Here" android:padding="15dp" android:textColorHint="#fff" android:textStyle="bold|italic" android:background="#000"/><!-- set background color black-->
- **Setting Background in EditText In Java class:**
- Below is the example code in which we set the background color of a edit text programmatically means in java class.
- EditText simpleEditText=(EditText)findViewById(R.id.*simpleEditText*); simpleEditText.setBackgroundColor(Color.*BLACK*);//set black background color
- **10. padding:** padding attribute is used to set the padding from left, right, top or bottom. In above example code of background we also set the 10dp padding from all the side's of edit text.

# Button

- In Android, **Button** represents a push button. A Push buttons can be clicked, or pressed by the user to perform an action. There are different types of buttons used in android such as CompoundButton, ToggleButton, RadioButton.
- Button is a subclass of TextView class and compound button is the subclass of Button class. **On a button we can perform different actions or events like click event, pressed event, touch event etc.**
- Android buttons are GUI components which are sensible to taps (clicks) by the user. *When the user taps/clicks on button in an Android app, the app can respond to the click/tap*. These buttons can be divided into two categories: the first is Buttons with text on, and second is buttons with an image on. A button with images on can contain both an image and a text. Android buttons with images on are also called *ImageButton*.

- **Button code in XML:**
- <Button android:id="@+id/simpleButton"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text=“Submit"/>

# Attributes of Button in Android:

- **1. id:** id is an attribute used to uniquely identify a text Button. Below is the example code in which we set the id of a Button.
- **2. gravity:** The gravity attribute is an optional attribute which is used to control the alignment of the text like left, right, center, top, bottom, center_vertical, center_horizontal etc.

- **3. text**: text attribute is used to set the text in a Button. We can set the text in xml as well as in the java class.
- <Button android:id="@+id/simpleButton" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:text="“Submit"/><!--display text on button-->
- **Setting Text Using Java class:**
- Button button = (Button) findViewById(R.id.simpleButton); button.setText("Learn Android @ AbhiAndroid");//set the text on button

- **4.textColor:** textColor attribute is used to set the text color of a Button.
- <Button android:id="@+id/simpleButton" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:text="“Submit" android:textColor="#f00"/><!-- red color for the text-->
- **Setting Text Color On Button Inside Java class:**
- Button simpleButton=(Button) findViewById(R.id.*simpleButton*); simpleButton.setTextColor(Color.*RED*);//set the red color for the text

- **5. textSize:** textSize attribute is used to set the size of the text on Button. We can set the text size in sp(scale independent pixel) or dp(density pixel).
- <Button android:id="@+id/simpleButton" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:text="AbhiAndroid" android:textSize="25sp" /><!--25sp text size-->
- **Setting textSize In Java class:**
- Button simpleButton=(Button)findViewById(R.id.*simpleButton*); simpleButton.setTextSize(25);//set the text size of button

- **6. textStyle:** textStyle attribute is used to set the text style of a Button. The possible text styles are bold, italic and normal. If we need to use two or more styles for a Button then “|” operator is used for that.
- <Button android:id="@+id/simpleButton" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:text="AbhiAndroid" android:textSize="20sp" android:textStyle="bold|italic"/><!--bold and italic text style-->

- **7. background:** background attribute is used to set the background of a Button. We can set a color or a drawable in the background of a Button.
- <Button android:id="@+id/simpleButton" android:layout_width="fill_parent" android:layout_height="wrap_content" android:layout_centerInParent="true" android:text="Download" android:textSize="20sp" android:padding="15dp" android:textStyle="bold|italic" android:background="#147D03" /><!--Background green color-->
- **Setting background in Button In Java class:**
- Button simpleButton=(Button)findViewById(R.id.*simpleButton*); simpleButton.setBackgroundColor(Color.*BLACK*);//set the black color of button background

