TPCT's

College of Engineering, Osmanabad

Laboratory Manual

SDL-II

Mobile Application Development (Android)

For

Third Year Students (CSE)

Manual Prepared by

Prof. Sujata A. Gaikwad

Author COE, Osmanabad

TPCT's

College of Engineering

Solapur Road, Osmanabad

Department of Computer Science & Engineering

Vision of the Department:

To achieve and evolve as a center of academic excellence and research center in the field of Computer Science and Engineering. To develop computer engineers with necessary analytical ability and human values who can creatively design, implement a wide spectrum of computer systems for welfare of the society.

Mission of the Department:

The department strives to continuously engage in providing the students with in-depth understanding of fundamentals and practical training related to professional skills and their applications through effective Teaching-Learning Process and state of the art laboratories pertaining to CSE and inter disciplinary areas. Preparing students in developing research, design, entrepreneurial skills and employability capabilities.

College of Engineering

Technical Document

This technical document is a series of Laboratory manuals of Computer Science & Engineering Department and is a certified document of College of engineering, Osmanabad. The care has been taken to make the document error-free. But still if any error is found, kindly bring it to the notice of subject teacher and HOD.

Recommended by,

HOD

Approved by,

Principal

FOREWORD

It is my great pleasure to present this laboratory manual for third year engineering students for the subject of SDL-II Mobile Application Development (Android) to understand and visualize the basic concepts of Android. SDL-II Mobile Application Development (Android) covers basic concepts of how to develop Android Applications.

This lab manual provides a platform to the students for understanding the basic concepts of Android. This practical background will help students to gain confidence in creating /developing Android Applications.

H.O.D
CSE Dept

LABORATORY MANUAL CONTENTS

This manual is intended for the Third Year students of Computer Science & Engineering in the subject of SDL-II Mobile Application Development (Android). This manual typically contains practical/ Lab Sessions related to Android in java-eclipse covering various aspects related to the subject for enhanced understanding.

Students are advised to thoroughly go through this manual rather than only topics mentioned in the syllabus as practical aspects are the key to understanding and conceptual visualization of theoretical aspects covered in the books.

SUBJECT INDEX:

Dos and Don’ts in Laboratory :-

1. Maintain Punctuality of time for lab and also for works and assignment completion.
2. Make entry in the Log Book as soon as you enter the Laboratory.
3. All the students should sit according to their roll numbers starting from their left to right.
4. All the students are supposed to enter the terminal number in the log book.
5. Do not change the terminal on which you are working.
6. All the students are expected to get at least the algorithm of the program/concept to be implemented.
7. Strictly follow the instructions given by the teacher/Lab Instructor.
8. Handle Equipments with care.
9. Turn off the machine once you are done using it.
10. Do not Install/Remove any software on system without permission.
11. Do not open any irrelevant Internet Sites on lab computer.
12. Do not plug in external devices without scanning them for computer viruses.

Instructions for Laboratory Teachers:-

1. Submission related to whatever lab work has been completed should be done during the next lab session.
2. The immediate arrangements for printouts related to submission on the day of practical assignments.

3. Students should be taught for taking the printouts under the observation of lab teacher.

4. The promptness of submission should be encouraged by way of marking and evaluation patterns that will benefit the sincere students.

Experiment No.1
Introduction to Android Operating System

Aim:- Introduction to Android Operating System.

Objective: To get the knowledge of android operating system background.

Outcome:- Aware of the android operating system.

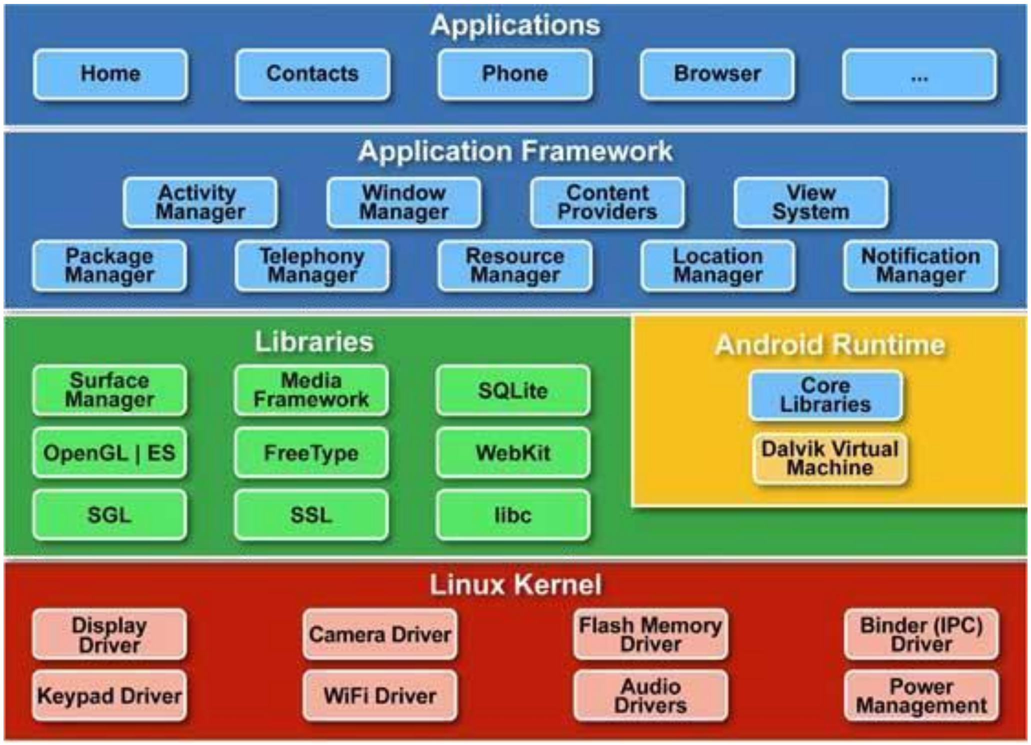
Software:

1. Microsoft Windows (07 or later)
2. The Android SDK (Starter Package)
3. Java Development Kit (JDK) 5 or 6.
4. Eclipse

Theory:

Android Architecture

Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram.



Linux kernel

At the bottom of the layers is Linux - Linux 2.6 with approximately 115 patches.

This provides basic system functionality like process management, memory management, device management like camera, keypad, display etc. Also, the kernel handles all the things that Linux is really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.

Libraries

On top of Linux kernel there is a set of libraries including open -source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc.

Android Runtime

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called **Dalvik Virtual Machine** which is a kind of Java Virtual Machine specially designed and optimized for Android.

The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine.

The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

Application Framework

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.

Applications

You will find all the Android application at the top layer. You will write your application to be installed on this layer only. Examples of such applications are Contacts Books, Browser, and Games etc.

Android UI

An Android application user interface is everything that the user can see and interact with. You have learned about the various layouts that you can use to position your views in an activity. This chapter will give you detail on various views.

A **View** is an object that draws something on the screen that the user can interact with and a **View Group** is an object that holds other View (and View Group) objects in order to define the layout of the user interface.

You define your layout in an XML file which offers a human-readable structure for the layout, similar to HTML. For example, a simple vertical layout with a text view and a button looks like this:

```
<?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_parent"

    android:orientation="vertical" >


    <TextView android:id="@+id/text"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"
        android:text="I am a TextView" />
```

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

      android:layout_height="wrap_content"

      android:text="I am a Button" />


/LinearLayout>
```

Android UI Controls:

There are number of UI controls provided by Android that allow you to build the graphical user interface for your app.

Sr. No.	UI Control & Description
1	TextView This control is used to display text to the user.
2	EditText EditText is a predefined subclass of TextView that includes rich editing capabilities.
3	AutoCompleteTextView The AutoCompleteTextView is a view that is similar to EditText, except that it shows a list of completion suggestions automatically while the user is typing.
4	Button A push-button that can be pressed, or clicked, by the user to perform an action.
5	ImageButton AbsoluteLayout enables you to specify the exact location of its children.
6	CheckBox An on/off switch that can be toggled by the user. You should use checkboxes when presenting users with a group of selectable options that are not mutually exclusive.
7	ToggleButton An on/off button with a light indicator.
8	RadioButton The RadioButton has two states: either checked or unchecked.
9	RadioGroup A RadioGroup is used to group together
10	ProgressBar The ProgressBar view provides visual feedback about some ongoing tasks, such as when u are performing a task in the background.
11	Spinner A drop-down list that allows users to select one value from a set.

12	TimePicker The TimePicker view enables users to select a time of the day, in either 24-hour mode or AM/PM mode.
13	DatePicker The DatePicker view enables users to select a date of the day.

Create UI Controls

As explained in previous chapter, a view object may have a unique ID assigned to it which will identify the View uniquely within the tree. The syntax for an ID, inside an XML tag is:

```
android:id="@+id/text_id"
```

To create a UI Control/View/Widget you will have to define a view/widget in the layout file and assign it a unique ID as follows:

```
<?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_parent"
   android:orientation="vertical" >


   <TextView android:id="@+id/text_id"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="I am a TextView" />
```

```
</LinearLayout>
```

Then finally create an instance of the Control object and capture it from the layout, use the following:

