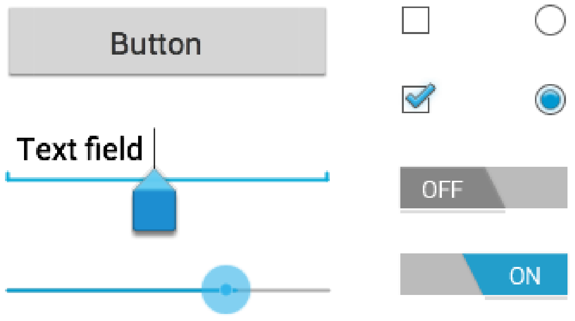
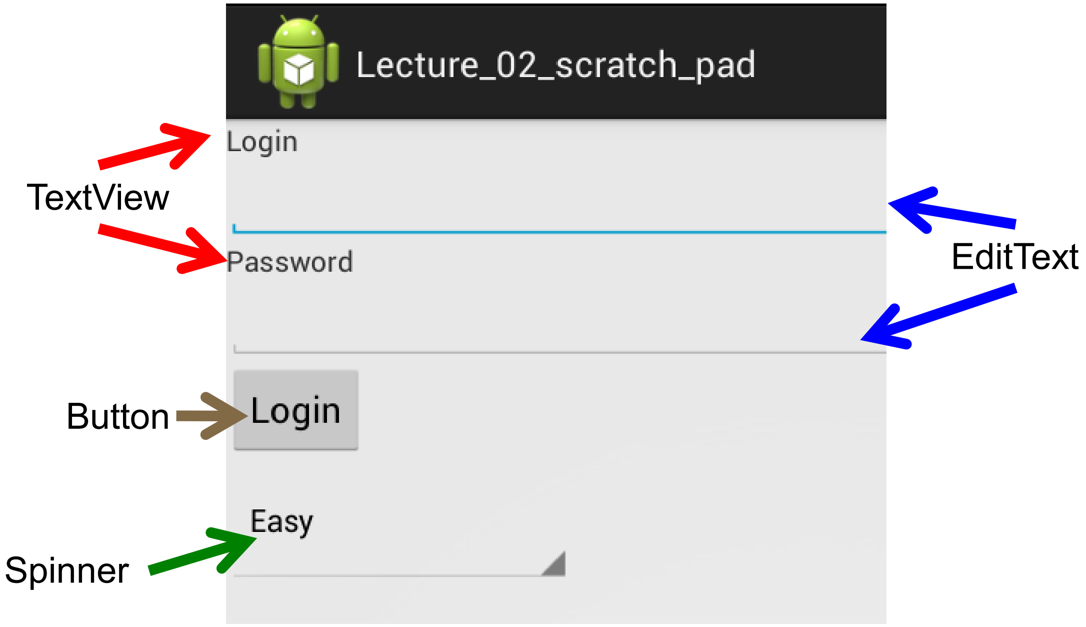
CONTROLS

Input Controls

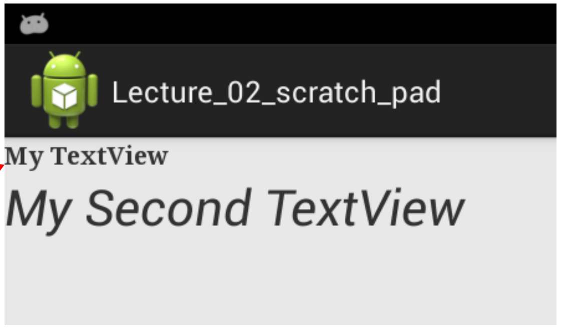
- UI Elements to be displayed on the screen
- Tools to help you build up your application
- Built based on View object





TextView

```
<TextView
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:textStyle="bold"
   android:typeface="serif"
   android:textSize="20px"
   android:text="My TextView"
   />
```



My TextView

My Second TextView

Size of Widget
- **fill_parent**
- **wrap_content**
- **px is one pixel. scale-independent pixels (sp) and density-independent pixels (dip)**

Font Style
- **Bold**
- **Italics**
- **Normal**

Font Type
- **Sans**
- **Serif**
- **Monospace**

Display Text Size

Text to be displayed

Reference Widgets from Kotlin

```
<TextView
    android:id="@+id/tvDisplay"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textStyle="bold"
    android:typeface="serif"
    android:textSize=”20sp"
    android:text=”For Display Only"
/>
```

- @+id is used when we want to specify an ID for a particular view object.
- @+id will add a resource into a R.class.