- **8. padding:** padding attribute is used to set the padding from left, right, top or bottom. In above example code of background we also set the 10dp padding from all the side's of button.
- **9. drawableBottom:** drawableBottom is the drawable to be drawn to the below of the text.

```
<Button
    android:id="@+id/simpleButton"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true"
    android:background="#147D03“
    android:text="Download Code"
    android:textSize="20sp"
    android:padding="15dp“
    android:textStyle="bold|italic"
    android:drawableBottom="@drawable/ic_launcher"/>
        <!--image drawable on button-->
```

- **10. drawableTop, drawableRight And drawableLeft:** Just like the above attribute we can draw drawable to the left, right or top of text.

```
<Button android:id="@+id/simpleButton"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true"
    android:background="#147D03"
    android:text="Download Code"
    android:textSize="20sp"
android:padding="15dp" android:textStyle="bold|italic"
    android:drawableRight="@drawable/ic_launcher"/><!-
    -image drawable on Right side of Text on button-->
```

# ImageButton

- In Android, Image Button is used to display a normal button with a custom image in a button. In simple words we can say, Image Button is a button with an image that can be pressed or clicked by the users. By default it looks like a normal button with the standard button background that changes the color during different button states.
- <ImageButton android:id="@+id/simpleImageButton" android:layout_width="wrap_content" android:layout_height="wrap_content" android:src="@drawable/home" />

# Attributes of ImageButton:

1. **id:** id is an attribute used to uniquely identify a image button. Below is the example code in which we set the id of a image button.
2. **src:** src is an attribute used to set a source file of image or you can say image in your image button to make your layout look attractive.

- <ImageButton android:id="@+id/simpleImageButton" android:layout_width="wrap_content" android:layout_height="wrap_content" **android:src="@drawable/home"**/>

<!--src(source)file from drawable folder which display an imagebutton-->

## ➢ ImageButton Using Java class:

- We can also set the source image at run time programmatically in java class. For that we use setImageResource() function as shown in below example code.
- ImageButton simpleImageButton = (ImageButton)findViewById(R.id.*simpleImageButton*); simpleImageButton.setImageResource(R.drawable.*home*); //set the image programmatically

- **3. background:** background attribute is used to set the background of an image button. We can set a color or a drawable in the background of a Button.
- <ImageButton android:id="@+id/simpleImageButton" android:layout_width="wrap_content" android:layout_height="wrap_content" android:src="@drawable/home“
- android:background="#000"/>
- <!-- black background color for image button-->
- **Using Java class:**
- ImageButton simpleImageButton = (ImageButton) findViewById(R.id.*simpleImageButton*); simpleImageButton.setBackgroundColor(Color.*BLACK*); //set black background color for image button

- **4. padding:** padding attribute is used to set the padding from left, right, top or bottom of the ImageButton.
- **paddingRight :** set the padding from the right side of the image button**.**
- **paddingLeft :** set the padding from the left side of the image button**.**
- **paddingTop :** set the padding from the top side of the image button**.**
- **paddingBottom :** set the padding from the bottom side of the image button**.**
- **padding :** set the padding from the all side's of the image button**.**

```
<ImageButton android:id="@+id/simpleImageButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#000"
    android:src="@drawable/home"
    android:padding="30dp"/><!-- set 30dp padding from all the sides
    of the view-->
```

# CheckBox

- In Android, Checkbox is a type of two state button either unchecked or checked in Android. Or you can say it is a type of on/off switch that can be toggled by the users. You should use checkbox when presenting a group of selectable options to users that are not mutually exclusive.
- <CheckBox android:id="@+id/simpleCheckBox" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Simple CheckBox"/>

- **Important Note:** You can check the current state of a check box programmatically by using isChecked() method. This method returns a Boolean value either true or false, if a check box is checked then it returns true otherwise it returns false. Below is an example code in which we checked the current state of a check box.
- //initiate a check box

```
CheckBox simpleCheckBox = (CheckBox) findViewById(R.id.simpleCheckBox);
//check current state of a check box (true or false)
Boolean checkBoxState = simpleCheckBox.isChecked();
```