```
TextView myText = (TextView) findViewById(R.id.text_id);
```

Android Event Handling

Events are a useful way to collect data about a user's interaction with interactive components of your app, like button presses or screen touch etc. The Android framework maintains an event queue into which events are placed as they occur and then each event is removed from the queue on a first-in, first-out (FIFO) basis. You can capture these events in your program and take appropriate action as per requirements.

There are following three concepts related to Android Event Management:

- **Event Listeners:** The **View** class is mainly involved in building up a Android GUI, same View class provides a number of Event Listeners. The Event Listener is the object that receives notification when an event happens.

- **Event Listeners Registration:** Event Registration is the process by which an Event Handler gets registered with an Event Listener so that the handler is called when the Event Listener fires the event.
- **Event Handlers:** When an event happens and we have registered and event listener for the event, the event listener calls the Event Handlers, which is the method that actually handles the event.

Event Listeners & Event Handlers

Event Handler	Event Listener & Description
onClick()	onClickListener() This is called when the user either clicks or touches or focuses upon any widget like button, text, image etc. You will use onClick() event handler to handle such event.
onLongClick()	onLongClickListener() This is called when the user either clicks or touches or focuses upon any widget like button, text, image etc. for one or more seconds. You will use onLongClick() event handler to handle such event.
onFocusChange()	onFocusChangeListener() This is called when the widget loses its focus i.e. user go away from the view item. You will use onFocusChange() event handler to handle such event.
onKey()	This is called when the user is focused on the item and presses or releases a hardware key on the device. You will use onKey() event handler to handle such event.
onTouch()	onTouchListener() This is called when the user presses the key, releases the key, or any movement gesture on the screen. You will use onTouch() event handler to handle such event.
onMenuItemClick()	onMenuItemClickListener() This is called when the user selects a menu item. You will use onMenuItemClick() event handler to handle such event.

There are many more event listeners available as a part of **View** class like OnHoverListener, OnDragListener etc which may be needed for your application. So I recommend to refer official documentation for Android application development in case you are going to develop a sophisticated apps.

Result: - Students should be aware of the android operating system.

Conclusion: Thus, students understood the History, Versions, Architecture, IDE (Eclipse), SDK of android.

Experiment No.2
Program for First Android Application

Aim:- Program for First Android Application.

Objective: To create a simple Android Application using Eclipse IDE.

Outcome:- Student will demonstrate the basic application using android.

Software:

1. Microsoft Windows (07 or later)
2. The Android SDK (Starter Package)
3. Java Development Kit (JDK) 5 or 6.
4. Eclipse

Procedure:

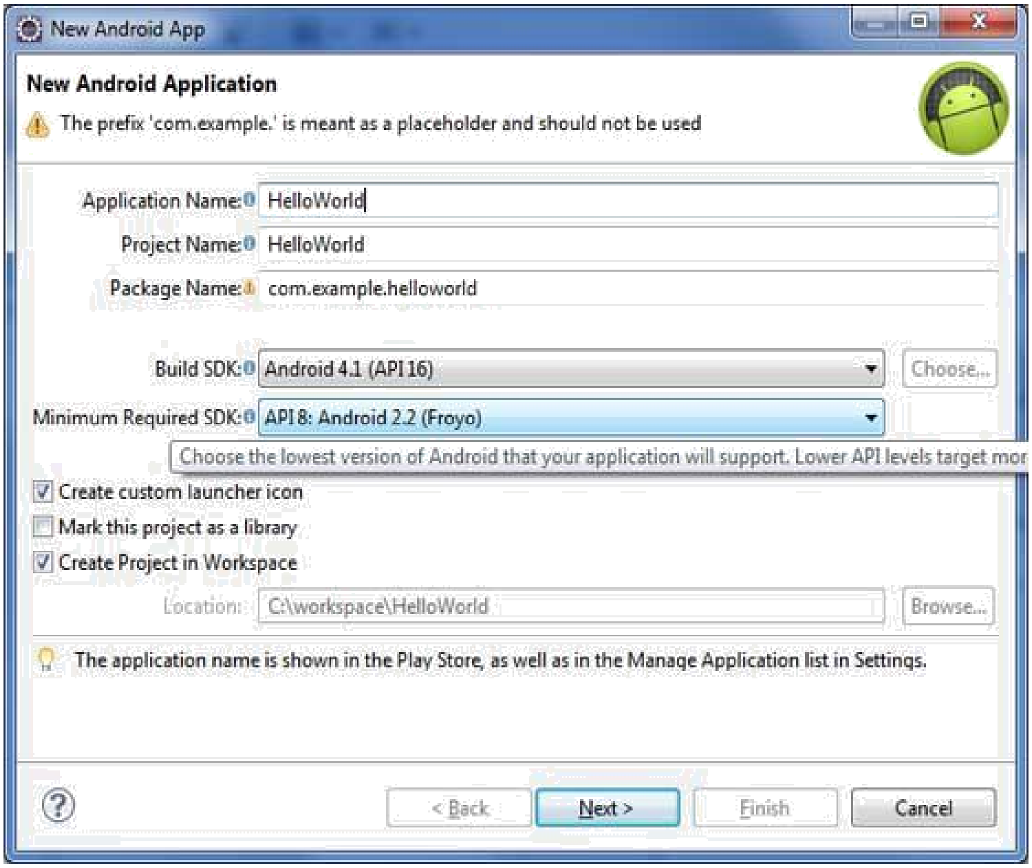
Create Android Application

The first step is to create a simple Android Application using Eclipse IDE. Follow the option

File → new → Project and finally select → Android New Application wizard from the wizard list. Now name your application as HelloWorld using the wizard window as follows:

Next, follow the instructions provided and keep all other entries as default till the final step.

Once your project is created successfully, you will have following project screen:



New Android Application

⚠ The prefix 'com.example.' is meant as a placeholder and should not be used

Application Name: HelloWorld

Project Name: HelloWorld

Package Name: com.example.helloworld

Build SDK: Android 4.1 (API 16)

Minimum Required SDK: API 8: Android 2.2 (Froyo)

Choose the lowest version of Android that your application will support. Lower API levels target mor

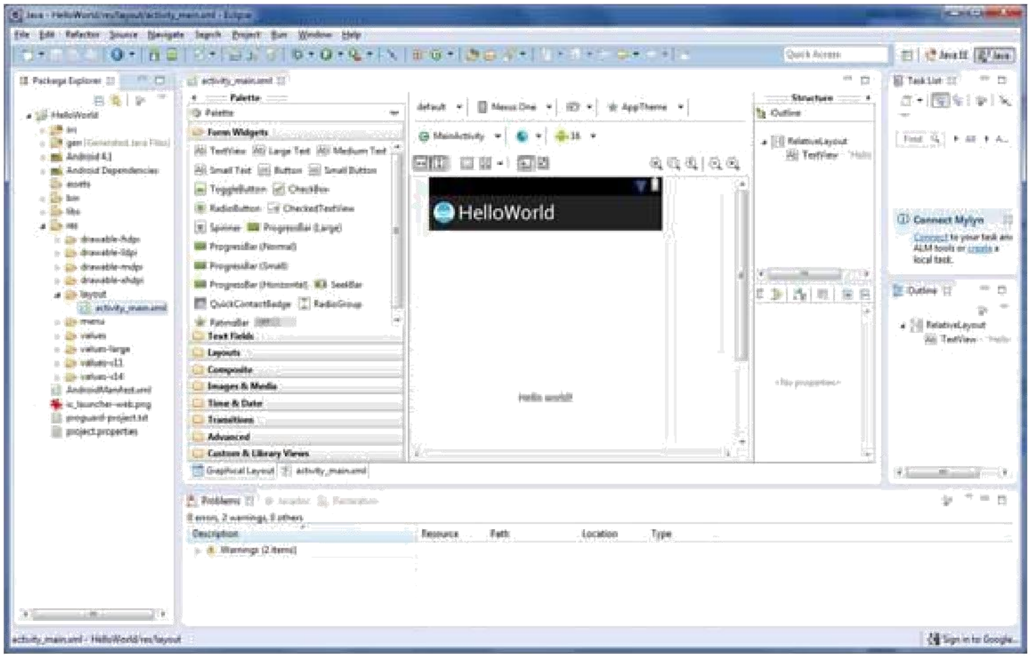
☑ Create custom launcher icon

□ Mark this project as a library

☑ Create Project in Workspace

Location: C:\workspace\HelloWorld

The application name is shown in the Play Store, as well as in the Manage Application list in Settings.



Theory:

Following section will give a brief overview few of the important application files.

The Main Activity File

The main activity code is a Java file MainActivity.java. This is the actual application file which ultimately gets converted to a Dalvik executable and runs your application. Following is the default code generated by the application wizard for *Hello World!* Application:

```
package com.example.helloworld;
import android.os.Bundle; import
android.app.Activity; import
android.view.Menu; import
android.view.MenuItem;
import android.support.v4.app.NavUtils; public
class MainActivity extends Activity {
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
   }
```

```
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu); return
        true;
    }
}
```

Here, *R.layout.activity_main* refers to the *activity_main.xml* file located in the *res/layout* folder. The*onCreate()* method is one of many methods that are fired when an activity is loaded.

The Layout File

The activity_main.xml is a layout file available in *res/layout* directory that is referenced by your application when building its interface. You will modify this file very frequently to change the layout of your application. For your "Hello World!" application, this file will have following content related to default layout:

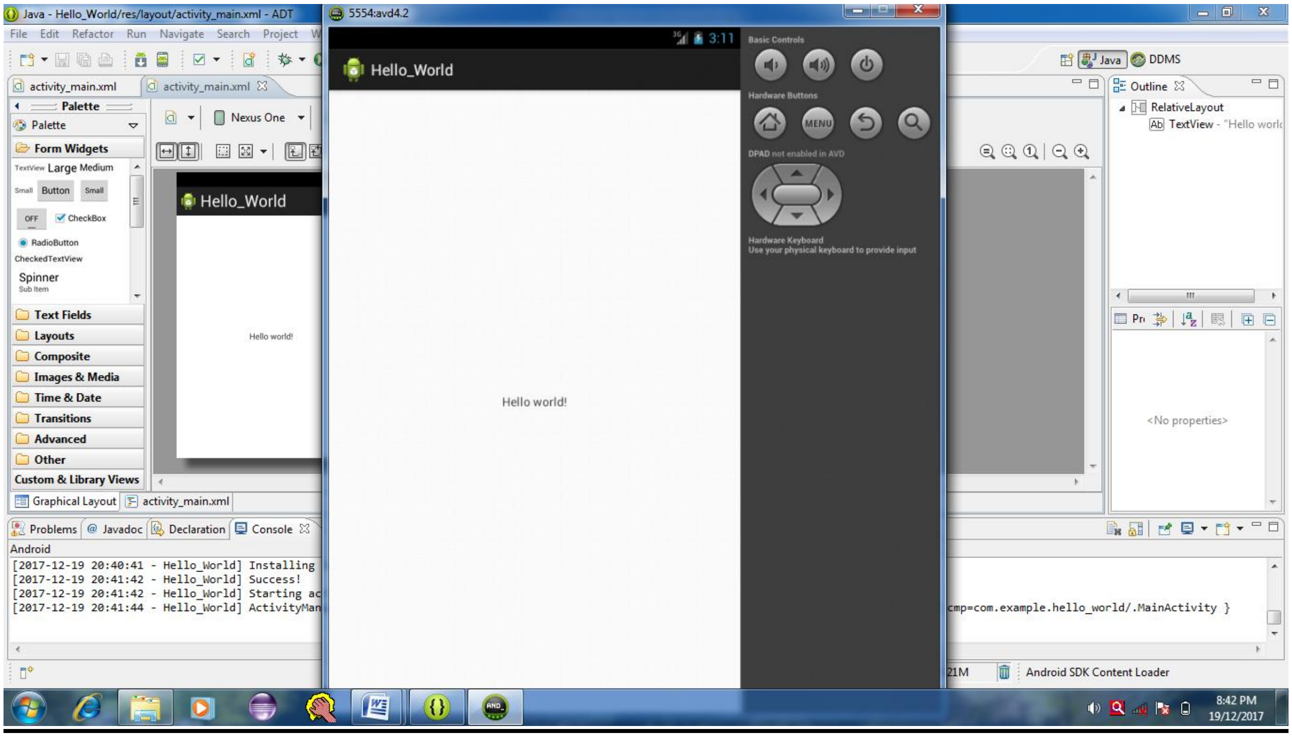
```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent" >

   <TextView android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_centerHorizontal="true"
       android:layout_centerVertical="true"
       android:padding="@dimen/padding_medium"
       android:text="@string/hello_world"
       tools:context=".MainActivity" />

