AAiT

ADDIS ABABA INSTITUTE OF TECHNOLOGY

አዲስ አበባ ቴክኖሎጂ ኢንስቲትዩት

ADDIS ABABA UNIVERSITY

አዲስ አበባ ዩኒቨርሲቲ

# CS 2122:- Mobile Programming

**By Mesfin Belachew /PhD/**

**Assistant Professor,**

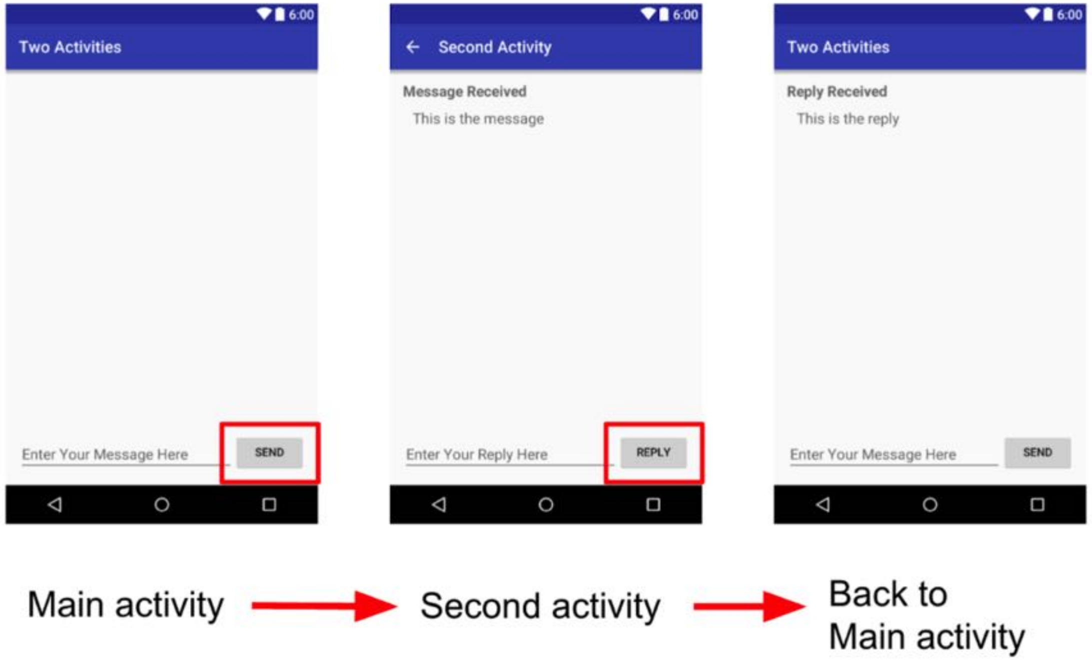
AAiT
ADDIS ABABA INSTITUTE OF TECHNOLOGY
አዲስ አበባ ቴክኖሎጂ ኢንስቲትዩት
ADDIS ABABA UNIVERSITY
አዲስ አበባ ዩኒቨርሲቲ

