ANDROID

Class notes

By

Mr . **Naveen** sir

MV ST

RS = 130/-

* Android *

(1) Open Source.

(2) Free from Royalty. (No payment)

(3) Device Independent.

Q. What is an Android ?

→ Open Source & Independent.

i) Android is a Linux based mobile O.S which is specially designed for mobile & tablet PC's.

ii) Now Android is available for wear, mobiles, tablets, android TV and android auto.

Software
↓
A set of program
↓
set of statements
↓
Instruction of computer
↓
by programming Lang.

* Software :

— A set of program align in order to perform a specific task.

— As per industry standards, software is categorised into two ways —

(1) System Software.

(2) Application Software.

① System Software :

— Makes hardware components to work.

— We will use it.

Ex. Android O.S.

② Application Software :

— Is a software which is design under client requirement.

— As per industry standards application software is categorised in two types.

a) Standalone Application.

b) Distributed Application.

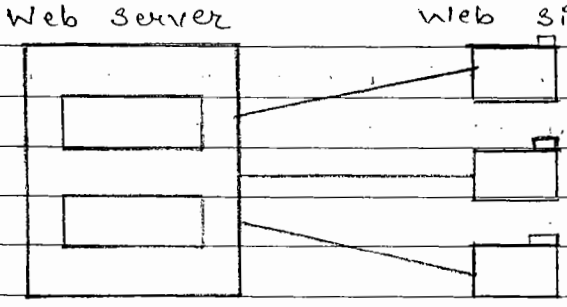
a) Standalone Application :

The software which is installed into a specific device and which works in the same device only, is called standalone application.

NOTE : All Android applications are standalone applications.

b) Distributed Application :

The application which is installed into the server side and which works in client device is called as distributed application.



Ex. Facebook, gmail, orkut.

* Android Versions :

Version		Nick Name
① 1.x		
android	1.0 ⟶	Android.
	1.1 ⟶	Beta Android.
	1.5 ⟶	cup cake.
	1.6 ⟶	Donut.

a) The android 1.x is the first version released by google incorporation.

b) The android 1.x is specially designed for mobiles only.

② 2.x		
android	2.0 ⟶	Eclair.
	2.0.1 ⟶	Eclair.
	2.1 ⟶	Eclair.
	2.2 ⟶	Froyo.
	2.3 ⟶	Ginger Bread.
	2.3.x ⟶	Ginger Bread.

a) The android 2.x is also designed for mobiles only, but from 2.x all other versions are started supporting google applications, such as google maps, chrome etc.

③ 3.x		
android	3.0 ⟶	Honey Comb.
	3.1 ⟶	Honey Comb.
	3.3 ⟶	Honey Comb.

a) The android 3.x is specially designed for tablet pc's.

④ 4.x

android 4.0 ⟶ Ice cream Sandwich.
4.0.3 ⟶ Ice cream Sandwich.
4.1.2 ⟶ Jelly Bean.
4.2 ⟶ Jelly Bean.
4.3 ⟶ Jelly Bean.
4.4.2 ⟶ KitKat
4.4w ⟶ KitKat (for wear).

a) The android 4.x version is compatible for both tablets and mobiles.

b) The android 4.4w version is used to develop android wear application.

⑤ 5.x

android 5.0 ⟶ Lollipop.
5.0.1 ⟶ Lollipop.
5.0.2 ⟶ Lollipop.
5.1 ⟶ Lollipop.
5.1.1 ⟶ Lollipop.

a) The android 5.x version is available for mobiles, tablets and Big Screen (TV).

* Android Features :

① Android OS is open Source :

It means by using android applications framework we can design any type of applications as well as we can modify android operating system also.

② Android OS is free from Royalty :

③ Android OS is device independent :

It means the android OS can be installed into any device.

④ Android OS will support IPC. (Interprocess Communication).

⑤ Android OS will support NFC. (New feild Communication) :

In NFC by touching two mobiles we can transfer the data from one mobile to another mobile.

NOTE : The two mobiles must maintain less than one meter distance.

- Android mobile will support text to speech & speech to text.

- Android mobile will support all types of audio's and video's formates.

- Android OS mobile will support all types of images.

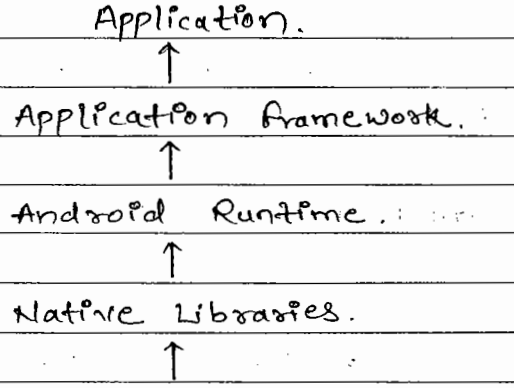
- Android OS mobile can be controlled through voice commands.

- Android OS mobile will support push mails, push notifications.

* Android Architecture :

The Android Architecture contains 5 - Layers -

① Linux Kernal.
② Native Libraries.
③ Android Runtime.
④ Application framework.
⑤ Application.



① Linux Kernal :

- It is a layar of your android architecture.
- This layer is responsible for memory mgmt, power mgmt, device mgmt, broadcast mgmt, service management etc.

② Native Libraries :

- The Native Libraries such as open GL, freetype, SQLite, media etc.

- A Native Library is a library written in a langauge that compiles down to native code for the platform it runs on.

- SQLite Library is responsible for database.

- freetype Library is responsible for font support.

- Media Libraries are responsible for playing & recording audio & video formats.

- Open GL (Open Graphic Library) is responsible for 2D & 3D graphic support.

- SSL (Secure Socet layer) is technically known as transport layer security and this library is responsible for encrypted communication between client & server.

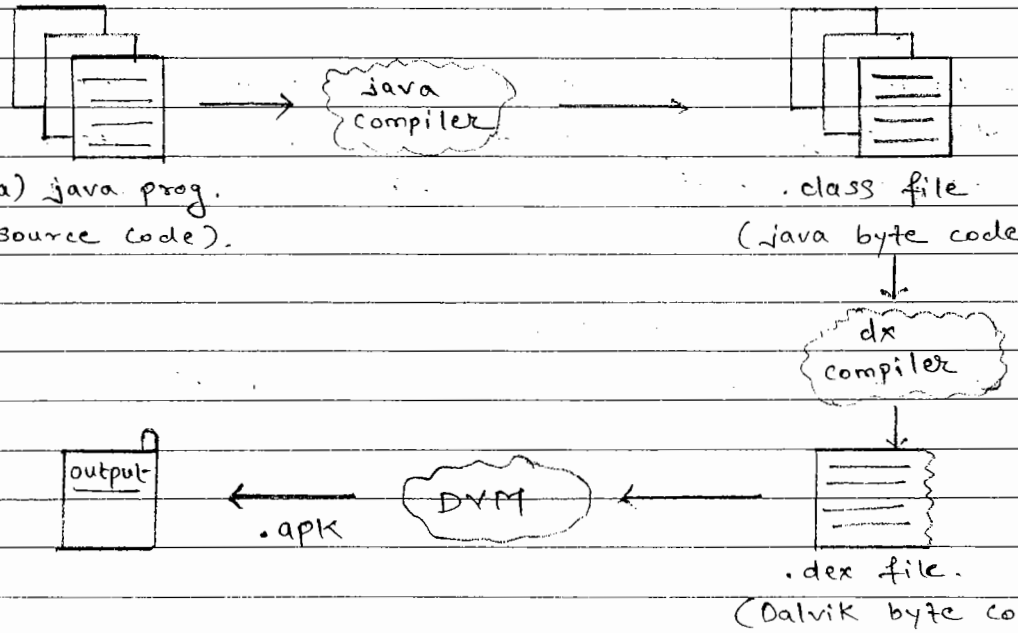
- Graphics supports the basic graphic operations for device, & used for basic applications graphics.

③ Android Runtime :

- In Android runtime there are core libraries and DVM.

- DVM stands for Dalvik Virtual Machine, which is responsible to run android applications.

- DVM is like JVM but it is optmised for mobile devices where it consumes less memory and faster performance.



a) .java file contains source code which can understand by java compiler.

b) .class files contains byte code which is provided by java compiler & which is understand by dx compiler.

c) .dex stands for dalvik executables files.

d) The dex file contains dalvik byte code which is provided by the dx compiler & is understand by DVM.

e) The DVM will execute dalvik executable files & generate ".apk" file

f) .apk stands for android package which is used to install application into a device.

(4) Android Application framework :

Q. What is framework ?

- Framework is a predefined architecture which contains set of classes, interfaces and abstract classes.

- Android framework includes android API's (Application programming interface) such as telephony, user interface, locations, content provider & package managers.

- The android application framework provides classes and interfaces to develop android applications.

⑤ Applications :

- All applications such as Home, Contact, settings, games, browsers are using android framework that uses android runtime & libraries.

- Android Runtime & Native Libraries are using Linux Kernal.

- Required software for android application development :

a) JDK - 8.

b) ADT Bundle / Android studio.

(IDE) Eclipse SDK (software Development Kit)

* ADT Bundle :

1. Download ADT Bundle.

2. Open ADT Bundle.

↳ In ADT Bundle open Eclipse folder.

↳ Run ".exe" file to open Eclipse IDE.

➤ Provide Workspace.

3. In Eclipse IDE menu -

Select → Window → Android SDK

4. After that install : Tools completedly.

Android 5.1.1.

Android 5.0.1.

Android 2.3.3.

Extras.

* Android Application Development :

→ Steps to create Android virtual Device Manager :

1. Open Eclipse IDE.
   ↳ In Eclipse IDE Menu → Window → AVD Manager (click) → create.

AVD Name : Sathya_4to6 (No spaces).
Device : 3.2" QVGA (ADP2)
Target : Android 2.3.3 - API Level 10.
CPU : ARM (armeabi)
Keyboard : ✓
Skin : Skin with dynamic hardware contents.

Next click on [OK] Button.

→ Steps to start AVD :

1. In Eclipse IDE Menu : Windows → AVD Manager.
2. In AVD Manger window, select created AVD and click on [start] button & then [Launch].

→ Steps to create Android Applications :

1. Open Eclipse IDE.
2. Menu → select file → New → Android Appln project.
3. Application Name [ ] (What i can see in mobile)

   Project Name [ ] (What i can see in Eclipse IDE)

   Package Name [ ] (* change Package).

click [Next] and then [Next]

4. Provide Icon to android application.
   - Image, clipart or text with required background with circle, square, none appln.
   - [Next]
   - [Next]
   - [Finish].

→ Steps to run android application in Eclipse :

1. Select the project from package Explorer in Eclipse.

2. Right click on selected project -
   choose Run as → Android application.

* File Hierarchy of Android Application Projects in Eclipse IDE :

→ MyFirstAndroidApp
- → src → (Source)
  - This folder contains .java files.
- → Gen → (Generated)
  - This folder contains auto generated files such as -
    - ⓐ R.java files
    - ⓑ BuildConfig.java files.

NOTE : We should not modify any files in 'Gen' folder.

→ Android 5.1.1
→ Android Private Libraries.
→ Android Dependencies.

} This are android system Resources which are help to generate .dex & .apk & Properties files.

→ bin → (binary)
  ↳ This folder contains .dex files & .apk files.

→ libs → (Libraries)
  ↳ This folder contains 'External Libraries' in the format of 'jar' files.
  ↳ Jar stands for Java archive.

→ res → (Resourses)
  ↳ drawable-hdpi
  ↳ drawable-idpi
  ↳ drawable-mdpi
  ↳ drawable-xhdpi
  ↳ drawable-xxhdpi
  } Copy Images.
  ↳ Layouts
    ↳ This folder contains user interface files in xml formats.
  ↳ menu
    ↳ Menu files.
  ↳ Values
    ↳ This folder contains style, string, diamensions resources.

* Steps to Create Android Application :

(1) Open Eclipse IDE.

(2) Create an android application project.
NOTE : In Create Activity window
Uncheck ☐ create Activity.
And Directly click on Finish button.

(3) Create a new Layout.

Project
→ res
→ layout
→ (Right click on New folder)
New → Android xml file.
ⓐ File : first_layout.
NOTE : Layout file Name must be in small letters only.
ⓑ From Root Element select Layout type
Ex. LinearLayout.
And click on Finish button.

(4). Create an Activity.

Project
→ src
→ (Right click on src folder & create a new package by selecting -
New → Package.)
NOTE : The package Name must match with "gen" folder, "package name".
Name : com.nmk.cwproject1;
And click on finish button.

(b) Right click on created package & select- New → class.

(c) Name : MyActivity.

And click on finish button.

* Layouts : [Presentation Layer.]

- A layout defines a visual structure where an end user can interact with it.

- This layout can be an activity or an widget. This layout are categorised into four types :

(1) LinearLayout.

(2) Relative Layout.

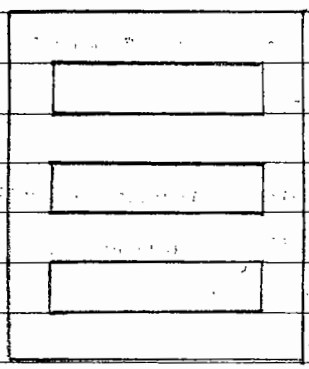
(3) Table Layout.

(4) Absolute Layout.

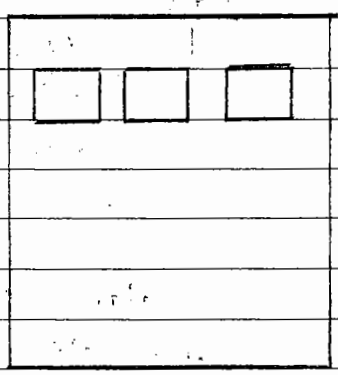
(1) Linear Layout :

- is a view group that aligns all child components in a single directions.
i.e vertical or Horizontal.

Vertical



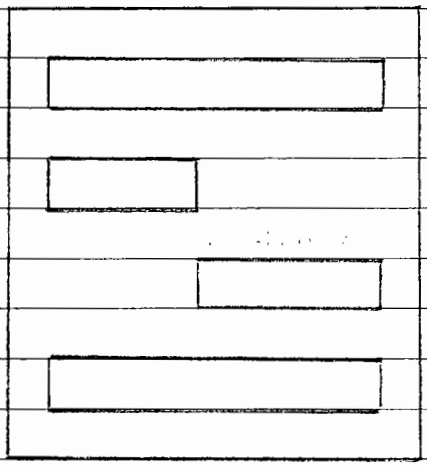
Horizontal.



② Relative Layout :

- Relative Layout allows you to position your components position.

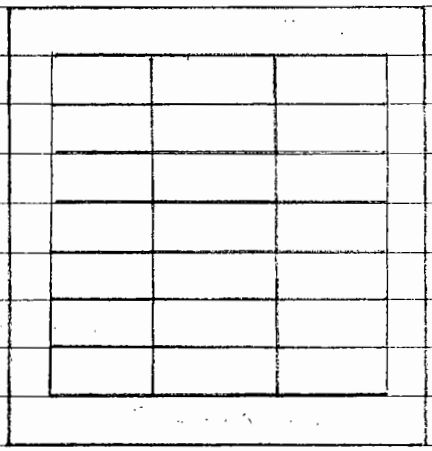
- Relative Layout is most flexible Layout that allows to position your components to display anywhere in the Layout.



* Relative Layout *

③ Table Layout :

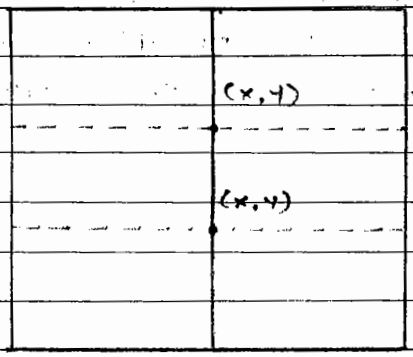
- is use to arrange your components based on rows & columns.



* Table Layout *

④ Absolute Layout :

— Absolute Layout is use to arrange your components based on x & y axis coordinates.



* Absolute Layout *

→ Linear Layout Snippet :

```
<LinearLayout
        android : id = "@+id/LL1"
        android : layout_width = "fill_parent"
        android : layout_height = "fill_parent"
        android : orentation = "vertical"
        android : background = "#789456"
        xmls : android = "-------------">

</LinearLayout>
```

→ Explaination :

<LinearLayout ——————→ (open Element)

```
        android : id = "@+id/LL1"
        android : layout_width = "fill_parent"
        android : layout_height = "fill_parent"
```

(Attributes)

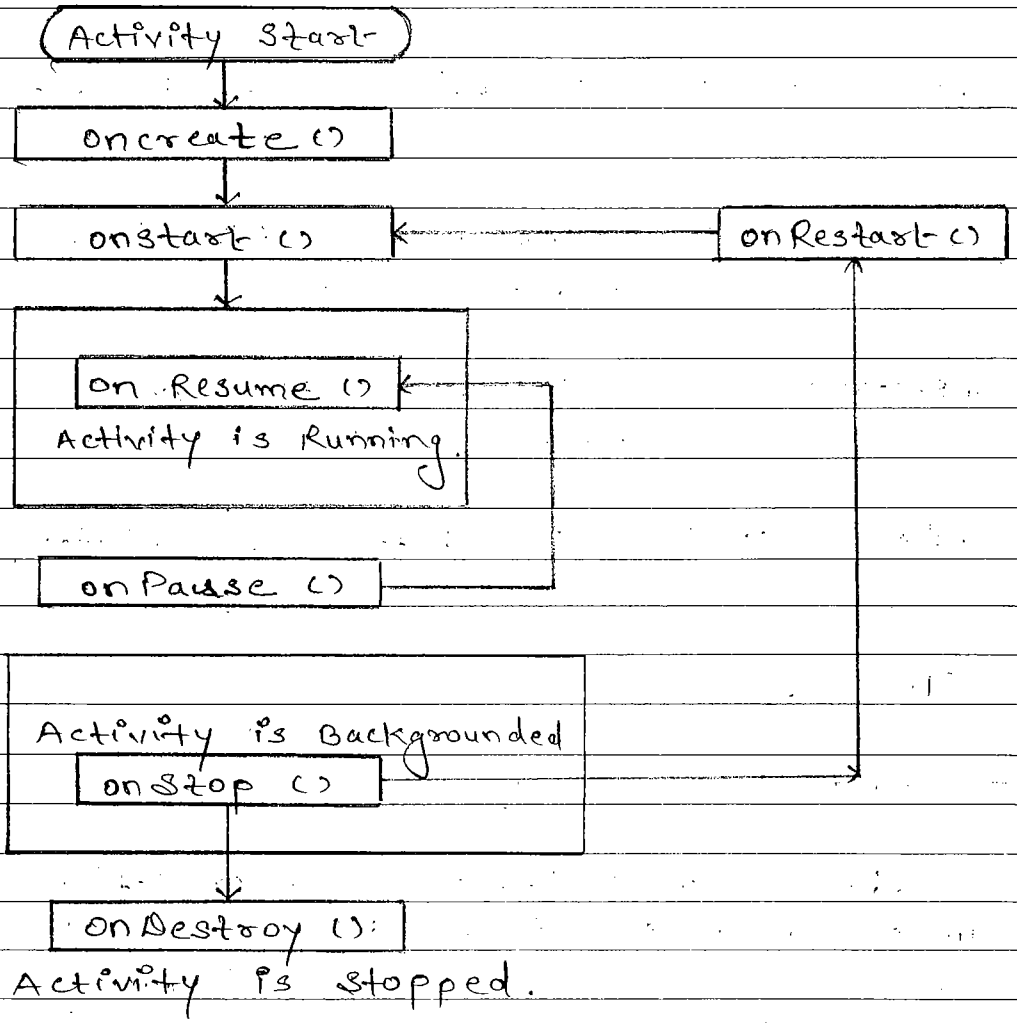
</LinearLayout> ——————→ (closed Elements)

* **Activity** :

- Activity is a java code that supports a screen or user interface. In otherwords, Activity is a building block of the user interface.

NOTE : Activity is a predefined class in Android. Every Application which has user interface must inherit activity to create a window.

* **Activity Lifecycle** :



* Activity Lifecycle Methods :

Methods.	Descriptions.
① onCreate () :	called when activity is first created.
② onStart () :	called when activity is becomming visible to user.
③ onResume () :	called when activity will start interacting with the user.
④ onPause () :	called when activity is not visible to user.
⑤ onStop () :	called when activity is no longer visible to user.
⑥ onRestart () :	called when after your activity is stopped, prior to start.
⑦ onDestroy () :	called before the activity is destroyed.

* Toast Class :

- Toast is a dialog.
- The android Toast is a predefined class ~~under~~ used to display information for the short period of time & Disappears after some time.

Methods	Descriptions.
① public static Toast makeText (Context context, char sequence text, int duration)	— Makes the toast containing text & duration.
② public void show ()	— Display Toast.

* Feilds (Public static final Variables)
  - [Nothing but constants]

Constant	Description
① Public static final int LENGTH_LONG	— Displays view for long duration of time.
② Public static final int LENGTH_SHORT	— Displays view for short duration of time.

→ Snippet :

```
- Toast t = Toast.makeText (this, "I am Toast",
                Toast.LENGTH_LONG);
- t.show ();
```

OR

```
- Toast.makeText (this, "I am Toast", Toast.LENGTH_LONG).
    show ();
```

* Activity Life Cycle Program :

* CW Project-1 *

→ My Activity class :

```
import android.app.activity;
import android.as.bundle;
import android.widget.Toast;

public class MyActivity extends Activity
{
    @Override
    protected void oncreate (bundle savedInstanceState)
    {
        super.oncreate (savedInstanceState);
        setContentView (R.layout.first_layout);
        Toast.makeText (this, "I am Toast",
                Toast.LENGTH_SHORT).show();
    }

    @Override
    protected void onStart()
    {
        Super.onstart();
        Toast.makeText (this; "I am onstart",
                Toast.LENGTH_SHORT).show();
    }

    @Override
    protected void onResume()
    {
        Super.onResume();
        Toast.makeText (this, "I am onResume",
                Toast.LENGTH_SHORT).show();
    }
```

```
@Override
protected void onPause ()
{
    Super. onPause ();
    Toast. makeText (this, "I am onPause",
        Toast. LENGTH_SHORT). Show ();
}

@override
protected void onStop ()
{
    Super. onStop ();
    Toast. makeText (this, "I am onstop",
State)          Toast. LENGTH_SHORT). show ();
}

@ Override
protected void onRestart ()
{
    Super. onRestart ();
    Toast. makeText (this, "I am onRestart",
        Toast. LENGTH_SHORT). show ();
}

@Override
protected void ondestroy ()
{
    Super. onDestroy ();
    Toast. makeText (this, "I am onDestroy"
        Toast. LENGTH_SHORT). Show ();
}
}
```

→ Manifeast file :

↳ Android Manifeast . xml.

```
< application
  _ _ _ _ _ _ _ _ _ _
  _ _ _ _ _ _ _ _ _ _
  _ _ _ _ _ _ _ _ _ _ >
    < activity
        android : name = " com. nmk. CWProject1. MyActivity" >
      < intent - filter >
        < action android : name = " android . intent.
              action . Main " />
        < category android : name = " android .
              intent. category . LAUNCHER " />
      </ intent - filter >
    </ activity >
</ application >
```

* CWProject-2 *

Q. Write an android application for changing Layout Background with image.

NOTE : Wherever the image in PC, copy that image into drawables folder and then use it.

↳ changeImageLayout.xml

```
<LinearLayout
    android:id="@+id/LL1"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:Background="@drawable/mat">

</LinearLayout>
```

↳ changeImageActivity.java

```
import android.app.activity;
import android.os.bundle;

public class changeImageActivity extends Activity
{
    @Override
    protected void onCreate(bundle savedInstanceState)
    {
        Super.onCreate(savedInstanceState);
        setContentView(R.Layout.changeImageLayout);
    }
}
```

↳ Android Manifeast . xml .

```
<application
    <activity
        android : name = "com.nmk.cwproject2.
                changeImageActivity" >
        <intent_filter>
            <action android:name = " android.intend.
                action.MAIN "/>
            <category android:name = " android.
                intent:category.LAUNCHER "/>
        </intent_filter>
    </activity>
</application>
```

* Input Controls :

— The input controls are interactive components in your application.

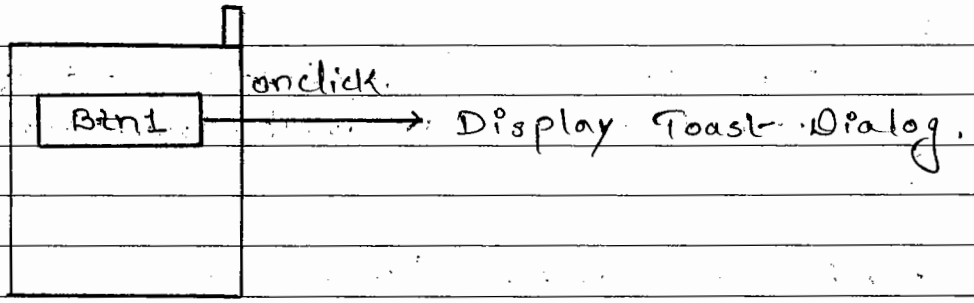
— The android provides a wide range of controls, you can use in your user interface.
Such as -
Buttons, Textfields, seekbars, checkboxes, zoom buttons, toggle Buttons, switch buttons, Image buttons etc.

* Buttons :

- The android is providing different types of buttons, such as - Radio Button, toggle button, compound button etc.

Q. Design an android application with a button. On Button click Display Toast Dialog.

* GW Project-3 *



↳ layout_btn.xml.

```
<LinearLayout

    android : layout_width = "fill_parent"
    android : layout_height = "fill_parent"
    android : orientation = "vertical"
    android : background = "@drawable/mal" >

    <Button

        android : id = "@+id/btn1"
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content"
        android : text = "Click Me"
        android : onclick = "display" />

</LinearLayout>
```

Q.

↳ Activitybtn1.java

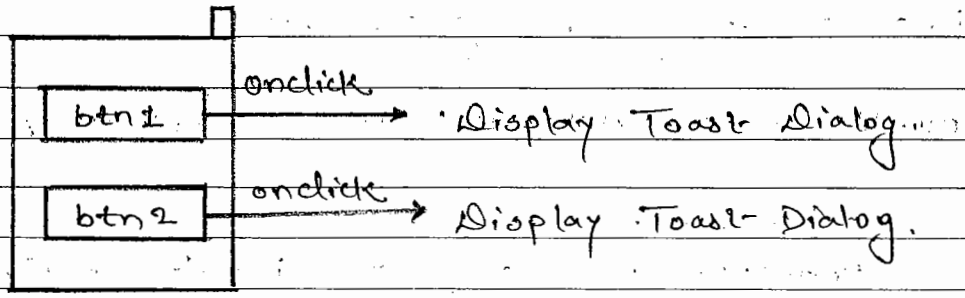
```
import android.app.activity;
import android.os.bundle;
import android.widget.Toast;

public class Activitybtn1 extents Activity
{
    @Override
    protected void oncreate(bundle savedInstancestate)
    {
        Super.oncreate(savedInstance.state);
        setContentView(R.Layout.layoutbtn);
    }

    public void display(View v)
    {
        Toast.makeText(this, "U Clicked Me",
            Toast.LENGTH_LONG).show();
    }
```

↳ Android Manifeast.xml.

```
<application

    <activity
        android:name="com.nmk.cwproject3.Activitybtn1"

        <intent-filter>
            <action
                android:name="android.intent.
                    action.MAIN"/>
            <category
                android:name="android.intent.category.
                    LAUNCHER"/>
        </intent_filter>
    </activity>
</application>.
```

Q. Design an android application with two Buttons.

* CW Project 4 *



NOTE : To import some files (shortcut key is) " shift + ctrl + O "

↳ layout Buttons . xml

```
<LinearLayout
    android : layout_width = " fill_parent"
    android : layout_height = " fill_parent"
    android : orientation = " vertical" >

    <Button
        android : id = "@+id / btn1"
        android : layout_width = " fill_parent"
        android : layout_height = " wrap_content"
        android : text = " Button 1 "
        android : onclick = " show " />

    <Button
        android : id = "@+id / btn2
        android : layout_width = " fill_parent"
        android : layout_height = " wrap_content"
        android : text = " Button2"
        android : onclick = " display " />
</LinearLayout>
```

↳ ActivityButtons.java

```
import android.app.activity;
import android.os.bundle;
import android.widget.Toast;

public class ActivityButtons extends Activity.
{
    @Override
    protected void onCreate (bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layoutButtons);
    }

    public void show ()
    {
        Toast.makeText (this, "In show", Toast.
            LENGTH_LONG).show ();
    }

    public void display ()
    {
        Toast.makeText (this, "In display",
            Toast.LENGTH_LONG).show ();
    }
}
```

↳ Android Manifest.xml

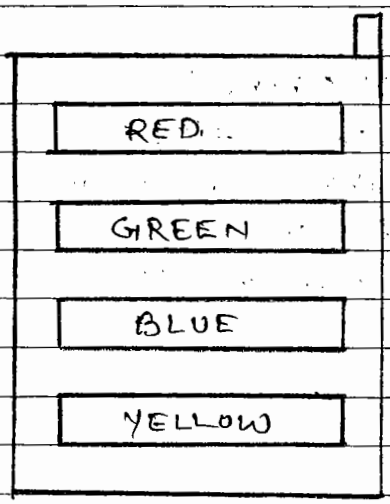
```
<LinearLayout

    android:
```

↳ Android Manifeast .xml

```
< application
    < activity
        android : name = "com.rmk . cwproject4 , ActivityButton
      <intent-filter >
        <action
            android : name = "android . intent . action . MAIN" />
        < category
            android : name = "android . intent : category . LAUNCHER" />
      </intent - filter>.
    </ activity >
</ application >
```

Q. Design an android application to change Layout Background color on Button click.

* CWProject-5 *



↳ layoutchangecolor.xml

```
<LinearLayout

    android:id="@+id/LL1"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"

<Button

    android:id="@+id/btn1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="RED"
    android:onclick="showRed" />

<Button

    android:id="@+id/btn2"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="GREEN"
    android:onclick="showGreen" />

<Button

    android:id="@+id/btn3"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="BLUE"
    android:onclick="showBlue" />
```

```
<Button

    android : id = "@+id/btn4"
    android : layout_width = "fill_parent"
    android : layout_height = "wrap_content"
    android : text = "YELLOW"
    android : onclick = "showYellow" />

</LinearLayout>
```

↳ ActivityColor . java.

```
import android.app.activity ;
import android.os.bundle ;
import android.
public class ActivityColor extends Activity
{
    @Override
    protected void onCreate (bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView ( R.layout.layoutChangeColor);
        ll = (LinearLayout) findViewByID (R.id.LL1);
    }

    LinearLayout ll ;

    public void showRed ()
    {
        ll.setBackgroundColor (Color.Red);
    }

    public void showGreen ()
    {
        ll.setBackgroundColor (Color.GREEN);
    }
```

```
    public void showBlue ()
    {
        ll.setBackgroundColor (Color.BLUE);
    }

    public void showYellow ()
    {
        ll.setBackgroundColor (Color.YELLOW);
    }
}
```

↳ Android Manifeast.xml.

```
< application

    < activity
        android:name = "com.nmk.cwproject5.ActivityColor" >

        < intent-filter >

            < action
                android:name = "android.intent.action.
                    MAIN" />

            < category
                android:name = "android.intent.
                    category.LAUNCHER" />

        </ intent-filter >

    </ activity >

</ application.>
```

* TextView :

- The TextView component is used to display text to the user on User Interface.

