DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

LAB MANUAL OF
MOBILE COMPUTING LAB
ETIT 452

Maharaja Agrasen Institute of Technology, PSP area,
Sector – 22, Rohini, New Delhi – 110086

(Affiliated to Guru Gobind Singh Indraprastha University,
Dwarka New Delhi)

INDEX OF THE CONTENTS

MAHARAJA AGRASEN INSTITUTE OF TECHNOLOGY

Vision of the Institute

To nurture young minds in a learning environment of high academic value and imbibe spiritual and ethical values with technological and management competence.

Mission of the Institute

The Institute shall endeavor to incorporate the following basic missions in the teaching methodology:

Engineering Hardware – Software Symbiosis

Practical exercises in all Engineering and Management disciplines shall be carried out by Hardware equipment as well as the related software enabling deeper understanding of basic concepts and encouraging inquisitive nature.

Life – Long Learning

The Institute strives to match technological advancements and encourage students to keep updating their knowledge for enhancing their skills and inculcating their habit of continuous learning.

Liberalization and Globalization

The Institute endeavors to enhance technical and management skills of students so that they are intellectually capable and competent professionals with Industrial Aptitude to face the challenges of globalization.

Diversification

The Engineering, Technology and Management disciplines have diverse fields of studies with different attributes. The aim is to create a synergy of the above attributes by encouraging analytical thinking.

Digitization of Learning Processes

The Institute provides seamless opportunities for innovative learning in all Engineering and Management disciplines through digitization of learning processes using analysis, synthesis, simulation, graphics, tutorials and related tools to create a platform for multi-disciplinary approach.

Entrepreneurship

The Institute strives to develop potential Engineers and Managers by enhancing their skills and research capabilities so that they become successful entrepreneurs and responsible citizens.

Vision of the Department:

To Produce "Creators of Innovative Technology"

Mission of the Department:

To provide an **excellent learning environment** across the computer science discipline to inculcate professional behavior, strong ethical values, innovative **research capabilities**and leadership abilities which enable them to become successful **entrepreneurs** in this globalized world.

1. To nurture an **excellent learning environment** that helps students to enhance their problem solving skills and to prepare students to be lifelong learners by offering a solid theoretical foundation with applied computing experiences and educating them about their **professional, and ethical responsibilities**.
2. To establish **Industry-Institute Interaction**, making students ready for the industrial environment and be successful in their professional lives.
3. To promote **research activities** in the emerging areas of technology convergence.
4. To build engineers who can look into technical aspects of an engineering solution thereby setting a ground for producing successful **entrepreneur.**

Program Educational Objective:

PEO1: To train students to have successful careers in computer engineering field or to be able to successfully pursue advanced degrees.

PEO2: To imbibe in students an ability to provide solutions to challenging problems in their profession by applying computer Engineering principles.

PEO3: Train students to communicate effectively, work collaboratively and exhibit high levels of professionalism and ethical responsibility.

PEO4: To motivate graduates to engage in life-long learning and professional development to adapt to rapidly changing work environment

1. Introduction to Mobile Computing Lab

1.1 Objective

- The objective of the Mobile Computing lab course is to introduce CSE students to the field of Mobile Apps development and cellular networks simulations using various open source tools. It would give them hands-on experience through a set of experiments using NS3, Android and XML.

1.2 Course outcomes

Students will be able to get:

452.1) To provide guidelines, design principles and experience in developing applications for small, mobile devices, including an appreciation of context and location aware services

452.2) To develop an appreciation of interaction modalities with small, mobile devices (including interface design for non-standard display surfaces) through the implementation of simple applications using XML, Android etc.

452.3) To introduce wireless communication and networking principles, that support connectivity to cellular networks, wireless internet and other data networks.

452.4) To understand the use of NS3 in simulating various types, topologies of Wireless networks

452.5) To appreciate the social and ethical issues of mobile computing, including privacy.

452.6) Gain an understanding of performance issues in advanced wireless and mobile networks.

2. Lab requirements (details of H/W & S/W to be used)

Hardware Requirements:

Intel i3/C2D Processor/2 GB RAM/500GB HDD/MB/Lan Card/

Key Board/ Mouse/CD Drive/15” Color Monitor/ UPS 24 Nos

Software Requirements: Linux(Fedora/Ubuntu), Android Studio, NS3 3.24 and above

3. LIST OF EXPERIMENTS

(as per GGSIPU Syllabus)

1. Write a WML program to print a formatted Text on the mobile Screen using various tags.
2. Write a WML program to connect multiple cards from same desk.
3. Write WML program to display table with three columns Image name, Image and third column contain hyperlink to open another card.
4. Write a WML program to create a form with multiple options.
5. Write a WML program to use the time control and to trigger On pick event
6. Write a WML script to find maximum out of two numbers with help of inbuilt function Lang.Max() and to find absolute value with help of inbuilt function Lang.abs()
7. Write a Program in NS3 to Simulate OLSR
8. Write a Program in NS3 to Simulate AODV
9. Make an application of your choice using WML or Android.
10. Write a Android Program design an application using Hybrid Approach.

4. LIST OF EXPERIMENTS

(Beyond Curricula)

1. Developing Android App such as E-Commerce based or E learning Based apps
2. Simulate College or office network using NS3.
3. Program to develop a calling application.(Android)
4. Program to develop a mailing application. (Android)
5. Program to display the latitude and longitude of a user. (Android)
6. Program to design multiple activity class. (Android)
7. Program to design a contact application. (Android)

5 FORMAT OF THE LAB RECORD TO BE PREPARED BY THE STUDENTS

The front page of the lab record prepared by the students should have a cover page as displayed below.

MOBILE COMPUTING

ETIT 452

Font should be (Size 20", italics bold, Times New Roman)

Faculty name

Student name

Roll No.:

Semester:

Font should be (12", Times Roman)

Maharaja Agrasen Institute of Technology, PSP Area,

Sector – 22, Rohini, New Delhi – 110086

6. Experiments according to the lab syllabus

Exp. no	Experiment Name	Date of performance	Date of checking	Marks	Signature

7.MARKING SCHEME FOR THE PRACTICAL EXAMS

There will be two practical exams in each semester.

i. Internal Practical Exam
ii. External Practical Exam

INTERNAL PRACTICAL EXAM

It is taken by the respective faculty of the batch.

MARKING SCHEME FOR THIS EXAM IS:

Total Marks: 40

Division of 10 marks per practical is as follows:

	Rubrics for : Laboratory (General)			
Sr No.	**Experiment Component (LAC)**	**Max. Marks**	**Grading Rubrics**	
			2 marks	**1 mark**
1	**Practical Performance**	**2**	**Completeness of practical, exhibits proficiency in using different types of inputs.**	**Incomplete practical, unformatted, lacks comments, Demonstrates no proficiency.**
2	**Output and Validation**	**2**	**Output is free of errors and output is obtained. Demonstrates excellent understanding of the concepts relevant to the experiment.**	**Output contains few logical errors and/or no output is obtained. Demonstrates partial understanding of the concepts relevant to the experiment.**
3	**Attendance and Viva Questions Answered**	**4**	**1. Four marks for answering more than 75% questions.** **2. Two marks for answering more then 50% questions.** **3. One mark for answering less then 50% questions.**	
4	**Timely Submission of Lab Record**	**2**	**On time submission**	**Late submission**

Each experiment will be evaluated out of 10 marks. At the end of the semester average of 8 best performed practical will be considered as marks out of 40.

EXTERNAL PRACTICAL EXAM

It is taken by the concerned lecturer of the batch and by an external examiner. In this exam student needs to perform the experiment allotted at the time of the examination, a sheet will be given to the student in which some details asked by the examiner needs to be written and at the last viva will be taken by the external examiner.

MARKING SCHEME FOR THIS EXAM IS:

Total Marks: 60

Division of 60 marks is as follows

1. Sheet filled by the student: 20
2. Viva Voice: 15
3. Experiment performance: 15
4. File submitted: 10

NOTE:

- Internal marks + External marks = Total marks given to the students
  (40 marks) (60 marks) (100 marks)
- Experiments given to perform can be from any section of the lab.

8. Introduction to WML

WML is a markup language designed especially for specifying and displaying content on WAP (Wireless Application Protocol) devices. WML is part of the WAP application environment, which requires the use of WML.

WML is the wireless equivalent of HTML for the Web. WML is based on XML and derived from xHTML (the XML version of HTML). There are many differences between WML and HTML. For example, WML has a different mechanism for linking between its pages called "cards" as compared to linking between HTML pages. WML browsers are stricter than HTML browsers by not being tolerant of errors. WML browsers enforce the WML requirement of matching closing "tags", an XML characteristic.

WML works with the WAP micro browsers found on WAP devices. This browser is cognizant of the limited capabilities of WAP devices such as

- small displays,
- limited processing power,
- limited memory,
- narrow bandwidth connection,
- limited battery use without recharging

To address the limitations of WAP devices, WML uses the metaphor of card decks, and each page is referred to as a card. The card is the basic unit of navigation and user interface. The user can view only card at a time. WML browsers read the whole deck (complete document) from the server to minimize interaction with the server. Consequently, when flipping (navigating) between the cards in a deck, the browser does not contact the server. This eliminates delays (because each card contains very little text and users are likely to move quickly from one card to another).

A WAP deck is the equivalent of a Web page, the card being the portion of the Web page that can be seen on the screen. Navigation within the cards of a deck is done within the WAP device just as scrolling a Web page is done within the Web device. (without contacting servers in both cases).

An HTML writer does not worry about screen or display boundaries. Instead, the Web browser manages issues relating to the screen boundaries. But a WML writer must be aware of screen boundaries of WAP devices when writing code for cards.

Web server requests are routed through WAP gateways (proxies). A Web server may generate WML content for WAP devices or it may simply dish out HTML (XML).

In case the Web server generates HTML (XML), the WAP gateways must convert the HTML (XML) to WML. Before sending the WML to the WAP device, the gateway compresses it to WMLC (the C in WMLC is for compressed). If the WML generated by the Web sever is WMLC, then the compression step is skipped. Incidentally, converting HTML automatically to WML typically does not produce good or even usable results. For best results, Web servers should generate WML for WAP devices.

Advantages of WML

Like HTML, WML is easy to use. However, compared to HTML, WML has the following advantages in the context of wireless:

- WML is part of the WAP standard and its use is required.
- Transmission of WML (WMLC) documents requires less bandwidth compared to HTML documents because WML documents are simpler and WML is compressed before it is sent to the WAP device.
- Compared to HTML documents, displaying WML documents requires less processing power and memory. Consequently, a WAP device can work with a less powerful (cheaper) CPU and the use of less power means that the battery can operate longer without recharging.
- WML provides support for limited graphics with a limited gray scale.

Disadvantages of WML:

- Like HTML, WML does specify how the content is to be displayed. Thus micro browsers on different WAP devices are likely to display the WML content differently.
- WAP devices such as WAP phones will not accept large decks (1.4K for some WAP phones).
- There are many variations between WAP phones, for example Screen sizes, keypads, and soft keys can be different. Consequently, WML decks should be tested on at least the important WAP devices. This variation is similar to the variation found with Web browsers and their platforms. The problem is harder in case of WML because there are many more WAP devices than Web browsers and their platforms. Also, it is harder to figure out the "least common denominator", i.e., set of features that will work reasonably well on all or most WAP devices.

Different Tags of WML are as follows:

Deck & Card Elements

WML Elements	**Purpose**
<!-->	Defines a WML comment
<wml>	Defines a WML deck (WML root)
<head>	Defines head information
<meta>	Defines meta information
<card>	Defines a card in a deck
<access>	Defines information about the access control of a deck
<template>	Defines a code template for all the cards in a deck

Text Elements

WML Elements	Purpose
** **	Defines a line break
<p>	Defines a paragraph
<table>	Defines a table
<td>	Defines a table cell (table data)
<tr>	Defines a table row
<pre>	Defines preformatted text

Text Formatting Tags

WML Elements	Purpose
<b>	Defines bold text
<big>	Defines big text
****	Defines emphasized text
<i>	Defines italic text
<small>	Defines small text
****	Defines strong text
<u>	Defines underlined text

Image Elements

WML Elements	Purpose
<img>	Defines an image

Anchor Elements

WML Elements	Purpose
<a>	Defines an anchor
<anchor>	Defines an anchor

Event Elements

WML Elements	Purpose
<do>	Defines a do event handler
<onevent>	Defines an onevent event handler
<postfield>	Defines a postfield event handler
<ontimer>	Defines an ontimer event handler
<onenterforward>	Defines an onenterforward handler
<onenterbackward>	Defines an onenterbackward handler
<onpick>	Defines an onpick event handler

Task Elements

WML Elements	Purpose
<go>	Represents the action of switching to a new card
<noop>	Says that nothing should be done
<prev>	Represents the action of going back to the previous card
<refresh>	Refreshes some specified card variables.

Input Elements

WML Elements	Purpose
<input>	Defines an input field
<select>	Defines a select group
<option>	Defines an option in a selectable list
<fieldset>	Defines a set of input fields
<optgroup>	Defines an option group in a selectable list

Variable Elements

WML Elements	**Purpose**
<setvar>	Defines and sets a variable
<timer>	Defines a timer

9. Introduction to NS3

Some of basic steps involved in making programs in NS3 are list below:

Simple client-server communication

1. Create a simple topology of two nodes (Node1, Node2) separated by a point-to-point link.
2. Setup a UdpClient on one Node1 and a UdpServer on Node2. Let it be of a fixed data rate Rate1.
3. Start the client application, and measure end to end throughput whilst varying the latency of the link.
4. Now add another client application to Node1 and a server instance to Node2. What do you need to configure to ensure that there is no conflict?
5. Repeat step 3 with the extra client and server application instances. Show screenshots of pcap traces which indicate that delivery is made to the appropriate server instance.

