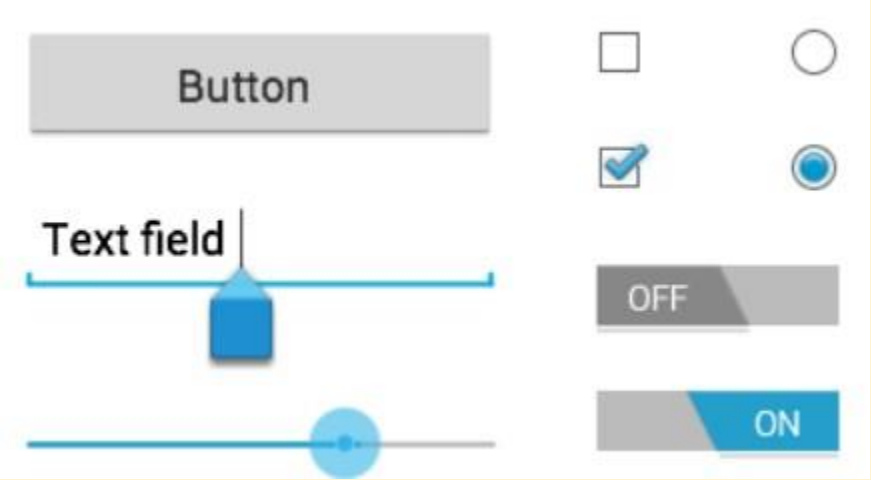
P P SAVANI UNIVERSITY

Module 3
Exploring User Interface Screen Elements



Topics to be covered

P P SAVANI UNIVERSITY

- Introducing Android Views and Layouts
- Displaying Text with TextView
- Retrieving Data From Users
- Using Buttons, Check Boxes and Radio Groups
- Getting Dates and Times from Users,
- Using Indicators to Display and Data to Users
- Adjusting Progress with SeekBar
- Providing Users with Options and Context Menus
- Handling User Events
- Working with Dialogs
- Working with Styles
- Working with Themes.

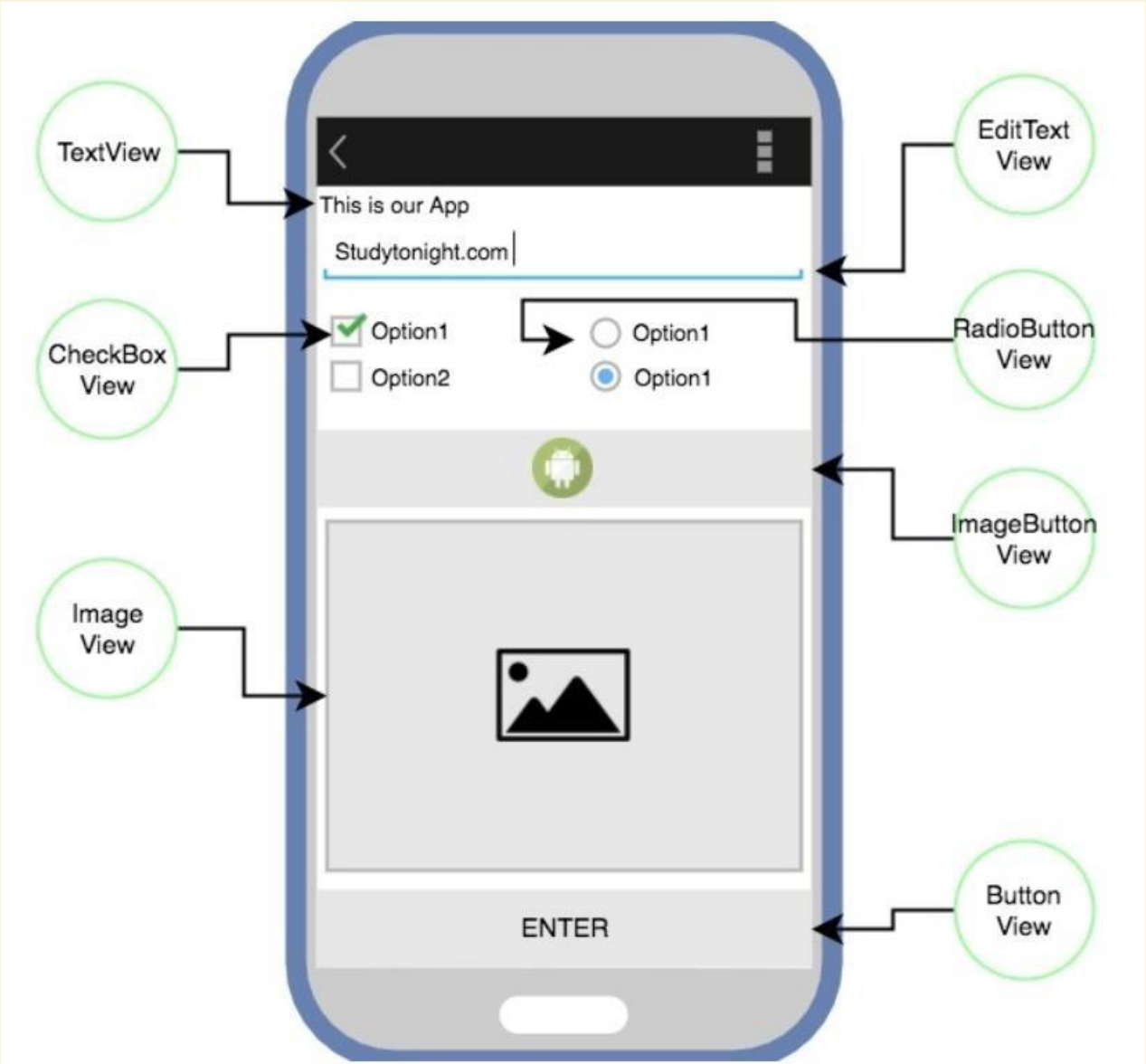
The Android View

P P SAVANI UNIVERSITY

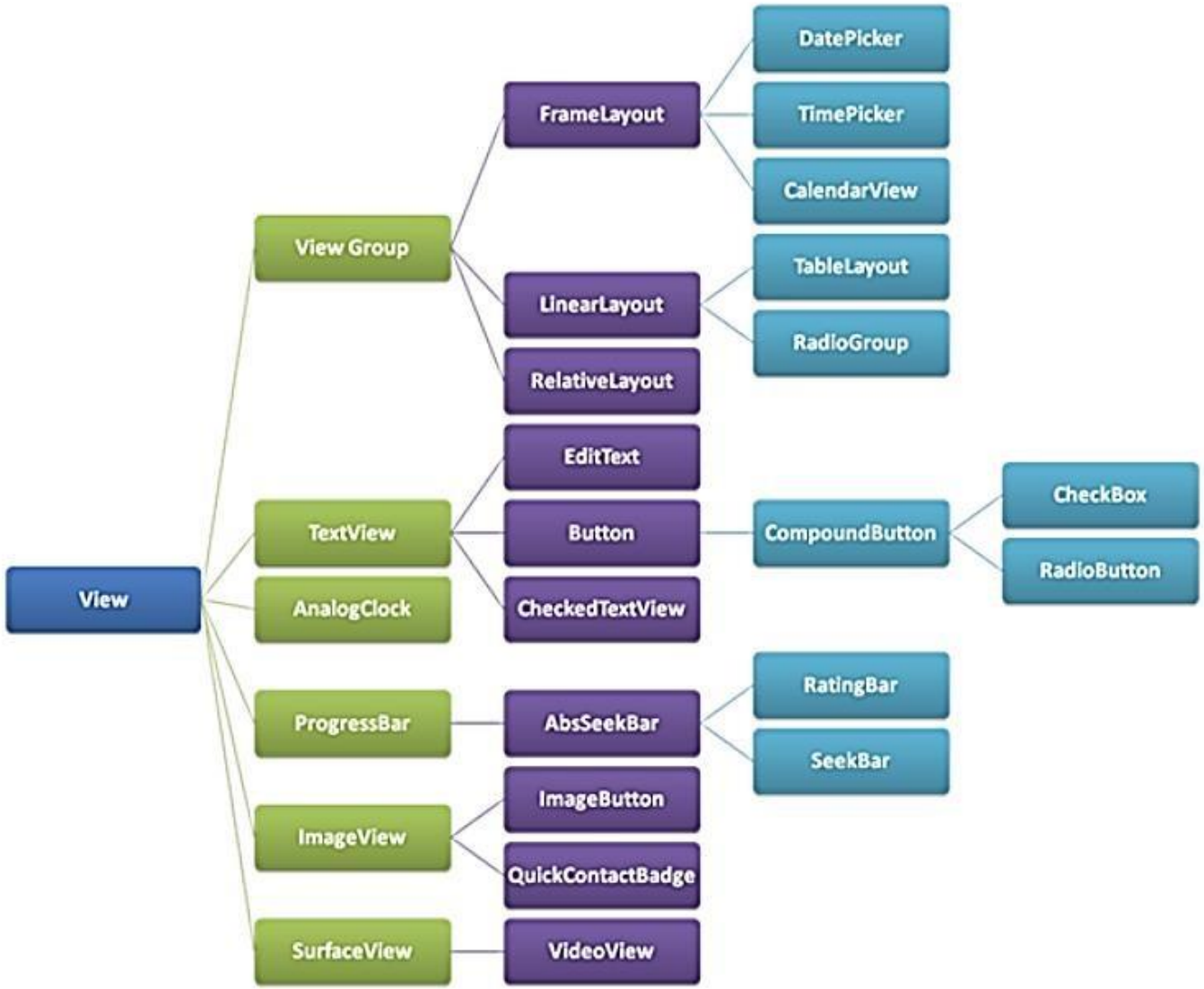
- The Android SDK has a Java package named android.view.
- This package contains a number of interfaces and classes related to drawing on the screen.
- View object refers to one of the classes within this package:

  Android.view.View
- The View class is the basic user interface building block within Android.
- It represents a rectangular portion of the screen. The View class servers as the base class for nearly all the user interface controls and layouts within the Android SDK.