# Android Programming

# Activities and Intents

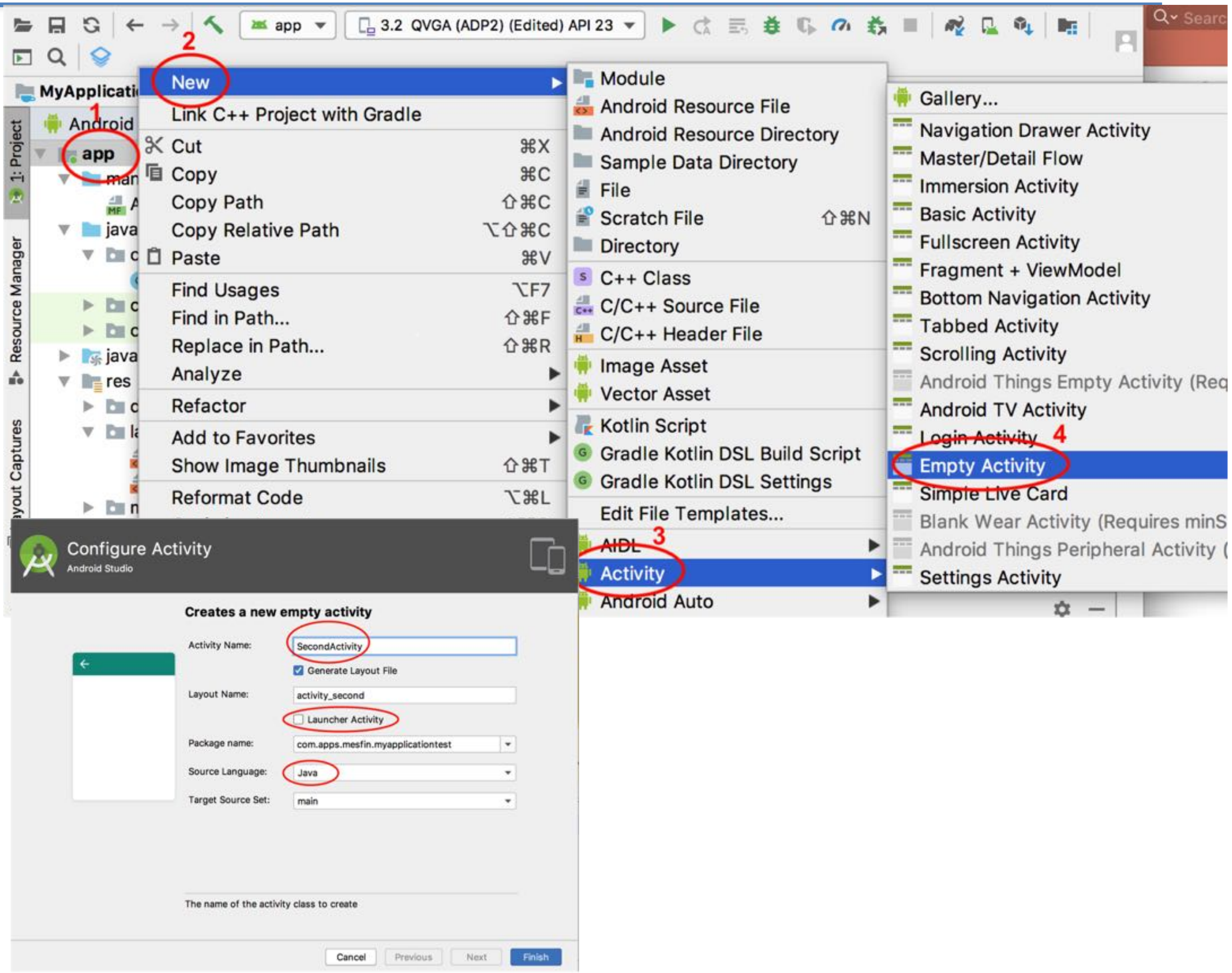
## 1 Activities and Intent

- One App may have multiple activities, each can be used to display a group of components that are related to each other,
- For example, when you click a button in the main activity, user can move to next activity, then from the second to third, etc. and backwards,



# Activates and Intents ….. Activities

- How to create second activity (main activity is created by default), select app, new, activity, empty activity,
- Name the second activity, don’t check the launcher activity option, select language and create,
- Once it is created, a number of changes will happen in different part of the program



Creates a new empty activity

Activity Name: SecondActivity

☑ Generate Layout File

Layout Name: activity_second

□ Launcher Activity

Package name: com.apps.mesfin.myapplicationtest

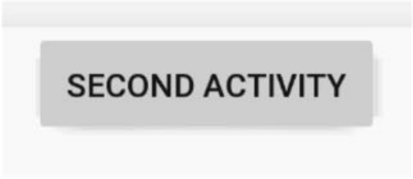
Source Language: Java

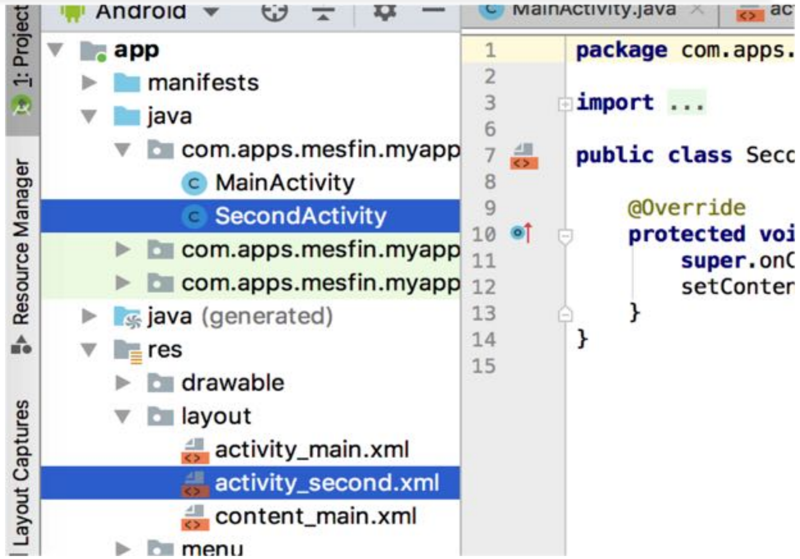
Target Source Set: main

The name of the activity class to create

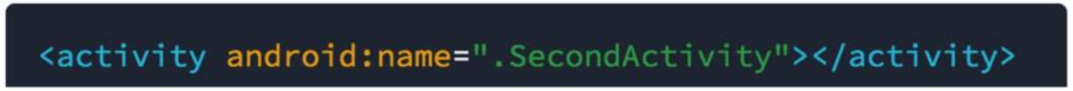
# Activates and Intents ….. Activities

- New files activity_second.xml, and SecondActivity.java are created,
- Changes in AndroidManifest.xml
- Open activity_second.xml and add some components you wish for example TextView
- Once it is done, add intent to call the created activity,
- You can add one button in the activity_main.xml and define onClick action to the button

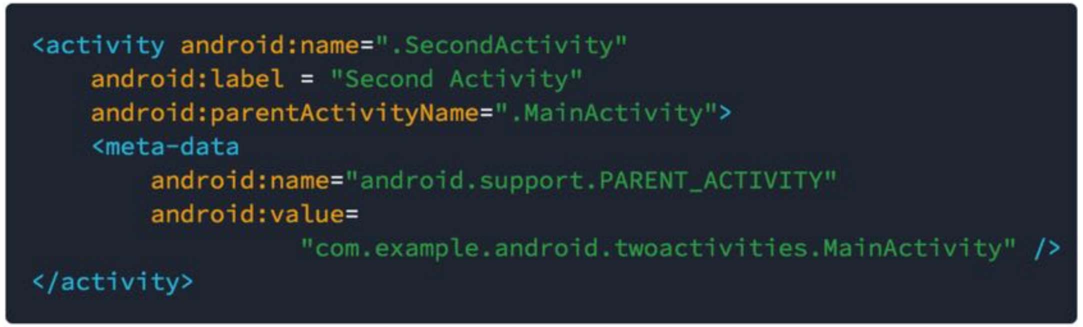




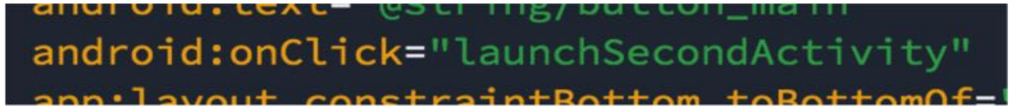
```
package com.apps.

import ...

public class Secc

    @Override
    protected voi
        super.onC
        setConter
    }
}
```



```
<activity android:name=".SecondActivity"></activity>
```



```
<activity android:name=".SecondActivity"
    android:label = "Second Activity"
    android:parentActivityName=".MainActivity">
    <meta-data
        android:name="android.support.PARENT_ACTIVITY"
        android:value=
                    "com.example.android.twoactivities.MainActivity" />
</activity>
```



```
android:text="@string/button_main"
android:onClick="launchSecondActivity"
app:layout_constraintBottom_toBottomOf="
```

