User Interface (UI) Controls

Asst. Prof. Monlica Wattana, Ph.D
Department of Computer Science,
Khon Kaen University

342 267 Mobile Device Programming

Outline

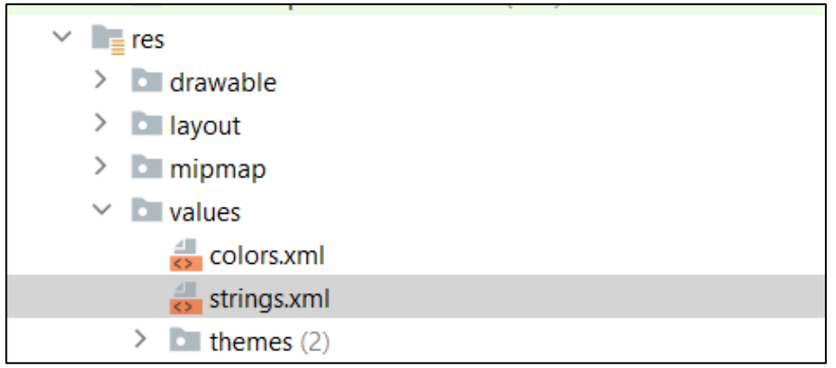
- String resources
- Toast
- TextView
- Button
- EditText
- ImageView
- CheckBox
- RadioButton
- Spinner
- Pickers
- ScrollView

String resources

- A string resource provides text strings for an application with optional text styling and formatting.
- There are three types of resources that can provide an application with strings:

    1. String :XML resource that provides a single string.
    2. String Array :XML resource that provides an array of strings.
    3. Quantity Strings (Plurals) : XML resource that carries different strings for pluralization.

- Path: app>> res>> values>> string.xml



String resources

1. String syntax:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <string  name="string_name ">text_string</string>
</resources>
```

Example: XML file saved at res/values/strings.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello!</string>
</resources>
```

String resources

Example: This layout XML applies a string to a View:

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

Example: This application code retrieves a string.

```
val string: String = getString(R.string.hello)
```

String resources

2. String array - syntax:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array  name="string_array_name">
        <item> text_string1 </item>
        <item> text_string2 </item>
        <item> text_string3 </item>
        ……..
    </string-array>
</resources>
```

String resources

2. String array

   Example: XML file saved at res/values/strings.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="planets_array">
        <item>Mercury</item>
        <item>Venus</item>
        <item>Earth</item>
        <item>Mars</item>
    </string-array>
</resources>
```

String resources

2. String array

Example: This application code retrieves a string.

```
val array: Array = resources.getStringArray(R.array.planets_array)
```

** XML layout could not access an array directly.

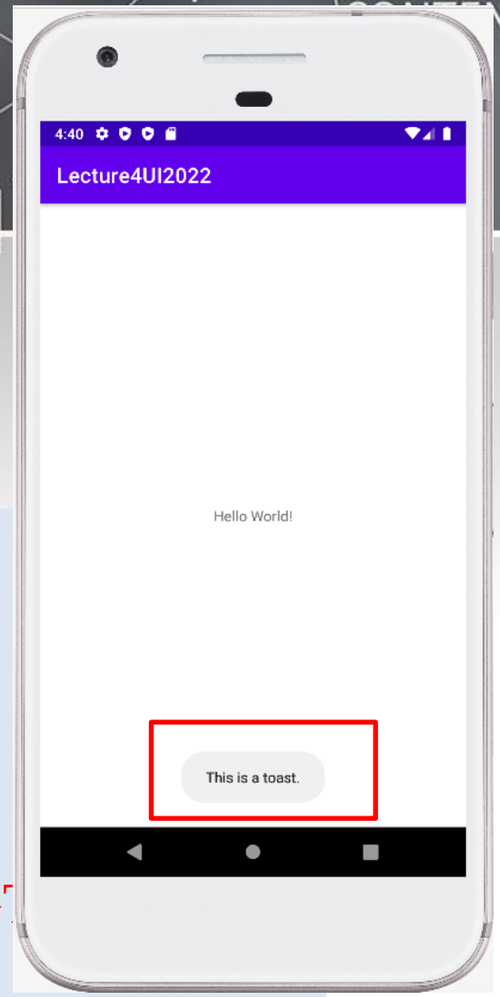
Special Characters

Character	Escaped form(s)
@	\@
?	\?
<	&lt;
&	&amp;
Single quote (')	Any of the following: • ' • \' • Enclose the entire string in double quotes ("This'll work", for example)
Double quote (")	Any of the following: • " • \"

Toast

- Toast is used to display a piece of text for a short span of time.

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

         val  toast :Toast = Toast.makeText ( this, "This is a toast.",Toast.LENG
               toast.show ( )
        //Toast.makeText ( this, "This is a toast.",Toast.LENGTH_LONG ).show ( )
    }
}
```



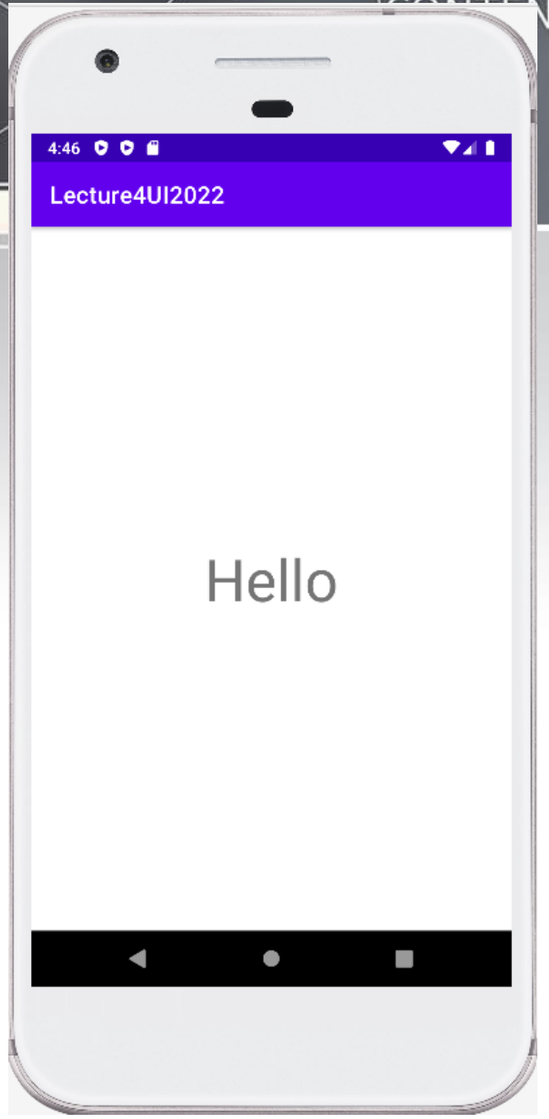
Toast.LENGTH_LONG

Toast.LENGTH_SHORT

XML Code to MainActivity.kt

Activity_main.xml

```
<TextView
        android:id="@+id/textView1"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:textSize="‘50sp" />
```



- Java to Kotlin : findViewByID()

MainActivity.kt

```
val text1 : TextView = findViewById(R.id.textView1)
    text1.text  = "Hello"
```

View binding

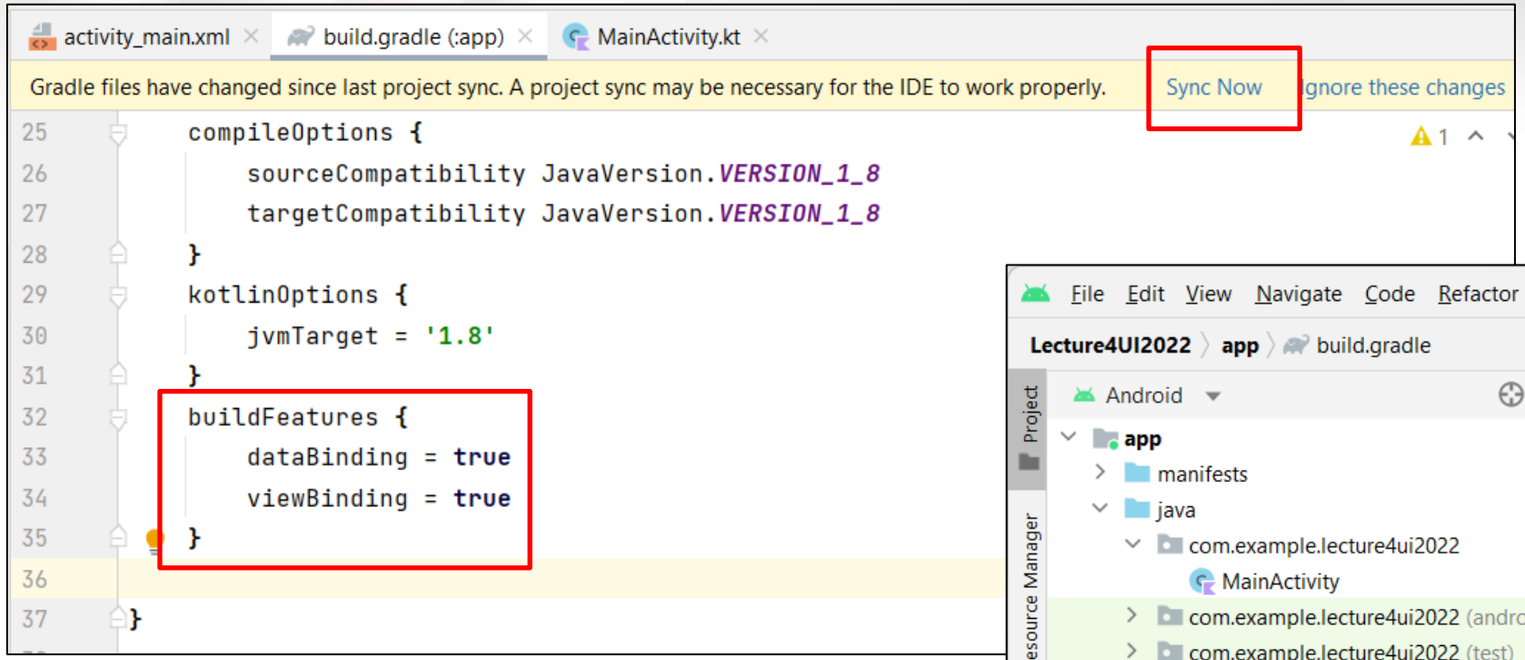
1. build.gradle file (Module: app) and Sync Now

```
android {
 ….
 ……
   buildFeatures {
        dataBinding = true
        viewBinding = true
    }
}
```

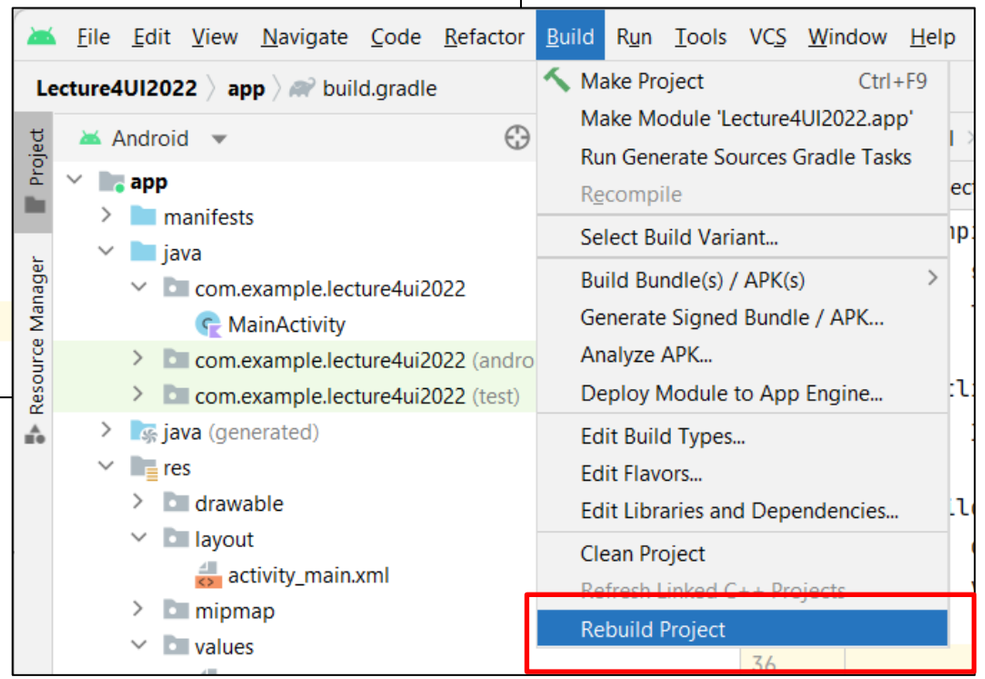
2. Menu Build >> Rebuild Project

View binding

1. build.gradle file (Module: app) and Sync Now



2. Menu Build >> Rebuild Project



View binding

3. MainActivity.kt

```
class MainActivity : AppCompatActivity() {

    private lateinit var binding: ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
    }
}
```

Text View

- Extends 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

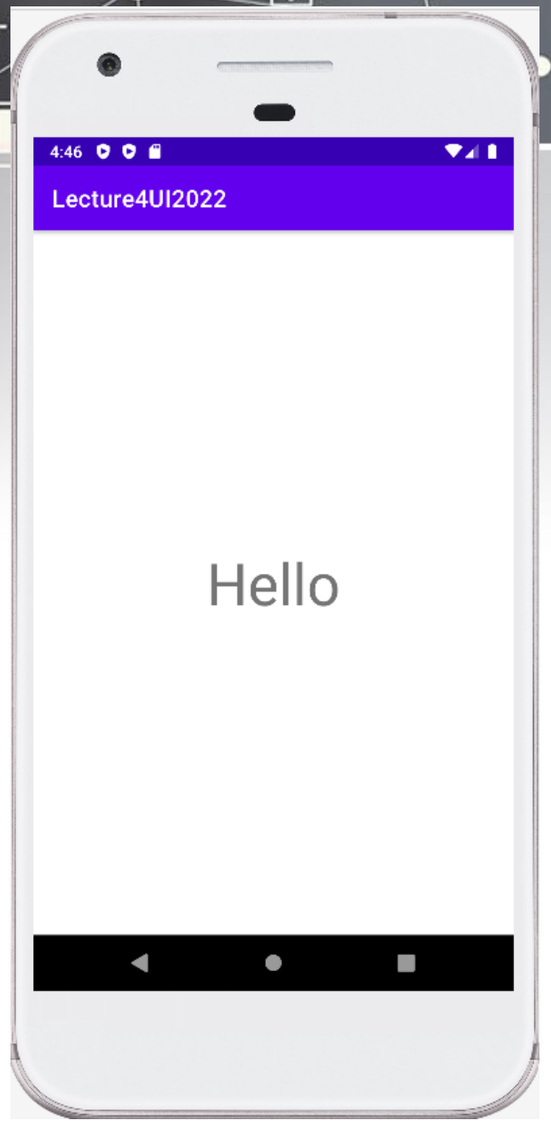
Text View

Example: XML file saved at res/values/strings.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello</string>
</resources>
```

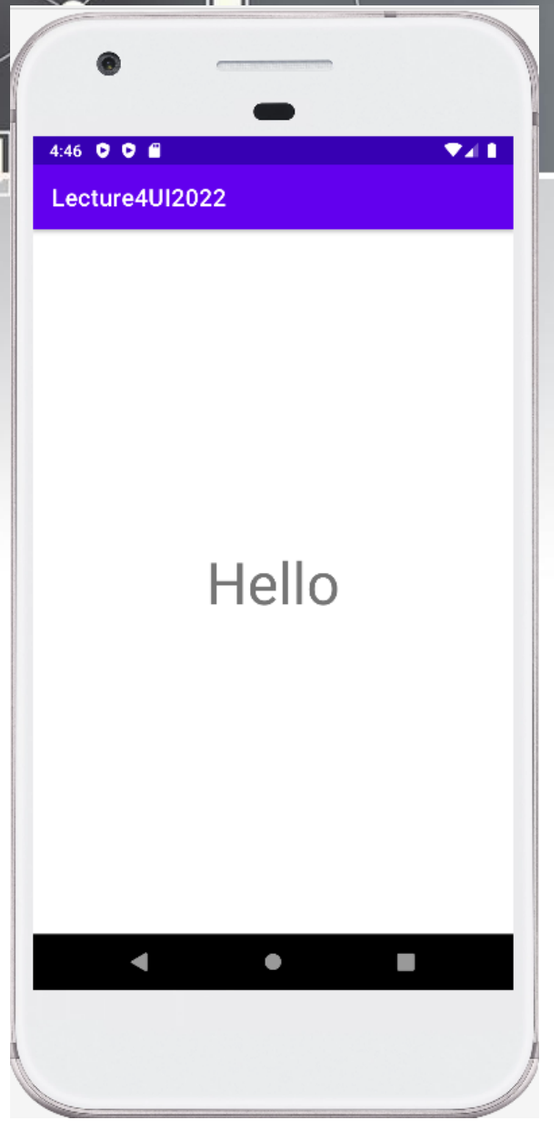
Example: This layout XML applies a string to a View:

```
<TextView
    android:id="@+id/textView1"
    android:layout_width=" wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    android:textSize="50sp"
    android:layout_centerInParent="true" />
```



Text View : MainActivity.kt

```
class MainActivity : AppCompatActivity() {
    private lateinit var  binding : ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

        binding.textViwe1.text = "Hello"
    }
}
```