TCP variants

1. Create a simple dumbbell topology, two client Node1 and Node2 on the left side of the dumbbell and server nodes Node3 and Node4 on the right side of the dumbbell. Let Node5 and Node6 form the bridge of the dumbbell. Use point to point links.
2. Install a TCP socket instance on Node1 that will connect to Node3.
3. Install a UDP socket instance on Node2 that will connect to Node4.
4. Start the TCP application at time 1s.
5. Start the UDP application at time 20s at rate Rate1 such that it clogs half the dumbbell bridge's link capacity.
6. Increase the UDP application's rate at time 30s to rate Rate2 such that it clogs the whole of the dumbbell bridge's capacity.
7. Use the ns-3 tracing mechanism to record changes in congestion window size of the TCP instance over time. Use gnuplot/matplotlib to visualise plots of cwnd vs time.
8. Mark points of fast recovery and slow start in the graphs.
9. Perform the above experiment for TCP variants Tahoe, Reno and New Reno, all of which are available with ns-3.

TCP and router queues

1. Create a simple dumbbell topology, two client Node1 and Node2 on the left side of the dumbbell and server nodes Node3 and Node4 on the right side of the dumbbell. Let Node5 and Node6 form the bridge of the dumbbell. Use point to point links.
2. Add drop tail queues of size QueueSize5 and QueueSize6 to Node5 and Node6, respectively.
3. Install a TCP socket instance on Node1 that will connect to Node3.
4. Install a TCP socket instance on Node2 that will connect to Node3.
5. Install a TCP socket instance on Node2 that will connect to Node4.
6. Start Node1--Node3 flow at time 1s, then measure it's throughput. How long does it take to fill link's entire capacity?
7. Start Node2--Node3 and Node2--Node4 flows at time 15s, measure their throughput.
8. Measure packet loss and cwnd size, and plot graphs throughput/time, cwnd/time and packet loss/time for each of the flows.
9. Plot graph throughput/cwnd and packet loss/cwnd for the first flow. Is there an optimal value for cwnd?
10. Vary QueueSize5 and QueueSize6.

Routing (Optimised Link State Routing)

1. Create a wireless mobile ad-hoc network with three nodes Node1, Node2 and Node3. Install the OLSR routing protocol on these nodes.
2. Place them such that Node1 and Node3 are just out of reach of each other.
3. Create a UDP client on Node1 and the corresponding server on Node3.
4. Schedule Node1 to begin sending packets to Node3 at time 1s.
5. Verify whether Node1 is able to send packets to Node3.
6. Make Node2 move between Node1 and Node3 such that Node2 is visible to both A and C. This should happen at time 20s. Ensure that Node2 stays in that position for another 15s.
7. Verify whether Node1 is able to send packets to Node3.
8. At time 35s, move Node2 out of the region between Node1 and Node3 such that it is out of each other's transmission ranges again.
9. Verify whether Node1 is able to send packets to Node3.
10. To verify whether data transmissions occur in the above scenarios, use either the tracing mechanism or a RecvCallback() for Node3's socket.
11. Plot the number of bytes received versus time at Node3.
12. Show the pcap traces at Node 2's Wifi interface, and indicate the correlation between Node2's packet reception timeline and Node2's mobility.

Wifi RTS/CTS

1. Setup a 5x5 wireless adhoc network with a grid. You may use examples/wireless/wifi-simple-adhoc-grid.cc as a base.
2. Install the OLSR routing protocol.
3. Setup three UDP traffic flows, one along each diagonal and one along the middle (at high rates of transmission).
4. Setup the ns-3 flow monitor for each of these flows.
5. Now schedule each of the flows at times 1s, 1.5s, and 2s.
6. Now using the flow monitor, observe the throughput of each of the UDP flows. Furthermore, use the tracing mechanism to monitor the number of packet collisions/drops at intermediary nodes. Around which nodes are most of the collisions/drops happening?
7. Now repeat the experiment with RTS/CTS enabled on the wifi devices.
8. Show the difference in throughput and packet drops if any.

Wifi Channels

1. Setup a 2-nodes wireless adhoc network. Place the nodes at a fixed distance in a 3d scenario.
2. Install all the relevant network stacks, up to and including UDP.
3. Setup a CBR transmission between the nodes, one acting as a server and one as a client. Take the iperf[1] behaviour as an example.
4. Setup counters and outputs for packets sent and received.
5. Schedule the simulation to run for enough time to obtain statistically relevant results (suggestion: analyze some test results and reduce the simulation time accordingly).
6. Repeat the simulation varying the distance between the nodes from a minimum of 1meter to the point where the nodes can't transmit/receive anymore.
7. Repeat the above varying the channel models and the transmission/receive parameters like node's position above the ground, transmission power, etc.
8. Show the differences between the various channel models, and comment them. Identify the channel model that is more appropriate for each case (indoor, outdoor, LoS, NLoS, etc.).

10. Experiment No 1

AIM: Write a WML program to print a formatted Text on the mobile Screen using various tags

a. Syntax

..text..
..images..
..links..

b. WML Tags and Attributes

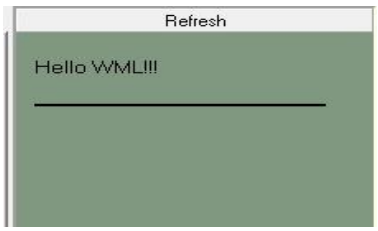
Attribute	Value	Description
align	left right center	Aligns the paragraph. Default is "left"
mode	wrap nowrap	Sets whether a paragraph should wrap lines or not.
xml:lang	*language_code*	Sets the language used in the element
class	*cdata*	Sets a class name for the element. The class name is case sensitive. An element can be connected to multiple classes. Multiple class names within the class attribute are separated by white space
id	*id*	Sets a unique name for the element

<b>	Defines bold text
<big>	Defines big text
	Defines emphasized text
<i>	Defines italic text
<small>	Defines small text
	Defines strong text
<u>	Defines underlined text

10.1 Write a WML program to print a Text on the mobile Screen

10.2 Write a WML program to display your name in bold, italic, small, big and emphasize format.

OUTPUT:



10.3 Short Questions

Q.1 What Is The Use Of Wml Decks?

Q.No2 What is WSDL?

Q.No 3What Are The Wml Variable?how To Use Them?

Q.No 4 How Can We Refresh Card Variables?

11.Experiment No 2

AIM: Write a WML program to connect multiple cards from same desk

Introduction about WML Tags:

The <anchor>...</anchor> tag pair is used to create an anchor link. It is used together with other WML elements called <go/>, <refresh> or <prev/>. These elements are called task elements and tell WAP browsers what to do when a user selects the anchor link

You can enclose Text or image along with a task tag inside <anchor>...</anchor> tag pair.

Attributes:

This element supports the following attributes:

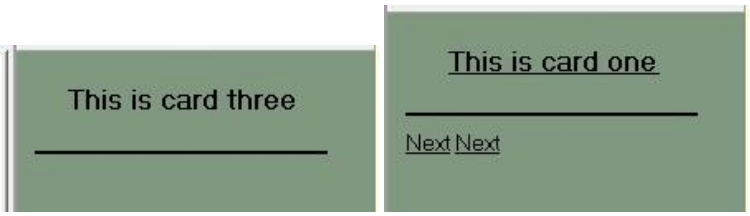
Attribute	Value	Description
title	cdata	Defines a text identifying the link
xml:lang	language_code	Sets the language used in the element
class	class data	Sets a class name for the element.
id	element ID	A unique ID for the element.

Sample Outline of the Program (Syntax)

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">
<wml>
<card title="Anchor Element">
<p>
  <anchor>
    <go href="nextpage.wml"/>
  </anchor>
</p>
<p>
  <anchor>
    <prev/>
  </anchor>
</p>
</card>
</wml>
```

11.2 Write a WML program to declare three cards and provide Hyper Link to move from one card to another card.

Output:



11.3 Short Questions

Q.1 What is the use of XML?

Q.2 What is Meta Data?

Q.3 What is the difference between HTML and WML?

Q.**4** What is XML DOM Document**?**

12. Experiment No.3

AIM: Write WML program to display table with three columns Image name, Image and third column contain hyperlink to open another card

Introduction about WML Tags:

The <table> element along with <tr> and <td> is used to create a table in WML. WML does not allow the nesting of tables

A <table> element should be put with-in ... elements.

Attributes:

The <table /> element supports the following attributes:

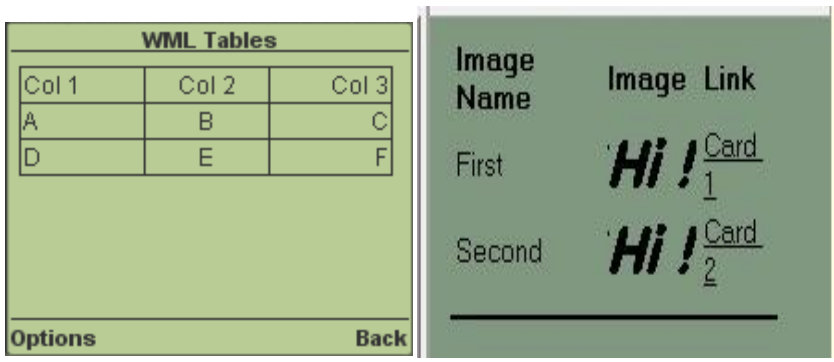
Attribute	Value	Description
columns	number	Sets the number of columns in the table
align	L C R	To specify the horizontal text alignment of the columns, you need to assign three letters to the align attribute. Each letter represents the horizontal text alignment of a column. The letter can be L, C, or R. For example, if you want the following settings to be applied to your table: First table column -- Left-aligned Second table column -- Center-aligned Third table column -- Right-aligned Then you should set the value of the align attribute to LCR.
xml:lang	language_code	Sets the language used in the element
class	class data	Sets a class name for the element.
id	element ID	A unique ID for the element.

Sample Outline of the Program (Syntax)

```
<wml>
<card title="WML Tables">
<p>
<table columns="3" align="LCR">
        <tr>
          <td>Col 1</td>
          <td>Col 2</td>
          <td>Col 3</td>
        </tr>
        <tr>
          <td>A</td>
          <td>B</td>
          <td>C</td>
```

```
        </tr>
        <tr>
          <td>D</td>
          <td>E</td>
          <td>F</td>
        </tr>
</table>
</p>
</card>
</wml>
```

OUTPUT:



12.1: **Write WML program to display user personal information like Id, Name, Address, Phone No and pass same information on another deck**.

Introduction about WML Tags:

The <input/> element is used to create input fields and input fields are used to obtain alphanumeric data from users.

Attributes:

This element supports the following attributes:

Attribute	Value	Description
name	text	The name of the variable that is set with the result of the user's input
maxlength	number	Sets the maximum number of characters the user can enter in the field
emptyok	true false	Sets whether the user can leave the input field blank or not. Default is "false"
format	A a N X x M	Sets the data format for the input field. Default is "*M". A = uppercase alphabetic or punctuation characters a = lowercase alphabetic or punctuation characters

	M *f nf	N = numeric characters X = uppercase characters x = lowercase characters M = all characters m = all characters *f = Any number of characters. Replace the f with one of the letters above to specify what characters the user can enter nf = Replace the n with a number from 1 to 9 to specify the number of characters the user can enter. Replace the f with one of the letters above to specify what characters the user can enter
size	number	Sets the width of the input field
tabindex	number	Sets the tabbing position for the select element
title	text	Sets a title for the list
type	text password	Indicates the type of the input field. The default value is "text". Password field is used to take password for authentication purpose.
value	text	Sets the default value of the variable in the "name" attribute
xml:lang	language_code	Sets the language used in the element
class	class data	Sets a class name for the element.
id	element ID	A unique ID for the element.

OUTPUT:

12.3 Short Questions:

Q.No.1. What is the function of WAP Gateway?

Q.No.2 What is Distillation technique in WAP?

Q.No.3 What is the use of UAProf?

Q.No.4 Why WML is called Light weight Language?

13. Experiment No.4

AIM: Write a WML program to create a form with multiple options

Introduction about WML Tags:

WML <select> Element:

The <select>...</select> WML elements are used to define a selection list and the <option>...</option> tags are used to define an item in a selection list. Items are presented as radiobuttons in some WAP browsers. The <option>...</option> tag pair should be enclosed within the <select>...</select> tags.

Attributes:

Attribute	Value	Description
iname	text	Names the variable that is set with the index result of the selection
ivalue	text	Sets the pre-selected option element
multiple	• true • false	Sets whether multiple items can be selected. Default is "false"
name	text	Names the variable that is set with the result of the selection
tabindex	number	Sets the tabbing position for the select element
title	text	Sets a title for the list
value	text	Sets the default value of the variable in the "name" attribute
xml:lang	language_code	Sets the language used in the element
class	class data	Sets a class name for the element.
id	element ID	A unique ID for the element.

Sample Outline of the Program (Syntax)

```
<wml>
<card title="Selectable List">
<p> Select a Tutorial :
 <select>
  <option value="htm">HTML Tutorial</option>
  <option value="xml">XML Tutorial</option>
  <option value="wap">WAP Tutorial</option>
```

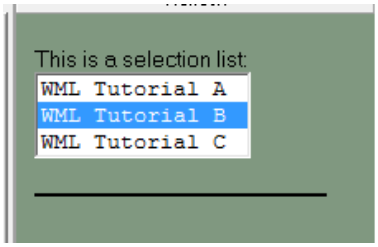
```
 </select>
</p>
</card>

</wml>
```

For Multiple Options:

```
<wml>
<card title="Selectable List">
<p> Select a Tutorial :
 <select multiple="true">
  <option value="htm">WML Tutorial A</option>
  <option value="xml">WML Tutorial  B</option>
  <option value="wap">WML Tutorial C</option>
 </select>
</p>
</card>
</wml>
```

OUTPUT:

Output:



13.4 Short Questions:

Q.No.1. What is Push and Pull technique in WAP?

Q.No.2 List out the Databases used to store Data of WML pages?

Q.No.3 What is the advantages of using XML DOM document?

Q.No.4 What is DTD?

14. Experiment No. 5

AIM: Write a WML program to use the time control and to trigger On pick event

Introduction about WML Tags:

The ontimer event is used to trigger an event after a given time period. Suppose students wants to display a message after 5 seconds of loading a card then you can use this event to do so.

Here is the syntax to define an event handler for **ontimer** event:

<onevent type="ontimer">

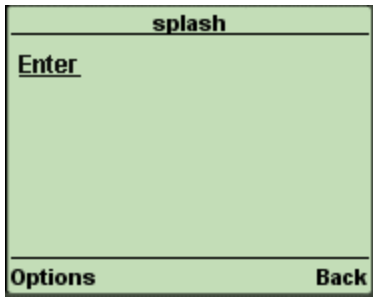
A task to be performed.

</onevent>

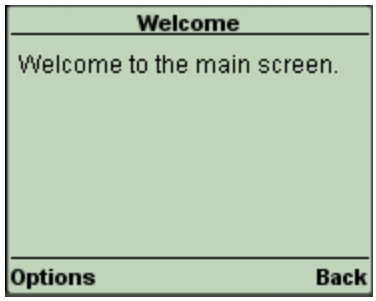
<timer value="50"/>

OUTPUT:

When you load this program, it shows you the following screen:



If you do not select given Enter option then after 5 seconds you will be directed to welcome page and following screen will be displayed automatically.



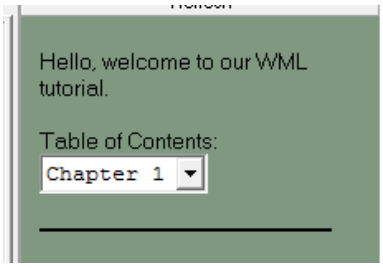
14.1 Write a WML program to trigger On pick event

Introduction about WML Tags:

The onpick attribute is a great shortcut if you are using a select menu. Instead of writing a lot of code that allows the user to go to another card if an option is selected, you can simply place the destination into the onpick attribute.

