ARUNAI ENGINEERING COLLEGE

Velu Nagar, Tiruvannamalai-606603.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Lab Manual

CS8662- Mobile Application Development Laboratory (VI semester)

Regulation 2017

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

1. Graduates will have successful career in Computer Science and related industries or pursue higher education and research or evolve as entrepreneurs.
2. Graduates will have the ability and attitude to adapt to emerging technological changes.
3. Graduates will excel as socially committed engineers with high ethical values, leadership qualities and empathy for the needs of society

PROGRAMME OUTCOMES (POs)

After going through the four years of study, Computer Science and Engineering Graduates will exhibit ability to:

PO#	Graduate Attribute	Programme Outcome
1	Engineering knowledge	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization for the solution of complex engineering problems.
2	Problem analysis	Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3	Design/development of solutions	Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and cultural, societal, and environmental considerations.
4	Conduct investigations of complex problems	Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5	Modern tool usage	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modeling to complex engineering activities, with an understanding of the limitations.
6	The engineer and society	Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7	Environment and sustainability	Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8	Ethics	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice
9	Individual and team work	Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings
10	Communication	Communicate effectively on complex engineering activities with the engineering community and with the society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions
11	Project management and finance	Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments
12	Life-long learning	Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

PROGRAM SPECIFIC OUTCOMES (PSOs)

By the completion of Computer Science and Engineering program the student will have following Program specific outcomes

1.To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

2.To apply software engineering principles and practices for developing quality software for scientific and business applications.

3. To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems.

CS8662 MOBILE APPLICATION DEVELOPMENT LABORATORY L T P C
0 0 4 2

OBJECTIVES

- To understand the components and structure of mobile application development frameworks
- for Android and windows OS based mobiles.
- To understand how to work with various mobile application development frameworks.
- To learn the basic and important design concepts and issues of development of mobile applications.
- To understand the capabilities and limitations of mobile devices.

LIST OF EXPERIMENTS

1. Develop an application that uses GUI components, Font and Colours.
2. Develop an application that uses Layout Managers and event listeners.
3. Write an application that draws basic graphical primitives on the screen.
4. Develop an application that makes use of databases.
5. Develop an application that makes use of Notification Manager.
6. Implement an application that uses Multi-threading
7. Develop a native application that uses GPS location information
8. Implement an application that writes data to the SD card.
9. Implement an application that creates an alert upon receiving a message
10. Write a mobile application that makes use of RSS feed
11. Develop a mobile application to send an email.
12. Develop a Mobile application for simple needs (Mini Project)

TOTAL: 60 PERIODS

LIST OF EQUIPMENT FOR A BATCH OF 30 STUDENTS

HARDWARE

Standalone desktops 30 Nos

SOFTWARE:

- C / C++ / Java or equivalent compiler GnuPG, Snort, N-Stalker or **Equivalent**

30 Nos.

JAVA

Java is a high-level programming language originally developed by Sun Microsystems. Java runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX. Java programming were "Simple, Robust, Portable, Platform-independent, Secured, High Performance, Multithreaded, Architecture Neutral, Object-Oriented, Interpreted, and Dynamic".

COURSE OUTCOMES

CS8662.1	Develop mobile applications using GUI and Layouts.
CS8662.2	Develop mobile applications using Event Listener.
CS8662.3	Develop mobile applications using Databases.
CS8662.4	Develop mobile applications using RSS Feed, Internal/External Storage, SMS, Multithreading and GPS.
CS8662.5	Analyze and discover own mobile app for simple needs.

MAPPING OF COURSE OUTCOMES WITH PROGRAMME OUTCOMES:

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CS8662.1	2	3	3	2	3	2	1	3	3	1	2	2
CS8662.2	2	3	3	2	3	2	-	3	-	-	2	1
CS8662.3	2	3	3	2	3	-	1	3	-	-	2	2
CS8662.4	2	3	3	1	3	2	3	3	-	-	2	2
CS8662.5	2	2	3	1	3	-	-	3	-	-	2	2
CS8662	2	3	3	2	3	2	1	3	3	-	2	2

MAPPING OF COURSE OUTCOMES WITH THE PROGRAM SPECIFIC OUTCOMES:

CO/PSO	PSO1	PSO2	PSO3
CS8662.1	3	3	3
CS8662.2	3	3	1
CS8662.3	3	3	2
CS8662.4	-	-	2
CS8662.5	3	2	2
CS8662	3	2	2

EVALUATION PROCEDURE FOR EACH EXPERIMENTS

S.No	Description	Mark
1.	Aim & Pre-Lab discussion	20
2.	Observation	20
3.	Conduction and Execution	30
4.	Output & Result	10
5.	Viva	20
Total		**100**

INTERNAL ASSESSMENT FOR LABORATORY

S.No	Description	Mark
1.	Observation	05
2.	Performance	05
2.	Viva	05
3.	Record	05
Total		**20**

Need to know before start coding

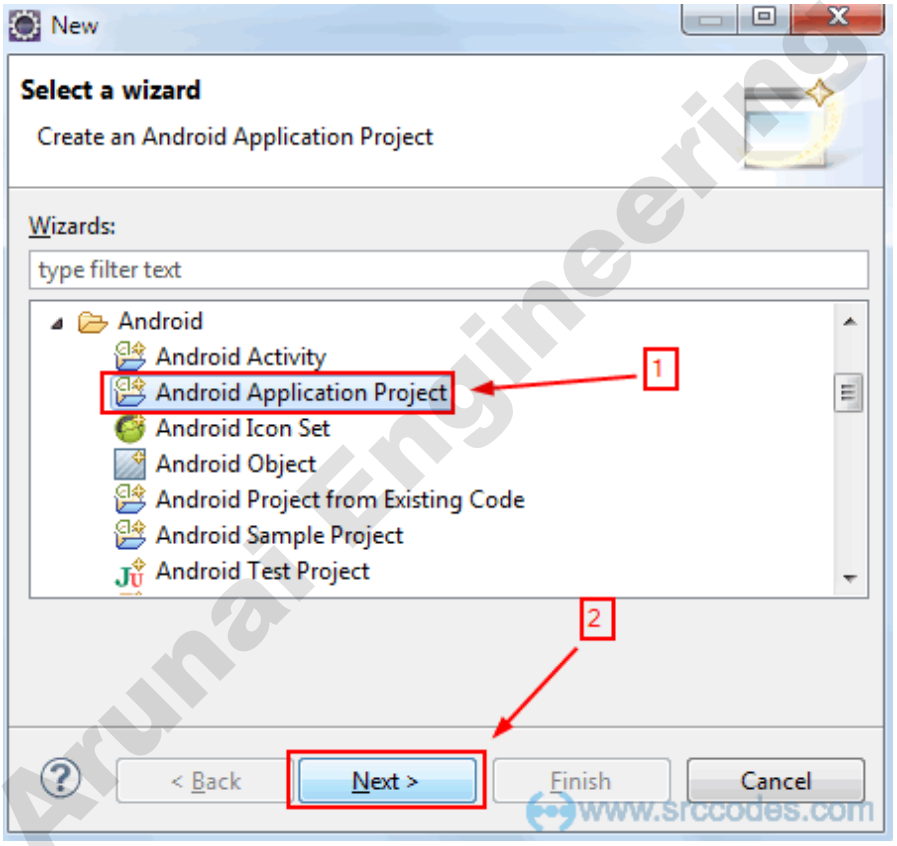
This tutorial will help you to write your first Android 'Hello World!' program. Here, we'll use Eclipse IDE with Android Developer Tools (ADT) plugin to build the application and Android Emulator - Android Virtual Device (AVD) to run the application which will draw 'Hello World!' text on the screen.

Tools & Technologies used :

1. JDK 1.6
2. Eclipse 3.7
3. Android SDK

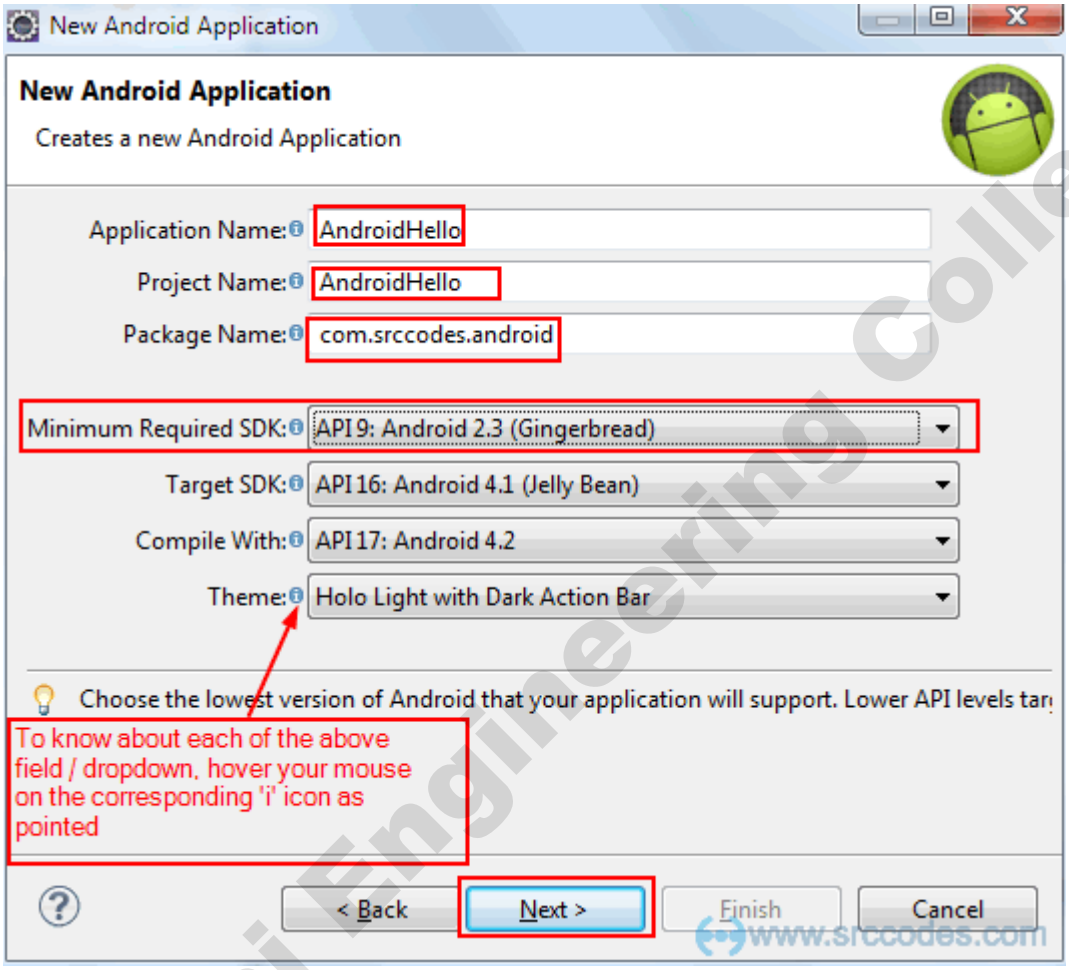
1. Create Android Project

Select from the menu File --> New --> Other --> Android --> Android Application Project (say 'AndroidHello') and click **Next** button.



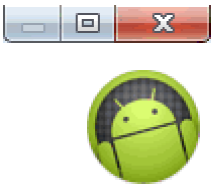
2. Configure Project Settings

Enter Application, Project and Package Name. Select '**Minimum Required SDK**' (lowest version of Android that this app supports), '**Target SDK**' (highest version of Android with which this application has been tested), '**Compile With**' (platform version against which this application will be compiled with) and '**Theme**' (Android UI style) from the corresponding theme. To make it simple you can leave the dropdown value as it is. Click **Next** button



Click **Next** button

New Android Application



C onfigure Project

Create custom launcher

icon Create activity

@ Mark this project asa library

Create project in Workspace

onBe

Location: C:\ projects\self dev\androidws\AndroidHello

Working sets

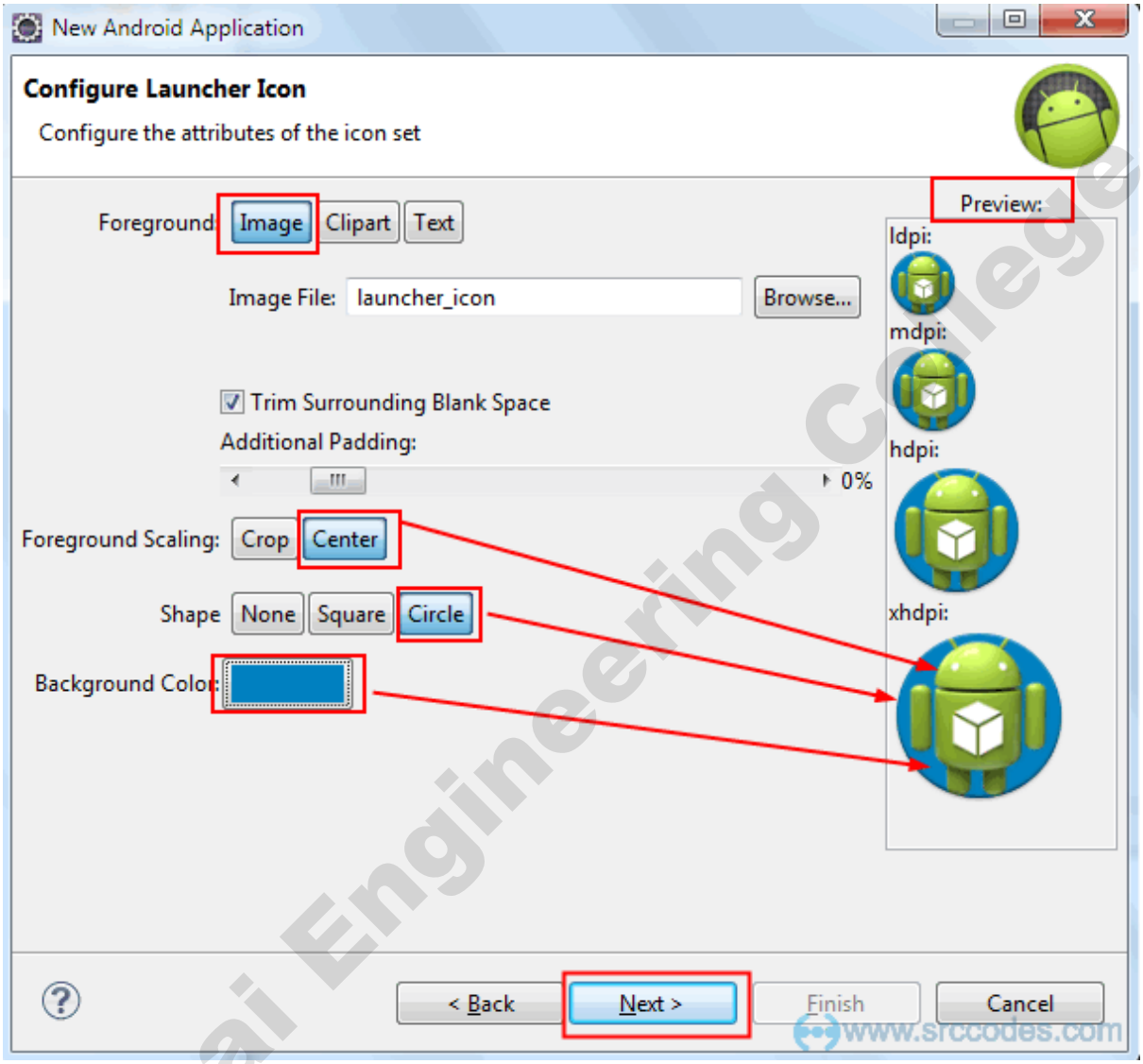
. eat Se

@ *And* project to orking

sets Wo rk ing sets:

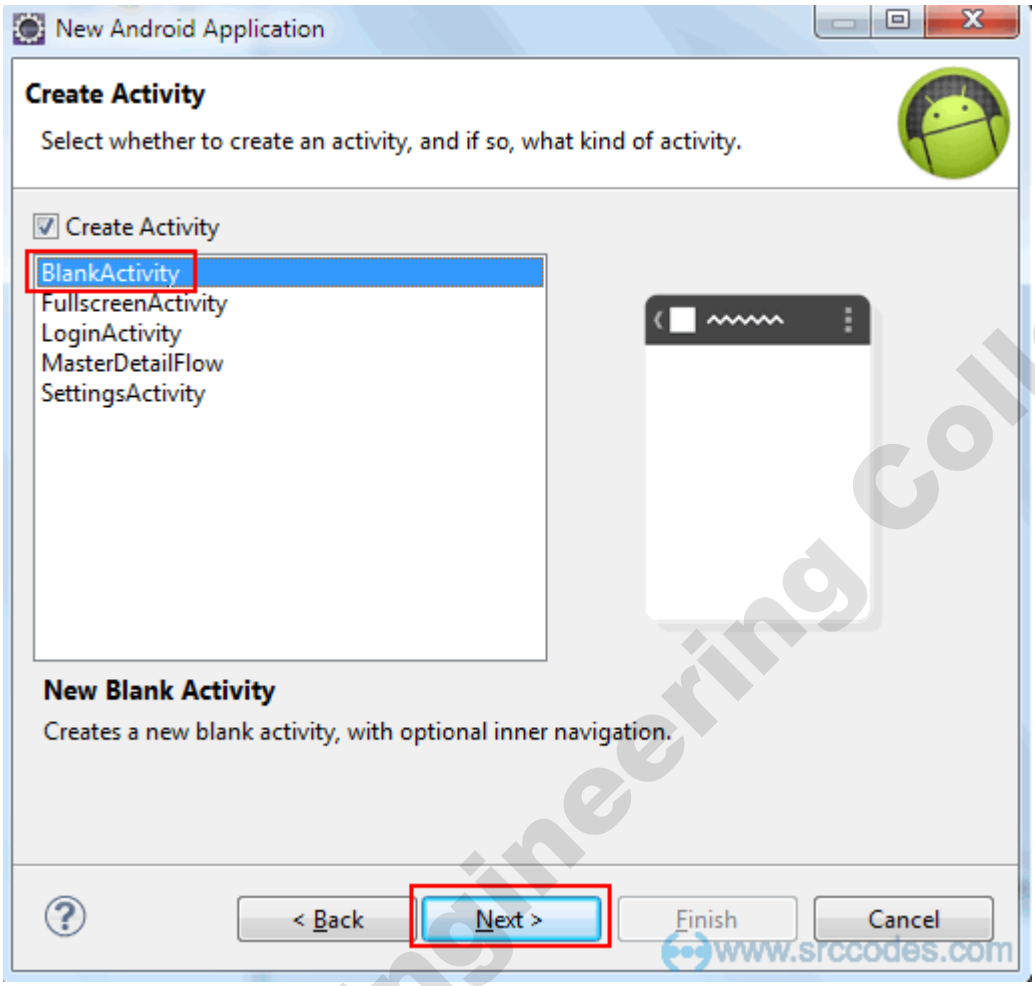
3. Configure App Launcher Icon

Choose your App icon and configure as per your requirement. For demonstration purpose, I have changed few settings as shown below