Reference Widgets from Kotlin- TextView

```
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)



         tvDisplay.text = "I am a changed text"
      }
}
```

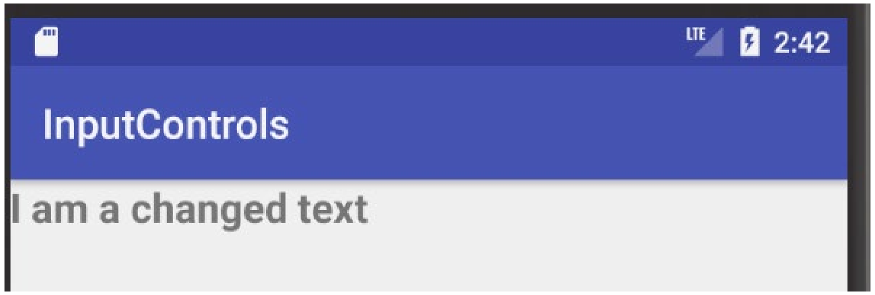
Import from XML resource

Update text value

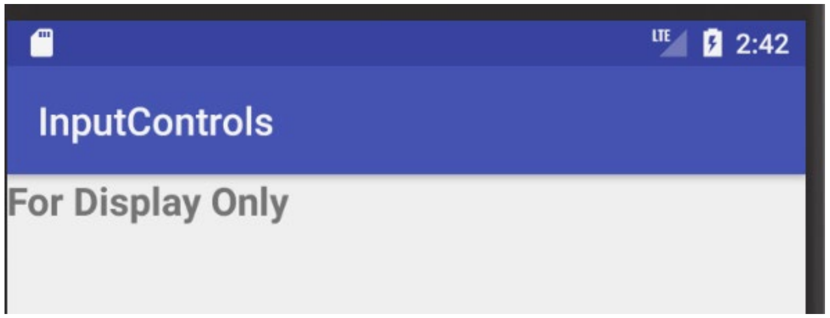
TextView ID

Reference Widgets from Java - TextView

TextView changed!!



I am a changed text



For Display Only

Reference Widgets from Kotlin- EditText

```
<EditText
  android:id="@+id/etDemo"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content"
  android:hint=”For Demo purpose"
/>
```

Reference Widgets from Kotlin- TextView

```
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        var strUserInput:String?=etDemo.text.toString()

    }
}
```

Import from XML resource

Method to retrieve String value

EditText ID

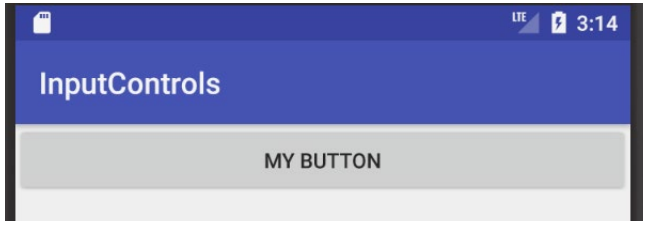
Text property

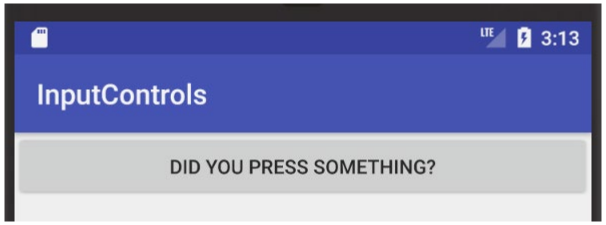
Implement Widget Listener Method 1 - Button

```
<Button
    android:id="@+id/btnDemo"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content"
  android:text="My Button"
/>
```

Implement Widget Listener Method 1 - Button

```
class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

       btnDemo.setOnClickListener(
            {

                btnDemo.text = "Did you press something?"
            }

        )

    }
}
```





Implement Widget Listener Method 2 - Button

```
<Button
   android:id="@+id/myButton"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content"
  android:text="My Button“
   android:onClick=“demoClickHandler”
/>
```

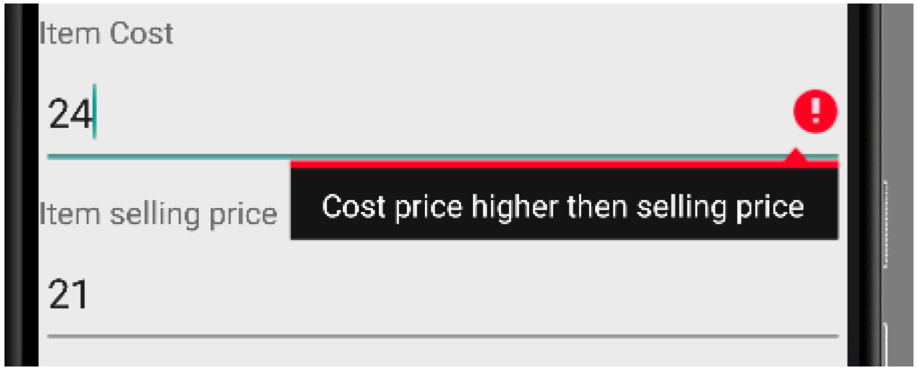
Implement Widget Listener Method 2 - Button

```
class MainActivity : AppCompatActivity() {

        override fun onCreate(savedInstanceState: Bundle?) {
            super.onCreate(savedInstanceState)
            setContentView(R.layout.activity_main)

    }//End of onCreate method

   fun demoClickHandler(v: View?)
   {

       btnDemo.text = "Is it time to do work?"

   }


}
```