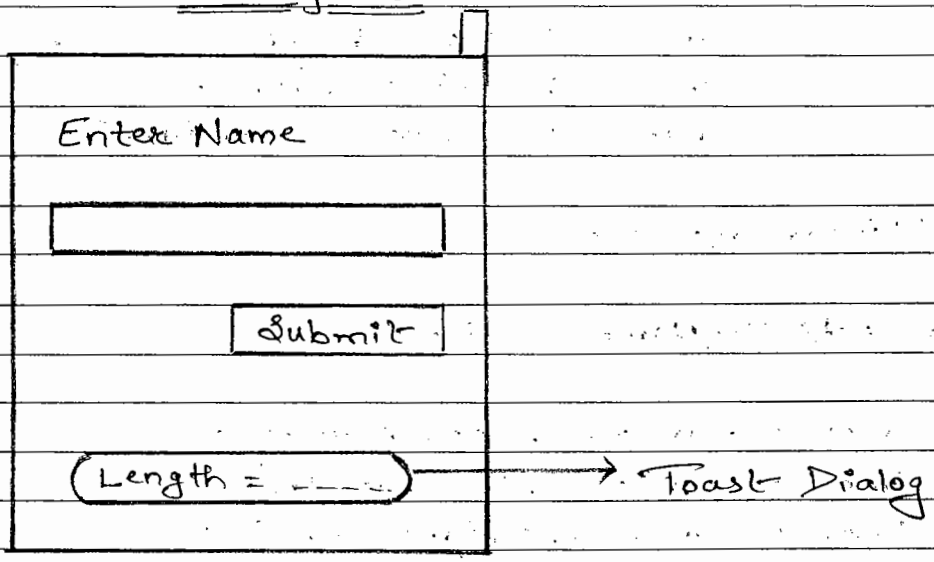
* EditText :

- EditText is used to allow a user to enter some text.

NOTE : EditText is nothing but a text view which is editable.

Q. Design an android application to read UserName and display the Name Length.

* CWProject6 *



↳ Layout_Name : xml :

```
<LinearLayout
    android : id = "@+id / LL1"
    android : layout_width = " fill_parent"
    android : layout_height = " fill_parent"
    android : orientation = " vertical" >
```

```
<TextView

    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : text = "Enter Name" />

<EditText

    android : id = "@+id/ Edt1"
    android : layout_width = "fill_parent"
    android : layout_height = "wrap_content"
    android
<Button

    android : id = "@+id / Btn1"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : text = "Submit"
    android : onclick = "BtnSubmit" />

</LinearLayout>
```

→ ActivityName . java

```
import android.app.Activity;
import android.os.bundle;
import android.widget.Toast;

public class ActivityName extends Activity
{
    @Override
    public void oncreate(bundle savedInstanceState)
    {
        Super.oncreate(savedInstanceState);
        setContentView(R.Layout.layout_Name);
    }
```

```
public void BtnSubmit (View v)
{
    EditText et = (EditText) findViewById (R.id.Edt1);
    String name = et.getText().toString();
    int len = name.length();
    Toast.makeText(this, "Length = "+len, 3000).show();
}
}
```

↳ Android Manifeast : xml

```
<application
    <activity
        android : name = "com.nmk.CWProject6.ActivityName" >
        <intent_filter>
            <action
                : android:name = "android.intent.action.Main" />
            <category
                android:name = "android.intent.category.LAUNCHER" />
        </intent_filter>
    </activity>
</application>
```

* Specifying the Keyboard Type :

— EditText can have different input types such as number, date, password, email address etc.

— You can specify the type of keyboard you want for your EditText object with the — "android : inputType" attribute.

Ex.

① "text" : Normal text Keyboard.

② "textEmailAddress" : Normal type keyboard with @ Keyboard character.

③ "textUri" : Normal text keyboard with the / character.

④ "number" : Basic Number Keyboard.

⑤ "Phone" : Phone Style Keyboard.

⑥ "textCapSentense : Normal text Keyboard that capitalize the first letter for each new sentance.

⑦ "textCapWords" : Normal text Keyboard that capitalize every words. Good for titles or person Names.

⑧ "textAutoCorrect" : Normal text Keyboard that corrects commonly mispelled words.

⑨ "textPassword" : Normal text Keyboard but the character enter turns into dots.

(10) "textMultiLine" : Normal text keyboard that allows user to input long string of texts that includes line breaks. (carriage returns)

* CWProject-7 *

→ Snippet :

↳ Layout_One . xml

```
<LinearLayout
    android : id = "@+id/LL1"
    android : layout_width = "fill_parent"
    android : layout_height = "fill_parent"
    android : orientation = "Vertical"

<EditText
    android : id = "@+id / Edt1"
    android : layout_width = "fill_parent"
    android : layout_height = "wrap_content"
    android : inputType = "textCapCharacters |
                           textMultiLine |
                           textAutoComplete" />

</LinearLayout>
```

Q. Design an android application for specifying the Keyboard type with EditText.

↳ Layout_One . xml

```
<LinearLayout
    android : layout_width = "fill_parent"
    android : layout_height = "fill_parent"
    android : orientation = "vertical" >

    <EditText

        android : id = "@+id / Edt1"
        android : layout_width = "fill_parent"
        android : layout_height = "wrap_content"
        android : inputType = "textCapCharacter ;
                               textMultiLine ;
                               textAutoComplete" />

</LinearLayout>
```

↳ ActivityOne . java

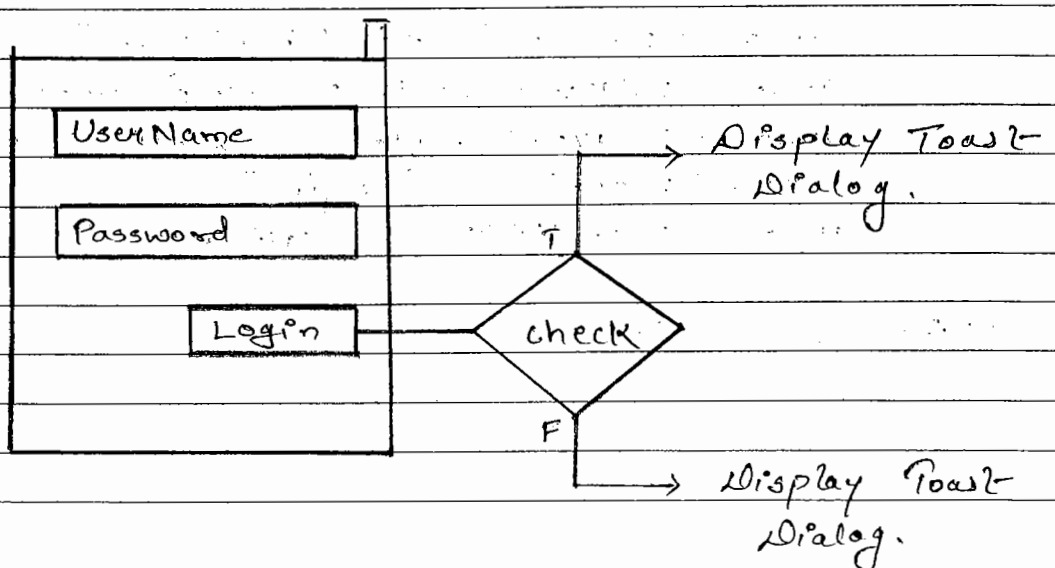
```
import android.app.Activity ;
import android.os.bundle ;
import android.IO ;

public void class ActivityOne extends Activity
{
    @Override
    public void onCreate (bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState) ;
        setContentView (R.layout.layout_One) ;
    }
}
```

↳ Android Manifeast . xml .

```
<application
    <activity
        android:name = "com.nmk.cwproject7.ActivityOne">
        <Intent_filter>
            <action
                android:name = "android.intent.action.
                                MAIN" />.
            <category
                android:name = "android.intent.category.
                                LAUNCHER" />
        </intent_filter>
    </activity>
</application>.
```

Q. Design an Android Application for UserName and Password validation.

* CW Project-8 *



↳ layout_Login . xml

```
< linearLayout
    android : layout_width = " fill_parent"
    android : layout_height = " fill_parent"
    android : orientation = " vertical " >

  <EditText
    android : id = "@+id / Ed UserName "
    android : layout_width = " fill_parent"
    android : layout_height = " wrap_content"
    android : inputType = " text"
    android : hint = " UserName" />

  <EditText
    android : id = "@+id / Ed Password "
    android : layout_width = " fill_parent"
    android : layout_height = "wrap_content"
    android : inputType = " textPassword"
    android : hint = " Password " />

  <Button
    android : id = " @+id / btnLogin "
    android : layout_width = " wrap_content"
    android : layout_height = "wrap_content"
    android : text = " Login "
    android : onClick = " Validate" />

</LinearLayout>
```

↳ ActivityLogin.java

```
import android.app.Activity;
import android.os.bundle;
import android.widget.Toast;

public class ActivityLogin extends Activity
{
    @Override
    protected void onCreate (bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_login);
    }

    public void validate (View v)
    {
        EditText et1 = (EditText) findViewById (R.layout
                        id.etUsername);

        EditText et2 = (EditText) findViewById (R.id.
                        Ed Password);

        String uname = et1.getText().toString();
        String Pass = et2.getText().toString();

        if (Uname.equals ("sathya"))
        {
            if (pass.equals ("students"))
            {
                Toast.makeText (this, "Valid User", 3000).show();
            }
            else {
                Toast.makeText (this, "Invalid Password", 3000).
                        show();
            }
        }
```

```
        else
        {
            Toast.makeText(this, "Invalid User", 3000).
                show();
        }
        et1.setText("");
        et2.setText("");
    }
}
```

↳ Android Manifeast.xml

```
<application

    <activity
        android:name = "com.nmk.cwproject8.ActivityLogin" >

        <intent_filter>

            <action
                android:name = "android.intent.action.MAIN" />

            <category
                android:name = "android.intent.category.
                    LAUNCHER" />

        </intent_filter>

    </activity>

</application>
```

* Switching between the Layouts :

- In android switching between the layouts is done by using "Intent" class.

- Android intent is a message that is passed between the components such as activities, contained providers, broadcast receivers, services etc.

- In generally, with startActivity() method is used to invoke an activity.

- Android intent are mainly used,
  * Start a service,
  * Launch and Activity,
  * Display a web page,
  * Display list of contacts,
  * Broadcast message,
  * Dial a phone call etc.

- In android Intents are categorised into Two types -

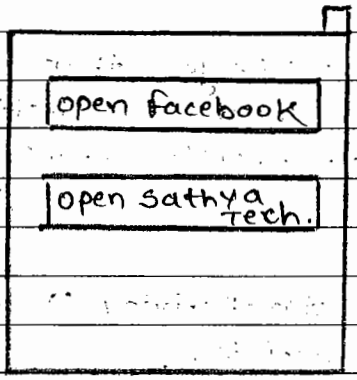
  (1) Implicit Intent,
  (2) Explicit Intent.

(1) Implicit Intent :

- Implicit intent doesn't specify the components. In this case, Intent provides information of available components provided by android systems.

Q. Implicit Intent Example to open browser on button click.

* CWProject 9 *



↳ layout_implicit_intent .xml

```
<LinearLayout

    android : id = "@+id / LL1"
    android : layout_width = "fill_parent"
    android : layout_height = "fill_parent"
    android : orentation = "vertical" >

    <Button

        android : id = "@+id / btnfb"
        android : layout_width = "fill_parent"
        android : layout_height = "wrap_content"
        android : Text = "open facebook"
        android : onclick = "OpenFB" />

    <Button
        android : id = "@+id / btnST"
        android : layout_width = "fill_parent"
        android : layout_height = "wrap_content"
        android : text = "Open Sathya Tech"
        android : onclick = "OpenST" />
</LinearLayout>
```

↳ ActivityImplicityIntent.java

```
import android.app.Activity;
import android.os.bundle;

public class ActivityImplicityIntent extents Activity
{
    @Override
    protected void oncreate (bundle savedInstance_state)
    {
        super.oncreate (savedInstanceState);
        setContentView (R.layout.layout_implicit_intent);
    }

    public void openFB (view v)
    {
        Intent i = new Intent (intent.ACTION_VIEW);
        i.setData (Uri.parse ("http://facebook.com"));
        startActivity (i);
    }

    public void openST (view v)
    {
        Intent i = new Intent (intent.ACTION_VIEW);
        i.setData (Uri.parse ("http://www.sathyaTech.com"));
        startActivity (i);
    }
}
```

↳ Android Manifeast.xml.

~~<LinearLayout~~.

→

```
<application
    <activity
        android:name = "com.mmk.cwproject9.
                        ActivityImplicitIntent" >
        <intent-filter>
            <action
                android:name = "android.intent.action.
                                MAIN" />
            <category
                android:name = "android.intent.category.
                                LAUNCHER" />
        </intent-filter>
    </activity>
</application>
```

* URI :

- Uri is a predefined abstract class of android.Net package

- Uri stands for "Uniform resource identifiers". It is a string of characters used to identify a name of a resource.

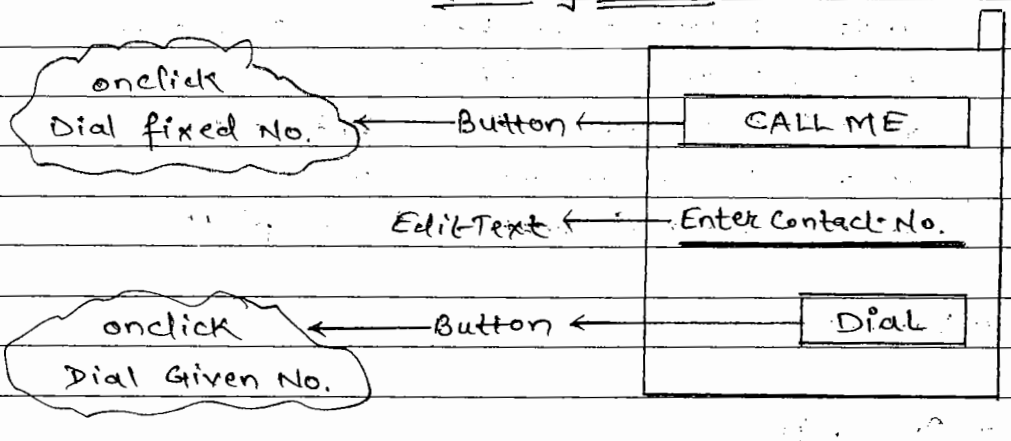
- Parse() is a predefined static method of Uri class, which is used to convert the given string into Uri resource.

Q. Design an android application for Dial a call.

NOTE : To work with call phone we need to take permission in manifeast file.

```
<uses_permission
    android : name = "android.permission.CALL_PHONE" />

<user_permission
    android : name = "android.permission.READ_PHONE_STATE" />
```

* CWProject10 *



↳ layout_dial . xml

```
<RelativeLayout

    android : layout_width = " fill_parent "
    android : layout_height = " fill_parent "

    <Button

        android : layout_width = " fill_parent "
        android : layout_height = " wrap_content "
        android : text = " call Me : 9966887700 "
        android : onClick = " dialfixed " />
```

```
<EditText

    android:id = "@+id/edContactNO"
    android:layout_width = "fill_parent"
    android:layout_height = "wrap_content"
    android:hint = "Enter Contact NO"
    android:layout_marginTop = "60dp" />

<Button

    android:id = "@+id/btnDial"
    android:layout_width = "wrap_content"
    android:layout_height = "wrap_content"
    android:text = "Dial"
    android:onClick = "CallContactNo"
    android:marginTop = "100dp"
    android:marginLeft = "250dp" />

</RelativeLayout>
```

↳ ActivityDial.java

```
import android.app.Activity;
import android.os.Bundle;

public class ActivityDial extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layoutdial);
    }
```

```
public void dialfixed (View v)
{
    Intent i = new Intent (intent.ACTION.CALL);
    i.setData (Uri.parse ("tel : 9984231890"));
    startActivity (i);
}

public void callContactNo (View v)
{
    EditText et = (EditText) findViewById (R.id.
                    edContactNo);

    String number = et.getText().toString();
    number = number.trim();

    Intent i = new Intent (intent.ACTION_CALL);
    i.setData (Uri.parse ("tell : " + ~~no~~), number));
    startActivity (i);
}
}
```

↳ Android Manifeast.xml

```
<uses-permission
    android:name = "android.permission.CALL-PHONE" />

<uses-permission
    android:name = "android.permission.READ_PHONE_STATE" />

e)

<application

    <activity
        android:name = "com.nmk.cwproject10.ActivityDial" >

        <intent-fitter >
```

```
<action
    android:name = "android.intent.action.MAIN" />
<category
    android:name = "android.intent.category.
                    LAUNCHER" />
</intent-filter>
</activity>
</application>
```

Q.

→ (2) Explicit Intent :

— Explicit Intent specifies the component (i.e Activity 2). In such case intent provides the external class to be invoked. The explicit switching can be done in three different ways -

(a) Normal switching.
(b) switching with data.
(c) switching with result data. (Book - 2)

A) Normal Switching :

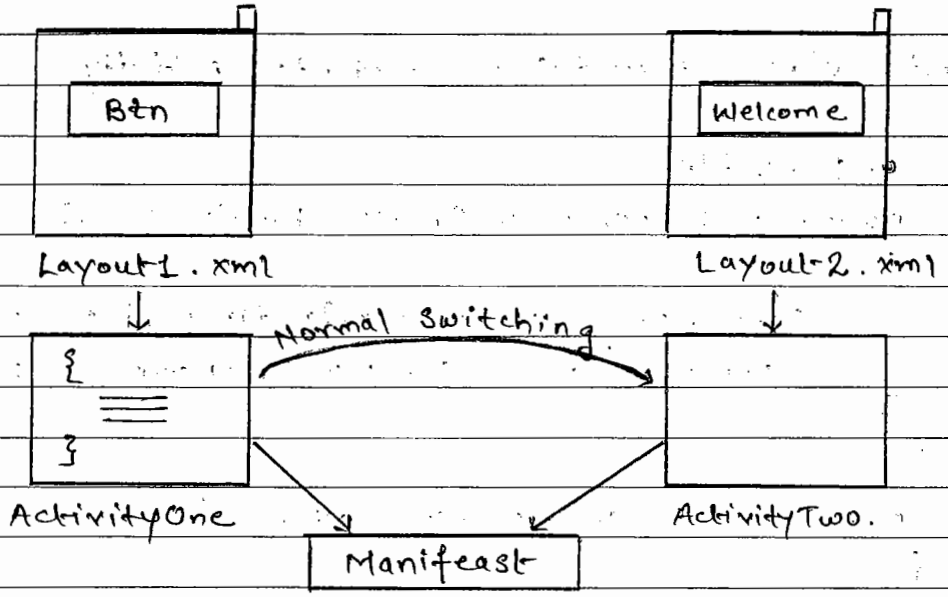
— Normal switching is done in between the two activities without transfering any kind of data.

— Syntax :

```
Intent ref = new Intent (Source activity,
                         destination activity.class);
startActivity (ref);
```

Q. Design an android application to switch in between the activities.

* CW Project-11 *



↳ layoutOne.xml

```
<LinearLayout

    android:layout_width = "fill_parent"
    android:layout_height = "fill_parent"
    android:orientation = "vertical" >

    <Button

        android:layout_width = "fill_parent"
        android:layout_height = "wrap_content"
        android:text = "Open Next Activity"
        android:onclick = "openNext" />

</LinearLayout>
```

↳ ActivityOne.java

```
import android.app.activity;
import android.os.bundle;

public class ActivityOne extends Activity
{
    @Override
    protected void onCreate(bundle savedInstanceState)
    {
        super.oncreate(savedInstanceState);
        setContentView(R.layout.layoutOne);
    }

    public void OpenNext(view v)
    {
        Intent ref = new Intent(this, ActivityTwo.
                                    class);
        startActivity(ref);
    }
}
```

↳ ~~layoutTwo~~ layoutTwo.xml

```
<LinearLayout

    android:layout_width = "fill_parent"
    android:layout_height = "fill_parent"
    android:orientation = "vertical" >

    <TextView

        android:layout_width = "fill_parent"
        android:layout_height = "wrap_content"
        android:text = "Welcome"
        android:layout_margin = "50dp" />
</LinearLayout>
```

↳ ActivityTwo.java

```
import android.app.Activity;
import android.os.Bundle;

public class ActivityTwo extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layoutTwo);
    }
}
```

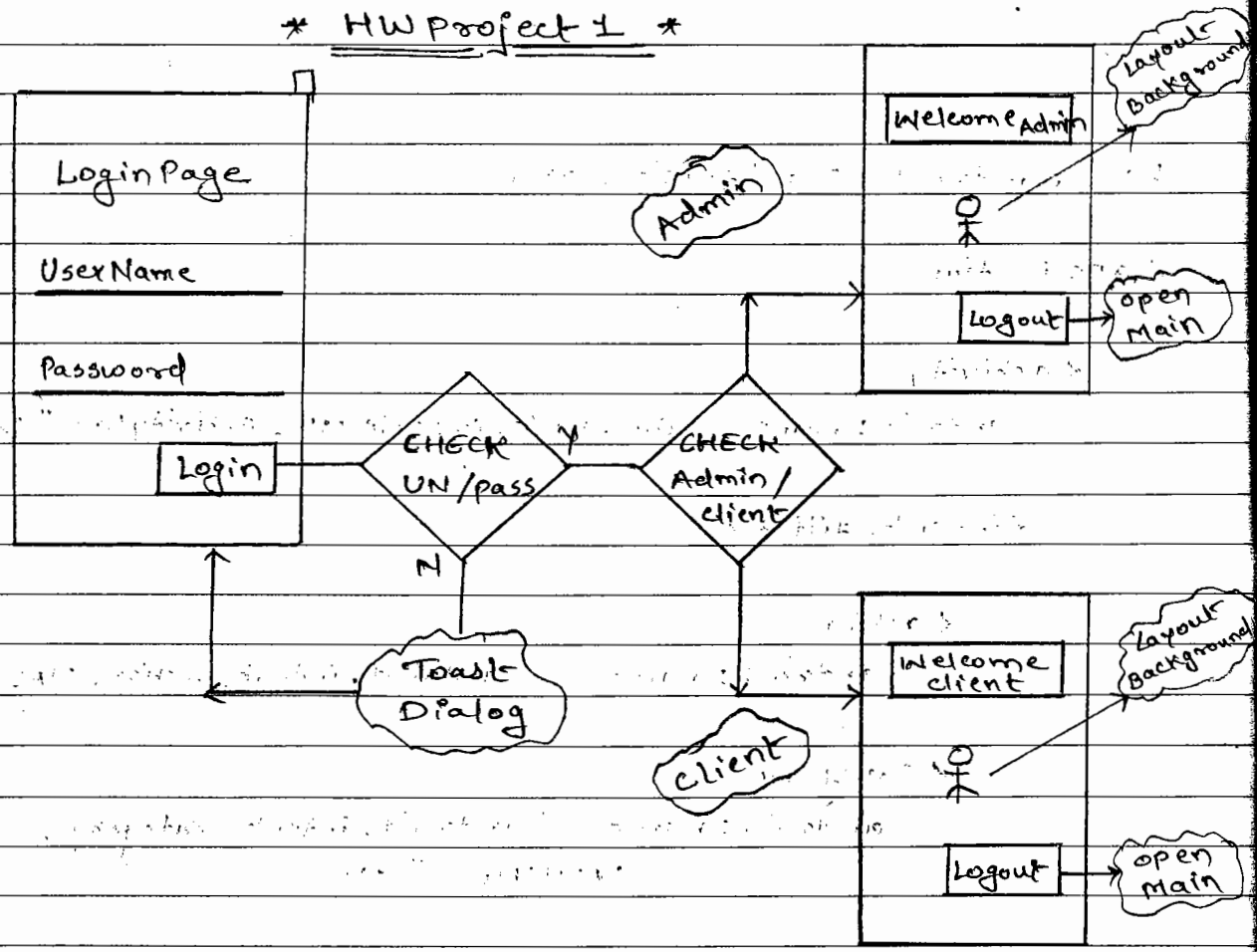
↳ AndroidManifest.xml

```
<application
    <activity
        android:name="com.nmk.cwproject11.ActivityOne">
        <intent-filter>
            <action
                android:name="android.intent.action.MAIN" />
            <category
                android:name="android.intent.category.
                    LAUNCHER" />
        </intent-filter>
    </activity>
```

```
<activity
    android:name = "com.nmk.cwproject-11.
                    ActivityTwo" >

</activity>

</application>
```

Q Design an android application to check the UserName & password, whether they are valid go to admin or client Activity.

* HW Project 1 *



→ NOTE :

* 5 clients must be called with diff UN & Pass.

* 1 Admin must be allowed with one UN & Pass.

↳ layout_login.xml

```
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:background="@drawable/red">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Login Page"
        android:textsize="30dp"
        android:textColor="#ffffff"
        android:layout_marginTop="30dp"
        android:layout_marginLeft="80dp" />

    <EditText
        android:id="@+id/etUserName"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="UserName"
        android:textSize="25dp"
        android:layout_marginTop="30dp"
        android:layout_marginLeft="20dp"
        android:layout_marginRight="20dp" />

    <EditText
        android:id="@+id/etPassword"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:hint="Password"
        android:textSize="25dp"
        android:layout_margin="20dp" />
```

```
<Button
    android : id = "@+id / btnLogin"
    android : layout_width = "fill_parent" wrap_content"
    android : layout_height = "wrap_content"
    android : text = "Login"
    android : textColor = "#ffffff"
    android : textsize = "30dp"
    android : onclick = "Validate"
    android : layout_marginTop = "130dp"
    android : layout_marginLeft = "210dp" />

</LinearLayout>
```

→ ActivityLogin.java.

```
import android.app.Activity;

public class ActivityLogin extends Activity
{
    @Override
    public void onCreate (Bundle savedInstanceState)
    {
        super.oncreate (savedInstanceState);
        setContentView (R.layout.layout_login);
    }

    Intent ref;
    public void validate (View v)
    {
        EditText et1 = (EditText) findViewById (
                        R.id.etUserName);
        EditText et2 = (EditText) findViewById
                        (R.id.etPassword);

        String uname = et1.getText().ToString();
        String pass = et2.getText().toString();
    }
}
```

```
        if (uname.equals("sathya") && pass.equals("Tech"))
        {
            ref = new Intent(this, ActivityAdmin.class);
            startActivity(ref);
        }
        else if (uname.equals("Raunak") && pass.equals("raunak"))
        {
            ref = new Intent(this, ActivityClient.class);
            startActivity(ref);
        }
        else if (uname.equals("Atul") && pass.equals("atul"))
        {
            ref = new Intent(this, ActivityClient.class);
            startActivity(ref);
        }
        else if (uname.equals("Yash") && pass.equals("yash"))
        {
            ref = new Intent(this, ActivityClient.class);
            startActivity(ref);
        }
        else if (uname.equals("Rahul") && Pass.equals("rahul"))
        {
            ref = new Intent(this, ActivityClient.class);
            startActivity(ref);
        }
        else if (uname.equals("Sunil") && pass.equals("sunil"))
        {
            ref = new Intent(this, ActivityClient.class);
            startActivity(ref);
        }
        else
        { Toast.makeText(this, "Invalid User", 3000).show(); }
        et1.setText("");
        et2.setText("");
    }
}
```

↳ layout_admin.xml

```
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:layout_background="@drawable/purple">

  <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Welcome Admin"
      android:textSize="30dp"
      android:textColor="#ffffff"
      android:layout_marginTop="30dp"
      android:layout_marginLeft="55dp" />

  <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:id="@+id/btnAdminLayout"
      android:text="Logout"
      android:textColor="#ffffff"
      android:textSize="20dp"
      android:onClick="AdminLogout"
      android:layout_marginTop="330dp"
      android:layout_marginLeft="200dp" />

</LinearLayout>
```

↳ ActivityAdmin . java .

```
import android.app.Activity;

public class ActivityAdmin extends Activity
{
    @Override
    public void onCreate (Bundle savedInstanceState)
    {
        super.oncreate (savedInstanceState);
        setContentView ( R.layout.layout_admin);
    }

    public void adminLogout (View v)
    {
        Intent ref = new Intent (this, ActivityLogin.class);
        startActivity (ref);
    }
}
```

↳ layout_client . xml

```
<LinearLayout
    android : layout_width = "fill_parent"
    android : layout_height = "fill_parent"
    android : orientation = "vertical"
    android : layout_background = "@drawable/purple" >

    <TextView
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content"
        android : ~~text~~ text = "Welcome client"
        android : textsize = "30dp"
        android : textColor = "#ffffff"
        android : layout_marginTop = "30dp"
        android : layout_marginLeft = "55dp" />
```

```
<Button
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : id = "@+id / btnclientLogout"
    android : text = "Logout"
    android : textcolor = "#ffffff"
    android : text-size = "20 dp"
    android : onClick = "clientLogout"
    android : layout_marginTop = "380dp"
    android : layout_marginLeft = "200 dp" />

</LinearLayout>
```

↳ Activity Client . java

```
import android.app.Activity;

public class ActivityClient extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView ( R.layout.layout_client);
    }

    public void clientLogout (View v)
    {
        Intent ref = new Intent (this, ActivityLogin.class);
        startActivity (ref);
    }
}
```

↳ Android Manifeasl- . xml

```
<application

    <activity
      android : name = " com. example . hwproject1 . Activity2ogin" >

      <intent_filter>

        <action
          android : name = " android . intent . action . MAIN-"/>

        <category
          android : name = " android . intent . category . LAUNCHER "/>

      </ intent_filter>

    </ activity>

</application> <activity
                android : name = " com . example . hwproject-1 .
                                  ActivityAdmin " >
              </ activity >

            < activity
              android : name = " com . example . hwproject-1 .
                                  Activity Client " >
            </ activity >

</ application >
```

By Switching with Data :

- If the switching is done by sending the data to the next activity is called as data with switching.

- To switch an activity with data we use "Bundle" container,

↳ Bundle :

- Bundle is generally used for passing data between various activities of android.

- It depends on you what type of values you want to pass, but Bundle can hold all types of values & pass to the next activity.

i) getIntent () :

It is a predefined method of activities class used to get the intent class object for the source activity.

Ex. Bundle bn = getIntent().getExtras();

* Methods of Intent Class :

- putExtras () :

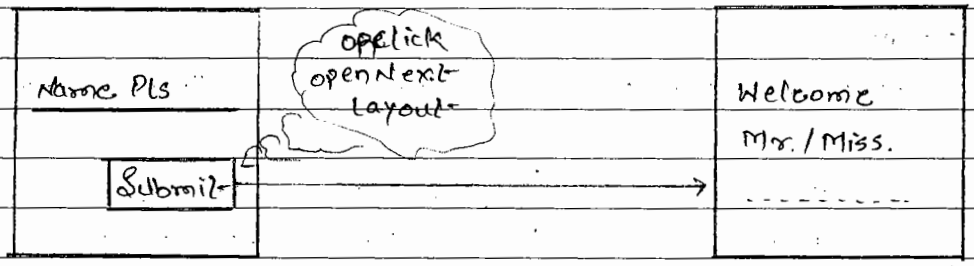
If you want to add information to your intent you can use this method. This information is represented as map (key, value).

- getExtras () :

This information is used to get the added information from intent.