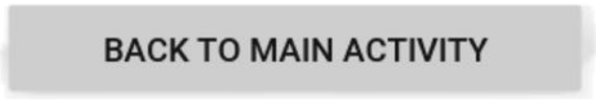
# Activates and Intents ….. Intents

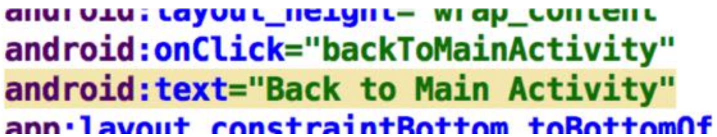
- Intent is created in the main Java code,
- Intent is used to activate the second Activity when the button is clicked,
- In MainActivity.java, add some instructions to define the intent,
- Intent constructor takes two arguments, the context and the class to be called, the java class SecondActivity.java

```
……….
public class MainActivity extends AppCompatActivity {
@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
}
public void launchSecondActivity(View view) {
    Intent intent = new Intent(this, SecondActivity.class);
    startActivity(intent);
}
………..
```

# Activates and Intents ….. Intents

- In the activity_second.xml we need to add a button to go back to main activity,
- Define OnClick action to the button created,
- Define the method backToMainActivity in the SecondActivity.java to finish this activity
- After adjusting the above user can go from main activity to second activity and back,
- Some activity can be associated in each activities, like transferring value from one activity to the second, etc.



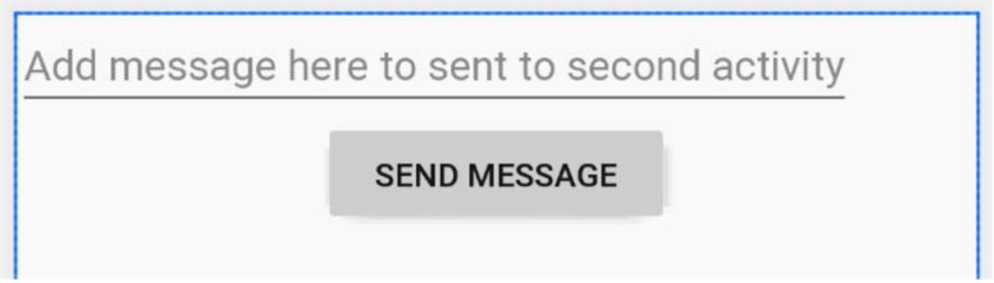


```
android:layout_height="wrap_content"
android:onClick="backToMainActivity"
android:text="Back to Main Activity"
app:layout_constraintBottom_toBottomOf
```

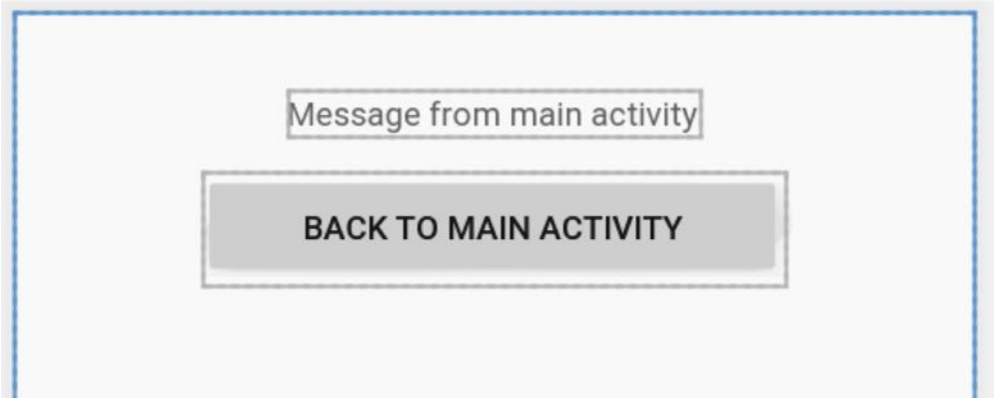
```
……….
public class SecondActivity extends AppCompatActivity {
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout. activity_second);
}
public void backToMainActivity(View view) {
           finish();
}
………..
```

# Activates and Intents ….. Intents

- How to transfer some value from main activity to the second activity,
- Define EditText component and send button in the main activity,
- In the second activity, create TextView and back button,
- After user enters a message in EditText, main activity, and click the send button, the same message should be displayed on the second activity,



Add message here to sent to second activity



Message from main activity

# Activates and Intents ….. Intents

- How to transfer some value from main activity to the second activity,
- Define EditText component and send button in the main activity,
- In the second activity, create TextVie and back button,
- After user enters a message in EditText, main activity, and click the send button, the same message should be displayed on the second activity,
- Hint how to implement ….. In MainAvtivity

```
……….
public class MainActivity extends AppCompatActivity {
public final static String EXT_MESSAGE ="com.app…..MESSAGE";
EditText msgFromMain;
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);
  msgFromMain = (EditText) findViewById(R.id.msgToTransfer);

}
public void launchSecondActivity(View view) {
   Intent intent = new Intent(this, SecondActivity.class);
   String message = msgFromMain.getText().toString();
   intent.putExtra(EXTRA_MESSAGE, message);
   startActivity(intent);
 }
}
………..
```