Commonly used Views



View Class Hierarchy



The Android Controls

P P SAVANI UNIVERSITY

- The Android SDK contains a Java package named android.widget.
- When we refer to controls, we are typically referring to a class within this package.
- The Android SDK includes classes to draw most common objects, including ImageView, FrameLayout, EditText and Button classes.
- All these controls are derived from the View class.

The Android Controls

P P SAVANI UNIVERSITY

- User Interface Controls can be static or programmatically.
- Each control you want to be able to access programmatically must have a unique identifier specified using the android:id attribute.
- You use this identifier to access the control with the findViewById() method in your Activity class.

```
TextView tv = (TextView) findViewById(R.id.textview01);
```

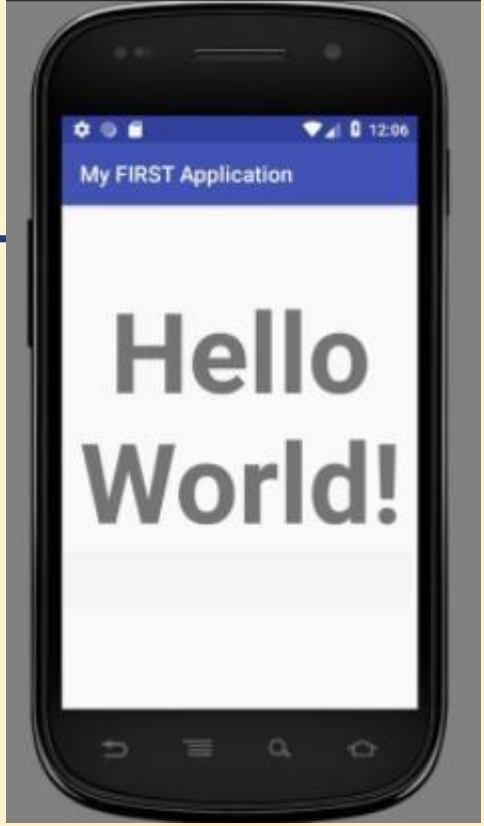
The Android Layout

P P SAVANI UNIVERSITY

- Found in android.widget package.
- It is a View object but it doesn't actually draw anything specific on the screen. Instead it is a parent container for organizing other controls(children).
- Each type of layout control draws its children using particular rules.
- For instance, the LinearLayout controls draw its child controls in a single horizontal row or a single vertical column.
- Details will be in Module 4.

Displaying Text with TextView

```
public class TextView
extends View implements ViewTreeObserver.OnPreDrawListener

java.lang.Object
  ↳ android.view.View
      ↳ android.widget.TextView
```



- **A user interface element that displays text to the user.**

```
<TextView
    android:id="@+id/text_view_id"
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:text="@string/hello" />
```

```
public class MainActivity extends Activity {

   protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final TextView helloTextView = (TextView) findViewById(R.id.text_view_id);
        helloTextView.setText(R.string.user_greeting);
    }
}
```

Creating Contextual Links in Text

P P SAVANI UNIVERSITY

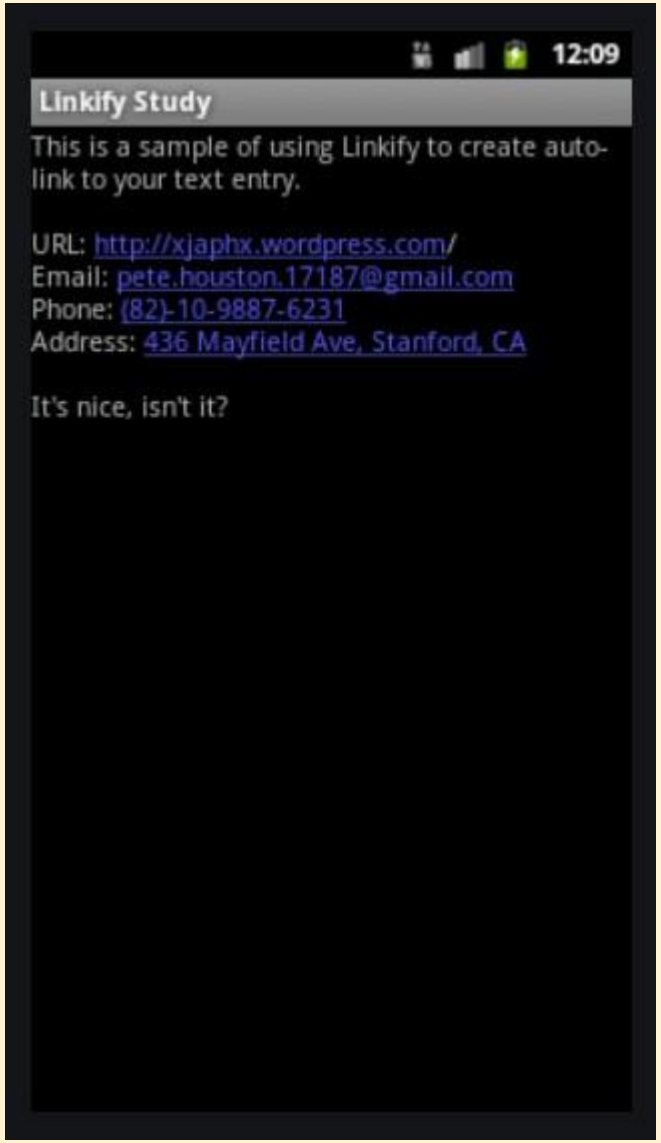
- **andorid:autoLink** => **By using this attribute, android can controls whether links(such as urls, emails, phone and address) are automatically found and converted to clickable links.**

1. **android:autoLink=”none”**
2. **android:autoLink=”email”**
3. **android:autoLink=”phone”**
4. **android:autoLink=”web”**
5. **android:autoLink=”map”**
6. **android:autoLink=”all”**

android:autoLink=”email”

P P SAVANI UNIVERSITY

```
<TextView
        android:id="@+id/txtViewEmail"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Email id: abc@gmail.com"
        android:autoLink="email"
        android:textSize="16sp"
        android:layout_margin="5dip">
</TextView>
```



Retrieving Data from Users with Text Fields EditText

P P SAVANI UNIVERSITY

- In Android, EditText is a standard entry widget in android apps.
- 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. The most simple example of EditText is Login or Sign-in form.

Using EditText Controls

P P SAVANI UNIVERSITY

```
public class EditText
extends TextView

java.lang.Object
  ↳ android.view.View
      ↳ android.widget.TextView
          ↳ android.widget.EditText
```

To retrieve data entered through Android EditText we do the following:

```
EditText simpleEditText = (EditText) findViewById(R.id.simpleEditText);
String editTextValue = simpleEditText.getText().toString();
```

```
<EditText
        android:id="@+id/Nametext"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="Enter your Name"
        android:inputType="text" />
```

activity_main.xml

Checking User Inputs with Input Filters

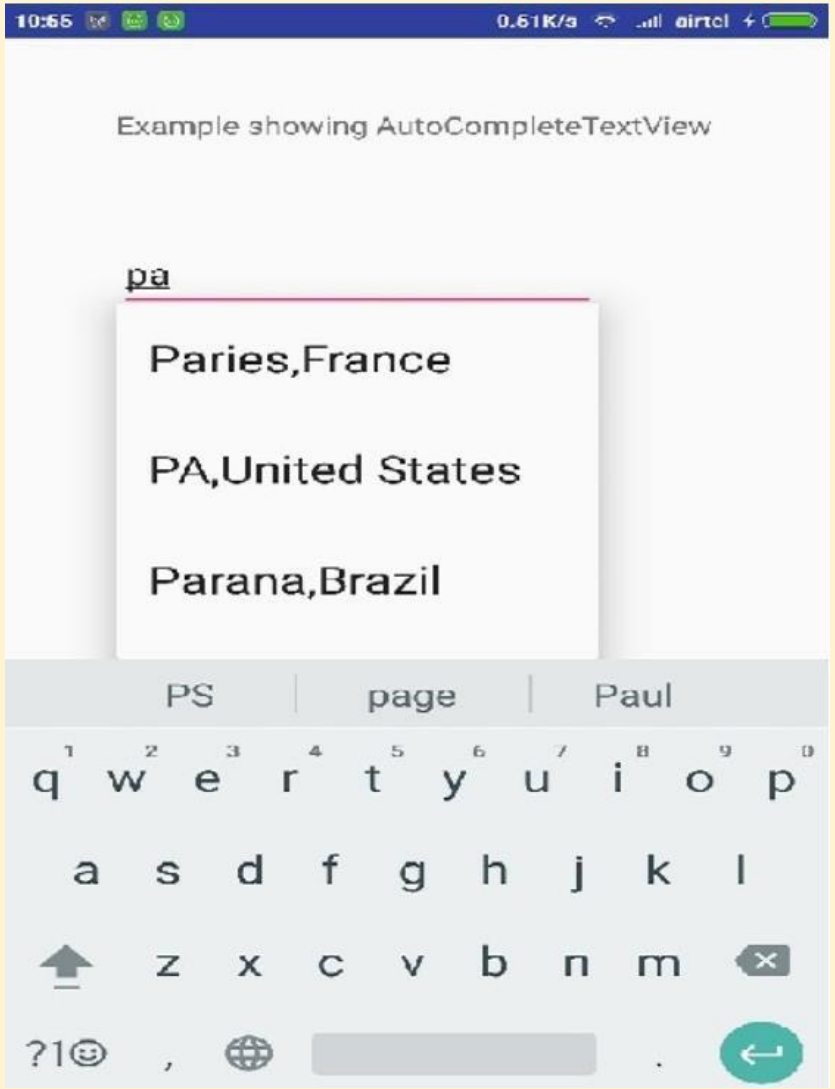
P P SAVANI UNIVERSITY

- Sometimes requirement comes in a way when one needs to restrict characters to be entered in Edit Text. Scenario can be “username should be greater than 6 characters”, “accepts only decimal number with 2 digits after fraction”, etc.

```
 EditText et = (EditText) findViewById(R.id.myEditText);
et.setFilters(new InputFilter[]{ new InputFilterMinMax("1", "12")});
```

```
 EditText editText = new EditText(this);
int maxLength = 3;
editText.setFilters(new InputFilter[] {new InputFilter.LengthFilter(maxLength)});
```