</RelativeLayout>
```

This is an example of simple *RelativeLayout* which we will study in a separate chapter. The *TextView* is an Android control used to build the GUI and it have various attribuites like *android:layout_width*,*android:layout_height* etc which are being used to set its width and height etc. The *@string* refers to the strings.xml file located in the res/values folder. Hence, @string/hello_world refers to the hello string defined in the strings.xml file, which is "Hello World!".

Running the Application

Let's try to run our Hello World! Application we just created. I assume you had created your AVD while doing environment setup. To run the app from Eclipse, open one of your project's activity files and click Run ▶ icon from the toolbar. Eclipse installs the app on your AVD and starts it and if everything is fine with your setup and application, it will display following Emulator window:

Output:



```
Android
[2017-12-19 20:40:41 - Hello_World] Installing
[2017-12-19 20:41:42 - Hello_World] Success!
[2017-12-19 20:41:42 - Hello_World] Starting ac
[2017-12-19 20:41:44 - Hello_World] ActivityMan
cmp=com.example.hello_world/.MainActivity }
```

Result: - Students are aware of the procedure for creating android applications.

Conclusion: Thus, students understood how to create and run android applications.

Experiment No.3

Program for building a simple user interface using a XML for UI layout

Aim:- Program for building a simple user interface using a XML for UI layout

Objective: Student should be able to design their own UI for android application using XML.

Outcome:- Student will demonstrate the basic application using UI in android.

Software:

5. Microsoft Windows (07 or later)
6. The Android SDK (Starter Package)
7. Java Development Kit (JDK) 5 or 6.
8. Eclipse

Code:

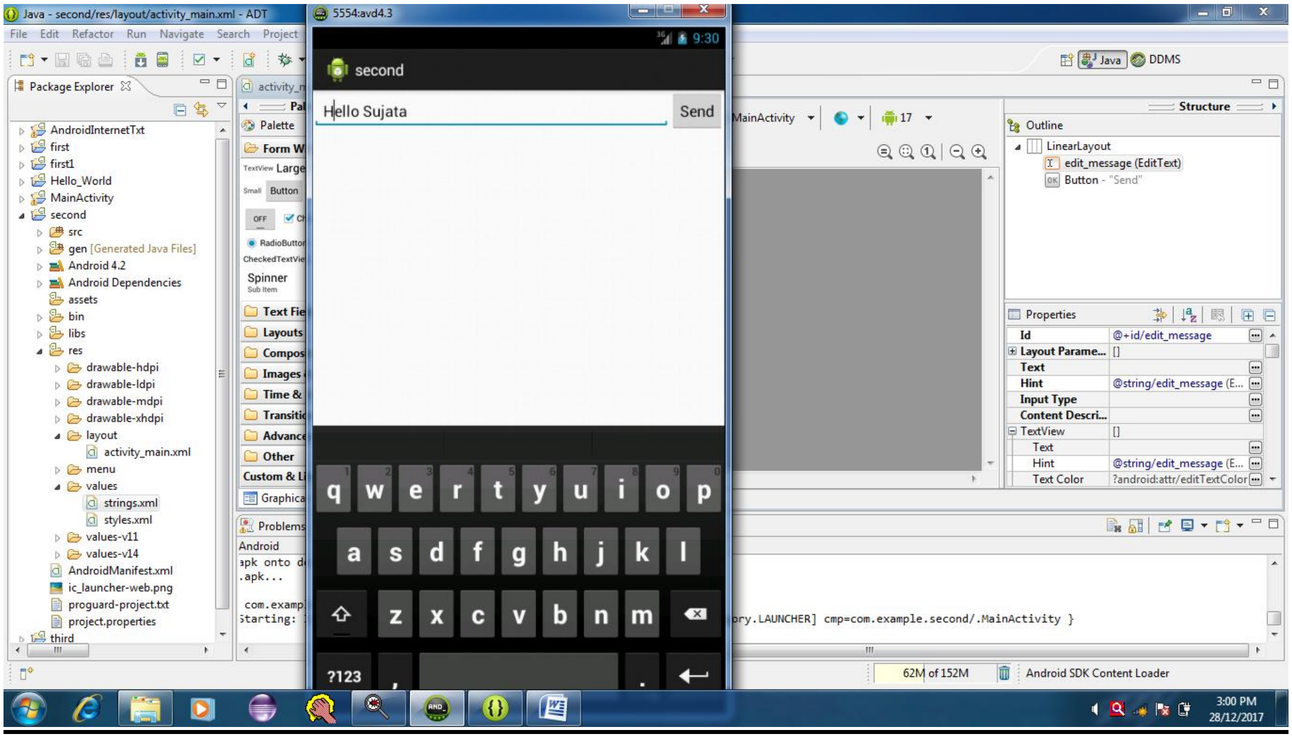
Activity_main.xml:

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

<EditText
   android:id="@+id/edit_message"
   android:layout_weight="1"
   android:layout_width="0dp"
   android:layout_height="wrap_content"
   android:hint="@string/edit_message"  />

<Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/button_send" />

</LinearLayout>
```

Res/values/Strings.xml:

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

    <string name="app_name">second</string>
    <string name="edit_message">Enter a message</string>
    <string name="button_send">Send</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">MainActivity</string>

</resources>
```

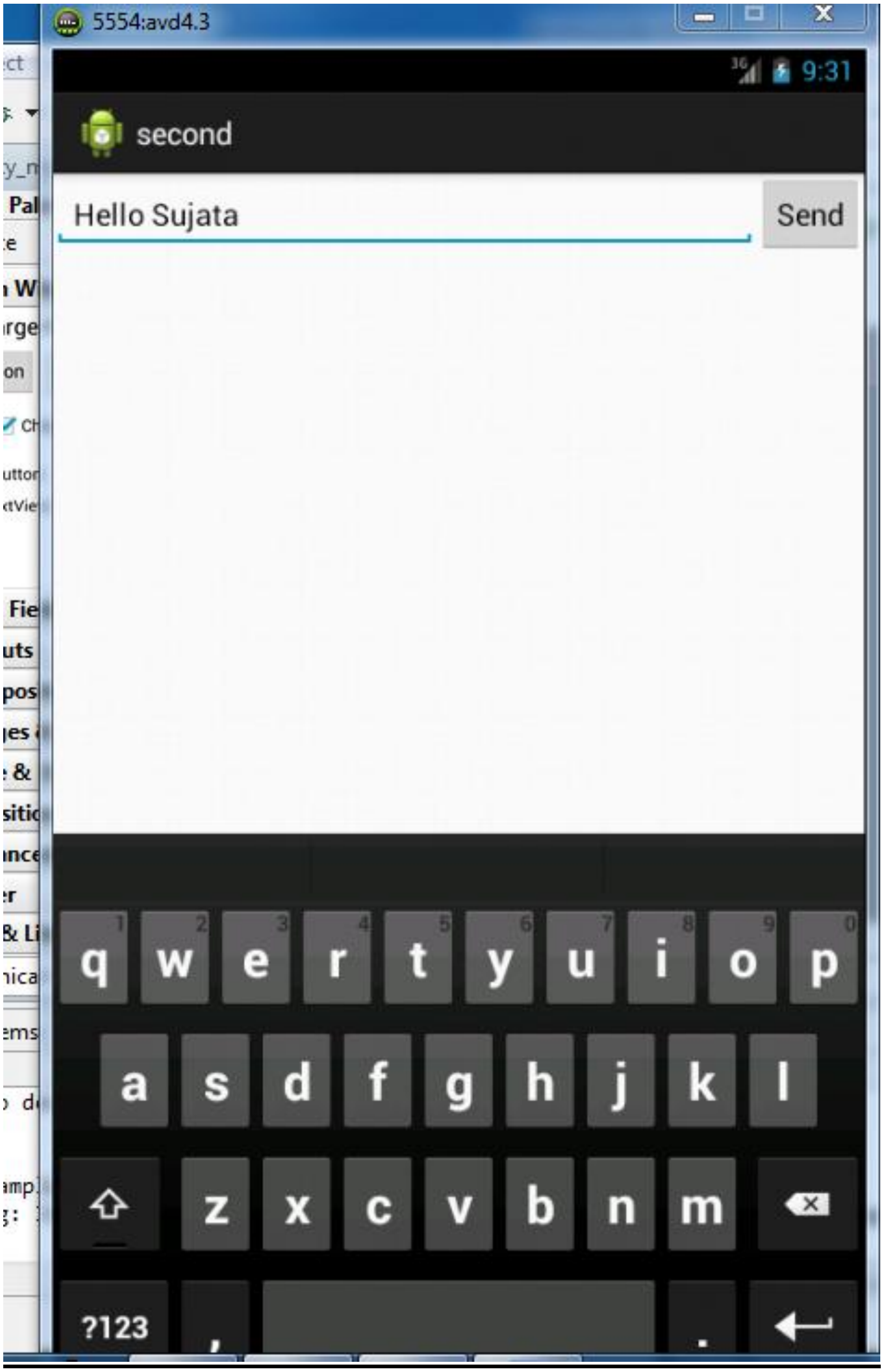
Output:



Hello Sujata

Id	@+id/edit_message
Layout Parame...	[]
Text	
Hint	@string/edit_message (E...
Input Type	
Content Descri...	
TextView	[]
Text	
Hint	@string/edit_message (E...
Text Color	?android:attr/editTextColor

```
Android
apk onto d
.apk...