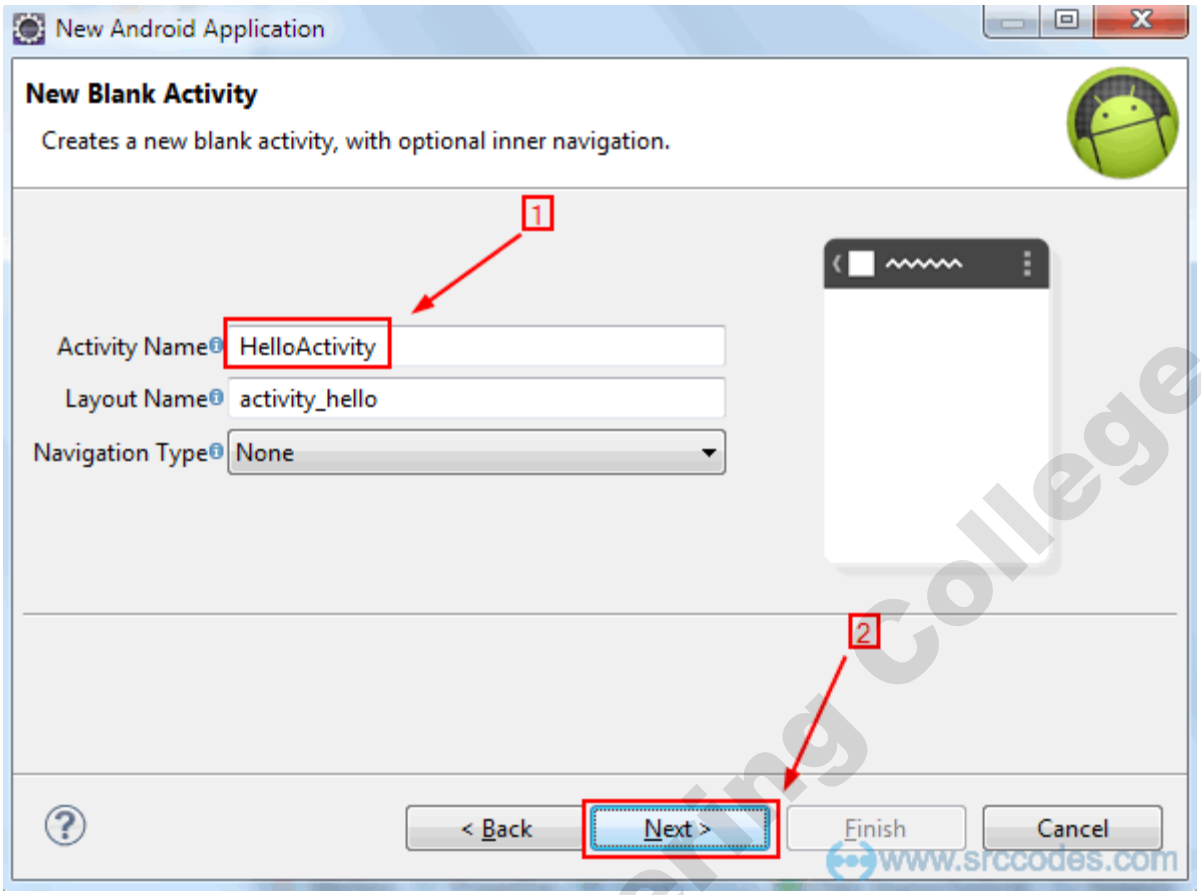


4. Create Activity

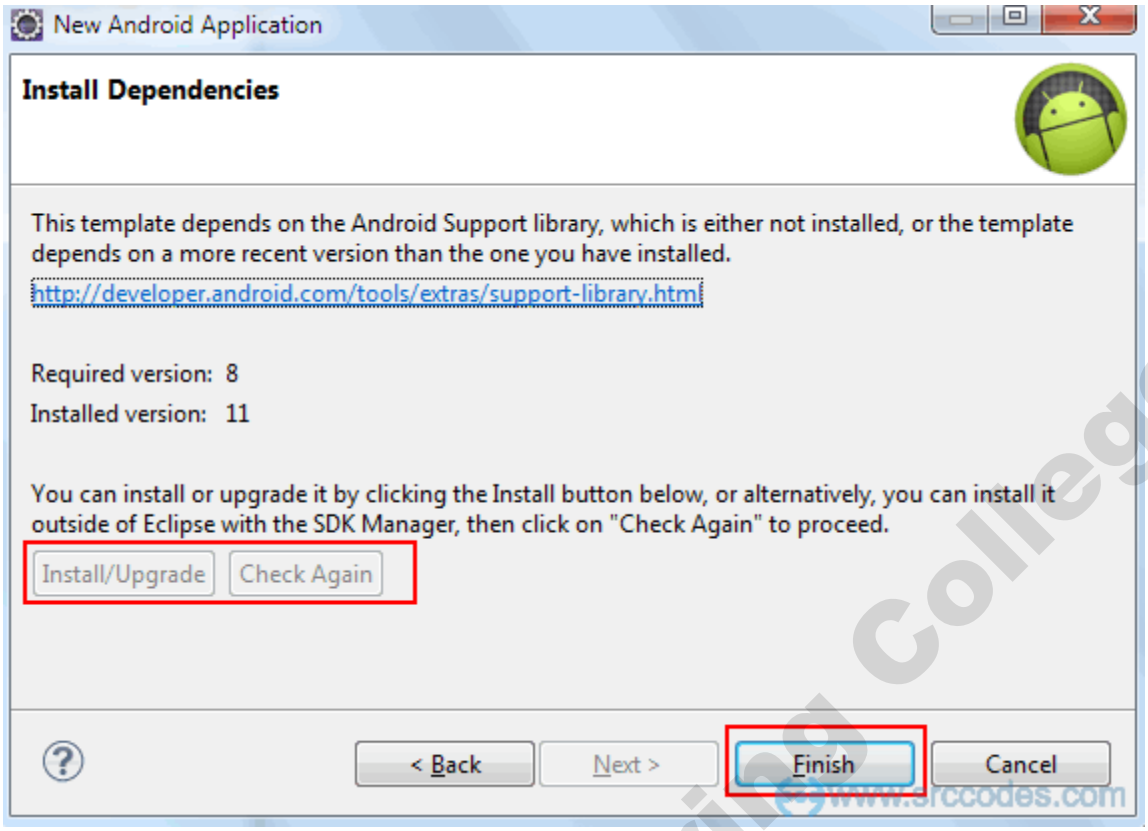
Choose an activity template (say '**BlankActivity**') and click **Next** button.



Enter '**Activity Name**' (say 'HelloActivity') and click **Finish** button.



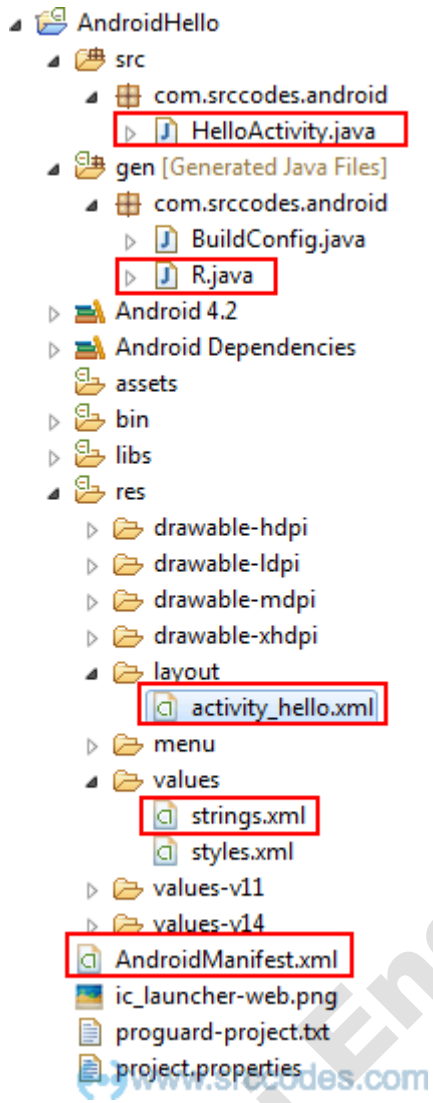
If **Finish** button is not enabled and **Next** is enabled that means required dependencies (Supporting library) are not installed. In this case click **Next** button and hit '**Install/Upgrade**' button to install or upgrade required dependencies.



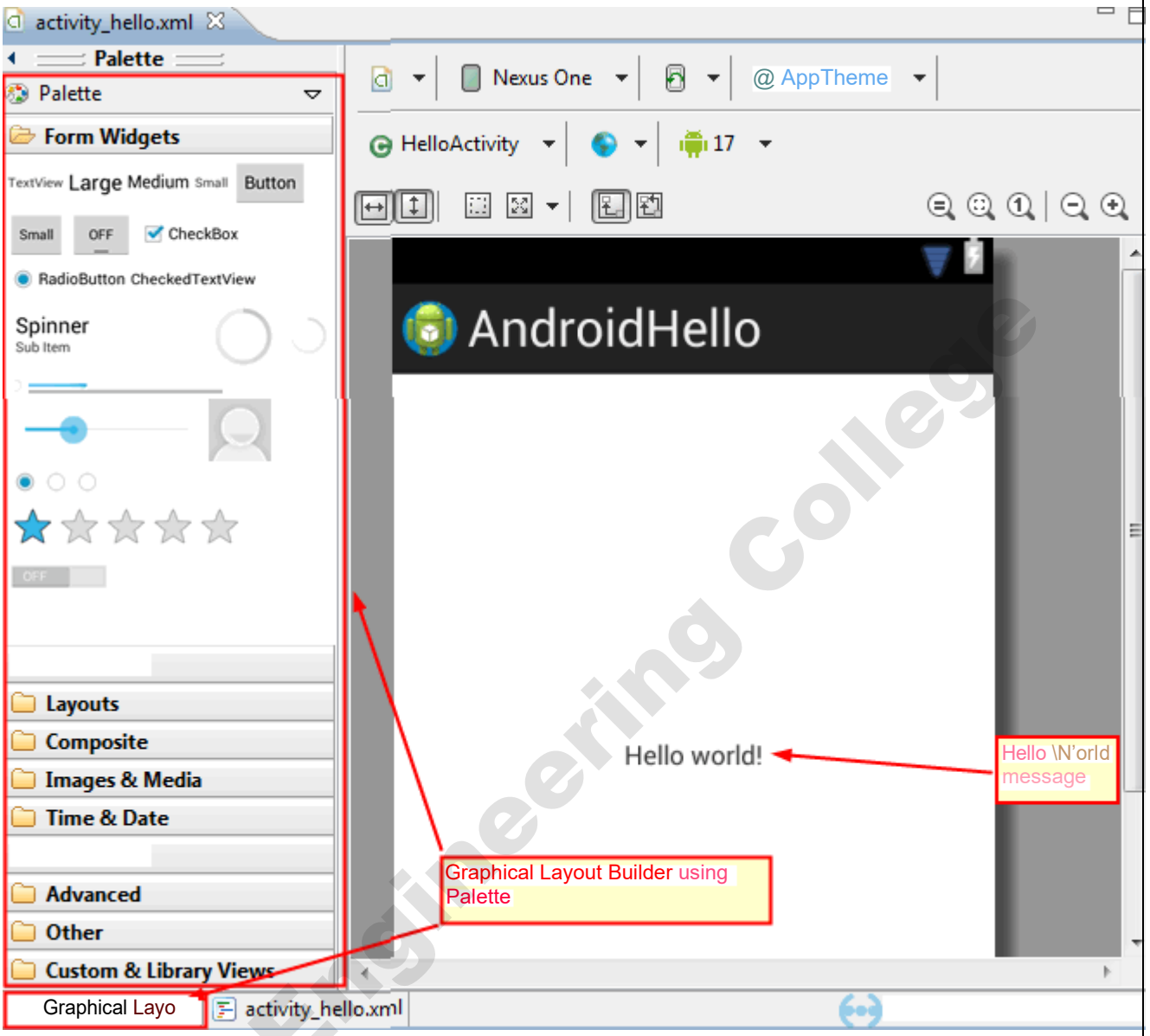
Finally click **Finish** button.

5. Overall Project Structure

Android project will be created with some default files as shown below



'android_hello.xml' (layout) will be opened using '**Android Common XML Editor**'. Here we can build UI by simply dragging and dropping UI components from the Palette.



6. Code

'hello_world' resource string contains the message 'Hello world!' which will be shown on launching of the application.

File : strings.xml

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

        <string name="app_name">AndroidHello</string>
        <string name="hello_world">Hello world!</string>
        <string name="menu_settings">Settings</string>

    </resources>
```

'activity_hello.xml' is the layout built using '**Android Common XML Editor**'. Instead of using a hard-coded string value ('Hello world!') in '<TextView>' element, the "@string/hello_world" value refers to a string resource defined in strings.xml.

File : activity_hello.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".HelloActivity" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true
        " android:layout_centerVertical="true"
        android:text="@string/hello_world" />

</RelativeLayout>
```

For this application we do not require to change anything in the generated activity code.

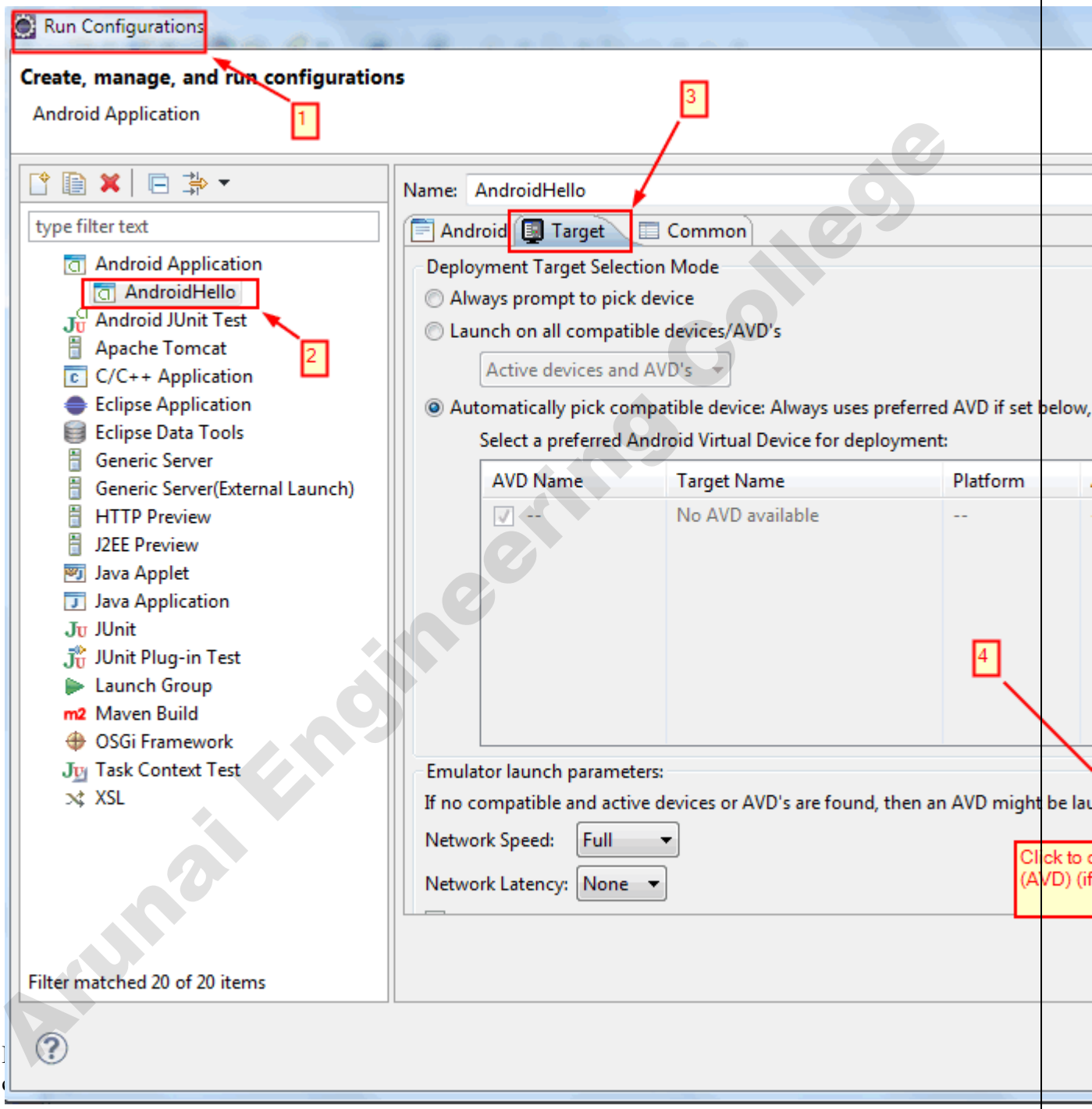
File : HelloActivity.java

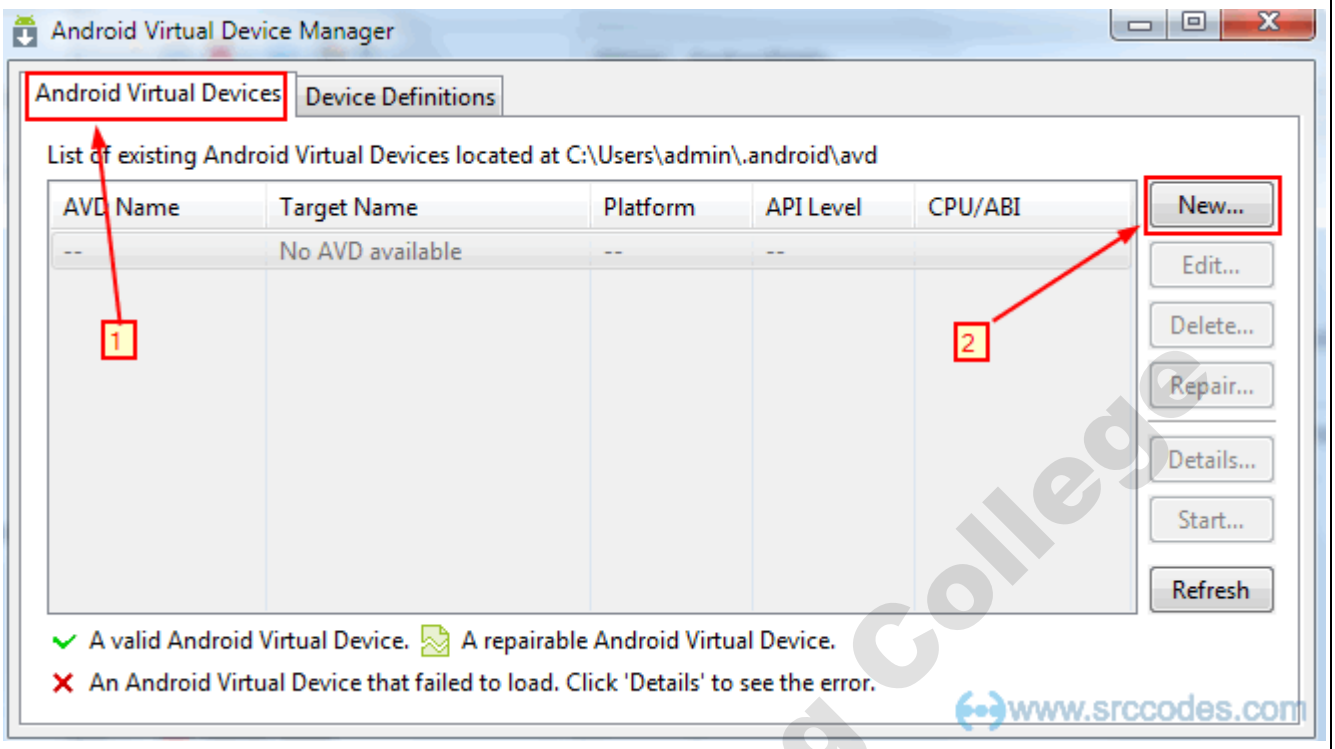
```
1          package com.srccodes.android;
2
3          import
4   android.os.Bundle;
5          import
6   android.app.Activity;
7          import
8   android.view.Menu;
9
10  public class HelloActivity extends Activity {
11
                    @Override
              protected void onCreate(Bundle
                    savedInstanceState) {
              super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_hello);
                              }
```

```
12
13                          @Override
14      public boolean onCreateOptionsMenu(Menu menu) {
15      // Inflate the menu; this adds items to the action bar if it
16                              is present.
17      getMenuInflater().inflate(R.menu.activity_hello, menu);
18                              return true;
19                                  }
20                                  }
21
```