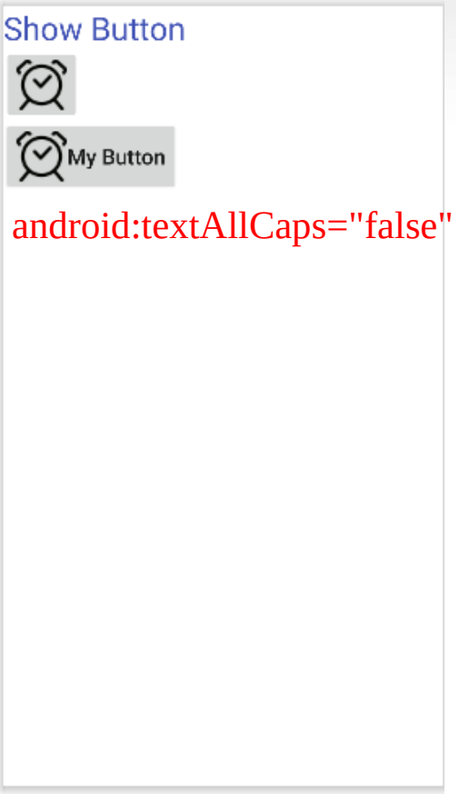


ImageButton & Button

- A Button control provides an area on the UI where a user can touch to initiate an action.
- Tag: <Button> or
  <androidx.appcompat.widget.AppCompatButton>

```
<ImageButton
        android:id="@+id/imageButton_id"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:src="@drawable/icon_name" />
```

```
<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/button_id"
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
   android:text="text" />
```

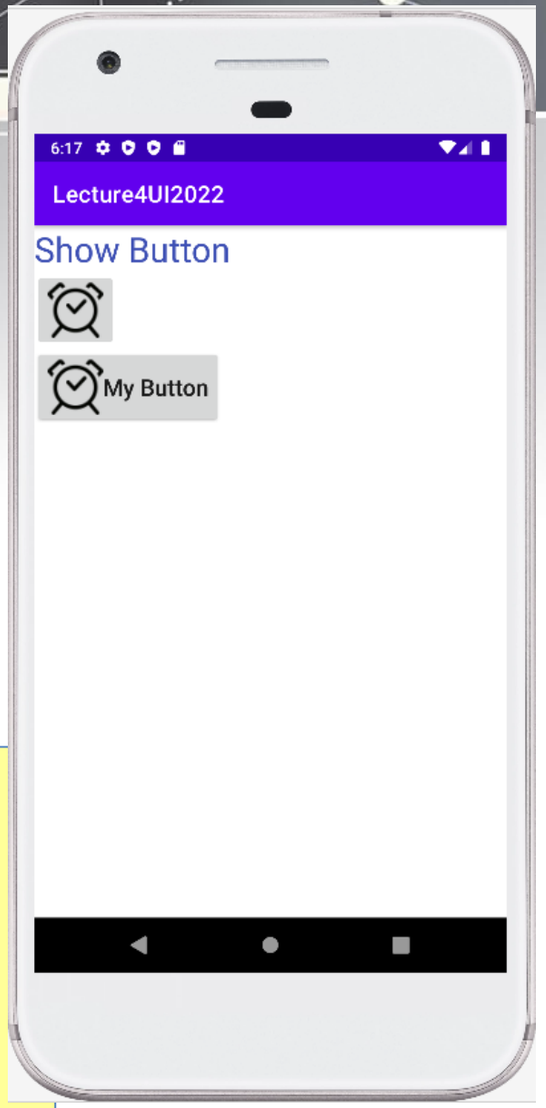


Button : activity_main.xml

- Add the **Android:onClick** attribute to the <Button> node

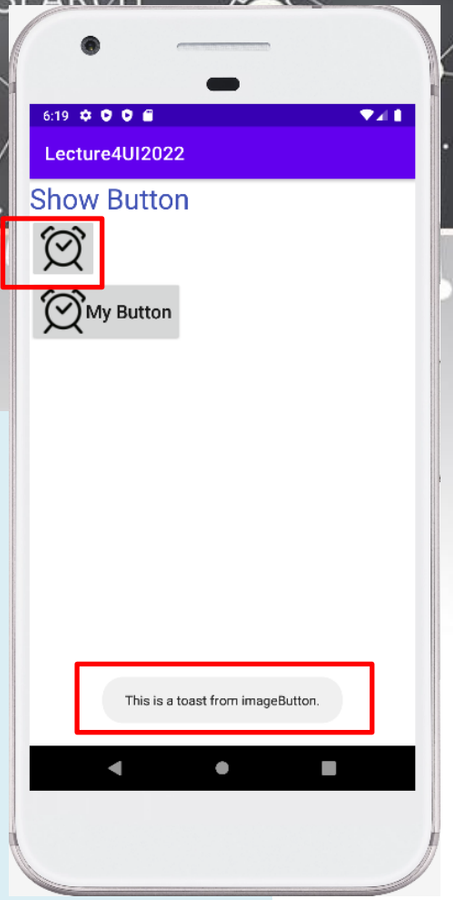
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Show Button"
        android:textSize="30sp"
        android:textColor="#3F51B5"/>
    <ImageButton
        android:id="@+id/imageButton"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:src="@drawable/clook" />
```

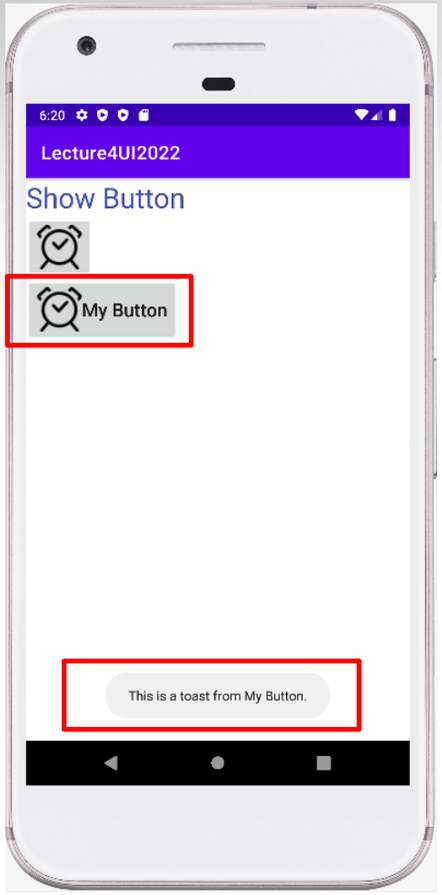
```
<androidx.appcompat.widget.AppCompatButton
        android:id="@+id/myButton"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:textSize="20sp"
        android:text="My Button"
        android:textAllCaps="false"
        android:drawableLeft="@drawable/clook"
        android:onClick="clickButton"  />
</LinearLayout>
```



Button : MainActivity.kt

```
class MainActivity : AppCompatActivity() {
   private lateinit var  binding : ActivityMainBinding
      override fun onCreate(savedInstanceState: Bundle?) {
          super.onCreate(savedInstanceState)

          binding = ActivityMainBinding.inflate(layoutInflater)
          setContentView(binding.root)

          binding.imageButton.setOnClickListener {
           Toast.makeText(this, "This is a toast from imageButton.", Toast.LENGTH_LONG).show()
        }
}

  fun clickButton(v : View){
       Toast.makeText(this, "This is a toast from My Button.", Toast.LENGTH_LONG).show()
    }

}
```





EditText

- An extension of TextView that possesses rich editing capabilities.
- It provide a text field to allow the user to input text.
- It can be either a single line or multiple line.
- Touching an EditText control places the cursor and automatically displays the soft keyboard.
- Users can select, cut, copy, and paste text, input auto-completion, and customize keyboard to suit input type.
- Tag: <EditText> or
  <androidx.appcompat.widget.AppCompatEditText>

EditText

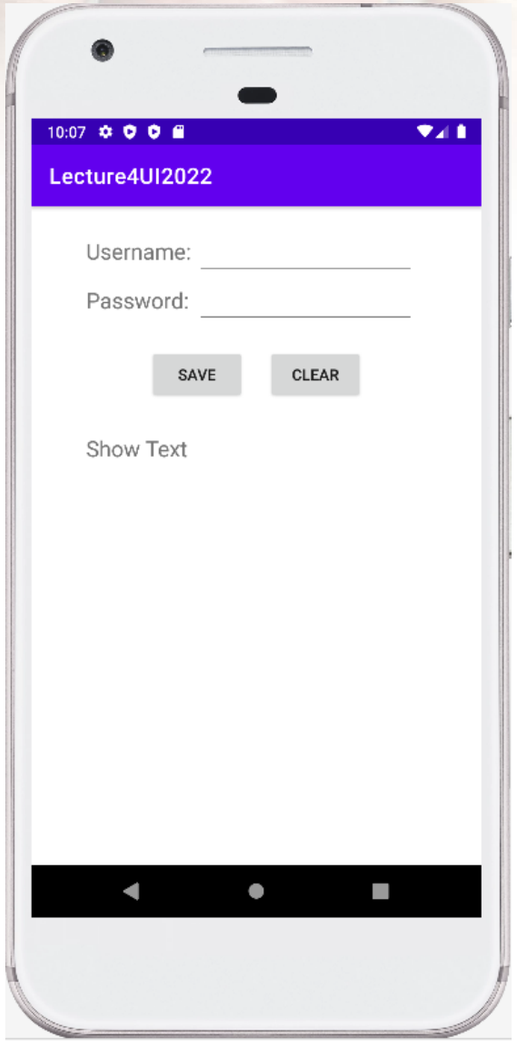
Attribute: android:inputType = ……

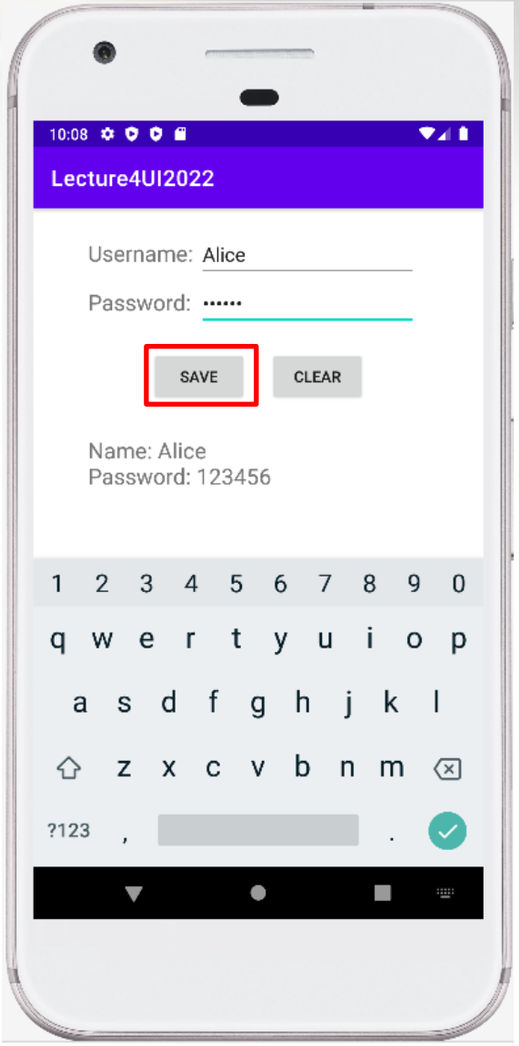
- "text" : the default and calls for a normal text keyboard.
- "textEmailAddress" : calls for normal text keyboard that includes the @ key.
- "number" : calls for a basic number keypad.
- "phone" : calls for a phone dial pad.
- "textPassword" : calls for normal text keyboard, but mask the text entered.

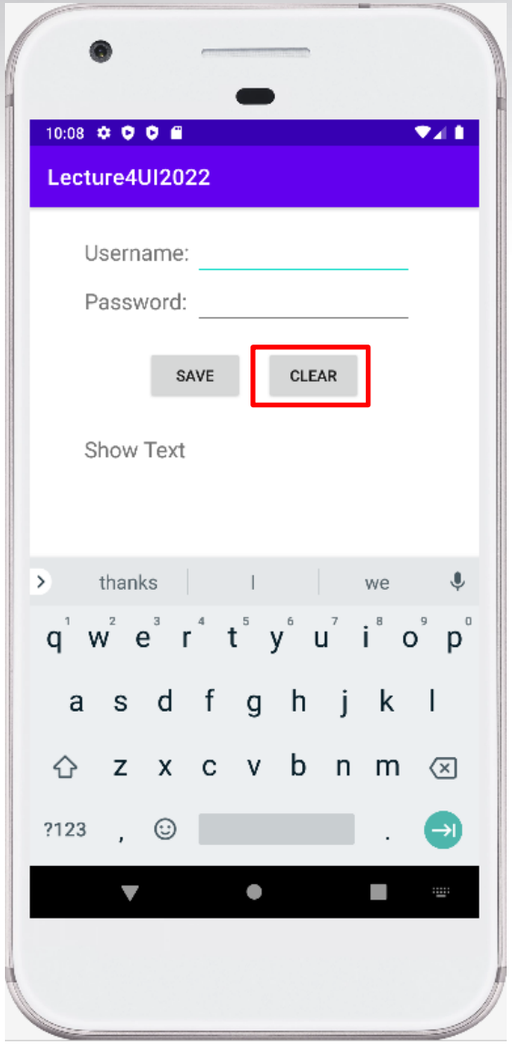
EditText

- "textMultiLine" : calls for normal text keyboard that allows users to enter long strings of text that include line breaks.
- "textCapSentence" : calls for normal text keyboard that capitalizes the first alphabet of each new sentence.
- "textAutoCorrect" : calls for normal text keyboard that provides helping words to correct common spelling errors.

EditText: Example



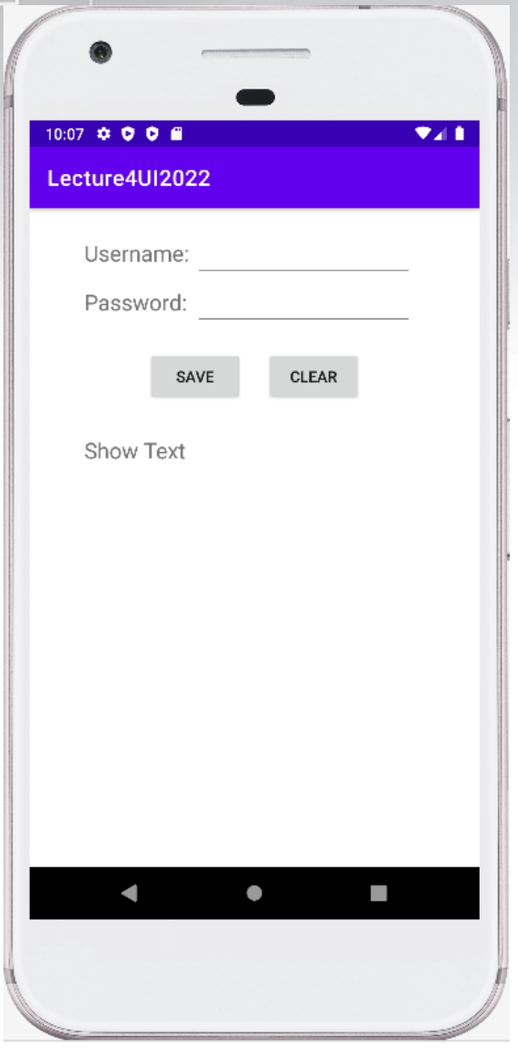




EditText : activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity">
   <TableLayout
      android:id="@+id/table"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginLeft="50dp">
   <TableRow>
      <TextView
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Username: "
         android:textSize="20sp" />
       <EditText
         android:id="@+id/edit_name"
         android:layout_width="200dp"
         android:layout_height="wrap_content"
         android:inputType="textShortMessage"
         android:hint=""/>
   </TableRow>
```

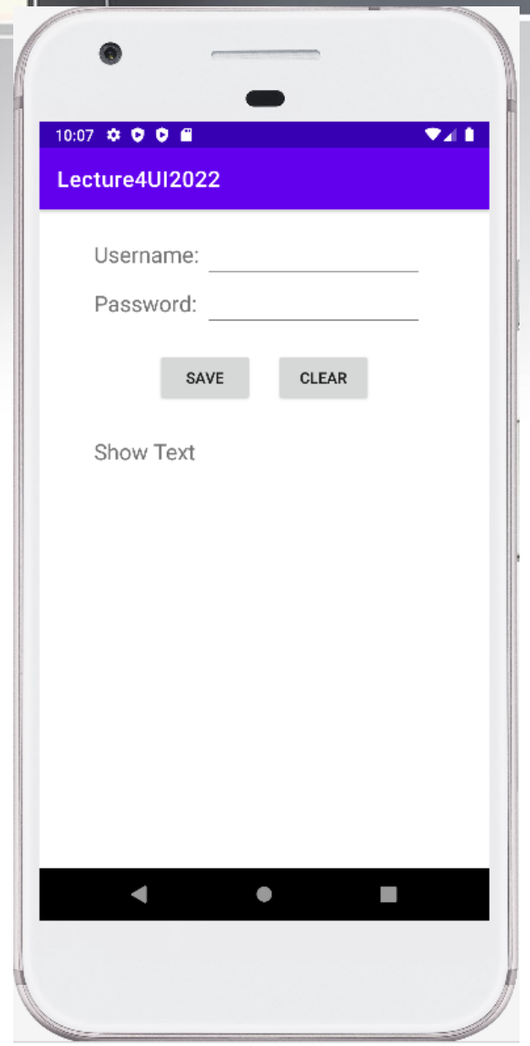
```
<TableRow>
        <TextView
           android:layout_width="wrap_content"
           android:layout_height="wrap_content"
           android:text="Password: "
           android:textSize="20sp" />
        <EditText
           android:id="@+id/edit_password"
           android:layout_width="200dp"
           android:layout_height="wrap_content"
           android:inputType="textPassword"
           android:hint=""/>
     </TableRow>
  </TableLayout>
```