com.examp
Starting:  ...ry.LAUNCHER] cmp=com.example.second/.MainActivity }
```



Hello Sujata

Result: - Students design their own UI for android application using XML.

Conclusion: Thus, students understood how to demonstrate the basic application using UI in android.

Experiment No.4

Program for developing an Android Application using a linear layout

Aim:- Program for developing an Android Application using a linear layout

Objective: Student should be able to design android application using linear layout.

Outcome: Student will demonstrate the android application using linear layout.

Software:

1. Microsoft Windows (07 or later)
2. The Android SDK (Starter Package)
3. Java Development Kit (JDK) 5 or 6.
4. Eclipse

Theory:

The Linear Layout is one of the simplest layout classes. It allows you to create simple UIs (or UI elements) that align a sequence of child Views in either a vertical or a horizontal line. Linear Layout is a view group that aligns all children in a single direction, vertically or horizontally.

Code:

Activity_main.xml:

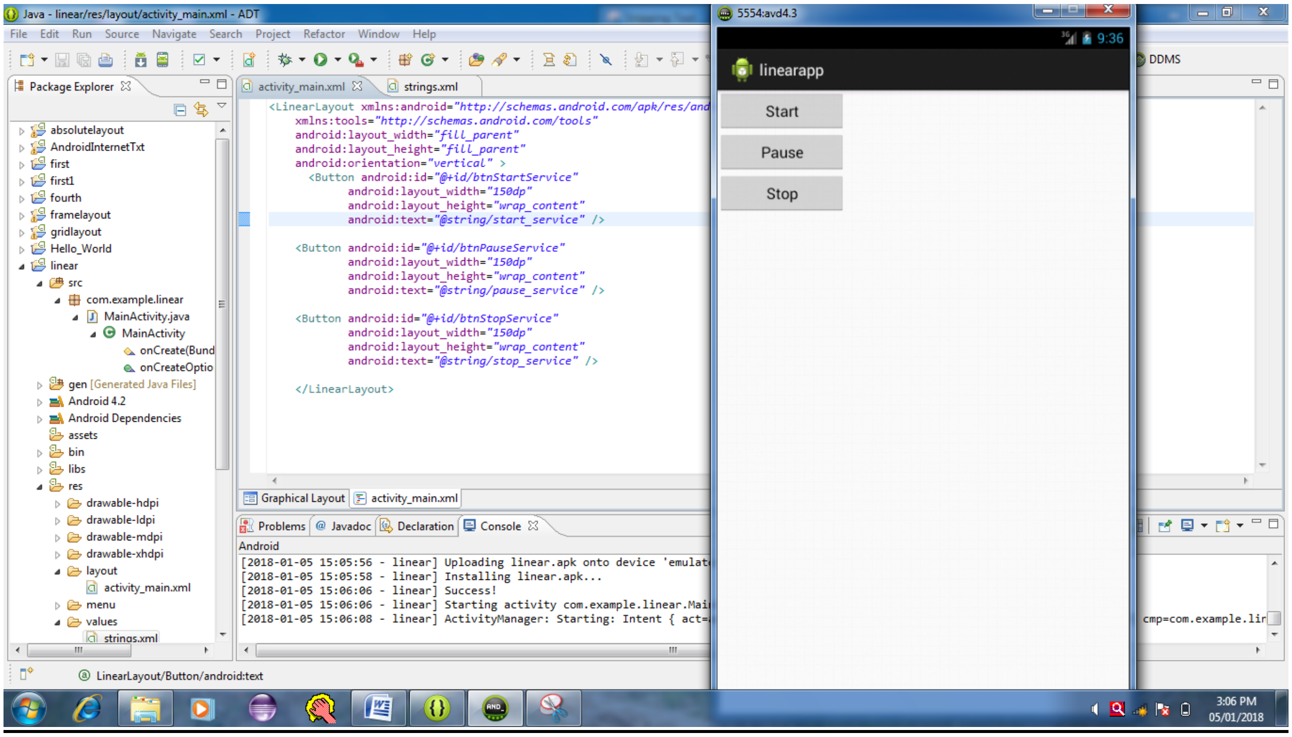
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="vertical" >
    <Button
        android:id="@+id/btnStartService"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:text="@string/start_service" />

   <Button android:id="@+id/btnPauseService"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:text="@string/pause_service" />
```

```
<Button android:id="@+id/btnStopService"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:text="@string/stop_service" />

  </LinearLayout>
```

Res/values/Strings.xml:

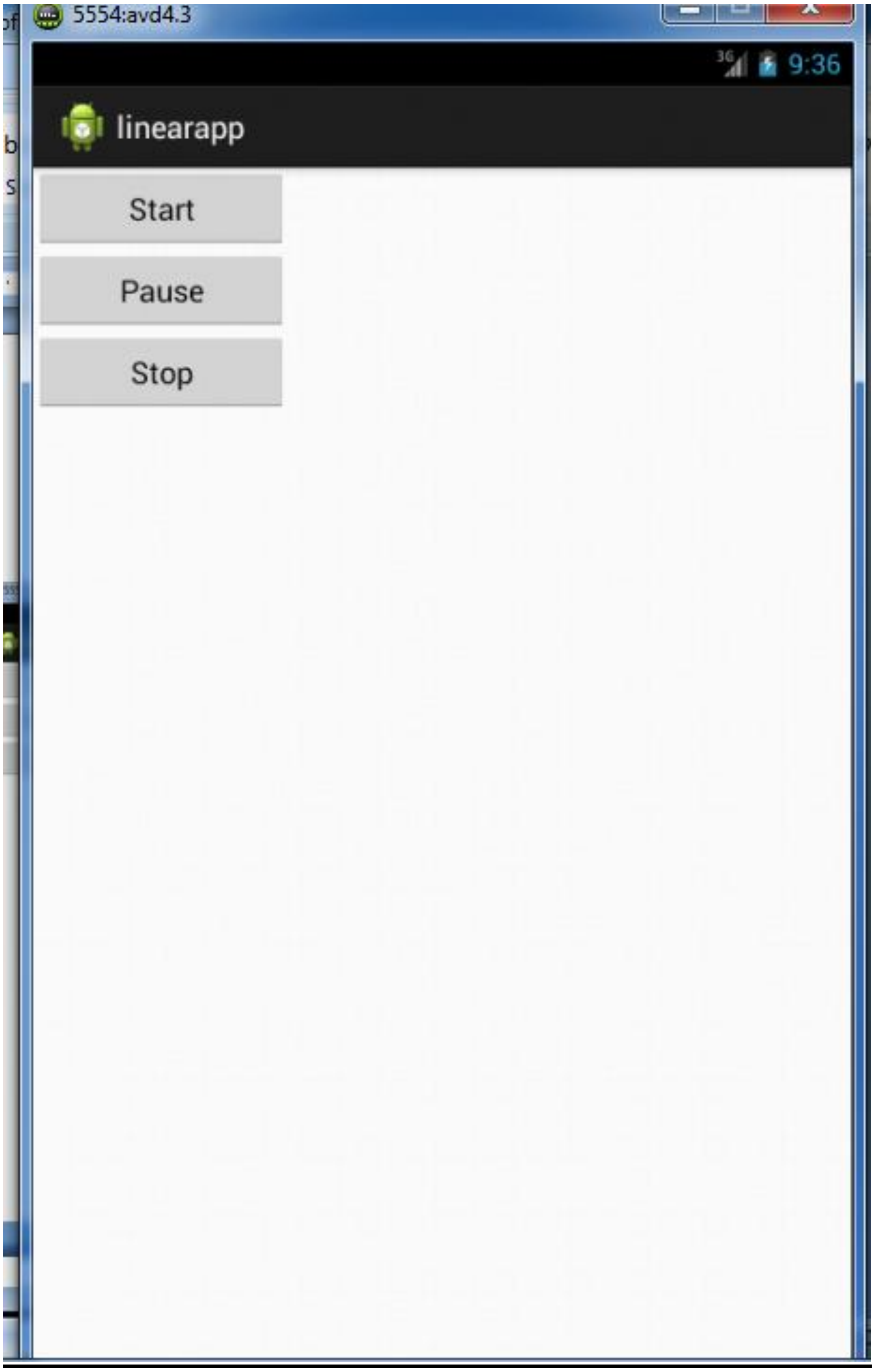
```
<?xml version="1.0" encoding="utf-8"?>
<resources>

  <string name="app_name">linearapp</string>
  <string name="menu_settings">Settings</string>
  <string name="start_service">Start</string>
  <string name="pause_service">Pause</string>
  <string name="stop_service">Stop</string>

</resources>
```

Output:



```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/and
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
      <Button android:id="@+id/btnStartService"
            android:layout_width="150dp"
            android:layout_height="wrap_content"
            android:text="@string/start_service" />

    <Button android:id="@+id/btnPauseService"
            android:layout_width="150dp"
            android:layout_height="wrap_content"
            android:text="@string/pause_service" />

    <Button android:id="@+id/btnStopService"
            android:layout_width="150dp"
            android:layout_height="wrap_content"
            android:text="@string/stop_service" />

    </LinearLayout>
```

```
Android
[2018-01-05 15:05:56 - linear] Uploading linear.apk onto device 'emulat
[2018-01-05 15:05:58 - linear] Installing linear.apk...
[2018-01-05 15:06:06 - linear] Success!
[2018-01-05 15:06:06 - linear] Starting activity com.example.linear.Mai
[2018-01-05 15:06:08 - linear] ActivityManager: Starting: Intent { act=
cmp=com.example.lir
```



Result: **-** Students design android application using linear layout.

Conclusion: Thus, students understood how to demonstrate an android application using linear layout.

Experiment No.5

Program for developing an Android Application using a Relative layout

Aim:- Program for developing an Android Application using a Relative layout.

Objective: Student should be able to design android application using relative layout.

Outcome: Student will demonstrate the android application using relative layout.

Software:

1. Microsoft Windows (07 or later)
2. The Android SDK (Starter Package)
3. Java Development Kit (JDK) 5 or 6.
4. Eclipse

Theory:

The Relative Layout provides a great deal of flexibility for your layouts, allowing you to define the position of each element within the layout in terms of its parent and other views. Relative Layout is a view group that displays child views in relative positions.

Code:

Activity_main.xml:

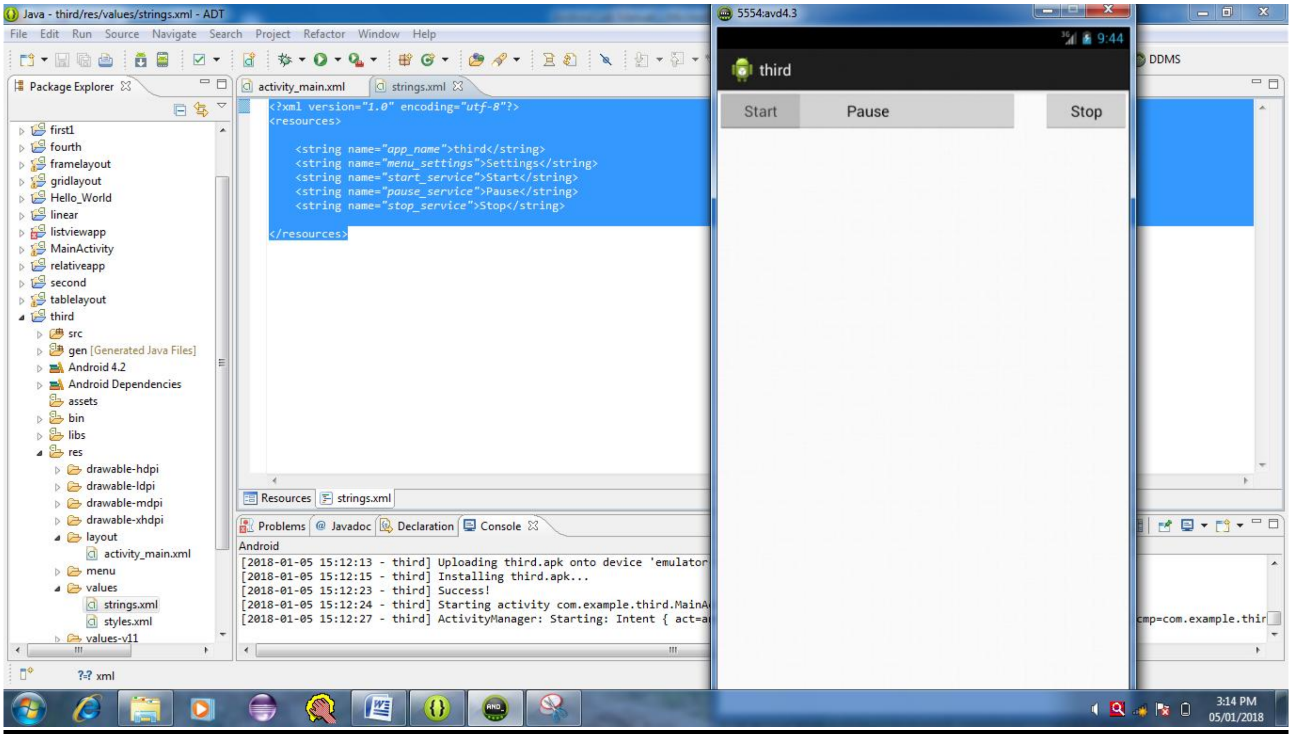
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:orientation="vertical">