Q. Design an android application to switch data between two activities.

* CWProject-12 *



↳ layout_one.xml

```
<LinearLayout
    android:width = "fill_parent"
    android:height = "fill_parent">

    <EditText
        android:id = "@+id/etName"
        android:layout_width = "fill_parent"
        android:layout_height = "wrap_content"
        android:hint = "Name Pls" />

    <Button
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:Text = "Submit"
        android:onClick = "openNext" />

</LinearLayout>
```

↳ ActivityOne.java

```
import android.app.activity;

public class ActivityOne extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.oncreate (savedInstanceState);
        setContentView (R.layout.layout_One);
    }

    public void openNext (View v)
    {
        EditText et = (EditText) findViewbyid (R.id.etName);

        String name = et.getText().toString();

        Intent i = new Intent (this, ActivityTwo.class);
        i.putExtras ("UN", name);
        startActivity (i);
    }
}
```

↳ layout_two.xml

```
<LinearLayout
    android: layout_width = "fill_parent"
    android: layout_height = "fill_parent"
    android: orientation = "vertical" >

    <TextView
        android: id = "@+id/etWelcome"
        android: layout_width = "wrap_content"
        android: layout_height = "wrap_content" />
</LinearLayout>
```

~~android : text = "~~

↳ ActivityTwo . java

```
import android.app.Activity;

public class ActivityTwo extends Activity
{
    @override
    public void oncreate (Bundle savedInstanceState)
    {
        super.Oncreate (savedInstanceState);
        setContentView ( R. layout. layout_two);

        Intent i = getIntent ();

        Bundle bn = i.getExtras ();

        String name = bn.getString ("UN");

        TextView tv = (TextView) findViewbyId (R.id.tvwelcome);

        tv. setText ("welcome Mr. / Miss. : " +name);

    }
}
```

↳ Android Manifeast . xml.

```
< application

    < activity
        android:name = " com.example.cwproject12. ActivityOne ">;

        <intent_filter >
```

```
<action
    android:name = "android.intent.action.MAIN" />

<category
    android:name = "android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

<activity
    android:name = "com.example.cwproject12.ActivityTwo" />
</activity>

</application>
```

* Android Toggle Button :

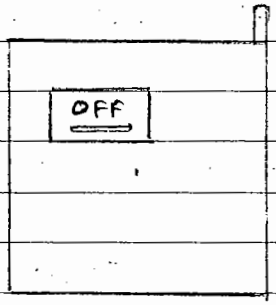
- The toggle Button is used to display checked and unchecked (on-off state). This buttons used to turn on or off sound, bluetooth, wifi, music.

↳ Methods of Toggle Button :

Methods	Description
① charSequenceGetTextOff()	Returns the text with buttons is not in the checked state.
② charSequenceGetTextOn()	Returns the text with buttons is in the checked state.

(3) void setChecked (boolean checked) — changes the checked state of this button.

Q. Design an Android application –



— * CWProject-14 *

↳ layout_toggle.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:paddingBottom = "@dimen/activity_vertical_margin"
    android:paddingLeft = "@dimen/activity_horizontal_margin"
    android:paddingRight = "@dimen/activity_horizontal_margin"
    android:paddingTop = "@dimen/activity_vertical_margin" >

    <ToggleButton
        android:id = "@+id/tgb1"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:onClick = "display" />

</RelativeLayout>
```

→

↳ ActivityToggle.java

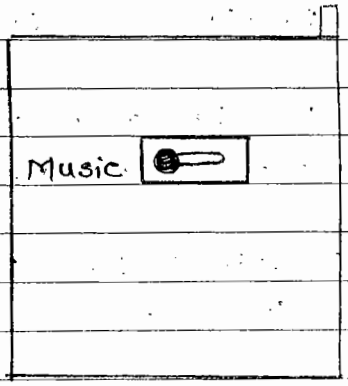
```
import android.app.Activity;

public class ActivityToggle extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        Super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_toggle);
    }

    public void display(View v)
    {
        ToggleButton tb = (ToggleButton) findViewById
                (R.id.tgb1);

        // tb.setChecked(true);

        if(tb.isChecked())
        {
            Toast.makeText(this, "ON", 3000).show();
        }
        else
        {
            Toast.makeText(this, "OFF", 3000).show();
        }
    }
}
```

↳ Android Manifest.xml

```
<application
    <activity
        android:name = "com.example.cwproject14.ActivityToggle">
        <intent_filter>
            <action
                android:name = "android.intent.action.MAIN" />
            <category
                android:name = "android.intent.category.
                    LAUNCHER" />
        </intent_filter>
    </activity>
</application>.
```

* Android Switch Button :

Q. Design and android application -



* cwproject-15 *

↳ layout_switch.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:paddingBottom = "@dimen/activity_vertical_margin"
    android:paddingLeft = "@dimen/activity_horizontal_margin"
    android:paddingRight = "@dimen/activity_horizontal_margin"
    android:paddingTop = "@dimen/activity_vertical_margin" >

    <Switch
        android:id = "@+id/sb1"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Music"
        android:onClick = "change" />

</RelativeLayout>
```

NOTE: In Android Manifest file change -

```
Android:minSDKVersion = "14";
```

↳ ActivitySwitch.java

```
import android.app.Activity;

public class ActivitySwitch extends Activity
{   Switch sw;
    @Override
    protected onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_switch);
    }
        sw = (Switch) findViewById (R.id.sb1);
        sw.setChecked (true);
}
```

```
public void change (View v)
{
    if (sw.isChecked())
    {
        Toast.makeText(this, "ON", 3000).show();
    }
    else
    {
        Toast.makeText(this, "OFF", 3000).show();
    }
}
```

↳ Android Manifest.xml

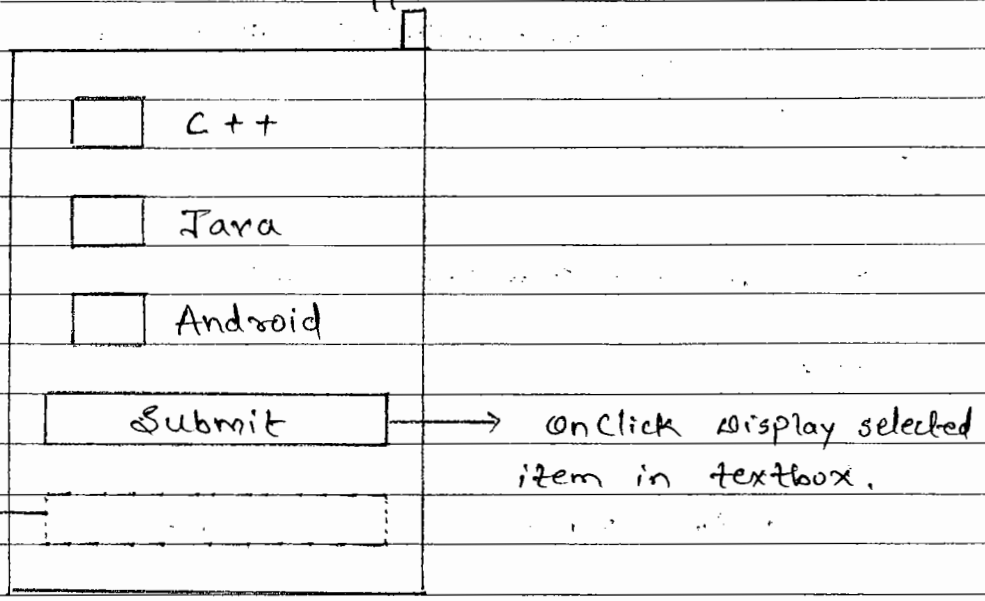
```
<application
    <activity
        android:name="com.ActivitySwitch"
    <intent-filter>
        <action
            android:name="android.intent.action.MAIN" />
        <category
            android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
</application>
```

* Android CheckBox :

— CheckBox allows the user to select one or more options from a set.

* cwproject16 *

Q. Design an Android Application.



↳ layout_checkbox.xml

```
<LinearLayout
    android:id = "@+id/LL1"
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:orientation = "vertical"
    android:gravity = "center" >

    <CheckBox
        android:id = "@+id/cb1"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "C pp"
        android:textsize = "20 dp"
```

```
        android : textColor = "#FF8811"
        android : textStyle = "italic"
or      android : layout_margin = "20 dp" />

    <CheckBox
        android : id = "@+id / cb2"
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content"
        android : text = "Java"
        android : textSize = "20 dp"
        android : textColor = "#FF8811"
        android : textStyle = "italic"
        android : layout_margin = "20 dp" />

    <checkBox
        android : id = "@+id / cb3"
ected   android : layout_width = "wrap_content"
        android : layout_height = "wrap_content"
        android : text = "Android"
        android : textSize = "20 dp"
        android : textColor = "#FF8811"
        android : ~~text col~~ textStyle = "italic"
        android : layout_margin = "20 dp" />

    <Button
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content"
        android : text = "Submit"
        android : onClick = "display"
        android : layout_margin = "20 dp" />

    < TextView
        android : id = "@+id / tv1"
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content" />

</LinearLayout>
```

↳ ActivityCheckBox.java

```
import android.app.Activity;

public class ActivityCheckBox extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_CheckBox);
    }

    public void display(View v)
    {
        String text = "";

        CheckBox cb1, cb2, cb3;
        cb1 = (CheckBox) findViewById(R.id.cb1);
        cb2 = (CheckBox) findViewById(R.id.cb2);
        cb3 = (CheckBox) findViewById(R.id.cb3);

        if (cb1.isChecked())
        {
            text = text + cb1.getText().toString();
        }
        if (cb2.isChecked())
        {
            text = text + cb2.getText().toString();
        }
        if (cb3.isChecked())
        {
            text = text + cb3.getText().toString();
        }
        TextView tv = (TextView) findViewById(R.id.tv1);
        tv.setText(text);
    }
}
```

↳ Android Manifeast. xml

```
< application
    < activity
        android : name = ". ActivityCheckBox"
        <intent-filter>
            < action
                android:name = "android.intent.action.MAIN" />
            < category
                android : name = "android.intent.category.
                    LAUNCHER" />
        </intent-filter>
    </activity>
</application>
```

* Dialog :

— A dialog is a small window that prompts the user to make a decision or enter additional information.

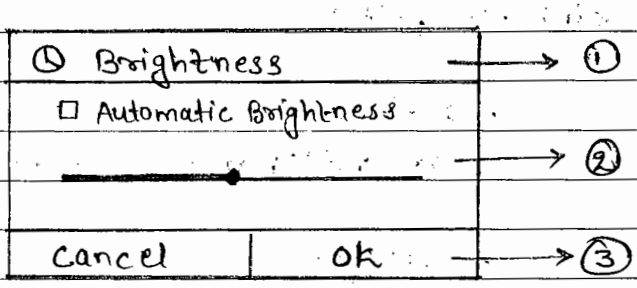
↳ In Dialogs there are 5 types :

① Toast Dialog.
② Alert Dialog.
③ Date Picker Dialog.
④ Time Picker Dialog.
⑤ Progress Dialog Box.

(2) Alert Dialog :

- Alert Dialog is a subclass of "Dialog", that can display a title, message with zero, one, two, three Buttons.

- Diagram :



- The alert dialog contains Three major Regions :

(1) Title : This is optional and this region is used to provide title to alert dialog.

(2) Content Area : This can display a message, a list or other custom layout.

(3) Action Buttons : There should be not more then three Buttons in a dialog or a dialog can be displayed without buttons also.

- Alert Dialog Methods :

(1) setTitle () : setTitle () is used for set-Title to alert dialog.

(2) setMessage () : setMessage () is used for setting messages to alert dialog.

③ setIcon() : setIcon() is used for set-dialog Icon to Alert Dialog.

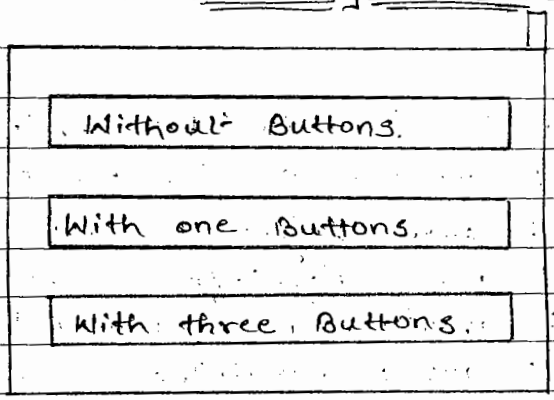
④ setPositionButton() : setPositionButton() is used to create a positive Button to alert dialog.

⑤ setNegativeButton() : setNegativeButton() is used to create a Negative Button to alert dialog.

⑥ setNeutralButton() : setNeutralButton() is used to create a Neutral Cancel Button.

Q. Design an Android Application.

* CWProject-17 *



↳ layout_Alert-dialog.xml.

```
< RelativeLayout
    android : layout_width = " match_parent"
    android : layout_height = " match_parent"
    android : paddingBottom = "@dimen / activity_vertical_margin"
    android : paddingLeft = "@dimen / activity_horizontal_margin"
    android : paddingRight = "@dimen / activity_horizontal_margin"
    android : paddingTop = "@dimen / activity_vertical_margin" >
```

```
<Button
    android:id="@+id/btnsubmit"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_text="Alert Dialog without
                         Buttons"
    android:onClick="displayOne" />

<Button
    android:id="@+id/btnsubmit1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/btnsubmit"
    android:layout_below="@+id/btnsubmit"
    android:layout_marginTop:="30dp"
    android:text="Alert Dialog with One Button"
    android:onclick="displayTwo" />

<Button
    android:id="@+id/btnsubmit2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/btnsubmit1"
    android:layout_below="@+id/btnsubmit1"
    android:layout_marginTop="30dp"
    android:text="Alert Dialog with three Dialog"
    android:onClick="displayThree" />

</RelativeLayout>.
```

↳ ActivityDialog.java.

```
import android.app.Activity;

public class ActivityDialog extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_alertdialog);
    }

    public void displayOne (View v)
    {
        AlertDialog.Builder ad = new AlertDialog.Builder (this);
        ad.setIcon (R.drawable.icon_fire);
        ad.setTitle ("Warning");
        ad.setMessage ("Don't Use Mobiles In Class");
        ad.show ();
    }

    public void displayTwo (View v)
    {
        AlertDialog.Builder ad = new AlertDialog.Builder (this);
        ad.setIcon (android.R.drawable.ic_menu_myplaces);
        ad.setTitle ("My Places");
        ad.setMessage ("In My Place Don't Use Mobiles");
        ad.setPositiveButton ("OK", new .onClickListner ()
        {
            @Override
            public void onClick (DialogInterface dialog, int which)
            {
                Toast.makeText (getApplicationContext (), "OK",
                                3000).show ();
            }
        });
```

```
    ad.show();
}

public void displayThree (View v)
{
    AlertDialog.Builder ad = AlertDialog.Builder (this);
    ad.setIcon (android.R.drawable.ic_menu_myplaces);
    ad.setTitle ("My Places");
    ad.setMessage ("In MyPlace, don't Use Mobiles");
    ad.setPositiveButton ("Yes", new onClickListener()
    {
        @Override
        public void onClick (getApplicationContext(), "Yes",
                             3000).show();
    });

    ad.setPositive
    ad.setNegativeButton ("No", new onClickListener()
    {
        @Override
        public void onClick(getApplicationContext(), "NO",
                            3000).show();
    });

    ad.set-

public void displayThree (View v)
{
    AlertDialog.Builder ad = AlertDialog.Builder (this);
    ad.setIcon (android.R.drawable.ic_menu_myplaces);
ad.setTitle ("My Places");
    ad.setMessage ("In My Place, Don't Use Mobiles");

    ad.setPositiveButton ("Yes", new onClickListener()
    {
        @Override
```

```
        public void onClick (DialogInterface dialog , int which)
        {
            Toast.makeText (getApplicationContext (), "Yes",
                                3000).show ();
        }
    });

    ad.setNegativeButton ("No", new onClickListener ()
    {
        @Override
        public void onClick (DialogInterface dialog, int which)
        {
            Toast.makeText (getApplicationContext (), "NO",
                                3000).show ();
        }
    });

    ad.setNeutralButton ("Cancel", new onClickListener ()
    {
        @Override
        public void onClick (DialogInterface dialog, int which)
        {
            Toast.makeText (getApplicationContext (), "Cancel",
                                3000).show ();
        }
    });

    ad.show ();
  }
}
```

↳ Android Manifeast. xml

↳ <application

<activity
android: name = ". ActivityDialog"

<intent_filter>

<action
android: name = "android. intent. action. MAIN" />

<category
android: name = "android. intent. category. LAUNCHER" />

</intent_filter>

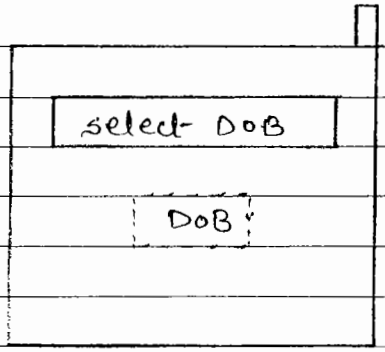
</Activity>

</application>

(3) Date Picker Dialog :

— Date Picker Dialog is a widget to select date. It allows you to ~~create~~ select date by - Day, Month and Year.

Q. Design an Android Application.



* GWProject-18 *

↳ layout_date.xml

```
< RelativeLayout
    android: layout_width = " match_parent "
    android : layout_height = " match_parent "
    android : paddingLeft = "@ dimen / activity_horizontal_margin"
    android : paddingRight = "@ dimen / activity_horizontal_margin"
    android : paddingBottom = "@ dimen / activity_vertical_margin"
AIN"/>  android : paddingTop = "@ dimen / activity_vertical_margin" >

    < Buttom
NCHER"/>    android : id = "@+id / btnDate"
        android : layout_width = " match_parent "
        android : layout_height = "wrap_content"
        android : text = " select DOB"
        android : onClick = " display" />

    < TextView
        android : id = "@+id / tv1"
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content"
date.   android : layout_below = "@+id / btnDate"
        android : layout_marginTop = " 30 dp"
        android : text = " DOB : " />

</ RelativeLayout >
```

↳ ActivityDate.java

```
import android.app.Activity;
import android.os.Bundle;
import java.util.Calender;
```

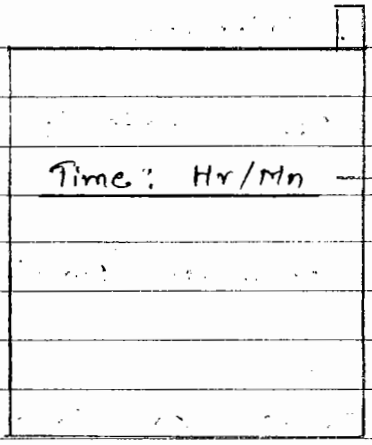
```
public class ActivityDate extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView ( R.layout.layout_date);
    };

    TextView e Tv ;

    public void display ( View v)
    {
        Calender ref = Calender.getIntance ();
        int date = ref.get ( calender. Day_of_Month );
        int month = ref.get ( calender. Month );
        int year = ref.get (Calender.year);

        tv = (TextView) findViewById ( R.id.tv1);

        DatePickerDialog dpd = new DatePickerDialog
                    (this, new onDateSetListener ()
        {
            @Override
            public void onDateSet ( DatePicker view,
                int year, int Month_of_Year, int
                dateofMonth )
            {
                tv.setText ("DOB :" + dateofMonth +
                    "/" + (monthofYear + 1) + "/" +
                    Year);
            }, year, month, date );

            dpd.show ();
        }
}
```

↳ Android Manifeast. xml

```
< application
    < activity
        android : name = ".ActivityDate"
        < intent_filter >
            < action
              : android : name = "android. intent. action. MAIN" />
            < category
              android : name = "android. intent. category. LAUNCHER" />
        </ intent_filter >
    </ activity >
</ application >
```

④ Time Picker Dialog :

Q. Design an Android Application.



* CW Project-19 *

↳ layout_time . xml

```
< RelativeLayout
    android : id = "@+id/RL1"
    android : layout_width = "match_parent"
    android : layout_height = "match_parent"
    android : paddingBottom = "@dimen/activity_vertical_margin"
    android : paddingLeft = "@dimen/activity_horizontal_margin"
    android : paddingRight = "@dimen/activity_horizontal_margin"
    android : paddingTop = "@dimen/activity_vertical_margin" />

    < EditText
        android : id = "@+id/edtime"
        android : layout_width = "match_parent"
        android : layout_height = "wrap_content"
        android : hint = "Time : Hr : Mn" />

</ RelativeLayout >
```

↳ ActivityTime . java

```
import android.app.Activity;
import android.os.Bundle;
import java.util.Calender;

public Class ActivityTime extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        EditText et;
        $uper.onCreate (savedInstanceState);
        setContentView (R.layout.edtime);
                                    layout_time.
```

```
et = (EditText) findViewById (R.id.edtime);

et.setOnClickListener (new onClickListener ()
{
    @Override
    public void onClick (View v)
    {
        Calender ref = calender.getInstance ();
        int hr = ref.get (calender.Hour_Of_Day);
        int min = ref.get (calender.MINUTE);

        getTime (hr, min);
    }
});
}

public void getTime (int hr, int min)
{
    new TimePickerDialog (this, new onTimeSetListener ()
    {
        @Override
        public void onTimeSet (TimePicker view,
                int hourofday, int minute)
        {
            et.setTime
            et.setText (hourofday + ":" + minute);
        }
    }, hr, min, false).show ();
}
}
```

↳ Android Manifest.xml.

< application

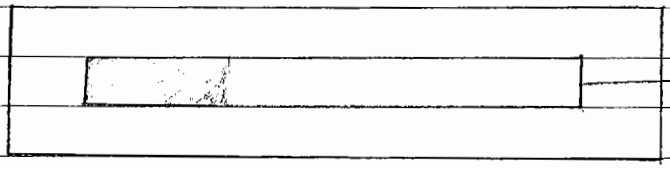
→

```
<activity
    android:name=".ActivityTime">
    <intent-filter>
        <action
            android:name="android.intent.action.MAIN"/>
        <category
            android:name="android.intent.category.
                LAUNCHER"/>
    </intent-filter>
</activity>
</application>
```

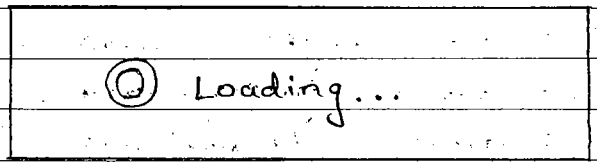
* **Progress Dialog :**

- A dialog showing a progress indicator and optional text message or view.
- Only a text message or a view can be used at the same time.
- The progress dialog can be displayed in two ways -

① STYLE_HORIZONTAL : Creates a progress dialog with a horizontal progress bar.



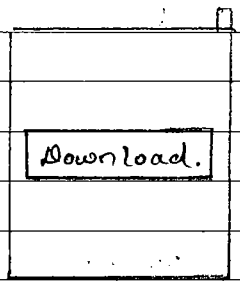
(2). STYLE_SPINNER : creates a progress dialog with circle, spinner progress Bar.



- By default, Progress dialog is Spinner Style.
- To change progress dialog style, we use "setProgressStyle();"

Ex. setProgressStyle (ProgressDialog.STYLE.HORIZONTAL);

Q. Design an Android Application.



* cwproject20 *

↳ layout_progress.xml.

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:paddingBottom = "@dimen/activity_vertical_margin"
    android:paddingLeft = "@dimen/activity_horizontal_margin"
    android:paddingRight = "@dimen/activity_horizontal_margin"
    android:paddingTop = "@dimen/activity_vertical_margin" >
```

→

```
<Button
    android : id = "@+id / btnDownload"
    android : layout_width = "match_parent"
    android : layout_height = "wrap_content"
    android : layout_alignParentBottom = "true"
    android : layout_centerHorizontal = "true"
    android : layout_marginBottom = "82 dp"
    android : layout_marginRight = "28 dp"
    android : onClick = "display"
    android : text = "Download" />

</RelativeLayout>
```

↳ ActivityProgress.java [Example 1]

```
import android.app.Activity;

public class ActivityProgress extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_progress);
    }

    public void display (View v)
    {
        AlertDialog.Builder ad = new AlertDialog.Builder
                                 (this);
        ad.setIcon (R.drawable.ic_action_download);
        ad.setTitle ("Download");
        ad.setMessage ("Yes to Download");
```

→

```
ad.setPositiveButton ("Yes", new onClickListener ()
{
    @Override
    public void onClick (DialogInterface dialog, int which)
    {
        displayProgress ();
    }
});

ad.setNeutralButton ("Cancel", new onClickListener ()
{
    @Override
    public void onClick (DialogInterface dialog, int which)
    {
        display.cancel ();
    }
});
ad.show ();
}

ProgressDialog pd;

public void displayProgress ()
{
    pd. new ProgressDialog (this);
    // pd.setProgressStyle (ProgressDialog.STYLE_HORIZONTAL);
    pd.setMessage ("Please wait under Process");
    pd.show ();
    pd.setCancelable (true);

    new Thread (new Runable ()
    {
        @Override
        public void run ()
        {
```

→

```
try
{
    for (int i=0; i<=10 ; i++)
    {
        Thread.sleep (1000);
        pd.incrementProgressBy (10);
    }
}
catch (Exception ex)
{ }
pd.dismiss ();
}).start ();
}
}.
```

↳ Android Manifeast.xml

```
<application
    <activity
        android:name = " ~~com~~ . .ActivityProgress "
        <intent_filter>
            <action
                android:name = " android.intent.action.MAIN " />
            <category
                android:name = " android.intent.category.
                                LAUNCHER " />
        </intent_filter>
    </activity>
</application>
```

Q. What is the difference between dismiss() and cancel().

— A dialog is dismissed when its job is finished and it is being removed from the screen.

— A dialog is cancelled when the user want to excape the dialog and press the back button.

* Android Spinner :

— Spinner provides a quick way to seled one value from a set.

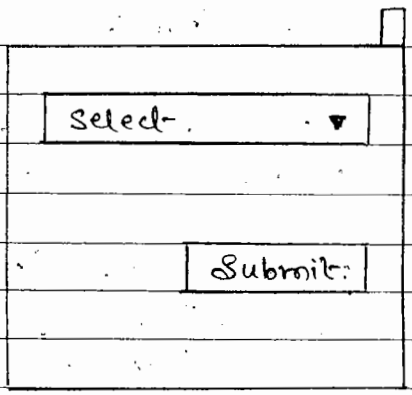
— In the default state the spinner provides currently selected value.

— By Touching a spinner display a dropdown menu with avaible set of items.

Q. Design an Android Application.

* cwproject-21 *

"/>



↳ layout_spinner.xml

```
<RelativeLayout>
    android:id = "@+id/RLL"
    android:layout_width = "match_parent"
    android:layout_height = "wrap_content"
    android:paddingBottom = "@dimen/activity_vertical_margin"
    android:paddingLeft = "@dimen/activity_horizontal_margin"
    android:paddingRight = "@dimen/activity_horizontal_margin"
    android:paddingTop = "@dimen/activity_vertical_margin" />

    <Spinner
        android:id = "@+id/spin1"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content" />

</RelativeLayout>
```

↳ ActivitySpinner.java

```
import android.app.Activity;
import android.os.Bundle;

public class ActivitySpinner extends Activity
{
    Spinner sp;
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_spinner);

        sp = (Spinner) findViewById (R.id.spin1);
        String colours[] = {"RED", "GREEN", "BLUE",
                            "YELLOW"};
```

}
}

```
        ArrayAdapter aa = new ArrayAdapter (this,
                android.R.layout.simple_dropdown_list_item_1
                sp.setAdapter (aa);

    }

    public void display (View v)
    {
        RelativeLayout rl1 = (RelativeLayout) findViewById
                                (R.id.RL1);
        String item = (String) sp.getSelectedItem ();

        if (item.equals (colours[0])
        {
            rl1 : getBackGround Color (Color.RED);
        }
        else
        {
            if (item.equals (colour[1])
            {
                rl1.setBackGround Color (Color.GREEN);
            }
            else
            {
                if (item.equals (colours[2])
                {
                    rl1.setBackGround Color (Color.BLUE);
                }
                else
                {
                    if (item.equals (colours[3])
                    {
                        rl1.setBackGround Color (Color.YELLOW);
                    }
                }
            }
        }
    }
}
```

↳ AndroidManifest.xml

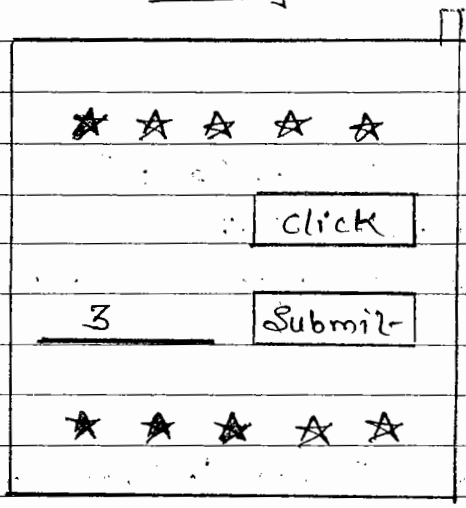
```
<application
    <activity
        android:name = ".ActivitySpinner"
        <intent-filter>
            <action
                android:name = "android.intent.action.MAIN" />
            <category
                android:name = "android.intent.category.
                                LAUNCHER." />
        </intent-filter>
    </activity>
</application>
```

* Rating Bar :

- Android Rating Bar can be used to get the rating item to the user.
- The rating bar returns a float value like 1.5, 2.0, 2.5 etc.
- The getRating() of android Rating Bar class returns the rating Number.
- The setRating (float) method of android Rating Bar class is used to set the Rating.

Q. Design an Android Application:-

* CWProject-22 *



↳ layout_Rating.xml

```
<RelativeLayout
    android:id = "@+id/RL1"
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:paddingBottom = "@dimen/activity_vertical_margin"
    android:paddingLeft = "@dimen/activity_horizontal_margin"
    android:paddingRight = "@dimen/activity_horizontal_margin"
    android:paddingTop = "@dimen/activity_vertical_margin" >

    <RatingBar
        android:id = "@+id/rb1"
        android:width = "wrap_content"
        android:height = "wrap_content"
        android:layout_centerhorizontal = "true"
        android:layout_alignParentTop = "true" />

    <Button
        android:id = "@+id/btnClick"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
```

android : alignParentRight = "true"
android : layout_below = "rb2"
android : text = "click" />
~~android :~~

<EditText
android : id = "@+id/ed1"
android : layout_width = "wrap_content"
android : layout_height = "wrap_content" />
~~android :~~

<Button
android : id = "@+id/~~ed~~ btnSubmit"
android : layout_width = "wrap_content"
android : layout_height = "wrap_content"
android : alignParentRight = "~~right true~~) true"
android : layout_below = "btnclick"
android : text = "" "Submit" />

<RattingBar
android : id = "@+id/rb2"
android : layout_width = "wrap_content"
android : layout_height = "wrap_content"
android : layout_centerHorizontal = "true"
android : layout_below = "btnsubmit" />

</RelativeLayout>

1 → ActivityRating.java

import android.app.Activity;

public class ActivityRating extends Activity
{ RatingBar rb; rb1; Button b1, b2; EditText et;
@Override →

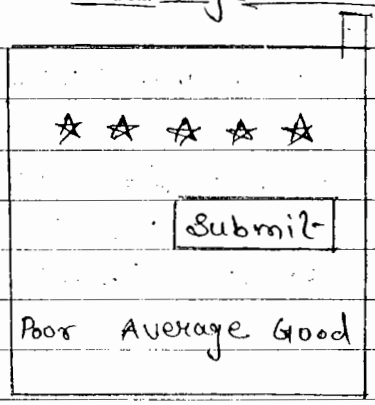
```
protected void onCreate (Bundle savedInstanceState)
{
    Super. onCreate (savedInstanceState);
    setContentView ( R. layout. layout_Rating);

    rb = (RatingBar) findViewById (R.id.rb1);
    b1 = (Button) findViewById ( R. id. btnclick);

    b1. setOnClickListener (new onClickListener ()
    {
        @Override
        public void onClick (View v)
        {
            float rating = rb. getRating ();
            Toast. makeText ( getApplicationContent (),
                " Rating = " , 3000). show ();
        }
    });

    rb1. = (RatingBar) findViewById (R.id. rb2);
    b2 = (Button) findViewById (R. id. btnSubmit);
    et = (EditText) findViewById (R. id. ed1);

    b2. setOnclickListener (new OnClickListener()
    {
        @Override
        public void onClick (View v)
        {
            float rating = float. parseFloat ( et.getText()
                .tostring ());
            rb1. setRating (rating);
        }
    });
}
}
```
t;

↳ Android Manifeast.xml

```
<application
  <activity
    android:name=".ActivityRating"
    <intent_filter>
      <action
        android:name="android.intent.action.MAIN" />
      <category
        android:name="android.intent.category.
                      LAUNCHER" />
    </intent_filter>
  </activity>
</application>
```

Q. Design an Android Application -

* CWproject23 *



* if rating is less then equal 2.5, Display Poor.
* if rating is less then equal 4, Display Average.
* if rating is more than 4, display Good.

↳ layout_rating.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:paddingBottom = "@dimen/activity_vertical_margin"
    android:paddingLeft = "@dimen/activity_horizontal_margin"
    android:paddingRight = "@dimen/activity_vertical_margin">

    <RatingBar
        android:id = "@+id/rb1"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:alignParentRight = "true" />
        ~~android:layo~~

    <Button
        andid:id = "@+id/btnSubmit"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Submit"
        android:onClick = "display" />

    <TextView
        android:id = "@+id/TvPoor"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Poor"
        android:textColor = "#Red" />

    <TextView
        android:id = "@+id/TVAvg"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Average"
        android:textColor = "#Orange" />
```

```
<TextView
    android : id = "@+id / tvGood"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : text = "Good"
    android : textColor = "#Green" />

</RelativeLayout>
```

↳ ActivityRating.java

```
import android.app.Activity;

public class ActivityRating extends Activity
{
    RatingBar rb;
    Button b1;
    TextView tv1, tv2, tv3;

    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_rating);

        rb = (RatingBar) findViewById (R.id.rb1);
        b1 = (Button) findViewById (R.id.btnSubmit);
        tv1 = (TextView) findViewById (R.id.tvPoor);
        tv2 = (TextView) findViewById (R.id.tvAvg);
        tv3 = (TextView) findViewById (R.id.tvGood);
    }

    tv1.setVisibility (TextView.Invisible);
    tv2.setVisibility (TextView.Invisible);
    tv3.setVisibility (TextView.Invisible);
```

```
b1.setOnClickListner (new onClickListener()
{
    @Override
    public void onClick (View v)
    {
        float rating = rb.getRating();

        if (rating <= 2.5)
        {
            tv1.setVisibility (TextView.Visible);
            tv2.setVisibility (TextView.InVisible);
            tv3.setVisibility (TextView.Invisible);
        }
        else if (rating <= 4)
        {
            tv2.setVisibility (TextView.Visible);
            tv1.setVisibility (TextView.Invisible);
            tv3.setVisibility (TextView.Invisible);
        }
        else
        {
            tv3.setVisibility (TextView.Visible);
            tv1.setVisibility (TextView.InVisible);
            tv2.setVisibility (TextView.Invisible);
        }
        rb.setRating (0.0F);
    }
});
}
}
```

↳ Android Manifest.xml

< application

→

```
<activity
    android:name = ".ActivityRating"

    <intent-filter>

        <action
            android:name = "android.intent.action.MAIN" />

        <category
            android:name = "android.intent.category.
                            LAUNCHER" />
    </intent-filter>

</activity>

</application>
```

* Web View :

- webView is a view that displays web page.
- webview is a view that display web pages in your application layout.
- You can also specify HTML strings and show it inside your application by using web view.