EditText : activity_main.xml

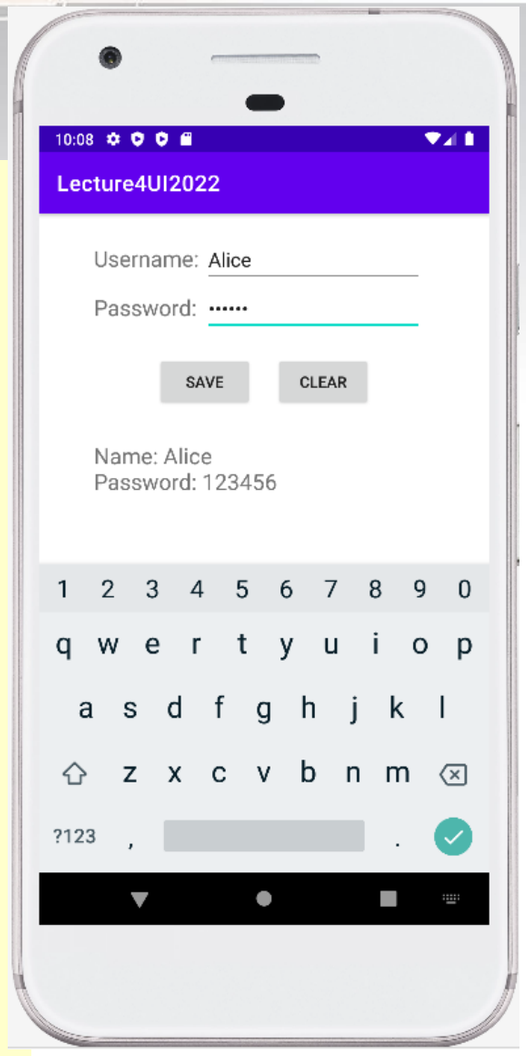
```
<RelativeLayout
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/table"
    android:layout_centerHorizontal="true">
  <androidx.appcompat.widget.AppCompatButton
    android:id="@+id/btn_save"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:onClick="showResult"
    android:text="Save" />
```

```
<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/btn_clear"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="20dp"
    android:layout_toRightOf="@id/btn_save"
    android:onClick="clearResult"
    android:text="Clear" />
  </RelativeLayout>
 <TextView
    android:id="@+id/txt_show"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Show Text"
    android:textSize="20sp"
    android:layout_marginTop="30dp"
    android:layout_marginLeft="50dp"
    android:layout_below="@id/button"/>
</RelativeLayout>
```



EditText : MainActivity.kt

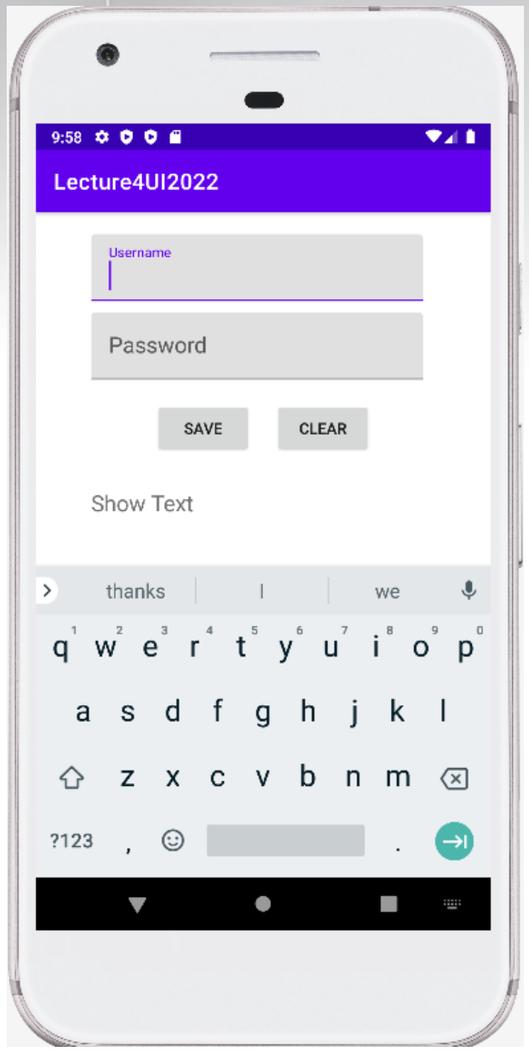
```
class MainActivity : AppCompatActivity() {

    private lateinit var  binding : ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
}

    fun showResult(v : View){
        binding.txtShow.text = "Name: ${binding .editName.text }\n Password: ${binding.editPassword.text}"
    }
    fun clearResult(v: View){
        binding.editName.text?.clear()
        binding.editPassword.text?.clear()
        binding.txtShow.text ="Show Text"
    }
}
```



TextInputLayout

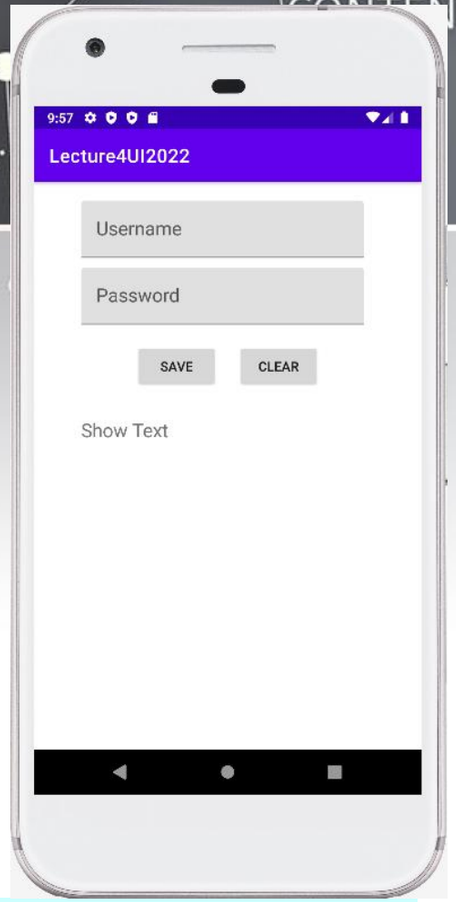
- build.gradle file : implementation ‘com.google.android.material:material:1.2.0’
- Use on API 26+

```
<com.google.android.material.textfield.TextInputLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/form_username">

    <com.google.android.material.textfield.TextInputEditText
            android:layout_width="match_parent"
            android:layout_height="wrap_content“
            android:inputType="textPassword"/>

</com.google.android.material.textfield.TextInputLayout>
```



TextInputLayout

activity_main.xml

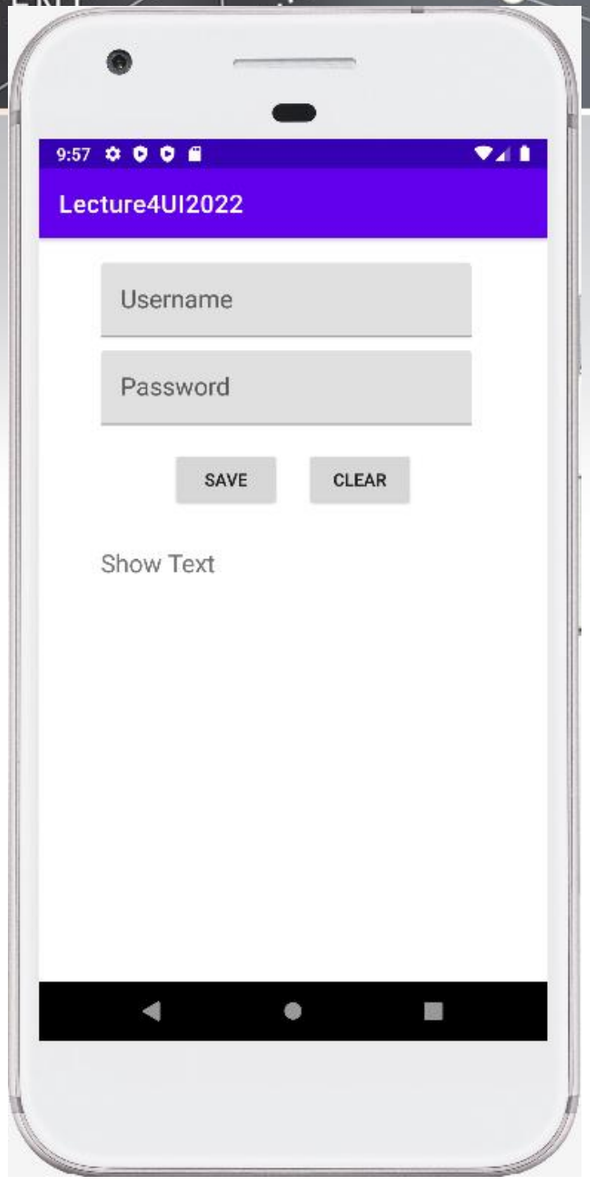



```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <TableLayout
        android:id="@+id/table"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:layout_marginLeft="50dp">
        <TableRow>
            <com.google.android.material.textfield.TextInputLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:hint="User Name">

<com.google.android.material.textfield.TextInputEditText
                    android:id="@+id/edit_name"
                    android:layout_width="300dp"
                    android:layout_height="wrap_content"
                    android:textSize="20sp"
                    android:inputType="textShortMessage"/>
            </com.google.android.material.textfield.TextInputLayout>
        </TableRow>
```

TextInputLayout

activity_main.xml(cont.)

```
<TableRow>
        <com.google.android.material.textfield.TextInputLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="10dp"
            android:hint="Password">
            <com.google.android.material.textfield.TextInputEditText
                android:id="@+id/edit_password"
                android:layout_width="300dp"
                android:layout_height="wrap_content"
                android:textSize="20sp"
                android:inputType="textPassword"/>
        </com.google.android.material.textfield.TextInputLayout>
    </TableRow>
</TableLayout>
```

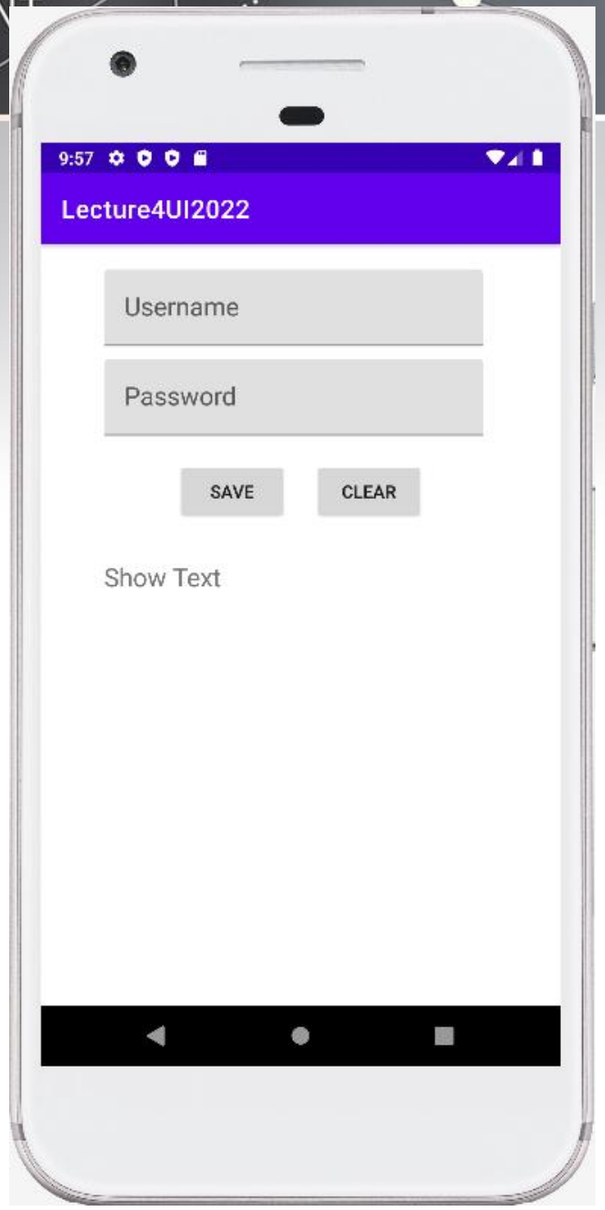


TextInputLayout

activity_main.xml(cont.)

```
<RelativeLayout
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/table"
    android:layout_centerHorizontal="true">
  <androidx.appcompat.widget.AppCompatButton
    android:id="@+id/btn_save"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:onClick="showResult"
    android:text="Save" />
```

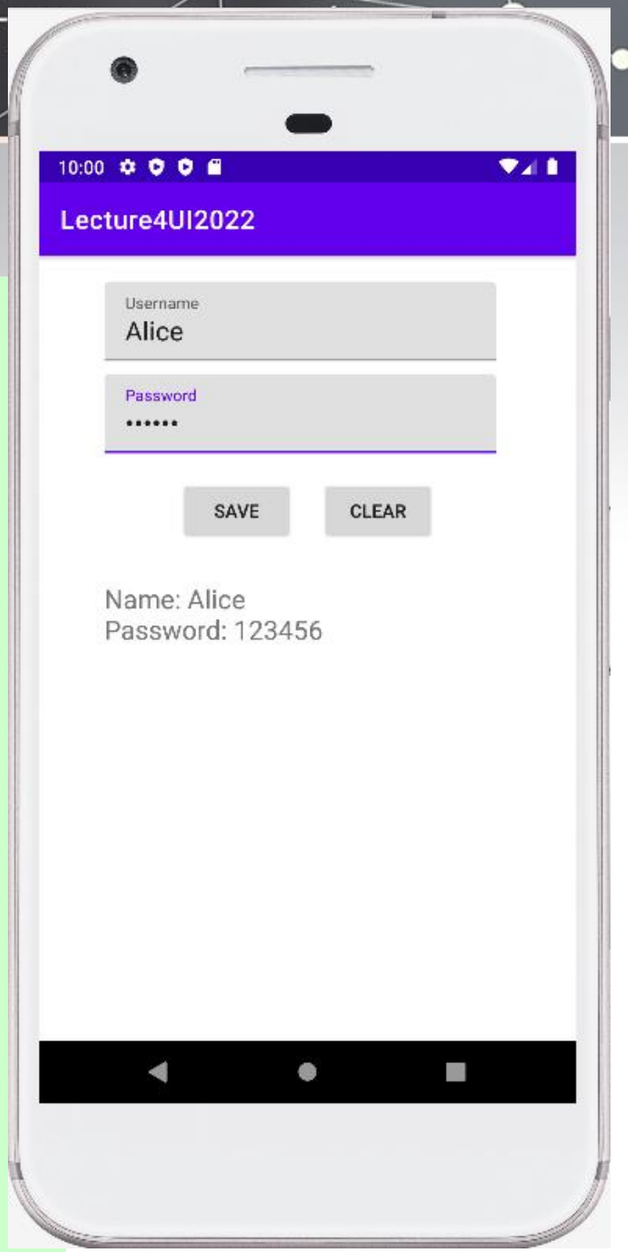
```
<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/btn_clear"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="20dp"
    android:layout_toRightOf="@id/btn_save"
    android:onClick="clearResult"
    android:text="Clear" />
  </RelativeLayout>
<TextView
    android:id="@+id/txt_show"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Show Text"
    android:textSize="20sp"
    android:layout_marginTop="30dp"
    android:layout_marginLeft="50dp"
    android:layout_below="@id/button"/>
</RelativeLayout>
```



TextInputLayout

MainActivity.kt

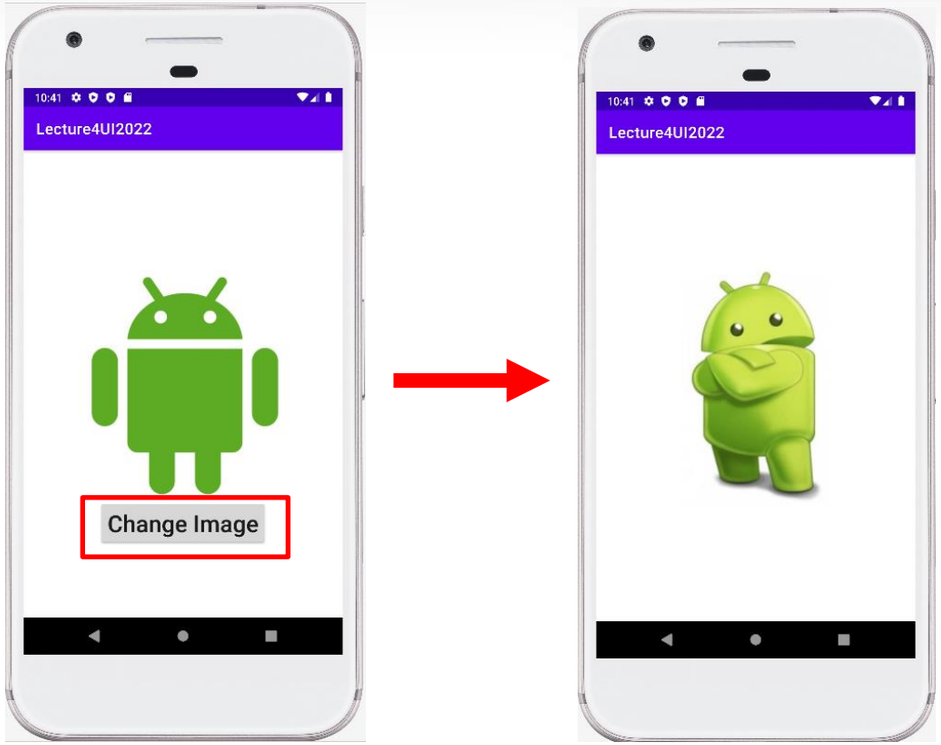
```
class MainActivity : AppCompatActivity() {
    private lateinit var  binding : ActivityMainBinding
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
}
    fun showResult(v : View){
        binding.txtShow.text = "Name: ${binding .editName.text }\n +
                                                Password: ${binding.editPassword.text}\n"
    }
    fun clearResult(v: View){
        binding.editName.text?.clear()
        binding.editPassword.text?.clear()
        binding.txtShow.text ="Show Text"
    }
}
```



ImageView

- To display an image file.
- Tag: <ImageView> or <androidx.appcompat.widget.AppCompatImageView

```
<androidx.appcompat.widget.AppCompatImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/picture_name />
```