Helping the User with Autocompletion



AutoCompleteTextView

```
<AutoCompleteTextView
…
/>
```

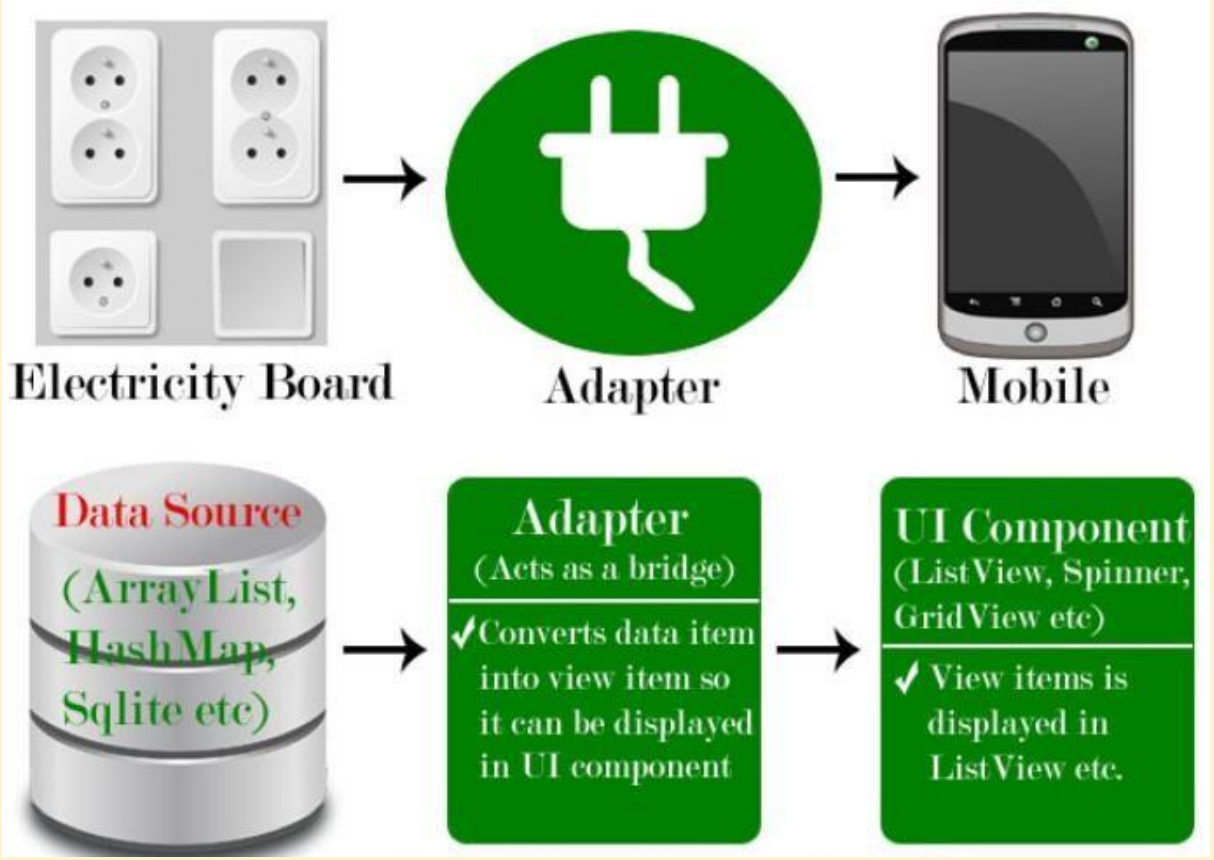
“MainActivity.java” for AutoCompletionTextView


```
public class MainActivity extends Activity {
   AutoCompleteTextView autocomplete;

   String[] arr = { "Paries,France", "PA,United States","Parana,Brazil",
      "Padua,Italy", "Pasadena,CA,United States"};

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

      autocomplete = (AutoCompleteTextView)
      findViewById(R.id.autoCompleteTextView1);

      ArrayAdapter<String> adapter = new ArrayAdapter<String>
      (this,android.R.layout.select_dialog_item, arr);

      autocomplete.setThreshold(2);
      autocomplete.setAdapter(adapter);
   }
}
```

Adapter in Android

P P SAVANI UNIVERSITY

- In Android, Adapter is a bridge between UI component and data source that helps us to fill data in UI component. It holds the data and send the data to an Adapter view then view can takes the data from the adapter view and shows the data on different views like as ListView, GridView, Spinner etc



ArrayAdapter In Android

P P SAVANI UNIVERSITY

- **Whenever we have a list of single items which is backed by an Array, we can use ArrayAdapter. For instance, list of phone contacts, countries or names.**

```
ArrayAdapter(Context context, int resource, int textViewResourceId, T[] objects)
```

```
String animalList[] = {"Lion","Tiger","Monkey","Elephant","Dog","Cat","Camel"};
ArrayAdapter arrayAdapter = new ArrayAdapter(this, R.layout.list_view_items, R.id.textView, animalList);
```

MultiAutoCompleteTextView

P P SAVANI UNIVERSITY

- One most important difference between MultiAutoCompleteTextView and AutoCompleteTextView is that AutoCompleteTextView can hold or select only single value at single time but MultiAutoCompleteTextView can hold multiple string words value at single time. These all values are separated by comma(,).

MultiAutoCompleteTextView Example

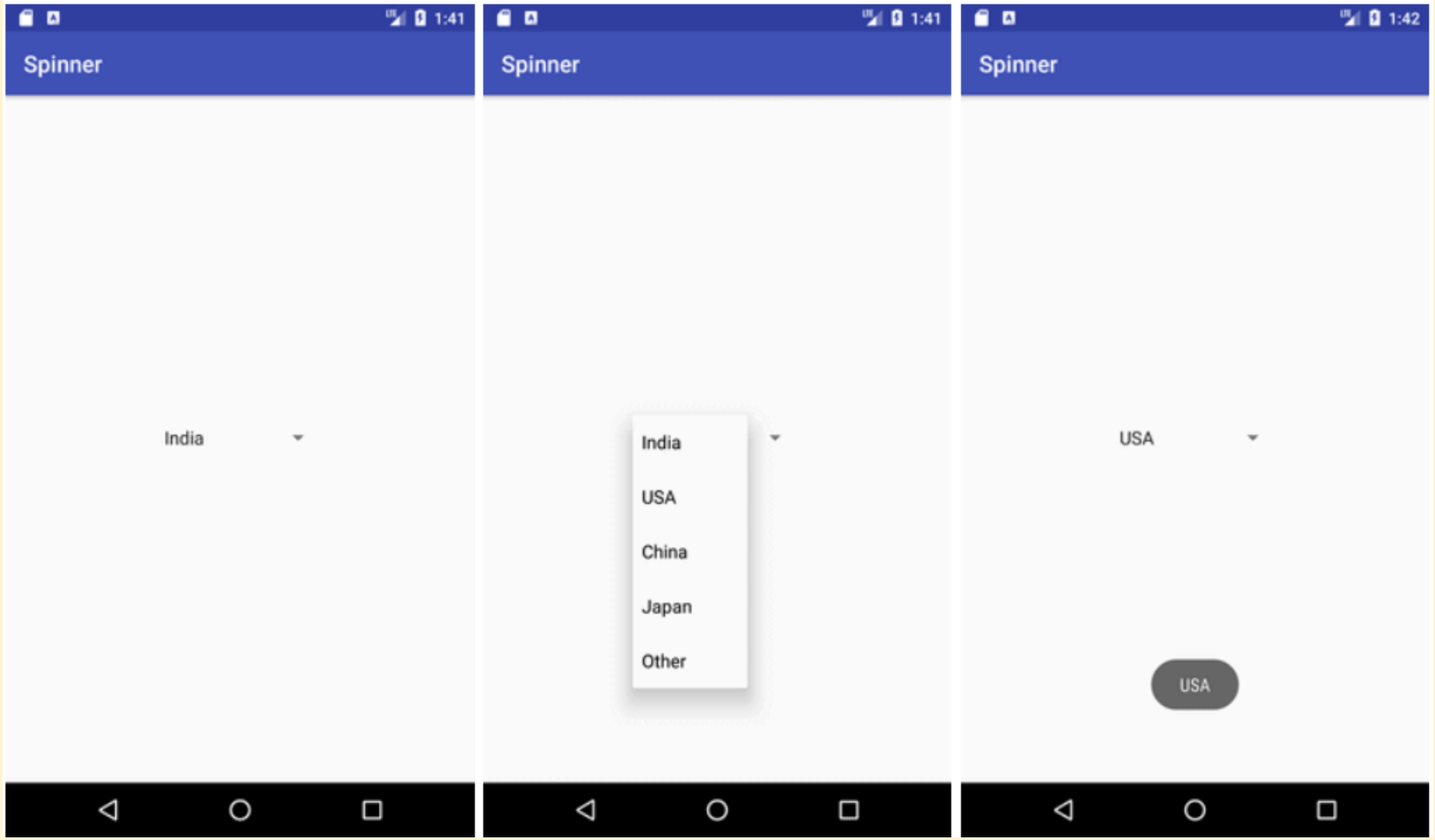
P P SAVANI UNIVERSITY

```
String[] androidVersionNames = {"Aestro", "Blender", "CupCake", "Donut", "Eclair",
"Froyo", "Gingerbread", "HoneyComb", "IceCream Sandwich", "Jellibean", "Kitkat", "
Lollipop", "MarshMallow"};


@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);

// initiate a MultiAutoCompleteTextView

MultiAutoCompleteTextView simpleMultiAutoCompleteTextView = (MultiAutoCompleteText
View) findViewById(R.id.simpleMultiAutoCompleteTextView);

// set adapter to fill the data in suggestion list

ArrayAdapter<String> versionNames = new ArrayAdapter<String>(this, android.R.layou
t.simple_list_item_1, androidVersionNames);

simpleMultiAutoCompleteTextView.setAdapter(versionNames);


// set threshold value 1 that help us to start the searching from first character

simpleMultiAutoCompleteTextView.setThreshold(1);

// set tokenizer that distinguish the various substrings by comma

simpleMultiAutoCompleteTextView.setTokenizer(new MultiAutoCompleteTextView.CommaTo
kenizer());

}
```