EditText – Display error message

- setError method in EditText class
- Displays error input to user

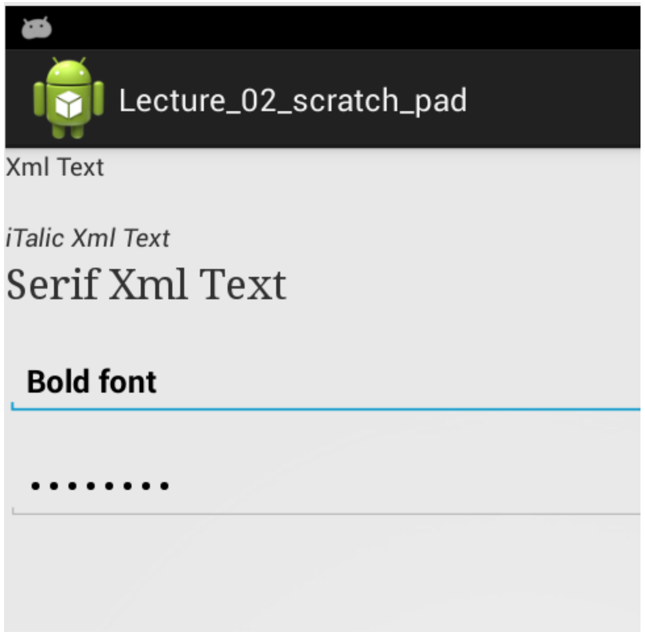
```
if (costPrice > sellingPrice) {
    etItemCostPrice.setError("Cost price higher then selling price");
}
```

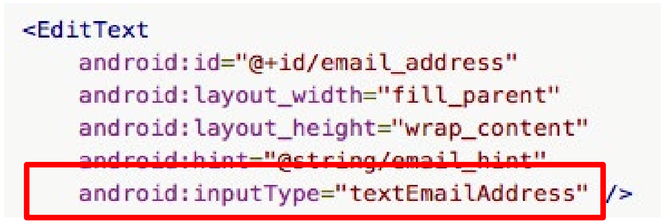


Cost price higher then selling price

Input Controls - EditText

- Allows user to specify keyboard using the following inputType
  - “text” - Normal text keyboard.
  - "textEmailAddress” – Normal text keyboard with the @ character.
  - "number” – Basic number keypad.
  - "phone” – Phone-style keypad.
- Other behavior
  - “textCapSentences” – capitalizes the first letter for each new sentence.
  - "textCapWords” – capitalizes every word.
  - "textAutoCorrect” – corrects commonly misspelled words.
  - "textPassword” – characters entered turn into dots.
  - "textMultiLine” – allow users to input long strings of text that include line breaks (carriage returns).



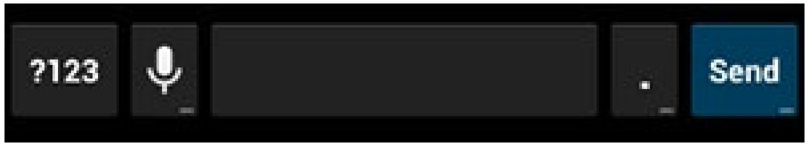


```
<EditText
    android:id="@+id/email_address"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/email_hint"
    android:inputType="textEmailAddress" />
```

Source : https://developer.android.com/guide/topics/ui/controls/text.html

Input Controls - EditText

- Keyboard Actions
  - Allows you to specify an action to be made when users have completed their input
  - Action specifies the button that appears in place of the carriage return key and the action to be made, such as "Search" or "Send."
  - specify the action by setting the android:imeOptions attribute.
- Responding to action button events



```xml
<EditText
    android:id="@+id/search"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/search_hint"
    android:inputType="text"
    android:imeOptions="actionSend" />
```

```kotlin
search.setOnEditorActionListener(object:TextView.OnEditorActionListener{

    override fun onEditorAction(p0: TextView?, p1: Int, p2: KeyEvent?): Boolean {

        var handled = false
        if(p1 == EditorInfo.IME_ACTION_SEND)
        {
            sendMessage()
            handled = true
        }

        return handled

    }
})
```

Source : https://developer.android.com/guide/topics/ui/controls/text.html