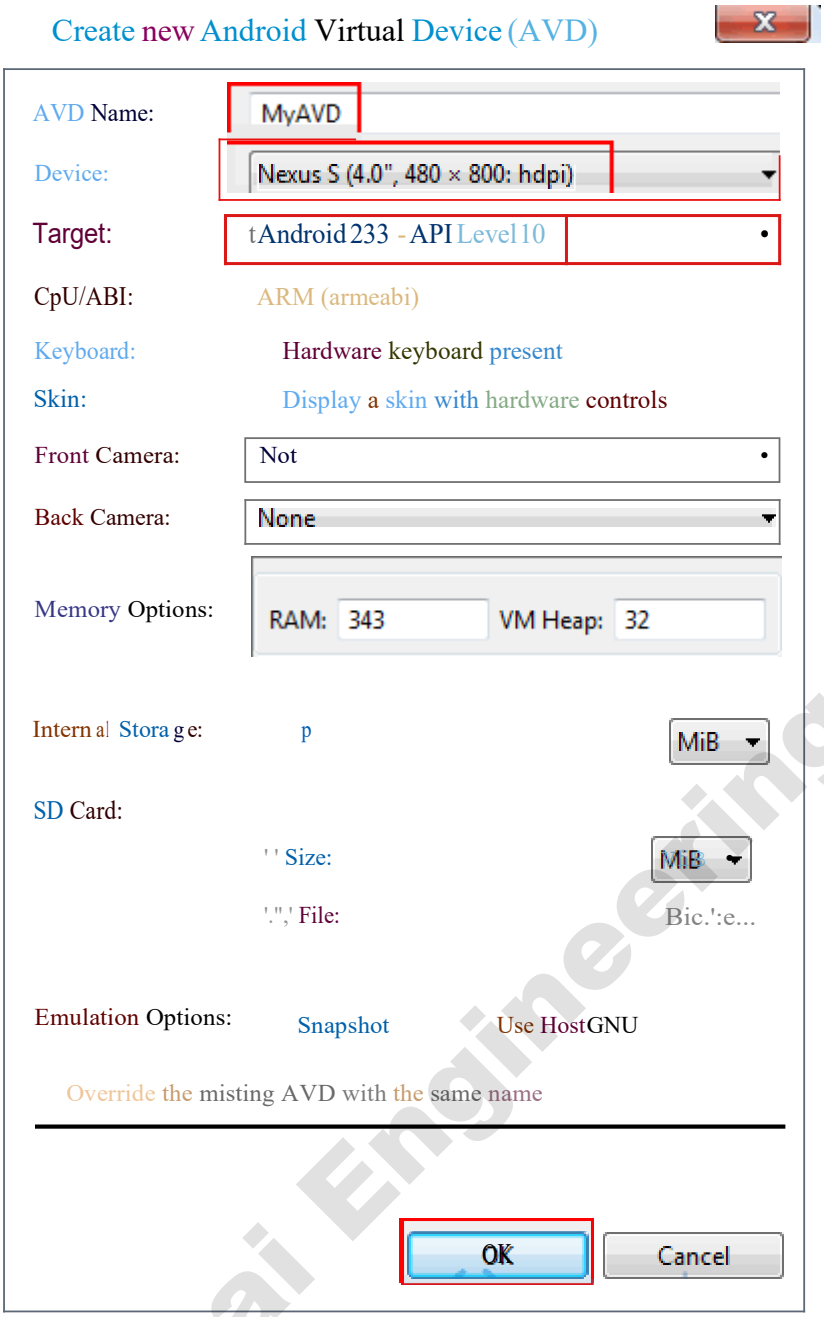
7. Run Configuration

Right click on the project and from the context menu select 'Run As' --> 'Run Configurations..'





Configure AVD as shown below and click **OK** button.



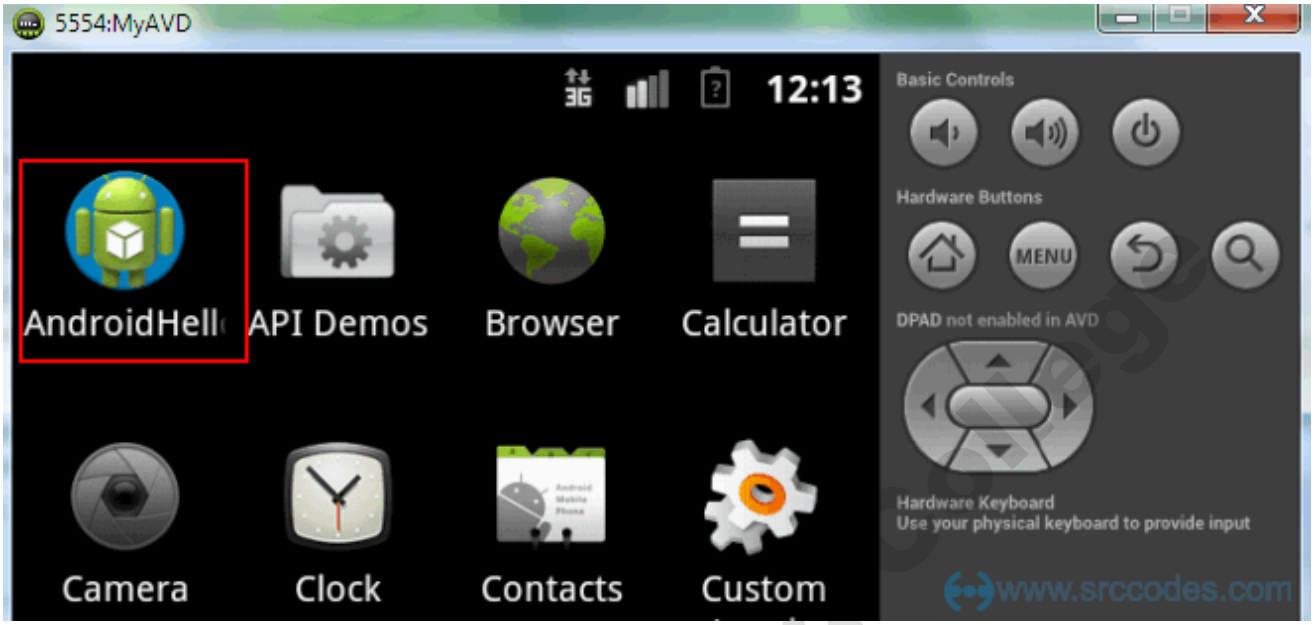
8. Run Application

Right click on the project and from the context menu select 'Run As' --> 'Android Application'.

9. Output

Eclipse ADT will start the AVD and launch your application with 'Hello world!' message on the screen.

Click home icon in the emulator and click the launcher icon to find your application. There you'll see the app icon which we have configured at step #3.



Ex. No.1 Develop an application that uses GUI components, Font and Colours

Introduction:

Android offers a sophisticated and powerful componentized model for building your UI, based on the fundamental layout classes: View and ViewGroup. To start with, the platform includes a variety of prebuilt View and ViewGroup subclasses called widgets and layouts, respectively that you can use to construct your UI.

A partial list of available widgets includes Button, TextView, EditText, ListView, CheckBox, RadioButton, Gallery, Spinner, and the more special- purpose AutoCompleteTextView, ImageSwitcher, and TextSwitcher.

Among the layouts available are LinearLayout, FrameLayout, RelativeLayout, and others. For more examples, see Common Layout Objects.

If none of the prebuilt widgets or layouts meets your needs, you can create your own View subclass. If you only need to make small adjustments to an existing widget or layout, you can simply subclass the widget or layout and override its methods.

Aim:

To develop an android application that uses GUI Components,Fonts and colours.

Procedure:

1. Create a new project in Eclipse by navigating to **File ⇒ New ⇒Android Project** and fill all the required details.

2. Open **activity_main.xml** by navigating to **res ⇒layout⇒activity_main.xml** and type the following code

```
<RelativeLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingBottom="@dimen/activity_vertical_margin"
  android:paddingLeft="@dimen/activity_horizontal_margin"
  android:paddingRight="@dimen/activity_horizontal_margin"
  android:paddingTop="@dimen/activity_vertical_margin"
  android:id="@+id/screen"
  tools:context=".MainActivity" >

  <EditText
    android:id="@+id/editText1
    "
    android:layout_width="wrap_co
    ntent"
    android:layout_height="wrap_co
    ntent"
    android:layout_alignParentTop=
    "true"
    android:layout_centerHorizontal
    ="true"
    android:layout_marginTop="42d
    p" android:ems="10"
    android:inputType="textPersonN
    ame" android:hint="Enter your
    name here">

    <requestFocus />
  </EditText>

  <TextView
    android:id="@+id/TextView1"
    android:layout_width="wrap_co
    ntent"
    android:layout_height="wrap_c
    ontent"
    android:layout_centerVertical="
    true"
    android:textAlignment="center"
```

```
    android:fontFamily="sans-serif-medium"

/>
 <Button
      android:id="@+id/button1"
      android:layout_width="wrap_cont
      ent"
      android:layout_height="wrap_con
      tent"
      android:layout_below="@+id/edi
      tText1"
      android:layout_centerHorizontal=
      "true"
      android:layout_marginTop="24dp
      " android:text="Click Here" />
```

```
<RadioButton
   android:id="@+id/
   red"
   android:layout_width="wrap_content
   "
   android:layout_height="wrap_conten
   t"
   android:layout_alignLeft="@+id/Tex
   tView1"
   android:layout_below="@+id/TextVi
   ew1"
   android:layout_marginTop="37dp"
   android:text="Red" />

<RadioButton
   android:id="@+id/b
   lue"
   android:layout_width="wrap_conte
   nt"
   android:layout_height="wrap_conte
   nt"
   android:layout_above="@+id/check
   Box2"
   android:layout_toRightOf="@+id/b
   utton1" android:text="Blue" />

<CheckBox
   android:id="@+id/checkBox2"
   android:layout_width="wrap_co
   ntent"
   android:layout_height="wrap_c
   ontent"
   android:layout_alignLeft="@+id/edi
   tText1"
   android:layout_below="@+id/red"
   android:layout_marginTop="36dp"
   android:text="Background" />

<CheckBox
   android:id="@+id/checkBox1"
   android:layout_width="wrap_co
   ntent"
   android:layout_height="wrap_c
   ontent"
   android:layout_alignBaseline="@+id/che
   ckBox2"
   android:layout_alignBottom="@+id/chec
   kBox2"
   android:layout_alignRight="@+id/editTe
   xt1" android:layout_marginRight="19dp"
   android:text="Font" />

<RadioButton
```

```
        android:id="@+id/gr
        een"
        android:layout_width="wrap_content
        "
        android:layout_height="wrap_conten
        t"
        android:layout_alignBaseline="@+id
        /blue"
        android:layout_alignBottom="@+id/
        blue"
        android:layout_toLeftOf="@+id/che
        ckBox1" android:text="Green" />

    </RelativeLayout>
```

3. This will design app with GUI components like Textbox,Buttons,Checkboxes, Radio buttons.

4. Open MainActivity.java by navigating through src->MainActivity.java and type the following code.

```
package com.example.guiapp;
```

```
import android.os.Bundle; import
android.app.Activity; import
android.graphics.Color; import
android.view.Menu; import
android.view.View; import
android.widget.Button;
import android.widget.CheckBox; import
android.widget.EditText; import
android.widget.RadioButton; import
android.widget.TextView; import
android.graphics.Typeface;

public class MainActivity extends Activity { Button

mbutton;
EditText medit;
TextView mtext;
RadioButton red;
RadioButton green;
RadioButton blue;
CheckBox bck;
CheckBox font;
public void setActivityBackgroundColor(int color) { View view
= this.getWindow().getDecorView();
view.setBackgroundColor(color);
}
@Override
protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main); mbutton=(Button)findViewById(R.id.button1);
medit=(EditText)findViewById(R.id.editText1);
mtext=(TextView)findViewById(R.id.TextView1); mbutton.setOnClickListener(new
View.OnClickListener() {

@Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
mtext.setText("Hi"+"              "+medit.getText());
}
});
red=(RadioButton)findViewById(R.id.red); red.setOnClickListener(new
View.OnClickListener() {
```

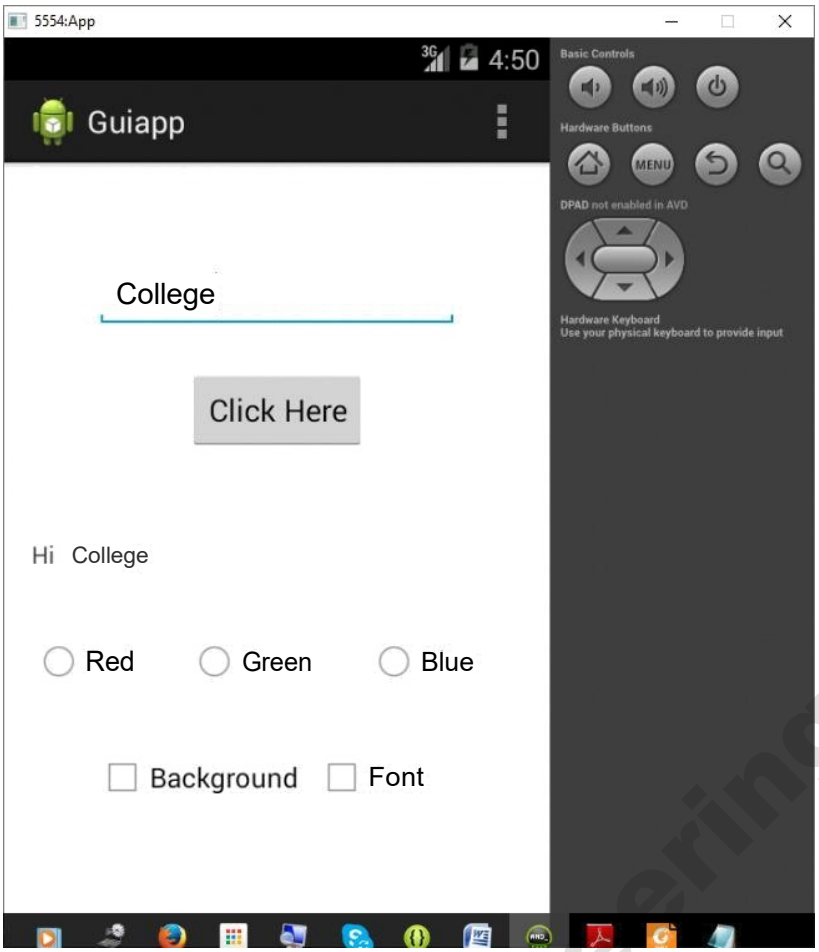
```
@Override
public void onClick(View v) {
// TODO Auto-generated method stub
mtext.setTextColor(Color.parseColor("#ff0000"));
}
});
green=(RadioButton)findViewById(R.id.green); green.setOnClickListener(new
View.OnClickListener() {

@Override
public void onClick(View v) {
// TODO Auto-generated method stub
mtext.setTextColor(Color.parseColor("#008000"));
}
});
blue=(RadioButton)findViewById(R.id.blue); blue.setOnClickListener(new
View.OnClickListener() { @Override
public void onClick(View v) {
// TODO Auto-generated method stub
mtext.setTextColor(Color.parseColor("#0000ff"));
}
});

bck=(CheckBox)findViewById(R.id.checkBox2); bck.setOnClickListener(new
View.OnClickListener() {

@Override
public void onClick(View v) {
// TODO Auto-generated method stub

setActivityBackgroundColor(0xffffff00);
// Set the color
}
});
font=(CheckBox)findViewById(R.id.checkBox1);
font.setOnClickListener(new View.OnClickListener() {

@Override
public void onClick(View v) {
// TODO Auto-generated method stub
```

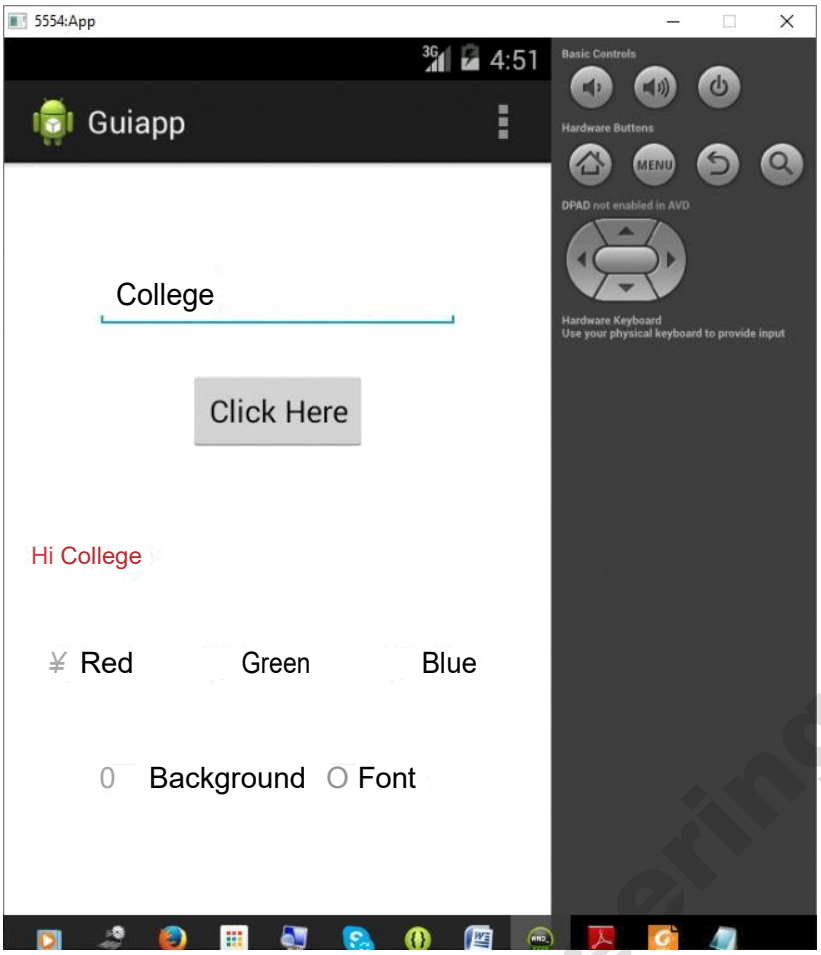
```
Typeface face = Typeface.createFromAsset(getAssets(), "fonts/BRUSHSCRIPTSTD.OTF");
mtext.setTypeface(face);

}
});
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present. getMenuInflater().inflate(R.menu.main,
menu);
return true;
}
}
```

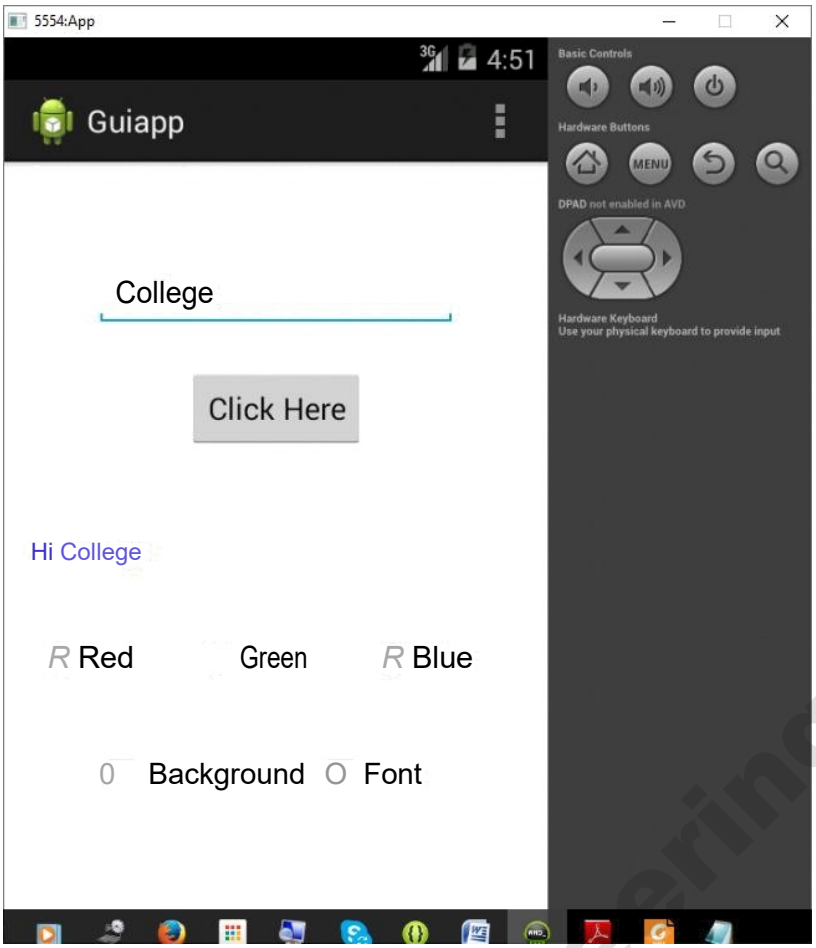
5. Run the app by rightclick on project package ,choose RunAs andchoose Android Applcation.

