~~MAD~~ MAD

Assignment-3

1812h011B001
Vishwas Acharya

Page No.: 1
Date: / /

1) What is View and View Groups in Android?

Ans • VIEW

- It is a basic building block of UI in android.
- A view is a small rectangular box that responds to user inputs.
- For example :- the EditText class is used to accept the input from users in android apps, which is a subclass of view, and another example of the TextView class which is used to display text labels in Android apps is also a subclass of View.
- View refers to the android.view.View class
- android.view.View which is the base class of all UI classes.

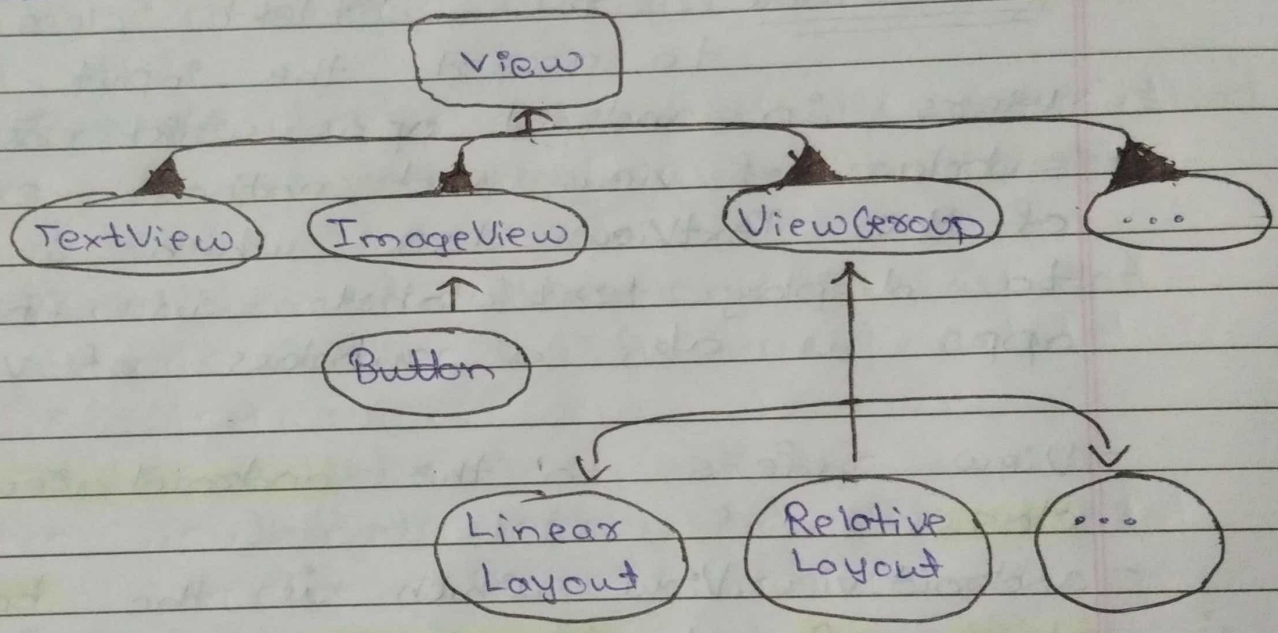
• VIEWGROUP

- It is the invisible container. It holds View and ViewGroup
- It is collection of Views (TextView, EditText, ListView, etc...), somewhat like a container.
- A ViewGroup object is a layout, that is, a container of other ViewGroup objects and View Objects

- For example :- LinearLayout is the ViewGroup that contains Button (View), and other Layout also.

- ViewGroup refers to the android.view.ViewGroup class
- ViewGroup is the base class for Layouts

- Figure :-

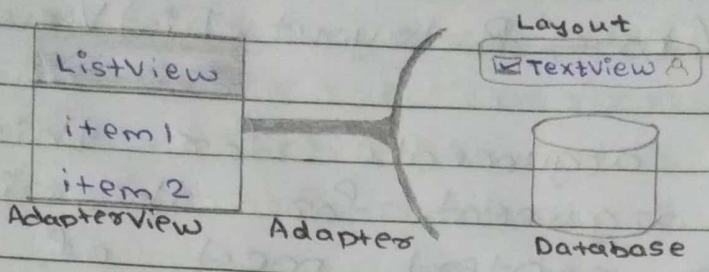


2) What is Adapter View Class in Android?

Ans - AdapterView is a ViewGroup that displays items loaded into an adapter.
- The most common type of adapter comes from an array-based data source.
- It is a bridge between an AdapterView and the underlying data for that view.
- An AdapterView is a group of widgets

components in android that include the ListView, Spinner, and GridView.

- Figure :-



- Hence, is known as AdapterViewClass

3) Explain all types of AdapterView in Android.

Ans - Adapter is a ViewGroup subclass child views are determined by an adapter that binds adapterview object to data of some type.