Giving Users Choices Using Spinner Controls

- It can be used to display the multiple options to the user in which only one item can be selected by the user.
- Android spinner is like the drop down menu with multiple values from which the end user can select only one value.
- Android spinner is associated with AdapterView. So you need to use one of the adapter classes with spinner..

Giving Users Choices Using Spinner Controls



```
String[] country = { "India", "USA", "China", "Japan", "Other"};

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
   //Getting the instance of Spinner and applying OnItemSelectedListener on it
    Spinner spin = (Spinner) findViewById(R.id.spinner);
    spin.setOnItemSelectedListener(this);

    //Creating the ArrayAdapter instance having the country list
    ArrayAdapter aa = new ArrayAdapter(this,android.R.layout.simple_spinner_item,country);
    aa.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
    //Setting the ArrayAdapter data on the Spinner
    spin.setAdapter(aa);

}

//Performing action onItemSelected and onNothing selected
@Override
public void onItemSelected(AdapterView<?> arg0, View arg1, int position, long id) {
    Toast.makeText(getApplicationContext(),country[position] , Toast.LENGTH_LONG).show();
}
```

P P SAVANI UNIVERSITY

File: MainActivity.java

The code to display item on the spinner and perform event handling

User Selections with Buttons and Switches

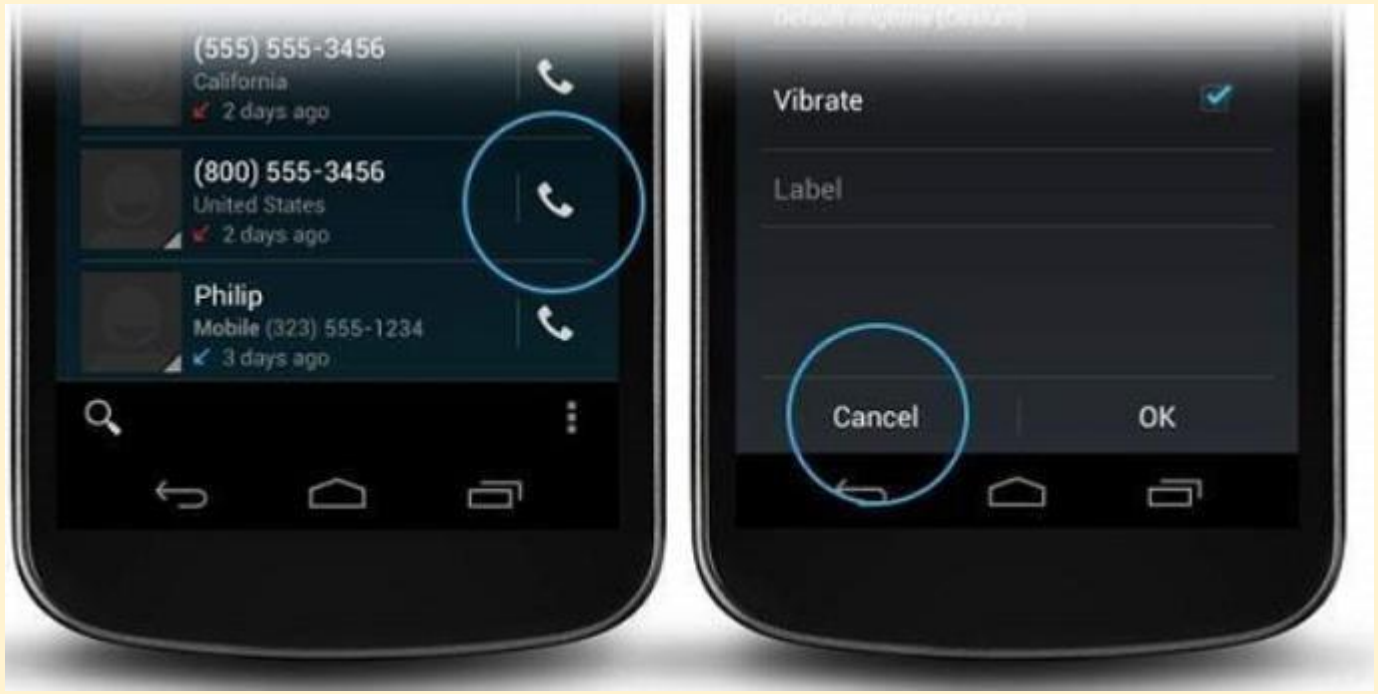
P P SAVANI UNIVERSITY

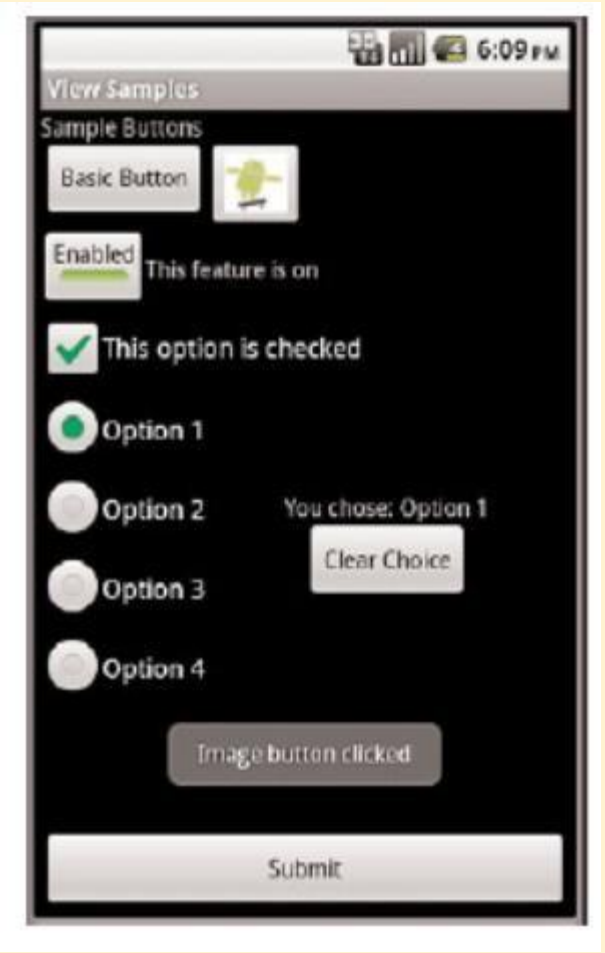
- Using Basic Buttons
- Using Checkbox and ToggleButton Controls
- Using RadioGroup and RadioButton

Using Basic Buttons

P P SAVANI UNIVERSITY

- A basic Button is often used to perform some sort of action, such as submitting a form or confirming a selection.
- A basic Button control can contain a text or image label.
- The android.widget.Button class provides a basic Button implementation in the Android SDK.
- Common Button Text such as “Yes”,”No”,”OK”,”Cancel” or “Submit”.