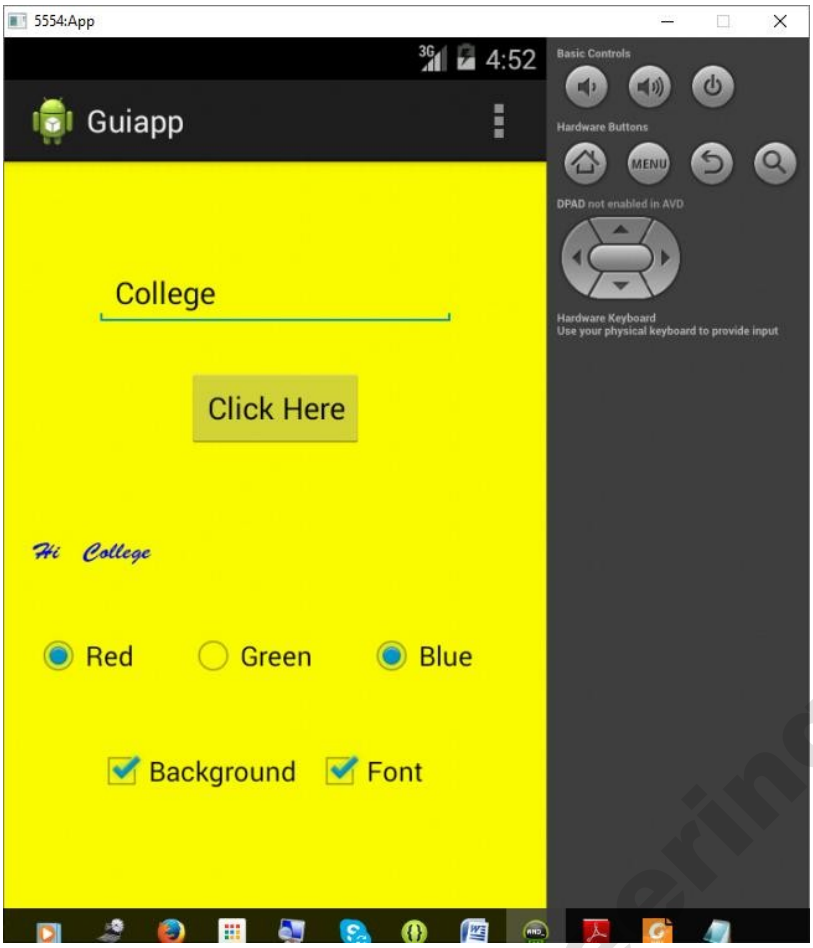
6. The output will be run in android emulator.

Output:









Result:

Thus an android application with GUI components has been developed successfully.

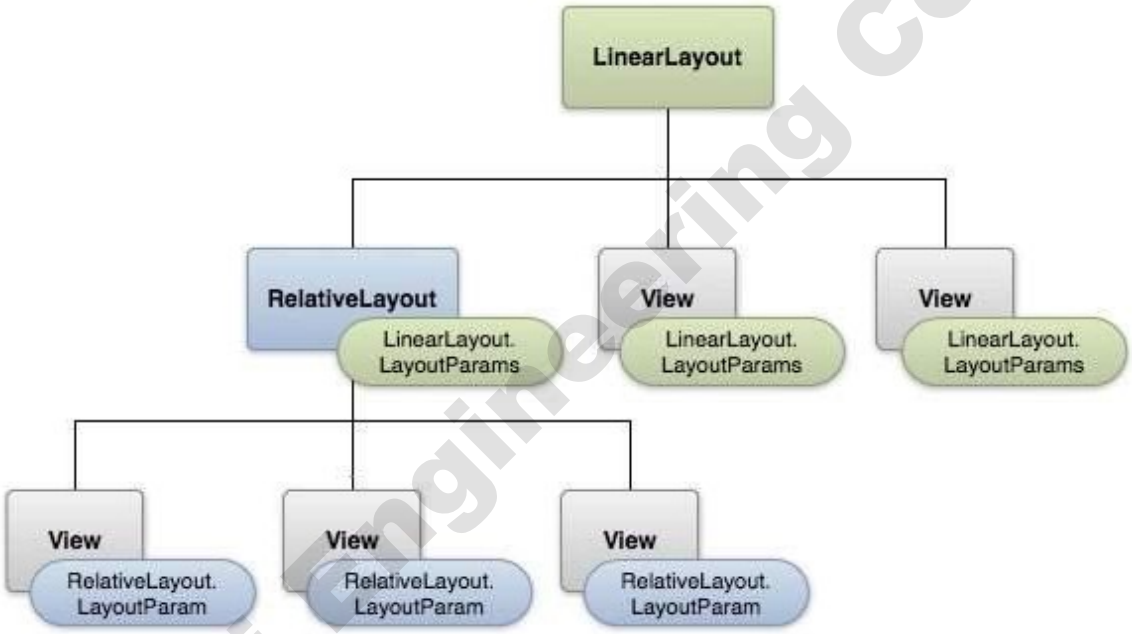
Ex.No.2 Develop an application that uses Layout Managers and event listeners

Introduction:

The basic building block for user interface is a **View** object which is created from the View class and occupies a rectangular area on the screen and is responsible for drawing and event handling. View is the base class for widgets, which are used to create interactive UI components like buttons, text fields, etc.

The **ViewGroup** is a subclass of **View** and provides invisible container that hold other Views or other ViewGroups and define their layout properties.

At third level we have different layouts which are subclasses of ViewGroup class and a typical layout defines the visual structure for an Android user interface and can be created either at run time using **View/ViewGroup** objects or you can declare your layout using simple XML file **main_layout.xml** which is located in the res/layout folder of your project.



Aim:

To develop an applications with different kinds of layouts and event listeners.

Procedure:

1. Create a new project in Eclipse by navigating to **File ⇒ New ⇒Android Project** and fill all the required details.

2. Create a new layout file under **res ⇒layout,**save in the name of Linear.xml and type the following code

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Parent linear layout with vertical orientation -->
<LinearLayout
xmlns:android="http://schemas.android.com/apk/re
s/android" android:orientation="vertical"
android:layout_width="match_parent"
android:layout_height="match_parent">

<TextView android:layout_width="fill_parent"
android:layout_height="wrap_content" android:text="Email:"
android:padding="5dip"/>

<EditText android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:layout_marginBottom="10dip"/>

<Button android:layout_width="fill_parent"
android:layout_height="wrap_content" android:text="Login"/>
<Button android:layout_width="fill_parent"
android:layout_height="wrap_content" android:text="Back"/>
<!-- Child linear layout with horizontal orientation -->
<LinearLayout android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:orientation="horizontal"
android:background="#2a2a2a"
android:layout_marginTop="25dip">

<TextView android:layout_width="fill_parent"
android:layout_height="wrap_content" android:text="Home"
android:padding="15dip" android:layout_weight="1"
android:gravity="center" android:textColor="#ffff00"
/>

<TextView android:layout_width="fill_parent"
android:layout_height="wrap_content" android:text="About"
android:padding="15dip" android:layout_weight="1"
android:gravity="center" android:textColor="#ffff00"/>

</LinearLayout>

</LinearLayout>
```

3. Create another new layout file under **res ⇒layout,**save in the name of Relative.xml and type the following code

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="wrap_content">
```

```
<TextView android:id="@+id/label"
android:layout_width="fill_parent"
android:layout_height="wrap_content" android:text="Email"
/>

<EditText android:id="@+id/inputEmail" android:layout_width="fill_parent"
android:layout_height="wrap_content" android:layout_below="@id/label" />

<Button android:id="@+id/btnLogin"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_below="@id/inputEmail"
android:layout_alignParentLeft="true"
android:layout_marginRight="10px" android:text="Login" />

<Button android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_toRightOf="@id/btnLogin"
android:layout_alignTop="@id/btnLogin" android:text="Back" />

<Button android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentBottom="true" android:text="Register new
Account" android:layout_centerHorizontal="true"/>
</RelativeLayout>
```

4. Create another new layout file under **res ⇒layout,**save in the name of Table.xml and type the following code

```
<TableLayout
xmlns:android="http://schemas.android.com/apk/re
s/android" android:layout_width="match_parent"
android:layout_height="match_parent"
android:shrinkColumns="*" android:stretchColumns="*" android:background="#ffffff">
<!-- Row 1 with single column -->
<TableRow
android:layout_height="wrap_c
ontent"
android:layout_width="fill_pare
nt"
android:gravity="center_horizo
ntal">
<TextView
android:layout_width="match_parent"
android:layout_height="wrap_content" android:textSize="18dp"
android:text="Row 1" android:layout_span="3"
android:padding="18dip" android:background="#b0b0b0"
android:textColor="#000"/>
</TableRow>

<!-- Row 2 with 3 columns -->
```

```
<TableRow
android:id="@+id/tableRow1"
android:layout_height="wrap_co
ntent"
android:layout_width="match_p
arent">
<TextView
android:id="@+id/TextView04" android:text="Row 2
column 1" android:layout_weight="1"
android:background="#dcdcdc"
android:textColor="#000000"
android:padding="20dip" android:gravity="center"/>
<TextView
android:id="@+id/TextView04" android:text="Row 2
column 2" android:layout_weight="1"
android:background="#d3d3d3"
android:textColor="#000000"
android:padding="20dip" android:gravity="center"/>
<TextView
android:id="@+id/TextView04" android:text="Row 2 column 3"
```

```
android:layout_weight="1"
android:background="#cac9c9"
android:textColor="#000000"
android:padding="20dip" android:gravity="center"/>
</TableRow>

<!-- Row 3 with 2 columns -->
<TableRow
android:layout_height="wrap_c
ontent"
android:layout_width="fill_pare
nt"
android:gravity="center_horizo
ntal">
<TextView
android:id="@+id/TextView04" android:text="Row 3
column 1" android:layout_weight="1"
android:background="#b0b0b0"
android:textColor="#000000"
android:padding="20dip" android:gravity="center"/>

<TextView
android:id="@+id/TextView04" android:text="Row 3
column 2" android:layout_weight="1"
android:background="#a09f9f"
android:textColor="#000000"
android:padding="20dip" android:gravity="center"/>
</TableRow>
<Button android:layout_width="fill_parent"
android:layout_height="wrap_content" android:text="Back"/>
</TableLayout>
```

5. Open MainActivity.java by navigating through src->MainActivity.java and type the following code.

```
package

com.example.layoutevent;

import

com.example.layoutevent.R;

import android.os.Bundle;
import android.app.Activity;
import
android.view.Menu;
import
android.view.View;
import
android.widget.Button;

public class MainActivity extends

Activity { Button Linear;
```

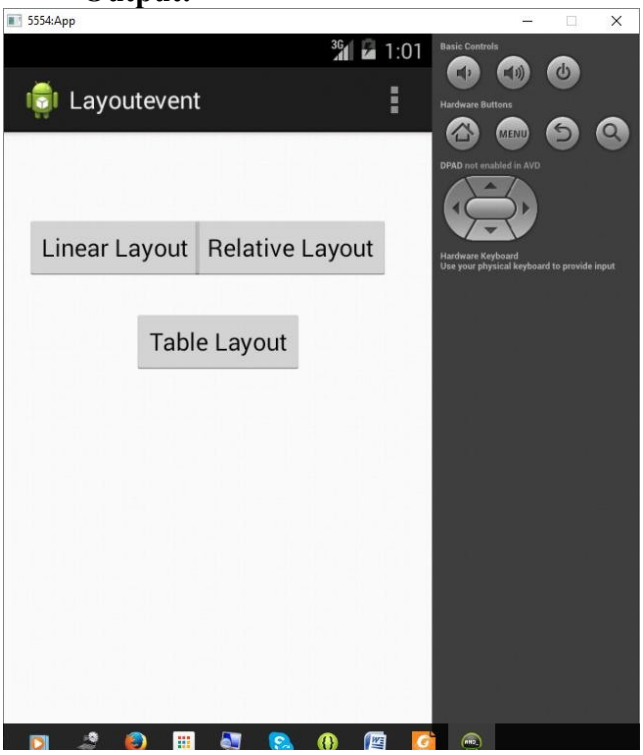
```
Button
Relative;
Button Table;
@Override
protected void onCreate(Bundle
savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
Linear=(Button)findViewById(R.id.button1);
Relative=(Button)findViewById(R.id.button2
);
Table=(Button)findViewById(R.id.button3);

Linear.setOnClickListener(new

View.OnClickListener() { @Override
public void onClick(View arg0) {
// TODO Auto-generated
method stub
setContentView(R.layout.linear
);
}
```

```
});

Relative.setOnClickListener(new

View.OnClickListener() { @Override
public void onClick(View arg0) {
// TODO Auto-generated
method stub
setContentView(R.layout.relati
ve);
}
});

Table.setOnClickListener(new

View.OnClickListener() { @Override
public void onClick(View arg0) {
// TODO Auto-generated
method stub
setContentView(R.layout.table)
;
}
});
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if
it is present. getMenuInflater().inflate(R.menu.main,
menu);
return true;
}
}
```

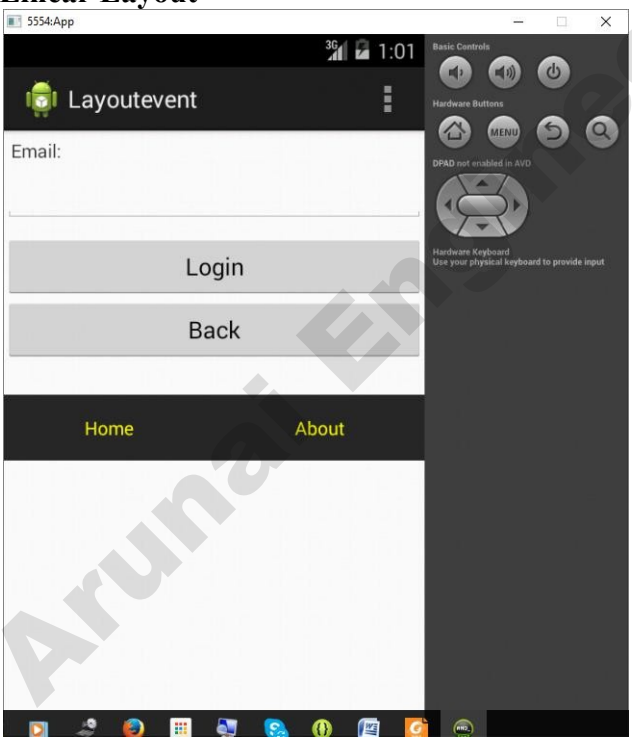
6. Run the app by rightclick on project package ,choose RunAs and choose Android Applcation.

7. The output will be run in android emulator.

Output:



Linear Layout



Relative Layout

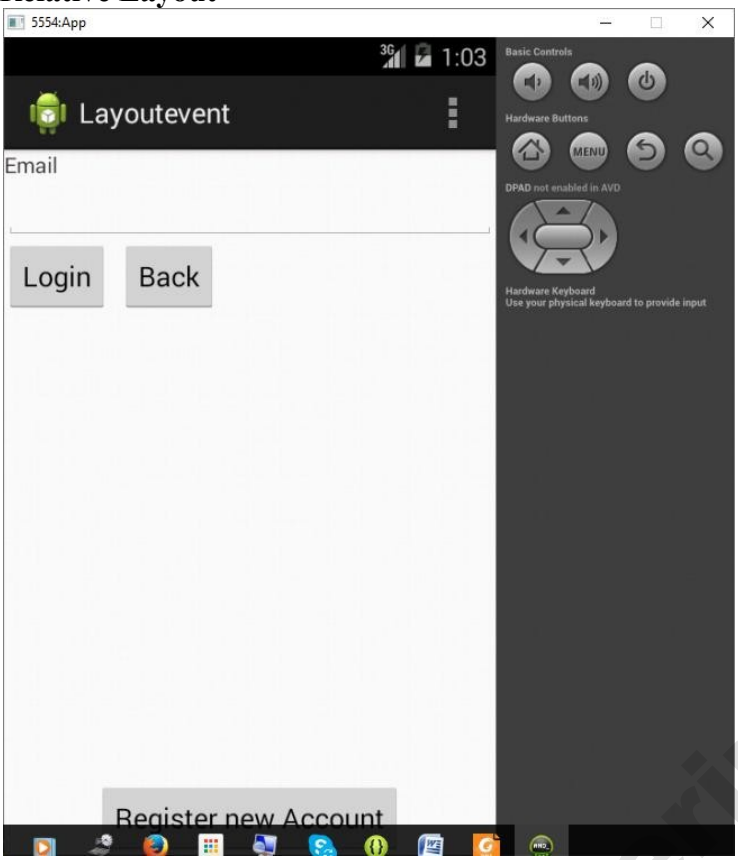
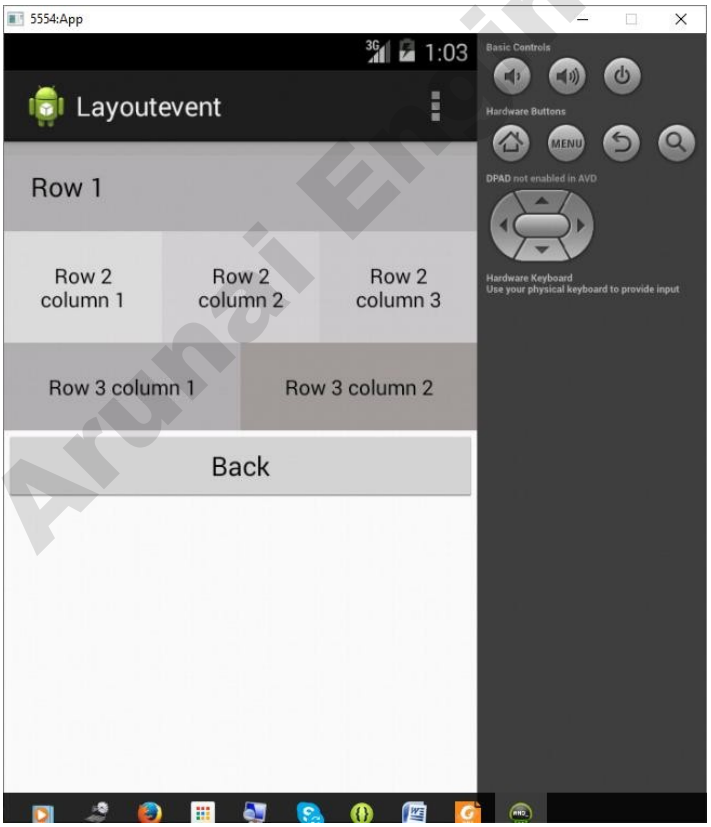


Table Layout:



Result:

Thus an android application with different layout has been developed successfully.

