
# MOBILE APPLICATION DEVELOPMENT (EI-333)

## Lecture:04

## "BASIC VIEWS"

ANDROID

# BASIC VIEWS

ANDROID

# WHAT IS A VIEW?

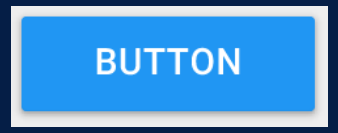
`View` subclasses are basic user interface building blocks

- Display text (`TextView` class), edit text (`EditText` class)
- Buttons (Button class), menus, other controls
- Scrollable (ScrollView, RecyclerView)
- Show images (ImageView)
- CheckBox
- RadioButton

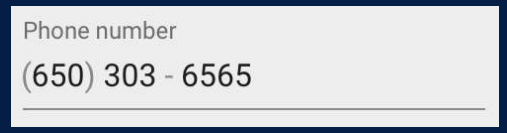
etc.

# Examples of VIEW subClass

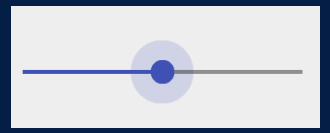
Button




EditText



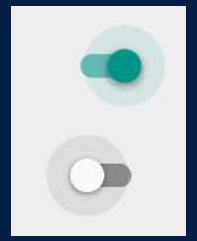

Slider



CheckBox

RadioButton

Switch



ANDROID

# VIEW ATTRIBUTES

- Color, dimensions, positioning
- May have focus (e.g., selected to receive user input)
- May be interactive (respond to user clicks)
- May be visible or not
- Relationships to other views

ANDROID

# Attributes

- Every View and ViewGroup object supports their own variety of XML attributes
- Some attributes are specific to a View object , these attributes are inherited by any View objects that extend this class
- Some attributes are common to all View objects, because they are inherited from the root View class
- Other attributes are considered "layout parameters "that describe certain layout orientations of the View object

# ID attribute

- Any View object may have an integer ID
- uniquely identify the View within the tree
- the ID is typically assigned in the layout XML file as a string
- This attribute is common to all View objects

android:id"=@+id/my_button"

# Using the ID

- The syntax for an ID, inside an XML tag is:

  android:id"=@+id/my_button"

- Referencing an Android resource ID:

  android:id="@android:id/my_button"

In the layout.xml file:

```
<Button android:id"=@+id/my_button "
        android:layout_width"=wrap_content "
        android:layout_height"=wrap_content "
        android:text"=@string/my_button_text/">
```

In the java code:

```
Button myButton = (Button) findViewById(R.id.my_button);
```

android

# Text View

- Displays text to the user and optionally allows them to edit it
- TextView is a complete text editor, however the basic class is configured to not allow editing

# Edit Text

- EditText is a thin veneer over TextView that configures itself to be editable
- Properties:
  - capitalize to have the control capitalize words, the beginning of sentences
  - phoneNumber property if you need to accept a phone number
  - password property if you need a password field
  - single line by setting the singleLine property to true

# Button

- Button represents a push-button widget
- Push-buttons can be pressed, or clicked, by the user to perform an action

# Button XML declaration

```
<Button android:id="@+id/ccbtn1"
        android:text="@+string/basicBtnLabel"
        android:typeface="serif"
        android:textStyle="bold"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />
```

# Basic Views

- Button
- ImageButton
- CheckBox
- ToggleButton
- RadioButton
- RadioGroup

# Button

- Represents a push-button widget

```
<Button android:id="@+id/btnOpen"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Open" />
```

# ImageButton

- Similar to the Button view, except that it also displays an image

```
<ImageButton android:id="@+id/btnImg1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:src="@drawable/icon" />
```

# CheckBox

- A special type of button that has two states: checked or unchecked

```
<CheckBox android:id="@+id/chkAutosave"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Autosave" />

<CheckBox android:id="@+id/star"
    style="?android:attr/starStyle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

# RadioGroup and RadioButton

- The RadioButton has two states: either checked or unchecked. Once a RadioButton is checked, it cannot be unchecked.
- A RadioGroup is used to group together one or more RadioButton views, thereby allowing only one RadioButton to be checked within the RadioGroup.

```
<RadioGroup android:id="@+id/rdbGp1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical" >
    <RadioButton android:id="@+id/rdb1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Option 1" />
    <RadioButton android:id="@+id/rdb2"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Option 2" />
</RadioGroup>
```

# ToggleButton

- Displays checked/unchecked states using a light indicator

```
<ToggleButton android:id="@+id/toggle1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

# ProgressBar View

- Provides visual feedback of some ongoing tasks, such as when you are performing a task in the background.
- For example, you might be downloading some data from the Web and need to update the user about the status of the download.

```
<ProgressBar android:id="@+id/progressbar"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

# AutoCompleteTextView View

- The AutoCompleteTextView is a view that is similar to EditText (in fact it is a subclass of EditText), except that it shows a list of completion suggestions automatically while the user is typing.

```
<AutoCompleteTextView android:id="@+id/txtCountries"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content" />
```

# THANK YOU ☺