- Types

1) Array Adapter :- You can use this adapter when your data source is an array.
- By default, ArrayAdapter creates a view for each array item by calling toString() on each item & placing the contents in a TextView.
- Consider you have an array of strings you want to display in a ListView,

initialize a new arrayadapter using a constructor to specify the layout for each string and the string array.

```
ArrayAdapter adapter = new ArrayAdapter
<String> (this, R.layout.ListView, stringArray);
```

- Here are arguments for this constructor.
  - ↳ First argument for this is the application context most of case, keep it this.
  - ↳ Second argument will be layout defined in XML file & having TextView for each string in the array.
  - ↳ Final argument is an array of strings which will be populated in the text view.

- Once you have array adapter created, then simply call setAdapter() on your ListView object as follows-

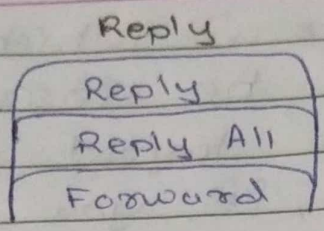
```
ListView listView = (ListView) findViewById(
R.id.listview);
listView.setAdapter(adapter);
```

2. **Simple Cursor Adapter** :- You can use this adapter when your data source is a database cursor.
- When using SimpleCursorAdapter, you must specify a layout to use for each row.

in the cursor & which columns in the cursor should be inserted into which views of the layout.
- For example, if you want to create a list of people's names & phone numbers, you can performs a query that returns a cursor containing a row for each person & column for the names & numbers.

```
String[] fromColumns = {ContactsContract.Data.
DISPLAY_NAME, ContextContract.CommonDataKinds.Phone.NUMBER};
int[] toViews = {R.id.display_name,
                 R.id.phone_number};
```

3) **Spinner Adapter** :- It provide a quick way to select one value from a set.
- In the default state, a spinner shows its currently selected value.
- Touching the spinner displays a dropdown menu with all other available values, from which the user can select a new one.
- It is like the drop down menu with multiple values from which the end user can select only one value.
- For example, when you using Gmail application you would get dropdown menu as shown in figure.



4) **Base Adapter** :- Common base class of common implementation for an Adapter that can be used in both ListView implementation the specialized List Adapter & Spinner.

5) **Cursor Adapter** :- It makes it easy to use when the resource of a listview is coming from database & you can have more control over the binding of data values to layout controls.
- In the newView() method, you simply inflate the view & return it.
- Adapter that exposes data from a Cursor to a ListView widget
- The Cursor must include a column named "_id" or this class will not work.

4) What is Screen Orientation?

Ans - Screen Orientation, also known as screen rotation, is the attribute of activity element in android.
- When screen orientation change from one state to other, it is also known as configuration change.

Page No.:
Date: / /

- **States of Screen Orientation**

- There are various possible screen orientation states for any android application such as:

↳ ActivityInfo.SCREEN_ORIENTATION_PORTRAIT
↳ ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE
↳ ActivityInfo.SCREEN_ORIENTATION_SENSOR
...

- The initial orientation of the screen has to be defined below:-

android:screenOrientation="orientation_type"
↳ It should be defined in AndroidManifest.XML

5) Explain how to Create Menus in Android?

Ans - It is the primary menus of android. They can be used for settings, search, delete item etc.
- It is an important part of UI component which is used to provide some common functionality around the application
- Here, is a steps for create menu.

STEP-1 **Open an Activity class :-** Select your application package & choose "File", "New", then "class" & enter a name of your choice remember to make your class

extende the activity class & add it to the application manifest.

STEP.2 Create a Resources Folder :- The "res" folder holds all of your application resources. To create a ~~memory~~ menu you need a menu folder, so create on inside the "res" folder by selecting it & choosing "File", "New", then "Folders" & entering "menu" as the name.

STEP.3 Create a Menu XML File :- Choose the folder & create a new file by selecting "File", "New", then "File" & entering a name.

```
<menu xmls:android="http://schemas.android.com/apk/res/andro
</menu>
```

STEP.4 Add Items to your Menu :- You can add one or more items to your options menu depending on the needs of your own project

```
<item android:id="@+id/menu_name"
      android:title="menu_name"/>
```

STEP.5 Create Icons for your Menu Items :- Once you have your icons in their folders you can alter your menu item XML to include them as follows:

```
<item android:id="@+id/menu_name"
      android:icon="@drawable/menu_name"
      android:title="menu_name"/>
```

STEP 6 **Inflate Your Name Resource:-** Add the following method to your Java code, inside the class declaration & after the "onCreate" method:

```
public boolean onCreateOptionsMenu (Menu menu)
{
MenuInflater inflater = getMenuInflater();
inflater.inflate (R.menu.my-options_ menu, menu);
return true;
}
```

STEP 7 **Detect User Interaction:-** Add following method outline after the "onCreateOptionMenu" method:

```
public boolean
onOptionItemSelected (MenuItem item) {
// respond to menu item selection}
```

STEP 8 **Respond to Menu Item Selection:-** Add a switch statement to your method. using the following syntax:

```
switch (item.getItemId()){
case R.id.menu_name:
startActivity (new Intent (this, menu_name.class);
return true;
```

Page No.:
Date: / /

```
case R.id.menu_name:
startActivity(new Intent(this, menu_name.class));
return true;

default:
return super.onOptionsItemSelected(item);
}
```