Ex.No.3 Develop an application that draws basic graphical primitives on the screen

Aim:

To develop anandroid application that draws basic graphical primitives on the screen.

Procedure:

1. Create a new project in Eclipse by navigating to **File ⇒ New ⇒Android Project** and fill all the required details.

2. Open **activity_main.xml** by navigating to **res ⇒layout⇒activity_main.xml** and type the following code

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical
_margin"
android:paddingLeft="@dimen/activity_horizontal
_margin"
android:paddingRight="@dimen/activity_horizonta
l_margin"
android:paddingTop="@dimen/activity_vertical_m
argin" tools:context=".MainActivity" >

<TextView
android:layout_width="wrap_co
ntent"
android:layout_height="wrap_c
ontent"
android:text="@string/hello_wo
rld" />

</RelativeLayout>
```

4. Open **MainActivity.java** by navigating through **src->MainActivity.java** and type the following code.

```
package com.example.drawingshapes; import
android.content.Context; import
android.graphics.Bitmap;
import android.graphics.BitmapShader; import
android.graphics.Canvas;
import android.graphics.ComposePathEffect; import
android.graphics.CornerPathEffect; import
android.graphics.DiscretePathEffect; import
```

android.graphics.LinearGradient; import
android.graphics.Paint;
import android.graphics.Path;

```
import android.graphics.PathEffect; import
android.graphics.RectF; import
android.graphics.Shader;
import android.graphics.SweepGradient; import
android.graphics.drawable.Drawable;
import android.graphics.drawable.ShapeDrawable; import
android.graphics.drawable.shapes.ArcShape; import
android.graphics.drawable.shapes.OvalShape; import
android.graphics.drawable.shapes.PathShape; import
android.graphics.drawable.shapes.RectShape;
import android.graphics.drawable.shapes.RoundRectShape; import
android.graphics.drawable.shapes.Shape;
import android.os.Bundle; import
android.view.View;

public class MainActivity extends GraphicsActivity { @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState); setContentView(new
SampleView(this));
}

private static class SampleView extends View { private
ShapeDrawable[] mDrawables;

private static Shader makeSweep() { return
new SweepGradient(150, 25,
new int[] { 0xFFFF0000, 0xFF00FF00, 0xFF0000FF, 0xFFFF0000 },
null);
}

private static Shader makeLinear() { return new
LinearGradient(0, 0, 50, 50,
new int[] { 0xFFFF0000, 0xFF00FF00, 0xFF0000FF },
null, Shader.TileMode.MIRROR);
}

private static Shader makeTiling() {
int[] pixels = new int[] { 0xFF00FF00, 0xFF0000FF,0xFFFF0000, 0}; Bitmap bm =
Bitmap.createBitmap(pixels, 2, 2, Bitmap.Config.ARGB_8888);
```

```
return new BitmapShader(bm, Shader.TileMode.REPEAT,
Shader.TileMode.REPEAT);
}

private static class MyShapeDrawable extends ShapeDrawable { private Paint
mStrokePaint = new Paint(Paint.ANTI_ALIAS_FLAG);

public MyShapeDrawable(Shape s) { super(s);
mStrokePaint.setStyle(Paint.Style.STROKE);
}

public Paint getStrokePaint() { return
mStrokePaint;
}

@Override protected void onDraw(Shape s, Canvas c, Paint p) { s.draw(c, p);
s.draw(c, mStrokePaint);
}
}

public SampleView(Context context) {
super(context);
setFocusable(true);

float[] outerR = new float[] { 12, 12, 12, 12, 0, 0, 0, 0 };
RectF inset = new RectF(6, 6, 6, 6);
float[] innerR = new float[] { 12, 12, 0, 0, 12, 12, 0, 0 };

Path path = new Path();
path.moveTo(50, 0);
path.lineTo(0, 50);
path.lineTo(50, 100);
path.lineTo(100, 50);
path.close();

mDrawables = new ShapeDrawable[7];
mDrawables[0] = new ShapeDrawable(new RectShape()); mDrawables[1] =
new ShapeDrawable(new OvalShape());
mDrawables[2] = new ShapeDrawable(new RoundRectShape(outerR, null, null));
mDrawables[3] = new ShapeDrawable(new RoundRectShape(outerR, inset,
```

```
null));
mDrawables[4] = new ShapeDrawable(new RoundRectShape(outerR, inset, innerR));
mDrawables[5] = new ShapeDrawable(new PathShape(path, 100, 100)); mDrawables[6] = new
MyShapeDrawable(new ArcShape(45, -270));

mDrawables[0].getPaint().setColor(0xFF0000FF); mDrawables[1].getPaint().setColor(0xFF00FF00);
mDrawables[2].getPaint().setColor(0xFFFF0000); mDrawables[3].getPaint().setShader(makeSweep());
mDrawables[4].getPaint().setShader(makeLinear()); mDrawables[5].getPaint().setShader(makeTiling());
mDrawables[6].getPaint().setColor(0x88FF8844);

PathEffect pe = new DiscretePathEffect(10, 4); PathEffect pe2 =
new CornerPathEffect(4);
mDrawables[3].getPaint().setPathEffect(
new ComposePathEffect(pe2, pe));

MyShapeDrawable msd = (MyShapeDrawable)mDrawables[6];
msd.getStrokePaint().setStrokeWidth(4);
}

@Override protected void onDraw(Canvas canvas) { int x = 10;
int y = 10;
int width = 300; int
height = 50;

for (Drawable dr : mDrawables) { dr.setBounds(x,
y, x + width, y + height); dr.draw(canvas);
y += height + 5;
}
}
}
}
```

5. Create a new java file with the name of **GraphicsActivity.java** and type the following code.

```
package com.example.drawingshapes; import
android.app.Activity;
import android.os.Bundle; import
android.view.View; import
android.view.ViewGroup;

class GraphicsActivity extends Activity {
private static final boolean TEST_PICTURE = false;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
}

@Override
public void setContentView(View view) { if
(TEST_PICTURE) {
ViewGroup vg = new PictureLayout(this);
vg.addView(view);
view = vg;
}

super.setContentView(view);
}
}
```

6. Create a new java file with the name of **PictureLayout.java** and type the following code.

```
package com.example.drawingshapes; import
android.content.Context; import
android.graphics.Canvas; import
android.graphics.Picture; import
android.graphics.Rect;
import android.graphics.drawable.Drawable; import
android.util.AttributeSet;
import android.view.View; import
android.view.ViewGroup; import
android.view.ViewParent;
public class PictureLayout extends ViewGroup { private
final Picture mPicture = new Picture(); public
PictureLayout(Context context) { super(context);
}
public PictureLayout(Context context, AttributeSet attrs) { super(context, attrs);
}
@Override
public void addView(View child) { if
(getChildCount() > 1) {
throw new IllegalStateException("PictureLayout can host only one direct child");
}
super.addView(child);
}
@Override
public void addView(View child, int index) { if
(getChildCount() > 1) {
throw new IllegalStateException("PictureLayout can host only one direct child");
}
super.addView(child, index);
}
@Override
public void addView(View child, LayoutParams params) { if
(getChildCount() > 1) {
throw new IllegalStateException("PictureLayout can host only one direct child");
}
```

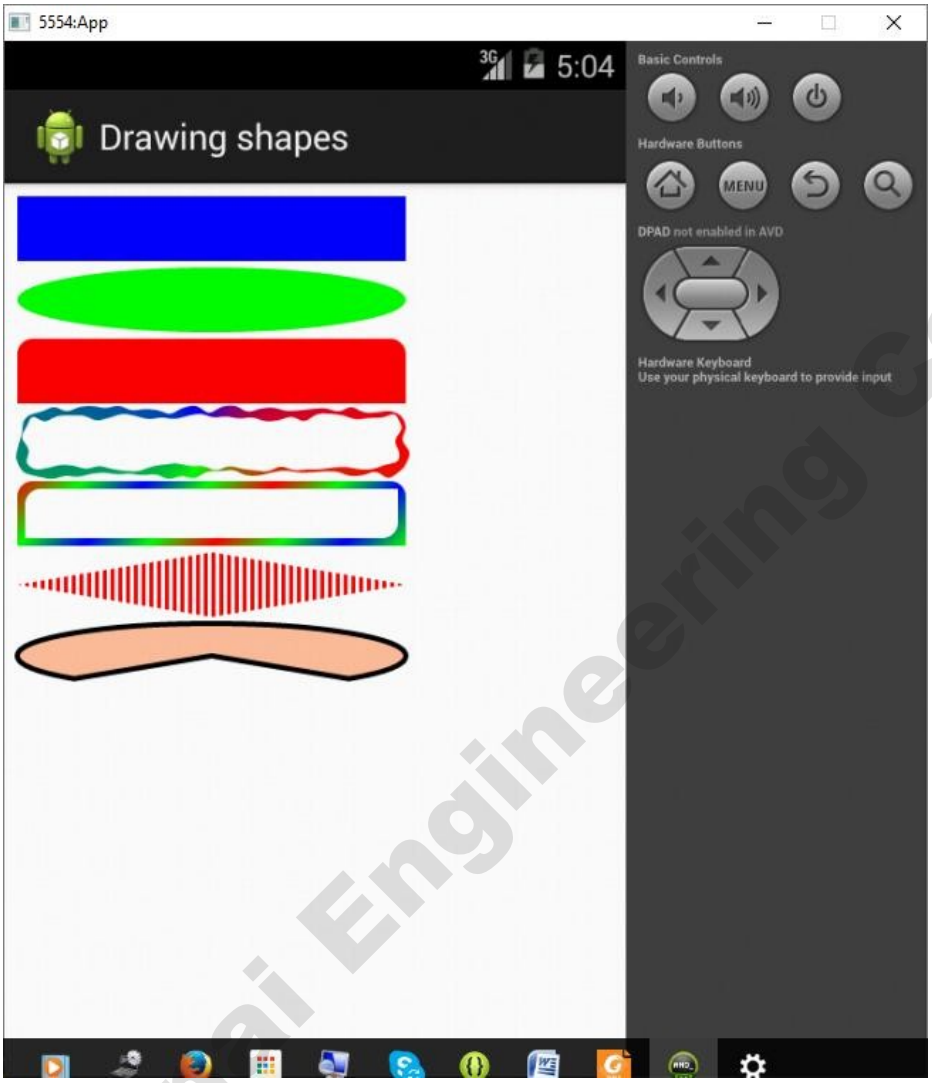
```
super.addView(child, params);
}
@Override
public void addView(View child, int index, LayoutParams params) { if
(getChildCount() > 1) {
throw new IllegalStateException("PictureLayout can host only one direct child");
}
super.addView(child, index, params);
}
@Override
protected LayoutParams generateDefaultLayoutParams() { return new
LayoutParams(LayoutParams.MATCH_PARENT,
LayoutParams.MATCH_PARENT);
}
@Override
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) { final int count =
getChildCount();
int maxHeight = 0; int
maxWidth = 0;
for (int i = 0; i < count; i++) { final View
child = getChildAt(i);
if (child.getVisibility() != GONE) {
measureChild(child, widthMeasureSpec, heightMeasureSpec);
}
}
maxWidth += getPaddingLeft() + getPaddingRight(); maxHeight +=
getPaddingTop() + getPaddingBottom(); Drawable drawable =
getBackground();
if (drawable != null) {
maxHeight = Math.max(maxHeight, drawable.getMinimumHeight()); maxWidth =
Math.max(maxWidth, drawable.getMinimumWidth());
}
setMeasuredDimension(resolveSize(maxWidth, widthMeasureSpec), resolveSize(maxHeight,
heightMeasureSpec));
}
private void drawPict(Canvas canvas, int x, int y, int w, int h, float sx,
float sy) {
canvas.save(); canvas.translate(x,
y); canvas.clipRect(0, 0, w, h);
canvas.scale(0.5f, 0.5f);
canvas.scale(sx, sy, w, h);
```

```
canvas.drawPicture(mPicture);
canvas.restore();
}

@SuppressWarnings("unused")
@Override
protected void dispatchDraw(Canvas canvas) { super.dispatchDraw(mPicture.beginRecording(getWidth(),
getHeight())); mPicture.endRecording();
int x = getWidth()/2; int y =
getHeight()/2; if (false) {
canvas.drawPicture(mPicture);
} else {
drawPict(canvas, 0, 0, x, y, 1, 1);
drawPict(canvas, x, 0, x, y, -1, 1);
drawPict(canvas, 0, y, x, y, 1, -1);
drawPict(canvas, x, y, x, y, -1, -1);
}
}
@Override
public ViewParent invalidateChildInParent(int[] location, Rect dirty){ location[0] = getLeft();
location[1] = getTop();
dirty.set(0, 0, getWidth(), getHeight()); return
getParent();
}
@Override
protected void onLayout(boolean changed, int l, int t, int r, int b) { final int count =
super.getChildCount();
for (int i = 0; i < count; i++) { final View
child = getChildAt(i);
if (child.getVisibility() != GONE) { final int
childLeft = getPaddingLeft(); final int childTop =
getPaddingTop(); child.layout(childLeft,
childTop, childLeft + child.getMeasuredWidth(),
childTop + child.getMeasuredHeight());

}
}
}
}
```

7. Run the app by rightclick on project package ,choose RunAs andchoose Android Applcation.

8. The output will be run in android emulator.

Output:



Result:

Thus the basic graphical primitives on the screen has been drawn in an android application successfully.

Ex.No.4 Develop an application that makes use of database

Introduction:

SQLite is a opensource SQL database that stores data to a text file on a device.
Android comes in with built in SQLite database implementation.

SQLite supports all the relational database features. In order to access this database, you don't need to establish any kind of connections for it like JDBC,ODBC e.t.c

Database - Package

The main package is android.database.sqlite that contains the classes to manage your own databases

Aim:

To develop an application to create, update, delete ,modify the data in a database.

Procedure:

1. Create a new project in Eclipse by navigating to **File ⇒ New ⇒Android Project** and fill all the required details.

2. Open **activity_main.xml** by navigating to **res ⇒layout⇒activity_main.xml** and type the following code

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:id="@+id/myLayout"
android:stretchColumns="0"
android:layout_width="fill_par
ent"
android:layout_height="fill_pa
rent">
<TextView android:text="title"
android:layout_x="110dp"
android:layout_y="10dp"
android:layout_width="wrap_content
"
android:layout_height="wrap_conten
t"/>
<TextView android:text="roll_no"
android:layout_x="30dp"
android:layout_y="50dp"
android:layout_width="wrap_content
"
android:layout_height="wrap_conten
t"/>
<EditText
android:id="@+id/editRollno"
android:inputType="number"
android:layout_x="150dp"
```

```
android:layout_y="50dp"
android:layout_width="150dp"
android:layout_height="40dp"/>
<TextView android:text="name"
```

```
android:layout_x="30dp"
android:layout_y="100dp"
android:layout_width="wrap_cont
ent"
android:layout_height="wrap_con
tent"/>
<EditText
android:id="@+id/editName"
android:inputType="text"
android:layout_x="150dp"
android:layout_y="100dp"
android:layout_width="150dp"
android:layout_height="40dp"/>
<TextView android:text="marks"
android:layout_x="30dp"
android:layout_y="150dp"
android:layout_width="wrap_cont
ent"
android:layout_height="wrap_con
tent"/>
<EditText
android:id="@+id/editMarks"
android:inputType="number"
android:layout_x="150dp"
android:layout_y="150dp"
android:layout_width="150dp"
android:layout_height="40dp"/>
<Button
android:id="@+id/btnAdd"
android:text="add"
android:layout_x="30dp"
android:layout_y="200dp"
android:layout_width="100dp"
android:layout_height="40dp"/>
<Button
android:id="@+id/btnDelete"
android:text="delete"
android:layout_x="150dp"
android:layout_y="200dp"
android:layout_width="100dp"
android:layout_height="40dp"/>
n
<Button
android:id="@+id/btnModify"
android:text="modify"
android:layout_x="30dp"
android:layout_y="250dp"
android:layout_width="100dp"
android:layout_height="40dp"/>
<Button
android:id="@+id/btnView"
android:text="view"
android:layout_x="150dp"
```

```
android:layout_y="250dp"
android:layout_width="100dp"
android:layout_height="40dp"/>
<Button
android:id="@+id/btnViewAll"
android:text="view_all"
android:layout_x="30dp"
android:layout_y="300dp"
android:layout_width="100dp"
android:layout_height="40dp"/>
<Button
android:id="@+id/btnShowInfo"
android:text="show_info"
android:layout_x="150dp"
android:layout_y="300dp"
android:layout_width="100dp"
android:layout_height="40dp"/>
</AbsoluteLayout>
```

3. Open **MainActivity.java** by navigating through **src->MainActivity.java** and type the following code.

package com.example.databaseeg;

```
import android.app.Activity;
import android.app.AlertDialog.Builder; import
android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase; import
android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener; import
android.widget.Button;
import android.widget.EditText;

public class MainActivity extends Activity implements OnClickListener
{
EditText editRollno,editName,editMarks;
Button btnAdd,btnDelete,btnModify,btnView,btnViewAll,btnShowInfo; SQLiteDatabase db;
/** Called when the activity is first created. */ @Override
public void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState); setContentView(R.layout.activity_main);
editRollno=(EditText)findViewById(R.id.editRollno);
editName=(EditText)findViewById(R.id.editName);
editMarks=(EditText)findViewById(R.id.editMarks);
btnAdd=(Button)findViewById(R.id.btnAdd);
btnDelete=(Button)findViewById(R.id.btnDelete);
btnModify=(Button)findViewById(R.id.btnModify);
btnView=(Button)findViewById(R.id.btnView);
btnViewAll=(Button)findViewById(R.id.btnViewAll);
btnShowInfo=(Button)findViewById(R.id.btnShowInfo);
btnAdd.setOnClickListener(this); btnDelete.setOnClickListener(this);
btnModify.setOnClickListener(this); btnView.setOnClickListener(this);
btnViewAll.setOnClickListener(this); btnShowInfo.setOnClickListener(this);
db=openOrCreateDatabase("StudentDB", Context.MODE_PRIVATE, null);
```

```
db.execSQL("CREATE TABLE IF NOT EXISTS student(rollno VARCHAR,name
VARCHAR,marks VARCHAR);");
}
public void onClick(View view)
{
if(view==btnAdd)
{
if(editRollno.getText().toString().trim().length()==0|| editName.getText().toString().trim().length()==0||
editMarks.getText().toString().trim().length()==0)
{
showMessage("Error", "Please enter all values"); return;
}
db.execSQL("INSERT INTO student VALUES('"+editRollno.getText()+"','"+editName.getText()+
"','"+editMarks.getText()+"');");
showMessage("Success", "Record added");
clearText();
}
if(view==btnDelete)
{
if(editRollno.getText().toString().trim().length()==0)
{
showMessage("Error", "Please enter Rollno"); return;
}
Cursor c=db.rawQuery("SELECT * FROM student WHERE
rollno='"+editRollno.getText()+"'", null); if(c.moveToFirst())
{
db.execSQL("DELETE FROM student WHERE
rollno='"+editRollno.getText()+"'"); showMessage("Success",
"Record Deleted");
}
else
{
showMessage("Error", "Invalid Rollno");
}
clearText();
}
if(view==btnModify)
{
```