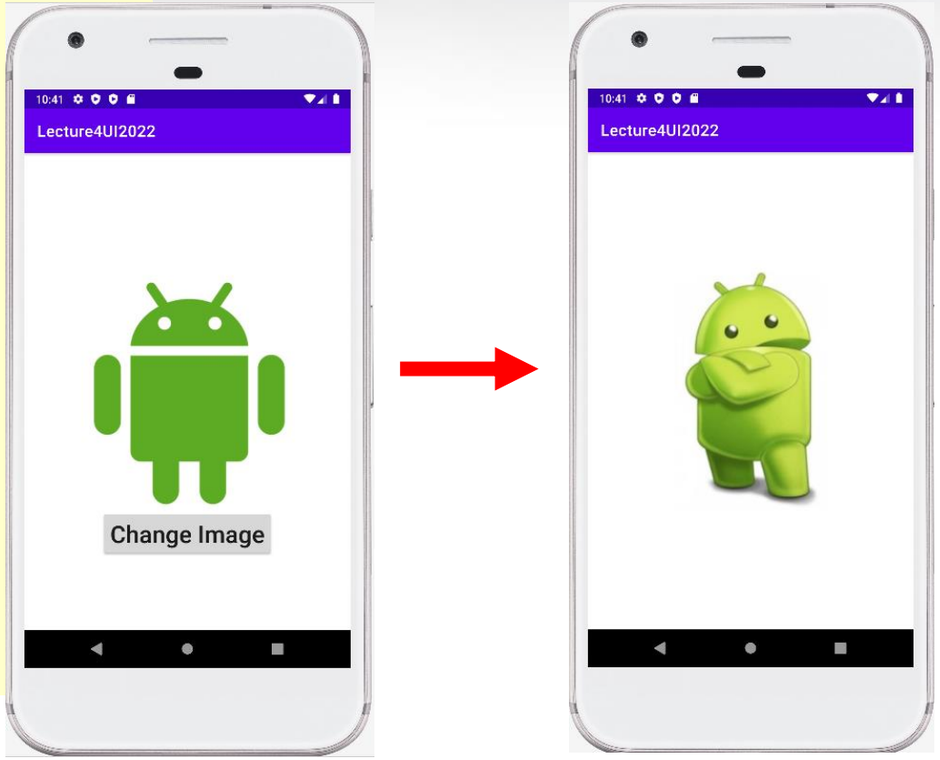


ImageView: activity_main.xml

```
<xml version="1.0" encoding="utf-8" ?>
<RelativeLayout
      xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:tools="http://schemas.android.com/tools"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      tools:context=".MainActivity">
   <androidx.appcompat.widget.AppCompatImageView
         android:id="@+id/imageView1"
         android:layout_width="300dp"
         android:layout_height="300dp"
         android:layout_centerInParent="true"
         android:src="@drawable/logo_old"/>
   <androidx.appcompat.widget.AppCompatButton
         android:id="@+id/btnChangeImage"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:textSize="30sp"
         android:layout_below = "@id/imageView1"
         android:layout_centerHorizontal="true“
         android:textAllCaps="false"
         android:text="Change Image"/>
</RelativeLayout>
```

ImageView : MainActivity.kt

```
class MainActivity : AppCompatActivity() {

 private lateinit var  binding : ActivityMainBinding
     override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

        binding.btnChangeImage.setOnClickListener {
        binding .imageView1.setImageResource(R.drawable.logo_new)
        binding.btnChangeImage.isVisible = false
}
```



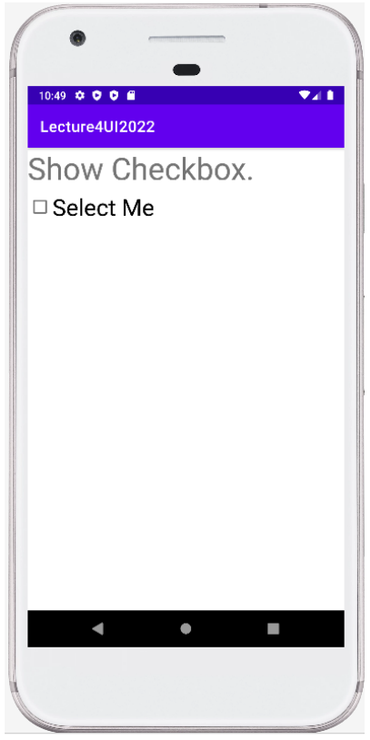
Checkbox

- A CheckBox allows users to select or unselect an option.
- To use a set of checkboxes when we want to allow uses to select multiple options that are mutually exclusive.
- Tag: < CheckBox> or <androidx.appcompat.widget.AppCompatCheckBox>

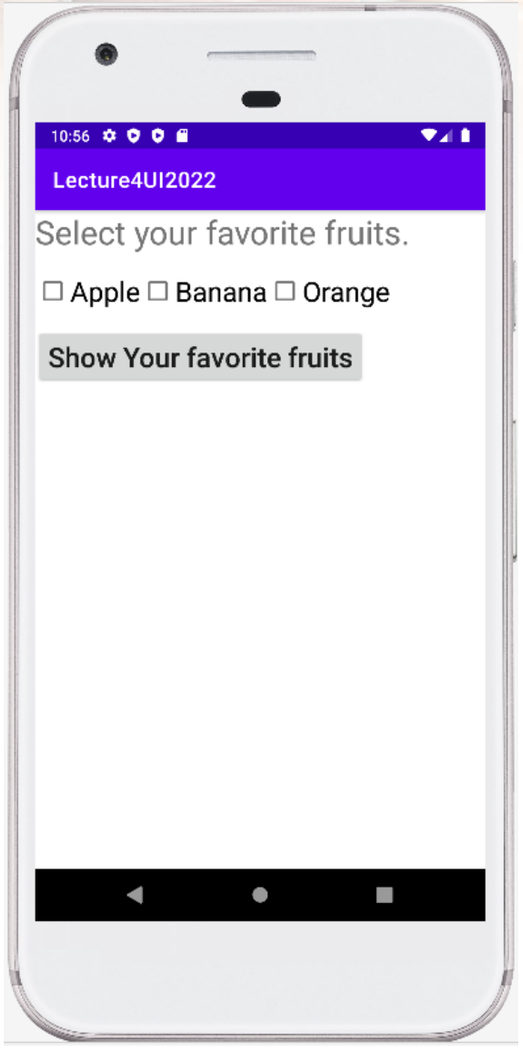
```
<androidx.appcompat.widget.AppCompatCheckBox
        android:id="@+id/checkBox1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Select Me"
        android:textSize="30sp" />
```

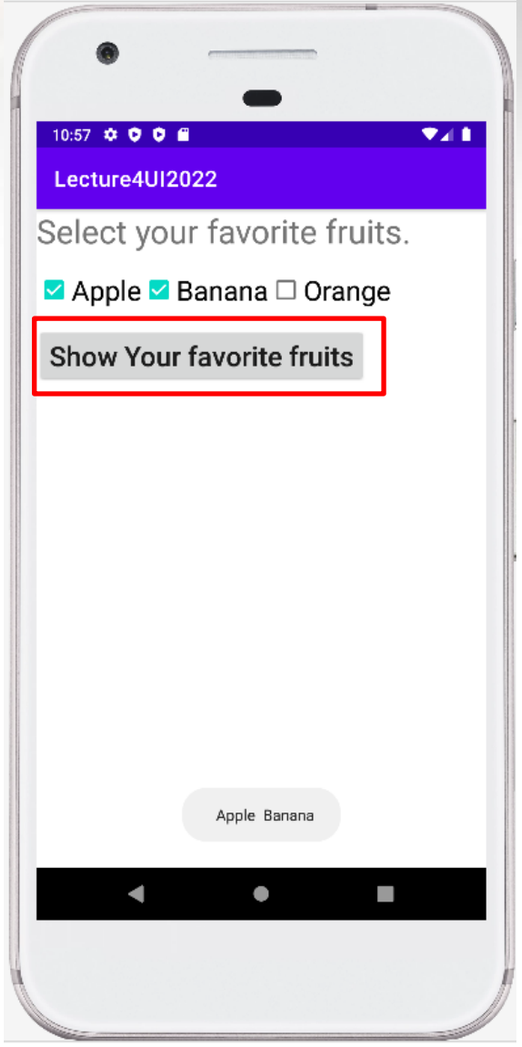
Set Checkbox

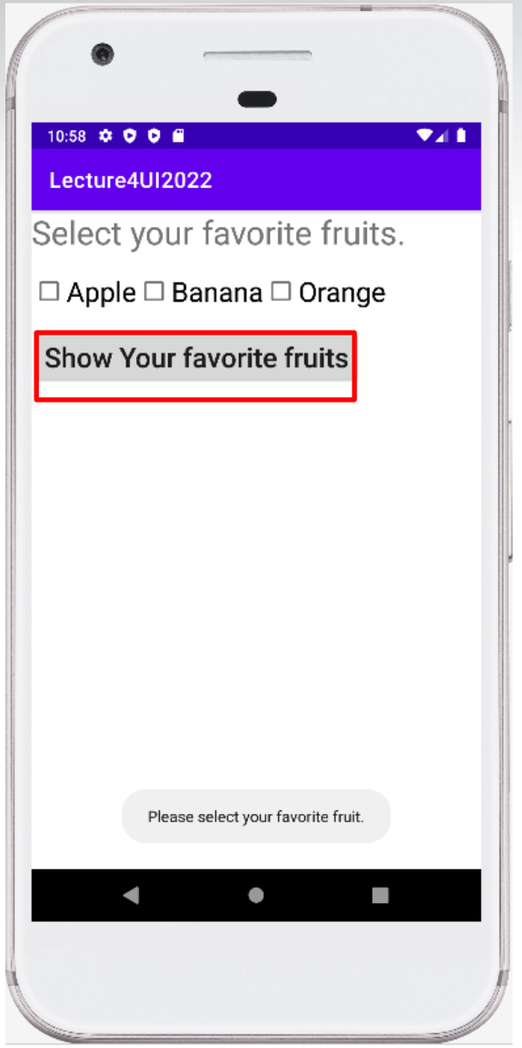
checkBoxName.isChecked = false // uncheck



Check Box : Example

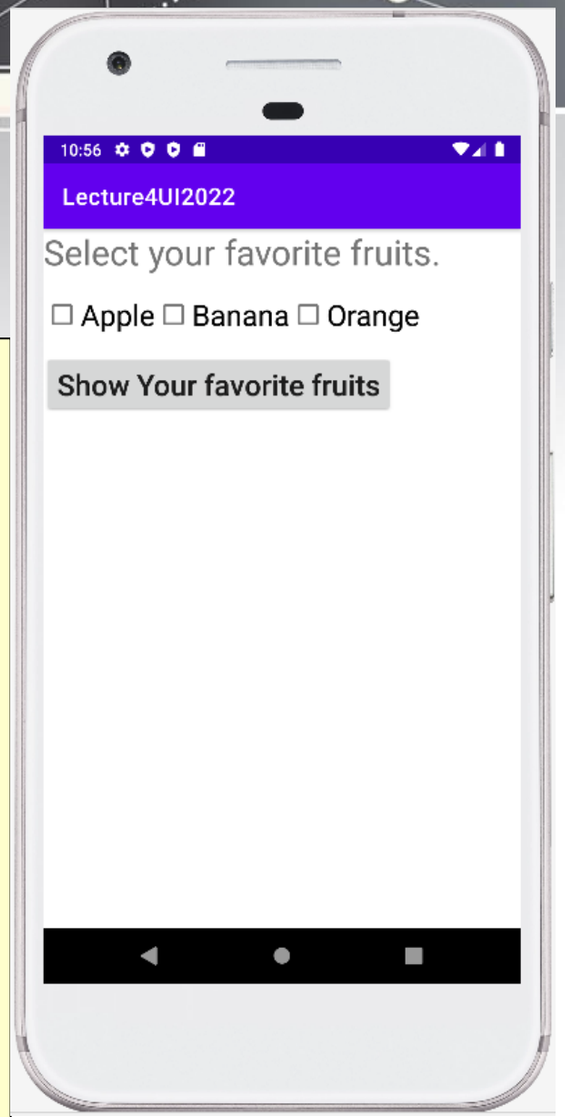






Check Box : activity_main.xml

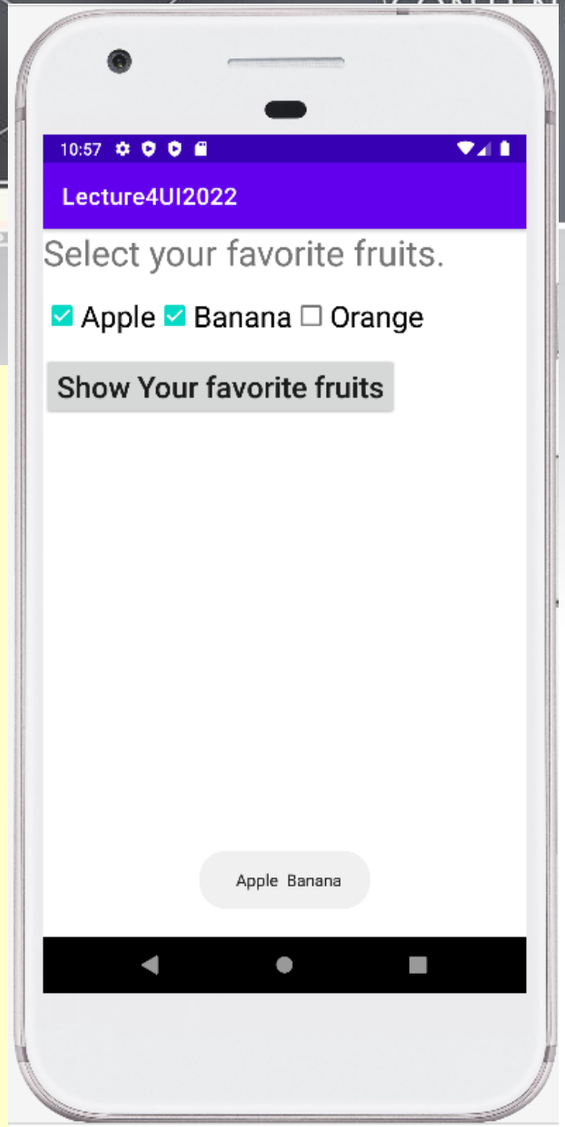
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical“
    ……../>
 <TextView
      android:id="@+id/text1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Select your favorite fruits."
      android:textSize=“30sp"/>
   <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="horizontal"
      android:layout_marginTop="10dp"
      android:layout_marginBottom="10dp">
       <androidx.appcompat.widget.AppCompatCheckBox
         android:id="@+id/apple"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Apple"
         android:textSize="25sp"/>
      <androidx.appcompat.widget.AppCompatCheckBox
         android:id="@+id/banana"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Banana"
         android:textSize="25sp" />
       <androidx.appcompat.widget.AppCompatCheckBox
         android:id="@+id/orange"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Orange"
         android:textSize="25sp"/>
   </LinearLayout>
   <androidx.appcompat.widget.AppCompatButton
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Show Your favorite fruits"
      android:textSize="25sp"
      android:textAllCaps="false"
      android:onClick="showCheckBox"/>
</LinearLayout>
```



Check Box : MainActivity.kt

```
class MainActivity : AppCompatActivity() {
    private lateinit var  binding : ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
}
fun showCheckBox(view:View ){
         var  str :String = ""
        if (binding.apple.isChecked){
            str += "  " + binding.apple.text;
        }
        if (binding.banana.isChecked){
            str +="  " + binding.banana.text;
        }
        if (binding.orange.isChecked){
            str +="  " + binding.orange.text;
        }
```

```
var str_new = if (str.isNotEmpty()) str else "Please select your favorite fruit."

var  toast :Toast = Toast.makeText ( this, str_new, Toast.LENGTH_LONG )
toast.show ( )
   }
```



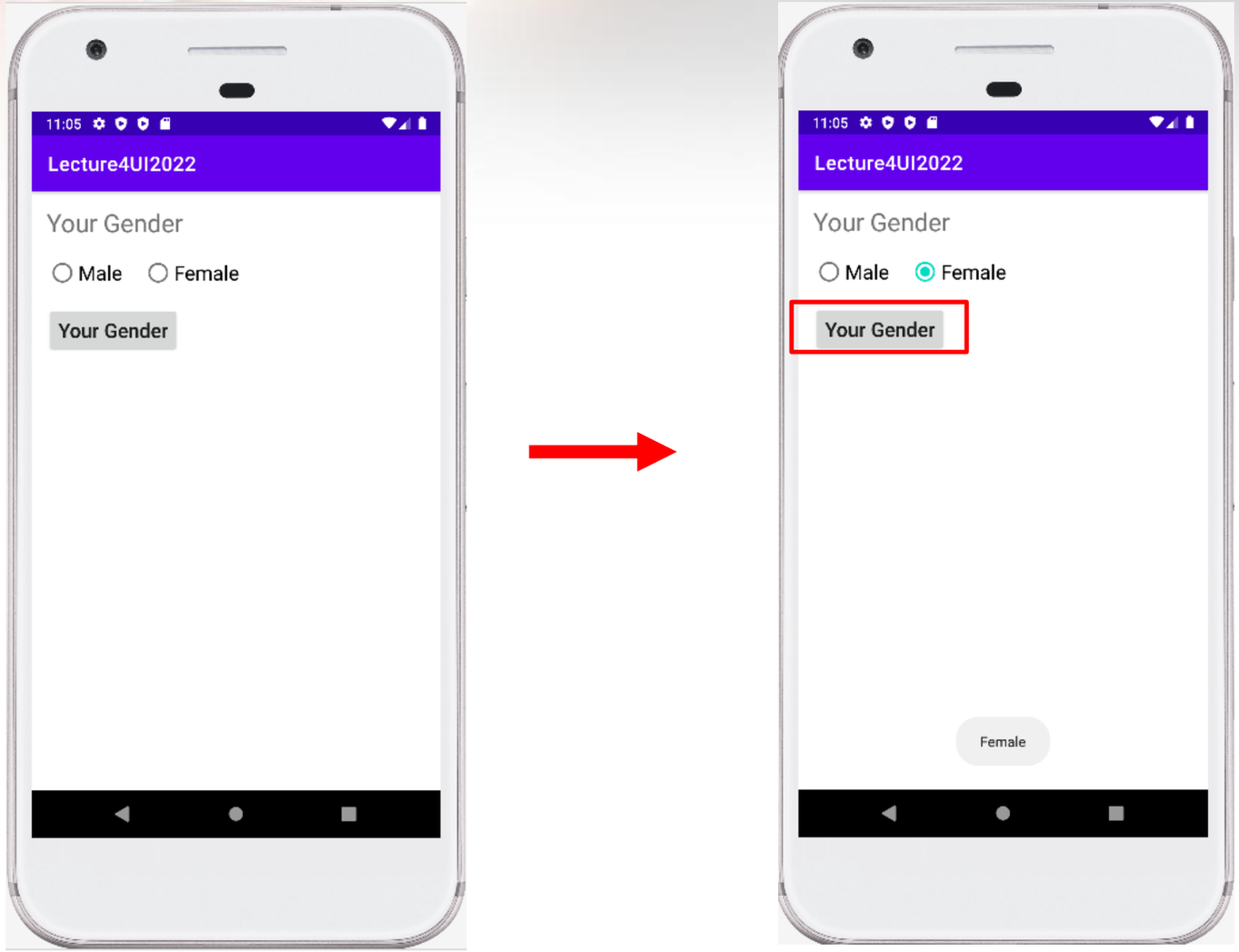
RadioButton

- A RadioButton offers users an option to choose.
- It belongs to a group of radio buttons whereby only one of them can be selected at any one time, such as the selection of gender.
- How to ensure that, the answer is "grouping them together in a RadioGroup".

Tag: <RadioButton> or
<androidx.appcompat.widget.AppCompatRadioButton>