# Attributes of CheckBox:

- **2. checked:** checked is an attribute of check box used to set the current state of a check box. The value should be true or false where true shows the checked state and false shows unchecked state of a check box. The default value of checked attribute is false. We can also set the current state programmatically.
- <CheckBox android:id="@+id/simpleCheckBox" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Simple CheckBox" android:checked="true"/> <!--set the current state of the check box-->

- **3. gravity:** The gravity attribute is an optional attribute which is used to control the alignment of the text in CheckBox like left, right, center, top, bottom, center_vertical, center_horizontal etc.
- <CheckBox android:id="@+id/simpleCheckBox" android:layout_width="fill_parent" android:layout_height="wrap_content" android:text="Simple CheckBox" android:checked="true" android:gravity="right|center_vertical"/> <!-- gravity of the check box-->

- **4. text:** text attribute is used to set the text in a check box. We can set the text in xml as well as in the java class.

<CheckBox android:id="@+id/simpleCheckBox" android:layout_width="wrap_content" android:layout_height="wrap_content" android:checked="true" android:text="Text Attribute Of Check Box"/> <!--displayed text of the check box-->

- **In Java class:**

/*Add in Oncreate() funtion after setContentView()*/ // initiate check box

CheckBox simpleCheckBox = (CheckBox) findViewById(R.id.***simpleCheckBox***);

// displayed text of the check box

simpleCheckBox.setText("**Text Attribute Of Check Box**");

- **5. textColor:** textColor attribute is used to set the text color of a check box.
- <CheckBox android:id="@+id/simpleCheckBox" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Text is Red Color" android:textColor="#f00" android:checked="true"/> <!-- red color for the text of check box

- **6. textSize:** textSize attribute is used to set the size of text of a check box. We can set the text size in sp(scale independent pixel) or dp(density pixel).
- <CheckBox android:id="@+id/simpleCheckBox" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Text size Attribute Of Check Box" android:textColor="#00f" android:checked="false" android:textSize="20sp"/><!--set Text Size of text in CheckBox-->
- **7. textStyle:** textStyle attribute is used to set the text style of the text of a check box. The possible text styles are bold, italic and normal. If we need to use two or more styles for a text view then "|" operator is used for that.
- <CheckBox android:id="@+id/simpleCheckBox" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Text Style Attribute Of Check Box" android:textColor="#44f" android:textSize="20sp" android:checked="false" android:textStyle="bold|italic"/>

- **8. background:** background attribute is used to set the background of a check box. We can set a color or a drawable in the background of a check box.
- <CheckBox android:id="@+id/simpleCheckBox" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Text size Attribute Of Check Box" android:textColor="#fff" android:textSize="20sp" android:textStyle="bold|italic" android:checked="true"
- android:background="#000" />

- **9. padding:** padding attribute is used to set the padding from left, right, top or bottom.
- **paddingRight :**set the padding from the right side of the check box**.**
- **paddingLeft :**set the padding from the left side of the check box**.**
- **paddingTop :**set the padding from the top side of the check box**.**
- **paddingBottom :**set the padding from the bottom side of the check box**.**
- **Padding :**set the padding from the all side’s of the check box**.**

- <CheckBox android:id="@+id/simpleCheckBox" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Padding Attribute Of Check Box" android:textColor="#44f" android:textSize="20sp" android:textStyle="bold|italic" android:checked="false" android:padding="30dp"/> <!--30dp padding from all side’s-->

# RadioButon

- RadioButon is a two state button that can be checked or unchecked. If a radio button is unchecked then a user can check it by simply clicking on it. Once a RadiaButton is checked by user it can’t be unchecked by simply pressing on the same button. It will automatically unchecked when you press any other RadioButton within same Radio Group.