→ Methods :

① destroy() — This method destroy the internal state of webview.

② getTitle() — This method return the title of the current page.

③ getUrl() — This method returns the Url of the current page.

④ loadUrl() — This method load the Url into the webView.

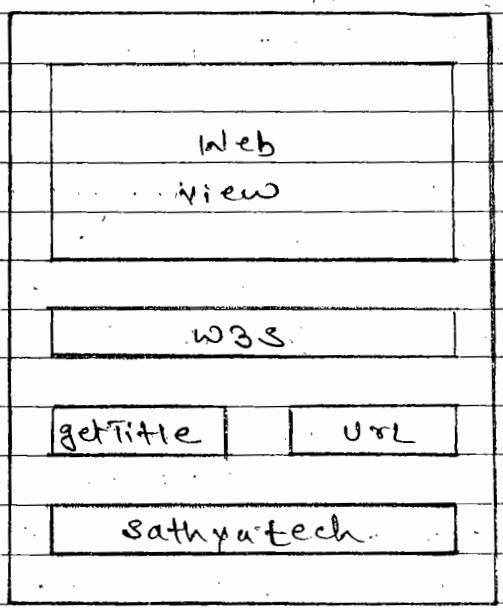
NOTE :

— The webview component is using internet services of your device, so it is mandetory to take internet permission.

```
<User_permission
    android:name = "android.permission.Internet" />
```

Q. Design an Android Application.

* cwProject-24 *



↳ layout_web.xml

```
< RelativeLayout                    match_parent
    android:layout_width = "wrap_content"
    android:layout_height = "match_parent" >

    < WebView
        android:id = "@+id/wv1"
        android:layout_width = "wrap_content"
        android:layout_height = "250dp" />

    < Button  android:onClick = "displayW3S"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "W3S"
        android: android:id = "@+id/btnW3S" />

    < Button  android:onClick = "displayST"
        android:id = "@+id/btnST"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Sathya Tech" />

    < Button  android:onClick = "displayTitle"
        android:id = "@+id/btngetTitle"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "getTitle" />

    < Button  android:onClick = "displayUrl"
        android:id = "@+id/btnUrl"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Url" />

</RelativeLayout >
```

↳ ActivityWeb.java [Example 1]

```
import android.app.Activity;

public class ActivityWeb extends Activity
{
    WebView wv;
    Button b1, b2, b3, b4;

    @Override
    protected void onCreate(Bundle)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout-web);

        wv = (WebView) findViewById(R.id.wv1);
        b1 = (Button) findViewById(R.id.btnW3S);
        b2 = (Button) findViewById(R.id.btnST);
        b3 = (Button) findViewById(R.id.btngetTitle);
        b4 = (Button) findViewById(R.id.btnUrl);

        b1.setOnClickListener(new onClickListener()
        {
            @Override
            public void onClick(View v)
            {
                wv.loadUrl("http://www.w3schools.com");
            }
        });

        b2.g.setOnClickListener(new onClickListener()
        {
            @Override
            public void onClick(View v)
            {
                wv.loadUrl("http://sathyaTech.com");
            }
        });
```

```
b3. setOnClickListener (new onClickListener ()
{
    @Override
    public void onClick (View v)
    {
        Toast.makeText (getApplicationContext (),
            wv.getTitle (), 3000).show ();
    }
});

b4. setOnclickListener (new onClickListener ()
{
    @Override
    public void onClick (View v)
    {
        Toast.makeText (getApplicationContext (),
            wv.getUrl (), 3000).show ();
    }
});
}
}
```

↳ Android Manifest.xml

```
<user_permission android:name = "android.permission.INTERNET" />
<application

  <activity
    android:name = ".ActivityWeb"

    <intent_filter>

      <action
        android:name = "android.intent.action.MAIN" />
      <category
        android:name = "android.intent.category.LAUNCHER" />
    </intent_filter>
  </activity>
</application>
```

↳ ActivityWeb.java

```
import android.app.Activity;

public class ActivityWeb extends Activity
{   WebView WV; Button b1, b2, b3, b4;
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout-web);
        wv = (WebView) findViewById (R.id.wv1);
    }

    public void displayW3s (View v)
    {
        WV = loadUrl ("http://www.w3schools.com");
    }

    public void displayST (View v)
    {
        wv = loadUrl ("http://SathyaTech.com");
    }

    public void displayTitle (View v)
    {
        Toast.makeText (getApplicationContext(),
                wv.getTitle(), 3000).show();
    }

    public void displayUrl (View v)
    {
        Toast.makeText (getApplicationContext(),
                wv.getUrl(), 3000).show();
    }
}
```

Q. Example On HTML String in WebView :

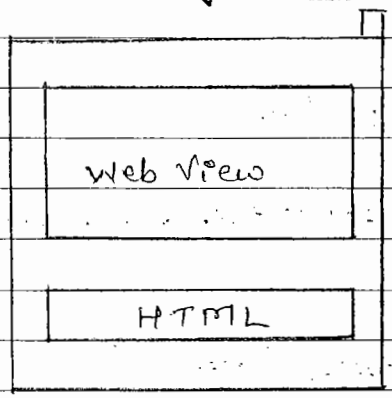
→ loadData (String data , String mimeType , String encoding);

↳ mime : Stands for multipurpose Internet Mail Extensions.

↳ Encoding is UTF :
(U : Stands for Universal coded character Set + T Stands for Transformation format + 8 Bit)

Q. Design An Android Application :

* CWProject24 *



— layout_html : xml

```
<RelativeLayout
    android : layout_width = "match_parent"
    android : layout_height = "match_parent"

    <Button
        android : onClick = "display"
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content"
        android : text = "HTML" />
</RelativeLayout>
```

↳ Activityhtml.java

```
import- android.app.Activity

public class Activityhtml extends Activity
{   WebView wv;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        Super.onCreate(savedInstanceState);
        setContentView(R.layout.layouthtml);
    }

    public void displayHTML(View v)
    {
        String data = "<HTML><TITLE> Sathya </ITITLE>
                      <BODY bgcolor = red>" + "<FONT size=5
                      color = Yellow> Sathya Tech" +
                      "</FONT></BODY></HTML>";

        wv.loadData(data, "text/html", "UTF-8");
    }
}
```

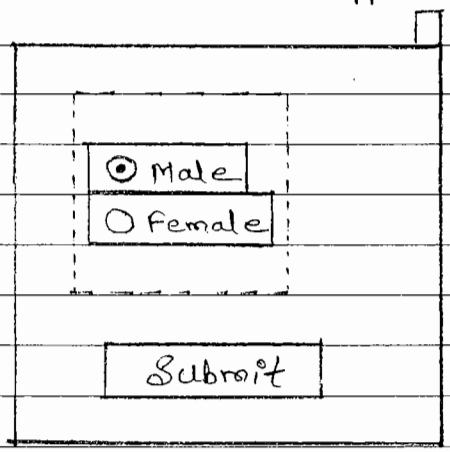
↳ Android Manifeast.xml

```
<User-permission android:name = "android.permission.INTERNET"
                                                    />

<application
    <activity android:name = ".Activityhtml"
    <intent_filter>
        <action android:name = "android.intent.action.MAIN"/>
        <category android:name = "android.intent.category.LAUNCHER"/>
    </intent_filter>
  </activity>
</application>.
```

* Radio Buttons :

- Radio Buttons allows the user to select one option from a set.
- Android is providing Radio Buttons Group and Radio Button.
- Radio Button Group is used to group more then one Radio Buttons.

Q. Design an Android Application -



* CWproject25 *

↳ layout_radio.xml

```
< RelativeLayout
    android : ~~name~~ = layout_width = "match_parent"
    android : layout_height = "match_parent" >

    < RadioButton Group
        android : id = "@+id / rbg1"
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content"
        android : orientation = "horizontal" >
```

```
<RadioButton
    android : id = "@+id / rb1"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : text = "Male" />

<Radio Button
    android : id = "@+id/ rb2"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : text = "Female" />

</Radio Group>

<Button
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : onClick = "Submit"
    android : text = "Submit" />

</RelativeLayout>
```

→ ActivityRadio . java .

```
import android.app.Activity;

public class ActivityRadio extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super . onCreate (savedInstanceState);
        setContentView ( R. layout. layout_radio );
    }
```

→

```
public void submit (View v)
{
    RadioGroup rg = (RadioGroup) findViewById(R.id.rbg1);
    int id = rg.getCheckedRadioButtonId();
    RadioButton rb = (RadioButton) findViewById(id);
    Toast.makeText(this, rb.getText(), 3000).show();
}
}
```

↳ Android Manifest.xml

```
<application
    <activity
        android:name = ".ActivityRadio" >
        <intent_filter>
            <action
                android:name = "android.intent.action.MAIN" />
            <category
                android:name = "android.intent.category.LAUNCHER" />
        </intent_filter>
    </activity>
</application>
```

* Android Image View :

- Image View is an UserInterface that use to display images in your application.

* BITMAP (Bitmap Image) :

↳ Android supports Bitmap files in three formats
- .png (perfered), .jpg (acceptable),
- .gif (discouraged).

* File Location :

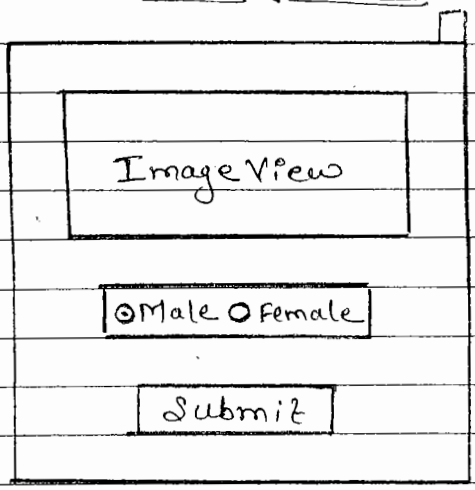
↳ res / drawable / filename.png (.png, .jpg or .gif).
↳ The filename is used as resource Id.

* Resource Reference :

↳ In Java : R.drawable.filename.
↳ In xml : @[package.]drawable/filename.

Q. Design an Android Application.

* CWProject-26 *



1→ layout_image . xml

```
< RelativeLayout
    android : layout_width = " match_parent "
    android : layout_height = " match_parent " >

    < ImageView
        android : id = " @+id / IV1 "
        android : layout_width = " match_parent "
        android : layout_height = " 250 dp "
        android : src = " @drawable / mario " />

    < Button
        android : id = " @+id / ~~Had~~ btnSubmit "
        android : layout_width = " wrap_content "
        android : layout_height = " wrap_content "
        android : centerHorizontal = " true "
        android : onclick = " change "
        android : text = " submit " />

    < RadioGroup
        android : id = " @+id / radGr1 "
        android : layout_width = " wrap_content "
        android : layout_height = " wrap_content "
        android : orientation = " horizontal " >

        < RadioButton
            android : id = " @+id / rad1 "
            android : layout_width = " wrap_content "
            android : layout_height = " wrap_content "
            android : checked = " true "
            android : text = " Mario " />
```

→

```
<RadioButton
    android:id="@+id/rad2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Android" />

</RadioGroup>

</RelativeLayout>
```

↳ ActivityImage.java

```
Import android.app.Activity

public void ActivityImage extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        Super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_image);
    }

    public void change(View v)
    {
        ImageView iv = (ImageView) findViewById(R.id.iv1);
        RadioGroup rg = (RadioGroup) findViewById(R.id.radG1);
        int id = rg.getCheckedRadioButtonId();
        RadioButton rb = (RadioButton) findViewById(id);
        if(rb.getText().equals("Mario"))
        {
            iv.setImageResource(R.drawable.mario);
        }
        else
        {
            iv.setImageResource(R.drawable.img1);
        }
    }
}
```

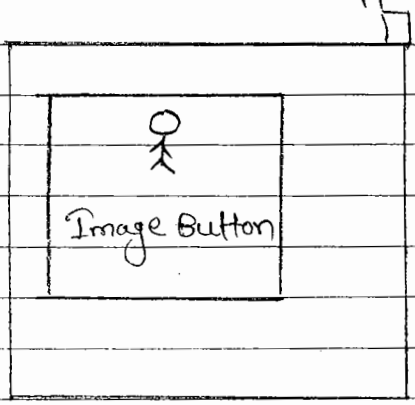
↳ Android Manifeast?. xml

```
<application
    <activity android:name = ". ActivityImage" >
        <intent_filter>
            <action android:name ="android:intent.action.MAIN"/>
            <category
                android:name ="android.intent.category.LAUNCHER" />
        </intent_filter>
    </activity>
</application>
```

* Image Button :

- Image Button is nothing but a push button with image insted of Text.

Q. Design an Android Application -



↳ layout_imagebutton . xml

```
<RelativeLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">

    <ImageButton
        android:id="@+id/ib1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/icon1" />

</RelativeLayout>
```

→ ActivityImageButton.java

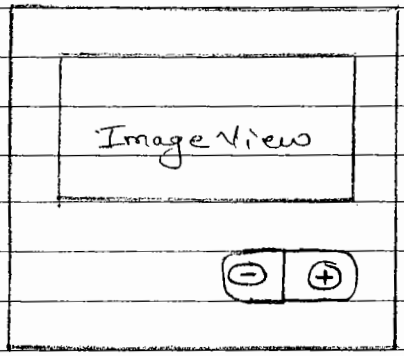
```
import android.app.Activity;

public class ActivityImageButton extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_imagebutton);
    }
}
```

→ AndroidManifest.xml

```
<application
    <activity android:name=".ActivityImageButton"
        <intent-filter>
            <action
                android:name="android.intent.action.MAIN" />
            <category
                android:name="android:intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
```

* Zoom Controls :

Q. Design an Android Application.



* cwproject-27 *

↳ layout_zoom.xml

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:id="@+id/imgview1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/icon1" />

    <ZoomControls
        android:id="@+id/zoomControl"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

↳ ActivityZoom.java

```
import android.app.Activity;

public class ActivityZoom extends Activity
{
    ZoomControls zc;
    ImageView iv;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_zoom);

        iv = (ImageView) findViewById(R.id.imgView1);
        zc = (ZoomControls) findViewById(R.id.zoomcontrol);

        zc.setOnClick
        zc.setOnZoomInClickListener(new onClickListener()
        {
            @Override
            public void onClick(View v)
            {
                float x = iv.getScaleX();
                float y = iv.getScaleY();

                iv.setScaleX(x + 0.2F);
                iv.setScaleY(y + 0.2F);
            }
        });

        zc.setOnZoomOutClickListener(new onClickListener()
        {
            @Override
            public void onCreate(View v)
            {
```

→

```
        float x = iv.getScaleX();
        float y = iv.getScaleY();

        iv.~~get~~setScaleX(x - 0.2f);
        iv.setScaleY(y - 0.2f);
      }
    });
  }
}
```

→ Android Manifest.xml

```
<application
  <activity android:name = ".Activityzoom" >
    <intent-filter>
      <action android:name = "android.intent.action.MAIN" />
      <category android:name = "android.intent.category.LAUNCHER" />
    </intent-filter>
  </activity>
</application>
```

* Android Push Notifications :

- A notification is a message you can display to the user outside of your application normal user Interface.

- Android provides "NotificationManager" class for notifications.

- In order to use this class you need to instantiate an object of this class by requesting the android system through - "getSystemService()"

Ex.
```
NotificationManager NM = (NotificationManager)
    getSystemService (Context.NOTIFICATION_SERVICE);
```

- Then after we need to create "Notification" through Notification class and specifying it attributes such as - Icon, Title, and Time.

Ex.
```
Notification notify = new Notification (andriod.R.drawable.
stat_notify_more, title, System.currentTimeMillis());
```

- Then after we need to create "PendingIntent" object by passing context and Intent as a parameter.

Ex.
```
PendingIntent pending = PendingIntent.getActivity (
    getApplicationContext(), 0, new Intent(), 0);
```

- Then after we need to call "setLatestEventInfo()" method of Notification class and pass the pendingIntent along with notification subject, body details.

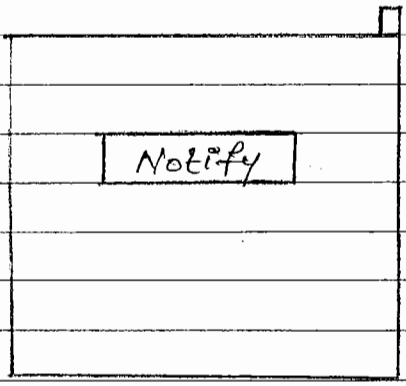
Ex.
```
notify.setLatestEventInfo (getApplicationContext(), subject,
    body, pending);
```

- And then finally call the ~~notification~~ notify() of the "NotificationManager" class.

```
NM.notify(0, notify);
```

Q. Design an android application -

* CWProject-28 *



↳ layout_notify.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <Button
        android:layout_width = "~~calldisplay~~ wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Notify"
        android:onClick = "callDisplay" />

</RelativeLayout>
```

↳ ActivityNotify.java

```
import android.app.Activity;
```

()

```
public ~~void~~ class ActivityNotify extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super. onCreate (savedInstanceState);
        set-contentView (R. layout. layout_notify);
    }

    public void callDisplay (View v)
    {
        NotificationManager NM = (NotificationManager)
            getSystemService (this. NOTIFICATION_SERVICE);

        Notification notify = new Notification (R. drawable.
            icon_name, "Sample Notification", System.
                currentTimeMillis ());

        PendingIntent pi = PendingIntent. getActivity
            (this, 0, new Intent (), 0);

        notify. setLatestEventInfo (this, "Notification",
            "A Sample Test Notification", pi);

        NM ~~notify~~. notify (0, notify);
    }
}
```

↳ Android Manifest . xml

```
< application

    < activity
        android : name = ". ActivityNotify" >
```

```
<intent-filter>

    <action
        android:name="android.intent.action.MAIN" />

    <category
        android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

</application>
```

Q. Example to open a layout on Notification click.

↳ layout_notify.xml

* NOTE : Same as CWProject28.

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Notify"
        android:onClick="callDisplay" />

</RelativeLayout>
```

↳ layout_next . xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <ImageView
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:src = "@drawable/duck" />

</RelativeLayout>
```

↳ ActivityNotify . java

```
import android.app.Activity;

public class ActivityNotify extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        NotificationManager NM = (NotificationManager)
            getSystemService (this.NOTIFICATION_SERVICE);

        Notification notify = new Notification (R.drawable.
        icon_name, "Sample Notification", System.currentTimeMillis
                                                    ());

        PendingIntent pi = PendingIntent.getActivity (this, 0,
            new Intent (), 0);

        notify.setLatestEventInfo (this, "Notification",
            "A Sample Test Notification", pi);

        notify.notify (0, notify);
    }
}
```

↳ Android Manifeast . xml

```
<application
  <activity
    android:name = ". ActivityNotify " >
  <intent_filter >
    <action
      android:name = " android.intent.action. MAIN " />
    <category
      android:name = "android.intent.category. LAUNCHER " />
  <intent_filter>
</activity>
<activity
  android:name = ". ActivityNext " >
</activity >
</application>
```

↳ ActivityNext . java

```
public class ActivityNext extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_next);
    }
}
```

* Multimedia :

- In Mobile technology multimedia represents audio operations, video operations and Image operations.

(1) Audio Operations :

- We can play & control the ~~audio~~ audio files in android with the help of "MediaPlayer" class.

- The mediaplayer class is providing set of predefined methods to do operations on media.

Methods	Description
(1) public void setDataSource()	Set the Data Source, (file path or http url) to use.
(2) public void prepare()	prepare the player for playback syncronously.
(3) public void start()	It starts or resume the playback.
(4) public void stop()	It stops the playback.
(5) public void pause()	It pause the playback.
(6) public boolean isPlaying()	Checks if mediaplayer is playing.
(7) public void seekTo(int Millis)	seek the specified time in milliseconds.

(8) public void setLooping (boolean looping) — Set the player for looping or non-looping.

(9) public boolean isLooping() — checks if the player is looping or non-looping.

(10) public void setVolume (float leftVolume, float rightVolume) — set the volume on this player.

Q. Example Application to play a music in Background.

NOTE : - Create a new folder in 'res' folder and name it as "raw" and copy the music file into created folder.

- The Audio fileName must be in small letters.

* CWProject-29 *

(1) The audio fileName is used as resource Id.

↳ layout_music.xml

```
< RelativeLayout
    android: layout_width = "Match_parent"
    android: layout_height = "match_parent"
    android: background = "#ff8811" >

</ RelativeLayout >
```

looping

is

ing.

this

and

usic

:d.

↳ ActivityMusic.java

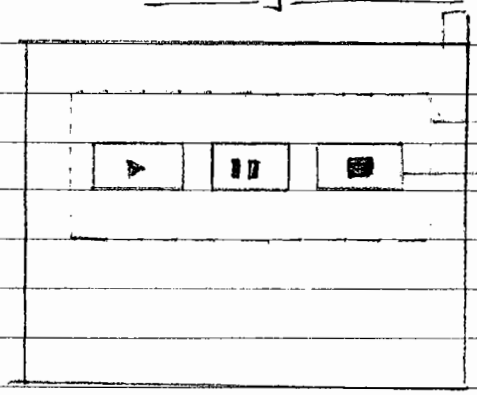
```
import android.app.Activity;

public class ActivityMusic extends Activity
{
    MediaPlayer mp;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(savedInstanceState);

        mp = MediaPlayer.create(this, R.raw.and);
        mp.start();
    }
}
```

↳ Android Manifeast.xml

```
<application

  <activity
    android:name = ".ActivityMusic";

    <intent-filter>

      <action
        android:name = "android.intent.action.MAIN" />

      <category
        android:name = "android.intent.category.LAUNCHER" />

    </intent-filter>

  </activity>
</application>
```

Q. Example program to play, pause and stop a music.

* CWProject-30 *



↳ layout_music.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <LinearLayout
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:layout_margin = "20dp" >

        <ImageButton
            android:layout_width = "wrap_content"
            android:layout_height = "wrap_content"
            android:onClick = "playMusic"
            android:layout_margin = "10dp"
            android:src = "@drawable/playImg" />

        <ImageButton
            android:layout_width = "wrap_content"
            android:layout_height = "wrap_content"
            android:onClick = "pauseMusic"
            android:layout_margin = "10dp"
            android:src = "@drawable/pauseImg" />
```

```
<ImageButton
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : onClick = "stopMusic"
    android : layout_margin = "10 dp"
    android : src = "@drawable / stopImg" />

</LinearLayout>

</RelativeLayout>
```

↳ ActivityMusic.java

```
import android.app.Activity;

public class ActivityMusic extends Activity
{
    Multiplayer mp;
    @Overwrite
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView ( R.layout.layout_music);
    }

    public void playMusic (View v)
    {
        mp= MediaPlayer.create ( R.raw.md);

        try {
            mp.prepare ();
            mp.start ();
            Toast.makeText (this, "Music Start", 3000).show();
            }
        catch {
            Toast.makeText (this, ""+e.message(), 3000).show();
            }
    }
```

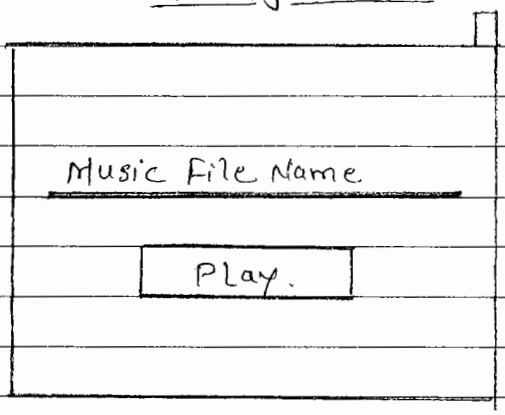
```
public void pauseMusic (View v)
{
    mp.pause();
    Toast.makeText(this, "Music Pause", 3000).show();
}

public void stopMusic (View v)
{
    mp.stop();
    Toast.makeText(this, "Music stop", 3000).show();
}
}
```

→ Android Manifeast.xml.

```
<application

  <activity
  android:name = ".ActivityMusic"

  <intent-filter>

    <action
    android:name = "android.intent.action.MAIN" />

    <category
    android:name = "android.intent.category.LAUNCHER" />

  </intent-filter>

</activity>

</application>
```

Q. Example program to play a music which is Located in sd_card / music folder.

* cwproject-31 *



- Syntax :

↳ mp.setDataSource ("sdcard / folderName / fileName");

(Provides your music file source.)

↳ layout_music.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <EditText
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:id = "@+id/etMusicName"
        android:layout_margin = "20dp"
        android:hint = "Music File Name" />

    <Button
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Play"
        android:onClick = "playMusic" />
</RelativeLayout>
```

↳ ActivityMusic.java

```
import android.app.Activity;

public class ActivityMusic extends Activity
{  MediaPlayer
   ~~Multiplayer~~ mp;
   EditText et;

   @Override
   protected void onCreate(Bundle savedInstanceState)
   {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.layout_music);
   }

   public void playMusic(View v)
   {
      et = (EditText) findViewById(R.id.etMusicName);
      String mname = et.getText().toString();

      try {
         mp.setDataSource("sdCard/Music/mname");
         mp.prepare();
         mp.start();
         }
      catch (Exception ex)
      {
        Toast.makeText(this, ""+e, 3000).show();
      }
   }
}
```

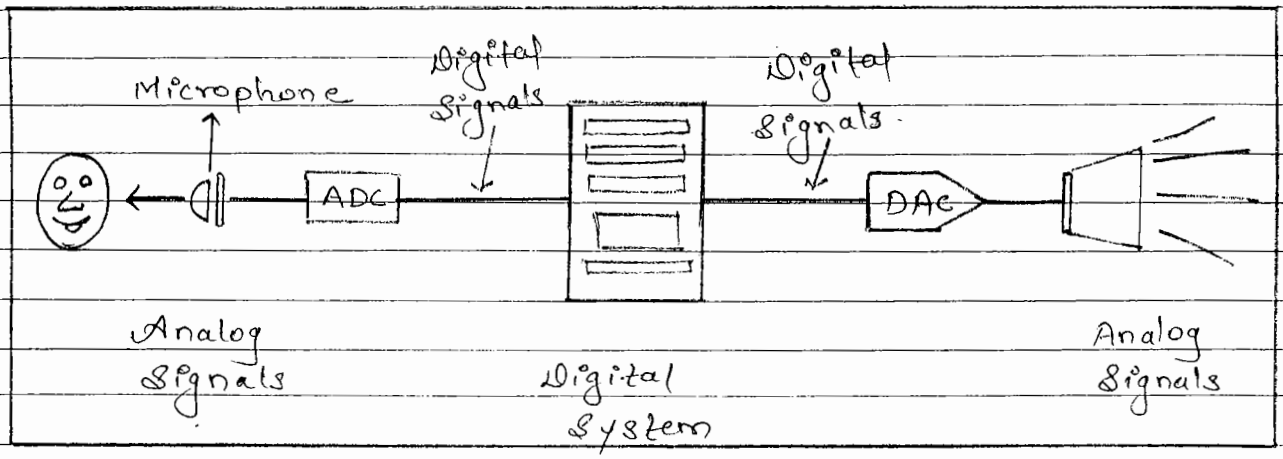
↳ Android Manifeast . xml

```
< application

  < activity
    android : name = " . Activity Music "

    <intent_filter>

      < action
        android : name = "android . intent. action. MAIN " />

      < category
        android : name = "android. intent.. category. LAUNCHER " />

    < </intent_filter>

    </activity>

</application>
```

* Voice Recording and Playing :

- android. media . MediaRecorder class is used to record audio & video files.

	Methods	Description
①	setAudioSource ()	This method specifies the source of audio to be recorded.
②	setVideoSource ()	This method specifies the source of video to be recorded.
③	set-Output-Format ()	This method specifies the audio encoder to be used. Encoding is a format of encoding anlog signals to digital signals.



④ setOutput-file () — This method configures the path to the file into which the recorded audio is to be stored.

⑤ Stop () — This method stops the recording process.

⑥ release () — This method should be called when recorder instance is needed.

* User_Permission :

- <user_permission
  android:name = "android.permission.RECORD_AUDIO" />

- <user_permission
  android:name = "android.permission.WRITE_EXTERNAL_STORAGE" />

— To record a audio, the source is "MIC".

— The audio output format is "THREE_GPP".

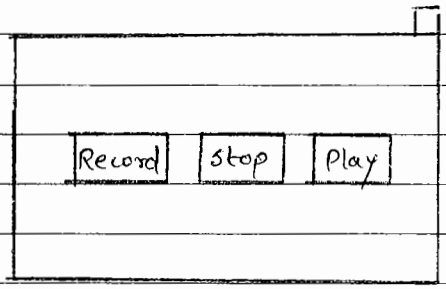
— The audio encoder is AMRNB.

NOTE :

AMRNB → Adaptive Multirate Narrow Band.

Q. Design an android Application.

* CWProject-32 *



↳ NOTE :

- Enviornment. GetExternalStorageDirectory ()

↳ This method returns the primary External Storage Directory.

- getAbsolutePath ()

↳ It is a predefined method of 'File' class which returns the absolute path of specified directory.

↳ layout_record.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <Button
        android:id = "@+id/record"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:onClick = "Record" />
```

```
<Button
    android : id = "@+id/stop"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : onClick = "Stop" />

<Button
    android : id = "@+id/play"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : onClick = "play" />

</RelativeLayout>
```

↳ ActivityRecord.java.

```
import android.app.Activity;

public class ActivityRecord extends Activity
{
    Button b1, b2, b3;
    String outputfile;
    MediaRecorder mr;

    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView ( R.layout.layout_record);

        b1 = (Button) findViewById ( R.id.record);
        b2 = (Button) findViewById (R.id.stop);
        b3 = (Button) findViewById (R.id.play);

        b2.setVisibility (Button.INVISIBLE);
        b3.setVisibility (Button.INVISIBLE);
```

```
outputfile = Enviornment.getExternalStorageDirectory()
             .getAbsolutePath() + "/myrecording";

mr. = new MediaRecorder();

mr.setAudioSource(MediaRecorder.AudioSource.
                  MIC);

mr.setOutputFormat(MediaRecorder.Output-Format.
                   THREE-GPP);

mr.setAudioEncoder(MediaRecorder.outputFormat.
                   AMR_NB);
}

public void record(View v)
{
    b1.setVisibility(Button.InVisible);
    b2.setVisibility(Button.Visible);

    try {
        mr.setOutput-File(output-File);
        mr.prepare();
        mr.start();
        Toast.makeText(this, "started", 3000).show();
    }
    catch(Exception ex)
    {
        Toast.makeText(this, "" + ex, 3000).show();
    }
}

public void stop1(View v)
{
    b2.setVisibility(Button.Invisible);
    b3.setVisibility(Button.Visible);
```

```
        mr.stop();
        mr.release();
        mr = null;
        Toast.makeText(this, "Stopped", 3000).show();
    }

    public void play(View v)
    {
        b3.setVisibility(Button.InVisible);
        b1.setVisibility(Button.Visible);

        try
        {
            MediaPlayer mp = new MediaPlayer();
            mp.setDataSource(outputfile);
            mp.start();
            Toast.makeToast(this, "Play", 3000).show();
        }
        catch(Exception ex)
        {
            Toast.makeText(this, ""+e, 3000).show();
        }
    }
}
```

↳ Android Manifeast.xml

```
<uses-permission android:name="android.permission.
                                RECORD_AUDIO" />
<uses-permission android:name="android.permission.
                                WRITE_EXTERNAL_STORAGE" />

<application
    ---------
    ---------
</application>
```

* Image Capture through Android Device :

- Android.provider.MediaStore class contains Metadata for all avaiable media on both internal & External Storage devices.