<Button
   android:id="@+id/btnStartService"
   android:layout_width="100dp"
   android:layout_height="wrap_content"
   android:text="@string/start_service"/>
<Button
   android:id="@+id/btnPauseService"
   android:layout_width="350dp"
   android:layout_height="wrap_content"
   android:text="@string/pause_service"
   />
```

```
<Button
   android:id="@+id/btnStopService"
   android:layout_width="100dp"
   android:layout_height="wrap_content"
   android:layout_alignParentRight="true"


 android:layout_alignParentTop="true"
   android:text="@string/stop_service"   />
</RelativeLayout>
```

Res/values/Strings.xml:

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

   <string name="app_name">third</string>
   <string name="menu_settings">Settings</string>
   <string name="start_service">Start</string>
   <string name="pause_service">Pause</string>
   <string name="stop_service">Stop</string>

</resources>
```

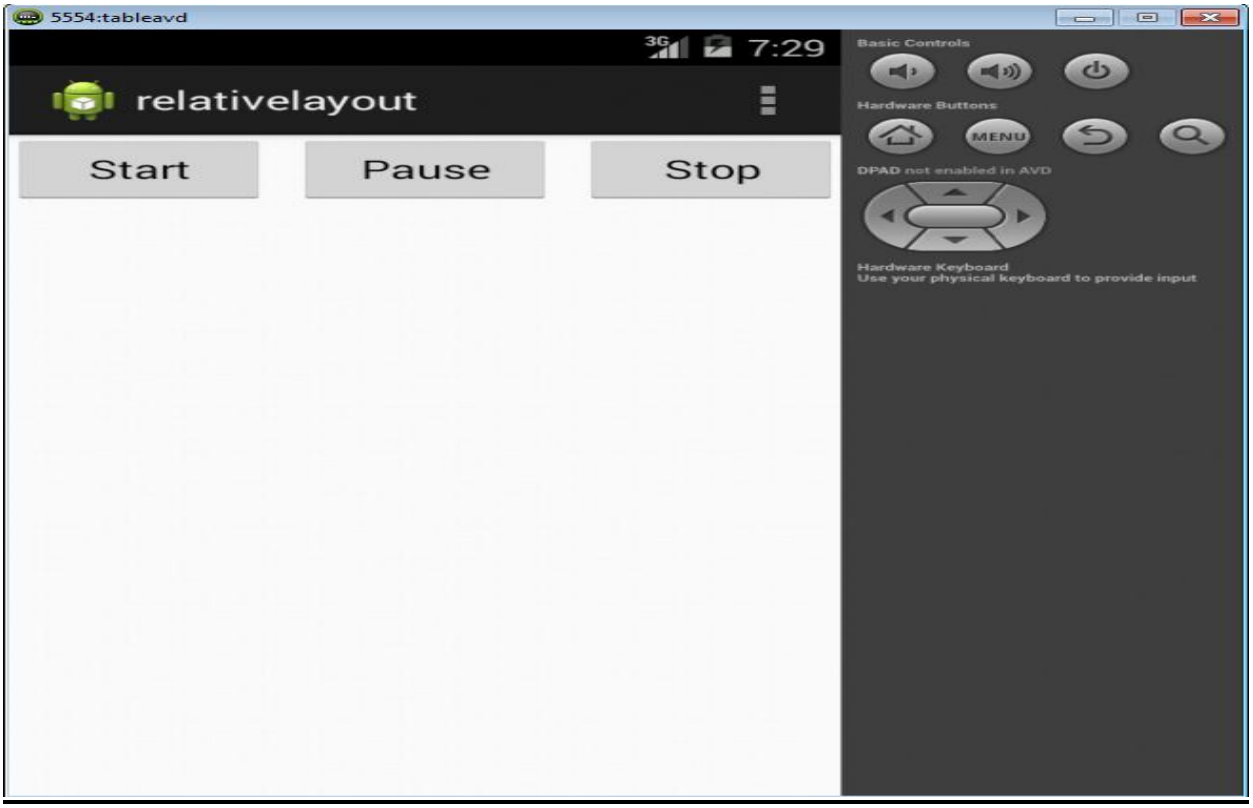
Output:



Java - third/res/values/strings.xml - ADT

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

    <string name="app_name">third</string>
    <string name="menu_settings">Settings</string>
    <string name="start_service">Start</string>
    <string name="pause_service">Pause</string>
    <string name="stop_service">Stop</string>

</resources>
```

```
Android
[2018-01-05 15:12:13 - third] Uploading third.apk onto device 'emulator
[2018-01-05 15:12:15 - third] Installing third.apk...
[2018-01-05 15:12:23 - third] Success!
[2018-01-05 15:12:24 - third] Starting activity com.example.third.MainA
[2018-01-05 15:12:27 - third] ActivityManager: Starting: Intent { act=a
```



Result: - Students design an android application using relative layout.

Conclusion: Thus, students understood how to demonstrate an android application using relative layout.

Experiment No.6

Program for developing an Android Application using a Table layout

Aim:- Program for developing an Android Application using a Table layout.

Objective: Student should be able to design android application using Table layout.

Outcome: Student will demonstrate the android application using Table layout.

Software:

1. Microsoft Windows (07 or later)
2. The Android SDK (Starter Package)
3. Java Development Kit (JDK) 5 or 6.
4. Eclipse

Theory:

Table Layout is a view that groups views into rows and columns.

Code:

Activity_main.xml:

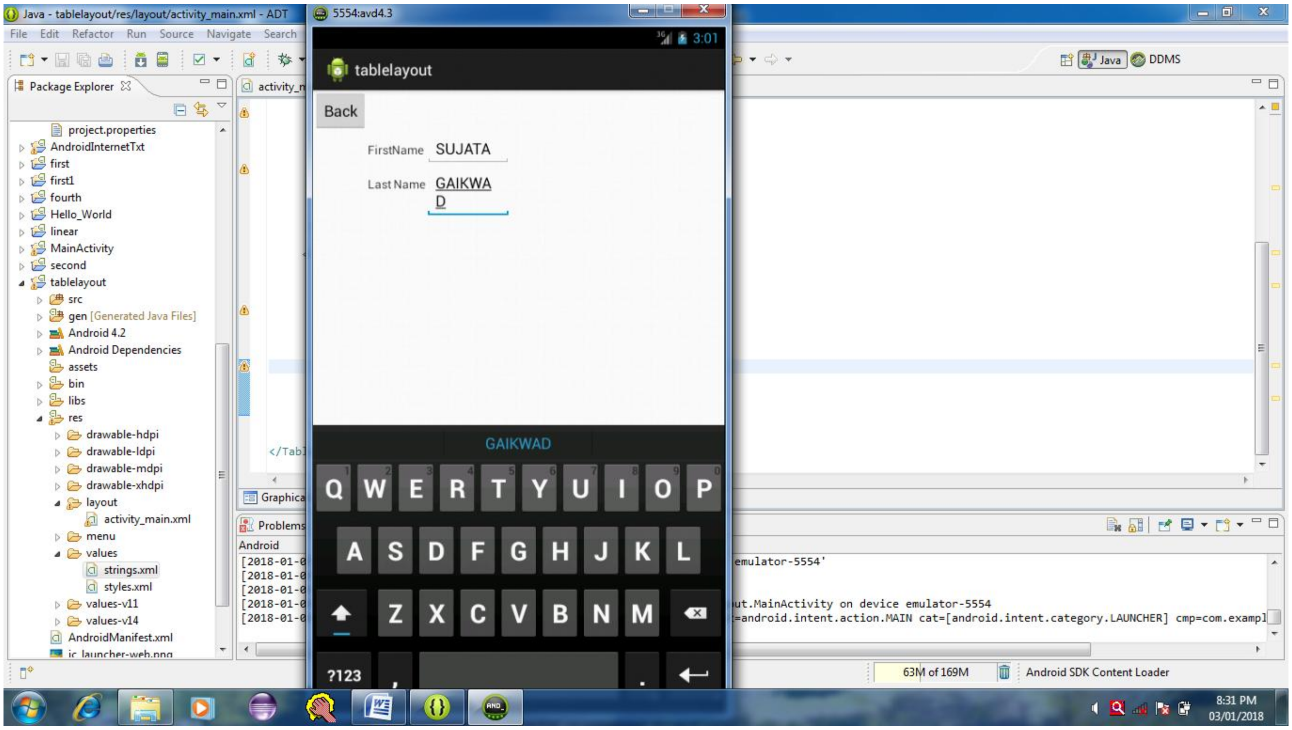
```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent" >
    <TableRow>
      <Button
        android:id="@+id/backbutton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Back" />
</TableRow>


  <TableRow>
     <TextView
```

```
            android:text="FirstName"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_column="1" />

        <EditText
            android:width="100dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"/>
    </TableRow>

    <TableRow>

        <TextView

            android:text="Last Name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_column="1"  />
        <EditText
            android:width="100dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"/>
    </TableRow>

</TableLayout>
```

Res/values/Strings.xml

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

    <string name="app_name">tablelayout</string>
    <string name="menu_settings">Settings</string>

</resources>
```