```
Select a tutorial :
<select title="tutorials" name="selection_list">
  <option onpick="#xhtml">XHTML Tutorial</option>
  <option onpick="#wap">WAP Tutorial</option>
</select>
```

Output:



14.3 Short Questions:

Q.No.1 Does WAP run over GPRS?

Q.No.2 Which Security is used in WAP?

Q.No.3 Is WML case sensitive?

Q.No.4 What does Post field tag do?

15. Experiment No. 6

AIM : Write a WML script to find maximum out of two numbers with help of inbuilt function Lang.Max() and to find absolute value with help of inbuilt function Lang.abs()

Introduction about WML Script:

Syntax: Lang.max(value1, value2)

value1:

The mandatory value1 parameter can be any positive or negative single-precision floating-point number or any positive or negative integer number (including zero). The floating-point number must reside on or between the largest number and the smallest nonzero number supported by the browser. These boundary values can be determined using the Float.maxFloat and Float.minFloat functions. The integer number must reside on or between the largest positive integer and the largest negative integer supported by the browser. These boundary values can be determined by using the Lang.maxInt and Lang.minInt functions.

value2:

The mandatory value2 parameter can be any positive or negative single-precision floating-point number or any positive or negative integer number (including zero). The floating-point number must reside on or between the largest number and the smallest nonzero number supported by the browser. These boundary values can be determined using the Float.maxFloat and Float.minFloat functions. The integer number must reside on or between the largest positive integer and the largest negative integer supported by the browser. These boundary values can be determined by using the Lang.maxInt and Lang.minInt functions.

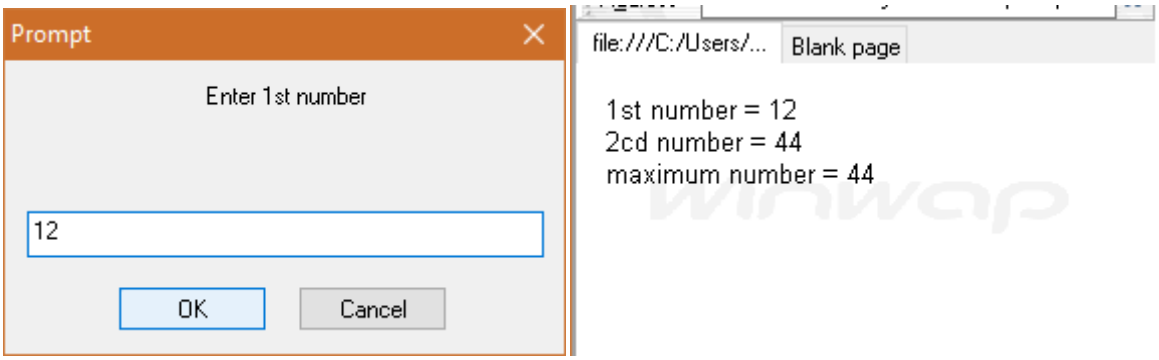
Returns the maximum of two numbers.

The Lang.max function compares two given numbers and then returns the larger value. If the two numbers prove to be equal, the first number is selected. The value and type (floating-point or integer) of the selected number is not changed.

```
extern function findmax()
{
  var result1 = Dialogs.prompt("Enter 1st number", "");
  var result2 = Dialogs.prompt("Enter 2cd number", "");
  var maxnum = Lang.max(result1, result2);
  WMLBrowser.setVar("number1", result1);
  WMLBrowser.setVar("number2", result2);
  WMLBrowser.setVar("maxnumber", maxnum);
```

```
  WMLBrowser.go("MaxExample.wml#card2");
};
```

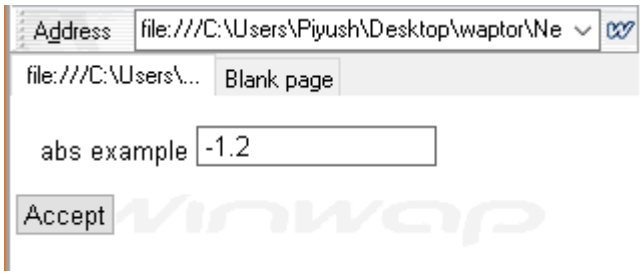
OUTPUT:



Syntax: Lang.abs(value)
Value: The mandatory value parameter can be any positive or negative single-precision floating-point number or any positive or negative integer number (including zero). The floating-point number must reside on or between the largest number and the smallest nonzero number supported by the browser. These boundary values can be determined using the Float.maxFloat and Float.minFloat functions.

```
extern function findabs(n)
{
  var absnum = Lang.abs(n);
  WMLBrowser.setVar("number", n);
  WMLBrowser.setVar("absnumber", absnum);
  WMLBrowser.go("abs.wml#card2");
};
```

Output:



15.3 Short Questions:

1.What is the function of WMLScript?

2.What is the use of WML in WMLScript?

3.What is the use of WML decks? - WMLScript

4.What is the process where WML cards request the device to access WAP? - WMLScript

5.What is the support of mobile devices for WMLScript?

6.What is the process of adding the client side logic to WAP using WMLScript?

7.What is the purpose of using WMLScript?

8.What are the data types used in WMLScript?

9.What are the different components of WMLScript?

10.What is the function of WMLScript Control Statements?

11.How does WMLScript function written?

12.What are the standard libraries used by WMLScripts?

16. Experiment No. 7

AIM: Write a Program in NS3 to Simulate OLSR

16.1 STEPS:

1. Create a wireless mobile ad-hoc network with three nodes Node1, Node2 and Node3. Install the OLSR routing protocol on these nodes.
2. Place them such that Node1 and Node3 are just out of reach of each other.
3. Create a UDP client on Node1 and the corresponding server on Node3.
4. Schedule Node1 to begin sending packets to Node3 at time 1s.
5. Verify whether Node1 is able to send packets to Node3.
6. Make Node2 move between Node1 and Node3 such that Node2 is visible to both A and C. This should happen at time 20s. Ensure that Node2 stays in that position for another 15s.
7. Verify whether Node1 is able to send packets to Node3.
8. At time 35s, move Node2 out of the region between Node1 and Node3 such that it is out of each other's transmission ranges again.
9. Verify whether Node1 is able to send packets to Node3.
10. To verify whether data transmissions occur in the above scenarios, use either the tracing mechanism or a RecvCallback() for Node3's socket.
11. Plot the number of bytes received versus time at Node3.
12. Show the pcap traces at Node 2's Wifi interface, and indicate the correlation between Node2's packet reception timeline and Node2's mobility.

OLSR routing over some point-to-point links

```
 Network topology
 // n0
\ 5 Mb/s, 2ms
  \ 1.5Mb/s, 10ms
n2 -------------------------n3---------n4
  /
 / 5 Mb/s, 2ms
  n1
// - all links are point-to-point links with indicated one-way BW/delay
 // - CBR/UDP flows from n0 to n4, and from n3 to n1
 // - UDP packet size of 210 bytes, with per-packet interval 0.00375 sec.
 // (i.e., DataRate of 448,000 bps)
 // - DropTail queues
 // - Tracing of queues and packet receptions to file "simple-point-to-point-olsr.tr"
```

16.2 Program:

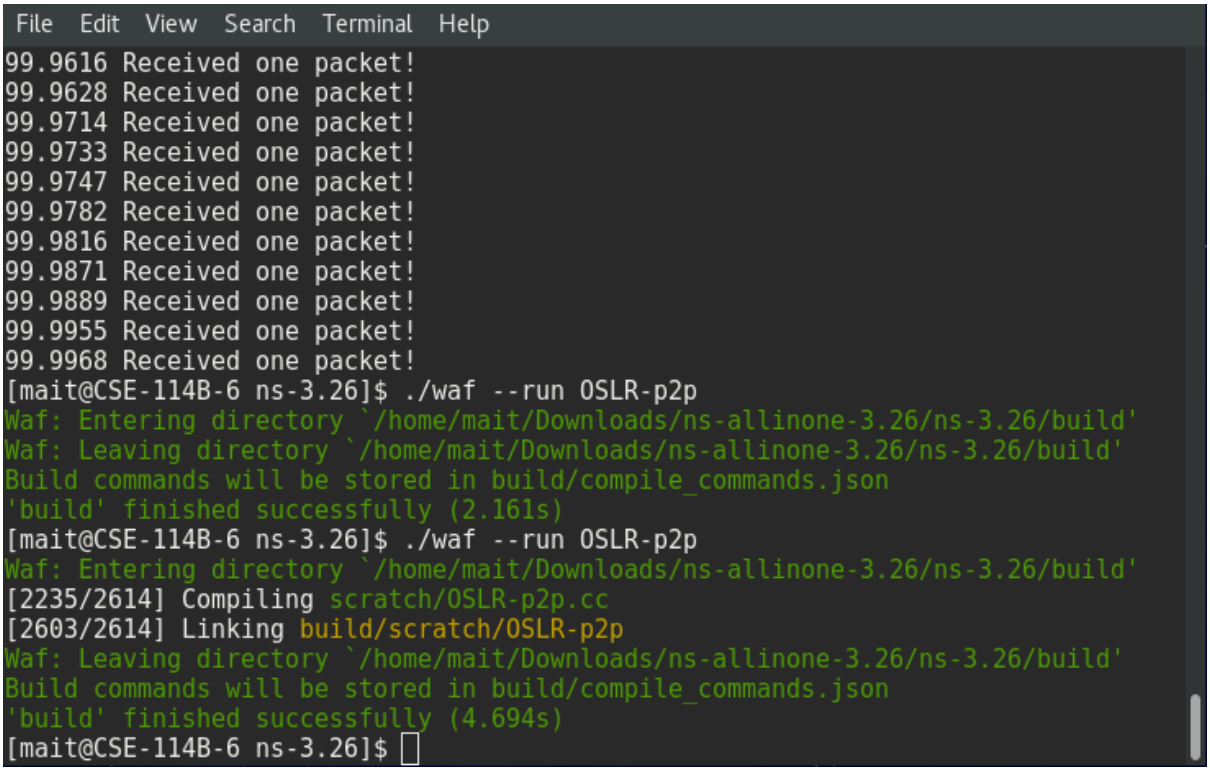
```
 #include <iostream>
#include <fstream>
#include <string>
#include <cassert>

#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include "ns3/olsr-helper.h"
#include "ns3/ipv4-static-routing-helper.h"
#include "ns3/ipv4-list-routing-helper.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("SimplePointToPointOlsrExample");

int
main (int argc, char *argv[])
{
 // Users may find it convenient to turn on explicit debugging
 // for selected modules; the below lines suggest how to do this
#if 0
 LogComponentEnable ("SimpleGlobalRoutingExample", LOG_LEVEL_INFO);
#endif

 // Set up some default values for the simulation. Use the

 Config::SetDefault ("ns3::OnOffApplication::PacketSize", UintegerValue (210));
 Config::SetDefault ("ns3::OnOffApplication::DataRate", StringValue ("448kb/s"));

 //DefaultValue::Bind ("DropTailQueue::m_maxPackets", 30);

 // Allow the user to override any of the defaults and the above
 // DefaultValue::Bind ()s at run-time, via command-line arguments
 CommandLinecmd;
 cmd.Parse (argc, argv);

 // Here, we will explicitly create four nodes. In more sophisticated
 // topologies, we could configure a node factory.
 NS_LOG_INFO ("Create nodes.");
 NodeContainer c;
 c.Create (5);
 NodeContainer n02 = NodeContainer (c.Get (0), c.Get (2));
 NodeContainer n12 = NodeContainer (c.Get (1), c.Get (2));
 NodeContainer n32 = NodeContainer (c.Get (3), c.Get (2));
 NodeContainer n34 = NodeContainer (c.Get (3), c.Get (4));

 // Enable OLSR
```

```
  NS_LOG_INFO ("Enabling OLSR Routing.");
  OlsrHelperolsr;

  Ipv4StaticRoutingHelper staticRouting;

  Ipv4ListRoutingHelperlist;
  list.Add (staticRouting, 0);
  list.Add (olsr, 10);

InternetStackHelper internet;
  internet.SetRoutingHelper (list); // has effect on the next Install ()
  internet.Install (c);