# Activates and Intents ….. Intents

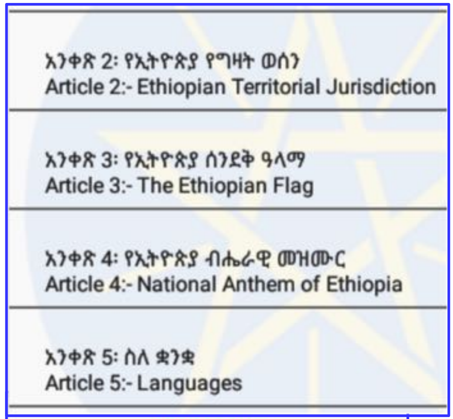
- How to transfer some value from main activity to the second activity,
- Define EditText component and send button in the main activity,
- In the second activity, create TextVie and back button,
- After user enters a message in EditText, main activity, and click the send button, the same message should be displayed on the second activity,
- Hint how to implement ….. in MainAvtivity, and in SecondActivity

```
……….
public class SecondActivity extends AppCompatActivity {
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_second);

  Intent intent = getIntent();
  String message = intent.getStringExtra(MainActivity.EXTRA_MESSAGE);
  TextView textView = findViewById(R.id.textView);
  textView.setText(message);
}
public void backToMainActivity(View view) {
 finish();
  }
}
……….
```

# List View

## 2 ListView

- is a view group that displays a list of related content,
- list is usually can be scrolled vertically,
- two type of list view, ListView & SpinnerView,
- ListView - displays a vertical long list, text, text & image, multiple items, etc.

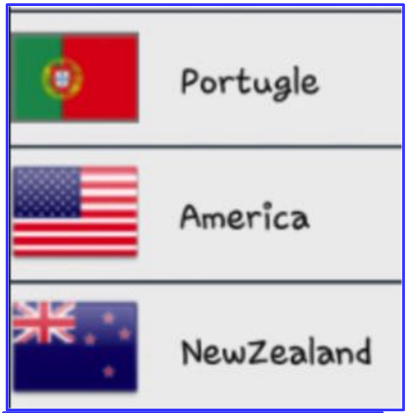


አንቀጽ 2፡ የኢትዮጵያ የግዛት ወሰን
Article 2:- Ethiopian Territorial Jurisdiction

አንቀጽ 3፡ የኢትዮጵያ ሰንደቅ ዓላማ
Article 3:- The Ethiopian Flag

አንቀጽ 4፡ የኢትዮጵያ ብሔራዊ መዝሙር
Article 4:- National Anthem of Ethiopia

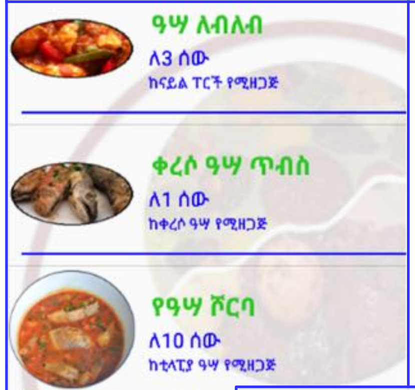
አንቀጽ 5፡ ስለ ቋንቋ
Article 5:- Languages



Portugle

America

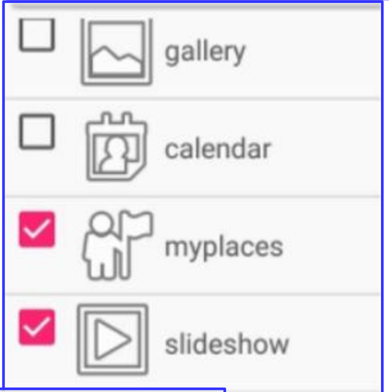
NewZealand



ዓሣ ለብለብ
ለ3 ሰው
ከናይል ፐርች የሚዘጋጅ

ቀረሶ ዓሣ ጥብስ
ለ1 ሰው
ከቀረሶ ዓሣ የሚዘጋጅ

የዓሣ ሾርባ
ለ10 ሰው
ከቲላፒያ ዓሣ የሚዘጋጅ

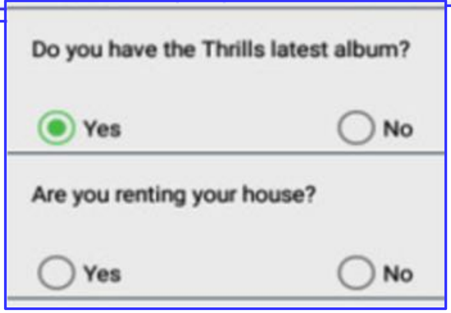


☐ gallery

☐ calendar

☑ myplaces

☑ slideshow



Do you have the Thrills latest album?

● Yes ○ No

Are you renting your house?

○ Yes ○ No

# List View ……

- In activity xml file we need to define ListView, similarly to all components,
- Define a string array to contain all the lists to be displayed in the MainActivity.java,

```
……….
<RelativeLayout
   android:layout_width="wrap_content"
   android:layout_height="wrap_content”
    android:gravity="center">
    <ListView
        android:id="@+id/list_fruits"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</RelativeLayout>
……….
```