(555) 555-3456
California
2 days ago
(800) 555-3456
United States
2 days ago
Philip
Mobile (323) 555-1234
3 days ago
Vibrate
Label
Cancel
OK


6:09 PM
View Samples
Sample Buttons
Basic Button
Enabled
This feature is on
This option is checked
Option 1
Option 2
You chose: Option 1
Clear Choice
Option 3
Option 4
Image button clicked
Submit

A Button Listener

P P SAVANI UNIVERSITY

- **A button won't do anything, other than animate, without some code to handle the click event.**

```
setContentView(R.layout.buttons);

final Button basic_button = (Button) findViewById(R.id.basic_button);

basic_button.setOnClickListener(new View.OnClickListener() {

public void onClick(View v) {

Toast.makeText(ButtonsActivity.this,

"Button clicked", Toast.LENGTH_SHORT).show();

}

});
```

ImageButton

P P SAVANI UNIVERSITY

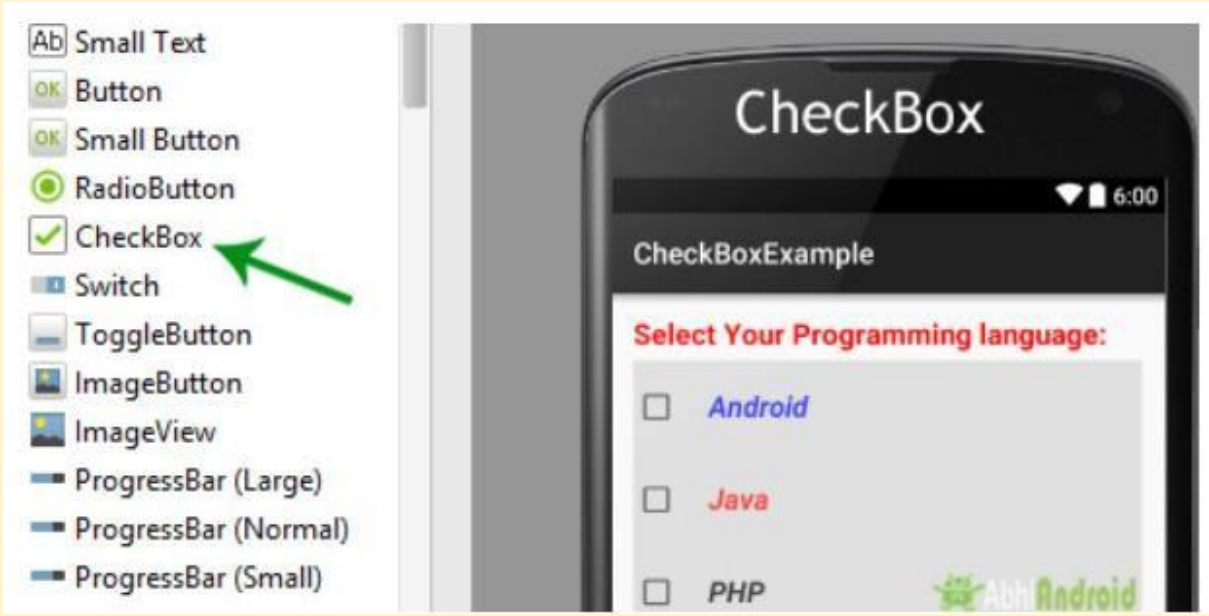
- A button with its primary label as an image is an ImageButton.
- An ImageButton is, for most purposes, almost exactly like a basic button.
- Click actions are handled in the same way. The primary difference is that you can set its src attribute to be an image

```
<ImageButton

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:id="@+id/image_button"

android:src="@drawable/droid" />
```

Checkbox Control

P P SAVANI UNIVERSITY

- The check box button is often used in lists of items where the user can select multiple items.
- The Android check box contains a text attribute that appears to the side of the check box.
- This is used in a similar way to the label of a basic button. In fact, it's basically a TextView next to the button.



Checkbox Control

P P SAVANI UNIVERSITY

- 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. CompoundButton is the parent class of CheckBox class.
- In android there is a lot of usage of check box. For example, to take survey in Android app we can list few options and allow user to choose using CheckBox.
- The user will simply checked these checkboxes rather than type their own option in EditText. Another very common use of CheckBox is as remember me option in Login form.

Listener on Checkbox

P P SAVANI UNIVERSITY

```
final CheckBox check_button = (CheckBox) findViewById(R.id.checkbox);

check_button.setOnClickListener(new View.OnClickListener() {

public void onClick (View v) {

TextView tv = (TextView)findViewById(R.id.checkbox);

tv.setText(check_button.isChecked() ?

“This option is checked” :

“This option is not checked”);

}

});
```

Example


```
    android = (CheckBox) findViewById(R.id.androidCheckBox);
    android.setOnClickListener(this);
    java = (CheckBox) findViewById(R.id.javaCheckBox);
    java.setOnClickListener(this);
    python = (CheckBox) findViewById(R.id.pythonCheckBox);
    python.setOnClickListener(this);
    php = (CheckBox) findViewById(R.id.phpCheckBox);
    php.setOnClickListener(this);
    unity3D = (CheckBox) findViewById(R.id.unityCheckBox);
    unity3D.setOnClickListener(this);
}

@Override
public void onClick(View view) {

    switch (view.getId()) {
        case R.id.androidCheckBox:
            if (android.isChecked())
                Toast.makeText(getApplicationContext(), "Android", Toast.LENGTH_LONG).show();
            break;
        case R.id.javaCheckBox:
            if (java.isChecked())
                Toast.makeText(getApplicationContext(), "Java", Toast.LENGTH_LONG).show();
            break;
        case R.id.phpCheckBox:
            if (php.isChecked())
                Toast.makeText(getApplicationContext(), "PHP", Toast.LENGTH_LONG).show();
            break;
        case R.id.pythonCheckBox:
            if (python.isChecked())
                Toast.makeText(getApplicationContext(), "Python", Toast.LENGTH_LONG).show();
            break;
        case R.id.unityCheckBox:
            if (unity3D.isChecked())
                Toast.makeText(getApplicationContext(), "Unity 3D", Toast.LENGTH_LONG).show()
            break;
    }
}
```

ToggleButton

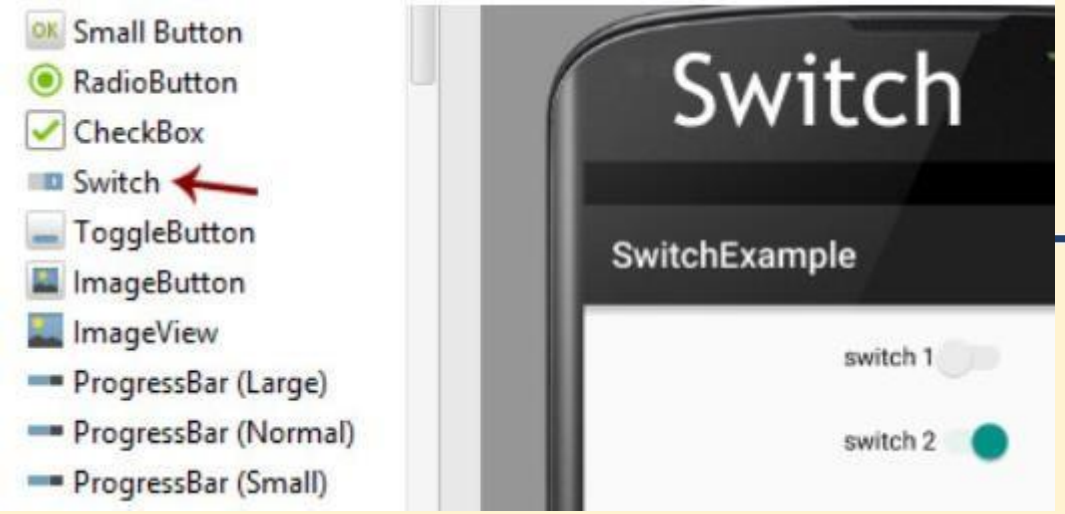
P P SAVANI UNIVERSITY

- A Toggle Button is similar to a check box in behavior but is usually used to show or alter the on or off state of something. Like the CheckBox, it has a state (checked or not).
- Also like the check box, the act of changing what displays on the button is handled for us.
- Unlike the CheckBox, it does not show text next to it.
- Instead, it has two text fields. The first attribute is textOn, which is the text that displays on the button when its checked state is on. The second attribute is textOff, which is the text that displays on the button when its checked state is off. The default text for these is “ON” and “OFF,” respectively.
- The most simple example of ToggleButton is doing on/off in sound, Bluetooth, wifi, hotspot etc

ToggleButton (On/Off)

P P SAVANI UNIVERSITY

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    // initiate toggle button's
    simpleToggleButton1 = (ToggleButton) findViewById(R.id.simpleToggleButton1);
    simpleToggleButton2 = (ToggleButton) findViewById(R.id.simpleToggleButton2);
    submit = (Button) findViewById(R.id.submitButton);
    submit.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            String status = "ToggleButton1 : " + simpleToggleButton1.getText() + "\n" + "ToggleButton2 : " + simpleToggleButton2.getText();
            Toast.makeText(getApplicationContext(), status, Toast.LENGTH_SHORT).show(); // display the current state of toggle button's
        }
    });
}
```



Switch (On/Off)

P P SAVANI UNIVERSITY

- In Android, Switch is a two-state toggle switch widget that can select between two options.
- It is used to display checked and unchecked state of a button providing slider control to user.
- Switch is a subclass of CompoundButton. It is basically an off/on button which indicate the current state of Switch.
- It is commonly used in selecting on/off in Sound, Bluetooth, WiFi etc.

Switch (On/Off)

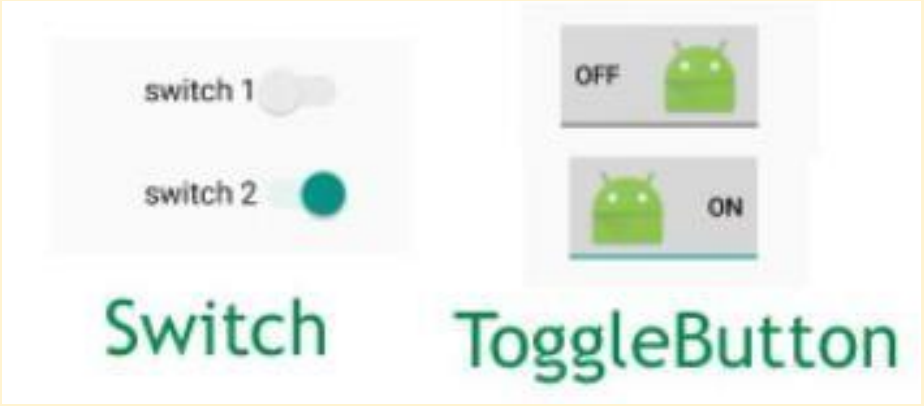
P P SAVANI UNIVERSITY