  // We create the channels first without any IP addressing information
  NS_LOG_INFO ("Create channels.");
  PointToPointHelper p2p;
  p2p.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
  p2p.SetChannelAttribute ("Delay", StringValue ("2ms"));
  NetDeviceContainer nd02 = p2p.Install (n02);
  NetDeviceContainer nd12 = p2p.Install (n12);
  p2p.SetDeviceAttribute ("DataRate", StringValue ("1500kbps"));
  p2p.SetChannelAttribute ("Delay", StringValue ("10ms"));
  NetDeviceContainer nd32 = p2p.Install (n32);
  NetDeviceContainer nd34 = p2p.Install (n34);

  // Later, we add IP addresses.
  NS_LOG_INFO ("Assign IP Addresses.");
Ipv4AddressHelper ipv4;
  ipv4.SetBase ("10.1.1.0", "255.255.255.0");
  Ipv4InterfaceContainer i02 = ipv4.Assign (nd02);

 ipv4.SetBase ("10.1.2.0", "255.255.255.0");
  Ipv4InterfaceContainer i12 = ipv4.Assign (nd12);

  ipv4.SetBase ("10.1.3.0", "255.255.255.0");
  Ipv4InterfaceContainer i32 = ipv4.Assign (nd32);

  ipv4.SetBase ("10.1.4.0", "255.255.255.0");
Ipv4InterfaceContainer i34 = ipv4.Assign (nd34);

  // Create the OnOff application to send UDP datagrams of size
  // 210 bytes at a rate of 448 Kb/s from n0 to n4
  NS_LOG_INFO ("Create Applications.");
  uint16_t port = 9; // Discard port (RFC 863)

  OnOffHelper onoff ("ns3::UdpSocketFactory",
  InetSocketAddress (i34.GetAddress (1), port));
  onoff.SetConstantRate (DataRate ("448kb/s"));

  ApplicationContainer apps = onoff.Install (c.Get (0));
 apps.Start (Seconds (1.0));
```

```
apps.Stop (Seconds (10.0));

// Create a packet sink to receive these packets
PacketSinkHelpersink ("ns3::UdpSocketFactory",
InetSocketAddress (Ipv4Address::GetAny (), port));

apps = sink.Install (c.Get (3));
apps.Start (Seconds (1.0));
apps.Stop (Seconds (10.0));

// Create a similar flow from n3 to n1, starting at time 1.1 seconds
onoff.SetAttribute ("Remote",
AddressValue (InetSocketAddress (i12.GetAddress (0), port)));
apps = onoff.Install (c.Get (3));
apps.Start (Seconds (1.1));
apps.Stop (Seconds (10.0));

// Create a packet sink to receive these packets
apps = sink.Install (c.Get (1));
apps.Start (Seconds (1.1));
apps.Stop (Seconds (10.0));

AsciiTraceHelper ascii;
p2p.EnableAsciiAll (ascii.CreateFileStream ("simple-point-to-point-olsr.tr"));
p2p.EnablePcapAll ("simple-point-to-point-olsr");

Simulator::Stop (Seconds (30));

NS_LOG_INFO ("Run Simulation.");
Simulator::Run ();
Simulator::Destroy ();
NS_LOG_INFO ("Done.");

return 0;
}
```

AIM: Write a program to show the introduction of multi point relay set as a subset of the neighbor set does not destroy the connectivity properties of the network

16.3 Output:


File Edit View Search Terminal Help
99.9616 Received one packet!
99.9628 Received one packet!
99.9714 Received one packet!
99.9733 Received one packet!
99.9747 Received one packet!
99.9782 Received one packet!
99.9816 Received one packet!
99.9871 Received one packet!
99.9889 Received one packet!
99.9955 Received one packet!
99.9968 Received one packet!
[mait@CSE-114B-6 ns-3.26]$ ./waf --run OSLR-p2p
Waf: Entering directory `/home/mait/Downloads/ns-allinone-3.26/ns-3.26/build'
Waf: Leaving directory `/home/mait/Downloads/ns-allinone-3.26/ns-3.26/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (2.161s)
[mait@CSE-114B-6 ns-3.26]$ ./waf --run OSLR-p2p
Waf: Entering directory `/home/mait/Downloads/ns-allinone-3.26/ns-3.26/build'
[2235/2614] Compiling scratch/OSLR-p2p.cc
[2603/2614] Linking build/scratch/OSLR-p2p
Waf: Leaving directory `/home/mait/Downloads/ns-allinone-3.26/ns-3.26/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (4.694s)
[mait@CSE-114B-6 ns-3.26]$

16.4 Short Questions:

Q. No 1 List of Security Issues in Adhoc Networks?

Q. No. 2 What is Multi Casting?

Q. No. 3 What is MANET?

Q. No 4 What are the Characteristics of MANETs?

17. Experiment No. 8

AIM: Write a Program in NS3 to Simulate AODV

17.1 Introduction

Ad hoc On-Demand Distance Vector (AODV) [1] is a routing protocol for mobile ad-hoc net- works. AODV is a reactive routing protocol which means that routes are discovered when necessary, routes are also maintained as long as necessary. Every node maintains its monoton- ically increasing sequence number which increases every time the node notices change in the neighborhood topology. Routing tables are used to store routing information where data is stored as follows:

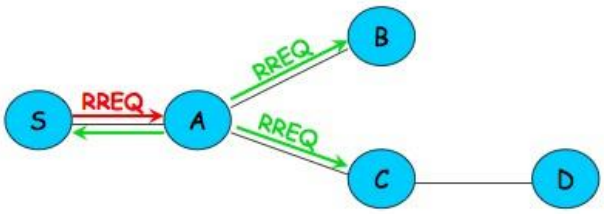
<destination addr, next-hop addr, destination sequence number, life_time>

life_time gets updated every time the route is used (If not used within its life_time, route expires).

3 AODV Routing Protocol

AODV Routing Protocol works as follows: Assuming in the network in the figure below we need to discover route between S and D.

Figure 1: AODV protocol working on five nodes.



It works as follows:

1. Node S needs route to D (no routes in routing table)
2. S Creates a Route Request (RREQ): D^0s IP addr, D^0s seq # , S^0s IP addr, S^0s seq #, hopcount (0)
3. Node S broadcasts RREQ to neighbors
4. Node A receives RREQ

(a) Makes reverse route entry for S (dest=S, nexthop=S, hopcount=1) (b) It has no route to D, so rebroadcasts RREQ

5. Node C receives RREQ

   (a) Makes reverse route entry for S (dest=S, nexthop=A, hopcount=2)

   (b) It has a route to D, and the seq # for route to D is >= D^0s seq # in RREQ (c) C creates a Route Reply (RREP)

   (d) Enter D^0s IPAddr, seq#, S^0s IPAddr, Hopcount to D (1)

   (e) Unicasts RREP to A

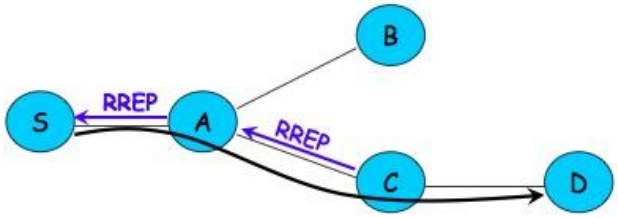
6. Node A receives RREP

   (a) Makes forward route entry to D (dest=D, nexthop=C, hopcount=2) (b) Unicasts RREP to S

7. Node S receives RREP

   (a) Makes forward route entry to D (dest = D, nexthop = A, hopcount=3) (b) Sends data packets on route to D

Figure 2: AODV protocol: Node S receives PREP.



17.2 Program

Source Code:

```
#include "ns3/aodv-module.h"
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/mobility-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/wifi-module.h"
#include "ns3/v4ping-helper.h"
#include <iostream>
#include <cmath>

using namespace ns3;
```

```
class AodvExample
{
public:
 AodvExample ();
 bool Configure (int argc, char **argv);
 void Run ();
 void Report (std::ostream & os);

private:

 // parameters
 uint32_t size;
 doublestep;
 doubletotalTime;
 boolpcap;
 boolprintRoutes;

 // network
 NodeContainernodes;
 NetDeviceContainerdevices;
 Ipv4InterfaceContainerinterfaces;

private:
 void CreateNodes ();
void CreateDevices ();
 void InstallInternetStack ();
 void InstallApplications ();
};

int main (int argc, char **argv)
{
 AodvExampletest;
 if (!test.Configure (argc, argv))
 NS_FATAL_ERROR ("Configuration failed. Aborted.");

 test.Run ();
 test.Report (std::cout);
 return 0;
}

//-----------------------------------------------------------------------------
AodvExample::AodvExample () :
 size (10),
 step (100),
 totalTime (10),
 pcap (true),
 printRoutes (true)
{
}

bool
```

```
AodvExample::Configure (int argc, char **argv)
{
 // Enable AODV logs by default. Comment this if too noisy
 // LogComponentEnable("AodvRoutingProtocol", LOG_LEVEL_ALL);

 SeedManager::SetSeed (12345);
 CommandLinecmd;

 cmd.AddValue ("pcap", "Write PCAP traces.", pcap);
 cmd.AddValue ("printRoutes", "Print routing table dumps.", printRoutes);
 cmd.AddValue ("size", "Number of nodes.", size);
 cmd.AddValue ("time", "Simulation time, s.", totalTime);
 cmd.AddValue ("step", "Grid step, m", step);

 cmd.Parse (argc, argv);
 returntrue;
}

void
AodvExample::Run ()
{
// Config::SetDefault ("ns3::WifiRemoteStationManager::RtsCtsThreshold", UintegerValue
        (1)); // enable rts cts all the time.
 CreateNodes ();
 CreateDevices ();
 InstallInternetStack ();
 InstallApplications ();

 std::cout <<"Starting simulation for "<<totalTime<<" s ...\n";

 Simulator::Stop (Seconds (totalTime));
 Simulator::Run ();
 Simulator::Destroy ();
}

void
AodvExample::Report (std::ostream &)
{
}

void
AodvExample::CreateNodes ()
{
 std::cout <<"Creating "<< (unsigned)size<<" nodes "<<step<<" m apart.\n";
 nodes.Create (size);
 // Name nodes
 for (uint32_t i = 0; i <size; ++i)
 {
 std::ostringstream os;
 os <<"node-"<< i;
 Names::Add (os.str (), nodes.Get (i));
```

```
 }
 // Create static grid
 MobilityHelpermobility;
 mobility.SetPositionAllocator ("ns3::GridPositionAllocator",
 "MinX", DoubleValue (0.0),
 "MinY", DoubleValue (0.0),
 "DeltaX", DoubleValue (step),
"DeltaY", DoubleValue (0),
 "GridWidth", UintegerValue (size),
 "LayoutType", StringValue ("RowFirst"));
 mobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel");
 mobility.Install (nodes);
}

 void
        AodvExample::CreateDevices ()
 {
 WifiMacHelper wifiMac;
 wifiMac.SetType ("ns3::AdhocWifiMac");
 YansWifiPhyHelper wifiPhy = YansWifiPhyHelper::Default ();
 YansWifiChannelHelper wifiChannel = YansWifiChannelHelper::Default ();
 wifiPhy.SetChannel (wifiChannel.Create ());
 WifiHelperwifi;
 wifi.SetRemoteStationManager ("ns3::ConstantRateWifiManager", "DataMode",
        StringValue ("OfdmRate6Mbps"), "RtsCtsThreshold", UintegerValue (0));
 devices = wifi.Install (wifiPhy, wifiMac, nodes);

 if (pcap)
 {
 wifiPhy.EnablePcapAll (std::string ("aodv"));
 }
 }

 void
 AodvExample::InstallInternetStack ()
 {
 AodvHelper aodv;
// you can configure AODV attributes here using aodv.Set(name, value)
 InternetStackHelperstack;
 stack.SetRoutingHelper (aodv); // has effect on the next Install ()
 stack.Install (nodes);
 Ipv4AddressHelperaddress;
 address.SetBase ("10.0.0.0", "255.0.0.0");
 interfaces = address.Assign (devices);

 if (printRoutes)
 {
 Ptr<OutputStreamWrapper> routingStream = Create<OutputStreamWrapper>
        ("aodv.routes", std::ios::out);
 aodv.PrintRoutingTableAllAt (Seconds (8), routingStream);
 }
```

```
}

void
AodvExample::InstallApplications ()
{
 V4PingHelper ping (interfaces.GetAddress (size - 1));
 ping.SetAttribute ("Verbose", BooleanValue (true));

 ApplicationContainer p = ping.Install (nodes.Get (0));
 p.Start (Seconds (0));
 p.Stop (Seconds (totalTime) - Seconds (0.001));
 // move node away
 Ptr<Node> node = nodes.Get (size/2);
 Ptr<MobilityModel> mob = node->GetObject<MobilityModel> ();
 Simulator::Schedule (Seconds (totalTime/3), &MobilityModel::SetPosition, mob, Vector
        (1e5, 1e5, 1e5));
}
```

Output:

