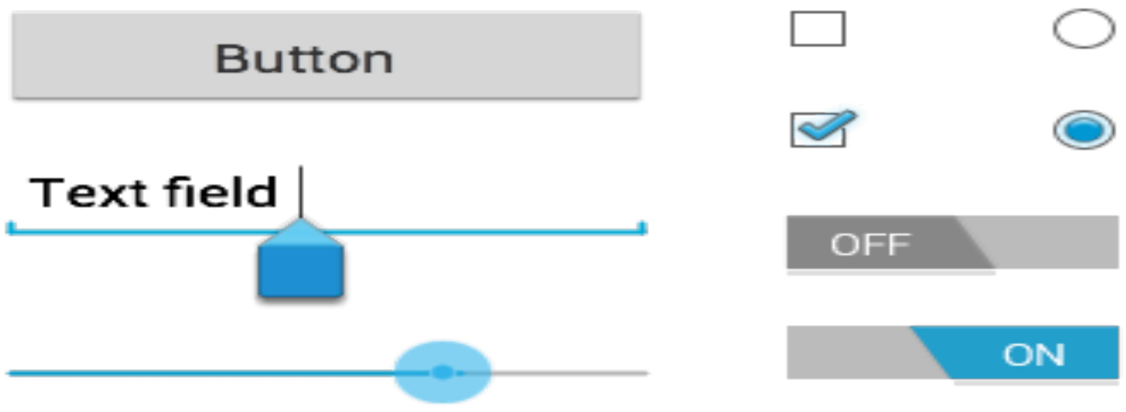
Android UI Control

Android – UI Control

UI or Input

- In android **UI** or **input** controls are the interactive or View components that are used to design the user interface of an application. In android we have a wide variety of UI or input controls available, those are TextView, EditText, Buttons, Checkbox, ProgressBar, Spinners, etc.

- Widgets are subclasses of View. They are used to create interactive UI components such as buttons, checkboxes, labels, text fields, etc.



Android – UI Control

UI or Input

- Generally, in android the user interface of an app is made with a collection of **View** and **ViewGroup** objects.

- The **View** is a base class for all UI components in android and it is used to create interactive UI components such as TextView, EditText, Checkbox, Radio Button, etc. and it is responsible for event handling and drawing.

- The **ViewGroup** is a subclass of **View** and it will act as a base class for layouts and layout parameters. The ViewGroup will provide invisible containers to hold other Views or ViewGroups and to define the layout properties.

- In android, we can define a UI or input controls in two ways, those are
    - Declare UI elements in XML
    - Create UI elements at runtime

Android – UI Control

Declare UI Elements in XML

- Following is the example of defining UI controls (TextView, EditText, Button) in the XML file (**activity_main.xml**) using LinearLayout.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"    android:layout_width="match_parent"
  android:layout_height="match_parent">
  <TextView
    android:id="@+id/fstTxt"   android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Enter Name" />
  <EditText
    android:id="@+id/name"  android:layout_width="wrap_content"
    android:layout_height="wrap_content"       android:ems="10"/>
  <Button
    android:id="@+id/getName"        android:layout_width="wrap_content"
    android:layout_height="wrap_content"       android:text="Get Name" />
</LinearLayout>
```

Android – UI Control

Load XML Layout File from an Activity

- Once we are done with the creation of layout with UI controls, we need to load the XML layout resource from our activity **onCreate()** callback method as shown below.

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

Android – UI Control

Create UI Element at Runtime

- Create our own custom **View** and **ViewGroup** objects programmatically with required layouts.

- Following is the example of creating UI elements (TextView, EditText, Button) in LinearLayout using custom **View** and **ViewGroup** objects in an activity programmatically.

Android – UI Control

Create UI Element at Runtime