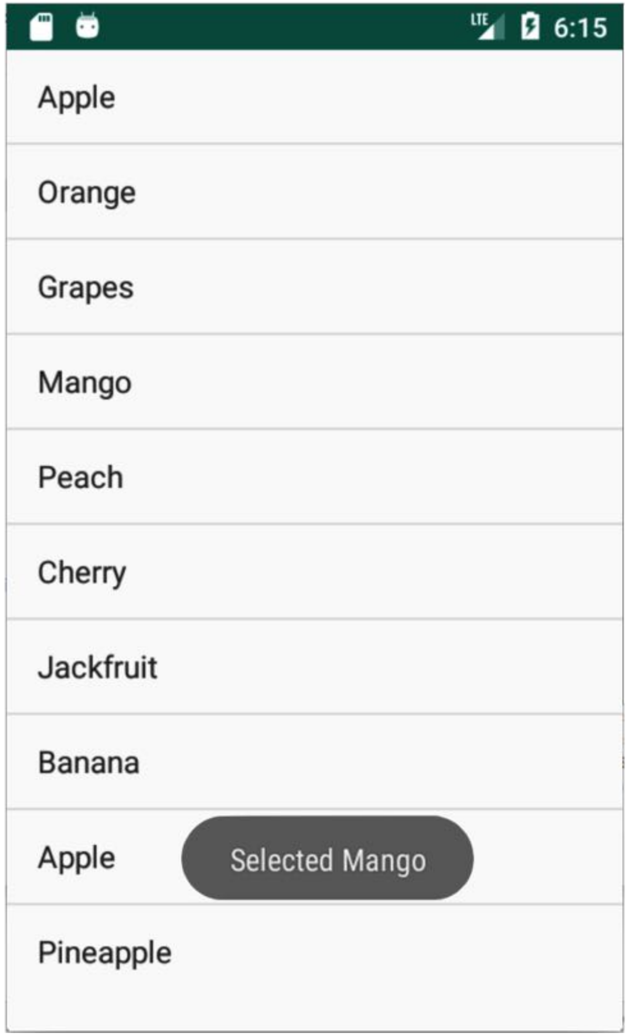
```
……….
public class MainActivity extends AppCompatActivity {
String[] fruits = { "Apple", "Orange", "Grapes", "Mango", "Peach",
"Cherry", "Jackfruit", "Banana", "Apple", "Pineapple"};
@Override
protected void onCreate(Bundle savedInstanceState) {
……….
}
………..
```

# ListView ……

- In activity xml file we need to define ListView, similarly to all components,
- Define a string array to contain all the lists to be displayed in the MainActivity.java,
- Define the ListView in the MainActivity to associate with specific actions when the list is clicked,
- We need also to define a list adaptor to hold the list to be created,
- Set also item click listener to react to the selection

```
……….
public class MainActivity extends AppCompatActivity {
ListView listFruits;
@Override
protected void onCreate(Bundle savedInstanceState) {
……….
  listFruits = (ListView) findViewById(R.id.list_fruits);
  ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
  android.R.layout.simple_list_item_1, fruits);
  listFruits.setListAdapter(adapter);

  listFruits.setOnItemClickListener(
        new AdapterView.OnItemClickListener() {
        @Override
         public void onItemClick(AdapterView<?> adapterView,
              View view, int i, long l) {
            Toast.makeText(getApplicationContext(), "Selected ” +
             fruits[i], Toast.LENGTH_SHORT).show();
        }
  });
}
………..
```

# List View …… Example



6:15

Apple

Orange

Grapes

Mango

Peach

Cherry

Jackfruit

Banana

Apple

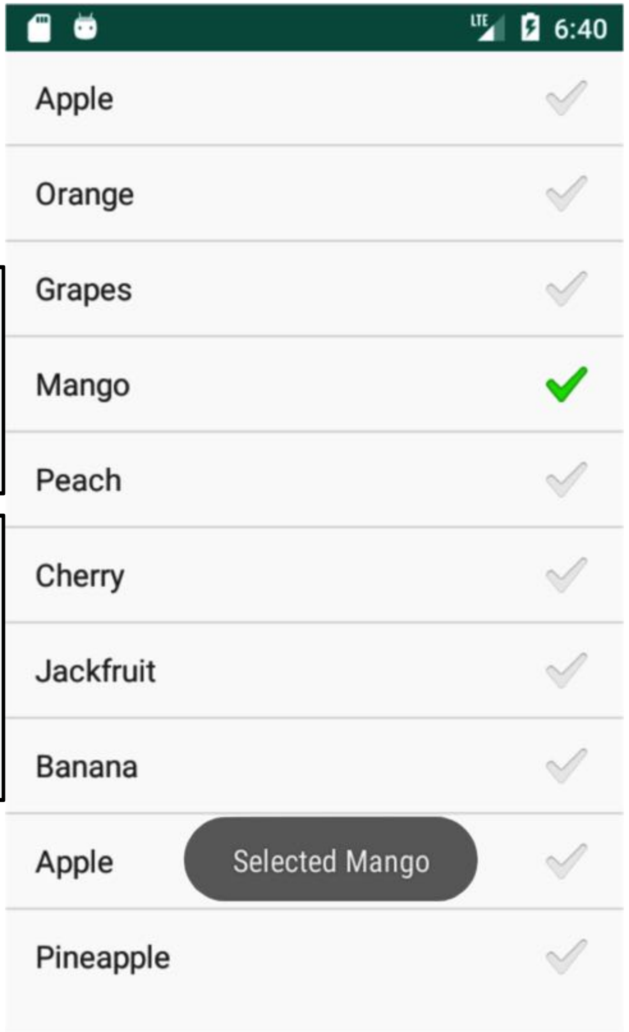
Selected Mango

Pineapple

- Try this  for more practice, change list to checked layout, add choice mode.

```
……….
  ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
  android.R.layout.simple_list_item_1, fruits);
………..
```

```
……….
  ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
  android.R.layout.simple_list_item_checked, fruits);
  listFruits.setChoiceMode(1); // change this value for more
………..
```