```
[2262/2614] Compiling scratch/scratch-simulator.cc
[2530/2614] Linking build/scratch/second
[2531/2614] Linking build/scratch/first
[2532/2614] Linking build/scratch/star
[2568/2614] Linking build/scratch/OSLR-p2p
[2599/2614] Linking build/scratch/static-routing-slash32
[2600/2614] Linking build/scratch/subdir/subdir
[2601/2614] Linking build/scratch/AODV
[2602/2614] Linking build/scratch/scratch-simulator
[2603/2614] Linking build/scratch/DSDV
Waf: Leaving directory `/home/mait/Downloads/ns-allinone-3.26/ns-3.26/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (16.495s)
Creating 10 nodes 100 m apart.
Starting simulation for 10 s ...
PING  10.0.0.10 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=0 ttl=56 time=2057 ms
64 bytes from 10.0.0.10: icmp_seq=1 ttl=56 time=1059 ms
64 bytes from 10.0.0.10: icmp_seq=2 ttl=56 time=60 ms
64 bytes from 10.0.0.10: icmp_seq=3 ttl=56 time=8 ms
--- 10.0.0.10 ping statistics ---
10 packets transmitted, 4 received, 60% packet loss, time 9999ms
rtt min/avg/max/mdev = 8/796/2057/969.9 ms
[mait@CSE-114B-6 ns-3.26]$
```

17.3 Short Questions

Q.No1, How routing in Adhoc networks different from fixed networks?

Q.No 2. What is hidden and exposed terminal problem in Adhoc Networks?

Q.No 3 What is Hiper Access?

Q.No 4 What are hybrid routing Protocols?

18. Experiment No. 9

AIM: Make an application of your choice using WML or Android.

18.1 Steps:

1. Set up the Android development environment

Students are required to have Windows XP or later, or Mac OS X 10.5.8 or a later version to start Android application development process. Then, there are four tools that students will need and they are available on the Internet for free:

1. Java JDK5 or JDK6
2. Android SDK
3. Eclipse IDE for Java Developers (optional)
4. Android Development Tools (ADT) Eclipse Plugin (optional)

2. Setup Java Development Kit (JDK)

Students can download the JDK and install it, which is pretty easy. After that, students just have to set PATH and JAVA_HOME variables to the folder where you have java and javac.

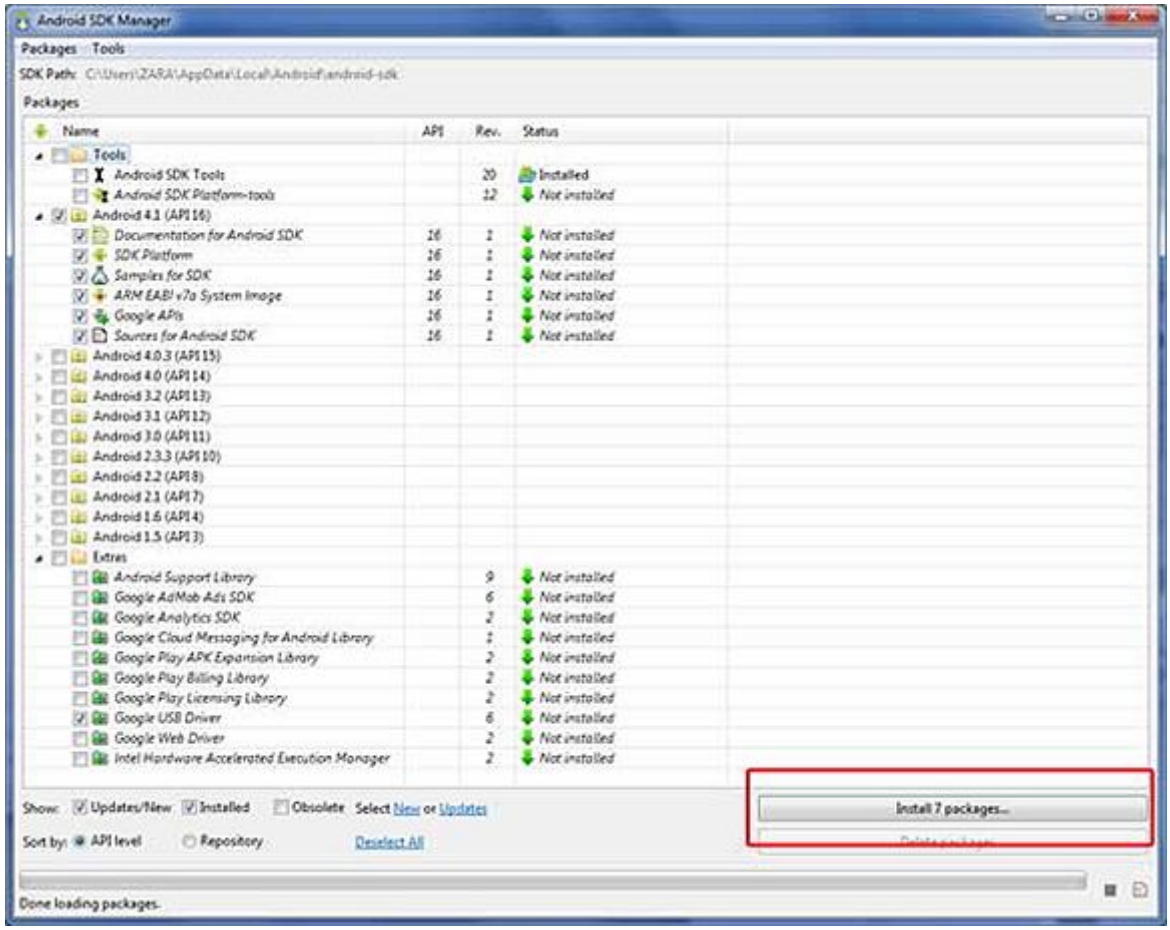
Note for Windows Users: If you installed the JDK in C:\jdk1.6.0_15 then you will have to add the following two lines in your C:\autoexec.bat file.

set PATH=C:\jdk1.6.0_15\bin;%PATH%

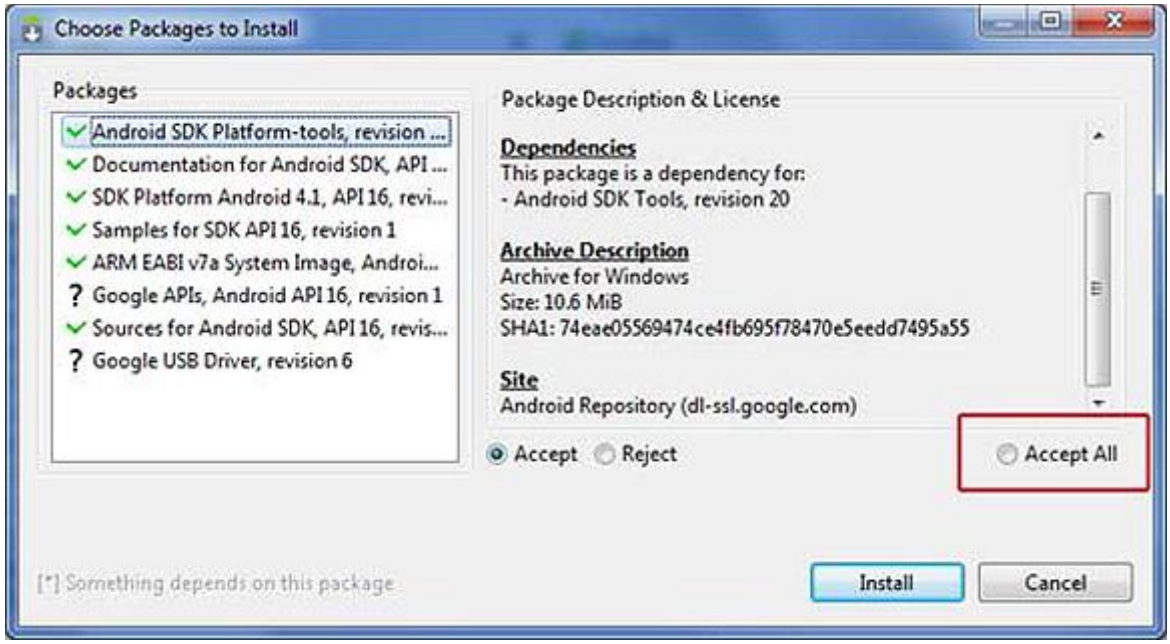
set JAVA_HOME=C:\jdk1.6.0_15

3. Configure Android SDK

After you have successfully installed the Android SDK, it is time to configure it. After installing the Android SDK, you will get a window like this:



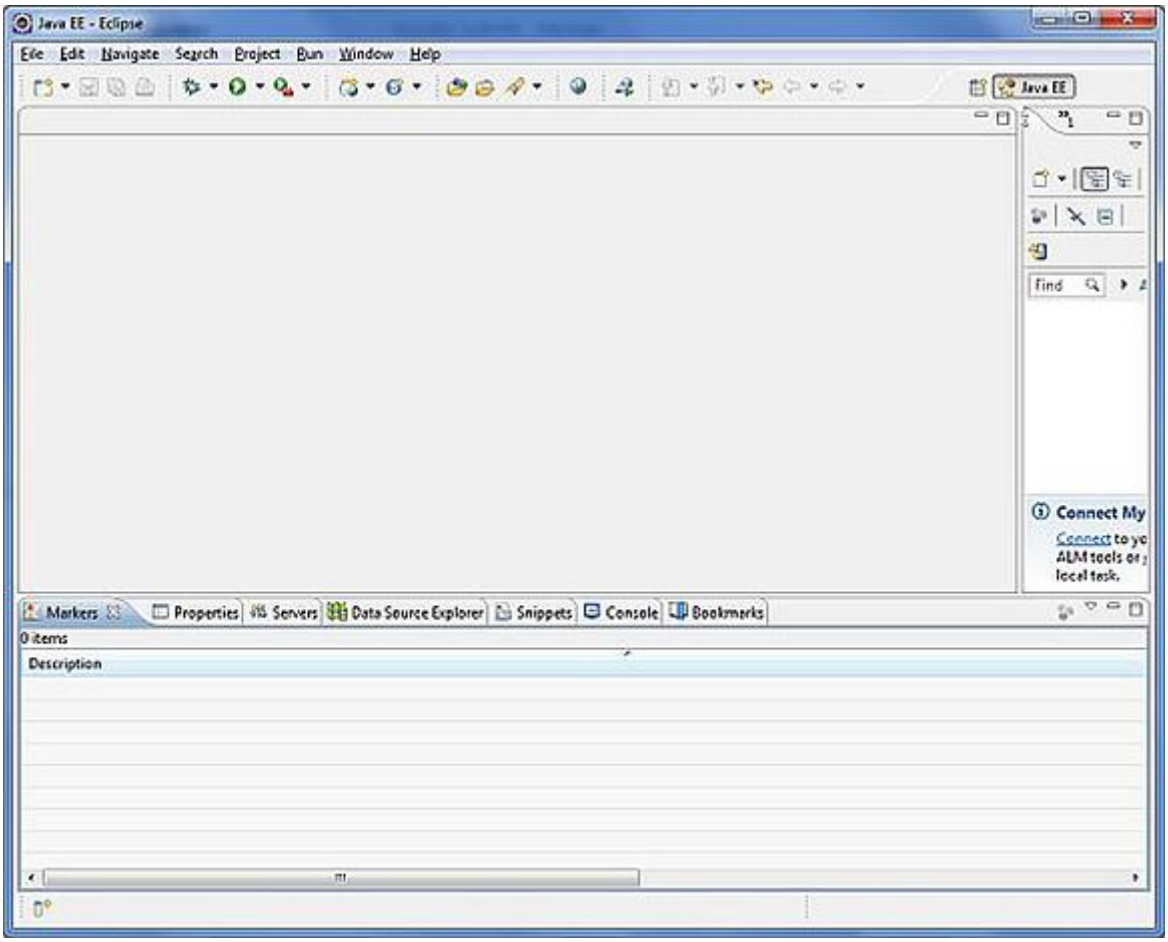
Just de-select the Documentation for Android SDK and Samples for SDKpackages if you want to reduce the installation size and time. Click on Install 7packages to continue with the installation. You will get a dialogue box like this:



It will take some time to install, Once it is done, you can close the SDK manager.

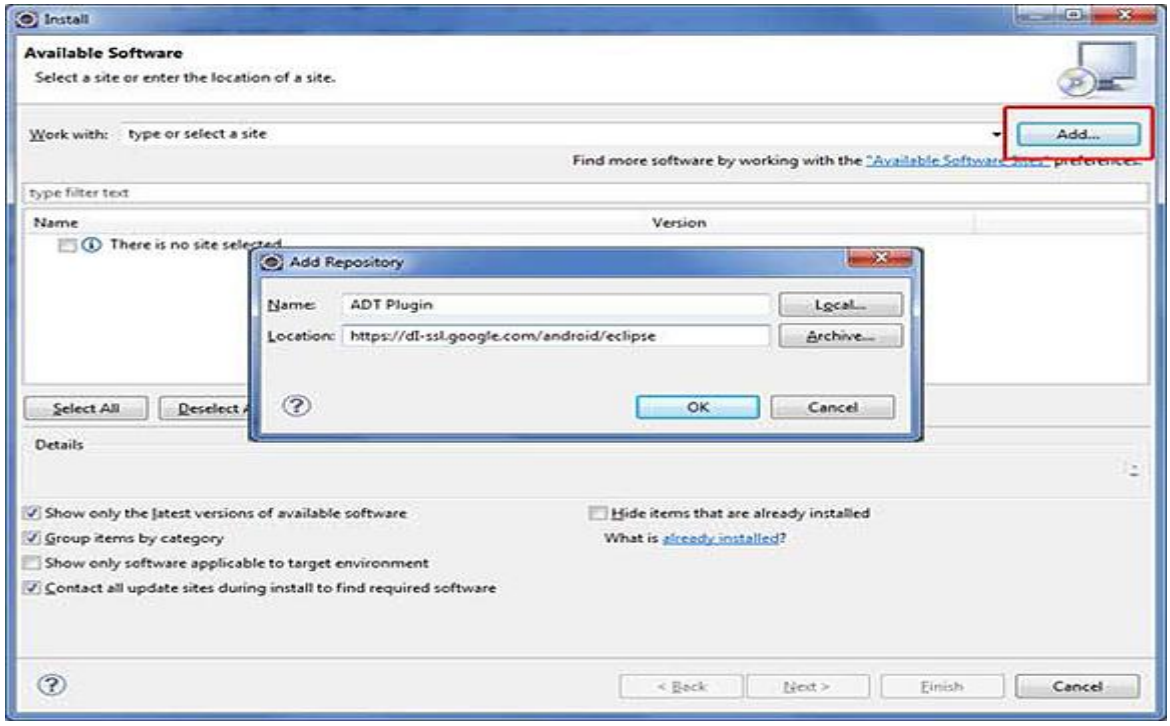
4. Setup Eclipse IDE

Install the latest version of Eclipse. After successful installation, it should display a window like this:

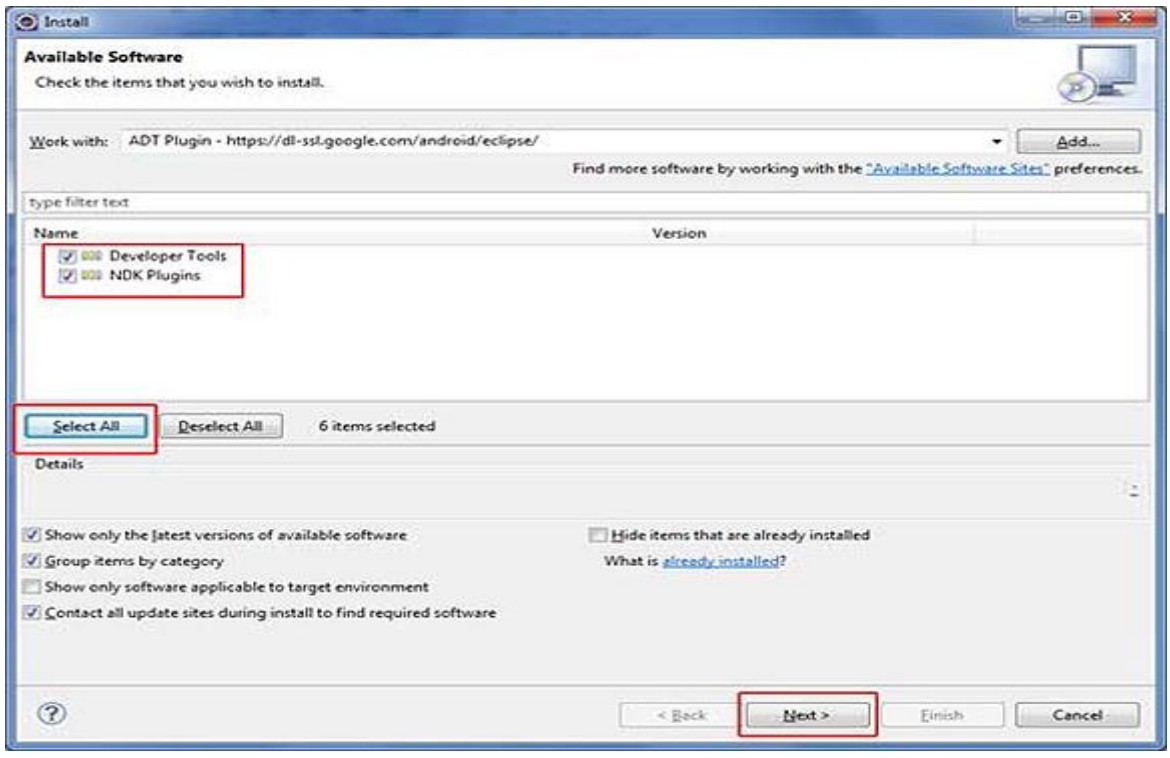


5. Setup Android Development Tools (ADT) Plug-in

Here students will learn to install the Android Development Tool plugin for Eclipse. To do this, students have to click on Help > Software Updates > Install New Software. This will display the following dialogue box.



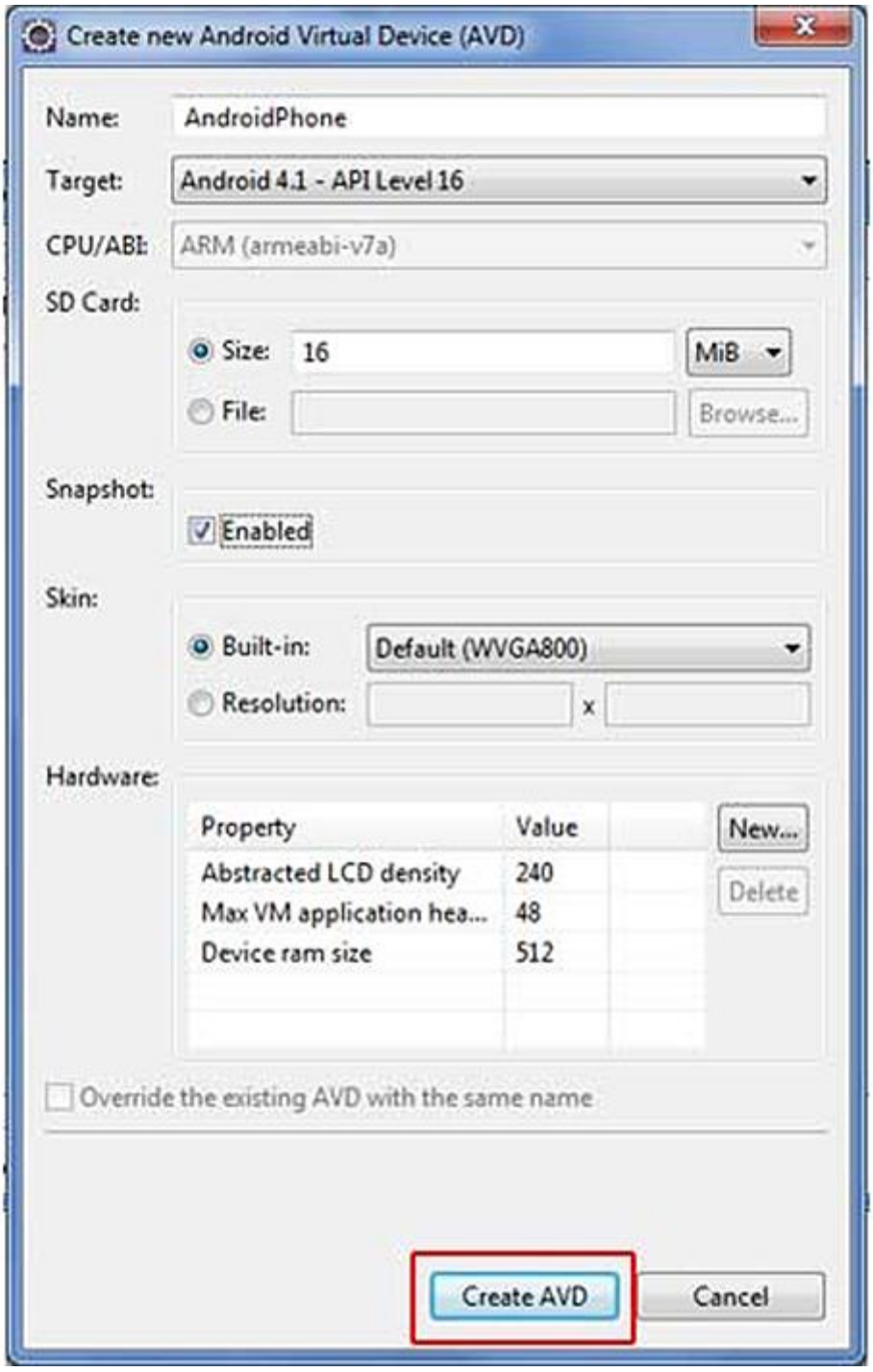
Just click on the Add button as shown in the picture and add https://dl-ssl.google.com/android/eclipse/ as the location. When you press OK, Eclipse will start to search for the required plug-in and finally it will list the found plug-ins.



6. Create Android Virtual Device

The last step is to create Android Virtual Device, which students will use to test your Android applications. To do this, open Eclipse and Launch Android AVD Manager from options Window

> AVD Manager and click on New which will create a successful Android Virtual Device. Use the screenshot below to enter the correct values.



Before we write the code, you need to know how to take input from the user. The most efficient way of taking input from the user is to use the Scanner class, which is found in the java.io package as it is just a two-step process.