```
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // initiate view's
        simpleSwitch1 = (Switch) findViewById(R.id.simpleSwitch1);
        simpleSwitch2 = (Switch) findViewById(R.id.simpleSwitch2);
        submit = (Button) findViewById(R.id.submitButton);
        submit.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String statusSwitch1, statusSwitch2;
                if (simpleSwitch1.isChecked())
                    statusSwitch1 = simpleSwitch1.getTextOn().toString();
                else
                    statusSwitch1 = simpleSwitch1.getTextOff().toString();
                if (simpleSwitch2.isChecked())
                    statusSwitch2 = simpleSwitch2.getTextOn().toString();
                else
                    statusSwitch2 = simpleSwitch2.getTextOff().toString();
                Toast.makeText(getApplicationContext(), "Switch1 :" + statusSwitch1 + "\n" + "Switch2 :" + statusSwitch2, Toast.LENGTH_LONG).show();
                // display the current state for switch's
            }
        });

    }
```

ToggleButton Vs Switch In Android

P P SAVANI UNIVERSITY

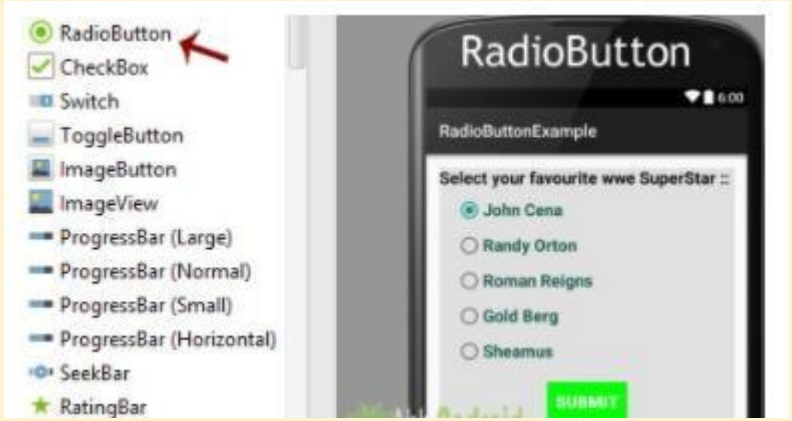
- ToggleButton allow the users to change the setting between two states like turn on/off your wifi, Bluetooth etc from your phone's setting menu.
- Since, Android 4.0 version (API level 14) there is an another kind of ToggleButton called Switch which provide the user slider control.



RadioButton and RadioGroup

P P SAVANI UNIVERSITY

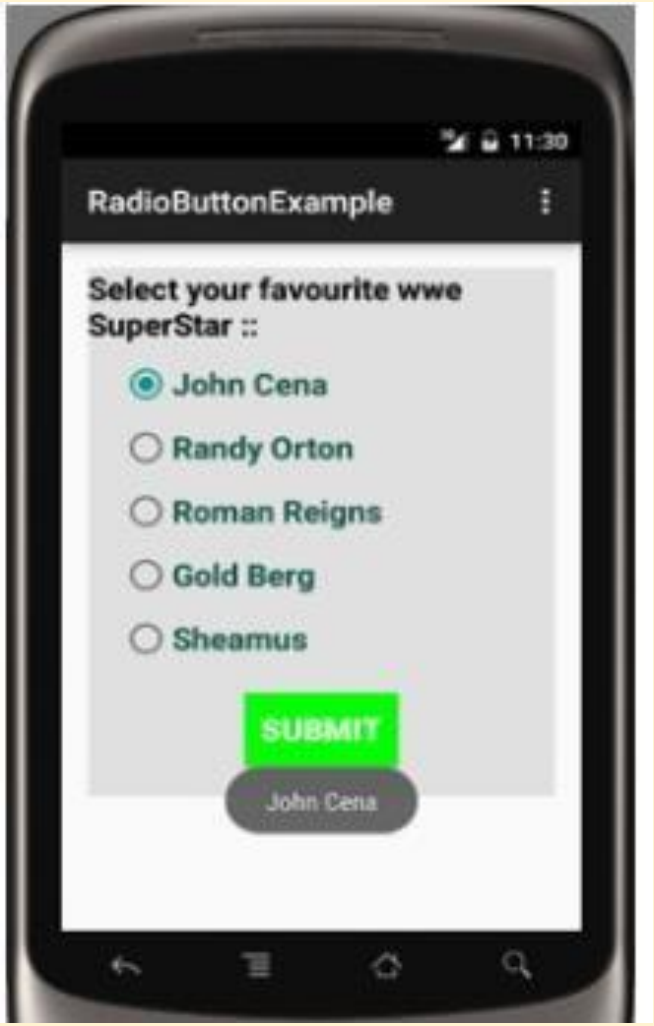
- RadioButton are mainly used together in a RadioGroup.
- In RadioGroup checking the one radio button out of several radio button added in it will automatically unchecked all the others.
- It means at one time we can checked only one radio button from a group of radio buttons which belong to same radio group.
- The most common use of radio button is in Quiz Android App code.



RadioGroup and RadioButton(Example)

P P SAVANI UNIVERSITY

```
RadioButton johnCena, randyOrton, goldBerg, romanReigns, sheamus;
String selectedSuperStar;
Button submit;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    johnCena = (RadioButton) findViewById(R.id.johnCena);
    randyOrton = (RadioButton) findViewById(R.id.randyOrton);
    goldBerg = (RadioButton) findViewById(R.id.goldBerg);
    romanReigns = (RadioButton) findViewById(R.id.romanReigns);
    sheamus = (RadioButton) findViewById(R.id.sheamus);
    submit = (Button) findViewById(R.id.submitButton);
    submit.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (randyOrton.isChecked()) {
                selectedSuperStar = randyOrton.getText().toString();
            } else if (sheamus.isChecked()) {
                selectedSuperStar = sheamus.getText().toString();
            } else if (johnCena.isChecked()) {
                selectedSuperStar = johnCena.getText().toString();
            } else if (romanReigns.isChecked()) {
                selectedSuperStar = romanReigns.getText().toString();
            } else if (goldBerg.isChecked()) {
                selectedSuperStar = goldBerg.getText().toString();
            }
            Toast.makeText(getApplicationContext(), selectedSuperStar, Toast.LENGTH_LONG).show();
        }
    });
}
```



DatePicker

P P SAVANI UNIVERSITY

- In Android, DatePicker is a widget used to select a date. It allows to select date by day, month and year in your custom UI (user interface).
- If we need to show this view as a dialog then we have to use a DatePickerDialog class.
- For selecting time Android also provides timepicker to select time

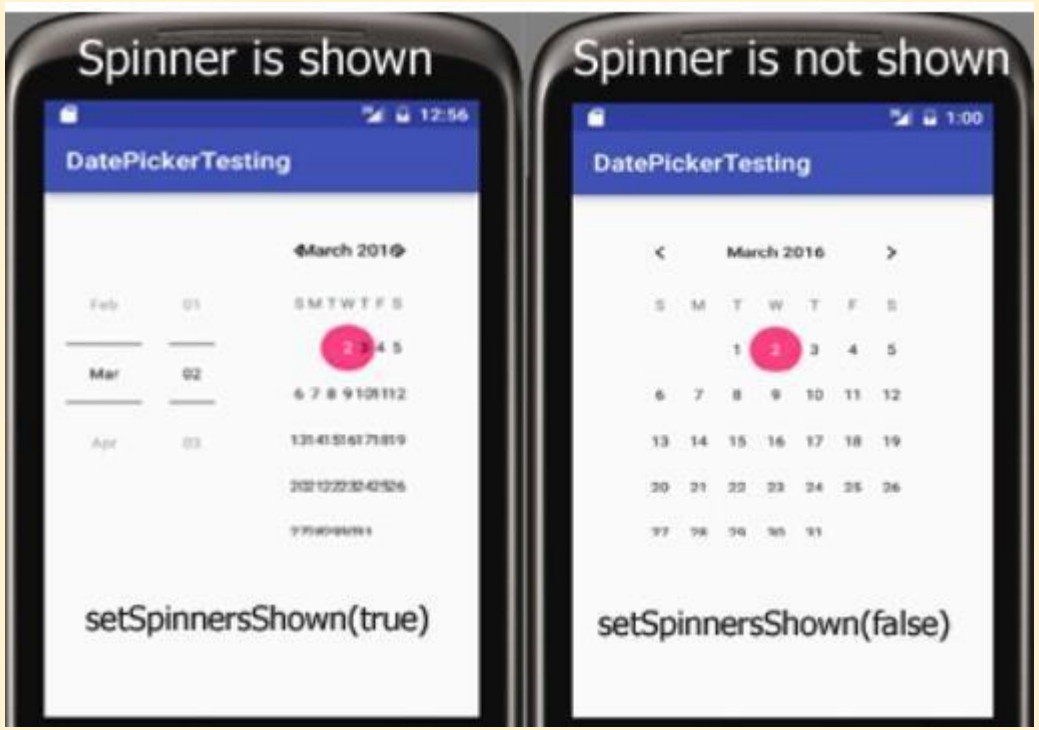
Methods of DatePicker

P P SAVANI UNIVERSITY

- **getDayOfMonth():** to get the selected day of the month from a date picker.
- **getMonth():**to get the selected month from a date picker
- **getYear():**to get the selected year from a date picker.
- **getFirstDayOfWeek():**to get the first day of the week.
- **setSpinnersShown(boolean shown) :** to set whether the spinner of the date picker in shown or not

setSpinnersShown(boolean shown) for DatePicker