```
<RadioGroup
        android:id="@+id/radio"
        android:orientation="horizontal“
        …… >
    <RadioButton
        android:id="@+id/radioButton1"
        android:text="RadioButton1“
        ……./>
    <RadioButton
        android:id="@+id/radioButton2"
        android:text=" RadioButton2“
        …….. />
</RadioGroup>
```

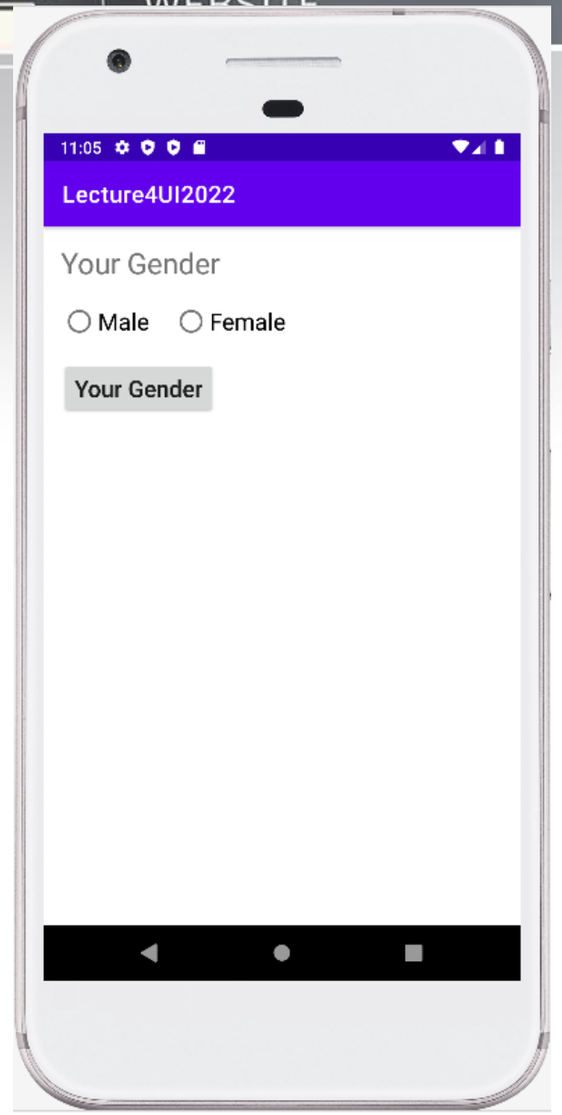
RadioButton: Example



RadioButton : activity_main.xml

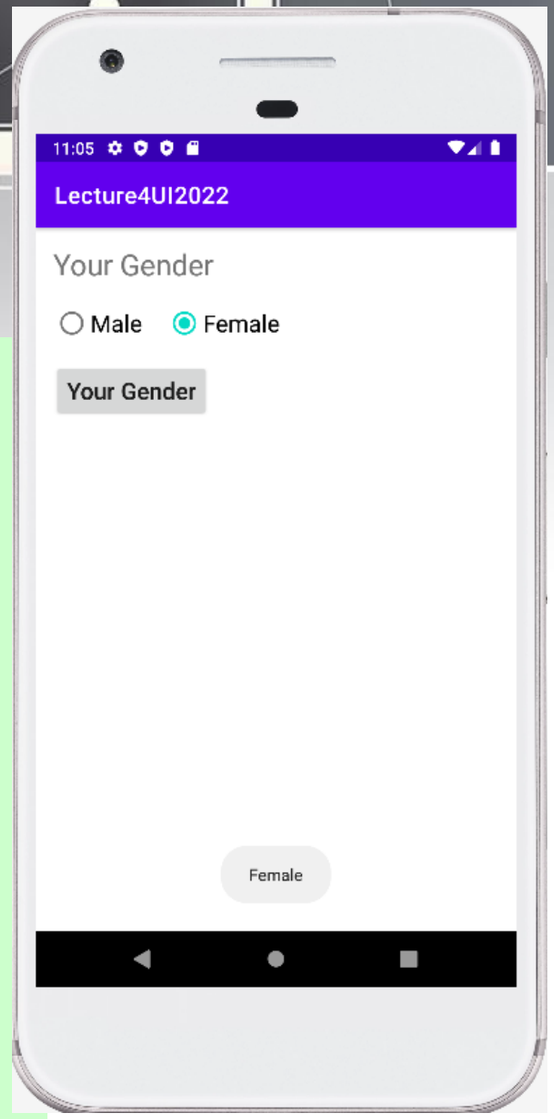
```
<?xml version="1.0"  encoding="utf-8"?>
<LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="match_parent“
        android:padding="15dp"
        android:orientation="vertical">
  <TextView
       android:id="@+id/text1"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:text="Your Gender"
       android:textSize="25sp"/>
  <RadioGroup
       android:id="@+id/radioGroup"
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:orientation="horizontal"
       android:layout_marginTop="10dp">
    <androidx.appcompat.widget.AppCompatRadioButton
         android:id="@+id/male"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Male"
         android:textSize="20sp"
         android:layout_marginRight="20dp"/>
```

```
    <androidx.appcompat.widget.AppCompatRadioButton
         android:id="@+id/female"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Female"
         android:textSize="20sp"
         android:layout_marginRight="20dp" />
  </RadioGroup>
    <androidx.appcompat.widget.AppCompatButton
       android:id="@+id/buttonClick"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:text="Your Gender"
       android:textSize="20sp"
       android:textAllCaps="false"
       android:layout_marginTop="10dp" />
</LinearLayout>
```



RadioButton : MainActivity.kt

```
class MainActivity : AppCompatActivity() {
    private lateinit var  binding : ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
       super.onCreate(savedInstanceState)
       binding = ActivityMainBinding.inflate(layoutInflater)
       setContentView(binding.root)

       addListenerOnButton()
}

 fun addListenerOnButton() {
       binding.buttonClick.setOnClickListener {
          var selectID: Int = binding.radioGroup.checkedRadioButtonId
          var radioButtonChecked: RadioButton = findViewById(selectID)
          var toast: Toast = Toast.makeText(this, radioButtonChecked.text, Toast.LENGTH_LONG)
          toast.show()
       }
}
```



Spinner

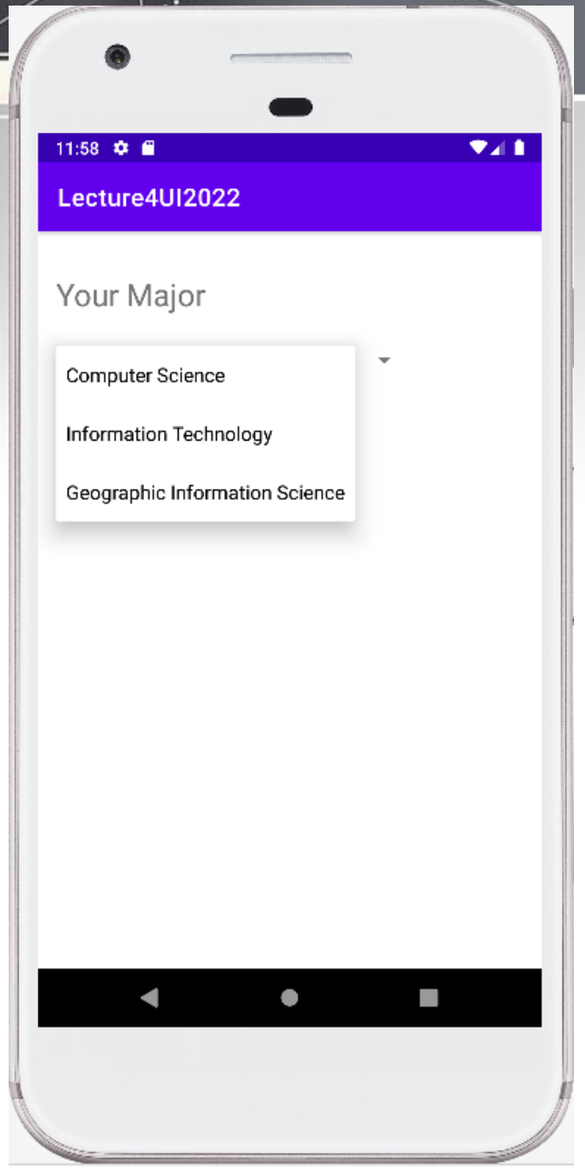
- A **Spinner** is just a fanciful name for the all-too-familiar dropdown list.
- Tag: <Spinner> or <androidx.appcompat.widget.AppCompatSpinner>

```
<androidx.appcompat.widget.AppCompatSpinner
    android:id="@+id/dayOfWeek"
    android:layout_width=" wrap_content"
    android:layout_height="wrap_content" />
```

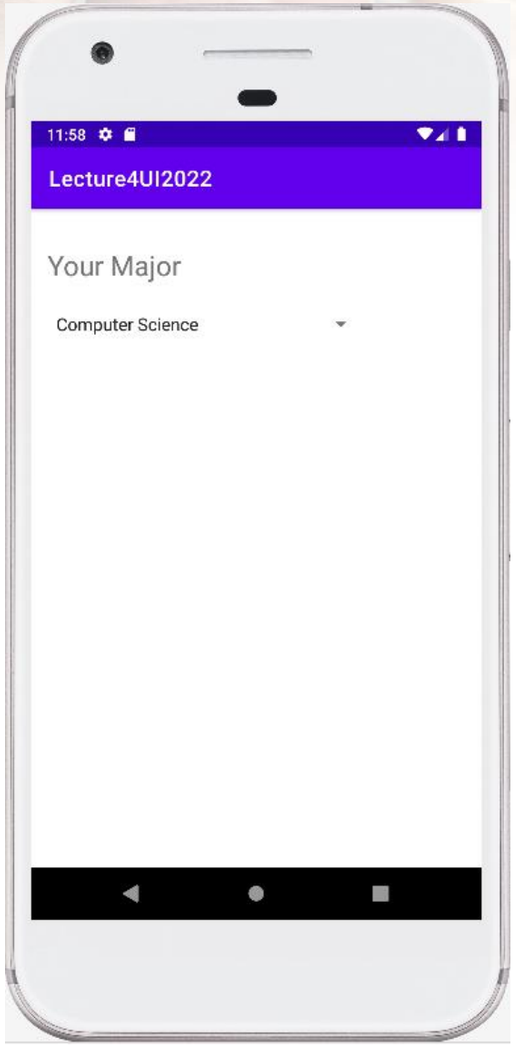
Spinner

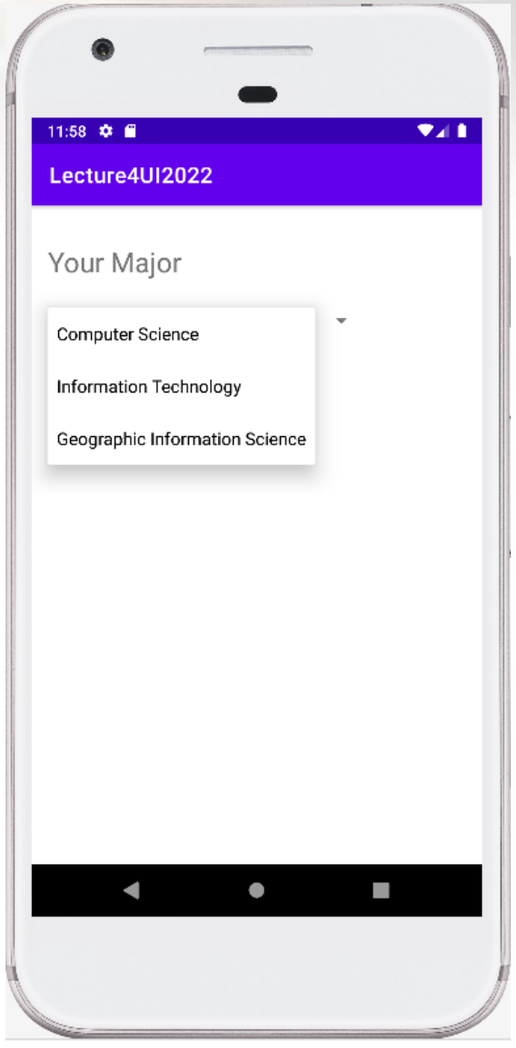
strings.xml

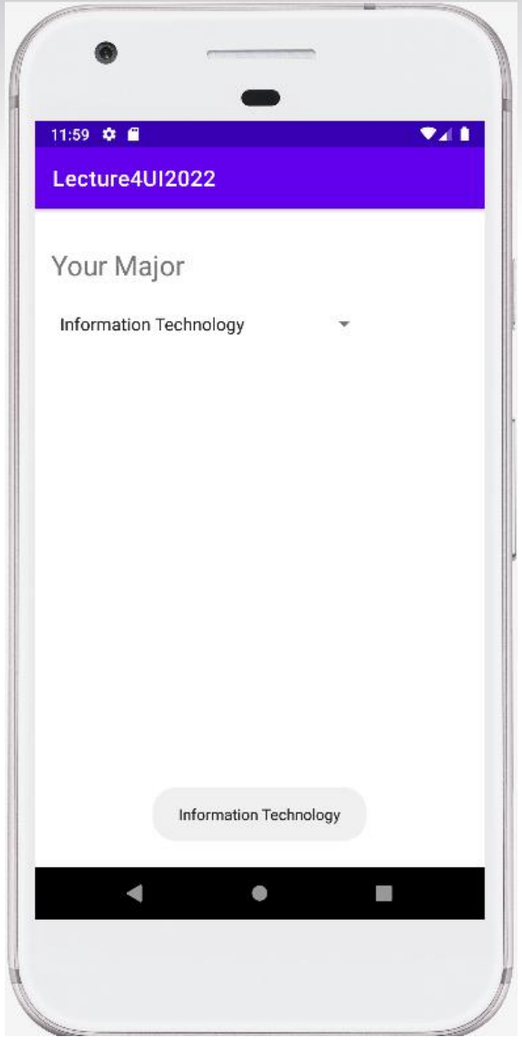
```
<resources>
    <string name="app_name">Spinner Test</string>

    <string-array name=“majorName_array">
        <item>Computer Science</item>
        <item>Information Technology</item>
        <item>Geographic Information Science</item>
    </string-array>

</resources>
```



Spinner: Example

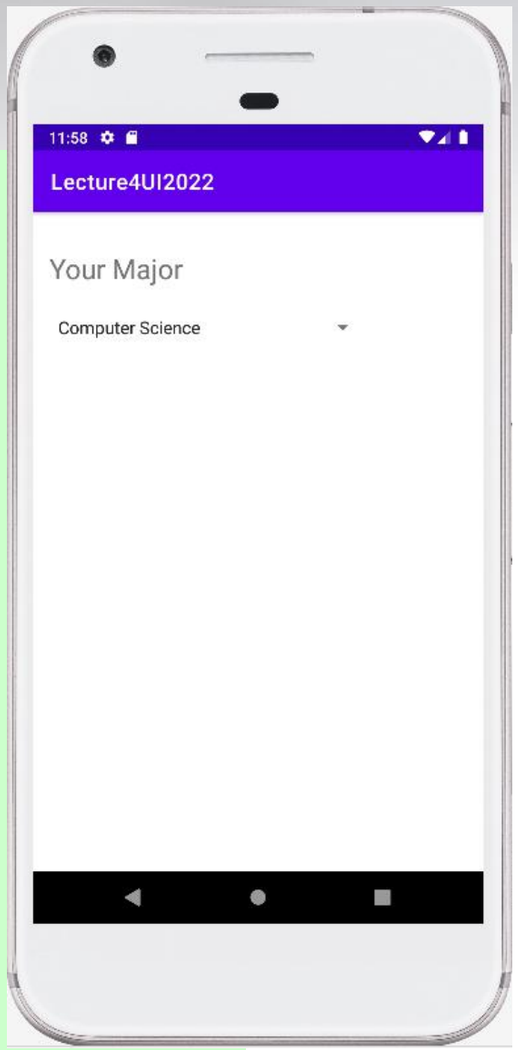






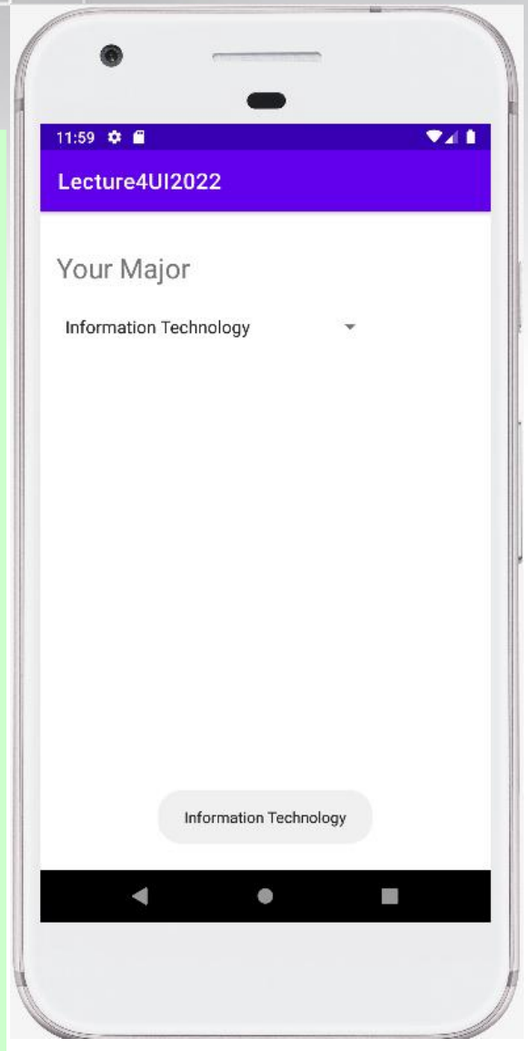
Spinner: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
<TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Your Major"
      android:textSize="25sp"
      android:layout_marginTop="20dp"/>
   <androidx.appcompat.widget.AppCompatSpinner
      android:id="@+id/major"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginTop="25dp"/>
</LinearLayout>
```