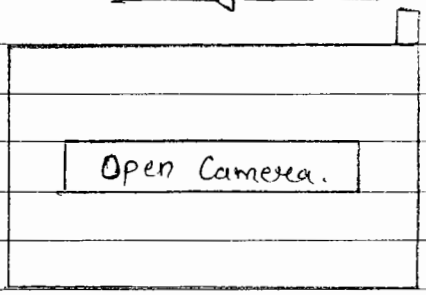
- Snippet- :

```
Intent i = new Intent (MediaStore.ACTION_IMAGE_CAPTURE);
startActivity (i);
```

- It is a standard intent Action that can send to have the camera application capture and image and return it.

Q. Example Program to open camera.

* cwProject-33 *



↳ layout_camera.xml

```
< RelativeLayout
    android : layout_width = "match_parent"
    android : layout_height = "match_parent" >

    <Button
        android : id = "@+id/btnOpenCamera"
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content"
        android : text = "Open Camera"
        android : onClick = "Open" />
</RL>
```

↳ ActivityCamera.java

```
import android.app.Activity;

public class ActivityCamera extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_camera);
    }

    public void open(View v)
    {
        Intent i = new Intent(MediaStore.
                ACTION_IMAGE_CAPTURE);
        startActivity(i);
    }
}
```

↳ Android Manifeast.xml

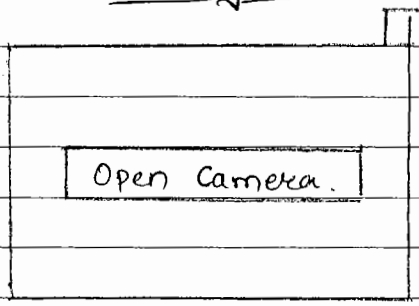
```
<application

  <activity
    android:name = ".ActivityCamera"

    <intent-filter>
      <action android:name = "android.intent.action.MAIN" />

      <category
        android:name = "android.intent.category.LAUNCHER" />
    </intent-filter>
  </activity>
</application>
```

- Snippet :

```
Intent i = new Intent (MediaStore . ACTION_VIDEO_
                                    CAPTURE);
startActivity (i);
```

Q. Example to capture a video from camera.

* CWProject-33 *



↳ layout_camera . xml

```
< RelativeLayout
    android : layout_width = "match_parent"
    android : layout_height = "match_parent"

    < Button
        android : layout_width = "match_parent"
        android : layout_height = "wrap_content"
        android : text = "Open Camera"
        android : onClick = "Open" />

</ RelativeLayout >
```

↳ Activity Camera . java

```
import android . app . Activity ;
```

→

```
public class Activity Camera extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super. onCreate ( savedInstanceState);
        setContentView ( R. layout. layout_camera);
    }

    public void open (View v)
    {
        Intent- i = new Intent (MediaStore.
                                ACTION_VIDEO_CAPTURE);
        start-Activity (i);
    }
}
```

(1)

↳ Android Manifeast . xml

```
< application

    < activity  android: name = " . Activity Camera"

        < intent- filter>
            < action
                android: name = "android. intent. action. MAIN " />

            < category
                android: name = "android. intent. category.
                                LAUNCHER " />

        </ intent- filter>

    </ activity>

</ application >
```

(2) (3) (4) (5)

* Playing a Video :

- Android MediaPlayer class works with video files the same as Audio files but there is small difference in between the both -
  i.e. For video playing we need to create a surface on layout to play video.

- We use start() and stop() methods to control the play.

- Android.widget.VideoView class is providing set of Methods that can be used in video play.

- **Methods :**

①	setVideoPath()	This video is used to specify the path of the video file to be played. This can either be a load path or a remote url.
②	setVideoUri()	This method is same as above method accepted the parameters.
③	start()	This method is used to start the video playback.
④	stop()	This method is used to stop the video playback.
⑤	pause()	This method pause the video.

- Load the video :

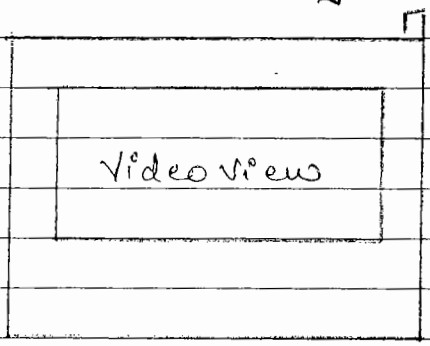
```
Uri video1 = Uri.parse("android.resource://" +
        getPackageName() + "/" + R.raw.file_name);
```

- Steps :

- Create a new folder into 'res' folder and named it as "raw" & copy the video into it.

Q. Example to play video in VideoView.

* CWProject-34 *



↳ layout_video.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <VideoView
        android:id = "@+id/vv1"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:layout_margin = "10dp" />

</RelativeLayout>
```

↳ ActivityVideo.java

```
import android.app.Activity;

public class ActivityVideo extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_video);

        VideoView vv = (VideoView) findViewById (R.id.vv1);

        Uri ref = Uri.parse ("android.resource://" +
                  getPackageName () + "/" + R.raw.bunny);

        vv.setVideo~~URI~~Uri (ref);
        vv.start ();
    }
}
```

- Snippet :

```
MediaController mc = new MediaController (this);
vv.setMediaController (mc);
```

↳ ~~Android V~~ ActivityVideo.java

```
import android.app.Activity;

public class ActivityVideo extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState);
    {
```

→

```
Super.onCreate (savedInstanceState);
setContentView (R.layout.layout_video);
}
VideoView vv = (VideoView) findViewById (R.id.vv1);

Uri ref = Uri.parse ("android.resource://" +
          getPackageName() + "/" + R.raw.bunny);

vv.setVideoURI (ref);

MediaController mc = new MediaController (this);
vv.setMediaController (mc);

vv.start ();
}
}
```

↳ Android Manifeast.xml

```
<application
  <activity android:name = ".ActivityVideo" >
    <intent-filter>
      <action
        android:name = "android.intent.action.MAIN" />
      <category
        android:name = "android.intent.category.
                        LAUNCHER" />
    </intent_filter>
  </activity>
</application>
```

* Services : (Very Important Topic)

- Android Service is a component that is used to perform operations from background such as playing.

NOTE : The service doesn't have any User Interface.

- The service runs in background indefinitly even if application is destroyed.

- Moreover, Service can be bounded by a component to perform interactively and interprocess communication (IPC).

- Interprocess communication is the activity of sharing data across multiple and commonly specialized processes.

- There are two forms of services -
  1. Started Services.
  2. Bounded Services.

1. Started Services :

- A service is started when an application component starts it by calling "startService()" method.

- Once if the services is started, the service will run indefinetely.

- Usually a started service performs single operation and doesn't returns a result to the caller. Ex. file uploading & downloading, vibrator etc.

- The service is stoped by using "stopService()" method.

- The service can stop itself by calling "~~stopIt~~ "stopSelf()" method.

② Bounded Services :

- A service is bound when an application component binds it by calling "bindService()" method

- A bound service offers a client ~~service~~ server interface that allows components to interact with service, send request, get result & so on.

- To unbind a service we call ~~unbound~~ "unbindService()" method.

- Services are of Two types -
  ① pre-defined Services
  ② User-defined Services.

① Pre-defined Services :

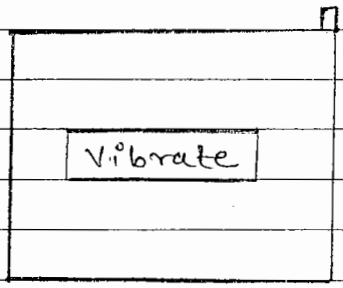
- The services which are providing by android Operating System by default.

Ex. ALARM_MANAGER, SMS MANAGER, VIBRATOR, LOCATION_MANAGER etc.

Q. Example on Vibrator Services.

* CWProject35 *

NOTE : To work with pre-defined Services we use - "getSystemService(P,P);"



↳ layout_vibrate.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <Button
        android:id = "@+id/btnVib"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Vibrator"
        android:onClick = "display" />

</RelativeLayout>
```

↳ ActivityVibrate.java

```
import android.app.activity;

public class ActivityVibrate extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_vibrate);
    }
```

→

```
public void display (View v)
{
    Vibrator vb = (Vibrator) getSystemService
                  (VIBRATOR_SERVICE);
    vb.Vibrate (10,000);
}
}
```

↳ Android Manifeast.xml

```
<~~api~~
<user_permission
    android:name = "android.permission.~~VIBERATOR~~ VIBRATE" />

<application

    <activity  android:name = ".AdivityVibrate"

        <intent_filter>

            <action
                android:name = "android.intent.action.MAIN" />

            <category
                android:name = "android.intent.category.LAUNCHER"/>

        </intent_filter>

    </activity>

</application>
```

* Alarm Service :

- Android provides android.app.AlarmManager class which is used to access system Alarm Services.

- The AlarmManager will release the lock after the execution of "onRecive()" method.

- ~~setMethod~~ set() of AlarmManager is used to register an Intent to be send.

- Syntax :

- set(int type, long triggerAtMills, PendingIntent operations);

  ↳ triggerAtMillis is the time in Milliseconds that the alarm should go off.

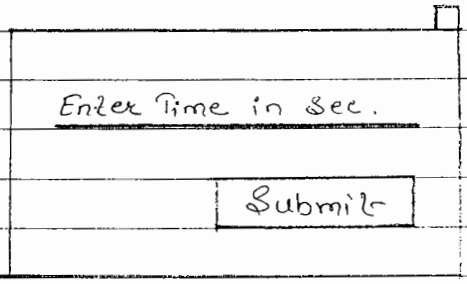
  ↳ Operations is the Intent instance that will send when the alarm go off.

- Ex.

```
alarmManager.set(AlarmManager.RTC_WAKEUP,
        system.currentTimeMillis() + (i * 1000),
        PendingIntent);
```

Q. Example on Alarm Manager, to ring Alarm.

* CWProject-36 *



↳ layout_alarm.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <EditText
        android:id = "@+id/et1"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:hint = "Enter Time in sec"
        android:layout_margin = "20dp" />

    <Button
        android:id = "@+id/btnsubmit"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Submit"
        android:onClick = "display" />

</RelativeLayout>
```

↳ ActivityAlarm.java

```
import android.app.Activity;

public class ActivityAlarm extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_alarm);
    }
```

→

```
public void display (View v)
{
    EditText et = (EditText) findViewById (R.id.et1);
    int sec = Integer.ParseInt (et.getText().toString());
    PendingIntent pi = PendingIntent.getActivity (this, 0, new Intent
        (this, ActivityPending.class, 0);

    AlarmManager am = (AlarmManager) getSystemService
        (ALARM_SERVICE);

    am.set (AlarmManager.RTC_WAKE UP, System.
        currentTimeMillis () + (sec * 1000), pi);
    }
}
```

↳ ActivityPending.java

```
import android.app.Activity;

public Class ActivityPending extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedIntanceState);

        MediaPlayer mp = MediaPlayer.create (this, R.raw.
            aud);

        mp.start ();
    }
}
```

→

↳ Android Manifeast . xml

```
<application
    <activity android:name = ". Activity Alarm" >
        <intent_filter>
            <action
                android:name = "android.intent.action.MAIN" />
            <category
                android:name = "android.intent.category.LAUNCHER" />
        </intent_filter>
    </activity>
</application>
```

* Broad Cast Receivers :

- Android.Content.BroadCastReceiver is a class which is providing set of Methods to work with broadcast-Receiver.

- A Broad Cast Receiver is a component that requires response to system wide Broadcast Announcement.

- Many Broad Casts organize by system.

Ex. A Broad Cast announcing that the screen has turn off, the Battery is low, or the picture was captured.

- The BroadCast Receiver don't display a User Interface, They may create Status Bar, Notification to alert the user, when a broadcast event occurs.

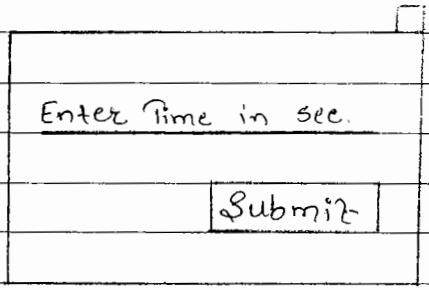
- onReceive () : This method takes two arguments like onReceive (context context, Intent intent);

- Intent can be a pendingIntent.

- NOTE : A receiver must be registered in manifeast file.
  < receiver android : name = " file-name " />

Q. Example on BroadCast Receiver.

* CWProject37 *



↳ ActivityBroadCast . java.

```
import android . app . Activity ;

public class ActivityBroadCast extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super . onCreate ( savedIntanceState);
        setContentView ( R. layout . layout-broadcast);
    }
```

→

```
public void display (View v)
{
    EditText et = (EditText) findViewById (R.id. et1");
    int sec = Integer . parseInt ( et. getText().
                    . to string ());

    Intent i = new Intent ( this , ActivityReceiver.class)

    PendingIntent pi = PendingIntent . getBroadCast
                    ( this , 0 , i , 0);

    AlarmManager am = (AlarmManager) getSystemServic
                    (ALARM_SERVICE);

    am. set ( AlarmManager , RTC_WAKEUP ,
            system. currentTimeMillis () +
            (sec * 1000) , pi);

    Toast. makeText ( this , "Time up", 3000). show();
    }
}
```

↳ ActivityReceiver . java

```
public class ActivityReceiver extends BroadCastReceiver
{
    @Override
    protected void onResume Receive (Context ctx ,
                    Intent intent)
    {
    Toast. makeText ( this, "BroadCast Receiver", 3000) show
                    ctx

    MediaPlayer mp = MediaPlayer . create
                    (ctx, R.raw . aud);
    mp. start ();
    }
}
```

↳ Android Manifest.xml

```
<application
    <activity android:name = "ActivityBroadcast"
        <intent_filter>
            <action android:name = "android.intent.action.MAIN" />
            <category android:name = "android.intent.category.LAUNCHER" />
        </intent_filter>
    </activity>
    <receiver android:name = "ActivityBroadcast" />
</application>
```

* SMS Manager :

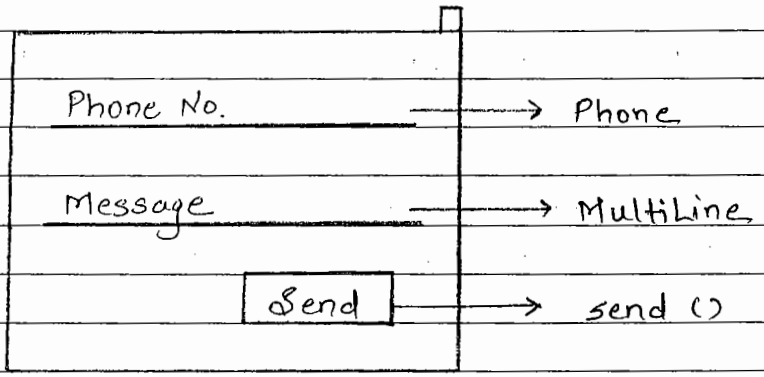
- In android we can use android.telephony.SmsManager class or build-in SMS applications to send SMS.
- SMS Manager class manages SMS operations such as Sending Data, Text SMS Messages.
- We cannot create object to SMS Manager class directly. We use getDefault() static method of SMS Manager class.
- To send a text message we use sendTextMessage() method.

- Snippet :

```
SmsManager.sendTextMessage("Phone_No", Null, "SMS message", null, null);
```

Q. Example on SMS Manager.

* CWProject-38 *



↳ Activity sms . java .

```
import android . app . Activity ;

public class Activity sms extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super . onCreate ( savedInstanceState);
        setContentView ( R . layout . layout_sms);
    }

    public void send ( View v)
    {
        EditText et1 = (EditText) findViewById (R.id. et1);
        EditText et2 = (EditText) findViewById ( R.id. et2);

        String phone_no = et1 . getText () . toString ();
        String mess = et2 . getText () . tostring ();

        phone_no = phone_no . trim ();
        mess = mess . trim ();
```

```
SMSManager sms = SMSManager.getdefault();
sms.sendTextMessage (phone_no, null, mess,
                     null, null);

NotificationManager NM = (NotificationManager)
    getSystemService (this.NOTIFICATION_SERVICE);

Notification notify = new Notification (R.drawable.
    icon_fire, "Mess", System.currentTimeMillis());

PendingIntent pi = PendingIntent.getActivity (this,
    0, new Intent(), 0);

notify.setLatestEventInfo (this, "Message Title",
    "sending Message", pi);

NM.notify (0, notify);
    }
}
```

↳ Android Manifeast.xml

```
<uses_permission
    android:name = "android.permission.SEND_MESSAGE" />
```

);
);

* User-Defined Services :

- Steps to create User-defined Services :

- Step 1 : Create any User-defined class and extends "android.app.service" class.

- Step 2 : override onBind() method of service class.

- Step 3 : override the service class life cycle methods.
  a). onCreate();
  b). onStart();
  c). onDestroy()

- Methods :

①	onBind()	This method is used to made a communication channel between Activity class and service class.
②	onCreate()	This method called ~~when~~ by the system when the service is first created. Do not call this method directly.
③	onStart()	This method called by the system when service is started.
④	onDestroy()	This method called by the system to notify a service that it is no longer used and is being removed.

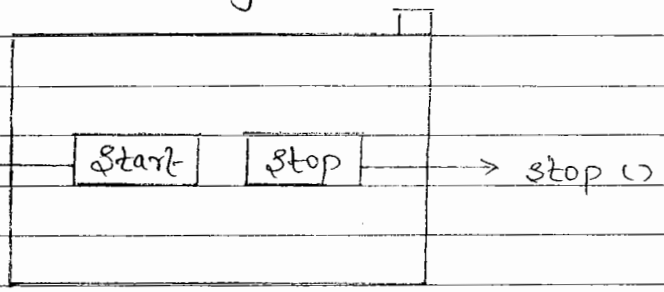
- startService (Intent service)
  ↳ Request that a given application service be started.

- stopService (Intent name)
  ↳ Request that a given application service be stopped.

- NOTE : User-defined service must be destroyed explicitly by a programmer.

Q. Example on User-defined Services.

* CWProject39 *



↳ ActivityService.java

```
import android.app.Activity;

public class ActivityService extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_service);
    }
```

→

```
public void start (View v)
{
    startService (new Intent (this, MyService.class));
}

public void stop (View v)
{
    stopService (new Intent (this, MyService.class));
}
}
```

→ MyService.java

```
public class MyService extends Service
{
    @Override
    public IBinder onBind (Intent intent)
    {
        return null;
    }

    MediaPlayer mp;

    @Override
    public void onCreate ()
    {
        Toast.makeText (this, "Create", 3000).show();
        mp = MediaPlayer.create (getApplicationContext(
                R.raw.aud);
        mp.setLooping (false);
    }

    @Override
    public void onStart (Intent intent, int startID){
    {
        mp.start ();
```

```
        Toast.makeText(getApplicationContext(), "start",
                3000).show();
    }

    @Override
    public void onDestroy()
    {
        mp.stop();
        Toast.makeText(getApplicationContext(), "Stop",
                3000).show();
    }
}
```

↳ Android Manifeast.xml

```
<application
    <activity android:name=".ActivityService">
        <intent_filter>
            <action android:name="android.intent.action.MAIN"/>
            <category
                android:name="android.intent..category.
                    LAUNCHER" />
        </intent_filter>
    </activity>
    <service android:name="com.example.
            cwproject39.Myservice" />
</application>
```

* Android Internal Storage :

- Internal Storage is a storage of the private data on the Device memory.
- By default, this files are private and are accessed by only your application and get deleted, When user deletes your application.

* Writing file :

- In order to use the internal storage to write some data into the file, call the openFileOutput() method with the name of the file and mode.

```
FileOutputStream fout = openFileOutput ("file_name",
                        MODE_WORLD_READABLE);
```

- The method openFileOutput() will return FileOutputStream reference.
- By Using FileOutputStream reference we can call write() method to write data into file.

Ex.

```
String str = "data";
fout.write (str.getbytes());
fout.close();
```

* Reading file :

- In order to read from the file we need to use openFileInput() method with the Name of the file.

— This method will return FileOutputStream reference.

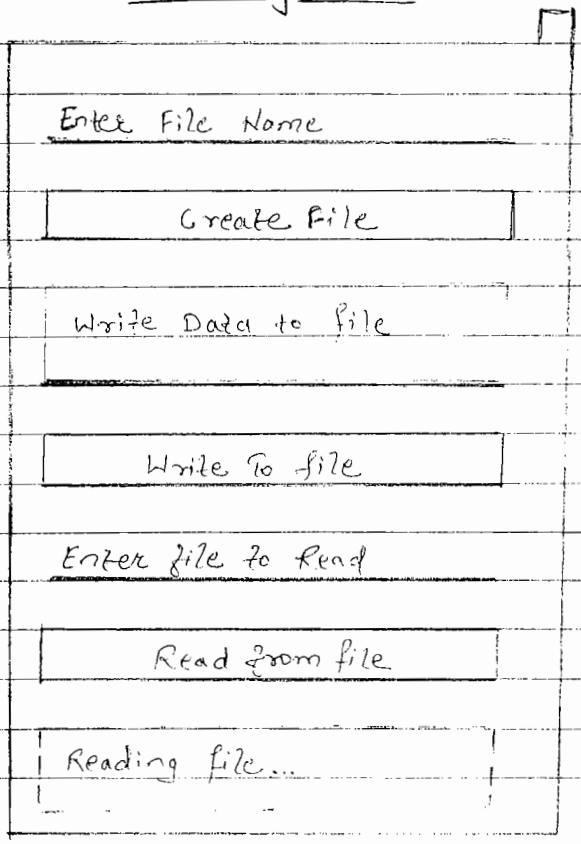
— By using FileOutputStream reference we can call read() method.

Ex.

```
FileInputStream fin = OpenFileInput (file);

int c;
String temp = "";
While (c = fin.read()) != -1)
{
    temp = temp + character.toString(char) c);
}
// string temp contains all the data from file.
fin.close();
```

Q. Design an android application to create a file and write the data and Read from file.

* CWProject 40 *



↳ layout_file.xml

```
<ScrollView
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <LinearLayout
        android:layout_width = "match_parent"
        android:layout_height = "match_parent"
        android:orientation = "vertical"
        android:gravity = "center" >

        <TextView
            android:id = "@+id/txtHeader" />

        <EditText
            android:id = "@+id/etCreatefile"
            android:layout_width = "match_parent"
            android:layout_height = "wrap_content"
            android:hint = "Enter file Name" />

        <Button
            android:id = "@+id/btnCreate"
            android:layout_width = "match_parent"
            android:layout_height = "wrap_content"
            android:onClick = "createFile"
            android:text = "Create File" />

        <EditText
            android:id = "@+id/etWrite"
            android:layout_width = "match_parent"
            android:layout_height = "wrap_content"
            android:hint = "Write Data to file"
            android:inputType = "textMultiLine" />
```

```
<Button
    android : id = "@+id / btnWrite"
    android : layout_width = "match_parent"
    android : layout_height = "wrap_content"
    android : text = "Write to file"
    android : onClick = "writeToFile" />

<EditText
    android : id = "@+id/ etFromFile"
    android : layout_width = "match_parent"
    android : layout_height = "wrap_content"
    android : hint = "Enter fileName to Read..." />

<Button
    android : id = "@+ id / btnRead"
    android : layout_width = "match_parent"
    android : layout_height = "wrap_content"
    android : text = "Read from file"
    android : onClick = "readfromfile" />

<TextView
    android : id = "@+id / textRead"
    android : layout_width = "match_parent"
    android : layout_height = "120 dp" />

</LinearLayout>

</ScrollView>
```

↳ ActivityFile.java

```
import android.app.Activity;
import java.io.FileInputStream;
import java.io.FileOutputStream;

public class ActivityFile extends Activity
{
    EditText fileName, writeText, readFile;
    FileOutputStream fout;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_file);
    }

    public void createFile(View v)
    {
        fileName = (EditText) findViewById(R.id.etCreatefile);
        String fname = fileName.getText().toString();
        try
        {
            fout.openFileOutput(fname, MODE_WORLD_READABLE);
            Toast.makeText(this, "File Created", 3000).show();
        }
        catch (Exception ex)
        {
            Toast.makeText(this, "File Not Created", 3000).show();
        }
    }
```

→

```
public void writeToFile (View v)
{
    writeText = (EditText) findViewById (R.id.etWrite);
    String str = writeText.getText().toString();

    try
    {
        byte barr[] = str.getbytes();
        fout.write (barr);
        Toast.makeText (this, "Data written to file", 3000).
        fout.close();                                    show();
    }
    catch (Exception ex)
    {
        Toast.makeText (this, "Data Not written to file",
                                        3000).show();
    }
}
public void readFromFile (View v)
{
    readFile = (EditText) findViewById (R.id.etFromFile);
    String strFromFile = readFile.getText().toString();
    try
    {
        String text = "";
        FileInputStream fin = openFileInput (strFromFile);
        int num = fin.read();

        while (num != -1)
        {
            text = text + (char) num;
            num = fin.read();
        }
        TextView tv = (TextView) findViewById (R.id.txtRead);
        tv.setText (text);
    }
```

→

```
        catch (Exception ex)
        {
            Toast.makeText(this, "Data Not Readable", 3000).
                                                    show();
        }
    }
}
```

↳ Android Manifeast.xml

```
<application
    <activity android:name = ". ActivityFile"
        <intent_filter>
            <action
                android:name = "android.intent.action.MAIN" />
            <category
                android:name = "android.intent.category.LAUNCHER" />
        </intent_filter>
    </activity>
</application>
```

* Android External Storage :

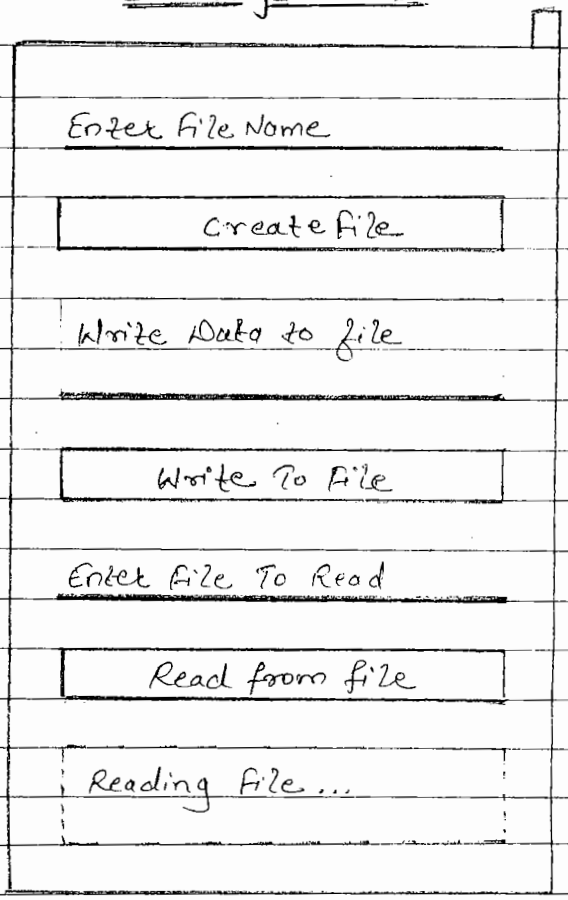
- Like Internal Storage we are able to save or read data from the device External memory such as sd card.

- To work with External memory we are using - "FileInputStream" and "FileOutputStream" classes.

Q. Design an android application to create a file into sd card and write the data & Read from file.

).
();

* CW Project 41 *



>

↳ layout_file . xml

NOTE : Same as the before layout.

↳ Activity file . java

```
public class ActivityFile extends Activity
{
    EditText filename, text, fromfile;
    File f;

    @Override
```

→

```
protected void onCreate (Bundle savedInstanceState)
{
    Super. oncreate (savedInstanceState)
    setContentView ( R. layout. layout_file );
}

public void createFile (View v)
{
    filename = (EditText) findViewById (R.id. filename);
    String fname = filename . getText (). toString ();
    try
    {
        f = new File ("/ sdcard / " + fname);
        f . createFile ();
        Toast. makeText (this, "File Created", 3000). show();
    }
    catch (Exception ex)
    {
        Toast. makeText-(this, " "+e , 3000). show ();
    }
}

public void saveFile (View v)
{
    text = (EditText) findViewById (R.id. tofile );
    String str = text. getText (). toString ();
    try
    {
        FileOutputStream fout = new FileOutput-Stream (f
        byte b[] = str. getbytes ();
        fout. write (b);
        Toast. makeText (this, "Data written to file",
                                    3000). show ();
        fout. close ();
    }
```

→