```
Scanner scanner =  new Scanner(System.in);

String input   =  scanner.next();
```

```
/* or String input   =  (new
Scanner(System.in)).next(); */
```

Students are recommended using the Scanner class because it works equally well on command line as well as with Eclipse that we will use to make the Rock Paper Scissors app.

7. Calling the Java code in Eclipse

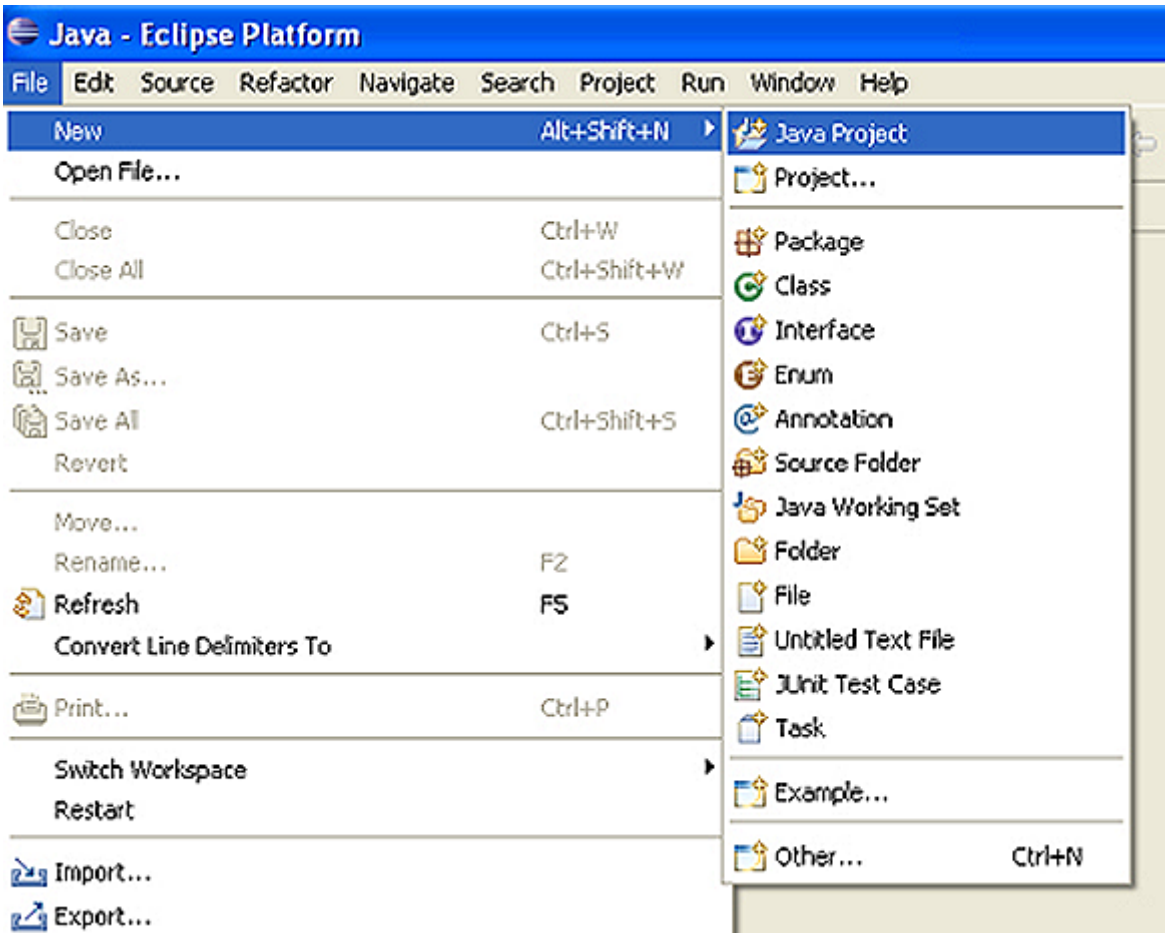
Sample Code is given below:--

```
import java.util.Scanner;

import java.util.Random;

public class Rock

{

public static void main(String[] args)

{

   String personPlay; //User's play -- "R", "P", or "S"

   String computerPlay = ""; //Computer's play -- "R", "P", or "S"

   int computerInt; //Randomly generated number used to determine     //computer's play

   String response;

   Scanner scan = new Scanner(System.in);

   Random generator = new Random();

   System.out.println("Hey, let's play Rock, Paper, Scissors!\n" +

                      "Please enter a move.\n" + "Rock = R, Paper" +

                      "= P, and Scissors = S.");


   System.out.println();

   //Generate computer's play (0,1,2)

   computerInt = generator.nextInt(3)+1;

   //Translate computer's randomly generated play to

   //string using if //statements
```

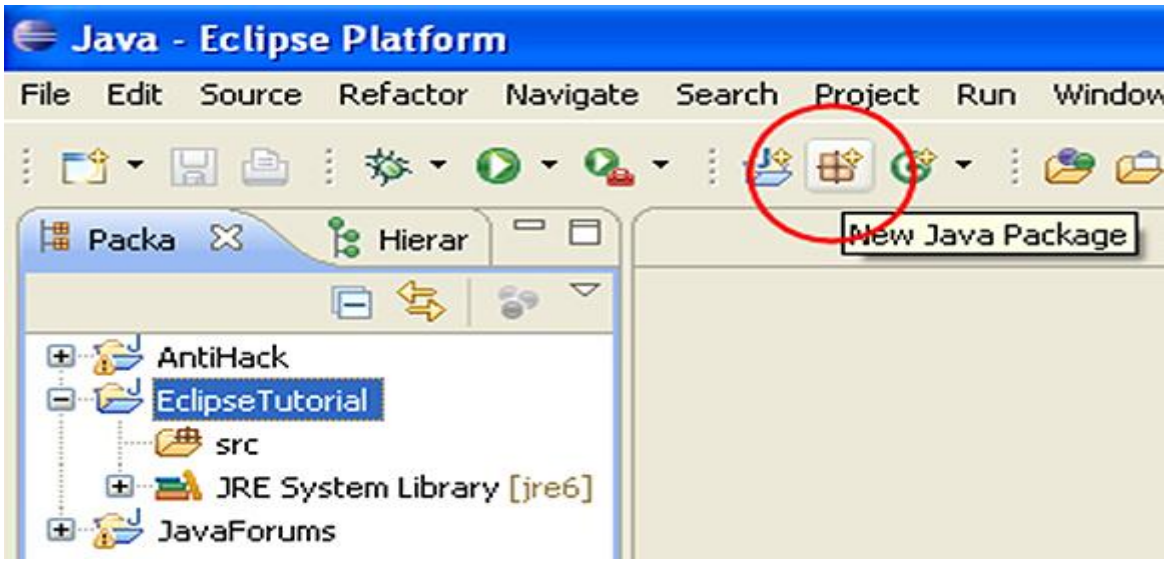
```
if (computerInt == 1)
   computerPlay = "R";
else if (computerInt == 2)
   computerPlay = "P";
else if (computerInt == 3)
   computerPlay = "S";
//Get player's play from input-- note that this is
// stored as a string
System.out.println("Enter your play: ");
personPlay = scan.next();
//Make player's play uppercase for ease of comparison
personPlay = personPlay.toUpperCase();
//Print computer's play
System.out.println("Computer play is: " + computerPlay);
//See who won. Use nested ifs
if (personPlay.equals(computerPlay))
   System.out.println("It's a tie!");
else if (personPlay.equals("R"))
   if (computerPlay.equals("S"))
      System.out.println("Rock crushes scissors. You win!!");
else if (computerPlay.equals("P"))
        System.out.println("Paper eats rock. You lose!!");
else if (personPlay.equals("P"))
   if (computerPlay.equals("S"))
   System.out.println("Scissor cuts paper. You lose!!");
else if (computerPlay.equals("R"))
        System.out.println("Paper eats rock. You win!!");
```