Spinner: MainActivity.kt

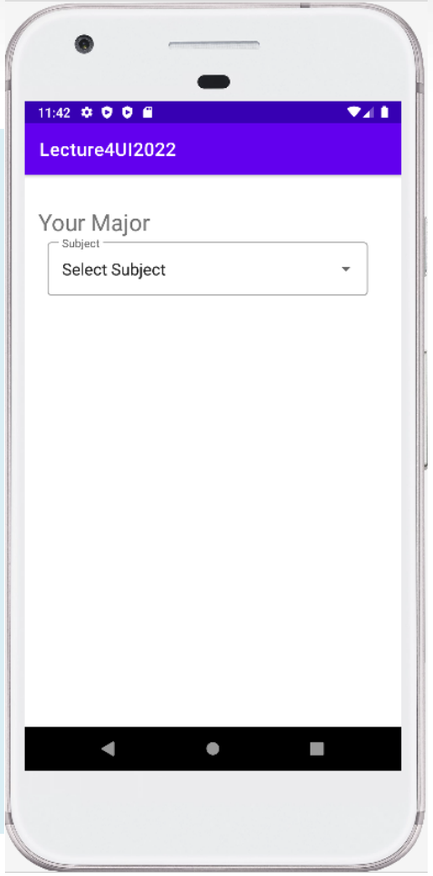
```
class MainActivity : AppCompatActivity() {
    private lateinit var  binding : ActivityMainBinding
    override fun onCreate(savedInstanceState: Bundle?) {
      super.onCreate(savedInstanceState)
      binding = ActivityMainBinding.inflate(layoutInflater)
      setContentView(binding.root)
      // Set spinner
      val majorArray = resources.getStringArray(R.array.majorName_array)
      val arrayAdapter = ArrayAdapter(this, android.R.layout.simple_spinner_item, majorArray)
      arrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)

       binding.major.adapter = arrayAdapter

      binding.major.onItemSelectedListener = object : AdapterView.OnItemSelectedListener {

         override fun onItemSelected(parent: AdapterView<*>, view: View, position: Int, id: Long) {
            // Display the selected item text on Toast
            Toast.makeText(this@MainActivity, majorArray[position], Toast.LENGTH_LONG).show()
          }
         override fun onNothingSelected(parent: AdapterView<*>) { // Another interface callback }
         }
      }}
```



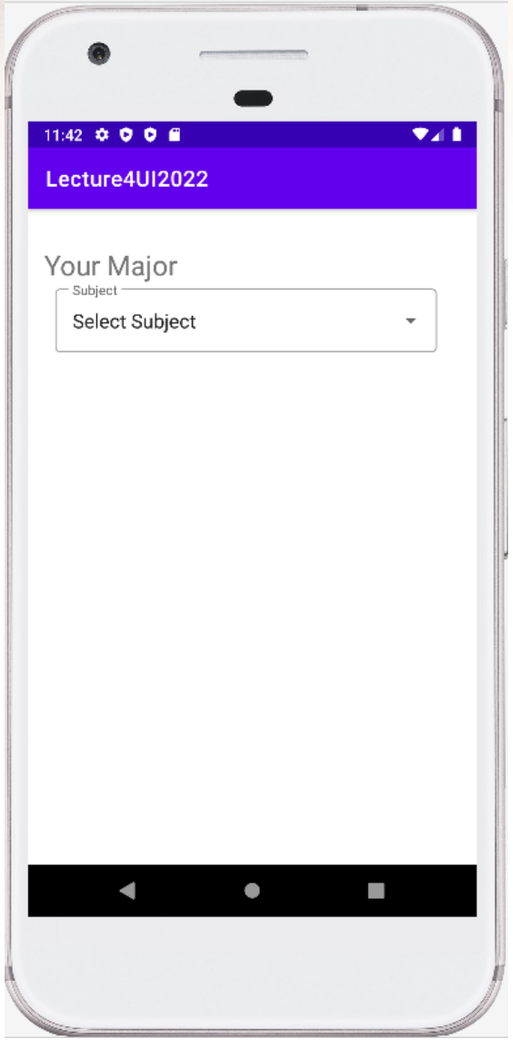
Dropdown by TextInputLayout

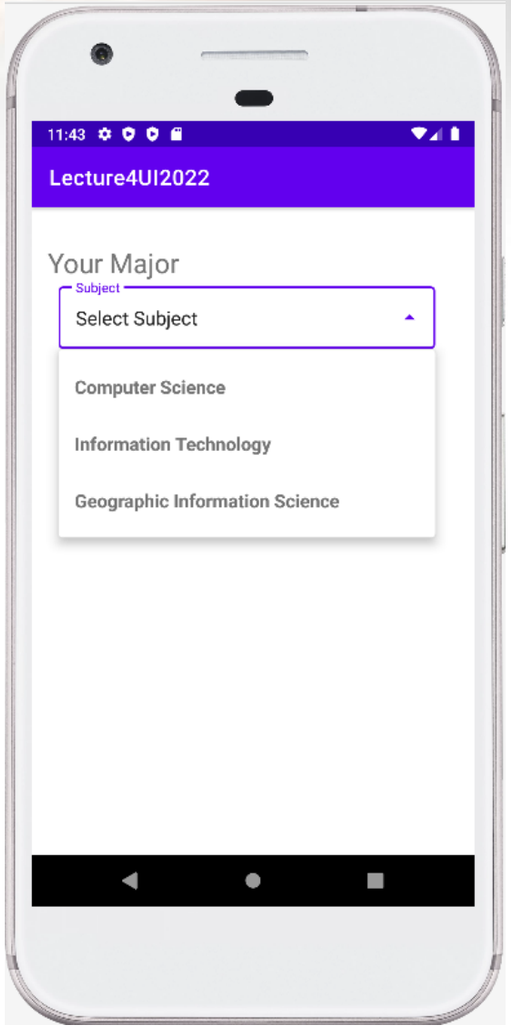
- TextInputLayout and AutoCompleteTextView are applied to create dropdown list.

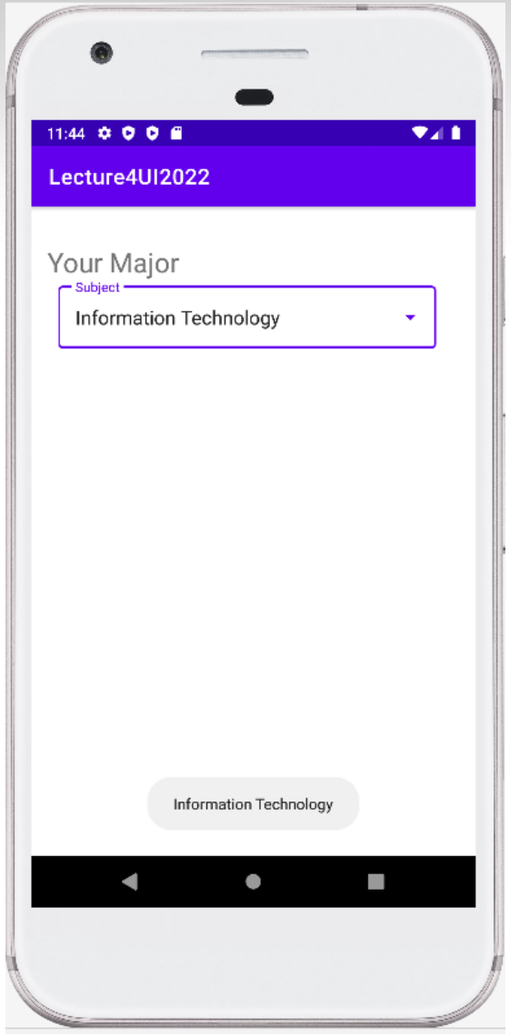
```
<com.google.android.material.textfield.TextInputLayout
      android:layout_width="350dp"
      android:layout_height="wrap_content"
     style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox.ExposedDropdownMenu"
      android:hint="….."
<AutoCompleteTextView
         android:id="@+id/…"
         android:layout_width="match_parent"
         android:layout_height="match_parent"
         android:layout_weight="1"
         android:inputType="none"
         android:text="Select Subject"/>
  </com.google.android.material.textfield.TextInputLayout>
```



Example

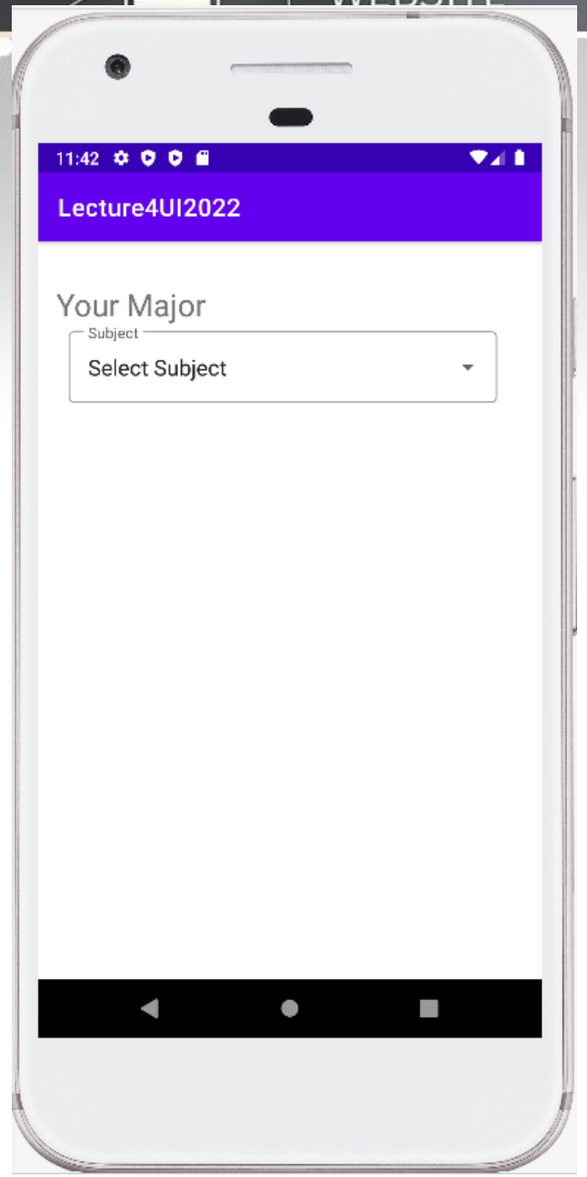

11:42
Lecture4UI2022
Your Major
Subject
Select Subject


11:43
Lecture4UI2022
Your Major
Subject
Select Subject
Computer Science
Information Technology
Geographic Information Science


11:44
Lecture4UI2022
Your Major
Subject
Information Technology
Information Technology

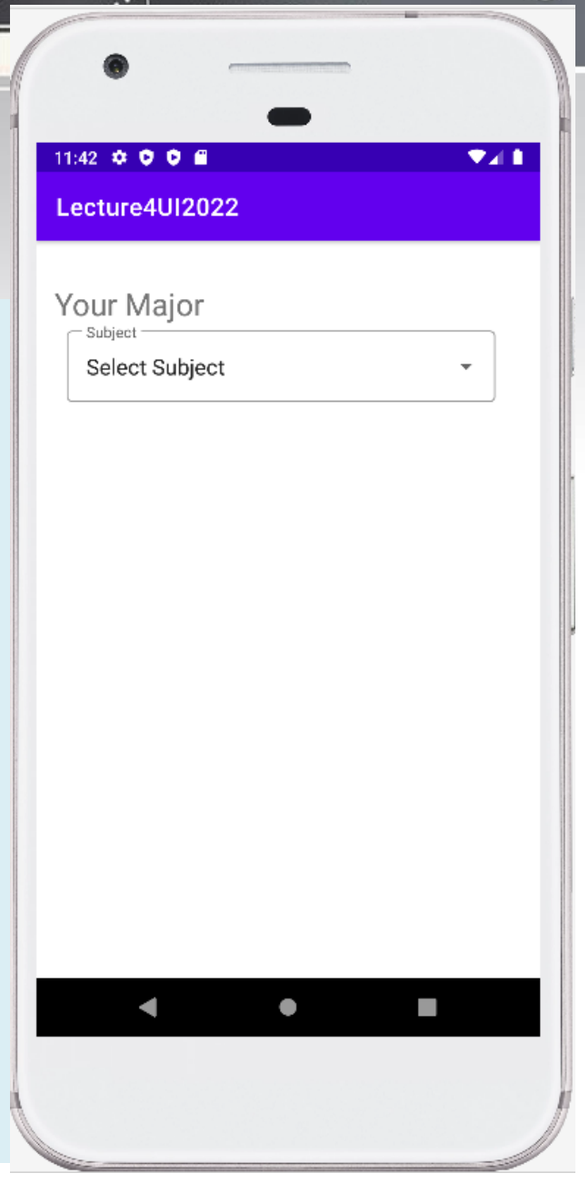
activity_main.xml

```
?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="15dp"
    tools:context=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Your Major"
        android:textSize="25sp"
        android:layout_marginTop="20dp"/>
```



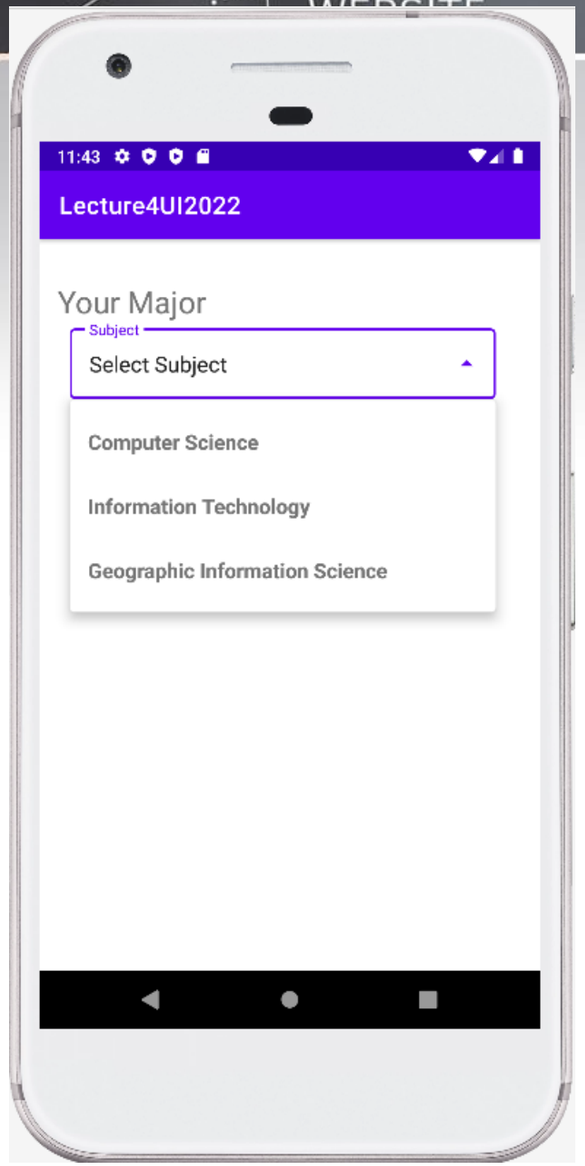
activity_main.xml(cont.)

```
<com.google.android.material.textfield.TextInputLayout
      android:layout_width="350dp"
      android:layout_height="wrap_content"
   style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox.ExposedDropdownMenu"
      android:hint="Subject"
      android:layout_marginLeft="10dp"
      android:layout_marginBottom="10dp">
      <AutoCompleteTextView
         android:id="@+id/major"
         android:layout_width="match_parent"
         android:layout_height="match_parent"
         android:inputType="none"
         android:textSize="18sp"
         android:text="Select Subject"/>
   </com.google.android.material.textfield.TextInputLayout>
</LinearLayout>
```