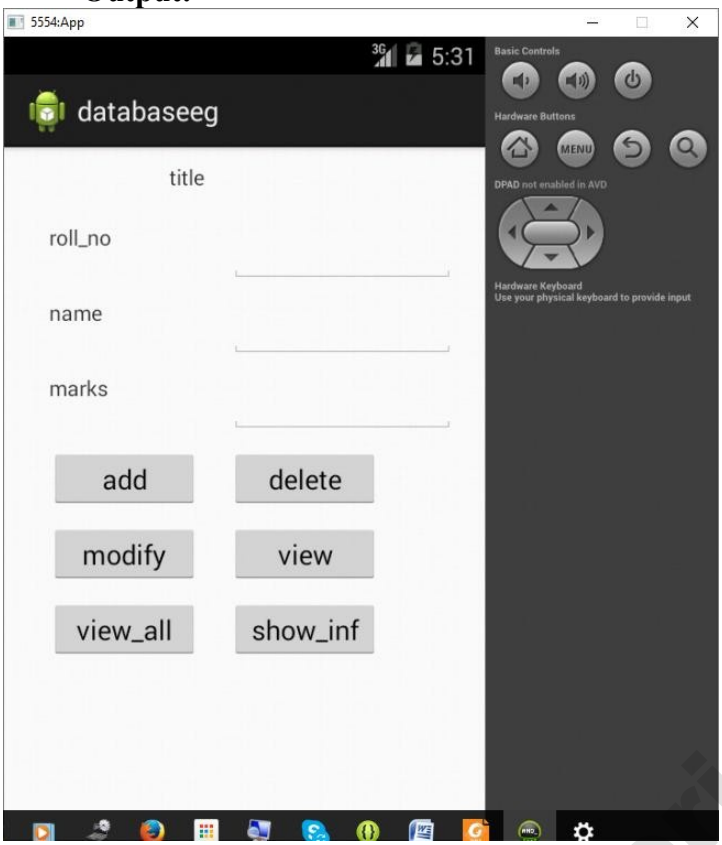
```
if(editRollno.getText().toString().trim().length()==0)
{
showMessage("Error", "Please enter Rollno"); return;
}
Cursor c=db.rawQuery("SELECT * FROM student WHERE
rollno='"+editRollno.getText()+"'", null); if(c.moveToFirst())
{
db.execSQL("UPDATE student SET
name='"+editName.getText()+"',marks='"+editMarks.getText()+   "' WHERE
rollno='"+editRollno.getText()+"'"); showMessage("Success", "Record
Modified");
}
else
{
showMessage("Error", "Invalid Rollno");
}
clearText();
}
if(view==btnView)
{
if(editRollno.getText().toString().trim().length()==0)
{
showMessage("Error", "Please enter Rollno"); return;
}
Cursor c=db.rawQuery("SELECT * FROM student WHERE
rollno='"+editRollno.getText()+"'", null); if(c.moveToFirst())
{
editName.setText(c.getString(1)); editMarks.setText(c.getString(2));
}
else
{
showMessage("Error", "Invalid Rollno"); clearText();
}
}
if(view==btnViewAll)
{
Cursor c=db.rawQuery("SELECT * FROM student", null);
```

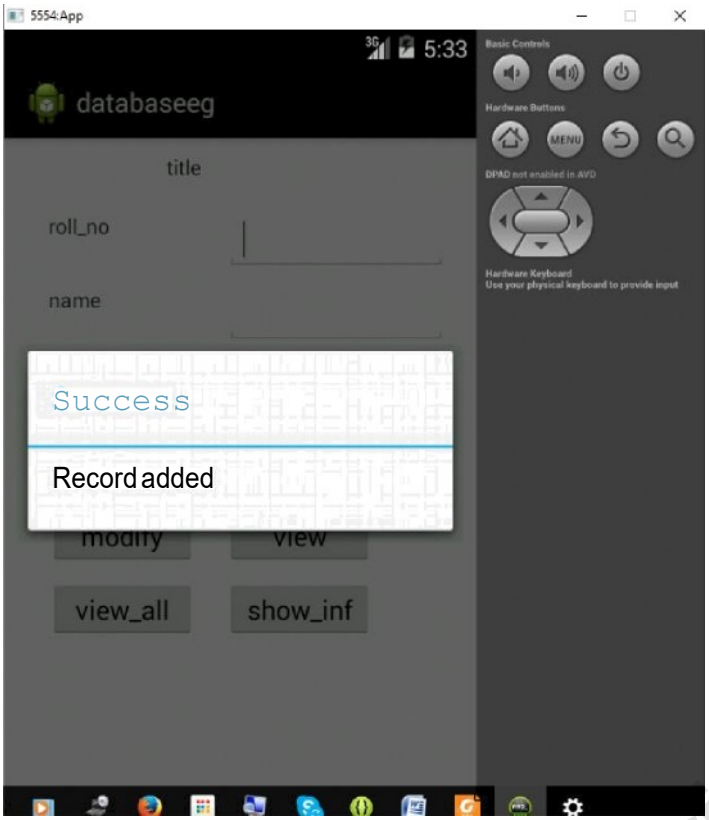
```
if(c.getCount()==0)
{
showMessage("Error", "No records found"); return;
}
StringBuffer buffer=new StringBuffer(); while(c.moveToNext())
{
buffer.append("Rollno: "+c.getString(0)+"\n");
buffer.append("Name: "+c.getString(1)+"\n");
buffer.append("Marks: "+c.getString(2)+"\n\n");
}
showMessage("Student Details", buffer.toString());
}
if(view==btnShowInfo)
{
showMessage("Student Management Application", "Arunai Engineering College");
}
}
public void showMessage(String title,String message)
{
Builder builder=new Builder(this);
builder.setCancelable(true);
builder.setTitle(title);
builder.setMessage(message);
builder.show();
}
public void clearText()
{
editRollno.setText("");
editName.setText("");
editMarks.setText("");
editRollno.requestFocus();
}
}
```

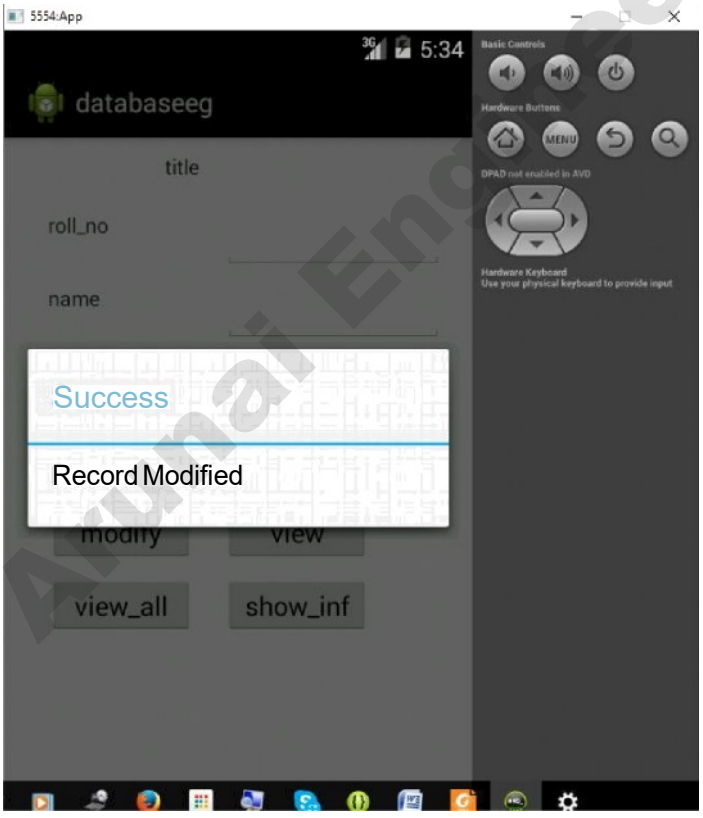
4. Run the app by rightclick on project package ,choose RunAs and choose Android Applcation.

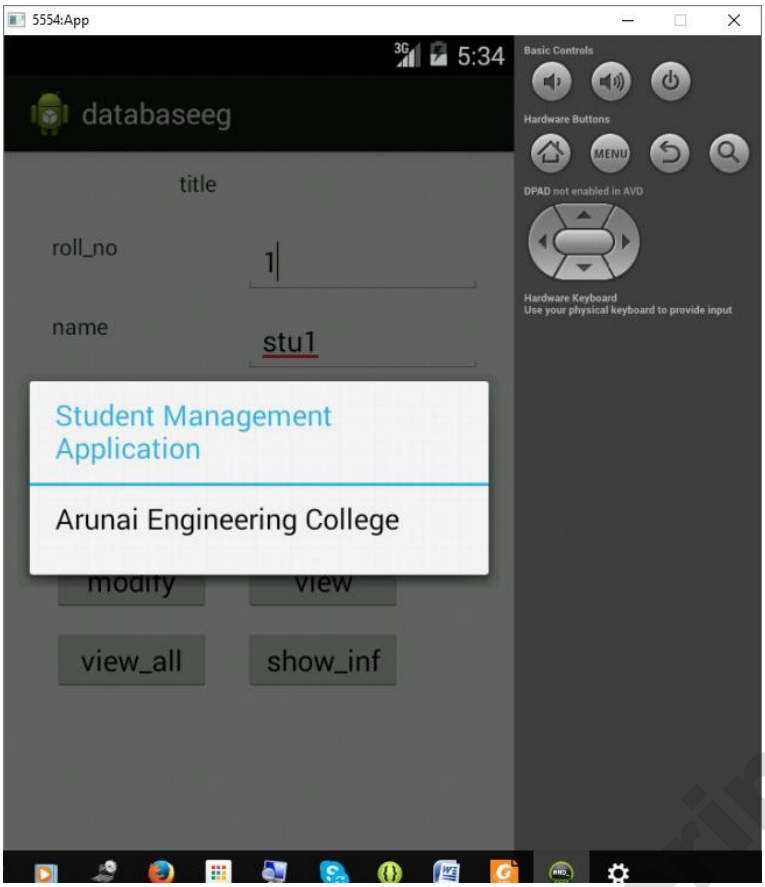
5. The output will be run in android emulator.

Output:









Result:

Thus an android based database application has been developed successfully.

Ex.No:5. Develop an application that makes use of Notification Manager.

Aim:

To develop an application that makes use of Notification Manager.

Procedure:

1.Open eclipse or android studio and select new android project .
2.Give project name and select next
3. Choose the android version.Choose the lowest android version(Android 2.2) and select next
4. Enter the package name.package name must be two word seprated by comma and click finish
5.Go to package explorer in the left hand side.select our project.
6.Go to res folder and select layout.Double click the main.xml file
7.Now you can see the Graphics layout window.

Program :

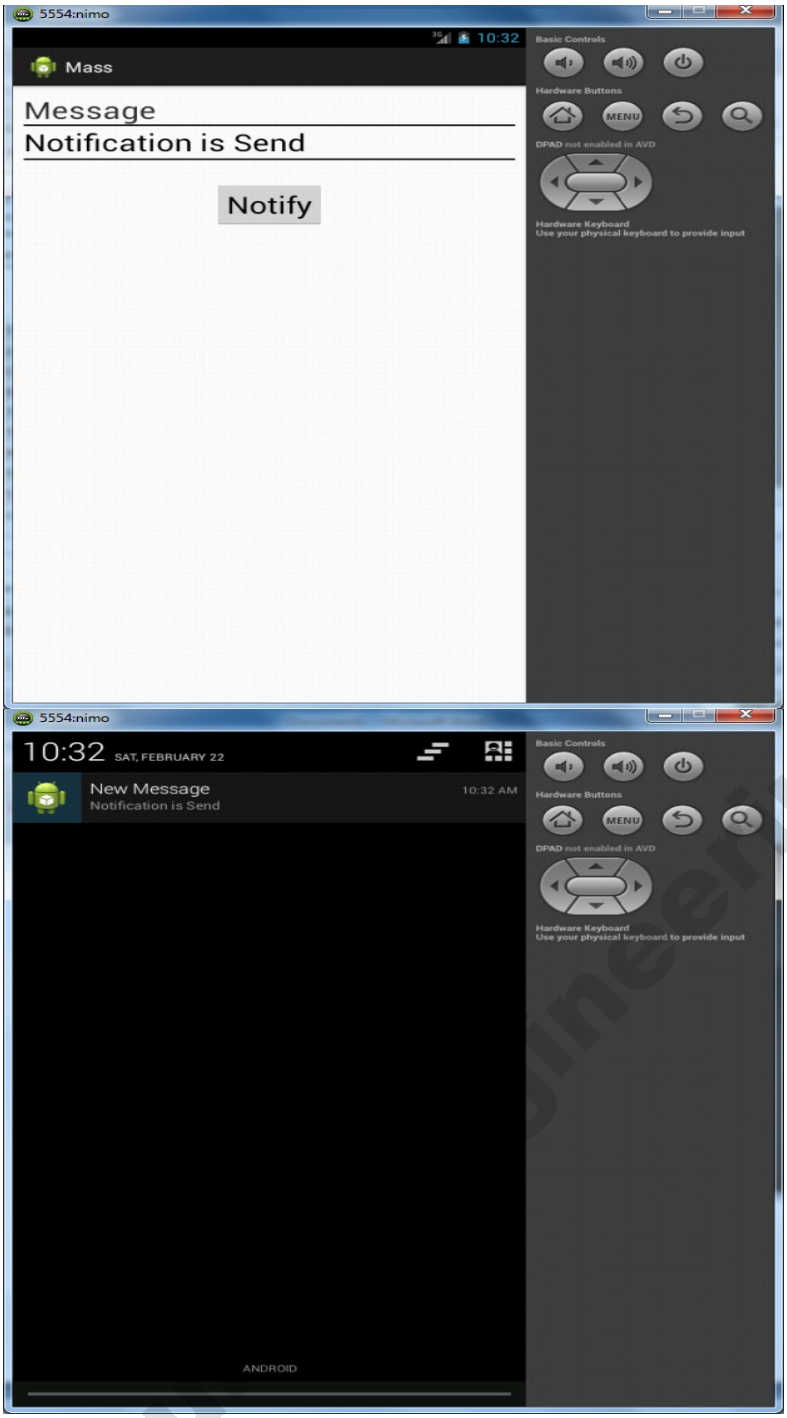
Activity Main :

```
<?xmlversion="1.0"encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"

android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_margin="10dp"

android:orientation="vertical">

<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Message"
android:textSize="30sp"/>

<EditText
android:id="@+id/editText"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:background="#ffffff"
android:singleLine="true"
android:textSize="30sp"/>

<Button
android:id="@+id/button"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_margin="30dp"
android:layout_gravity="center"
android:text="Notify"
android:textSize="30sp"/>

</LinearLayout>
```

Main Activity :

```
package com.example.mass;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Intent;
import android.os.Bundle;
import android.app.Activity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends Activity
{
   Button notify;
   EditText e;
   @Override
   protected void onCreate(Bundle savedInstanceState)
   {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      notify= (Button) findViewById(R.id.button);
      e= (EditText) findViewById(R.id.editText);

      notify.setOnClickListener(new View.OnClickListener()
      {
         @Override
         public void onClick(View v)
         {
            Intent intent = new Intent(MainActivity.this, MainActivity.class);
            PendingIntent pending = PendingIntent.getActivity(MainActivity.this, 0, intent, 0);
            Notification noti = new Notification.Builder(MainActivity.this).setContentTitle("New
Message").setContentText(e.getText().toString()).setSmallIcon(R.drawable.ic_launcher).setContentIntent(pendi
ng).build();
            NotificationManager manager = (NotificationManager)
getSystemService(NOTIFICATION_SERVICE);
            noti.flags |= Notification.FLAG_AUTO_CANCEL;
            manager.notify(0, noti);
         }
      });
   }
}
```

Output :



Result:

Thus an android application for notification manager has been developed and implemented successfull

Ex.No.6Implement an application that implements Multithreading

Introduction:

Multi-threading is defined as a feature through which we can run two or more concurrent threads of a process. In this a process, the common data is shared among all these threads also known as sub-processes exclusively. In android there are many ways through which multi-threading can be established in the application.

Objective:

- Understanding the basic concept of multithreading.
- Understanding of Handler class in android
- Understanding of Runnable Interface.

Multi-Threading In Android:

Multi-Threading in Android is a unique feature through which more than one threads execute together without hindering the execution of other threads.
Multi-Threading in Android is not different from conventional multi-Threading. A class can be thought of as a process having its method as it's sub-processes or threads. All these methods can run concurrently by using feature of Multi- Threading. In android, multi-Threading can be achieved through the use of many in-built classes. Out of them, Handler class is most commonly used.

Handler class in Android:

Handler class come from the Package android.os.Handler package and is most commonly used for multi-threading in android. Handler class provide sending and receiving feature for messages between different threads and handle the thread execution which is associated with that instance of Handler class.

In android class, every thread is associated with an instance of Handler class and it allows the thread to run along with other threads and communicate with them through messages.

Aim:

To develop an android application that implements Multithreading concept.

Procedure:

1. Create a new project in Eclipse by navigating to **File ⇒ New ⇒Android Project** and fill all the required details.

2. Open **activity_main.xml** by navigating to **res ⇒layout⇒activity_main.xml** and type the following code

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_p
arent"
tools:context=".MainActivity" >

<TextView
android:id="@+id/tv_time"
android:layout_width="wrap_co
ntent"
android:layout_height="wrap_c
ontent" android:textSize="10pt"
android:textColor="#444444"
android:layout_alignParentLeft=
"true"
android:layout_marginRight="9
dip"
android:layout_marginTop="20
dip"
android:layout_marginLeft="10
dip" android:text="Sleep
time(ms)"/>

<EditText
android:id="@+id/et_time
"
android:layout_width="15
0dip"
android:layout_height="wrap_content"
android:layout_alignParentLeft="true"
android:layout_below="@+id/tv_time"
android:layout_marginTop="15dp"
android:background="@android:drawable/editbox_b
ackground" android:ems="10"
android:inputType="text" >

<requestFocus />
</EditText>
```

```
<Button
android:id="@+id/btn_do
_it"
android:layout_width="20
0dip"
android:layout_height="wrap_c
ontent"
android:layout_alignParentLeft=
"true"
android:layout_below="@+id/et
_time" android:text="Run Async
task" />

<TextView
android:id="@+id/tv_resu
lt"
android:layout_width="40
0dip"
```

```
    android:layout_height="100dip"
    android:layout_alignLeft="@+id/t
    v_time"
    android:layout_centerVertical="tru
    e"
    android:layout_marginLeft="26dp"
    android:textColor="#AA0000"
    android:textSize="7pt" />
    </RelativeLayout>
```

3. Open **MainActivity.java** by navigating through **src->MainActivity.java** and type the following code.