```
        catch (Exception ex)
        {
            Toast.makeText(this, "" + e, 3000).show();
        }
    }

    public void openfile (View v)
    {
        fromfile = (EditText) findViewById (R.id.fromfile);
        String from = fromfile.getText().toString();
        try
        {
            String text = "";
            File f1 = new File ("/sdcard/" + from);
            FileInputStream fin = new FileInputStream (f1);
            int no = fin.read();

            while (no != -1)
            {
                text = text + (char) no;
                no = fin.read();
            }
            TextView tv = (TextView) findViewById (R.id.filetext);
            tv.setText (text);
        }
        catch (Exception ex)
        {
            Toast.makeText (this, "" + e, 3000).show();
        }
    }
}
```

↳ ~~Android Manifest.xml~~

→

↳ Android.Manifeast.xml

```
<app uses-permission
    android:name = "android.permission.WRITE_EXTERNAL_STORAGE" />

<application>

    <activity>
        android:name = ".ActivityFile"

        <intent-filter>

            <action android:name="android.intent.action.MAIN" />

            <category android:name = "android.intent.category.LAUNCHER" />

        </intent-filter>

    </activity>

</application>
```

* Resources : (Very Important concept in Android)

— You should always externalize resources such as images, strings, backgrounds etc from your application code, so that you can maintain them independently.

— Externalizing your resources also allows you to provide alternative resources that supports specific device configurations such as different Languages or screen sizes.

- In order to provide compatibility with different configurations you must organize your resources in your project "res / directory" using various sub-directories.

- Grouping Resource Types -

↳ You should place each type of resource in a specific sub-directory of your projects - "res / directory"

Ex. My Project /
src /
MyActivity.java
res /
drawable /
icon.png
layout /
main.xml
info.xml
values /
strings.xml

* String Resources :

- A single string that can be reference from application or from other resources files such as external layout.

↳ FILE_LOCATION :
res / values / file_Name.xml

↳ The <string> elements name will be used as the resource Id.

↳ RESOURCE REFERENCE :
In Java : R.string.string_name;
In Andro xml : @string / string_name;

* CWProject-42 *

↳ Code in res / values / strings . xml

```
<string name = "uname"> UserName </string>
<string name = "pass"> Password </string>
```

↳ Code in layout_file . xml

```
<TextView
    android : id = "@+id / tv1"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : text = "@string / uname" />
```

→ String Array :

— An array of strings that can be referenced it from the application.

↳ FILE_LOCATION :
res / values / file_name . xml

↳ The <string-array> elements name will be used as the resource ID.

↳ RESOURCE_REFERENCE :
In Java : R. array. string_array_name ;

↳ Code in res / values / strings. xml. * CWProject43 *

```
<string-array name = "list_of_lang" >
    <item> c-lang </item>
    <item> Cpp </item>
    <item> Java </item>
    <item> Android </item>
</string-array>
```

↳ Code in layout.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <Spinner
        android:id = "@+id/spin1"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content" />

</RelativeLayout>
```

↳ Code in Activity.java

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

        Spinner sp = (Spinner) findViewById (R.id.spin1);
        Resources ref = getResources ();

        String lang[] = ref.getStringArray (R.array.
                                    list_of_lang);

        ArrayAdapter aa = new ArrayAdapter (this,
            android.R.layout.simple_dropdown_item_1line, lang);

        sp = setAdapter (aa);
    }
}
```

- Affostrophies and Single & double cotes :

- If you have affostrophies or single or double cotes in your string you must either escape it (\') or enclose the whole string in double cotes (" ' " (" ");

Ex.

<String name = "one"> "It is a collection of object's" </String>

<String name = "two"> It is a collection of object\'s </String>

- String with HTML Markups. :

- You can add styling to your string with HTML Mark ups :

↳ Supported HTML Markups :

- - for Bold text.
- - for italic text.
- <u> - for underline text.

Ex.

<String name = "uname"> UserName </String>

<String name = "uname1"> UserName </String>

<String name = "uname2"> <u> UserName </u> </String>

* Color State List Resources :

- Color state list is an object you can define in xml that you can apply as a color but will actually change colors depending on the state of the view object.

Ex. - A Button widget can exist in one or several different states (pressed, focused, released etc.) and using color state list you can provide a different color during different states.

↳ FILE LOCATION :

res / color / file_name . xml

↳ The file_name will be used as resource ID.

↳ RESOURCE REFERENCE :

In Java : R. color. filename.

In xml : @[package.] color / filename.

- Elements :

<selector>

Required. This must be the first most element. contains one or more <item> elements.

<item>

Defines a color to use during certain states, as describe by its attributes, Must be a child of a <selector> element.

Attributes

android . color

Hexadecimal color,

android . state_pressed

Boolean "true" / "false".

android.state_focused
Boolean "true" / "false"

android.state_selected
Boolean "true" / "false"

android.state_checkable
Boolean "true" / "false"
[only useful if the object can transition between a checkable and non-checkable widget]

android.state_checked
Boolean "true" / "false"

android.state_enabled
Boolean "true" / "false"

* Colors :

- A color value defined in xml. The color is specified with an RGB value and alpha channal.

- You can also use a color resources when a drawable resource is expected in xml.

Ex. android:drawable = "@color/green".

↳ FILE LOCATION :
res / values / colors.xml

↳ The fileName is arbitary. The <colorY elements name will be used as a resource ID.

↳ RESOURCE REFERENCE :
In Java : R.color.color_name.
In xml : @[package.]color/color_name.

↳ Syntax :

```
<? xml version = "1.0" encoding = "utf-8" ?>

<resources>
    <color name = "color_name" > hex_color </ colors>
</ resources >
```

Q. Example of color state list :

* CWProject-44 *

— [res/colors/button_click_change_color.xml]

↳ button_click_change_color.xml

```
<selector>
    <item> android : state_pressed = "true"
           android : color = "#FF8811" />

    <item> android : color = "#1188FF" /></item>
</selector>
```

— [res/colors/checkbox_color.xml]

↳ checkbox_color.xml

```
<selector>
    <item  android : state_~~pressed~~ checked = "true"
           android : color = "#FF8822" />

    <item  android : color = "#2288FF" />
</selector>
```

— [res/layout/layout_one.xml]

↳ layout_one.xml

```
<Button
    android : textColor = "@color/button_click_change_color" />
```

<checkBox
android : textColor = "@ color / checkBox_color" />

↳ Activity ColorStateList . java

```
public class ActivityColorStateList extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super. onCreate ( SavedInstanceState);
        setContentView (R. layout. layout_one);
    }
}
```

↳ Android Manifeast . xml

Q. Example of Color State List.

* CWProject-45 *

— [res / colors / button_colors . xml]

↳ button_colors . xml

```
<selector>
    <item   android : state_pressed = "true"
            android . state color = "#FF8811" />

    <item   android . color = "#000000" />
</selector>
```

— [res / layout / layout_one . xml]

↳ layout_one . xml

```
<LinearLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:orientation = "vertical"
    android:gravity = "center"
    android:layout_margins = "10dp" >
    android:textColor = "@color/button_colors" />

    <Button
        android:text = "Admin Login"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content" />
        android:textColor = "@color/button_colors" />
    <Button
        android:text = "Employee Login"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:textColor = "@color/button_colors" />

    <Button
        android:text = "Client Login"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:textColor = "@color/button_colors" />

    <Button
        android:text = "User Login"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:textColor = "@color/button_colors" />

</LinearLayout>
```

↳ ActivityButtons.java

```
public class ActivityButtons extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        Super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_one);
    }
}
```

↳ Android Manifeast.xml

Q. Example program on Color Resources.

* CWProject 46 *

— [res / values / colors.xml]

↳ Colors.xml

```
<resources>
    <color name = "title_color"> #FF8811" </color>
    <color name = "default_color"> #000000" </color>
</resources>
```

— [res / colors / ~~color.xml~~ button_colors.xml]

↳ button_colors.xml

```
<selector>
    <item android.state_pressed = "true"
          android.color = "title_color" />
    <item android.color = "default_color" />
</selector>
```

- [res / layout / layout_button_color.xml]

↳ layout_button_color.xml

```
<Relative_Layout>
    android : layout_width = "match_parent"
    android : layout_width = "match_parent" >

    <Button
        android : id = "@+id / btn1"
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content" />

</Relative Layout>
```

↳ ActivityButtonColor.java

```
public class ActivityButtonColor extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_button_color);
    }
}
```

↳ Android Manifest.xml

* Drawable Resources :

- Drawable Resources is a resources for a graphic that can be drawn to be screen and which you can retrive with getDrawable () methods.

① Bitmap :

- Android supports Bitmap files in three formates -
  - .png (prefered),
  - .jpg (acceptable),
  - .gif (discouraged).

↳ FILE LOCATION :

res / drawable / filename.png (.png, .jpg, .gif)

↳ RESOURCE REFERENCE :

In Java : R.drawable.filename

In xml : @[package.] drawable / filename

- Java Snippet :

```
↳ Resources res = getResources ();
  Drawable drawable = res.getDrawable (R.
                      drawable.myImage);
```

② Layer List :

- A layer drawable is a drawable object that manages an array of other drawables.

- Each drawable in the list is drawn in the order of the list, the last drawable in the list is drawable on top.

↳ FILE LOCATION :

res / drawable / filename.xml

↳ RESOURCE LOCATION

In Java : R.drawable.filename

In xml : @[package.]drawable/filename.

↳ Elements :

<layer_list>

This must be the root element, contain one or more <item> elements.

<item>

Defines a drawable to place in the layer drawable, in a position defined by its attributes.

Accepts child <Bitmap> elements.

<item_attributes>

android : top

The top offset in pixel.

android : right

The right offset in pixel.

android : bottom

The bottom offset in pixel.

android : left

The left offset in pixel.

Q. Example program of Layer_List.

* CWProject_47 *

— [res / drawable / image_layers.xml]

↳ image_layers.xml

```
<layer_list>

    <item>
        <Bitmap  android:src="@drawable/icon3"
                 android:gravity="center" />
    </item>

    <item android:top="10dp" android:left="10dp">
        <Bitmap  android:src="@drawable/icon2"
                 android:gravity="center" />
    </item>

    <item android:top="20dp"
          android:left="20dp">
        <Bitmap  android:src="@drawable/icon1"
                 android:gravity="center" />
    </item>

</layer_list>
```

— [res / layout / layout-one . xml]

↳ layout-one . xml

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/image_layers" />

</RelativeLayout>
```

↳ ActivityLayer.java

```
public class ActivityLayer extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_one);
    }
}
```

↳ Android Manifest.xml

(3) State List :

- A state list drawable is a drawable object defines in xml that uses a several different images to represent the same graphic depending on the state of the object.

- Each graphic is represented by item element inside a single selector element.

↳ FILE LOCATION :

res / drawable / filename.xml

↳ Resource Reference :

In Java : R.drawable.filename

In xml : @[package:] drawable / filename.

Q. Example Program on State List.

* CWProject 47 *

- [res / drawable / state_list.xml]

↳ state_list.xml

```
<selector>
    <item   android:state_pressed = "true"
            android:drawable = "@drawable/icon1" />

    <item   android:drawable = "@drawable/icon2" />
</selector>
```

- [res / layout / layout_one.xml]

↳ layout_one.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <ImageButton
        android:src = "@drawable/state_list" />

    <ImageView
        android:src = "@drawable/state_list" />
</RelativeLayout>
```

- [src / (PackageName) / ActivityOne.java]

↳ ActivityOne.java

→

```
public class ActivityOne extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView ( R.layout.layout_One);
    }
}
```

1") — Android Manifest.xml.

4) Transition Drawable :

- A Transition Drawable is a drawable object that cross fade between the two drawable resources.
- Each drawable is represented an <item> element inside a single transition element.
- To Transition Forword, call startTransition ().
  To Transition Backward, call reverseTransition ().

↳ FILE LOCATION :
res / drawable / filename.xml

↳ ~~Res~~ RESOURCE REFERENCE :
In Java : R.drawable.filename.
In xml : @[package :] drawable/filename.

Q. Example Program on Transition Drawable.

* cwProject48 *

- [res / drawable / button_on_off.xml]

↳ button_on_off.xml

```
< ~~selector~~ transition >

    <item android:drawable="@drawable/on" />
    <item android:drawable="@drawable/off" />

</transition>
```

- [res / layout / layout_one.xml]

↳ layout_one.xml

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <ImageButton
        android:onClick="change" />

</RelativeLayout>
```

- [src / (packageName) / ActivityOne.java]

↳ ActivityOne.java.

```
public class ActivityOne extends Activity
{
    @Override
```

→

```
protected void onCreate (Bundle savedInstanceState)
{
    super.onCreate (savedInstanceState);
    setContentView (R.layout.layout_one);
}

public void change (View v)
{
    ImageButton ib = (ImageButton) findViewById
                        (R.id.imgbtn1);
    TransitionDrawable td = (TransitionDrawable)
                        ib.getDrawable ();
    td.~~setTransit~~ startTransition (5000);
}
}
```

— Android Manifest.xml.

⑤ Shape Drawable :

— Shape drawable is a generic shape define in xml.

↳ FILE LOCATION :

res/drawable/filename.xml

↳ RESOURCE REFERENCE :

In Java : R.drawable.filename

In xml : @[package:]drawable/filename.

↳ ELEMENTS :

< shape

The shape drawable. This must be the root element.

↳ attributes :

android : shape

Defines the type of shape. Valid values are -

value	Description
① "rectangle"	The rectangle that fills the containing view. This is a default shape.
② "Oval"	An Oval shape that fits the diamension of the containing view.
③ "line"	A horizontal line that spaces the width of the containing view. This shape requires the <stroke> element to define the width of the line.
④ "ring"	A ring shape.

- The following attributes are used only when "android : shape = "ring"".

* android : innerRadius

The radius for the inner part of the ring. (The hole in the middle), as a dimension value or diamension resource.

* android : innerRadiusRatio

The radius of the inner part of the ring expressed as a ratio of the rings width.

* android : thickness

The thickness of the ring, as a diamension value or diamension resource.

* android : uselevel

Boolean "true" if this is used as a LevelList-Drawable. This should normally be "false" or your shape may not appear.

<size>

The size of the shape.

↳ attributes :

android : height-

Diamension. The height of the shape.

android : width

Diamension. The width of the shape.

<solid>

A solid color to fill the shape.

↳ attributes :

android : color

color. The color to apply to the shape.

Q. Example Program on Shape Drawable.

* CWProject 49 *

— [res / drawable / oval_shape.xml]

↳ Oval_shape.xml.

→

```
<shape
    android : shape = "oval" >

    <size  android : width = "100dp"
           android : height = "50dp" />

    <solid  android : color = "#FF8811" />
</shape>
```

— [res / drawable / circle_shape.xml]

↳ circle_shape.xml

```
<shape
    android : shape = "oval" >

    <size  android : width = "100 dp"
           android : height = "100 dp" />

    <solid  android : color = "#FF8811" />
</shape>
```

— [res / drawable / ring_shape.xml]

↳ ring_shape.xml

```
<shape
    android : shape = "ring"
    android : innerRadius = "40 dp"
    android : thickness = "10 dp"
    android : useLevel = "false" >

    <size  android : width = "100 dp"
           android : height = "100 dp" />

    <solid  color = "#FF8811" />
</shape>
```

- [res / layout / layout_one . xml]

↳ layout_one . xml

```
<RelativeLayout
    android : layout_width = "match_parent"
    android : layout_height = "match_parent" >

    <Button
        android : background = "@drawable / oval_shape" />

    <Button
        android : background = "@drawable / circle_shape" />

    <Button
        android : background = "@drawable / oval_shape" />

    <Button
        android : background = "@drawable / circle_shape" />

    <Button
        android : background = "@drawable / ring_shape" />

</RelativeLayout>
```

- [Src / (packageName) / ActivityOne . java]

↳ ActivityOne . java .

```
public class ActivityOne extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super. onCreate (savedInstanceState);
        setContentView (R. layout. layout_one );
    }
}
```

(6) Corners :

< corners >

Creates rounded corners for the shape. Applies only when shape is in rectangle.

attributes :

* android : radius

The radius for all corners. This is overridden for each corner by the following attributes.

* android : topLeftRadius

The radius for top-left corners.

* android : topRightRadius

The radius for top-right corners.

* android : bottomLeftRadius

The radius for bottom-left corners.

* android : bottomRightRadius

The radius for bottom-right corners.

Q. Exampe Program on corners.

* CWProject-50 *

— [res / drawable / all_corners.xml]

↳ all_corners.xml

< Shape >

→

```
    <corners  android:radius = "20dp" />

    <solid  android:color = "#FF8811" />
</shape>
```

- [res / drawable / selected_corners.xml]

↳ selected_corners.xml

```
<shape
    <corners  android:topLeftRadius = "30dp"
              android:BottomRightRadius = "30dp" />

    <solid  android:color = "#FF8811" />
</shape>
```

- [res / layout-/ layout_one.xml]

↳ layout_one.xml

```
<RelativeLayout-
    android:layout_width = "match_parent"
    android:layout_height- = "match_parent" >

    <Button
        android:background = "@drawable / all_corners" />

    <Button
        android:background = "@drawable / selected_corners" />

</RelativeLayout->
```

— [src / (packageName) / ActivityOne.java]

↳ ActivityOne.java. →

```
public class ActivityOne extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        Super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_one);
    }
}
```

- Android Manifeast. xml

⑦ Gradient :

- <Gradient>

Specifics a gradient color for the shape.

↳ attributes :

* android : angle

The angle for gradient, in degree. 0 is left to right and 90 is bottom to top.
It must be multiple of 45. Default is 0.

* android : centerX

Float. The relative x-position for the center of the gradient.

* android : centerY

Float. The relative Y-position for the center of the gradient.

* android : start-Color

color. The starting color.

* android : centerColor

Color. Optional color that comes between the start and end colors.

* android : endColor

Color. The ending color.

* android : gradientRadius

Float. The radius for the gradient only applied when "android : type = "radial".

* android : type

Keyboard. The type of gradient pattern to apply.

	Value	Description
①	"linear"	A linear gradient. This is the default.
②	"radial"	A radial gradient. The start color is the center color.
③	"sweep"	A sweeping line gradient.

Q. Example Program for gradient.

* CWProject51 *

- [res / drawable / button_color_shape . xml]

↳ button_color_shape . xml

→

```
<shape
    android : shape = "oval" >

<size  android : width = "100dp"
       android : height = "100dp" />

<gradient  android : angle = "90"
           android : startColor = "#FF0000"
           android : centerColor = "#00FF00"
           android : endColor = "#0000FF"
           android : centerX = "0.4"
           android : centerY = "1.0"  />

</shape>
```

— [res / layout / layout_btn . xml]

↳ layout_btn . xml

```
<RelativeLayout
    android : layout_width = "match_parent"
    android : layout_height = "match_parent" >

    <Button
        android : background = "@ drawable / button_color_shape" />

</RelativeLayout>
```

↳ ActivityButton . java

```
public class ActivityButton extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
```

→

```
        Super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_btn);
    }
}
```

↳ Android Manifeast. xml.

* Padding :

- <Padding>

Padding to apply to the containing view element. (this pads the position of the view content, not the shape).

↳ attributes :

* android : left
  Diamension. LeftPadding.

* android : top
  Diamension. TopPadding.

* android : right
  Diamension. RightPadding.

* android : bottom
  Diamension. BottomPadding.

Q. Example Program for Padding.

* CWProject52 *

- [res / drawable / button_padding.xml]

→

↳ button_padding . xml

```
<shape >
    <padding
        android : paddingLeft = "30 dp"
        android : paddingTop = "80 dp" />

    <solid android : color = "#FF8811" />
</shape>
```

— [res / layout / layout_padding . xml]

↳ layout_padding . xml

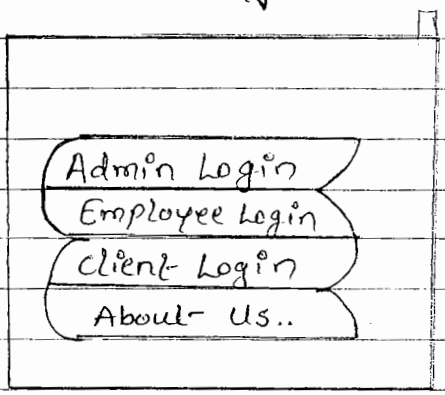
```
<RelativeLayout
    android : layout_width = "match_parent"
    android : layout_height = "match_parent" >

    <Button
        android : background = "@ drawable / button_padding" />

    <ImageButton
        android : background = "@ drawable / button_padding" />
</RelativeLayout >
```

↳ ActivityPadding . java.

```
public class ActivityPadding extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super . onCreate (savedInstanceState);
        setContentView ( R. layout. layout_padding);
    }
}.
```

Q. Example Program for Button Shape.

* GWProjed-53 *



← [res / layout / layout_btn.xml]

↳ layout_btn.xml

```
<LinearLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:orientation = "vertical"
    android:background = "@drawable/layout_shape"
    android:gravity = "center" >

    <Button
        android:id = "@+id/btn1"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Admin Login"
        android:background = "@drawable/button_shape1" />

    <Button
        android:id = "@+id/btn2"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Employee Login"
        android:background = "@drawable/button_shape2" />
```

```
<Button
    android : id = "@+id / btn3
    android : layout_width = "match_parent"
    android : layout_height = "wrap_content"
    android : text = "Client Login"
    android : background = "@drawable / button_shape1"

<Button
    android : id = "@+id / btn4
    android : layout_width = "match_parent"
    android : layout_height = "wrap_content"
    android : text = "About us"
    android : background = "@drawable / button_shape2" />

</LinearLayout>
```

— [res / drawables / layout_shape . xml]

↳ layout_shape . xml

```
< shape   android : shape = "rectangle" >

    < corners   android : topLeftRadius = "80 dp"
                android : bottomRightRadius = "80 dp" />

    < solid   android : color = "#FF8811" />
</ shape >
```

— [res / drawable / button_shape1 . xml]

↳ button_shape1 . xml

```
< shape >
    < corners   android : topLeftRadius = "40 dp"
                android : bottomRightRadius = "40 dp" />
```

```
<solid android:color="#CEF6F5" />

<stroke android:width="3dp"
        android:color="#000000" />
</shape>
```

— [res / drawable / button_shape2.xml]

↳ button_shape2.xml

```
<shape>
    <corners android:topRightRadius="40dp"
             android:bottomLeftRadius="40dp" />

    <solid android:color="#F3F781" />

    <stroke android:width="3dp"
            android:color="#000000" />
</shape>
```

— [src / (PackageName) / ActivityButton.java]

↳ ActivityButton.java

```
public class ActivityButton extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanseState);
        setContentView (R.layout.layout_btn);
    }
}
```

— Android Manifeast.xml

Q. Example Program for Ring Shape Button.

* CWProject-54 *

— [res / drawable / layout_shape.xml]

↳ layout_shape.xml

```
<shape
    android:shape = "ring"
    android:innerRadius = "100dp"
    android:thickness = "50dp"
    android:useLevel = "false" >

    <solid android:color = "#FF8811" />
</shape>
```

— [res / layout / layout_ring.xml]

↳ layout_ring.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:background = "@drawable / layout_shape" />

    <ImageButton
        android:id = "@+id / imgbtn1"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:background = "#FF8811"
        android:src = "@drawable / ic_action_add_grp" />
```

→

```
<ImageButton
    android : id = "@+id/ imgBtn2"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : background = "@ #FF8811"
    android : src = "@drawable / ic_action_place" />

<ImageButton
    android : id = "@+id/ imgBtn3"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : background = "#FF8811"
    android : src = "@drawable / ic_action_directions" />

</RelativeLayout>
```

— [src / (packageName) / ActivityRing . java]

↳ ActivityRing . java

```
public class ActivityRing extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super . onCreate (savedInstanceState);
        setContentView ( R. layout. layout_ring );
    }
}
```

— Android Manifeast . xml .

Q. Example Program for Animation.

* CWProject-55 *

— NOTE : Refer to xerox notes.

* Fragments :

— Dividing one screen into multiple parts is known as fragments.

— Fragments are introduced in 3.x versions.

— A fragments represent a beheavior or a portion of User Interface in an Activity.

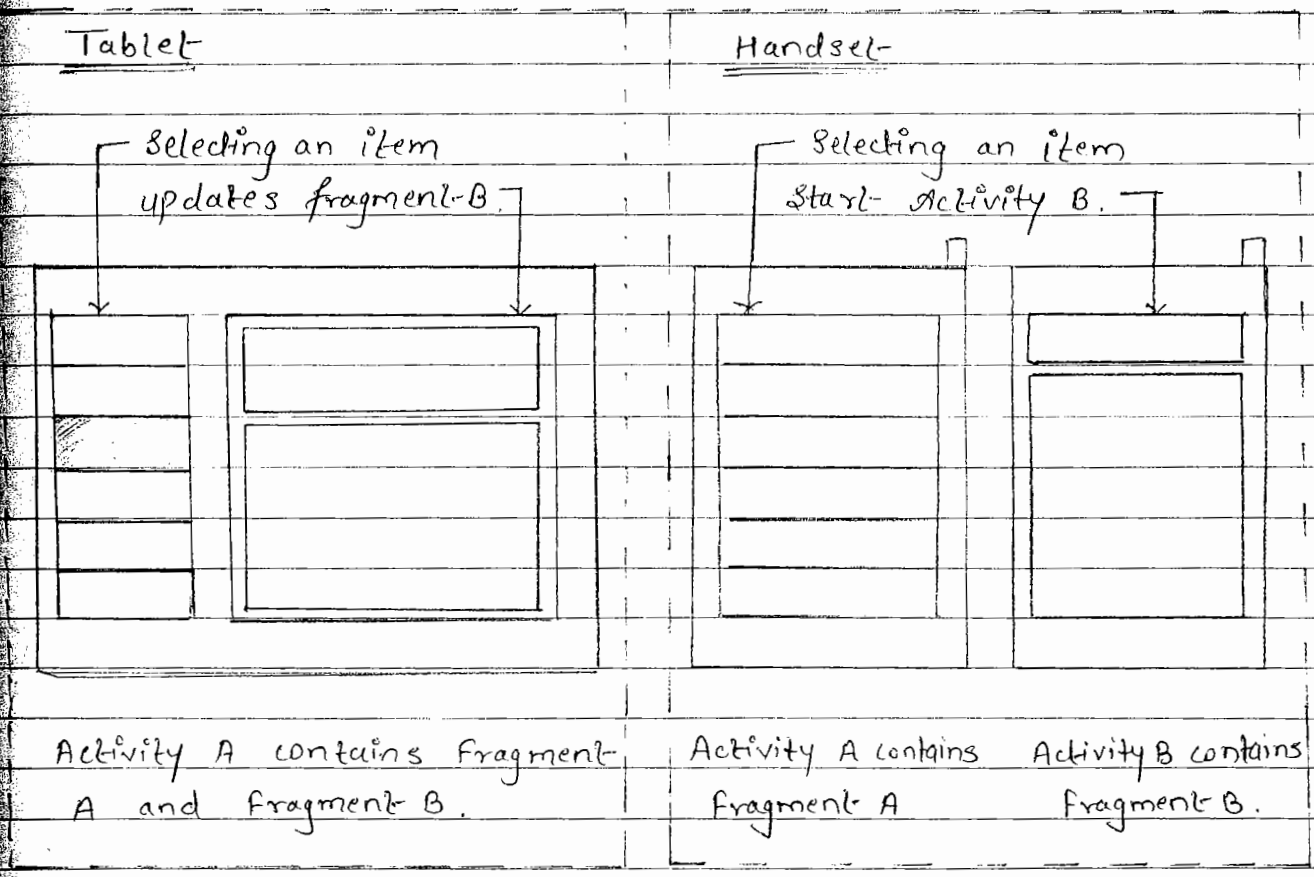
— You can combine multiple fragments in a single activity to build a multi-pane user Interface and re-use a fragment in multiple activities.

— A fragment must always be embeded in an activity.

Ex. When an activity is paused so, are all fragments in it and when an activity is destroyed so, are all fragments are destroyed.

— Android introduce fragments in API Level 11 and primarly to support more dianamic and flexible User Interface design on large screens such as Tablets.

— Diagram :



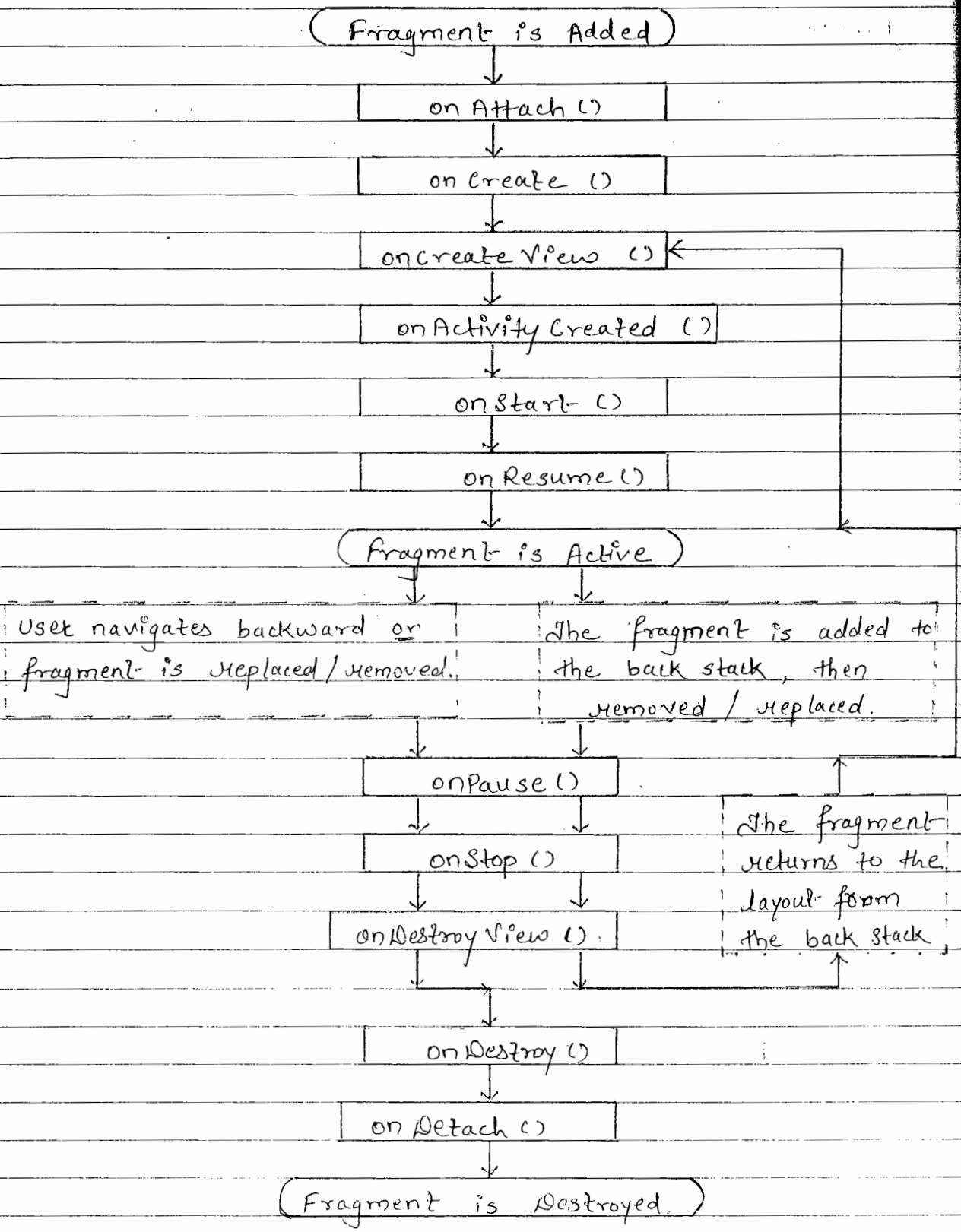
— Creating a fragments :

— To create a fragment we must create a sub-class of fragment class. (or an existing sub-class of it).

— The fragment class has code that- looks a lot like an activity.

— It contains call by methods similar to an activity as —
  - onCreate (),
  - onStart- (),
  - onPause (), etc.

— Fragment Life cycle :



	Methods	Description
1	onAttach ()	It is called only once when it is attached with activity.
2	onCreate (Bundle)	It is used to initialize the fragment.
3	onCreateView (LayoutInflator, ViewGroup, Bundle)	creates and returns view hierarchy.
4	onActivityCreated (Bundle)	It is invoke after the completion of onCreate () method.
5	onViewstateRestored (Bundle)	It provides information to the fragments that all the saved state of fragment view hierarchy has been restored.
6	onstart ()	Makes the fragment visible.
7	onResume ()	Makes the fragment interactive.
8	onPause ()	Is called when the fragment is no longer interactive.
9	onStop ()	Is called when fragment is no longer visible.
10	onDestroyView ()	Allows the fragment to clean up resources.
11	onDestroy ()	Allows the fragment to do final clean up of fragment state.

(12) onDetach() — It is called immediately prior to the fragment no longer being associated with its activity.

— Steps to create fragments :

(1) create a layout and divide the layout into required number of fragments.

(2) In fragment element we use class attribute to add fragment classes.

(3) A fragment is usually used as a part of the activity, User Interface and contributes its layout to the activity.

(4) To provide a layout for a fragment you must implement (override) "onCreateView()" method which ~~is~~ the android system calls when its type for fragment to draw its layout.

(5) Your implementation of onCreateView() method must return a view; that is the root of your fragment layout.

— Snippet :

```
public void onCreateView (LayoutInflater inflate,
                          ViewGroup container,
                          Bundle savedInstanceState)
{
    View v = inflater.inflate (R.layout.bodyFragment_layout,
                               container, false);
    return v;
}
```

- Switching from an fragment to another fragment :

- Snippet :

```
Fragment fragment = new Your_fragment_class ;

FragmentManager fragmentManager = getFragmentManager ();

FragmentTransaction fragmentTransaction =
                    fragmentManager . beginTransaction ();

fragmentTransaction . replace ( R.id. fragment_Id , fragment );

fragmentTransaction . addToBackStack (null);

fragmentTransaction . commit ();
```

- Description :

① FragmentManager :
- FragmentManager is a pre-defined interface of "android.app" package. This interface is for interacting with fragments objects inside of an activity.

② getFragmentManager :
- It is a pre-defined static method of fragment class which returns fragment-Manager reference for interacting with fragments associated with fragment Activity.

③ FragmentTransaction :
- It is a pre-defined interface of "android.app" package. This API for performing set of fragment operations.

④ BeginTransaction :

→

④ beginTransaction :
- It is a pre-defined instance method of fragmentManager Interface. This method start a series of edit operations on the fragments associated with fragmentManager.

⑤ replace (int containerViewId, Fragment fragment) :
- This method is used to replace a fragment in specified position.

⑥ addToBackStack (String name) :
- This method means that the transaction will be remembered after it is commited.

⑦ commit () :
- This method scheduled a commit of this transaction.

Q Example Program for fragment.

* CWProjectFragment *

- [res / layout / layout_main . xml]

↳ layout_main . xml.