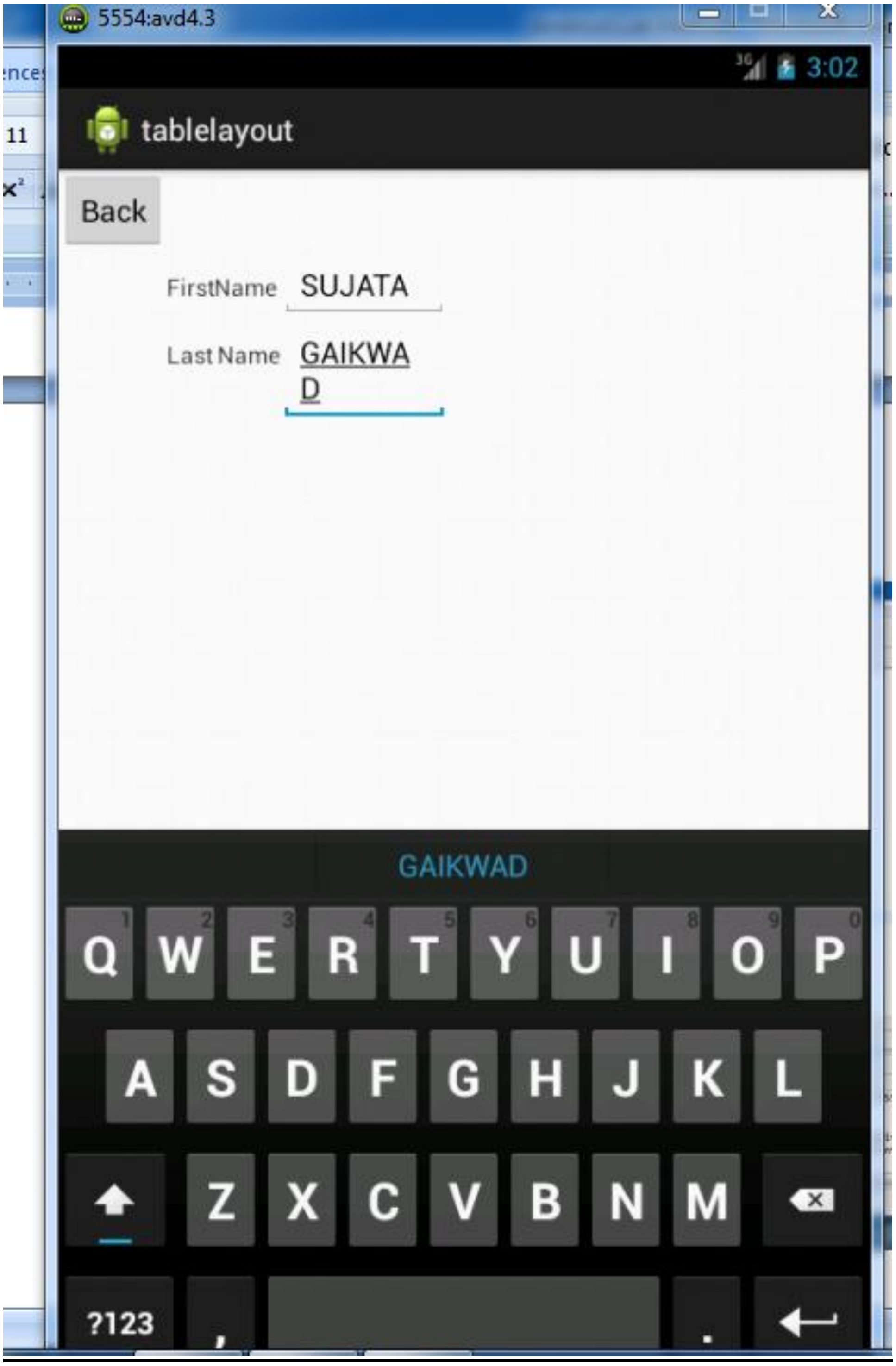
Output:



FirstName SUJATA

Last Name GAIKWAD

```
[2018-01-0                                    emulator-5554'
[2018-01-0
[2018-01-0
[2018-01-0                                    ut.MainActivity on device emulator-5554
[2018-01-0                                    =android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=com.exampl
```



FirstName SUJATA

Last Name GAIKWAD

Result: - Students design an android application using table layout.

Conclusion: Thus, students understood how to demonstrate an android application using table layout.

Experiment No.7

Program for developing an Android Application using Absolute layout

Aim:- Program for developing an Android Application using Absolute layout.

Objective: Student should be able to design android application using absolute layout.

Outcome: Student will demonstrate the android application using absolute layout.

Software:

1. Microsoft Windows (07 or later)
2. The Android SDK (Starter Package)
3. Java Development Kit (JDK) 5 or 6.
4. Eclipse

Theory:

Absolute Layout enables you to specify the exact location of its children.

Code:

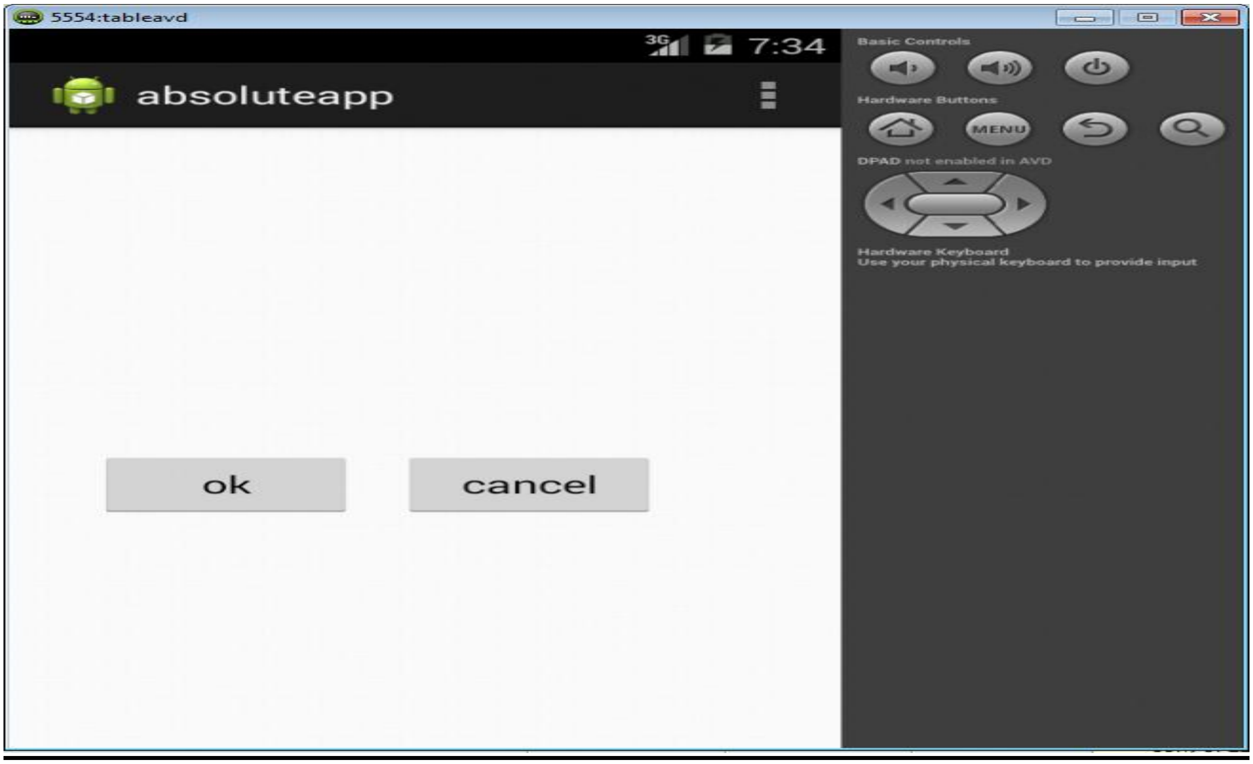
Activity_main.xml:

```
<AbsoluteLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent" >


   <Button
      android:layout_width="100dp"
      android:layout_height="wrap_content"
      android:text="ok"
      android:layout_x="50dp"
      android:layout_y="361dp" />

    <Button
      android:layout_width="100dp"
      android:layout_height="wrap_content"
      android:text="Cancel"
      android:layout_x="225dp"
      android:layout_y="361dp" />

</AbsoluteLayout>
```

Res/values/Strings.xml

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

    <string name="app_name">absolutelayout</string>
    <string name="menu_settings">Settings</string>

</resources>
```

Output:



Result: - Students design an android application using absolute layout.

Conclusion: Thus, students understood how to demonstrate an android application using absolute layout.

Experiment No.8

Program for developing an Android Application using a Frame layout.

Aim:- Program for developing an Android Application using a Frame layout.

Objective: Student should be able to design android application using Frame layout.

Outcome: Student will demonstrate the android application using Frame layout.

Software:

1. Microsoft Windows (07 or later)
2. The Android SDK (Starter Package)
3. Java Development Kit (JDK) 5 or 6.
4. Eclipse

Theory:

The Frame Layout is a placeholder on screen that you can use to display a single view.

Code:

a. MainActivity.Java code

```
package com.example.framelayout;

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

public class MainActivity extends Activity {

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

	}
}
```

b. activity_main.xml code

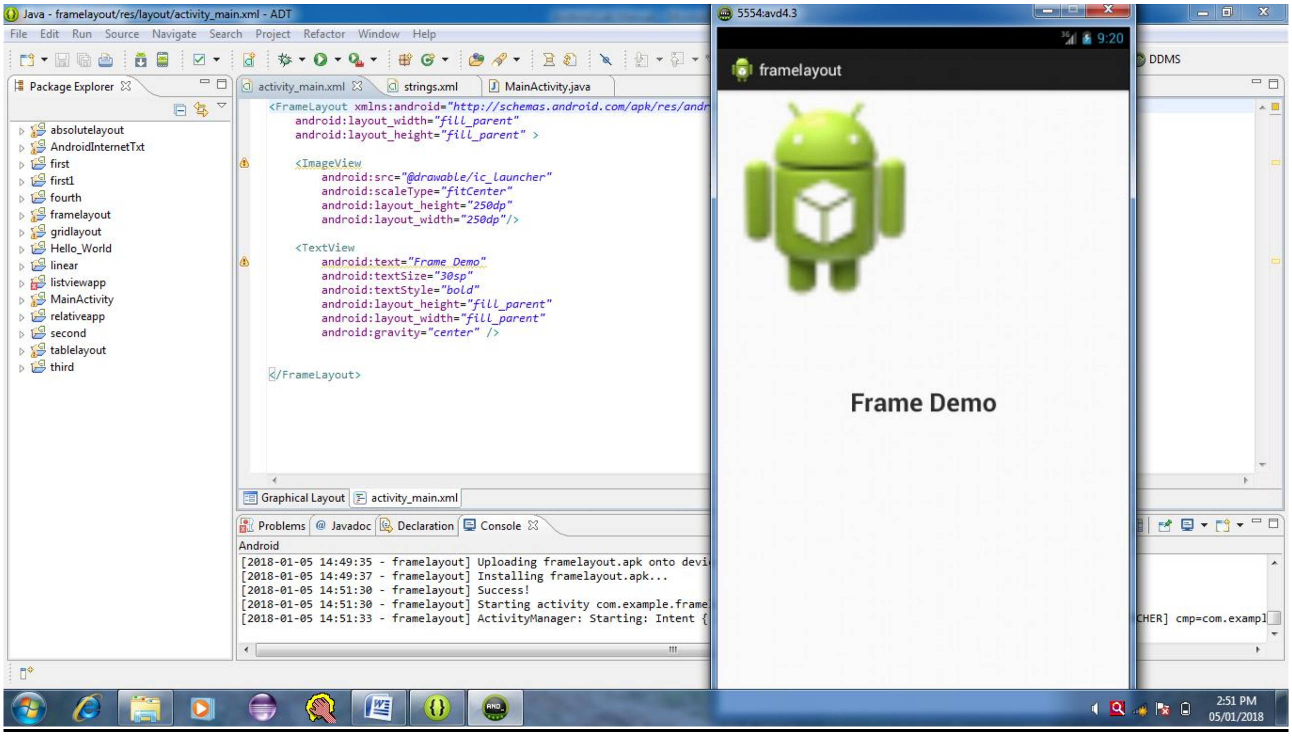
```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent" >
```