6:40

Apple ✓

Orange ✓

Grapes ✓

Mango ✓

Peach ✓

Cherry ✓

Jackfruit ✓

Banana ✓

Apple ✓

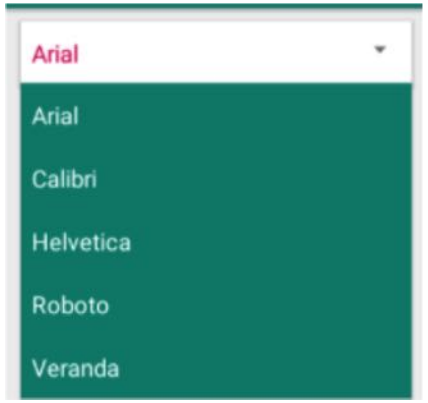
Selected Mango

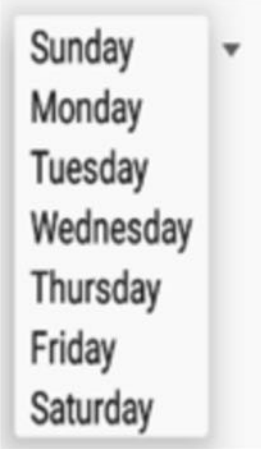
Pineapple ✓

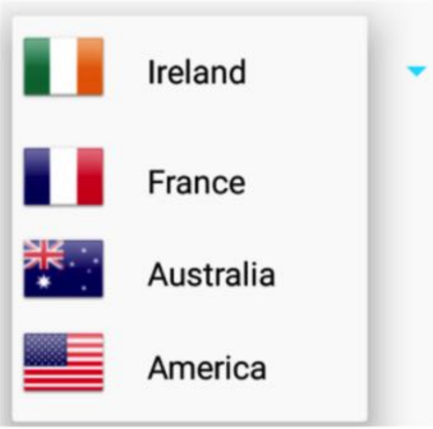
# Spinner View

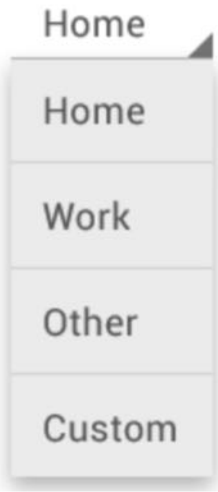
## 3 SpinnerView

- SpinnerView - displays one item at a time from the list,
- provide a quick way to select one value from a set,
- touching the spinner displays a dropdown menu with all other available values,
- displays a vertical long list, text, text & image, multiple items, etc.,









# Spinner View ……

- Spinner to your layout with the Spinner object in xml is to be defined,
- In activity xml file we need to define SpinerView, similarly to all components, with id, width and height,
- to populate the spinner with a list of choices, we need to specify a **SpinnerAdapter** in the source code

```
……….
 <Spinner
    android:id="@+id/spinner_direction"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
……….
```

# Spinner View ……

- Populate the Spinner (three major steps):
  1. defined a **string resource** file, in separate xml file under resource,
  2. supply the spinner with the array using an instance of **ArrayAdapter**, and
  3. Respond to user selection using **OnItemSelectedListener**

1

```
……….
<resources>
   <string-array name=”direction_array">
      <item>North</item>
      <item>South</item>
      <item>East</item>
      <item>West</item>
</string-array>
</resources>
……….
```

2

```
……….
Spinner spinner = (Spinner) findViewById(R.id.spinner_direction);

ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this,
      R.array.direction_array, android.R.layout.simple_spinner_item);

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

spinner.setAdapter(adapter);
……….
```

# Spinner View ……

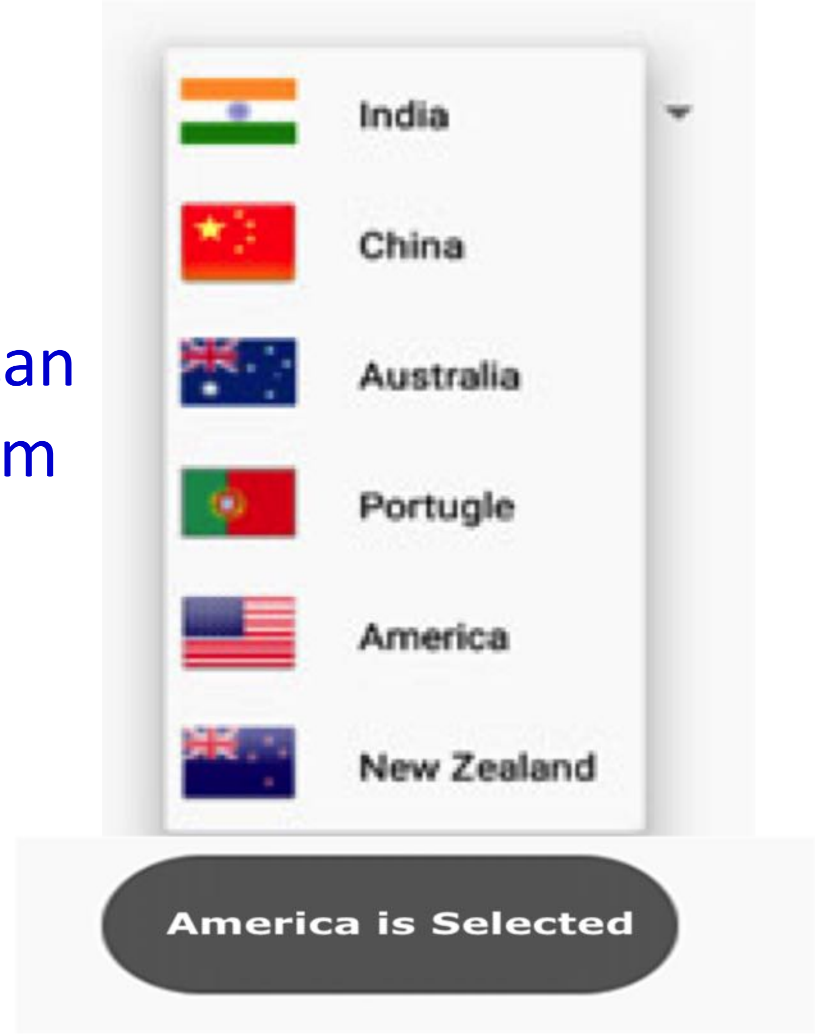
- **OnItemSelectedListener**

3

```
……….
spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
        @Override
        public void onItemSelected(AdapterView<?> adapterView, View view, int pos, long id)
        {
         // do if item is selected
        }
        @Override
        public void onNothingSelected(AdapterView<?> adapterView)
        {
         // do if nothing selected
        }
     });
……….
```