```
<LinearLayout
    android : layout_width = "match_parent"
    android : layout_height = "match_parent"
    android : orientation = "vertical"
    android : background = "@drawable / # FF8811" >
```

→

```
<fragment
    android : width = "match_parent"
    android : layout_height = "~~match_parent~~ 100 dp"
    android : id = "@+id/headfragment"
    android : layout_margin = "10 dp" ⇔
    class = "com.nmk.cwproject.fragment.HeadFragment" />

<fregment
    android : layout_width = "match_parent"
    android : layout_height = "250 dp"
    android : id = "@+id/bodyfragament"
    android : layout_margin = "10 dp"
    class = "com.nmk.cwproject.fragment.BodyFragment" />

<fragment
    android : layout_width = "match_parent"
    android : layout_height = "50 dp"
    android : id = "@+id/footerfragment"
    android : layout_margin = "10 dp"
    class = "com.nmk.cwproject.fragment.FooterFragment" />

</LinearLayout>
```

— [src / (PackageName) / ActivityMain.java]

↳ ActivityMain.java

```
public class ActivityMain extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView ( R.~~int~~layout.layout_main );
    }
}
```

— [res / layout / head_fragment_layout.xml]

↳ head_fragment_layout.xml

```
<LinearLayout
    android:layout_width = "match_parent"
    android:layout_width height = "100dp"
    android:background = "#FACC2E"
    android:orientation = "vertical"
    android:margin = "10dp" >

</LinearLayout>
```

— [res / layout / body_fragment_layout.xml]

↳ body_fragment_layout.xml

```
<LinearLayout
    android:layout_width = "match_parent"
    android:layout_height = "250dp"
    android:background = "#00FFFF"
    android:orientation = "vertical"
    android:margin = "10dp" >

</LinearLayout>
```

— [res / layout / footer_fragment_layout.xml]

↳ footer_fragment_layout.xml

```
<LinearLayout
    android:layout_width = "match_parent"
    android:layout_height = "50dp"
    android:background = "#FF8800"
    android:orientation = "vertical"
    android:margin = "10dp" >
</LinearLayout>
```

— [src / (PackageName) / HeadFragment.java]

```
public Class HeadFragment extends Fragment
{
    @Override
    public void onCreateView (LayoutInflater inflater,
      viewGroup container, Bundle savedInstanceState)
    {
        View v = inflater.inflate (R.layout.head_fragment_layout
                , container, false);
        return v;
    }
}
```

— [src / (PackageName) / BodyFragment.java]

```
public ~~vo~~ Class BodyFragment extends Fragment
{
    @override
    public void onCreateView (LayoutInflater inflate,
      viewGroup container, Bundle savedInstanceState)
    {
        View v = inflater.inflate (R.layout.body_fragment_layout,
                container, false);
        return v;
    }
}
```

— [src / (PackageName) / FooterFragment.java]

```
public Class FooterFragment extends Fragment
{   @override
    public void onCreateView (LayoutInflater inflate,
      viewGroup container, Bundle savedInstanceState)
    {
```

→

```
        View v = inflater.inflate(R.layout.footer_fragment_layout,
                                  container, false);
        return v;
    }
}
```

- **Android Manifest.xml**

```
<uses-sdk
    android:minSDKVersion = "11"
    android:targetSDKVersion = "21" />

<application>
    // only Activity needs to be registered.
</application>
```

* Menu Resources :

- A menu resource defines an application menu [Options Menu, Context Menu or pop up Menu] that can be inflated with MenuInflater.

↳ **FILE LOCATION :**

res / menu / fileName.xml

↳ **RESOURSE REFERENCE :**

In Java : R.menu.filename
In xml : @[package:]menu.filename.

- In Android, we have three types of menu's -
  a) Option's Menu,
  b) Contextual Menu and
  c) Pop up Menu.

① Option Menu :

- Menu's are used in application to handle some common functionally around the application.

Ex. Settings is a common option item found in every application.

- To modify the default menu of the applications we need to modify
  "res \ menu \ activity_main.xml" file.

Q. Example Program for Option Menu.

* CWProject-56 *

- [res / menu / main.xml]

↳ main.xml

```
<menu>
    <item  android:id = "@+id/action_settings"
           android:orderInCategory = "100"
           android:title = "@string/action_settings"
           app:showAsAction = "never" />

    <item  android:id = "@+id/one"
           android:title = "camera" />

    <item  android:id = "@+id/two"
           android:icon = "@drawable/ic_action_refresh"
           android:title = "refresh"
           app:showAsAction = "ifRoom" />
```

→

```
    <item   android:id = "@+id/two1"
            android:icon = "@drawable/icon_fire"
            android:title = "fire"
            app:showAsAction = "always" />
</menu>
```

— [res / layout / layout_menu.xml]

↳ layout_menu.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

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

</RelativeLayout>
```

— [~~res~~ src / (package Name) / ActivityMenu.java]

↳ ActivityMenu.java

```
import android.support.v7.app.ActionBarActivity;
import android.view.Menu;
import android.view.MenuItem;

public Class ActivityMenu extends ActionBarActivity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_menu);
    }
```

```
@Override
public boolean onCreateOptionMenu (Menu menu)
{
    getMenuInflater().inflate(R.menu.main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected (MenuItem item);
{
    int id = item.getItemId();
    if (id == R.id.action_settings)
    {
        return true;
    }
    return super.onOptionsItemSelected(item);
}
}
```

— Android Manifeast.xml.

— Sub Menu's in Option Menu's :

— [res / menu / my_option_menu.xml]

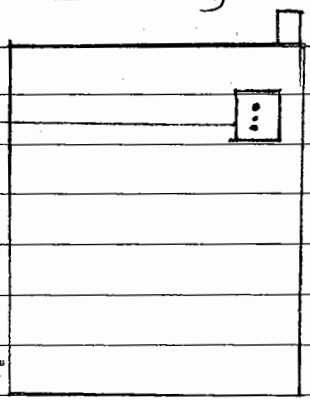
↳ my_option_menu.xml

```
<menu>
    <item android:id="@+id/one"
          android:title="File"
          app:showAsAction="always"
          android:setHeaderIcon="R.drawable.fire" >

        <menu>
            <item android:id="@+id/one1"
                  android:title="open" />
```

```
<item   android : id = "@+id / one2"
        android : title = "Close" />

<item   android : id = "@+id/ one3"
        android : title = " Save" />
</menu>
</item>
</menu>
```

— [res / layout / layout_option_menu . xml]

↳ layout_option_menu . xml




```
< RelativeLayout
    android : id = "@+id/ rl1 ;
    android : layout_width = "match_parent"
    android : layout_height = "match_parent"

    < ImageButton
        android : id = "@+id/ imgBtn1"
        android : layout_width = "wrap_content"
        android : layout_height = "wrap_content"
        android : src = "@ drawable / ic_action_overflow" />

</ RelativeLayout >
```

— [src / (packageName) / ActivityOptionMenu . java]

↳ ActivityOptionMenu . java .

```
import android . support . v7 . app . ActionBarActivity ;
import android . view . Menu ;
```

→

```
public class ActivityOptionMenu extends ActionBarActivity
{
    ImageButton iv;
    RelativeLayout rl;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_option_menu);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu)
    {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

- Android Manifeast.xml.

Q. Example Program for Options menu for changing Layout Background Color.

* CWProject57 *

- NOTE : While Creating Project, Select "Theme : Holo Light with Dark Action Bar."

- [res / menu / main.xml]

↳ main.xml.

→

```
<menu>
    <item  android:id = "@+id/red"
           android:title = "RED"
           android:showAsAction = "always" />

    <item  android:id = "@+id/green"
           android:title = "GREEN"
           android:showAsAction = "never" />

    <item  android:id = "@+id/blue"
           android:title = "BLUE"
           android:showAsAction = "ifRoom" />
</menu>
```

— [res / layout / layout_color.xml]

↳ layout_color.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:id = "@+id/rl1" >
</RelativeLayout>
```

— [src / (packageName) / Activity_color.java]

↳ ActivityColor.java

```
public class ActivityColor extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_color);
    }
}
```

```
@Override
public boolean onCreateOptionsMenu (Menu menu)
{
    getMenuInflater().inflate (R.menu.main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected (MenuItem item)
{
    int id = item.getItemId();
    RelativeLayout rl = (RelativeLayout) findViewById
                        (R.id.rl1);
    switch (id)
    {
        case R.id.red :
            rl.setBackgroundColor (Color.RED);
            break;

        case R.id.green :
            rl.setBackgroundColor (Color.GREEN);
            break;

        case R.id.blue :
            rl.setBackgroundColor (Color.BLUE);
            break;
    }
    return true;
}
}
```

— Android Manifeast.xml

Q. Example Program for Options Menus, by selecting an option it will display on Toast Dialog.

* CWProject 58 *

- [res / menu / main . xml]

↳ main . xml

```
<menu>
    <item  android:id = "@+id/one"
           android:title = "One"
           app:showAsAction = "always" />

    <item  android:id = "@+id/two"
           android:title = "Two"
           app:showAsAction = "ifRoom" />

    <item  android:id = "@+id/three"
           android:title = "Three"
           app:showAsAction = "never" />
</menu>
```

- [res / layout / layout_one . xml]

↳ layout_one . xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

</RelativeLayout>
```

- [~~res~~ src / (packagName) / ActivityOne . java]

↳ ActivityOne . java →

```
import android.support.v7.app.ActionBarActivity;

public Class ActivityOne extends ActionBarActivity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_one);
    }

    @Override
    public boolean onCreateOptionsMenu (Menu menu)
    {
        getMenuInflater().inflate (R.menu.main ~~main layout_one~~, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected (MenuItem item)
    {
        int id = item.getItemId ();
        switch (id)
        {
            case R.id.one :
                Toast.makeText (this, "one", 3000).show();
                break;

            case R.id.two :
                Toast.makeText (this, "two", 3000).show();
                break;

            case R.id.three :
                Toast.makeText (this, "Three", 3000).show();
                break;
        } return true;
    }
}
```

Q. Example Program for group Options Menu.

* GWProject-59 *

- [res / menu / main.xml]

↳ main.xml

```
< menu >
    <item   android:id = "@+id/Sample"
            android:title = "Sample"
            app:showAsAction = "ifRoom" />

    <group
        android:id = "@+id/color"
        android:checkableBehavior = "single"
        android:visible = "true" >

        <item   android:id = "@+id/red"
                android:title = "RED"
                android:showAsAction = "always" />

        <item   android:id = "@+id/green"
                android:title = "GREEN"
                app:showAsAction = "always" />

        <item   android:id = "@+id/blue"
                android:title = "BLUE"
                app:showAsAction = "always" />
    </group>
</menu>
```

- [res / layout / layout_one.xml]

↳ layout_one.xml

→

```
< RelativeLayout
    android : layout_width = "match_parent"
    android : layout_height = "match_parent" >

</ RelativeLayout >
```

— [src / (packageName) / ActivityOne.java]

↳ ActivityOne.java

```
import android.support.v7.app.ActionBarActivity;
import android.view.MenuItem;

public class ActivityOne extends ActionBarActivity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_one);
    }

    @Override
    public boolean onCreateOptionMenu (Menu menu)
    {
        getMenuInflater().inflate (R.menu.main, menu);
        return true;
    }
}
```

— Android Manifeast.xml

8. Example Program for Item with Icon.

* CWProject-60 *

- **NOTE** : Copy Png images into drawables.

- [res / menu / main.xml]

↳ main.xml

```
< menu >
    < item   android:id = "@+id/one"
             android:icon = "@drawable/car"
             android:title = "car"
             app:showAsAction = "always" />

    < item   android:id = "@+id/two"
             android:icon = "@drawable/icon_fire"
             android:title = "Fire"
             app:showAsAction = "always" />

    < item   android:id = "@+id/three"
             android:icon = "@drawable/myicon"
             android:title = "MyIcon"
             app:showAsAction = "always" />

    < item   android:id = "@+id/four"
             android:icon = "@drawable/rat"
             android:title = "Rat"
             app:showAsAction = "always" />
</ menu >
```

- [res / layout / layout_icon.xml]

↳ layout_icon.xml →

```
< RelativeLayout-
    android : id = "@+id / rl1"
    android : layout_width = "match_parent"
    android : layout_height = "match_parent" >

</ RelativeLayout->
```

— [src / (packageName) / ActivityIcon.java]

↳ ActivityIcon.java.

```
import android.support.v7.app.ActionBarActivity;
import android.view.MenuItem;

public class ActivityIcon extends ActionBarActivity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState)
        setContentView (R.layout.layout_icon);
    }

    @Override
    public boolean onCreateOptionsMenu (Menu menu)
    {
        getMenuInflater().inflate (R.menu.main, menu);
        return true;
    }
}
```

— Android Manifest.xml

(2) Contextual Menu's :

— Android context menu will appear when user press long click on the element. This menu is also known as Floating Menu.

— Methods :

① onCreateContextMenu (ContextMenu menu, View v, ContextMenuInfo menuInfo)	• This Method is called when the screen is long pressed and context-Menu view being build.
② add (int groupId, int itemId, int order, charSequence title)	• This method is used to add items to the menu. • This item displays the given title for its label.

↳ Parameters :

- groupId : The group identifier that this item should be a part of.
- itemId : Unique item id.
- order : The order of the item.
- title : The text to display for the item.

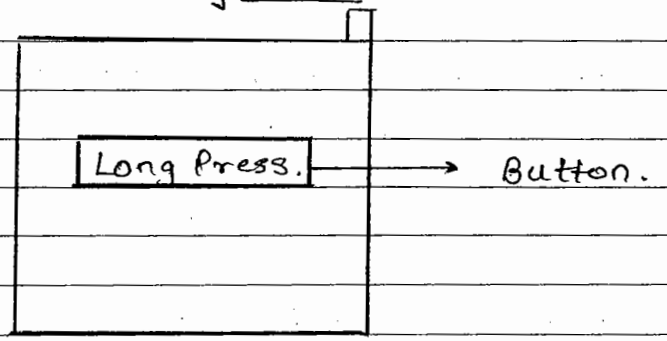
);

③ onContextItemSelected (MenuItem item)	• This Method is called whenever an item in the context menu is selected, and the parameters menuItem will provide necessary details on selected item.

(4) registerForContextMenu (View v) — This method is used to register a component to a context-Menu.

Q. Example Program for ContextMenu.

* CWProject-61 *



— layout_contextmenu.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <Button
        android:id = "@+id/btn1"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Long Press"
        android:onClick = "display" />

</RelativeLayout>
```

— ActivityContextMenu.java

```
import android.view.ContextMenu;
import android.view.MenuItem;
import android.view.ContextMenu.ContextMenuInfo;
import android.view.Views;
```

```
public class ActivityContextMenu extends Activity
{
    Button b1;

    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_contextMenu);
        b1 = (Button) findViewById (R.id.btn1);
        registerForContextMenu (b1);
    }

    @Override
    public void onCreateContextMenu (ContextMenu menu,
        View v, ContextMenuInfo menuInfo)
    {
        menu.setHeaderTitle ("Colours");
        menu.setHeaderIcon ("R.drawable.icon_fire");
        menu.add (1, 11, 1, "RED");
        menu.add (1, 22, 2, "GREEN");
        menu.add (1, 33, 3, "BLUE");
    }

    @Override
    public boolean onContextMenuItemSelected (MenuItem
                                              item)
    {
        return true;
    }

    public void display (View v)
    {
        Toast.makeText (this, "Button Clicked", 3000).show();
    }
}
```

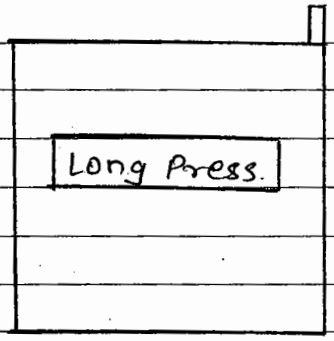
— Sub Menu's for Context Menus :

Q. Example Program for Sub Menus in Context Menus.

* CWProject-62 *

— ~~[res / menu / main.xml]~~

↳ ~~main.xml~~.



— layout_menu . xml

```
< RelativeLayout-
    android : layout_width = " match_parent"
    android : layout_height = " match_parent" >

    < Button
        android : id = " @+id / btn1 "
        android : layout_width = " wrap_content "
        android : layout_height = " wrap_content"
        android : text = " Long Press " />

</ RelativeLayout- >
```

— ActivityMenu . java

```
import android . view . ContextMenu ;
import android . view . subMenu ;
import android . view . View ;
import android . view . ContextMenu . ContextMenuInfo ;

public Class ActivityMenu extends Activity
{
    Button b1 ;
```

→

```
@Override
protected void onCreate (Bundle savedInstanceState)
{
    Super. oncreate (savedInstanceState);
    setContentView (R. layout. layout_menu);

    btn1 = (Button) findViewById (R. id. btn1);
    registerforContextMenu (btn1);
}

@Override
public void onCreateContextMenu (ContextMenu menu,
    View v, ContextMenuInfo menuInfo);
{
    menu. setHeaderTitle ("Menu Options");
    menu. setHeaderIcon (R. drawable. car);

    SubMenu ref = menu. addsubMenu ("File");
            ref. setHeaderIcon (R. drawable. myicon);
            ref. add (1, 11, 1, "New");
            ref. add (1, 22, 2, "Open");
            ref. add (1, 33, 3, "Save");
            ref. add (1, 44, 4, "close");

    SubMenu ref1 = menu. addsubMenu ("Edit");
            ref1. setHeaderIcon (R. drawable. mario);
            ref1. add (1, 11, 1, "cut");
            ref1. add (1, 22, 2, "copy");
            ref1. add (1, 33, 3, "paste");
}
}
```

— Android Menifeast. xml.

(3) Popup Menu :

- popup menu is used to display global actions, popup menu is an overflow menu like Spinner actions.

- popup menu is avaible for API Level - 11 ie. Android 3.0.

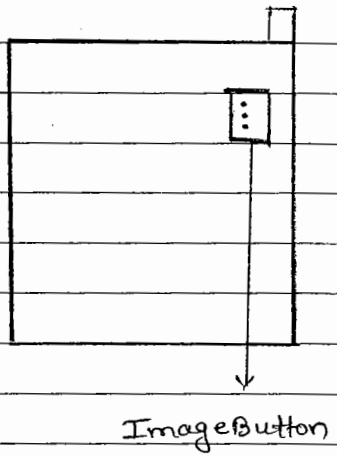
Q. Example Program for Popup Menu.

* CWProject-63 *

- NOTE : Change minSDKVersion = "11" in Manifeast file.

- [res / menu / main.xml]

```
<menu>
    <item android:id = "@+id/one"
          android:title = "RED" />

    <item android:id = "@+id/Two"
          android:title = "GREEN" />

    <item android:id = "@+id/Three"
          android:title = "BLUE" />
</menu>
```



- [res / layout-/ layout-popup.xml]

↳ layout-popup.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:id = "@+id/rl1" />
```

```
<ImageButton
    android : id = "@+id / iv"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : src = "@drawable / ic_action_overflow" />
</RelativeLayout>
```

— [src / (packageName) / ActivityPopUp . java]

↳ ActivityPopUp . java

```
import android.support.v7.app.ActionBarActivity;
import android.view.View;
import android.view.MenuItem;
import android.view.MenuItem;
import android.view.View.onClickListener;
import android.widget.ImageButton;
import android.widget.popupMenu;
import android.widget.PopupMenu.onMenuItemClickListener;

public class ActivityPopup extends ActionBarActivity
{
    ImageButton iv;
    RelativeLayout rl;

    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_popup);

        rl = (RelativeLayout) findViewById (R.id.rl1);
        iv = (ImageButton) findViewById (R.id.iv);
```

→

```
iv.setOnClickListener(new OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        popup pm = new popupMenu(getApplicationContext(),
                                 iv);
        pm.getMenuInflater().inflate(R.menu.main,
                                     pm.getMenu());

        pm.setOnMenuItemClickListener(new
                                      onMenuItemClickListener()
        {
            @Override
            public boolean onMenuItemClick(MenuItem item)
            {
                int id = item.getItemId();
                switch(id)
                {
                    case R.id.One:
                        rl.setBackgroundColor(Color.RED);
                        break;

                    case R.id.Two:
                        rl.setBackgroundColor(Color.GREEN);
                        break;

                    case R.id.Three:
                        rl.setBackgroundColor(Color.BLUE);
                }
                return true;
            }
        });
        pm.show();
    }
});
}
}
```

— SubMenu's For PopUp Menu's :

Q. Example Program for SubMenu's in Popup Menus.

— [res / Menu / my_popup.xml]

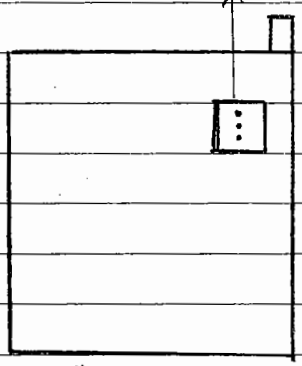
↳ my_popup.xml

```
<menu>
    <item android:id="@+id/one"
          android:title="File.">
        <menu>
            <group android:checkableBehaviour="single">
                <item android:id="@+id/one1"
                      android:title="New" />
                <item android:id="@+id/one2"
                      android:title="Open" />
                <item android:id="@+id/one3"
                      android:title="close" />
            </group>
        </menu>
    </item>

    <item android:id="@+id/Two"
          android:title="Edit-">
        <menu>
            <group android:checkableBehaviour="all">
                <item android:id="@+id/Two1"
                      android:title="cut-" />
                <item android:id="@+id/Two2"
                      android:title="copy" />
```

→

```
        <item  android:id = "@+id/Two3"
               android:title = "Paste" />
    </group>
  </menu>
</item>

<item  android:id = "@+id/Three"
       android:title = "Refactor" >
  <menu>
    <group  android:checkableBehaviour = "none" >
      <item  android:id = "@+id/three1"
             android:title = "Re1" />
      <item  android:id = "@+id/three2"
             android:title = "Re2" />
      <item  android:id = "@+id/three3"
             android:title = "Re3" />
    </group>
  </menu>
</item>
</menu>.
```

ImageButton.



— [res/layout-/layout_popup.xml]

↳ layout_popup.xml.

```
< RelativeLayout-
    android:id = "@+id/rl1"
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

  < ImageButton
      android:id = "@+id/iv"
      android:layout_width = "wrap_content"
      android:layout_height = "wrap_content"
      android:src = "@drawable/ic_action_overflow" />
</RelativeLayout >
```

— [src / (packageName) / ActivityPopup.java]

↳ ActivityPopup.java

```
import android.support.v7.app.ActionBarActivity;
import android.view.View;
import android.view.MenuItem;
import android.view.Menu;
import android.view.view.onClickListener;
import android.widget.popupMenu;
import android.widget.popupMenu.onMenuItemClickListener;

public Class ActivityPopup. extends ActionBarActivity
{
    ImageButton iv;
    RelativeLayout rl;

    @Override
    protected void onCreate (Bundle savedInstanceState)
{
    Super.onCreate (savedInstanceState);
    setContentView (R.layout.layout_popup);

    iv = (ImageButton) findViewById (R.id.iv);
    rl = (RelativeLayout) findViewById (R.id.rl);

    iv.setOnClickListener (new onclickListener ()
{
    @Override
    public void onClick (View v)
    {
        popupMenu pm = popupMenu (getApplicationContext(),
                                  iv);
        pm.getMenuInflater().inflate (R.menu.main,
                                      pm.getMenu());
```

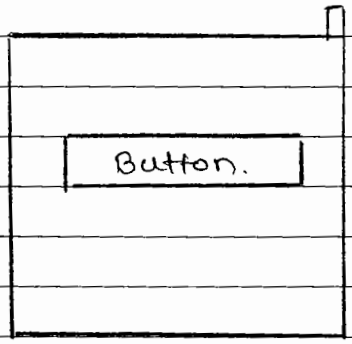
```
pm.setOnMenuItemClickListener ( new onMenuItemClickListener()
{
    @Override
    public boolean onMenuItemClick (MenuItem item)
    {
        int id = item.getItemId ();
        switch (id)
        {
            case R.id.one :
                rl.setBackgroundColor ( Color.RED) ;
                break ;

            case R.id.two :
                rl.setBackgroundColor (Color.GREEN) ;
                break ;

            case R.id.two2 :
                rl.setBackgroundColor (Color.BLUE) ;
                break ;
        }
        return true ;
    }
});
    pm.show ();
  }
});
}
```

— Android Manifeast . xml.

(c) — Adding Components like checkboxes, EditText, radioButton etc to AlertDialog :

* CWProject-64 *

— [res/layout-/layout-button.xml]



— ActivityButton.java

```
public class ActivityButton extends Activity
{   Button b1;
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layoutButton);

        b1 = (Button) findViewById (R.id.b1);
        b1.setOnClickListener (new onclickListener ()
        {
            @override
            public void onClick (View v)
            {
                show ();
            }
        });
    }
    EditText et;
```

→

```
public void show ()
{
    AlertDialog.Builder ad = new AlertDialog.Builder (this);
    ad.setIcon (R.drawable.ic_launcher);
    ad.setTitle ("Enter OTP");

    et = new EditText (this);
    et.setWidth (100);
    et.setHeight (50);
    et.setHint ("OTP");

    ad.setView (et);
    ad.setPositiveButton ("Yes", new DialogInterface.
                                    onClickListener ()
    {
        @Override
        protected public void onClick (DialogInterface dialog,
                                       int which)
        {
            Toast.makeText (getApplicationContext (),
                            et.getText ().toString (),
                            3000).show ();
        }
    });
    ad.show ();
}
}
```

— Android Manifest.xml

* ListView :

this);

- Android ListView is a View which groups several items and display them in vertical Scrollable List.
- The list items are automatically inserted to the list using an Adapter.
- An adapter which bridges between User Interface Component and data source, that fills data in User Interface component.
- **NOTE** : The ListView and GridView of subclasses of AdapterView.

Q. Example Program for ListView.

* CWProject-65 *

- [res / layout / layout_list.xml]

↳ layout_list.xml

```
<RelativeLayout
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    <ListView
        android:id = "@+id/l11"
        android:layout_width = "match_parent"
        android:layout_height = "match_parent" />

</RelativeLayout>
```

— [res / values / strings.xml]

```
<resource

    <string-array  android:name = "elements" >
        <item> RED </item>
        <item> GREEN </item>
        <item> BLUE </item>
        <item> YELLOW </item>
        <item> WHITE </item>
        <item> BLACK </item>
        <item> ORANGE </item>
    </string-array>

</resources>
```

— [src / (package Name) / ActivityList.java]

↳ ActivityList.java.

```
public class ActivityList extends Activity
{
    ListView lv;

    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_list);

        lv = (ListView) findViewById (R.id.lv1);

        Resources res = getResources ();

        String colors[] = res.getStringArray (R.array.elements);
```

```
ArrayAdapter aa = new ArrayAdapter (this,
        android.R.layout.simple_dropdown_list, colors);
lv.setAdapter (aa);

lv.setOnItemClickListener (new onItemClickListener ()
{
    @Override
    public void onItemClick (AdapterView <?> parent,
                View v, int position, long id)
    { String item = lv.getItemAtPosition (args0).toString ();
      Toast.makeText (getApplicationContext (), "selected"+item
                3000).show ();
    }
});

registerForContextMenu (lv);
}

@Override
public void onCreateContextMenu (ContextMenu menu,
        View v, ContextMenuInfo menuInfo)
{
    menu.add ("cut");
    menu.add ("copy");
    menu.add ("paste");
    menu.add ("Delete");
}
}
```

— Android Manifest.xml.