```
    <ImageView
      android:src="@drawable/ic_launcher"
      android:scaleType="fitCenter"
      android:layout_height="250dp"
      android:layout_width="250dp"/>

   <TextView
      android:text="Frame Demo"
      android:textSize="30sp"
      android:textStyle="bold"
      android:layout_height="fill_parent"
      android:layout_width="fill_parent"
      android:gravity="center" />


</FrameLayout>
```

Res/values/Strings.xml

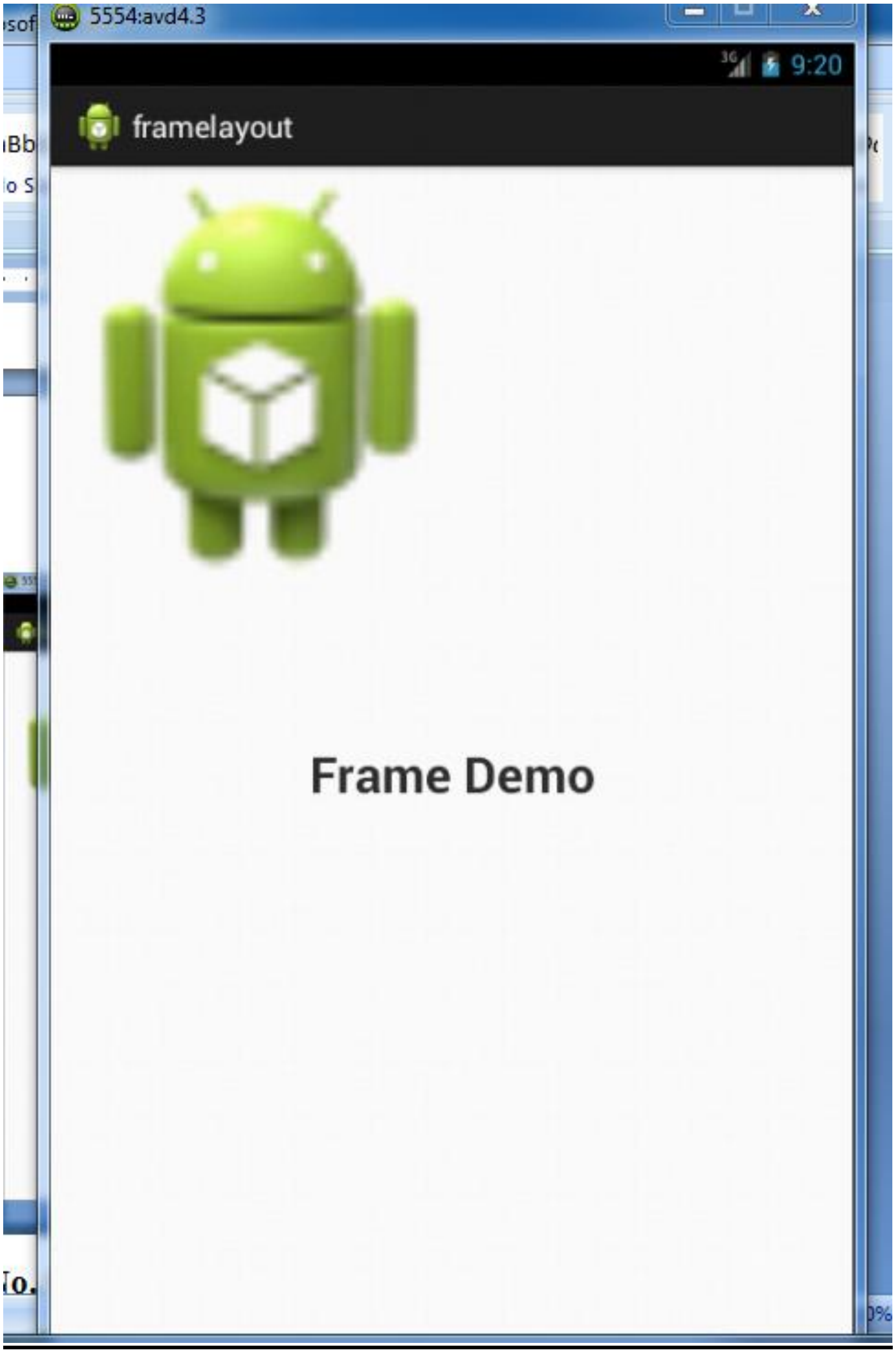
```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">framelayout</string>
    <string name="menu_settings">Settings</string>

</resources>
```

Output:



```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/andr
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <ImageView
        android:src="@drawable/ic_launcher"
        android:scaleType="fitCenter"
        android:layout_height="250dp"
        android:layout_width="250dp"/>

    <TextView
        android:text="Frame Demo"
        android:textSize="30sp"
        android:textStyle="bold"
        android:layout_height="fill_parent"
        android:layout_width="fill_parent"
        android:gravity="center" />


</FrameLayout>
```

```
Android
[2018-01-05 14:49:35 - framelayout] Uploading framelayout.apk onto devi
[2018-01-05 14:49:37 - framelayout] Installing framelayout.apk...
[2018-01-05 14:51:30 - framelayout] Success!
[2018-01-05 14:51:30 - framelayout] Starting activity com.example.frame
[2018-01-05 14:51:33 - framelayout] ActivityManager: Starting: Intent {                CHER] cmp=com.exampl
```



Frame Demo

Result: - Students design an android application using frame layout.

Conclusion: Thus, students understood how to demonstrate an android application using frame layout.

Experiment No.9

Developing an android application using Relative layout to display Date and time.

Aim:- Developing an android application using Relative layout to display Date and time.

Objective: Student should be able to design android application using relative layout to display Date and time.

Outcome: Student will demonstrate the android application using relative layout to display Date and time.

Software:

1. Microsoft Windows (07 or later)
2. The Android SDK (Starter Package)
3. Java Development Kit (JDK) 5 or 6.
4. Eclipse

Theory:

Relative Layout is a view group that displays child views in relative positions.

Code:

1. **MainActivity.Java**

```
package com.example.relativeapp;

import java.text.SimpleDateFormat;
import java.util.Date;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.widget.TextView;


public class MainActivity extends Activity {

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

        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy/MM/dd");
        Date date = new Date();
        String nowDate = dateFormat.format(date);
         TextView dateView = (TextView)findViewById(R.id.dates);
        dateView.setText(nowDate);

            SimpleDateFormat timeFormat= new SimpleDateFormat("HH:mm:ss");
            String nowTime = timeFormat.format(date);

            TextView timeView = (TextView)findViewById(R.id.times);
            timeView.setText(nowTime);

    }

    @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;
    }

}
```

2. **res/layout/activity_main.xml**

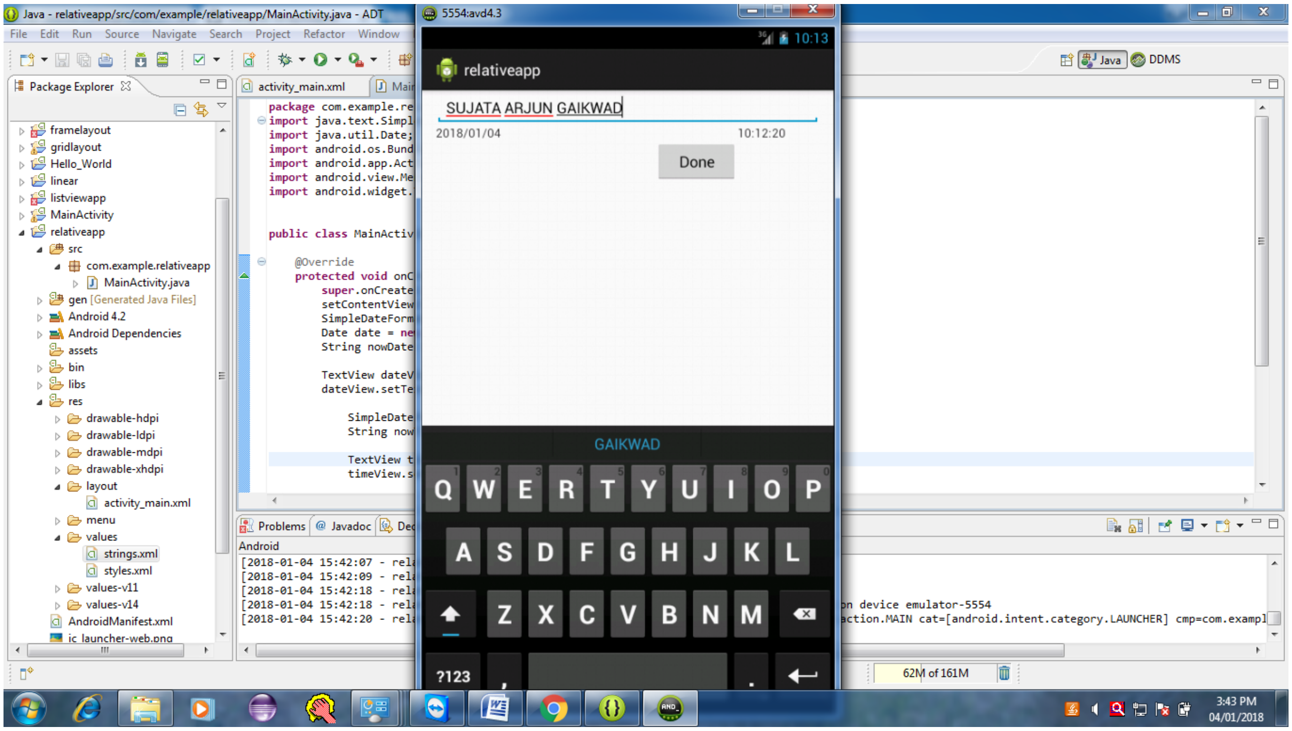
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"

   android:paddingLeft="16dp"
   android:paddingRight="16dp"    >

   <EditText android:id = "@+id/name"
          android:layout_width="fill_parent"
          android:layout_height="wrap_content"
          android:hint = "@string/reminder"   />

   <TextView
      android:id = "@+id/dates"
      android:layout_width="96dp"
      android:layout_height="wrap_content"
      android:layout_below="@id/name"
      android:layout_alignParentLeft="true"
      android:layout_toLeftOf="@+id/times" />


   <TextView android:id="@+id/times"
```

```
            android:layout_width="96dp"
            android:layout_height="wrap_content"
            android:layout_below="@id/name"
            android:layout_alignParentRight="true"  />

    <Button
        android:layout_width="96dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/dates"
        android:layout_toLeftOf="@+id/times"
        android:text="@string/done" />

</RelativeLayout>
```

3. res/values/strings.xml

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

    <string name="app_name">relativeapp</string>
    <string name="action_settings">Settings</string>
    <string name="reminder">Enter your name</string>
    <string name = "done">Done</string>