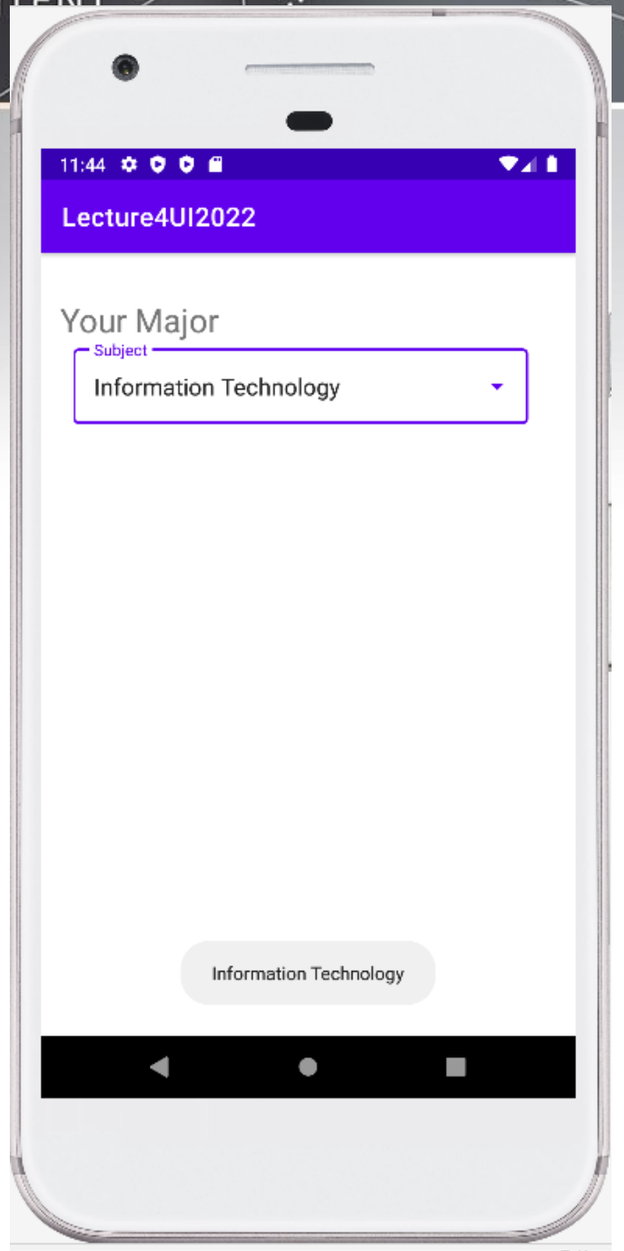
dropdown_item.xml

```
<TextView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/textView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="15dp"
    android:textSize="17sp"
    android:textStyle="bold"
    android:text="TextView" />
```



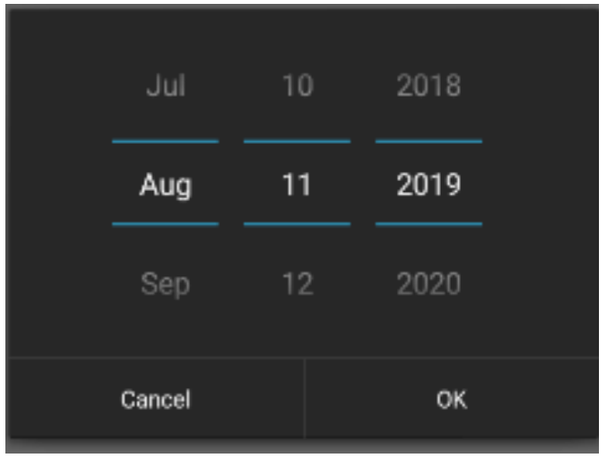
MainActivity.kt

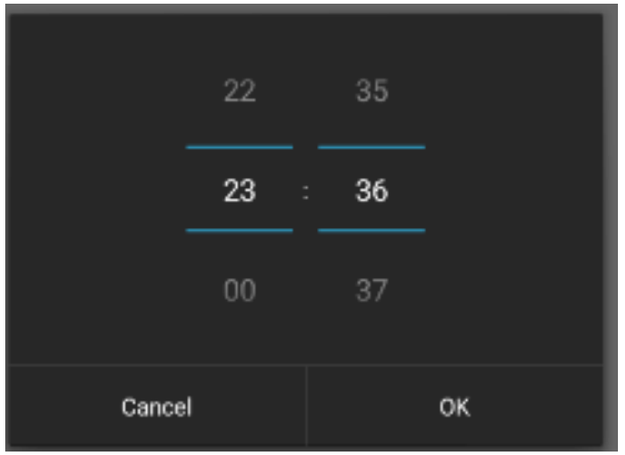
```
class MainActivity : AppCompatActivity() {
    private lateinit var  binding : ActivityMainBinding
    override fun onCreate(savedInstanceState: Bundle?) {
      super.onCreate(savedInstanceState)
      binding = ActivityMainBinding.inflate(layoutInflater)
      setContentView(binding.root)

     //Set Dropdown
      val major_array = resources.getStringArray(R.array.majorName_array)
      val arrayAdapter = ArrayAdapter(this, R.layout.dropdown_item, major_array )
      binding.major.setAdapter(arrayAdapter)

      binding.major.setOnItemClickListener { parent, _, position, _ ->
         val major = parent.getItemAtPosition(position) as String
         //Show Toast
         Toast.makeText(applicationContext,major,Toast.LENGTH_LONG).show()
      }
}
}
```



Pickers

- Android provides controls for the user to pick a time or a date as ready-to-use dialogs.
- Each picker provides controls for selecting each part of the time (hour, minute, AM/PM) or date (month, day, year).
- Using these pickers helps ensure that users can pick a time or date that is valid, formatted correctly, and adjusted to the user's locale
- **DatePicker** and **TimePicker**

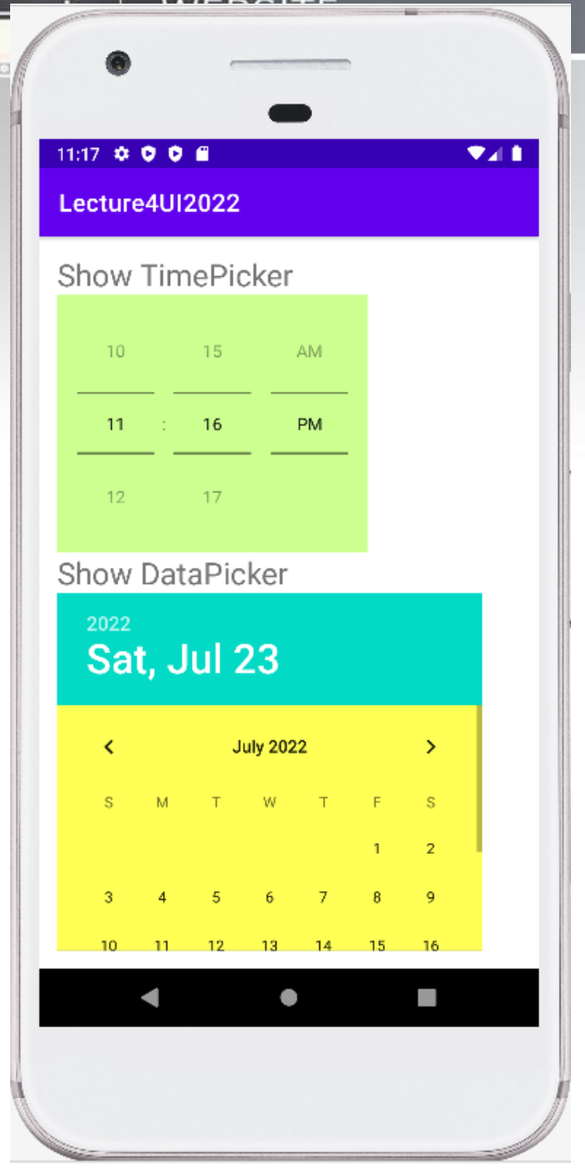




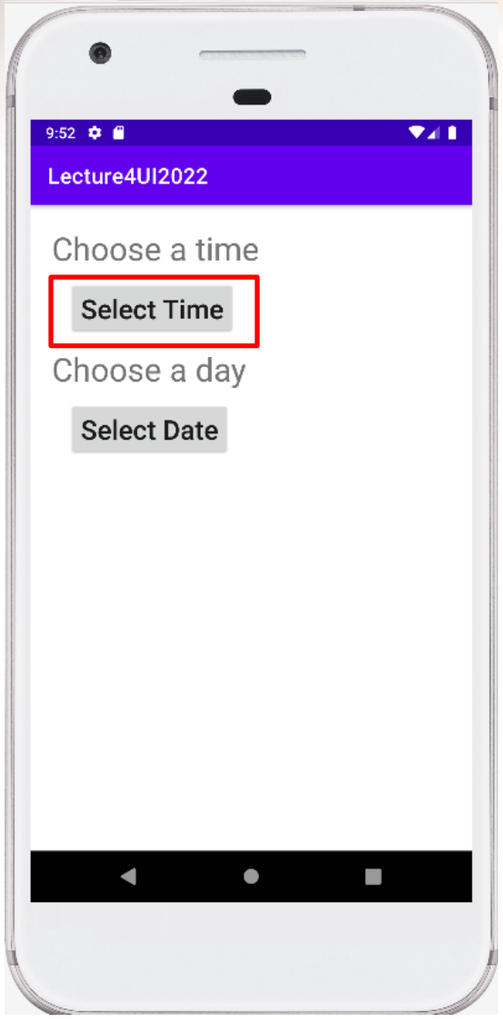
TimePicker and DatePicker Tag

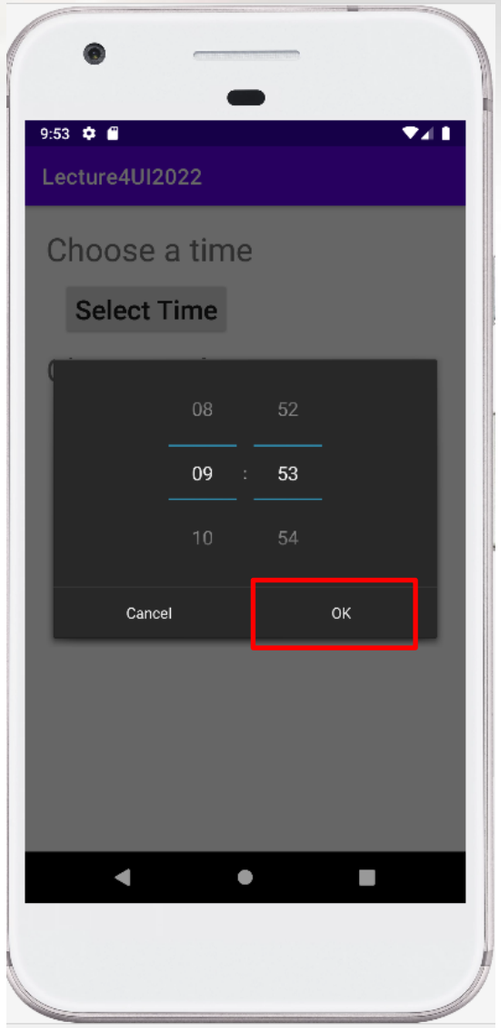
```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Show TimePicker"
        android:textSize="25sp"/>
    <TimePicker
        android:id="@+id/simpleTimePicker"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:timePickerMode="spinner"
        android:background="#CCFF90"/>
```

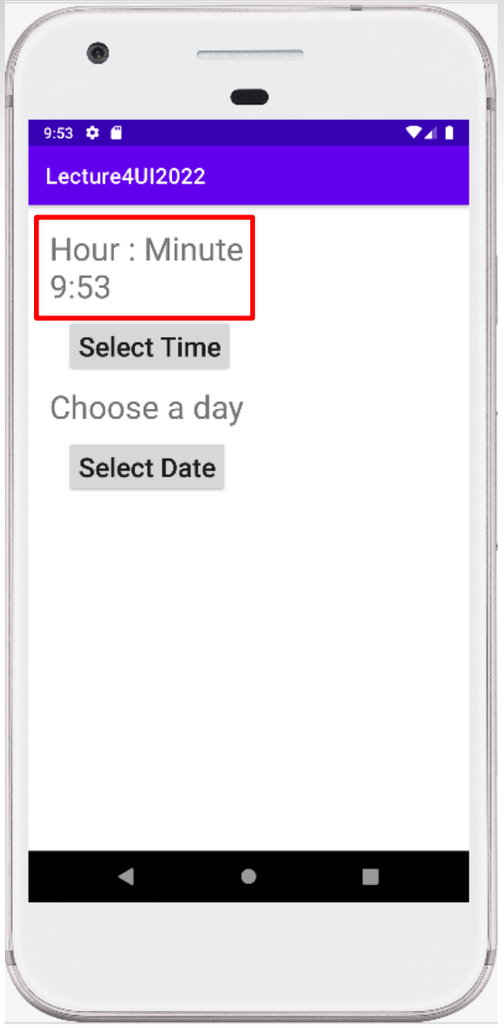
```
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Show DataPicker"
        android:textSize="25sp"/>
    <DatePicker
        android:id="@+id/simpleDatePicker"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content“
        android:datePickerMode="calendar"
        android:background="#fff5"/>
</LinearLayout>
```