```
package com.example.threadpgm; import
android.app.Activity; import
android.os.AsyncTask; import
android.os.Bundle;
import android.view.Menu; import
android.view.View; import
android.widget.Button; import
android.widget.EditText; import
android.widget.TextView;
/**
 * @author
 * AsyncTask exmple
*
*/
public class MainActivity extends Activity { private
Button button;
private EditText time;
private TextView finalResult;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
time = (EditText) findViewById(R.id.et_time); button =
(Button) findViewById(R.id.btn_do_it);
finalResult = (TextView) findViewById(R.id.tv_result);
button.setOnClickListener(new View.OnClickListener() { @Override
public void onClick(View v) {
AsyncTaskRunner runner = new AsyncTaskRunner(); String
sleepTime = time.getText().toString();
runner.execute(sleepTime);
}
});
}
```

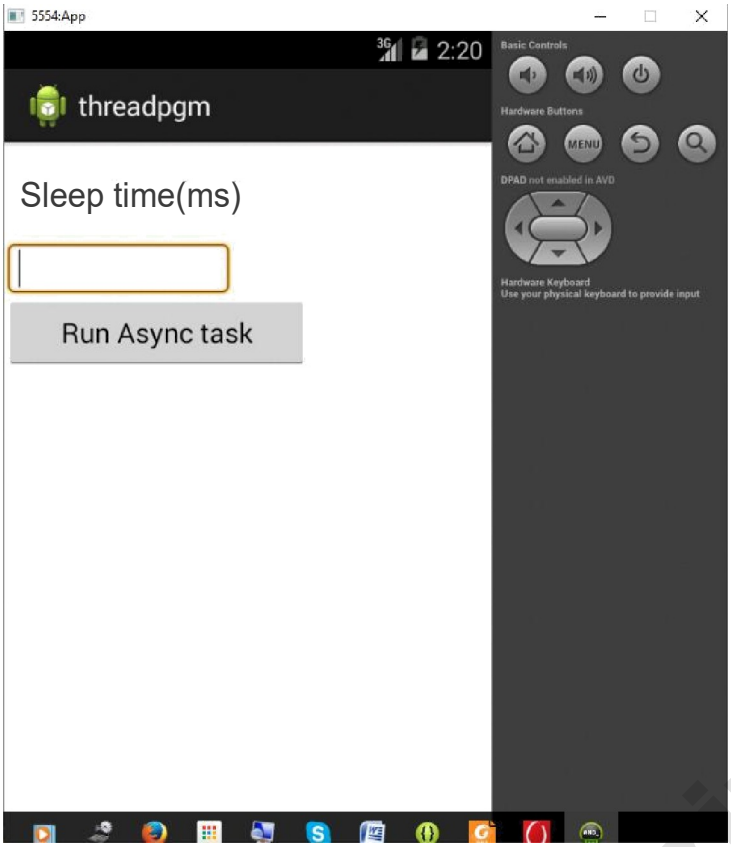
```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present.
//getMenuInflater().inflate(R.menu., menu); return
true;
}
/**
    * @author Prabu
    * Private class which runs the long operation. ( Sleeping for some time )
*/
private class AsyncTaskRunner extends AsyncTask<String, String, String> { private String resp;
@Override
protected String doInBackground(String... params) {
publishProgress("Sleeping..."); // Calls onProgressUpdate() try {
// Do your long operations here and return the result int time =
Integer.parseInt(params[0]);
// Sleeping for given time period
Thread.sleep(time);
resp = "Slept for " + time + " milliseconds";
} catch (InterruptedException e) {
e.printStackTrace();
resp = e.getMessage();
} catch (Exception e) {
e.printStackTrace(); resp =
e.getMessage();
}
return resp;
}
/*
    * (non-Javadoc)
*
    * @see android.os.AsyncTask#onPostExecute(java.lang.Object)
*/ @Override
protected void onPostExecute(String result) {
// execution of result of Long time consuming operation finalResult.setText(result);
}
/*
    * (non-Javadoc)
*
```

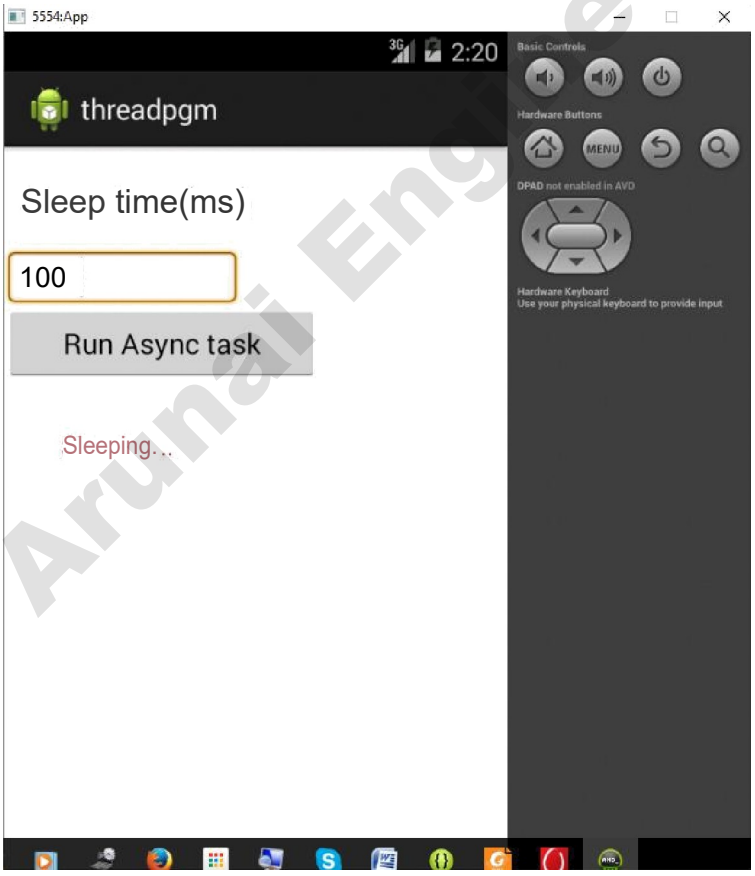
```
     * @see android.os.AsyncTask#onPreExecute()
*/ @Override
protected void onPreExecute() {
// Things to be done before execution of long running operation. For
// example showing ProgessDialog
}
/*
     * (non-Javadoc)
*
     * @see android.os.AsyncTask#onProgressUpdate(Progress[])
*/ @Override
protected void onProgressUpdate(String... text) {
finalResult.setText(text[0]);
// Things to be done while execution of long running operation is in
// progress. For example updating ProgessDialog
}
}
}
```

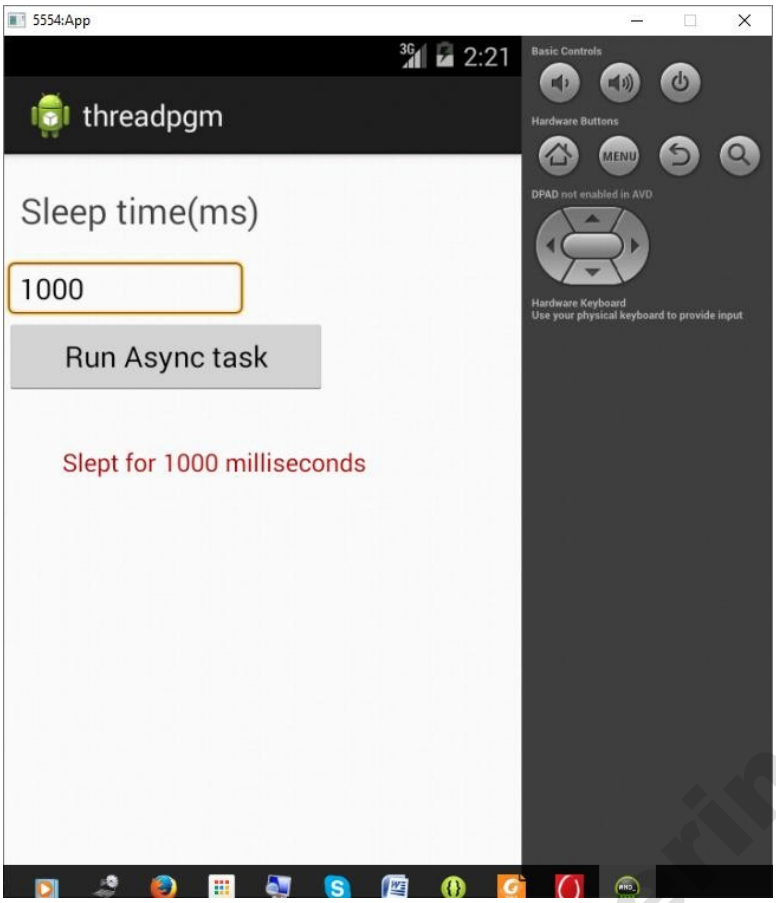
4. Run the app by rightclick on project package ,choose RunAs and choose Android Applcation.

5. The output will be run in android emulator.

Output:







Result:

Thus Multithreading has been implemented in an android application successfully.

Ex.No.7 Develop a native application that uses GPS location information

Introduction:

Android devices use the same global positioning technology as Google Maps and most third-party GPS tools do. This allows users to locate themselves on a map, find and navigate to destinations via detailed directions, and search maps using a number of different methods.

Aim:

To develop an application that tracks the GPS location and display the latitude and longitude.

Procedure:

1. Create a new project in Eclipse by navigating to **File ⇒ New ⇒Android Project** and fill all the required details.

2. Open **activity_main.xml** by navigating to **res ⇒layout⇒activity_main.xml** and type the following code

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical
_margin"
android:paddingLeft="@dimen/activity_horizontal
_margin"
android:paddingRight="@dimen/activity_horizonta
l_margin"
android:paddingTop="@dimen/activity_vertical_m
argin" tools:context=".MainActivity" >

<TextView
android:id="@+id/textView1"
android:layout_width="wrap_co
ntent"
android:layout_height="wrap_c
ontent"
android:text="@string/hello_wo
rld" />

<Button
android:id="@+id/btnShowLocation"
android:layout_width="wrap_content
"
android:layout_height="wrap_content
"
android:layout_below="@+id/textVie
w1"
```

```
android:layout_marginLeft="19dp"
android:layout_marginTop="68dp"
android:layout_toRightOf="@+id/tex
tView1" android:text="Show location"
/>

</RelativeLayout>
```

3. Create new java file under the src folder with the name of **GPSTracker.java** and type the following code

```
package com.example.locationgps;

import android.app.AlertDialog; import
android.app.Service; import
android.content.Context;
import android.content.DialogInterface; import
android.content.Intent;
import android.location.Location;
import android.location.LocationListener; import
android.location.LocationManager; import
android.os.Bundle;
import android.os.IBinder; import
android.provider.Settings; import
android.util.Log;

public class GPSTracker extends Service implements LocationListener { private final

Context mContext;

// flag for GPS status
boolean isGPSEnabled = false;

// flag for network status
boolean isNetworkEnabled = false;

// flag for GPS status
boolean canGetLocation = false;

Location location; // location double
latitude; // latitude double longitude;
// longitude

// The minimum distance to change Updates in meters
private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES = 10; // 10 meters

// The minimum time between updates in milliseconds
private static final long MIN_TIME_BW_UPDATES = 1000 * 60 * 1; // 1 minute

// Declaring a Location Manager
protected LocationManager locationManager;
```

```
public GPSTracker(Context context) {
this.mContext = context; getLocation();
}

public Location getLocation() { try {
locationManager = (LocationManager) mContext
.getSystemService(LOCATION_SERVICE);

// getting GPS status isGPSEnabled =
locationManager
.isProviderEnabled(LocationManager.GPS_PROVIDER);

// getting network status isNetworkEnabled =
locationManager
.isProviderEnabled(LocationManager.NETWORK_PROVIDER);

if (!isGPSEnabled && !isNetworkEnabled) {
// no network provider is enabled
} else { this.canGetLocation =
true;
// First get location from Network Provider if
(isNetworkEnabled) {
locationManager.requestLocationUpdates(
LocationManager.NETWORK_PROVIDER,
MIN_TIME_BW_UPDATES,
MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
Log.d("Network", "Network"); if
(locationManager != null) { location =
locationManager
.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);    if (location !=
null) {
latitude = location.getLatitude(); longitude =
location.getLongitude();
}
}
}
// if GPS Enabled get lat/long using GPS Services if
(isGPSEnabled) {
if (location == null) { locationManager.requestLocationUpdates(
```

```
LocationManager.GPS_PROVIDER,
MIN_TIME_BW_UPDATES,
MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
Log.d("GPS Enabled", "GPS Enabled"); if
(locationManager != null) {
location = locationManager
.getLastKnownLocation(LocationManager.GPS_PROVIDER);   if (location
!= null) {
latitude = location.getLatitude(); longitude =
location.getLongitude();
}
}
}
}
}

} catch (Exception e) {
e.printStackTrace();
}

return location;
}

/**
    * Stop using GPS listener
    * Calling this function will stop using GPS in your app
* */
public void stopUsingGPS(){
if(locationManager != null){
locationManager.removeUpdates(GPSTracker.this);
}
}

/**
    * Function to get latitude
* */
public double getLatitude(){ if(location
!= null){
latitude = location.getLatitude();
}

// return latitude
return latitude;
```

```
}

/**
    * Function to get longitude
* */
public double getLongitude(){
if(location != null){
longitude = location.getLongitude();
}

// return longitude
return longitude;
}

/**
    * Function to check GPS/wifi enabled
    * @return boolean
* */
public boolean canGetLocation() { return
this.canGetLocation;
}

/**
    * Function to show settings alert dialog
    * On pressing Settings button will lauch Settings Options
* */
public void showSettingsAlert(){
AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);

// Setting Dialog Title alertDialog.setTitle("GPS
is settings");

// Setting Dialog Message
alertDialog.setMessage("GPS is not enabled. Do you want to go to settings menu?");

// On pressing Settings button
alertDialog.setPositiveButton("Settings", new
DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog,int which) { Intent intent
= new
Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS); mContext.startActivity(intent);
```

```
}
});

// on pressing cancel button
alertDialog.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog, int which) { dialog.cancel();
}
});

// Showing Alert Message
alertDialog.show();
}

@Override
public void onLocationChanged(Location location) {
}

@Override
public void onProviderDisabled(String provider) {
}

@Override
public void onProviderEnabled(String provider) {
}

@Override
public void onStatusChanged(String provider, int status, Bundle extras) {
}

@Override
public IBinder onBind(Intent arg0) { return
null;
}

}
```

4. Open **MainActivity.java** by navigating through **src->MainActivity.java** and type the following code.

```
package com.example.locationgps;
```

```
import android.app.Activity; import
android.os.Bundle; import
android.view.View; import
android.widget.Button; import
android.widget.Toast;

public class MainActivity extends Activity { Button

btnShowLocation;

// GPSTracker class
GPSTracker gps;

@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

btnShowLocation = (Button) findViewById(R.id.btnShowLocation);

// show location button click event btnShowLocation.setOnClickListener(new
View.OnClickListener() {

@Override
public void onClick(View arg0) {
// create class object
gps = new GPSTracker(MainActivity.this);

// check if GPS enabled
if(gps.canGetLocation()){

double latitude = gps.getLatitude(); double
longitude = gps.getLongitude();

// \n is for new line
Toast.makeText(getApplicationContext(), "Your Location is - \nLat: " + latitude
+ "\nLong: " + longitude, Toast.LENGTH_LONG).show();
}else{
// can't get location
// GPS or Network is not enabled
// Ask user to enable GPS/network in settings
gps.showSettingsAlert();
```

```
}
}
});
}}
```

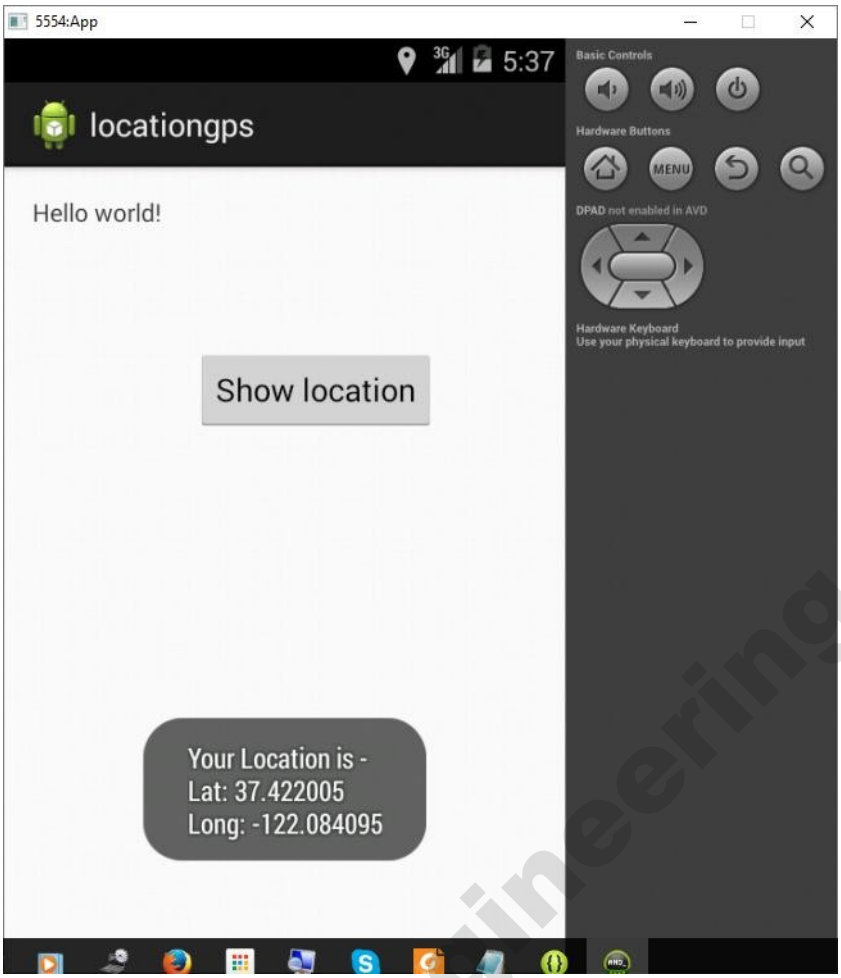
6. Open **AndroidManifest.xml** and add the following line

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

7. Run the app by rightclick on project package ,choose RunAs and choose Android Applcation.

8. The output will be run in android emulator.

Output:



Result:

Thus an android application has been created to track the location using GPS.

Ex.No.8Implement an application that writes data to the SD Card

Introduction:

This sample android program shows you how write and read a file from SD Card in Android. In this program four buttons are shown and a Edit box. When you type some text into the edit box and click, Save to SD Card button, the text is saved to a text file and saved to the SD Card. When you click clear button, the edit box contents are cleared. When you click, Read Sd card button the file is read from the SD card and the contents are copied to the edit box.

Aim:

To create an application that writes the data in to the file from SD card.