# Spinner View …… Example

- **Spinner Example (practical exercise)**
  - Create the shown Spinner and display the selected country as an output.
  - Hint: ***spinner****.getItemAtPosition(pos).toString()* can be used to extract the selected spinner item

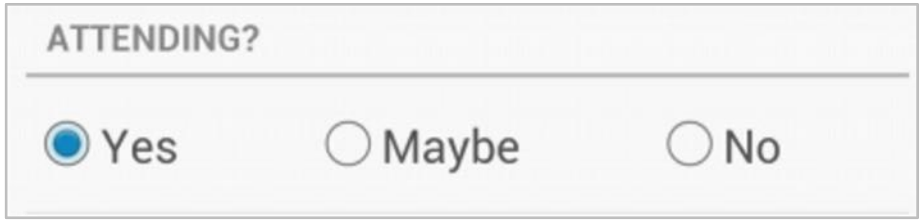


America is Selected

# Radio Button

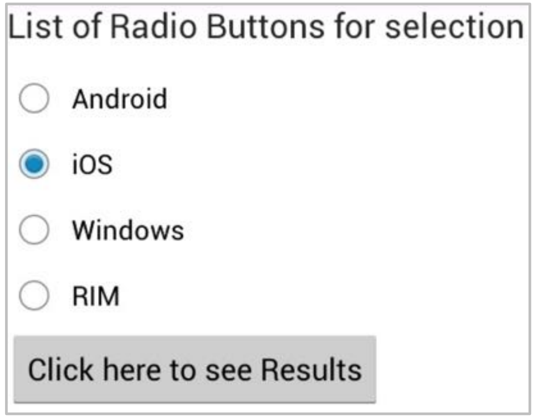
## 4 Radio Buttons

- allow the user to select one option from a set,
- selection for optional sets that are mutually exclusive,
- create a **RadioButton** in your layout (xml file),
- must group them together inside a **RadioGroup**, to achieve exclusivity,
- Only one button is selected from one group,



ATTENDING?

● Yes ○ Maybe ○ No



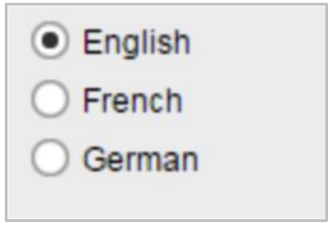
List of Radio Buttons for selection

○ Android

● iOS

○ Windows

○ RIM



● English

○ French

○ German

# Radio Button ……

- Radio button can be implemented in two major steps,
  1. define radio button elements in xml file (layout)
  2. define method to handles the click event  in the Activity (in the source code), this time a method onRadioButtonClicked is defined (onClick)

1

```
……….
  <RadioGroup
    android:id="@+id/selectGender"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical”>
    <RadioButton
      android:id="@+id/radio_male"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:checked="true"
      android:text="Male"
     android:onClick="onRadioButtonClicked"/>
    <RadioButton
      android:id="@+id/radio_female"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Female"
      android:onClick="onRadioButtonClicked” />
  </RadioGroup>
  ……….
```

# Radio Button ……

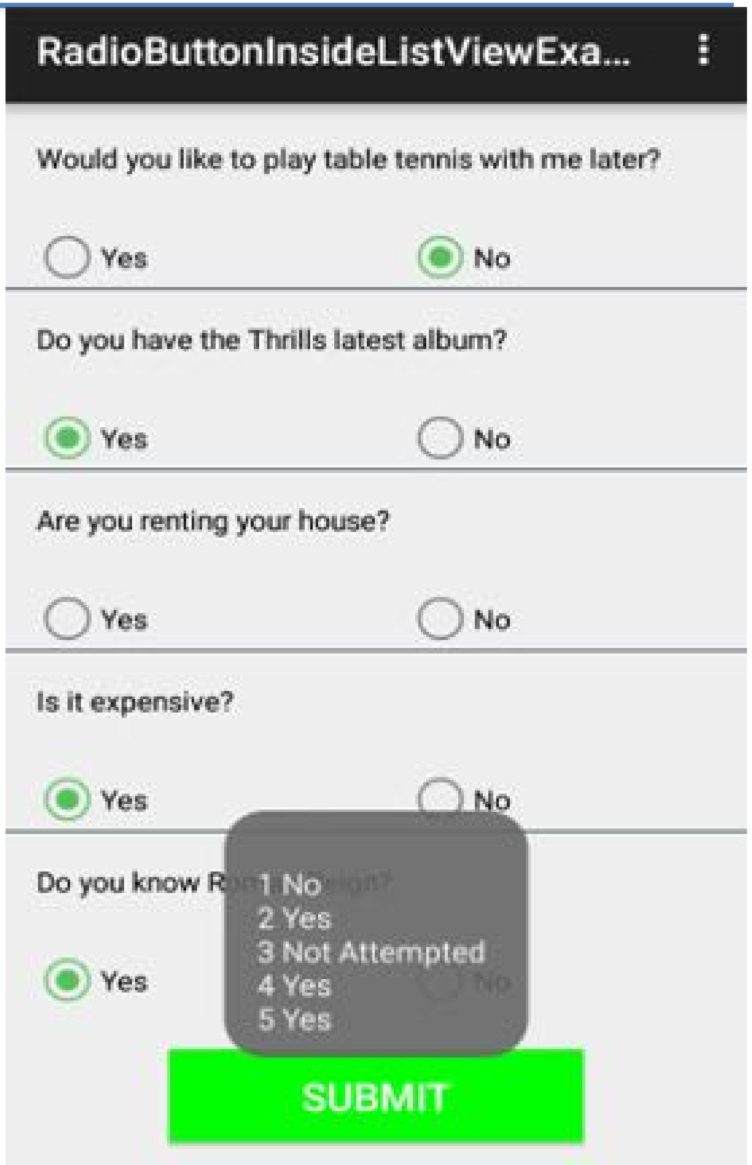
- Define onRadioButtonClicked method:
  - this method will be called when one of the radio button is clicked,
  - Use *Toast.makeText(this, "Male is clicked", Toast.LENGTH_SHORT).show();* to check the result for each cases