```
public class MainActivity extends AppCompatActivity {
        @Override
        protected void onCreate(Bundle savedInstanceState) {
           super.onCreate(savedInstanceState);
           TextView textView1 = new TextView(this);
           textView1.setText("Name:");
           EditText editText1 = new EditText(this);
           editText1.setText("Enter Name");
           Button button1 = new Button(this);
           button1.setText("Add Name");
           LinearLayout linearLayout = new LinearLayout(this);
           linearLayout.addView(textView1);
           linearLayout.addView(editText1);
           linearLayout.addView(button1);
           setContentView(linearLayout);
        }
    }
```

Android – UI Control

Width and Height

- When we define a UI controls in a layout using an XML file, we need to set width and height for every **View** and **ViewGroup** elements using **layout_width** and **layout_height** attributes.

- We used different values to set layout_width and layout_height, those are
  - match_parent
  - wrap_content

- If we set value **match_parent**, then the **View** or **ViewGroup** will try to match with parent width or height.

- If we set value **wrap_content**, then the **View** or **ViewGroup** will try to adjust its width or height based on the content.

Android – UI Control

Different Types of UI Controls

- TextView
- EditText
- AutoCompleteTextView
- Button
- ImageButton
- ToggleButton
- CheckBox
- RadioButton
- RadioGroup
- ProgressBar
- Spinner
- TimePicker
- DatePicker
- SeekBar
- AlertDialog
- Switch
- RatingBar

Android – UI Control

Different Types of UI Controls

- **Android TextView :** In android, TextView is a user interface control that is used to display the text to the user.

- **Android EditText :** In android, EditText is a user interface control which is used to allow the user to enter or modify the text.

- **Android AutoCompleteTextView :** In android, AutoCompleteTextView is an editable text view which is used to show the list of suggestions based on the user typing text. The list of suggestions will be shown as a dropdown menu from which the user can choose an item to replace the content of the textbox.

- **Android Button :** In android, **Button** is a user interface control that is used to perform an action when the user clicks or tap on it.

Android – UI Control

Different Types of UI Controls

- **Android Image Button :** In android, **Image Button** is a user interface control that is used to display a button with an image to perform an action when the user clicks or tap on it.
    - Generally, the Image button in android looks similar as regular Button and perform the actions same as regular button but only difference is for image button we will add an image instead of text.
- **Android Toggle Button :** In android, **Toggle Button** is a user interface control that is used to display ON (Checked) or OFF (Unchecked) states as a button with a light indicator.
- **Android CheckBox :** In android, **Checkbox** is a two-states button that can be either checked or unchecked.
- **Android Radio Button :** In android, **Radio Button** is a two-states button that can be either checked or unchecked and it cannot be unchecked once it is checked.

Android – UI Control

Different Types of UI Controls

- **Android Radio Group :** In android, **Radio Group** is used to group one or more radio buttons into separate groups based on our requirements.
    - In case if we group radio buttons using the radio group, at a time only one item can be selected from the group of radio buttons.
- **Android ProgressBar :** In android, **ProgressBar** is a user interface control which is used to indicate the progress of an operation.
- **Android Spinner :** In android, **Spinner** is a drop-down list which allows a user to select one value from the list.
- **Android TimePicker :** In android, **TimePicker** is a widget for selecting the time of day, either in 24-hour or AM/PM mode.
- **Android DatePicker :** In android, DatePicker is a widget for selecting a date.
- **Chronometer** — A Text View extension that implements a simple count-up timer.

Android – UI Control

Different Types of UI Controls

- **ViewFlipper** — A View Group that lets you define a collection of Views as a horizontal row in which only one View is visible at a time, and in which transitions between visible views can be animated.

- **VideoView** — Handles all state management and display Surface configuration for playing videos more simply from within your Activity.

- **QuickContactBadge** — Displays a badge showing the image icon assigned to a contact you specify using a phone number, name, email address, or URI. Clicking the image will display the quick contact bar, which provides shortcuts for contacting the selected contact — including calling and sending an SMS, email, and IM.

- **ViewPager** — Released as part of the Compatibility Package, the View Pager implements a horizontally scrolling set of Views similar to the UI used in Google Play and Calendar. The View Pager allows users to swipe or drag left or right to switch between different Views.

Android – Views

View and ViewGroup

- In android, **Layout** is used to define the user interface for an app or activity and it will hold the UI elements that will appear to the user.

- The user interface in an android app is made with a collection of View and ViewGroup objects.

- The android apps will contain one or more activities and each activity is a one screen of app.

- The activities will contain a multiple UI components and those UI components are the instances of View and ViewGroup subclasses.

Android – Views

View and ViewGroup

- **Android View:** The View is a base class for all UI components in android. For example, the **EditText** class is used to accept the input from users in android apps, which is a subclass of View.
- Following are the some of common View subclasses that will be used in android applications.
  TextView, EditText, Button, Checkbox, RadioButton, ImageButton, ProgressBar, Spinner
- **Android ViewGroup** : The ViewGroup is a subclass of View and it will act as a base class for layouts and layouts parameters. The ViewGroup will provide an invisible containers to hold other Views or ViewGroups and to define the layout properties.
- For example, Linear Layout is the ViewGroup that contains a UI controls like button, textview, etc. and other layouts also.
- Following are the commonly used ViewGroup subclasses in android applications. Linear Layout, Relative Layout, Table Layout, Frame Layout, Web View, List View, Grid View

Android – Views

Handling User Interaction Events

- **onKeyDown** — Called when any device key is pressed; includes the keyboard, hang-up, call, back, and camera buttons.
- **onKeyUp** — Called when a user releases a pressed key
- **onTrackballEvent** — Called when the device's trackball is moved
- **onTouchEvent** — Called when the touchscreen is pressed or released, or when it detects movement

THANKS