```
DatePicker simpleDatePicker = (DatePicker)findViewById(R.id.simpleDatePicker); // initiate a date picker

simpleDatePicker.setSpinnersShown(false); // set false value for the spinner shown function
```



DatePicker Example in Android Studio

P P SAVANI UNIVERSITY


```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    // initiate the date picker and a button
    simpleDatePicker = (DatePicker) findViewById(R.id.simpleDatePicker);
    submit = (Button) findViewById(R.id.submitButton);
    // perform click event on submit button
    submit.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // get the values for day of month , month and year from a date picker
            String day = "Day = " + simpleDatePicker.getDayOfMonth();
            String month = "Month = " + (simpleDatePicker.getMonth() + 1);
            String year = "Year = " + simpleDatePicker.getYear();
            // display the values by using a toast
            Toast.makeText(getApplicationContext(), day + "\n" + month + "\n" + year, Toast.LENGTH_LONG).sh
        }
    });
```

DatePicker Example 2 in Android Studio


```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // initiate the date picker and a button
        date = (EditText) findViewById(R.id.date);
        // perform click event on edit text
        date.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // calender class's instance and get current date , month and year from calender
                final Calendar c = Calendar.getInstance();
                int mYear = c.get(Calendar.YEAR); // current year
                int mMonth = c.get(Calendar.MONTH); // current month
                int mDay = c.get(Calendar.DAY_OF_MONTH); // current day
                // date picker dialog
                datePickerDialog = new DatePickerDialog(MainActivity.this,
                        new DatePickerDialog.OnDateSetListener() {

                            @Override
                            public void onDateSet(DatePicker view, int year,
                                                  int monthOfYear, int dayOfMonth) {
                                // set day of month , month and year value in the edit text
                                date.setText(dayOfMonth + "/"
                                        + (monthOfYear + 1) + "/" + year);

                            }
                        }, mYear, mMonth, mDay);
                datePickerDialog.show();
            }
        });

    }


}
```

TimePicker

P P SAVANI UNIVERSITY

- In Android, TimePicker is a widget used for selecting the time of the day in either AM/PM mode or 24 hours mode.
- The displayed time consist of hours, minutes and clock format.
- If we need to show this view as a Dialog then we have to use a TimePickerDialog class.

Methods of TimePicker

P P SAVANI UNIVERSITY

1. setCurrentHour(Integer currentHour)
2. setCurrentMinute(Integer currentMinute)
3. getCurrentHour()
4. getCurrentMinute()
5. setIs24HourView(Boolean is24HourView)
6. is24HourView()
7. setOnTimeChangedListener(TimePicker.OnTimeChangedListener onTimeChangedListener):

```
TimePicker simpleTimePicker = (TimePicker)findViewById(R.id.simpleTimePicker); // initiate a time picker

simpleTimePicker.setOnTimeChangedListener(new TimePicker.OnTimeChangedListener() {
@Override
public void onTimeChanged(TimePicker view, int hourOfDay, int minute) {

}
});
```

Example of TimePicker in Android Studio


```
TextView time;
TimePicker simpleTimePicker;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    //  initiate the view's
    time = (TextView) findViewById(R.id.time);
    simpleTimePicker = (TimePicker) findViewById(R.id.simpleTimePicker);
    simpleTimePicker.setIs24HourView(false); // used to display AM/PM mode
    // perform set on time changed listener event
    simpleTimePicker.setOnTimeChangedListener(new TimePicker.OnTimeChangedListener() {
        @Override
        public void onTimeChanged(TimePicker view, int hourOfDay, int minute) {
            // display a toast with changed values of time picker
            Toast.makeText(getApplicationContext(), hourOfDay + "  " + minute, Toast.LENGTH_SHORT).show(
            time.setText("Time is :: " + hourOfDay + " : " + minute); // set the current time in text vi
        }
    });
}
```

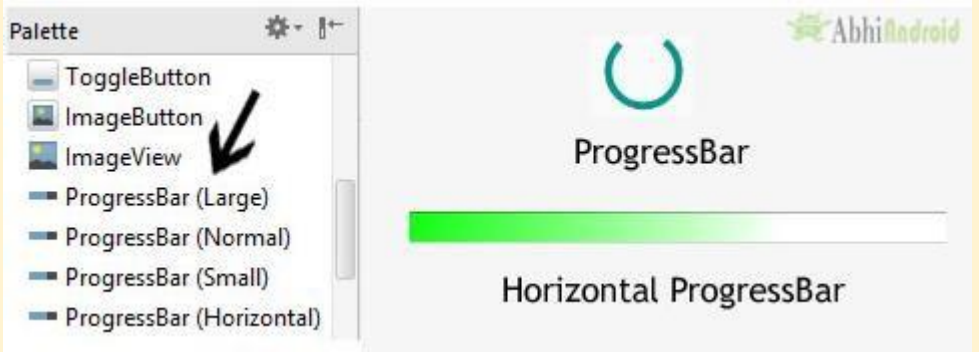
Example2 of TimePickerDialog in Android Studio

P P SAVANI UNIVERSITY

```
EditText time;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    //  initiate the edit text
    time = (EditText) findViewById(R.id.time);
    // perform click event listener on edit text
    time.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Calendar mcurrentTime = Calendar.getInstance();
            int hour = mcurrentTime.get(Calendar.HOUR_OF_DAY);
            int minute = mcurrentTime.get(Calendar.MINUTE);
            TimePickerDialog mTimePicker;
            mTimePicker = new TimePickerDialog(MainActivity.this, new TimePickerDialog.OnTimeSetListener()
                    @Override
                    public void onTimeSet(TimePicker timePicker, int selectedHour, int selectedMinute)
                        time.setText(selectedHour + ":" + selectedMinute);
                }
            }, hour, minute, true);//Yes 24 hour time
            mTimePicker.setTitle("Select Time");
            mTimePicker.show();

        }
    });
}
```

Indicating Progress with ProgressBar

- In Android, ProgressBar is used to display the status of work being done like analyzing status of work or downloading a file etc.
- In Android, by default a progress bar will be displayed as a spinning wheel but If we want it to be displayed as a horizontal bar then we need to use style attribute as horizontal.
- It mainly use the “android.widget.ProgressBar” class.



Important Methods Used In ProgressBar

1. getMax() – returns the maximum value of progress bar
2. getProgress() – returns current progress value

```
ProgressBar simpleProgressBar=(ProgressBar)findViewById(R.id.simpleProgressBar); // initiate the progress
int progressValue=simpleProgressBar.getProgress(); // get progress value from the progress bar
```

Example


```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    // initiate progress bar and start button
    final ProgressBar simpleProgressBar = (ProgressBar) findViewById(R.id.simpleProgressBar);
    Button startButton = (Button) findViewById(R.id.startButton);
    // perform click event on button
    startButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // visible the progress bar
            simpleProgressBar.setVisibility(View.VISIBLE);
        }
    });
}
```

Adjusting Progress with Seek Bars

P P SAVANI UNIVERSITY

- In Android, SeekBar is an extension of ProgressBar that adds a draggable thumb, a user can touch the thumb and drag left or right to set the value for current progress.
- SeekBar is one of the very useful user interface element in Android that allows the selection of integer values using a natural user interface. An example of SeekBar is your device's brightness control and volume control

Listener To Notify The Changes In SeekBar:

P P SAVANI
UNIVERSITY

- **SeekBar.OnSeekBarChangeListener** is a listener used as a callback that notifies client when the progress level of seekbar has been changed.
- **seekBarInstanceVariable.setOnSeekBarChangeListener(new OnSeekBarChangeListener() {…}**
  - This method is used to notify the user changes/actions in the SeekBar.

Methods Needs To Be Implemented

P P SAVANI UNIVERSITY

1.public void onProgressChanged (SeekBar seekBar, int progresValue, boolean fromUser) {...} –
This listener method will be invoked if any change is made in the SeekBar.

2. public void onStartTrackingTouch(SeekBar seekBar) {...} –
This listener method will be invoked at the start of user's touch event. Whenever a user touch the thumb for dragging this method will automatically called.

3. public void onStopTrackingTouch(SeekBar seekBar) {...} –
This listener method will be invoked at the end of user touch event. Whenever a user stop dragging the thump this method will be automatically called.

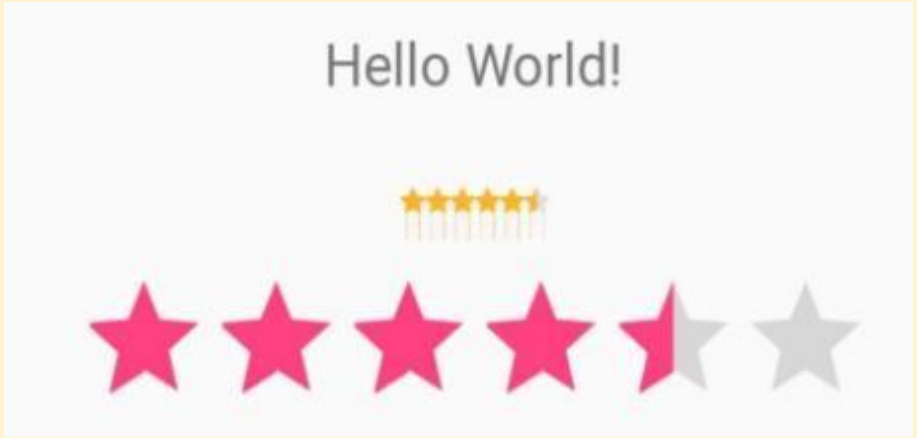
Example