</resources>
```

Output:



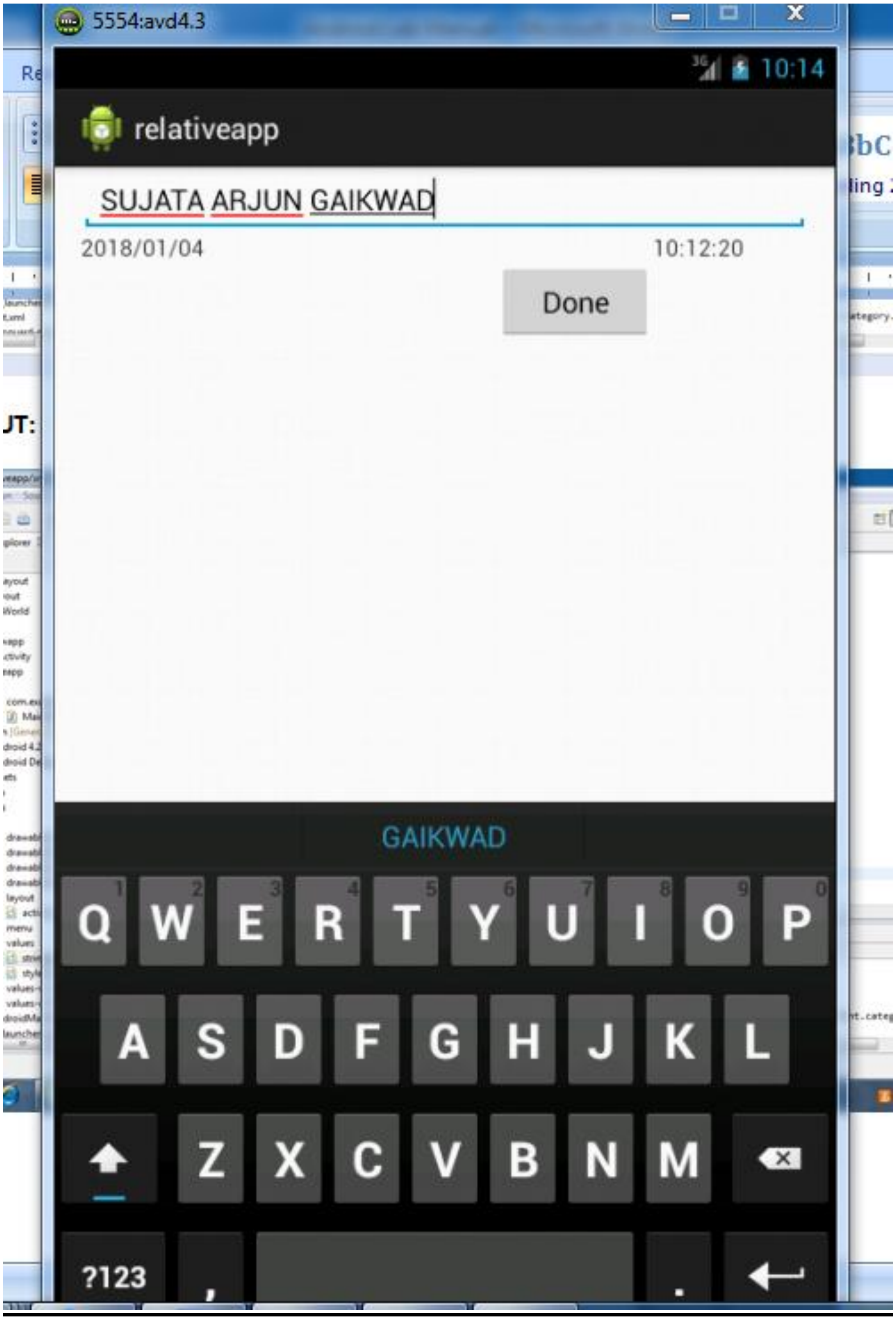
```
package com.example.re
import java.text.Simpl
import java.util.Date;
import android.os.Bund
import android.app.Act
import android.view.Me
import android.widget.

public class MainActiv

    @Override
    protected void onC
        super.onCreate
        setContentView
        SimpleDateForm
        Date date = ne
        String nowDate

        TextView dateV
        dateView.setTe

            SimpleDate
            String now

            TextView t
            timeView.s
```

SUJATA ARJUN GAIKWAD

2018/01/04 10:12:20

Done

```
[2018-01-04 15:42:07 - rel
[2018-01-04 15:42:09 - rel
[2018-01-04 15:42:18 - rel
[2018-01-04 15:42:18 - rel                          on device emulator-5554
[2018-01-04 15:42:20 - rel                          action.MAIN cat=[android.intent.category.LAUNCHER] cmp=com.exampl
```



SUJATA ARJUN GAIKWAD

2018/01/04 10:12:20

Done

Result: - Students design an android application using relative layout to display Date & Time.

Conclusion: Thus, students understood how to demonstrate an android application using relative layout to display Date & Time.

3.Quiz on the subject:-

1) What is Android?

Android is an open-source, Linux-based operating system that is used in mobiles, tablets, televisions etc.

2) Who is the founder of Android?

Andy Rubin.

3) Explain the Android application Architecture.

Following is a list of components of Android application architecture:

- **Services:** Used to perform background functionalities.
- **Intent:** Used to perform the inter connection between activities and the data passing mechanism.
- **Resource Externalization:** strings and graphics.
- **Notification:** light, sound, icon, notification, dialog box and toast.
- **Content Providers:** It will share the data between applications.

4) What are the advantages of android?

Open-source: It means no license, distribution and development fee.

Platform-independent: It supports windows, mac and Linux platforms.

Supports various technologies: It supports camera, Bluetooth, Wi-Fi, speech, EDGE etc. technologies.

Highly optimized Virtual Machine: Android uses highly optimized virtual machine for mobile devices, called DVM (Dalvik Virtual Machine).

5) Does android support other language than java?

Yes, android app can be developed in C/C++ also using android NDK (Native Development Kit). It makes the performance faster. It should be used with android SDK.

6) What are the core building blocks of android?

The core building blocks of android are:

- Activity

- View
- Intent
- Service
- Content Provider
- Fragment etc.

7) What is activity?

Activity is like a frame or window in java that represents GUI. It represents one screen of android.

8) What are the life cycle methods of android activity?

There are 7 life-cycle methods of activity. They are as follows:

1. onCreate()
2. onStart()
3. onResume()
4. onPause()
5. onStop()
6. onRestart()
7. onDestroy()

09) What is intent?

It is a kind of message or information that is passed to the components. It is used to launch an activity, display a web page, send sms, send email etc. There are two types of intents in android:

1. Implicit Intent
2. Explicit Intent

10) What is implicit intent in android?

Implicit intent is used to invoke the system components.

11) What is explicit intent in android?

Explicit intent is used to invoke the activity class.

12) How to call another activity in android?

```
Intent i = new Intent(getApplicationContext( ), ActivityTwo.class);
startActivity(i);
```

13) What is service in android?

A service is a component that runs in the background. It is used to play music, handle network transaction etc.

14) What is the name of database used in android?

SQLite: An open source and lightweight relational database for mobile devices.

15) What is AAPT?

AAPT is an acronym for android asset packaging tool. It handles the packaging process.

16) What is content provider?

Content providers are used to share information between android applications.

17) What is fragment?

Fragment is a part of Activity. By the help of fragments, we can display multiple screens on one activity.

18) What is ADB?

ADB stands for Android Debug Bridge. It is a command line tool that is used to communicate with the emulator instance.

19) What is NDK?

NDK stands for Native Development Kit. By using NDK, you can develop a part of app using native language such as C/C++ to boost the performance.

20) What is ANR?

ANR stands for Application Not Responding. It is a dialog box that appears if the application is no longer responding.

22) What is the Google Android SDK?

The Google Android SDK is a toolset which is used by developers to write apps on Android enabled devices. It contains a graphical interface that emulates an Android driven handheld environment and allow them to test and debug their codes.

23) What is an APK format?

APK is a short form stands for Android Packaging Key. It is a compressed key with classes, UI's, supportive assets and manifest. All files are compressed to a single file is called APK.

24) Which language does Android support to develop an application?

Android applications are written by using the java (Android SDK) and C/C++ (Android NDK).

25) What is ADT in Android?

ADT stands for Android Development Tool. It is used to develop the applications and test the applications.

26) What is View Group in Android?

View Group is a collection of views and other child views. It is an invisible part and the base class for layouts.

27) What is Adapter in Android?

An adapter is used to create a child view to present the parent view items.

28) What is nine-patch images tool in Android?

We can change bitmap images in nine sections as four corners, four edges and an axis.

29) Which kernel is used in Android?

Android is customized Linux 3.6 kernel.

30) What is application Widgets in Android?

Application widgets are miniature application views that can be embedded in other applications and receive periodic updates.

31) Which types of flags are used to run an application on Android?

Following are two types of flags to run an application in Android:

- FLAG_ACTIVITY_NEW_TASK
- FLAG_ACTIVITY_CLEAR_TOP

32) What is singleton class in Android?

A singleton class is a class which can create only an object that can be shared all other classes.

33) What is sleep mode in Android?

In sleep mode, CPU is slept and doesn't accept any commands from android device except Radio interface layer and alarm.

34) What do you mean by a draw able folder in Android?

In Android, a draw able folder is compiled visual resource that can use as a backgrounds, banners, icons, splash screen etc.

35) What is DDMS?

DDMS stands for Dalvik Debug Monitor Server. It gives the wide array of debugging features:

1. Port forwarding services
2. Screen capture
3. Thread and heap information
4. Network traffic tracking
5. Location data spoofing

36) Define Android Architecture?

Android architecture consists of 4 components:

1. Linux Kernel
2. Libraries
3. Android Framework
4. Android Applications

37) What is a portable Wi-Fi hotspot?

The portable Wi-Fi hotspot is used to share internet connection to other wireless devices.

38) Name the dialog box which is supported by android?

- Alert Dialog
- Progress Dialog
- Date Picker Dialog
- Time picker Dialog

39) Name some exceptions in android?

- Inflate Exception
- Surface.OutOfResourceException
- SurfaceHolder.BadSurfaceTypeException
- WindowManager.BadTokenException

40) What are the basic tools used to develop an android app?

- JDK
- Eclipse+ADT plugin
- SDK Tools

4. Conduction of Viva-Voce Examinations:

Teacher should conduct oral exams of the students with full preparation. Normally, the objective questions with guess are to be avoided. To make it meaningful, the questions should be such that depth of the students in the subject is tested. Oral examinations are to be conducted in cordial environment amongst the teachers taking the examination. Teachers taking such examinations should not have ill thoughts about each other and courtesies should be offered to each other in case of difference of opinion, which should be critically suppressed in front of the students. Practical examination should be conducted by internal examiner for three hours under the supervision of external examiner. External examiner should evaluate student by checking practical performance and conducting viva.

5. Evaluation and marking system:

Basic honesty in the evaluation and marking system is absolutely essential and in the process impartial nature of the evaluator is required in the examination system to become. It is a primary responsibility of the teacher to see that right students who are really putting up lot of hard work with right kind of intelligence are correctly awarded.

The marking patterns should be justifiable to the students without any ambiguity and teacher should see that students are faced with just circumstances.

External examiner should evaluate student by checking practical performance and conducting viva.