2

```
..........
public void  onRadioButtonClicked(View view) {
      boolean checked = ((RadioButton) view).isChecked();
      switch(view.getId()) {
          case R.id.radio_male:
              if (checked)
                  // action when male button is clicked
                  break;
          case R.id.radio_female:
              if (checked)
                  // action when female button is clicked
                  break;
      }
  }
..........
```

# Radio Button …… Example

- **Radio Button Example (practical exercise)**
  - **Create the shown layout and show the output when submit button is created.**



RadioButtonInsideListViewExa...

Would you like to play table tennis with me later?

○ Yes ● No

Do you have the Thrills latest album?

● Yes ○ No

Are you renting your house?

○ Yes ○ No

Is it expensive?

● Yes ○ No

Do you know R

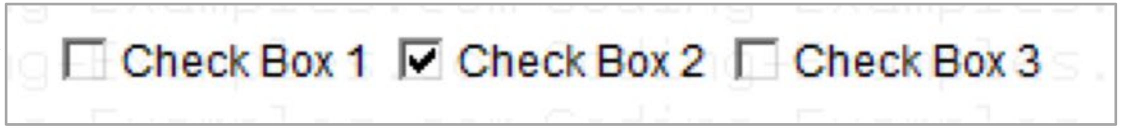
● Yes

1 No
2 Yes
3 Not Attempted
4 Yes
5 Yes

SUBMIT

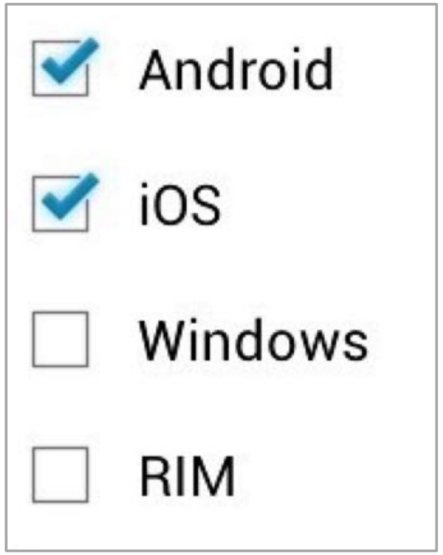
# Check Boxes

## 5 Check Boxes



☐ Check Box 1 ☑ Check Box 2 ☐ Check Box 3

- allow the user to select one or more options from a set,
- CheckBox element will be created in your layout (xml file), each will have id,
- Call specific method when each check box selected and act accordingly from there,



☑ Android

☑ iOS

☐ Windows

☐ RIM

# Check Boxes……

- We can use the following step to crate check boxes:
  1. Create the check box elements in your layout (in the xml file)
  2. Associate onClick  instruction to the method defined in the main activity (in the source code)

1

```
<LinearLayout
        android:id="@+id/checkBoxLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical” >
       <CheckBox
           android:id="@+id/checkbox_meat"
           android:layout_width="wrap_content"
           android:layout_height="wrap_content"
           android:text="Meat"
           android:onClick="onCheckboxClicked” />
       <CheckBox
           android:id="@+id/checkbox_cheese"
           android:layout_width="wrap_content"
           android:layout_height="wrap_content"
           android:text="Cheese"
           android:onClick="onCheckboxClicked” />
     </LinearLayout>
```

# Check Boxes……

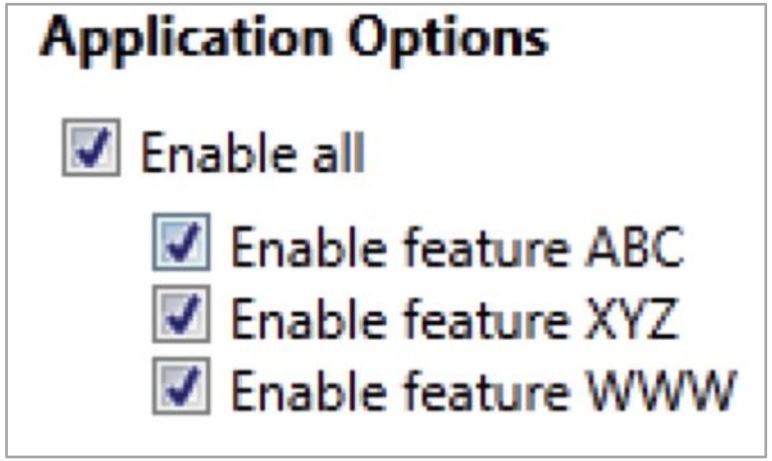
Define onCheckboxClicked method: (in the main activity)

2

```
public void onCheckboxClicked(View view) {
    boolean checked = ((CheckBox) view).isChecked();
    switch(view.getId()) {
        case R.id.checkbox_meat:
            if (checked) {
                Toast.makeText(getApplicationContext(), "Put some meat", Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(getApplicationContext(), "No meat", Toast.LENGTH_SHORT).show();
            }
            break;
        case R.id.checkbox_cheese:
            if (checked) {
                Toast.makeText(getApplicationContext(), "Put some cheese", Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(getApplicationContext(), "No cheese", Toast.LENGTH_SHORT).show();
            }
            break;
    } }
```

# Check Boxes …… Example

- **Check Boxes Example (practical exercise)**
  - Create the shown layout and show the output when a box is always selected.
  - Enable all box is used to check and unchecked all the other three check boxes,



**Application Options**

☑ Enable all

  ☑ Enable feature ABC
  ☑ Enable feature XYZ
  ☑ Enable feature WWW