Procedure:

1. Create a new project in Eclipse by navigating to **File ⇒ New ⇒Android Project** and fill all the required details.

2. Open **activity_main.xml** by navigating to **res ⇒layout⇒activity_main.xml** and type the following code

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertic
al"
android:layout_gravity="c
enter"
tools:context=".MainActivi
ty" >
<TextView
android:layout_width="fill_parent"
android:layout_height="wrap_cont
ent" android:gravity="center"
android:textAlignment="center"
android:text="Android Read/Write
File" />
<EditText
android:layout_width="fill_pare
nt"
android:layout_height="wrap_c
ontent"
android:id="@+id/fname"
android:hint="File Name" />
<EditText
android:layout_width="fill_p
arent"
android:layout_height="100p
x" android:id="@+id/ftext"
android:hint="File Text" />
```

```
<Button
android:layout_width="fill_pare
nt"
android:layout_height="wrap_c
ontent"
android:id="@+id/btnwrite"
android:text="Write File" />
<EditText
android:layout_width="fill_p
arent"
```

```
    android:layout_height="wrap_c
    ontent"
    android:id="@+id/fnameread"
    android:hint="File Name" />
    <Button
    android:layout_width="fill_pare
    nt"
    android:layout_height="wrap_c
    ontent"
    android:id="@+id/btnread"
    android:text="Read File" />
    <TextView
    android:layout_width="fill_pare
    nt"
    android:layout_height="wrap_c
    ontent"
    android:id="@+id/filecon" />

    </LinearLayout>
```

3. Create a new file under the src folder with the name of **FileOperations.java** and type the following code

```
package com.example.fileapp; import
java.io.BufferedReader; import
java.io.BufferedWriter; import
java.io.File;
import java.io.FileReader; import
java.io.FileWriter; import
java.io.IOException; import
android.util.Log; public class
FileOperations { public
FileOperations() {
}
public Boolean write(String fname, String fcontent){ try {
String fpath = "/sdcard/"+fname+".txt"; File file
= new File(fpath);
// If file does not exists, then create it if
(!file.exists()) { file.createNewFile();
}
FileWriter fw = new FileWriter(file.getAbsoluteFile());
BufferedWriter bw = new BufferedWriter(fw); bw.write(fcontent);
bw.close();
Log.d("Suceess","Sucess"); return
true;
} catch (IOException e) {
e.printStackTrace(); return false;
}
}
```

```
public String read(String fname){
BufferedReader br = null;
String response = null; try {
StringBuffer output = new StringBuffer(); String
fpath = "/sdcard/"+fname+".txt";
br = new BufferedReader(new FileReader(fpath)); String line = "";
while ((line = br.readLine()) != null) {
output.append(line +"n");
}
response = output.toString();
} catch (IOException e) {
e.printStackTrace(); return null;
}
return response;
}
}
```

4. Open **MainActivity.java** by navigating through **src->MainActivity.java** and type the following code.

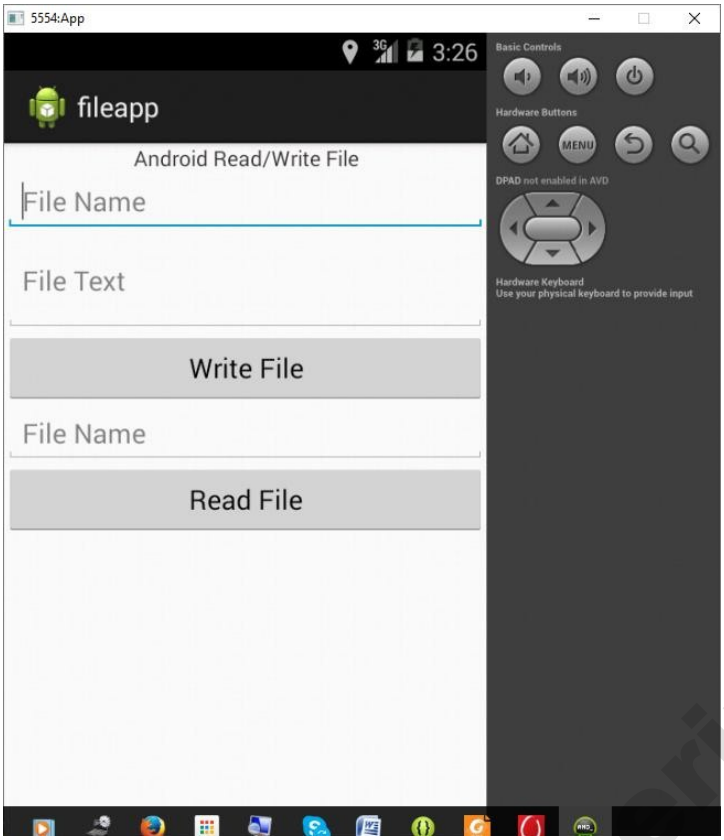
```
package com.example.fileapp; import
android.os.Bundle; import
android.view.View; import
android.widget.Button; import
android.widget.EditText;
import android.widget.TextView; import
android.widget.Toast; import
android.app.Activity;
public class MainActivity extends Activity { EditText
fname,fcontent,fnameread;
Button          write,read;
TextView          filecon;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
fname = (EditText)findViewById(R.id.fname); fcontent =
(EditText)findViewById(R.id.ftext); fnameread =
(EditText)findViewById(R.id.fnameread); write =
(Button)findViewById(R.id.btnwrite);
```

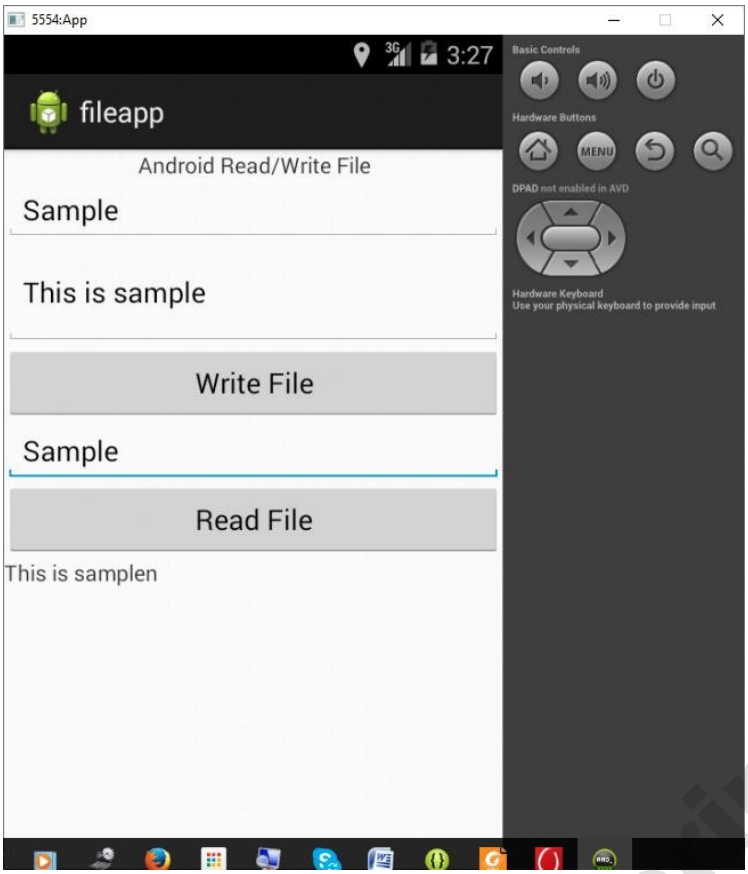
```
read = (Button)findViewById(R.id.btnread); filecon =
(TextView)findViewById(R.id.filecon);
write.setOnClickListener(new View.OnClickListener() { @Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
String filename = fname.getText().toString(); String
filecontent = fcontent.getText().toString(); FileOperations
fop = new FileOperations(); fop.write(filename, filecontent);
if(fop.write(filename, filecontent)){
Toast.makeText(getApplicationContext(), filename+".txt created", Toast.LENGTH_SHORT).show();
}else{
Toast.makeText(getApplicationContext(), "I/O error",
Toast.LENGTH_SHORT).show();
}
}
});
read.setOnClickListener(new View.OnClickListener() { @Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
String readfilename = fnameread.getText().toString();
FileOperations fop = new FileOperations();
String text = fop.read(readfilename); if(text
!= null){ filecon.setText(text);
}
else {
Toast.makeText(getApplicationContext(), "File not Found", Toast.LENGTH_SHORT).show();
filecon.setText(null);
}
}
});
}
}
```

5. Run the app by rightclick on project package ,choose RunAs and choose Android Applcation.

6. The output will be run in android emulator.

Output:





Result:

Thus an application has been created to write and read the file from the SD card successfully.

Ex.No.9 Implement an application that creates an alert upon receiving a message.

Introduction:

A notification is a message you can display to the user outside of your application's normal UI. When you tell the system to issue a notification, it first appears as an icon in the **notification area**. To see the details of thenotification, the user opens the **notification drawer**. Both the notification area and the notification drawer are system-controlled areas that the user can view at any time.

Aim:

To create an android application that creates an alert upon receiving a message.

Procedure:

1. Create a new project in Eclipse by navigating to **File ⇒ New ⇒Android Project** and fill all the required details.

2. Open **activity_main.xml** by navigating to **res ⇒layout⇒activity_main.xml** and type the following code

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical
_margin"
android:paddingLeft="@dimen/activity_horizontal
_margin"
android:paddingRight="@dimen/activity_horizonta
l_margin"
android:paddingTop="@dimen/activity_vertical_m
argin" tools:context=".MainActivity" >

<Button
android:id="@+id/notificationBu
tton"
android:layout_width="wrap_co
ntent"
android:layout_height="wrap_co
ntent"
android:layout_alignParentTop=
"true"
android:layout_centerHorizontal
="true"
android:layout_marginTop="46d
p" android:text="Notify" />

</RelativeLayout>
```

3. Open **MainActivity.java** by navigating through **src->MainActivity.java** and type the following code.

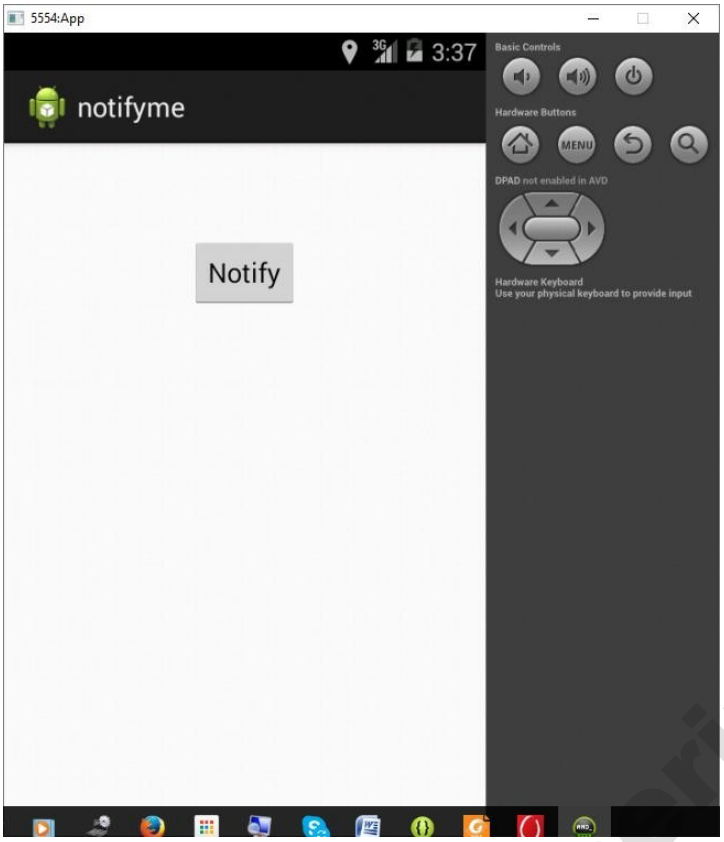
```
package com.example.notifyme; import
android.app.Activity;
```

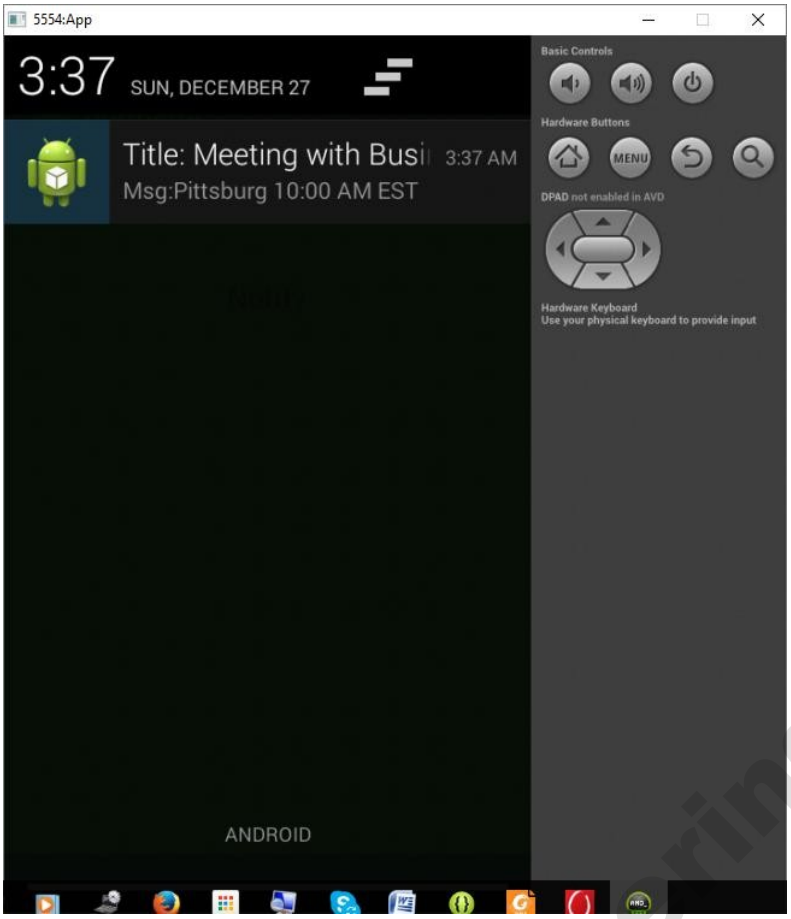
```
import android.app.Notification;
import android.app.NotificationManager; import
android.app.PendingIntent; import
android.content.Intent;
import android.os.Bundle; import
android.view.View; import
android.widget.Button;
public class MainActivity extends Activity {
@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
Button notificationButton = (Button) findViewById(R.id.notificationButton);
notificationButton.setOnClickListener(new View.OnClickListener() { @Override
public void onClick(View v) { Notify("Title:
Meeting with Business", "Msg:Pittsburg 10:00
AM EST ");
}
});
}
@SuppressWarnings("deprecation")
private void Notify(String notificationTitle, String notificationMessage) { NotificationManager
notificationManager = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
@SuppressWarnings("deprecation")
Notification notification = newNotification(R.drawable.ic_launcher, "New
Message", System.currentTimeMillis());
Intent notificationIntent = new Intent(this, MainActivity.class); PendingIntent
pendingIntent = PendingIntent.getActivity(this, 0, notificationIntent, 0);
notification.setLatestEventInfo(MainActivity.this, notificationTitle,
notificationMessage, pendingIntent); notificationManager.notify(9999,
notification);
}
}
```

4. Run the app by rightclick on project package ,choose RunAs andchoose Android Applcation.

5. The output will be run in android emulator.

Output:





Result:

Thus an android application has been developed to notify once message received.

Ex.No.10 Develop an application that makes use of RSS Feeds

Introduction:

RSS stands for Really Simple Syndication. RSS is an easy way to share your website updates and content with your users so that users might not have to visit your site daily for any kind of updates.

Aim:

To develop an application that makes use of RSS Feeds.

Procedure:

2. Create a new project in Eclipse by navigating to **File ⇒ New ⇒Android Project** and fill all the required details.

3. Open **activity_main.xml** by navigating to **res ⇒layout⇒activity_main.xml** and type the following code

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_p
  arent"
  android:orientation="vertical"
  >

  <ListView
    android:id="@+id/listMainView"
    android:layout_width="fill_parent
    "
    android:layout_height="wrap_con
    tent" >
  </ListView>

    </LinearLayout>
```

4. Create a new java file with the name of **ListListener.java** and type the following code

```
package com.example.richsitesummary; import
java.util.List;
import com.example.richsitesummary.RssItem; import
android.app.Activity;
import android.content.Intent; import
android.net.Uri; import
android.view.View;
import android.widget.AdapterView;
```

```
import android.widget.AdapterView.OnItemClickListener;

public class ListListener implements OnItemClickListener
{
List<RssItem> listItems;
```

```
Activity activity;
public ListListener(List<RssItem> listItems,Activity activity)
{
this.listItems=listItems;
this.activity=activity;
}
public void onItemClick(AdapterView<?> parent,View view,int pos,long id)
{
Intent i=new Intent(Intent.ACTION_VIEW);
i.setData(Uri.parse(listItems.get(pos).getLink()));
activity.startActivity(i);
}
}
```

5. Create another java file under src folder with the name of **RSSItem.java** and type the following code

```
package com.example.richsitesummary; public

class RssItem
{
private String title; private
String link; public String
getTitle() { return title;
}
public void setTitle(String title) { this.title
= title;
}
public String getLink() { return
link;
}
public void setLink(String link) { this.link
= link;
}
}
```

6. Create another java file under src folder with the name of **RSSParseHandler .java** and type the following code

```
package com.example.richsitesummary; import
java.util.ArrayList;
import java.util.List;
```

```
import org.xml.sax.Attributes; import
org.xml.sax.SAXException;
import org.xml.sax.helpers.DefaultHandler;
public class RssParseHandler extends DefaultHandler
{
private List<RssItem> rssItems; private
RssItem currentItem; private boolean
parsingTitle; private boolean parsingLink;
public RssParseHandler()
{
rssItems=new ArrayList<RssItem>();
}
public List<RssItem>getItems()
{
return rssItems;
}
@Override
public void startElement(String uri,String localName,String qName,Attributes attributes)throws SAXException
{
//ToDo autogenerated method stub super.startElement(uri,
localName, qName, attributes); if("content-item".equals(qName))
{
currentItem=new RssItem();
}
else if("title".equals(qName))
{
parsingTitle=true;
}
else if("url".equals(qName))
{
parsingLink=true;
}
}
public void endElement(String uri,String localName,String qName)throws SAXException
{
//ToDo autogenerated method stub
super.endElement(uri, localName, qName);
if("content-item".equals(qName))
{
```

```
rssItems.add(currentItem);
currentItem=null;
}
else if("title".equals(qName))
{
parsingTitle=false;
}
else if("title".equals(qName))
{
parsingLink=false;
}
}
@Override
public void characters(char ch[],int Start,int length)throws SAXException
{
super.characters(ch, Start, length);
if(parsingTitle)
{
if(currentItem!=null)
currentItem.setTitle(new String(ch,Start,length));
}
else if(parsingLink)
{
if(currentItem!=null)
{
currentItem.setLink(new String(ch,Start,length));
parsingLink=false;
}
}
}
}
```

7. Create another java file under src folder with the name of **RSSReader .java** and type the following code

```
package com.example.richsitesummary; import
java.util.List;
import javax.xml.parsers.SAXParser;
import javax.xml.parsers.SAXParserFactory; import
com.example.richsitesummary.RssItem; public class
RssReader
{
private String rssUrl;
```

```
public RssReader(String rssUrl)
{
this.rssUrl=rssUrl;

}
public List<RssItem>getItems()throws Exception
{
SAXParserFactory factory=SAXParserFactory.newInstance(); SAXParser
saxParser=factory.newSAXParser(); RssParseHandler handler=new
RssParseHandler(); saxParser.parse(rssUrl, handler);
return handler.getItems();
}
}
```

8. Open **MainActivity.java** by navigating through **src->MainActivity.java** and type the following code.

```
package com.example.richsitesummary; import
android.os.Bundle;
import android.app.Activity; import
android.util.Log; import
android.view.Menu;
import android.widget.ArrayAdapter; import
android.widget.ListView;
import com.example.richsitesummary.RssReader; public
class MainActivity extends Activity { @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
try
{
RssReader rssReader=new
RssReader("http://www.gov.hk/en/about/rss/govhkrss.data.xml"); ListView
Items=(ListView)findViewById(R.id.listMainView); ArrayAdapter<RssItem>
adapter=new
ArrayAdapter<RssItem>(this,android.R.layout.simple_list_item_1,rssReader.ge tItems());
Items.setAdapter(adapter);
Items.setOnItemClickListener(new ListListener(rssReader.getItems(),this));
}catch(Exception e)
{
```

```
Log.e("RssReader",e.getMessage());
}
}
@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present. getMenuInflater().inflate(R.menu.main,
menu);
return true;
}
}
```

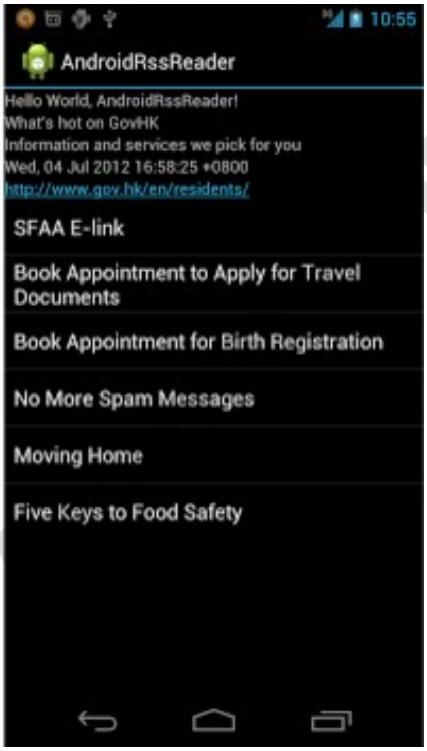
9. Open **AndroidManifest.xml** and add the following line.

```
<uses-permission android:name=”android.permission.INTERNET”/>
```

10. Run the app by rightclick on project package ,choose RunAs and choose Android Applcation.

11. The output will be run in android emulator.

Output:



Result:

Thus an android application to parse RSSFeed hs be developed successfully.