- <RadioGroup
  android:layout_width="wrap_content"
  android:layout_height="wrap_content">
  <RadioButton
  android:id="@+id/simpleRadioButton"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"/>
  <RadioButton
  android:id="@+id/simpleRadioButton1"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"/>
  </RadioGroup>

# Attributes of RadioButton In Android:

- **1. id:** id is an attribute used to uniquely identify a radio button.
- **2. checked:** checked attribute in radio button is used to set the current state of a radio button. We can set it either true or false where true shows the checked state and false shows unchecked state of a radio button. As usual default value of checked attribute is false. We can also set the current state in JAVA.
- <RadioButton android:id="@+id/simpleRadioButton" android:layout_width="wrap_content" android:layout_height="wrap_content" android:checked="true"/> <!-- set the current state of the radio button-->

- **3. text:** text attribute is used to set the text in a radio button. We can set the text both ways either in XML or in JAVA class.
- <RadioButton android:id="@+id/simpleRadioButton" android:layout_width="wrap_content" android:layout_height="wrap_content" android:checked="true" android:layout_centerHorizontal="true" android:text="I am a radiobutton" /> <!-- displayed text of radio button-->

- **4. gravity:** The gravity attribute is an optional attribute which is used to control the alignment of text like left, right, center, top, bottom, center_vertical, center_horizontal etc.
- **5. textColor:** textColor attribute is used to set the text color of a radio button.
- **6. textSize:** textSize attribute is used to set the size of the text of a radio button. We can set the text size in sp(scale independent pixel) or dp(density pixel).
- **7. textStyle:** textStyle attribute is used to set the text style of the text of a radio button. The possible text styles are bold, italic and normal. If we need to use two or more styles for a text view then "|" operator is used for that.
- **8. background:** background attribute is used to set the background of a radio button. We can set a color or a drawable in the background of a radio button.
- 9. **drawableBottom, drawableTop, drawableLeft And drawableRight:** hese attribute draw the drawable to the below of the text of RadioButton.

# ToggleButton (On/Off)

- In Android, Toggle Button is used to display checked and unchecked state of a button. Toggle Button basically an off/on button with a light indicator which indicate the current state of toggle button.
- <ToggleButton
  android:id="@+id/simpleToggleButton"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"/>

# Attributes of ToggleButton:

- **1. id:** id is an attribute used to uniquely identify a toggle button.
- **2. checked:** checked is an attribute of toggle button used to set the current state of a toggle button. The value should be true or false where true shows the checked state and false shows unchecked state of a toggle button. The default value of checked attribute is false. We can also set the current state programmatically.
- <ToggleButton android:id="@+id/simpleToggleButton" android:layout_width="wrap_content" android:layout_height="wrap_content" android:checked="true" /><!-- set the current state of the toggle button-->

- **3. gravity:** The gravity attribute is an optional attribute which is used to control the alignment of the text in ToogleButton like left, right, center, top, bottom, center_vertical, center_horizontal etc.
- **4. textOn And textOff:** textOn attribute is used to set the text when toggle button is in checked/on state. We can set the textOn in XML as well as in the java class.
- <ToggleButton android:id="@+id/simpleToggleButton" android:layout_width="wrap_content" android:layout_height="wrap_content" android:checked="true" android:layout_centerHorizontal="true" android:textOff="Disable" android:textOn="Enable"/> <!--text to be displayed whenever toggle button is checked-->

- **5. textColor:** textColor attribute is used to set the text color of a toggle button.
- **6. textSize:** textSize attribute set the size of the text of a toggle button. We can set the text size in sp(scale independent pixel) or dp(density pixel).
- **7. textStyle:** textStyle attribute is used to set the text style of the text of a Toggle button. You can set bold, italic and normal. If you need to use two or more styles for a text view then "|" operator is used for that.
- **8. background:** background attribute is used to set the background of a toggle button. We can set a color or a drawable in the background of a toggle button.
- **9. padding:** padding attribute is used to set the padding from left, right, top or bottom.
- **paddingRight :**set the padding from the right side of the toggle button**.**
- **paddingLeft :**set the padding from the left side of the toggle button**.**
- **paddingTop :**set the padding from the top side of the toggle button**.**
- **paddingBottom :**set the padding from the bottom side of the toggle button**.**
- **Padding :**set the padding from the all side's of the toggle button**.**

- **10. drawableBottom, drawableTop, drawableRight And drawableLeft:** These attribute draw the drawable below, top, right and left of the text of ToggleButton.