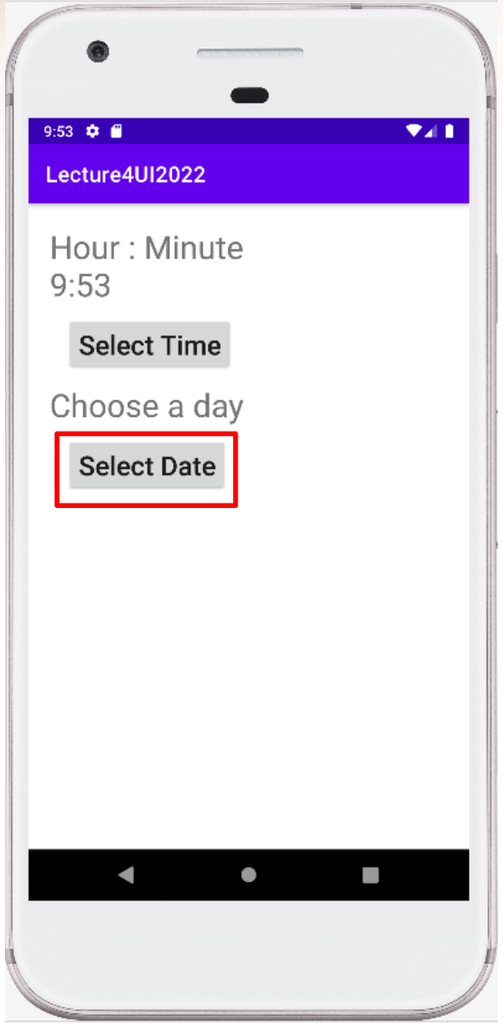
Ex. TimePicker and DatePicker

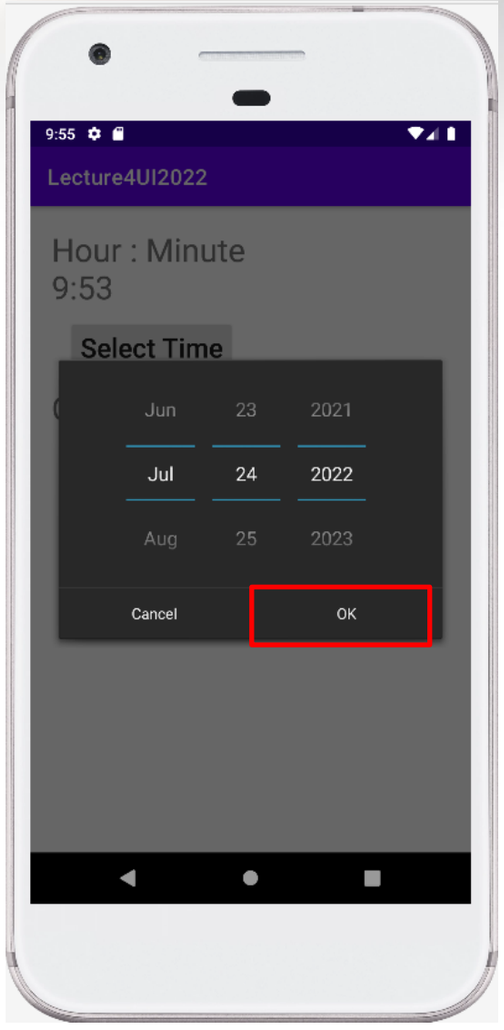


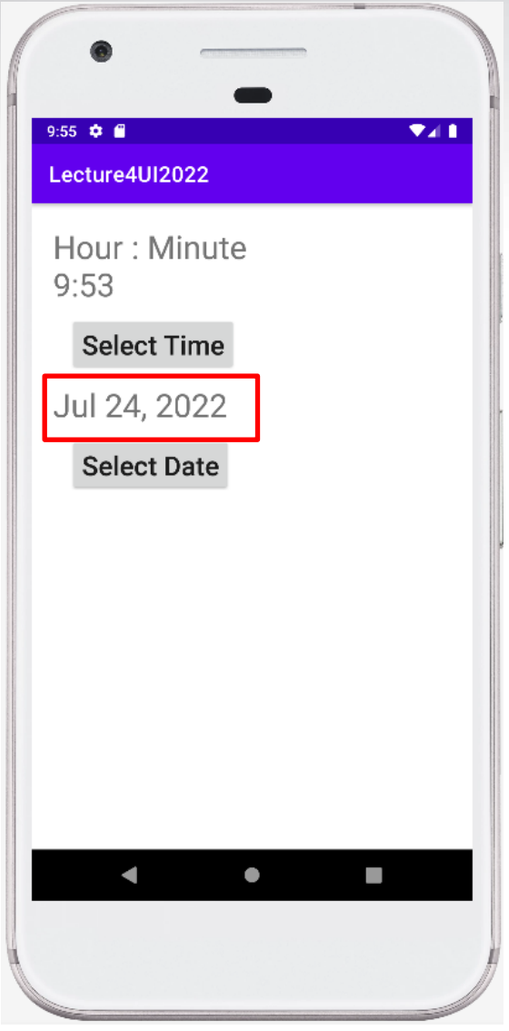




TimePicker and DatePicker



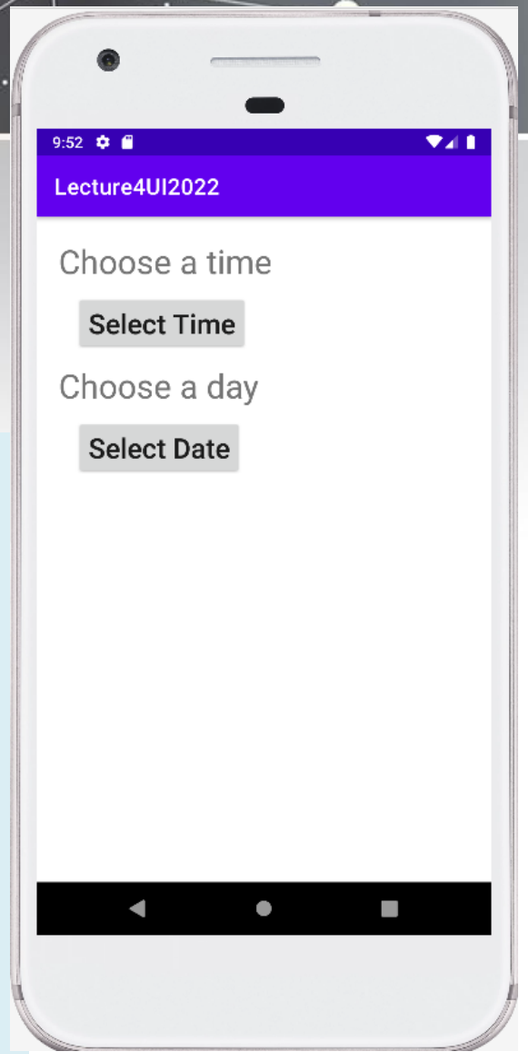




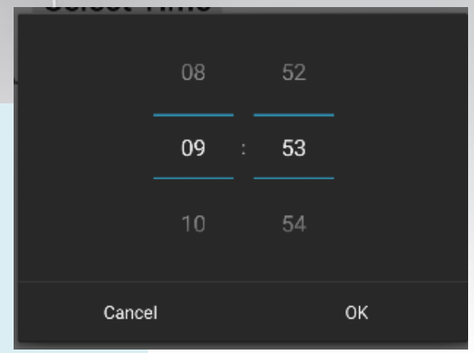
Pickers: activity_main.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:orientation="vertical"
      tools:context=".MainActivity">
<TextView
      android:id="@+id/text_time"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:padding=“10dp"
      android:textSize=“30sp"
      android:text="Choose a time"    />
   <androidx.appcompat.widget.AppCompatButton
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginLeft="25dp"
      android:textSize="25sp"
      android:text="Select Time"
      android:textAllCaps="false"
      android:onClick="showTimePickerDialog" />
```

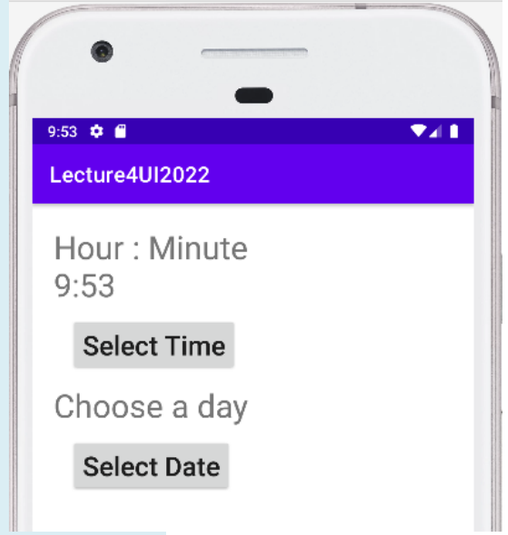
```
 <TextView
      android:id="@+id/text_date"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:padding=“10dp"
      android:textSize=“30sp"
      android:text="Choose a day"
      />
   <androidx.appcompat.widget.AppCompatButton
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginLeft="25dp"
      android:textSize="25sp"
      android:text="Select Date"
      android:textAllCaps="false"
      android:onClick="showDatePickerDialog" />
</LinearLayout>
```



Class : TimePickerFragment.kt

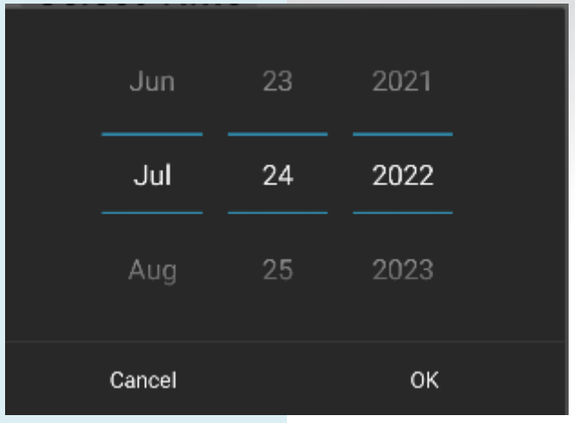



```
class TimePickerFragment:DialogFragment(), TimePickerDialog.OnTimeSetListener {
    override fun onCreateDialog(savedInstanceState: Bundle?): Dialog {
        // Use the current time as the default values for the picker
        val c = Calendar.getInstance()
        val hour = c.get(Calendar.HOUR_OF_DAY)
        val minute = c.get(Calendar.MINUTE)

        // Create a new instance of TimePickerDialog and return it
        return TimePickerDialog(activity, 2, this, hour, minute, true)
    }
    override fun onTimeSet(view: TimePicker, hourOfDay: Int, minute: Int) {
        // Do something with the returned time
        val tv: TextView = activity?.findViewById(R.id.text_time) as TextView
        val minuteNew : String = if(minute<10) "0${minute.toString()}" else  minute.toString()
        tv.text = "Hour : Minute\n$hourOfDay:$minuteNew"
    }
    // When user cancel the time picker dialog
    override fun onCancel(dialog: DialogInterface) {
        Toast.makeText(activity,"Please select a time.", Toast.LENGTH_LONG).show()
        super.onCancel(dialog)
    }
}
```



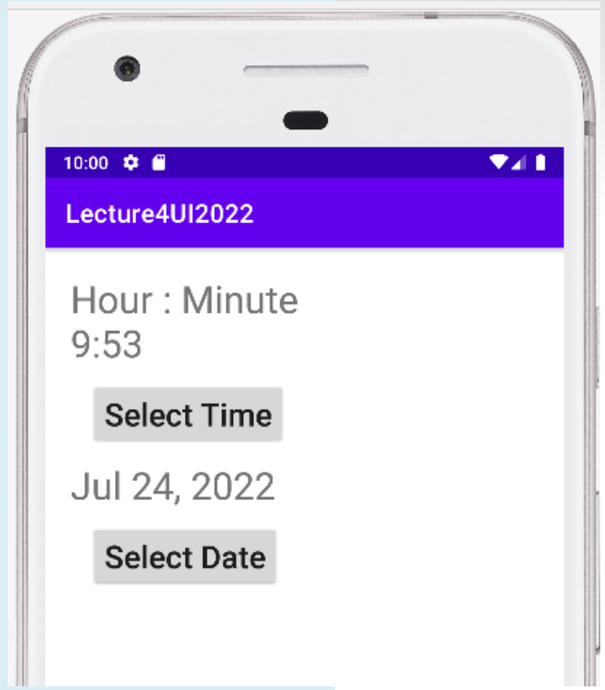
Class : DatePickerFragment.kt

```
class DatePickerFragment : DialogFragment(), DatePickerDialog.OnDateSetListener {
    private lateinit var calendar: Calendar
@SuppressLint("ResourceType")
override fun onCreateDialog(savedInstanceState: Bundle?): Dialog {
        // Initialize a calendar instance
        calendar = Calendar.getInstance()
        // Get the system current date
        val year = calendar.get(Calendar.YEAR)
        val month = calendar.get(Calendar.MONTH)
        val day = calendar.get(Calendar.DAY_OF_MONTH)

        // Initialize a new date picker dialog and return it
        return DatePickerDialog(
            requireContext(), // Context
            // Put 0 to system default theme or remove this parameter
            2, // Theme
            this, // DatePickerDialog.OnDateSetListener
            year, // Year
            month, // Month of year
            day // Day of month
        )
    }
```



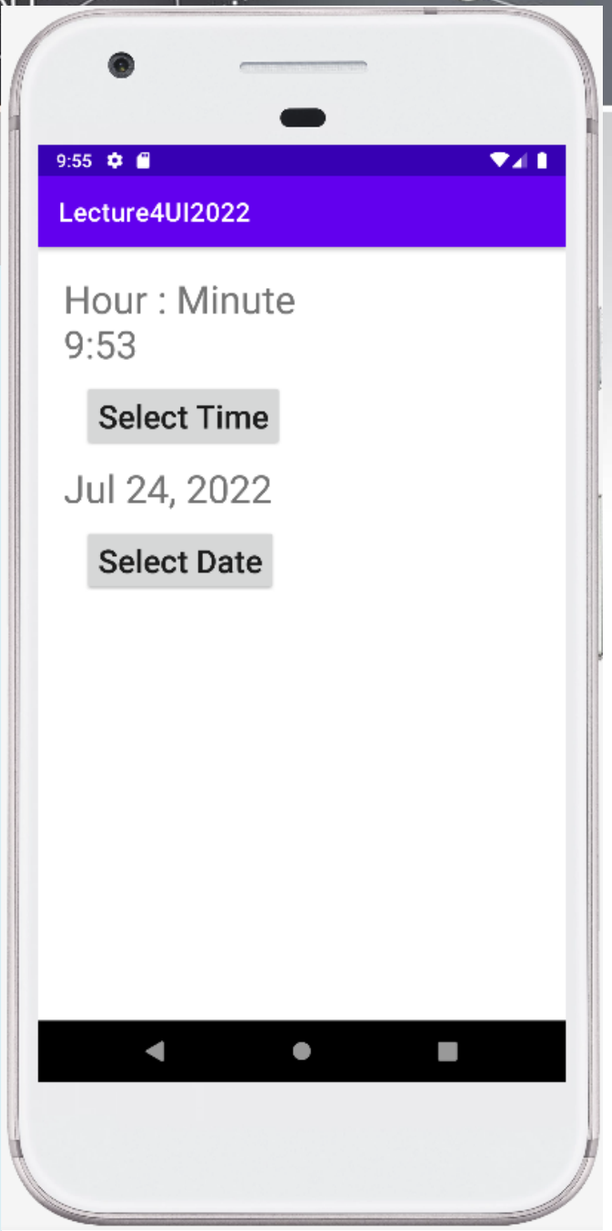
Class : DatePickerFragment.kt (Cont.)

```
// When date set and press ok button in date picker dialog
    override fun onDateSet(view: DatePicker, year: Int, month: Int, day: Int) {
        // Display the selected date in text view
        var tv : TextView? = activity?.findViewById(R.id.text_date)
        tv!!.text = formatDate(year,month,day)
    }
    override fun onCancel(dialog: DialogInterface) {
        Toast.makeText(activity,"Please select a date.", Toast.LENGTH_SHORT).show()
        super.onCancel(dialog)
    }
    // Custom method to format date
    private fun formatDate(year:Int, month:Int, day:Int):String{
        var calendar: Calendar = Calendar.getInstance();
        // Create a Date variable/object with user chosen date
        calendar.set(year, month, day)
        val chosenDate = calendar.time
        // Format the date picker selected date
        val df = DateFormat.getDateInstance(DateFormat.MEDIUM)
        return df.format(chosenDate)
    }
}
```



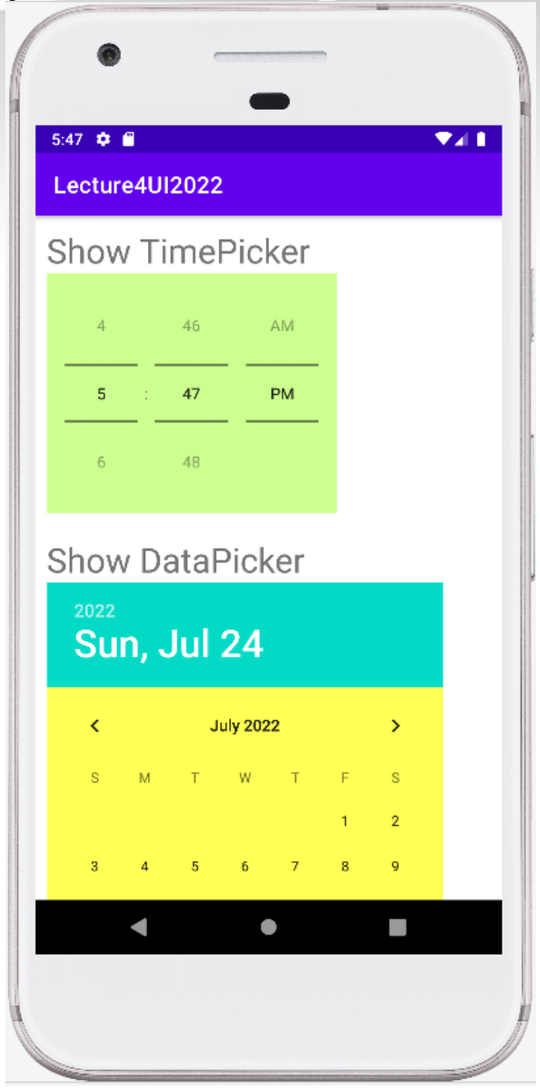
Pickers: MainActivity.kt

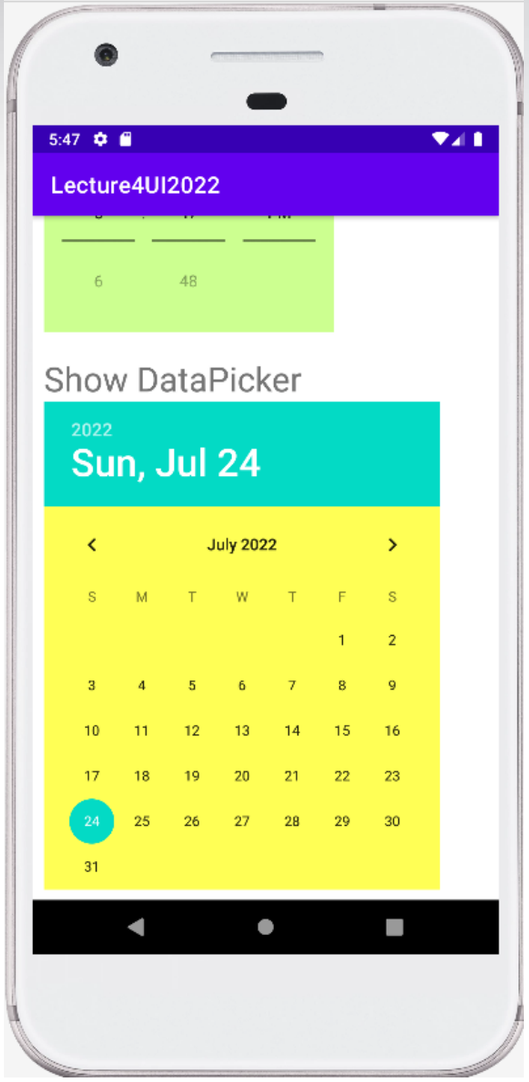
```
class MainActivity : AppCompatActivity() {
    private lateinit var  binding : ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
    }

    fun showTimePickerDialog(v: View){
        val newTimeFragment = TimePickerFragment()
        newTimeFragment.show(supportFragmentManager, "Time Picker")
    }
    fun showDatePickerDialog(v: View){
        val newDateFragment = DatePickerFragment()
        newDateFragment.show(supportFragmentManager, "Date Picker")
    }
}
```



ScrollView

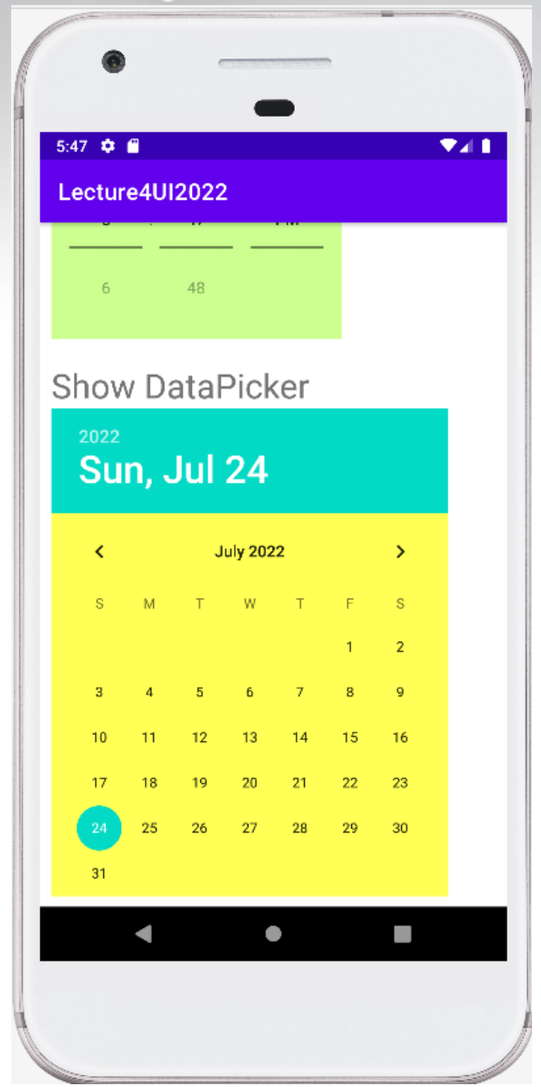
- ScrollView is a view group that is used to make vertically scrollable views.
- A scroll view contains a single direct child only. In order to place multiple views in the scroll view, one needs to make a view group(like LinearLayout) as a direct child and then we can define many views inside it.
- A ScrollView supports Vertical scrolling only, so in order to create a horizontally scrollable view, HorizontalScrollView is used.





ScrollView: activity_main.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
   <LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        tools:context=".MainActivity">


         <!-- everything to show -->


     </LinearLayout>
</ScrollView>
```



End of Chapter

References

- https://www.codeproject.com/Articles/807524/Android-UI-Layouts-and-Controls
- https://www.journaldev.com/9976/android-date-time-picker-dialog
- https://stuff.mit.edu/afs/sipb/project/android/docs/guide/topics/resources/string-resource.html
- https://www.mkyong.com/android/android-imageview-example/
- https://android--code.blogspot.com/2018/02/android-kotlin-timepickerdialog-example.html
- https://android--code.blogspot.com/2018/02/android-kotlin-datepickerdialog-example.html