```java
public class MainActivity extends AppCompatActivity {

    Button submitButton;
    SeekBar simpleSeekBar;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // initiate  views
        simpleSeekBar=(SeekBar)findViewById(R.id.simpleSeekBar);
        // perform seek bar change listener event used for getting the progress value
        simpleSeekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
            int progressChangedValue = 0;

            public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
                progressChangedValue = progress;
            }

            public void onStartTrackingTouch(SeekBar seekBar) {
                // TODO Auto-generated method stub
            }

            public void onStopTrackingTouch(SeekBar seekBar) {
                Toast.makeText(MainActivity.this, "Seek bar progress is :" + progressChangedValue,
                        Toast.LENGTH_SHORT).show();
            }
        });

    }
```

Displaying Rating Data with RatingBaí

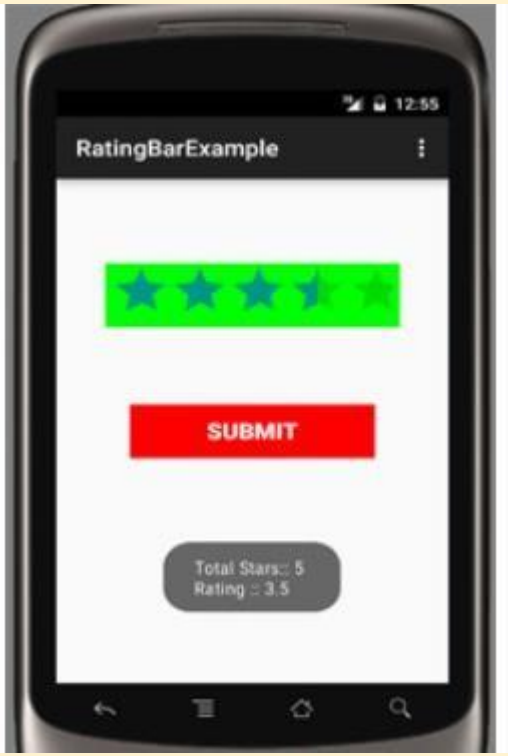
P P SAVANI UNIVERSITY

- Although the SeekBar is useful for allowing a user to set a value, such as the volume, the RatingBar has a more specific purpose: showing ratings or getting a rating from a user.
- By default, this progress bar uses the star paradigm with five stars by default. A user can drag across this horizontal to set a rating



RatingBar Example

P P SAVANI UNIVERSITY

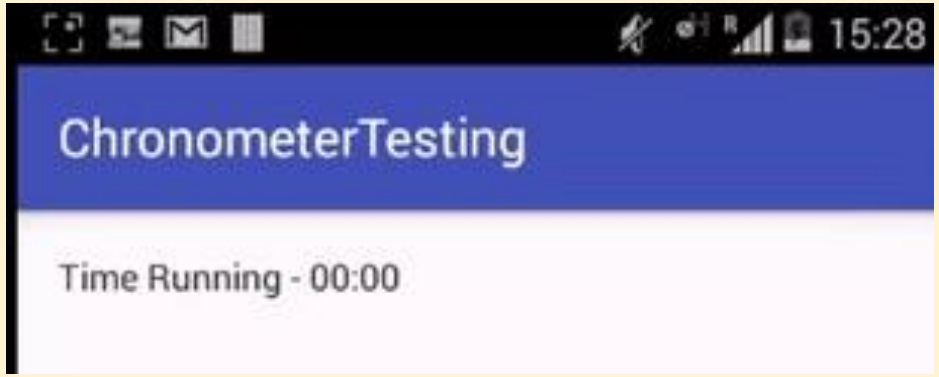



```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // initiate rating bar and a button
        final RatingBar simpleRatingBar = (RatingBar) findViewById(R.id.simpleRatingBar);
        Button submitButton = (Button) findViewById(R.id.submitButton);
        // perform click event on button
        submitButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // get values and then displayed in a toast
                String totalStars = "Total Stars:: " + simpleRatingBar.getNumStars();
                String rating = "Rating :: " + simpleRatingBar.getRating();
                Toast.makeText(getApplicationContext(), totalStars + "\n" + rating, Toast.LENGTH_LONG).shou
            }
        });
    }


}
```

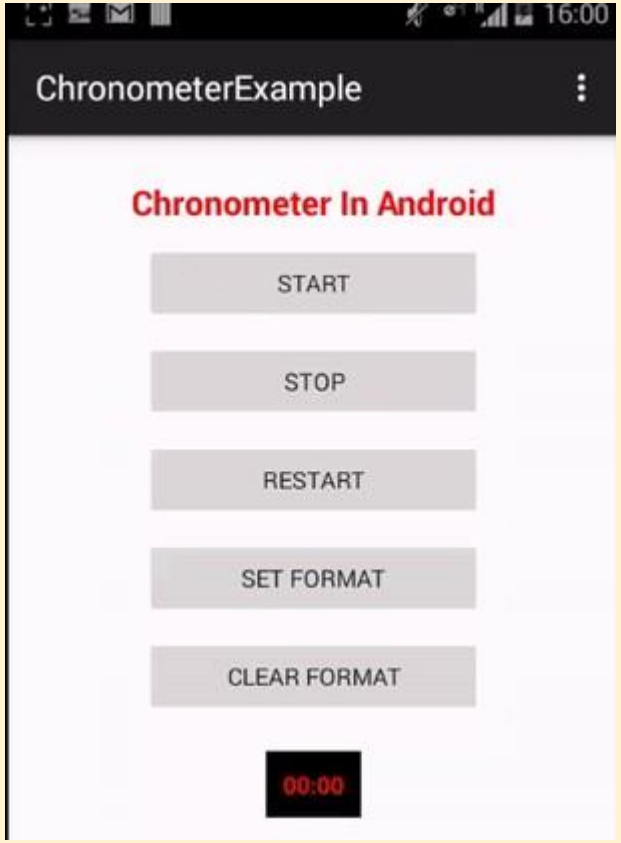
Showing Time Passage with Chronometer

P P SAVANI UNIVERSITY

- In Android, Chronometer is a class that implements a simple timer. Chronometer is a subclass of TextView. This class helps us to add a timer in our app

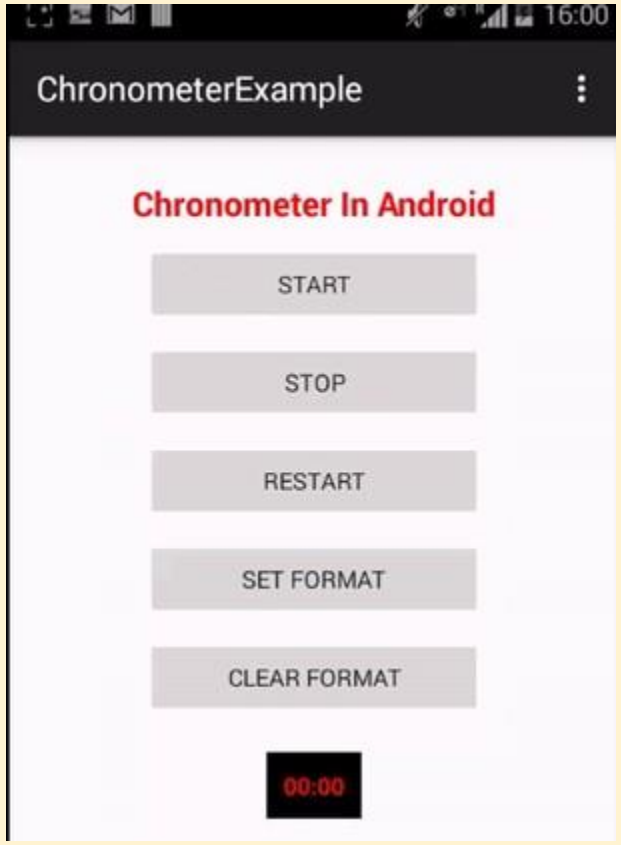


Chronometer Example




```
Chronometer simpleChronometer;
Button start, stop, restart, setFormat, clearFormat;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    // initiate views
    simpleChronometer = (Chronometer) findViewById(R.id.simpleChronometer);
    start = (Button) findViewById(R.id.startButton);
    stop = (Button) findViewById(R.id.stopButton);
    restart = (Button) findViewById(R.id.restartButton);
    setFormat = (Button) findViewById(R.id.setFormat);
    clearFormat = (Button) findViewById(R.id.clearFormat);
    // perform click  event on start button to start a chronometer
    start.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View v) {
            // TODO Auto-generated method stub

            simpleChronometer.start();
        }
    });

    // perform click  event on stop button to stop the chronometer
    stop.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View v) {
            // TODO Auto-generated method stub

            simpleChronometer.stop();
        }
    });
```

Chronometer Example




```
    // perform click  event on restart button to set the base time on chronometer
    restart.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View v) {
            // TODO Auto-generated method stub

            simpleChronometer.setBase(SystemClock.elapsedRealtime());
        }
    });

    // perform click  event on set Format button to set the format of chronometer
    setFormat.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View v) {
            // TODO Auto-generated method stub

            simpleChronometer.setFormat("Time (%s)");
        }
    });

    // perform click  event on clear button to clear the current format of chronmeter as you set thro
    clearFormat.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View v) {
            // TODO Auto-generated method stub

            simpleChronometer.setFormat(null);
        }
    });
}
```