```
its);
```

* GridView :

- Android GridView show items in two-diamensional scrolling Grid (Rows & columns) and Grid items are not necessarily pre-determined but they automatically inserted to the layout using a ListAdapter.

↳ **GridView Attributes :**

(1) android : id — This is the Id which uniquely identifies the layout.

(2) android : columnWidth — This specifies the fixed width of each column.
This could be in px, ~~dep~~ dp.

(3) android : gravity — Specifies the gravity within each cell. possible values are top, bottom, left, right, center, center_vertical.

(4) android : horizontal Spacing — Defines the default horizontal spacing between columns.
This could be in px, dp.

(5) android : numColumn — Defines how many columns to show. May be an integer value, such as "100" or auto_fit which means display as many columns as possible to fill the available spaces.

~~(6) android: StretchMode~~

(6) android : stretchMode

none : Stretching is disabled.

Spacing Width : Spacing between each column is stretched.

→

columnWidth : Each column is stretched equally.
spacingWidthUniform : The spacing between each column is uniformally stretched.

⑦ android:verticalSpacing — Defines the default vertical spacing between rows. This could be px, dp.

↳ BaseAdapter :
[Android.widget.BaseAdapter]

- BaseAdapter is super most class of all Adapter classes.
- When we are extending BaseAdapter class we must provide implementation to unimplemented methods of BaseAdapter class.
- ArrayAdapter is a class which can work with array of Data.
- ListAdapter is an Interface implemented by concreate Adapter class.

Q. Example Program for GridView.

* CWProject-66 *

- [res / layout / layout_grid.xml]

```
< GridView
    android:id = "@+id/gridview"
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:columnWidth = "90 dp"
    android:numColumns = "auto_fit"
```

→

```
        android:verticalSpacing = "10dp"
        android:horizontalSpacing = "10dp"
        android:stretchMode = "columnWidth"
        android:gravity = "center" />
</GridView>
```

— ActivityGrid.java

```
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.onItemClickListener;
import android.widget.GridView;

public Class ActivityGrid extends Activity
{   Protected void onCreate(Bundle savedInstanceState)
    {
        Super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_grid);

        GridView gv = (GridView) findViewById(R.id.gridview);
        gv.setAdapter(new MyImagesAdapter(this));

        gv.setOnItemClickListener(new onItemClickListener()
        {
            @override
            public void onItemClick(AdapterView<?> parent,
                    View view, int position, long id)
            {
                Toast.makeText(getApplicationContext(), " "+position,
                                3000).show();
            }
        });
    }
}
```

— [src / (PackageName) / MyImagesAdapter.java]

→

↳ MyImagesAdapter.java

```
import android.content.Context;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.GridView;
import android.widget.ImageView;

public class MyImageAdapter extends BaseAdapter
{
    Context context;
    int images[] = {
        R.drawable.img1 , R.drawable.img2 ,
        R.drawable.img3 , R.drawable.img4 ,
        R.drawable.img5 , R.drawable.img6 ,
        R.drawable.img7 , R.drawable.img8 ,
        R.drawable.img9 , R.drawable.img10 };

    // Construct-or.
    public MyImagesAdapter (Context ctx)
    {
        this.context = ctx;
    }

    @Override
    public int getCount ()
    {
        return images.length;
    }

    @Override
    public object getItem (int position)
    {
        return images [position];
    }
```

→

```
@Override
public long getItemId(int position)
{
    return 0;
}

@Override
public View getView(int position, View convertView,
                    ViewGroup parent)
{
    ImageView iv = new ImageView(context);
    iv.setImageResource(images[position]);
    iv.setScaleType(ImageView.ScaleType.center_crop);
    iv.setLayoutParams(new GridView.LayoutParams(70,70));
    return iv;
}
}
```

— Android Manifest.xml

* Wireless Networks :

- Bluetooth and Wifi comes under the Wireless Networks.

- (1) BLUETOOTH :
  - It is a way to exchange data with other Wireless devices.
  - Android provides Bluetooth API to perform several operations, such as -
    a) Scan Bluetooth Devices,
    b) Content and data transfer from one device to another.
    c) Managing multiple connections etc.
  - Android provides "Android.Bluetooth.BluetoothAdapter" class to communicate with bluetooth.
  - We cannot create object to the BluetoothAdapter class directly, To do this process we use, "getDefaultAdapter()" method.
  - "getDefault-Adapter()" method is a static method of BluetoothAdapter class, which returns reference of BluetoothAdapter.
  - Ex
```
BluetoothAdapter ba = BluetoothAdapter.getDefaultAdapter();
```
  - In order to enable the bluetooth of your device call Intent with "ACTION-REQUEST_ENABLE" BluetoothAdapter class constant.
  - Ex
```
Intent turnOn = new Intent(BluetoothAdapter.
                ACTION_REQUEST_ENABLE);
startActivity(turnOn);
```

- once you enable the ~~bt~~ bluetooth you can get a list of paired devices by calling "getBondedDevices()". This method returns a "Set of Paired Devices".

- Ex.
```
Set<BluetoothDevices> pairedDevices = BluetoothAdapter.
                                      .getBondedDevices();
```

- BluetoothAdapter class :

↳ Methods	Description
① Public Boolean isEnabled()	• Returns true if the bluetooth is currently enabled and ready for ~~to~~ use.
② Public Boolean enable()	• Turn On the local BluetoothAdapter.
③ public Boolean disable()	• Turn off the local BluetoothAdapter.
④ public Set<BluetoothDevices> getBondedDevices();	• Returns the set of Bluetooth ~~objects~~ Devices objects that are Bonded (paired) to the local Adapter.
⑤ public Boolean isDiscovering()	• Returns true if the ~~local~~ local BluetoothAdapter is currently in the device discovery process.
⑥ public Boolean startDiscovery()	• Starts the remote device discovery process.
↳ public String getName()	• gets the friendly bluetooth name of the local Bluetooth Adapter.

2→ public BDolean setName (String name)

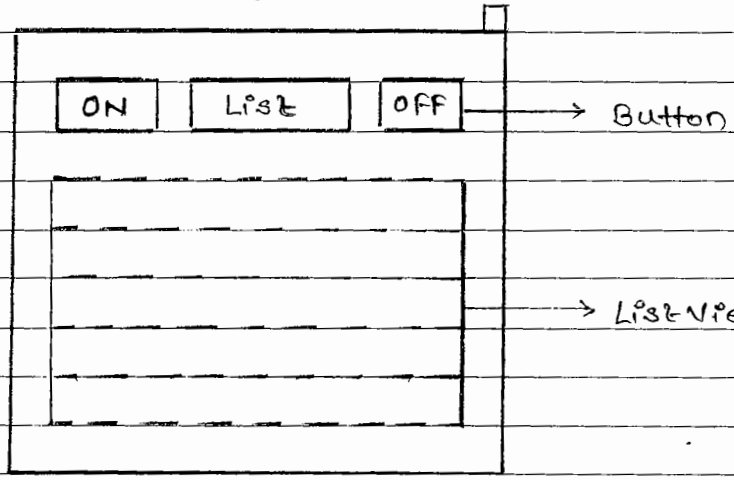
- Sets the friendly Bluetooth name to the local Bluetooth Adapter.
- This name is visible to remote Bluetooth Devices.

— Uses-permission :

```
< uses-permission android: ~~permission~~ name = "android.permission.BLUETOOTH" />

< uses-permission android : name = "android.permission.BLUETOOTH_ADMIN" />
```

Q. Example Program for Bluetooth.

* CWProject-67 *



— ActivityBluetooth.java.

```
import java.util.ArrayList;
import java.util.Set;
import android.widget.ArrayAdapter;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.Intent;
```

```
public class ActivityBluetooth extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.oncreate (savedInstanceState);
        setContentView (R.layout.layout_bluetooth);
        ba =. BluetoothAdapter.getDefaultAdapter ();
    }

    BluetoothAdapter ba;
    Set <BluetoothDevice> device;
    ListView lv;

    public void on (View v)
    {
        if (!ba.isEnabled ())
        {
            Intent i = new Intent (BluetoothAdapter.
                                   ACTION_REQUEST_ENABLE);
            startActivity (i);
        }
        else
        {
            Toast.makeText (this, "BT is On", 3000).show();
        }
    }

    public void list (View v)
    {
        lv = (ListView) findViewById (R.id.lv1);

        device = ba.getBondedDevices ();

        ArrayList list = new ArrayList ();
```

```
        for (BluetoothDevice bef : device)
        {
            list.add (bef.getName());
        }

        ArrayAdapter aa = new ArrayAdapter (this, android.R.
                layout.simple_dropdown_item_1line, list);
        lv.setAdapter (aa);
    }

    public void off (View v)
    {
        ba.disable ();
    }
}
```

— Android Manifeast.xml

```
<uses-permission android:name = "android.permission.
                                BLUETOOTH" />

<uses-permission android:name = "android.permission.
                                BLUETOOTH_ADMIN" />

; <application
    ---------- ----------
    ---------- ----------
</application>
```

② WIFI :

— Android provides "android.net.wifi.WifiManager" class.

— We cannot create object to WifiManager class directly because, it is existing system service so, we use, "getSystemService()" method.

— Ex.

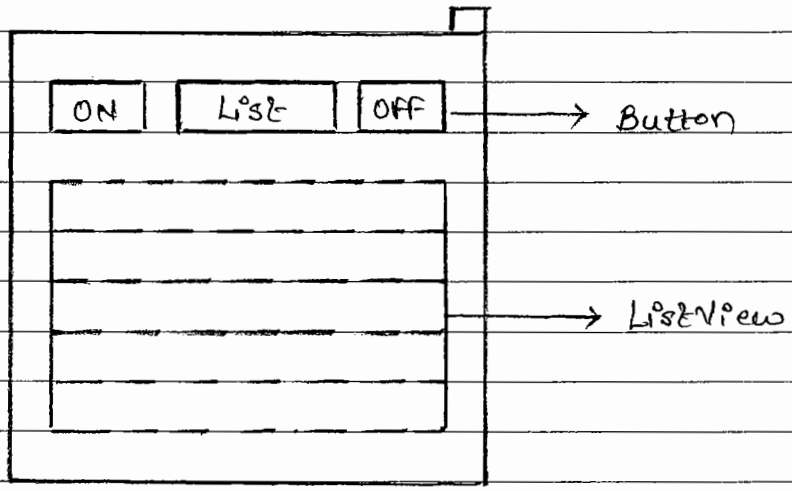
```
WifiManager wifi = (WifiManager) getSystemService
                   (context.WIFI_SERVICE);
wifi.setWifiEnabled(true);
wifi.setWifiEnabled(false);
```

— uses-permission :

```
<uses-permission android:name = "android.permission.
                                ACCESS_WIFI_STATE" />

<uses-permission android:name = "android.permission.
                                CHANGE_WIFI_STATE" />
```

Q Example Program for Wifi.

* CWProject-68 *



— ActivityWifi.java

```
import android.content.Context;
import android.net.wifi.WifiManager;

public class ActivityWifi extends Activity
{
    WifiManager wifi;
    ListView lv;

    public void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout-wifi);

        wifi = (WifiManager) getSystemService (Context.WIFI_SERVICE);
    }

    public void on (View v)
    {
        if (!wifi.isWifiEnabled ())
        {
            wifi.setWifiEnabled (true);
        }
        else
        {
            Toast.makeText (this, "Wifi is On", 3000).show ();
        }
    }

    public void off (View v)
    {
        wifi.setWifiEnabled (false);
    }
```

→

```
public void list (View v)
{
```

— Android Manifeast . java .

```
<uses-permission android:name = "android.permission.ACCESS_STATE_WIFI" />
<uses-permission android:name = "android.permission.CHANGE_WIFI_STATE" />

<application>
    <activity>
        <intent_filter>
            <action android:name = "android.intent.action.MAIN" />
            <category android:name = "android.intent.category.LAUNCHER" />
        </intent_filter>
    </activity>
</application>
```

* Android SeekBar :

- SeekBar is an extension of progress Bar that adds a dragable thumb.
- The user can touch the thumb and drag left to right to set the current progress level.
- SeekBar can attach a "seekBar.onSeekBarChangeListener" interface to notify the uses action.
- Syntax :

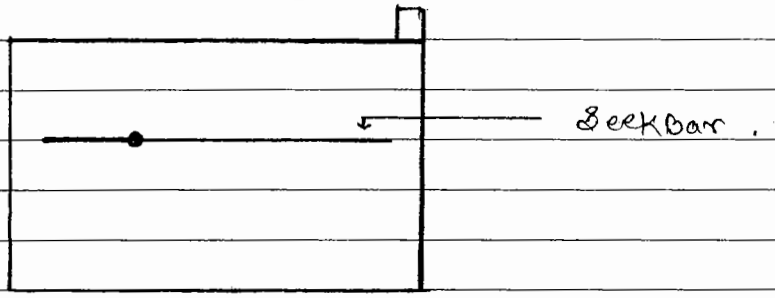
```
seekBar.~~on~~setOnSeekBarChangeListener (android.widget.SeekBar.
                                    OnSeekBarChangeListener)
```

↳ This interface callBack that notifies clients when the progress level has been changed.

Methods	Description
① onProgressChanged (SeekBar seekbar, int progress, boolean fromUser)	• Notifies when the progress level has changed.
② onStartTrackingTouch (SeekBar seekbar)	• Notifies when the user has started a touch gesture.
③ onStopTrackingTouch (SeekBar seekBar)	• Notifies when the user has finished a touch gesture.

Q. Example Program for SeekBar.

* CWProject-69 *



face

— layout_seek.xml

```
< RelativeLayout
    android : layout_width = " match_parent "
    android : layout_height = " match_parent " >

    < SeekBar
        android : id = " @+id / seek1 "
        android : layout_width = " match_parent "
        android : layout_height = " wrap_content "
        android : layout_margin = " 30dp "
        android : max = " 50 " />

</ RelativeLayout >
```

— NOTE : To change the SeekBar strength we use " android : max " attribute.

— ActivitySeek.java.

```
Import android.widget.SeekBar ;
Import android.widget.SeekBar.onSeekBarChangeListener ;
```

→

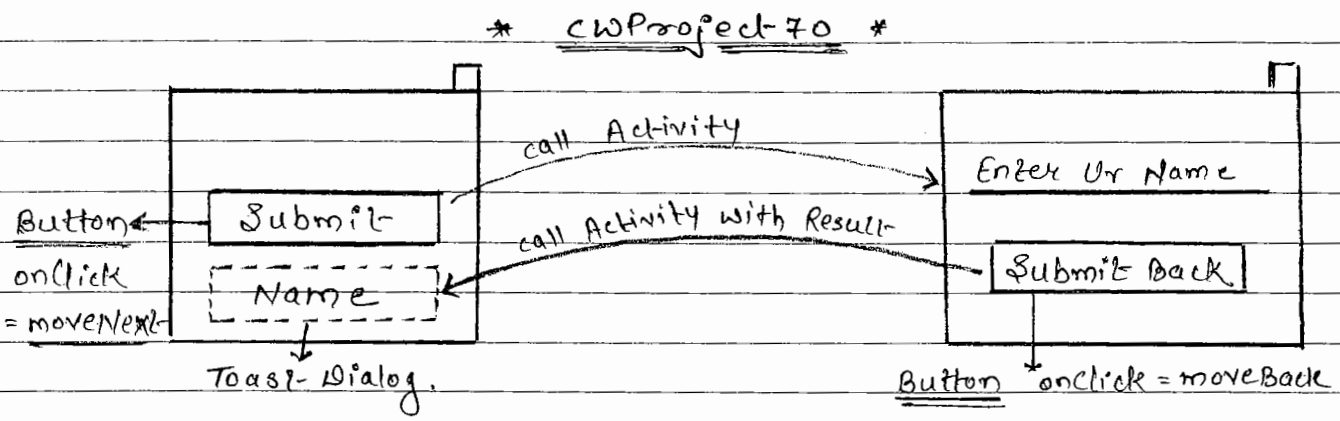
```
public class ActivitySeek extends Activity
{
    SeekBar sb;

    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_seek);

        sb = (SeekBar) findViewById (R.id.seek1);

        sb.setOnSeekBarChangeListener (new onSeekBarChange
                                       Listener ()
        {
            @Override
            public void onStopTrackingTouch (SeekBar seekBar)
            {
                Toast.makeText (getApplicationContext (), "STOP",
                                3000).show ();
            }

            @Override
            public void onStartTrackingTouch (SeekBar seekbar)
            {
                Toast.makeText (getApplicationContext (), "START",
                                3000).show ();
            }

            @Override
            public void onProgressChanged (SeekBar seekbar,
                    int progress, boolean fromUser)
            {
                Toast.makeText (getAapplicationContext (),
                    "Progress :" + progress, 3000).show ();
            }
        });
    }
}
```

* Switching With Results Data.
OR
* Start Activity With Results.

— By using startActivity for result method we can start a activity and we can send message (data) and we can receive message.

— To do this process we need to override "onActivityResult()" method.

Ex. Example Program for start-Activity With Results.



— **ActivityOne.java**

```
public Class ActivityOne extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_one);
    }
    public void moveNext (View v)
    {
        Intent i = new Intent (this, ActivityTwo.class);
        startActivityForResult (i, 2);
    }
```

```
@Override
protected void onActivityResult ( int requestCode ,
        int resultCode , Intent data)
{
    Super . onActivityResult ( requestCode , resultCode ,
                                data) ;

    if ( requestCode == 2)
    {
        String name = data. getStringExtra ("message");

        Toast. makeText ( this, " Hello : " + name , 3000). show()
    }
    else
    {
        Toast. makeText ( this, "No Result ", 3000 ) . show ();
    }
}
}
```

Android Manifeast . xml.

```
< application
    < activity > ____ </ activity >
    < activity > ____ </ activity >
</ application >
```

Activity Two . java.

```
public class ActivityTwo extends Activity
{
```

```
@Override
protected void onCreate (Bundle savedInstanceState)
{
    Super. onCreate (savedInstanceState);
    setContentView (R. layout. layout_two);
}

public void moveBack (View v)
{
    String mess = ((EditText) findViewById
        (R. id. et1). getText (). toString ());

    Intent i = new Intent ();
    i. putExtra ("message", mess);
    setResult (2, i);
    finish ();
}
}
```

Android Manifeast. xml

* DataBase *

- What is Data ?
- Data is a collection of Raw facts.

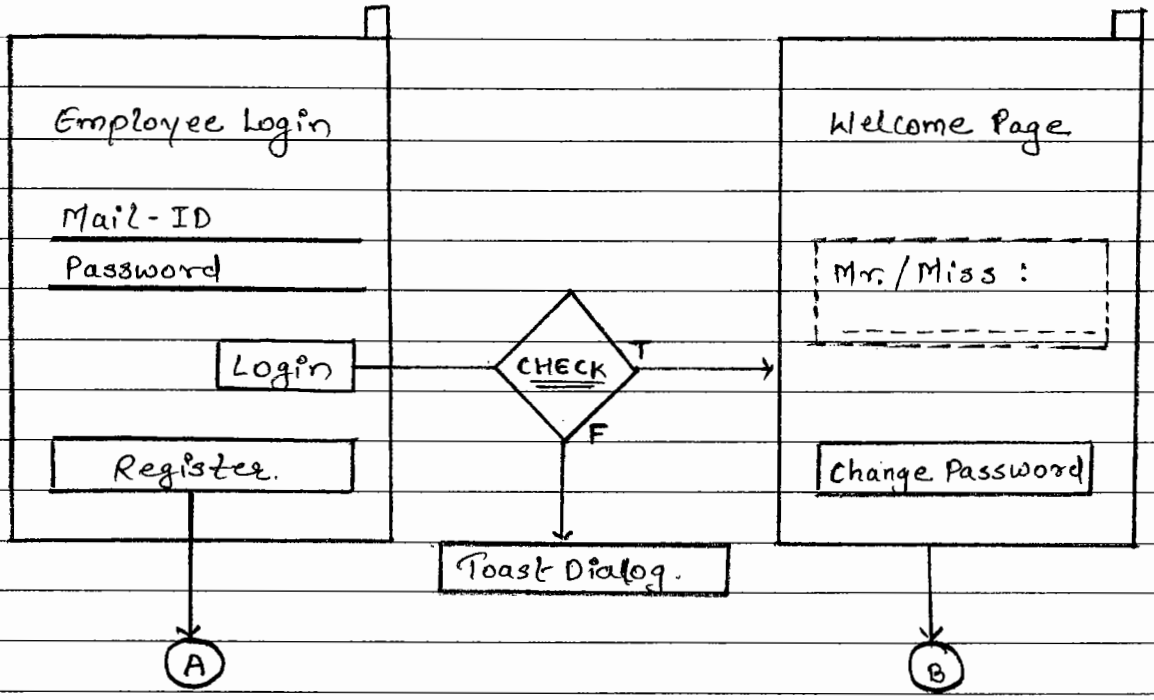
- What is Information ?
- Information is Data + Meaning which is known as Information.

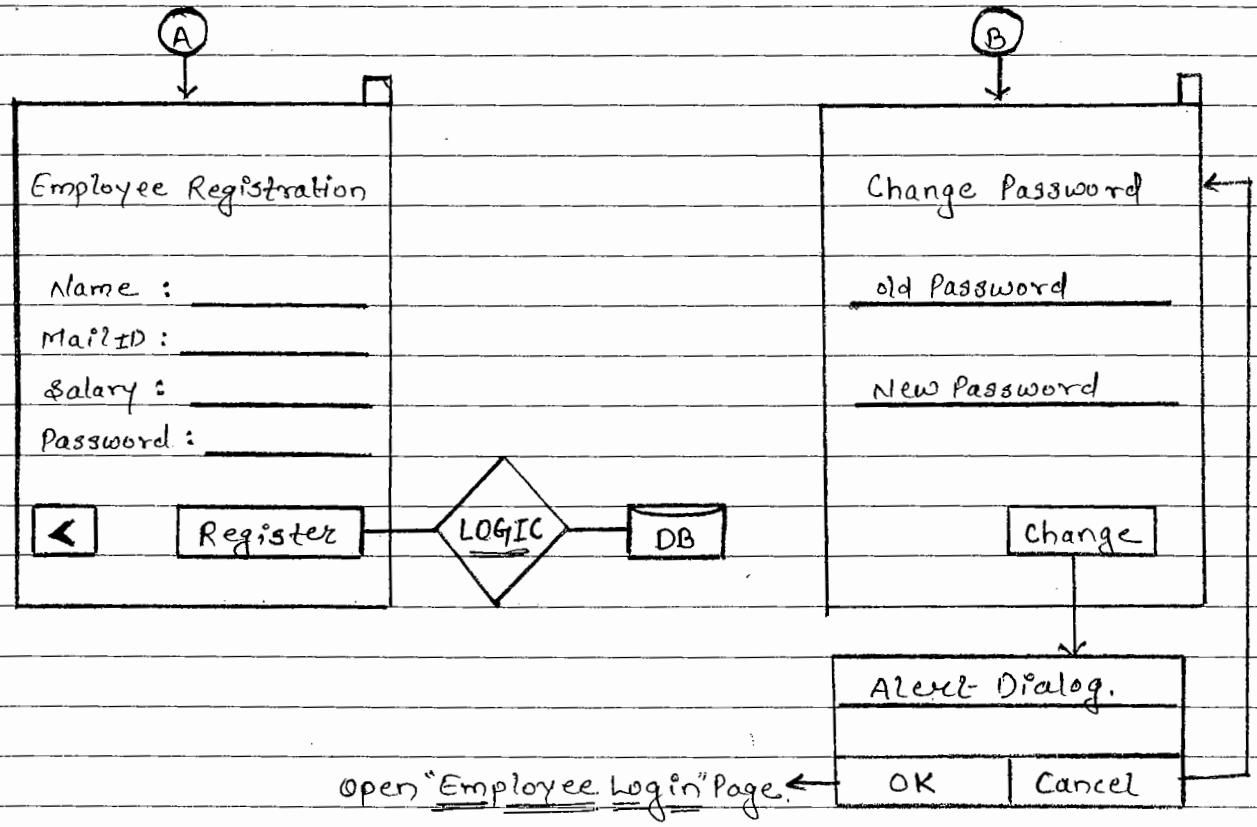
- DB → Employee_Data.

Name	Salary	Mail_ID	Password.

Q. Design an Android application for SQL Database Operations.

* CWProject-DB *





oms.

— ActivityEmployeeLogin . java

```
public Class ActivityEmployeeLogin extends Activity
{
    static DataBaseCreation dbc ;

    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super . onCreate (savedInstanceState);
        setContentView (R.layout. ~~login~~ layout_employee_login);
        dbc = CheckDB . createDB (this);
    }

    public static Class CheckDB
    {
        public static DataBaseCreation createDB (context context)
        {
```

→

```
        if (dbc == null)
        {
            dbc = new DataBaseCreation (context);
        }
        return dbc;
    }
}
public void checkDB (View v)
{
    EditText et1, et2;
    et1 = (EditText) findViewById (R.id.username);
    et2 = (EditText) findViewById (R.id.password);

    String email = et1.getText().toString();
    String pass = et2.getText().toString();

    if (dbc.loginCheck (email, pass))
    {
        Intent i = new Intent (this, EmployeeWelcomeActivity.
        i.putExtra ("mail", email);                    class);
        startActivity (i);
        finish ();
    }
}

public void register (View v)
{
    startActivity (new Intent (this, EmployeeRegistrationActivity.
    finish ();                                        class);
}
}
```

— EmployeeRegistrationActivity.java

```
public Class EmployeeRegistrationActivity extends Activity
{
    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        Super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_employee_register);
    }

    public void back (View v)
    {
        startActivity (new Intent (this, ActivityEmployeeLogin.class));
        finish ();
    }

    public void registerDB (View v)
    {
        EditText et1, et2, et3, et4;
        et1 = (EditText) findViewById (R.id.ename);
        et2 = (EditText) findViewById (R.id.email);
        et3 = (EditText) findViewById (R.id.esal);
        et4 = (EditText) findViewById (R.id.epass);

        String name = et1.getText().toString();
        String salary = et3.getText().toString();
        String mail = et2.getText().toString();
        String pass = et4.getText().toString();

        if (name.equals("") && mail.equals("") &&
            salary.equals("") && pass.equals(""))
        {
            Toast.makeText (this, "Empty Feilds", 3000).show();
        }
        else
```

→

```
        {
            if (ActivityEmployeeLogin.dbc.insertData(name,
                        salary, mail, pass))
            {
                Toast.makeText(this, "Employee Registered", 3000).show();
                startActivity(new Intent(this, ActivityEmployeeLogin.
                finish();                                    class));
            }
            else
            {
                Toast.makeText(this, "Employee Not Registered", 3000).show();
            }
        }
    }
}
```

— EmployeeWelcomeActivity.java.

```
public class EmployeeWelcomeActivity extends Activity
{
    Button b1;
    TextView tv;
    String email;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        Super.onCreate(savedInstanceState);
        setContentView(R.layout.employee_layout_employee_
                                                Welcome);

        Bundle ref = getIntent().getExtras();
        email = ref.getString("mail");

        tv = (TextView) findViewById(R.id.Welcome);
        tv.setText("Mr./Miss. :" + email);
```

```
        b1 = (Button) findViewById (R.id.change);
        b1.setOnClickListener (new onClickListener ()
        {
            @Override
            public void onClick (View v)
            {
                Intent i = new Intent (getApplicationContext (),
                        EmployeeChangePasswordActivity.class);
                i.putExtra ("mail", email);
                startActivity (i);
            }
        });
    }
}
```

— EmployeeChangePasswordActivity.java.

```
public class EmployeeChangePasswordActivity extends Activity
{
    String email;

    @Override
    protected void onCreate (Bundle savedInstanceState)
    {
        super.onCreate (savedInstanceState);
        setContentView (R.layout.layout_employee_change_password);

        Bundle ref = getIntent().~~toString()~~; getExtras ();
        email = ref.getString ("mail");
    }

    public void change (View v)
    {
        showAlert ();
    }
```

→

```
public void showAlert ()
{
    AlertDialog.Builder ad = new AlertDialog.Builder (this);
    ad.setTitle ("Password");
    ad.setIcon (R.drawable.ic_action_settings);
    ad.setMessage ("Do update Password... Press Yes
                    else Cancel");

    ad.setPositiveButton ("Yes", new onClickListener ()
    {
        @Override
        public void onClick (~~View v~~) DialogInterface dialog,
                                        int which)
        {
            EditText et1, et2;
            et1 = (EditText) findViewById (R.id.oldPassword);
            et2 = (EditText) findViewById (R.id.newPassword);

            String opass = et1.getText ().toString ();
            String npass = et2.getText ().toString ();

            if (ActivityEmployeeLogin.dbc.changePassword
                                    (email, opass, npass))
            {
                startActivity (new Intent (getApplicationContext (),
                                    ActivityEmployeeLogin.class));
                finish ();
            }
        }
    });

    ad.setNeturalButton ("cancel", new onClickListener ()
    {
        @Override
        public void onClick (DialogInterface dialog, int which)
        { dialog.cancel (); }
    }); ad.show ();
} }
```

— DataBaseCreation . java.

```
import android.content.Context;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public Class DataBaseCreation extends SQLiteOpenHelper
{
    Static String DB_NAME = "SathyaDB";
    Static int DB_VERSION = 1;
    Context context;

    public DataBaseCreation (context ref)
    {
        super (ref, DB_NAME, null, DB_VERSION);
        context = ref;
    }

    @Override
    public void onCreate (SQLiteDatabase db)
    {
        String qry = " create table Employee_Data (NAME text,
                     SALARY real, EMAIL text primary key,
                     pa PASSWORD text-)";
        try {
            db.execSQL (qry);
        }
        catch (Exception ex)
        {
            Toast.makeText (context, "" + ex.getMessage (), 3000).show();
        }
    }
```

→

```
public boolean insertData (String name, String email,
                           String sal, String pass)
{
    float salary = Float.parseFloat (sal);
    try {
        String qry = "insert into Employee_Data Values
                    ('"+name+"','"+salary+"',
                    '"+email+"','"+pass"')";

        SQLiteDatabase db = getWritableDatabase ();
        db.execSQL (qry);
        db.close ();
        return true;
    }
    catch (Exception ex)
    {
        Toast.makeText (context, ""+ex.getMessage (),
                                    3000).show ();
        return false;
    }
}

public boolean loginCheck (String email, String pass)
{
    try {
        String qry = "select password from Employee_Data
                    where EMAIL = '"+email+"'";

        SQLiteDatabase db = getWritableDatabase ();
        Cursor c = db.rawQuery (qry, null);
        c.moveToFirst ();
        String password = c.getString (0);

        if (password.equals (pass))
        {
```

```
        Toast.makeText(context, "Valid Employee", 3000).show();
        return true;
    }
    else
    {
        Toast.makeText(context, "InValid Employee", 3000).show();
        return false;
    }
} → catch (Exception ex) {
        Toast.makeText(context, "InValid Employee", 3000).show();
public }
public boolean changePassword(String email, String opass,
                              String npass)
{
    email = email.trim();
    opass = opass.trim();
    npass = npass.trim();

    String qry = "Select password from Employee_Data where
                  EMAIL = '" + email + "'";
    SQLiteDatabase db = getWriteableDatabase();
    Cursor c = db.rawQuery(qry, null);
    c.moveToFirst();
    String password = c.getString(0);

    if (password.equals(opass))
    {
        try {
            String qry1 = "update Employee_Data set
              password = '" + npass + "' where
              EMAIL = '" + email + "'";
            db.execSQL(qry1);
            Toast.makeText(context, "Password Update",
                           3000).show();
            return true;
        }
```

→

```
        catch (Exception ex)
        {
            Toast.makeText(context, "Sorry Not Updated", 3000).show();
            return false;
        }
    }
    else
    {
        Toast.makeText(context, "old Password Not Matching",
        return false;                                3000), show();
    }
}

@Override
protected void onUpgrade (SQLiteDatabase db,
                          int oldVersion, int newVersion)
{ }
}
```

— Android Manifeast.xml

```
<application
    <activity android:name = "com.nmk.cwprojectDB.ActivityEmployee
                                                    Login">
        <intent_filter>
            <action android:name = "android.intent.action.MAIN" />
            <category android:name = "android.intent.category.LAUNCHER" />
        </intent_filter>

    </activity>

    <Activity android:name = "com.nmk.cwprojectDB.EmployeeRegisterActivity" />
    <activity android:name = "com.nmk.cwprojectDB.EmployeeWelcomeActivity" />
    <activity android:name = "com.nmk.cwprojectDB.
                                    EmployeeChangePasswordActivity" />
</application>
```