```
else if (personPlay.equals("S"))
    if (computerPlay.equals("P"))
    System.out.println("Scissor cuts paper. You win!!");
else if (computerPlay.equals("R"))
      System.out.println("Rock breaks scissors. You lose!!");
else
    System.out.println("Invalid user input.");
```

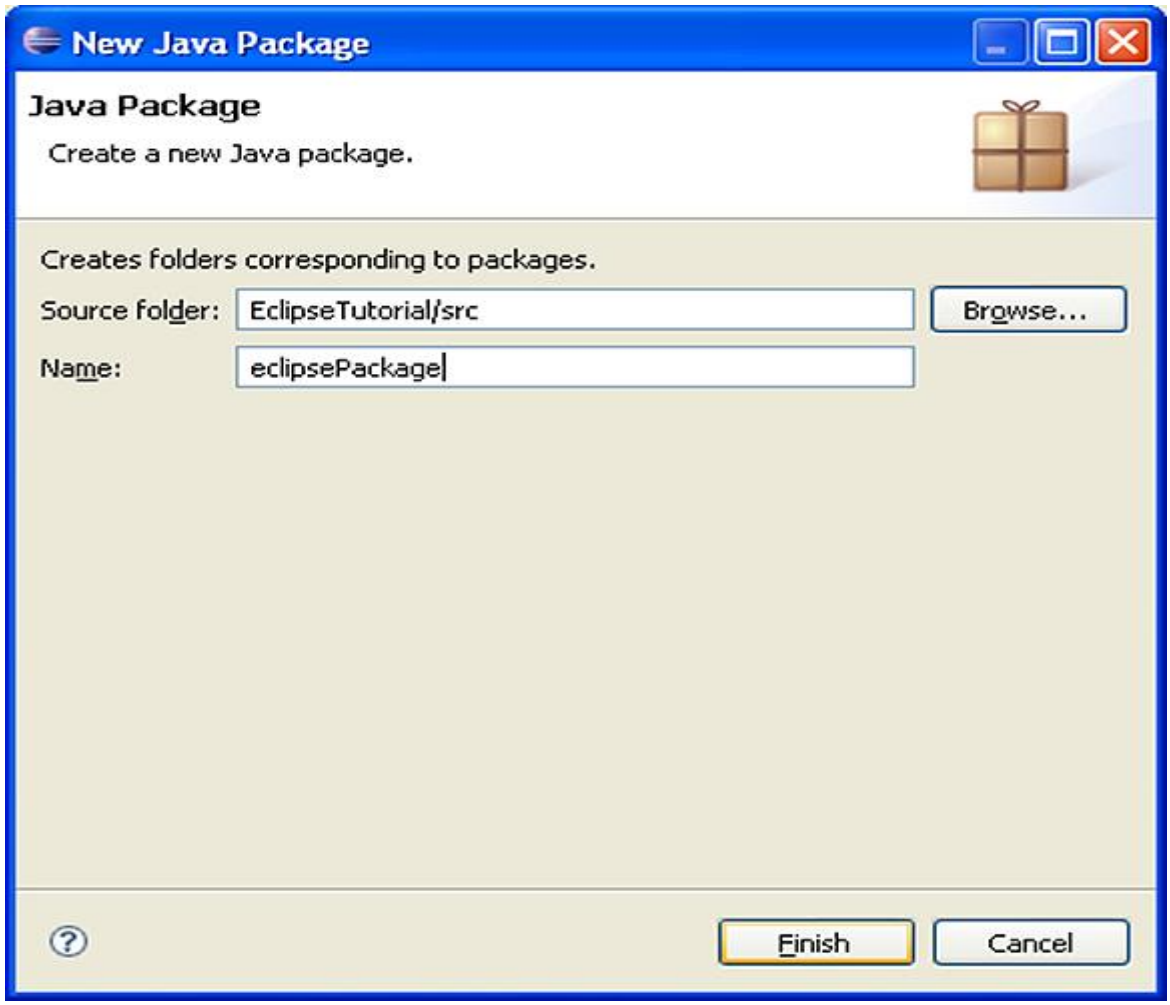
Now calling the Java code in Eclipse is a tricky process and it helps to pay attention while you're doing this. Launch Eclipse and click File > New > Java Project



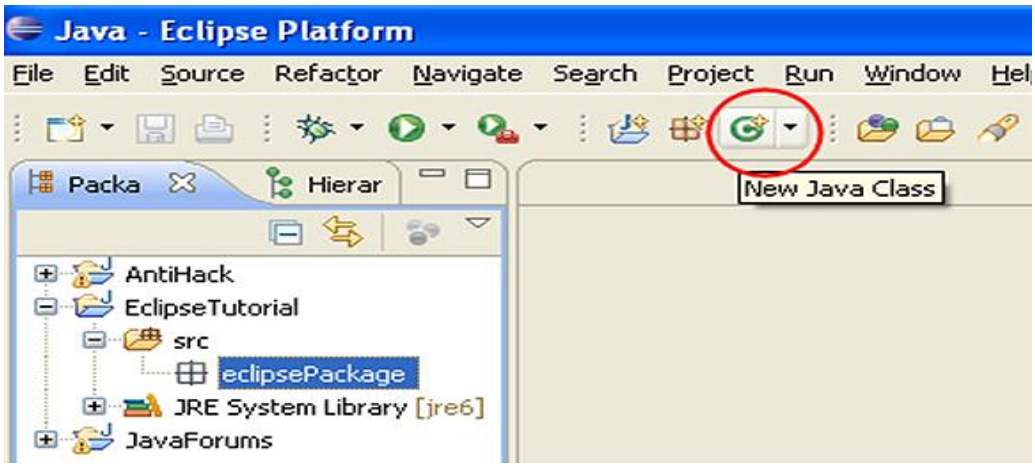
When the Create a Java Project box appears, it's time to give your project a name. Click on Finish to save it and it should appear in the Package Explorer window. Then we are supposed to add a package which will contain all our package files. Click on New Java Package icon to do this, as shown in the screenshot below.



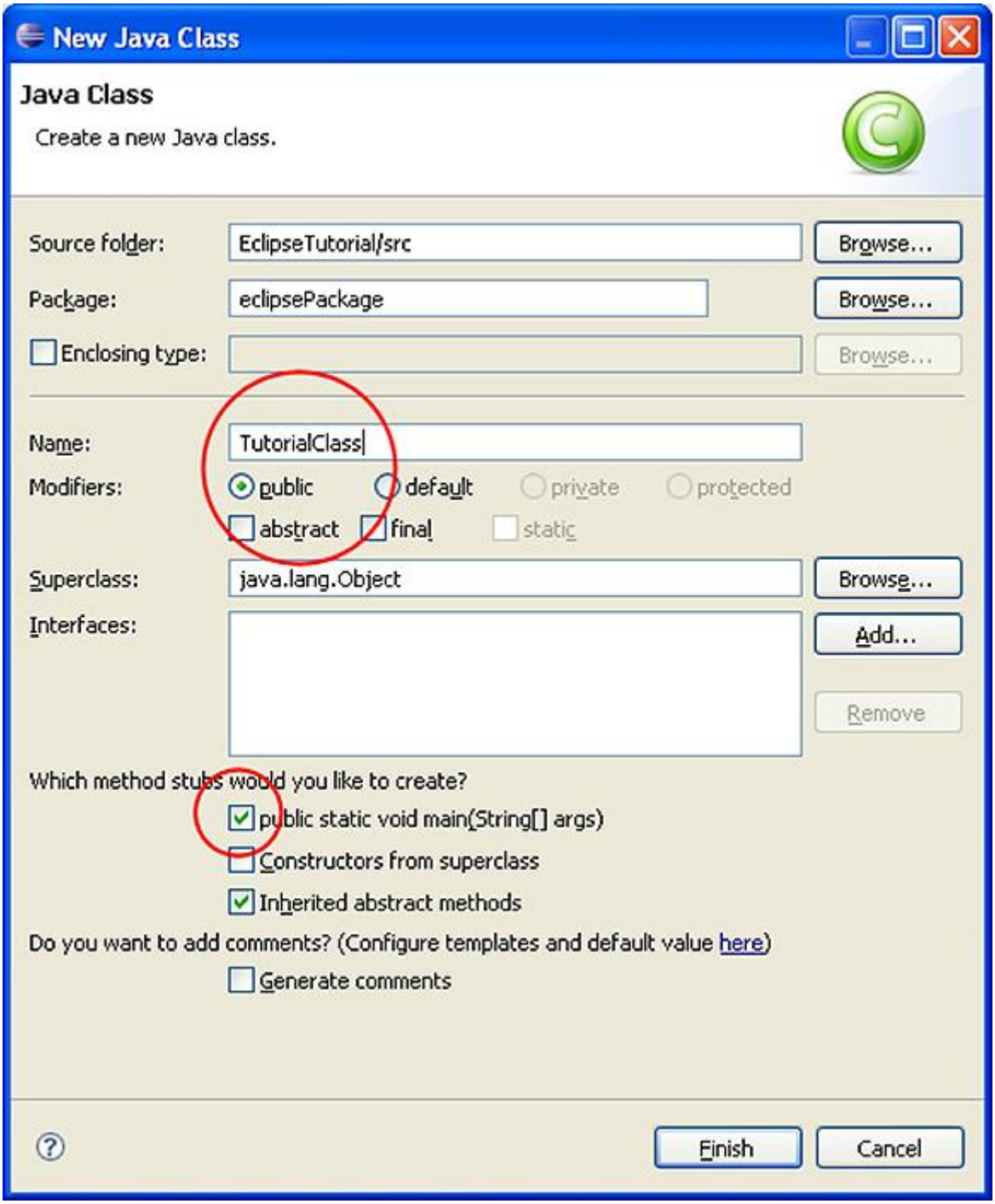
Name your project and then click Finish.



Now we need to add a Java Class, which is as easy as adding a Java Package.



After giving it a name, make sure that the following options are checked:



After you create a new class, it will show up in the Work Space where you can write or copy the code.

Now you need to build the application and to do this, Right Click on your Android Project and select Android Tools->Export Signed Application Package. After selecting the export button, select Create new keystore and it will take you to the location where you want to save it, so give it a name and save it. Fill in all the required fields that are self-explanatory and save it. You have successfully exported the apk file to your computer and you can test the app it on your android device.

8. Testing

Testing is as important as developing the app because your app will be of no use if it doesn't run properly or shows errors. Luckily, there is an integrated testing framework in Android Framework, which you can use to test all the aspects of your application. SDK tools can also help set up and test applications. SDK will help you test different aspects of your app no matter if you are planning on running your tests within an emulator or any Android device.

It is recommended using ADT for the testing process, as it is comparatively easier than the other tools. Using ADT, you can easily create a test project and link it to the application under test. The great thing about the ADT is that it automatically adds the necessary <instrumentation> element in the test package's manifest file.

9. To create a test project in Eclipse with ADT

Launch Eclipse from the Start Menu and click on File > New > Other and from the drop-down menu, click on Android Test Project, then at the bottom of the dialog click Next. Enter any name you want and in the Test Target panel, set An Existing Android Project and browse to the Rock Paper Scissors app you made. Now you should be able to see the wizard completing the Test Target Package, Application Name, and Package Name fields for you.

Choose the Android SDK platform from the Build Target Panel that the application to be tested is using. Click the Finish button to complete the Wizard and if it is disabled, look for error messages at the top to fix any problems.

18.3 Short Questions:

Q.No What is an Activity in Android?

Q.No 2 What is an APK format?

Q.No 3 What is an Intent?

Q.No 4 What is an Android Manifest File?

19. Experiment No. 10

AIM: Write a Android Program to create list view, grid view and database connectivity

Introduction: GRID VIEW

In android GridView is a view group that display items in two dimensional scrolling grid (rows and columns), the grid items are not necessarily predetermined but they are automatically inserted to the layout using a ListAdapter. Users can then select any grid item by clicking on it. GridView is default scrollable so we don't need to use ScrollView or anything else with GridView.

Adapter Is Used To Fill Data In Gridview: To fill the data in a GridView we simply use adapter and grid items are automatically inserted to a GridView using an Adapter which pulls the content from a source such as an arraylist, array or database. You can read full Adapter tutorial here.

GridView in Android Studio: Gridview is present inside Containers. From there you can drag and drop on virtual mobile screen to create it. Alternatively you can also XML code to create it.

Basic GridView code in XML:

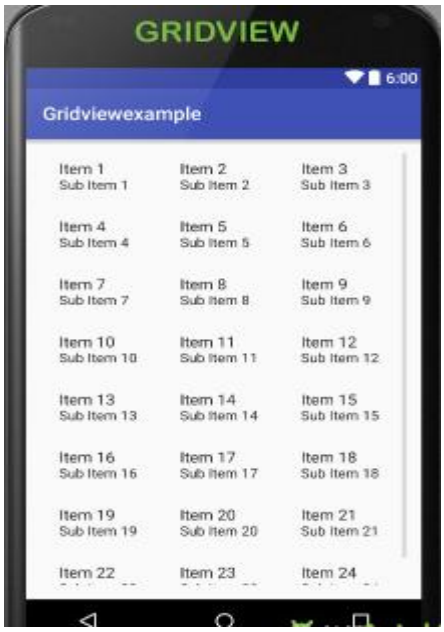
<GridView

android:id="@+id/simpleGridView"

android:layout_width="fill_parent"

android:layout_height="wrap_content"

android:numColumns="3"/>



Introduction: LIST VIEW

List of scrollable items can be displayed in Android using ListView. It helps you to displaying the data in the form of a scrollable list. Users can then select any list item by clicking on it. ListView is default scrollable so we do not need to use scroll View or anything else with ListView.

ListView is widely used in android applications. A very common example of ListView is your phone contact book, where you have a list of your contacts displayed in a ListView and if you click on it then user information is displayed.

Adapter: To fill the data in a ListView we simply use adapters. List items are automatically inserted to a list using an Adapter that pulls the content from a source such as an arraylist, array or database.

ListView in Android Studio: Listview is present inside Containers. From there you can drag and drop on virtual mobile screen to create it. Alternatively you can also XML code to create it.

Here is Android ListView XML Code:

```
<ListView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/simpleListView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    tools:context="abhiandroid.com.listexample.MainActivity">
</ListView>
```

Introduction: Database Connectivity

SQLite

SQLite is a relational database management system (RDBMS). If most RDBMSs such as MySQL, Oracle, etc. are standalone server processes, then SQLite is embedded because it is provided in the form of a library that is linked in applications.

Like other RDBMSs, data is accessed in a SQLite database by using Structured Query Language (SQL).

Android SQLite Java Classes

Cursor: a class provides access to the results of a database query. Its methods include:

close(): release all resources used by cursor and close it.

getCount(): returns the number of rows contained within the result set.

moveToFirst(): moves to the first row in the result set.

moveToLast(): moves to the last row in the result set.

moveToNext(): moves to the next row in the result set.

move(): moves by a specified offset from the current position in the result set.

get<type>() (such as getInt(), getDouble(), so on): returns the value of the specified <type> contained at the specified column index of the row at the current cursor position.

SQLiteDatabase: provides the primary interface between the application code and underlying SQLite database. Its methods include:

insert(): inserts a new row into a database table.

delete(): deletes rows from a database table

query(): performs a specified database query and returns matching results via a Cursor object.

execSQL(): executes a single SQL Statement that does not return result data.

rawQuery(): executes an SQL query statement and returns matching results in the form of a Cursor object.

SQLiteOpenHelper : is designed to make it easier to create and update databases. Its methods include:

onCreate(): called when the database is created for the first time.

onUpgrade(): called in the event that the application code contains a more recent database version number reference.

onOpen(): called when the database is opened.

getWritableDatabase(): opens or creates a database for reading and writing.

getReadableDatabase(): creates or opens a database for reading only.

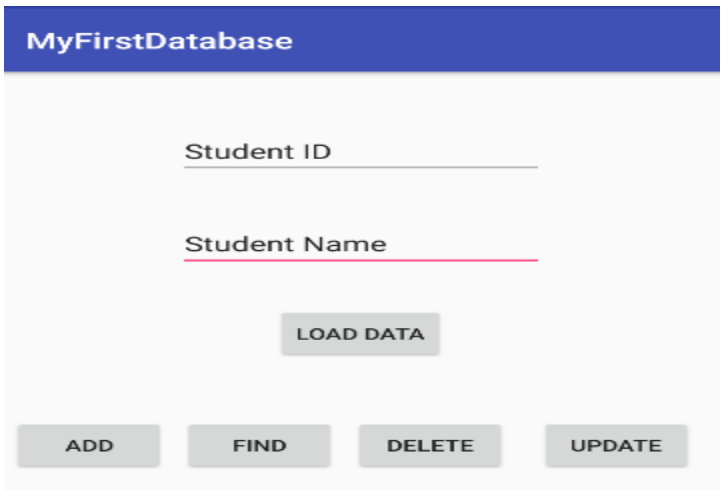
close(): closes the database.

ContentValues : allows key/value pairs to be declared consisting of table column identifiers and the values to be stored in each column. Its methods include:

put(): adds a value to the set.

For Example: Creating a Data Application, we have to follow certain steps as follows.

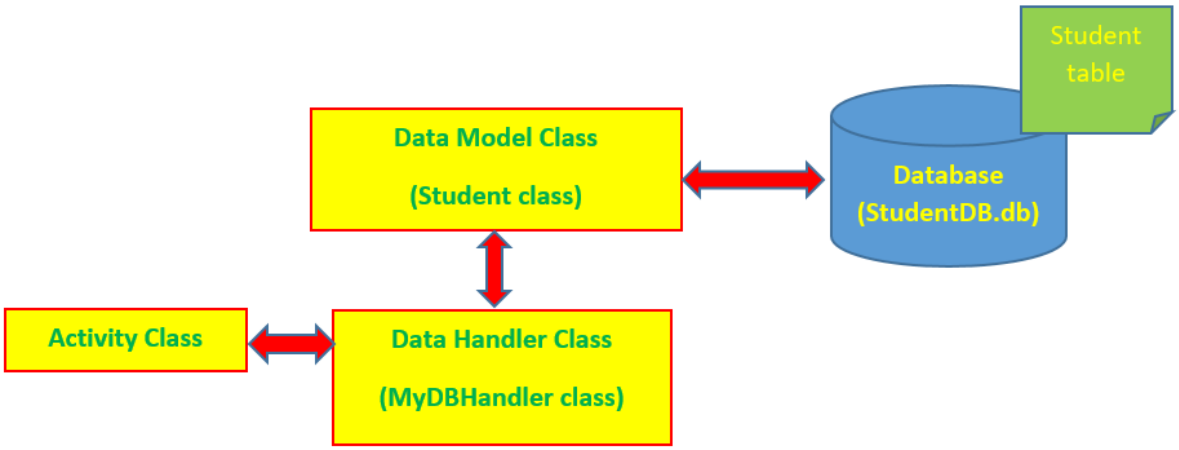
We will create a database Android application that has a UI as follows:



My application will interact with a database named StudentDB.db, which contains a single table named Student. The Student table schema will look like this:

Column	DataType
StudentID	Integer/PK/AutoIncrement
StudentName	Text

The application will consist of an activity and a database handler class (MyDBHandler class). The database handler will be a subclass of SQLiteOpenHelper and will provide an abstract layer between the underlying SQLite database and the activity class. A third class (Student class) will need to be implemented to hold the database entry data as it is passed between the activity and the handler. My application model can be shown in the following figure:

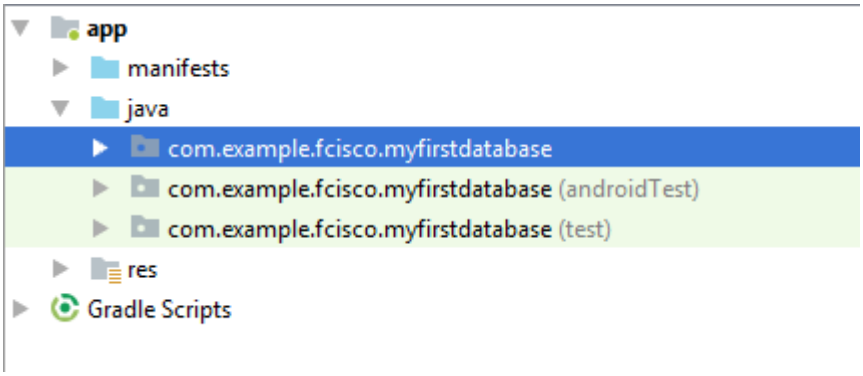


Data Model Class

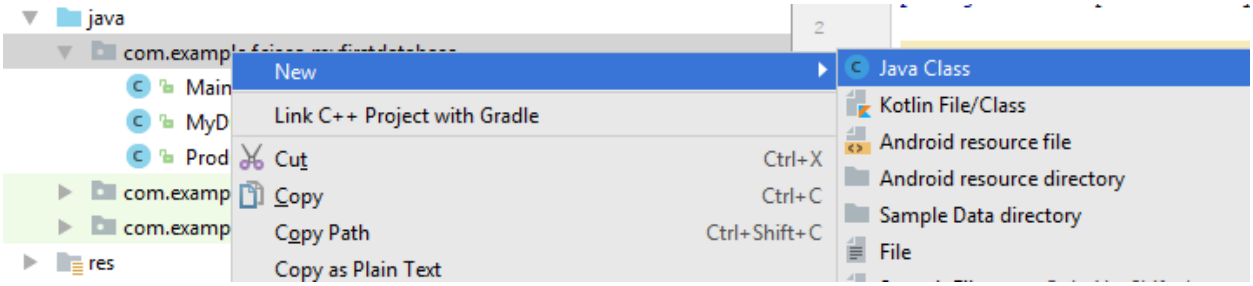
The Student class contains fields, constructors, and properties as follows:

Student Class
Fields
Student ID StudentName
Constructors
Student()
Student(id, studentname)
Properties
setID(), getID()
setStudentName(), getStudentName()

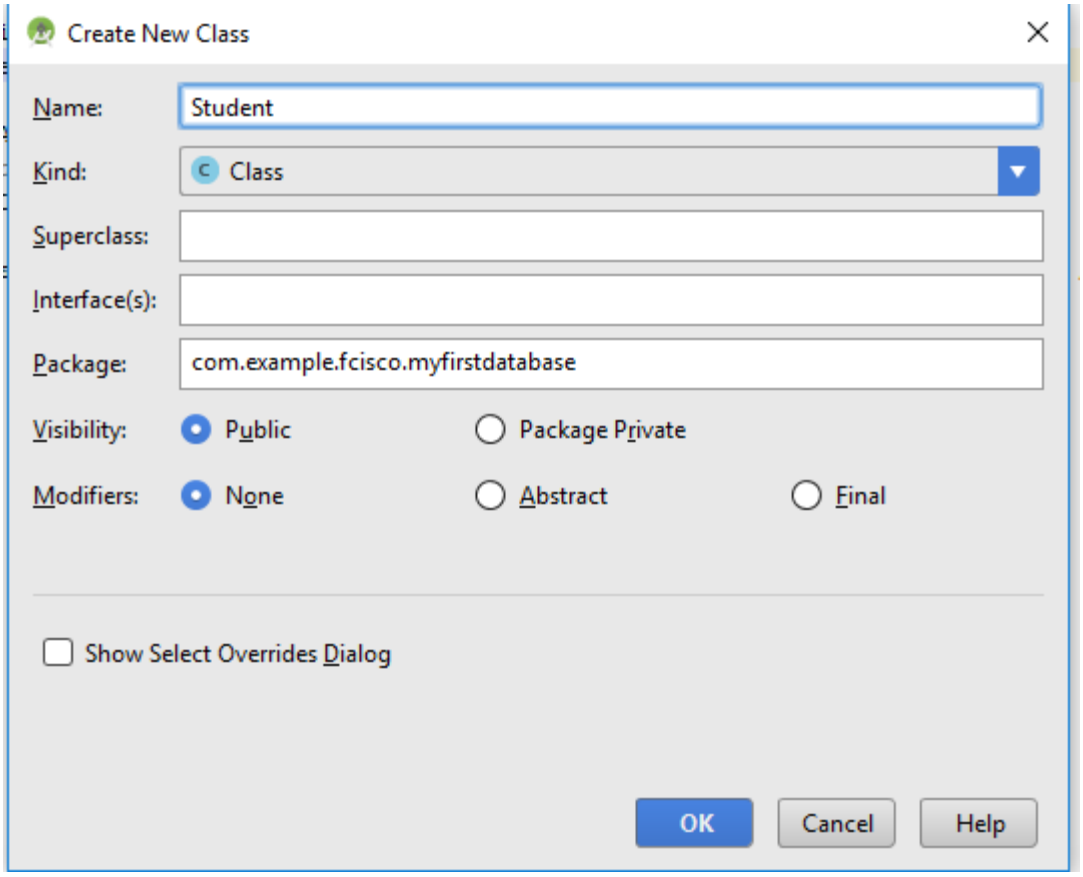
We create the Student class in the Android Studio by selecting app > java.



Right-click the myfirstdatabase package and selecting New > Java Class.



Type Student in the Name item, maintain the default options, and click the OK button:



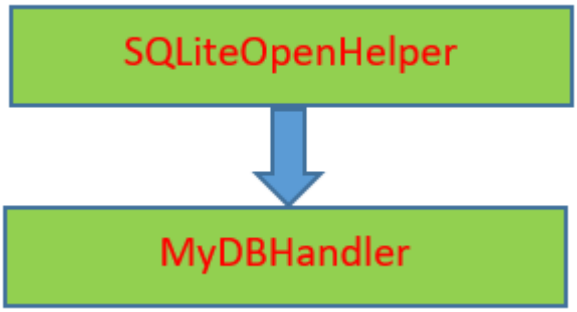
Adding the following lines of code for the Student class:

```
public class Student {
 // fields
 private int studentID;
 private String studentName;
 // constructors
 public Student() {}
 public Student(int id, String studentname) {
   this.studentID = id;
   this.studentName = studentname;
  }
```

```
 // properties
public void setID(int id) {
 this.studentID = id;
}
public int getID() {
 return this.studentID;
}
public void setStudentName(String studentname) {
 this.studentName = studentname;
}
public String getStudentName() {
 return this.studentName;
}
}
```

Data Handler Class

The database handler class is a subclass of SQLiteOpenHelper class, named MyDBHandler, as in the following figure:



The MyDBHandler class contains fields, constructors, and methods, as follows:

MyDBHandler class
Fields
DATABASE_VERSION
DATABASE_NAME
COLUMN_ID
COLUMN_NAME
Constructor
MyDBHandler()
Methods
onCreate()
onUpgrade()
addHandler()
deleteHandler()
updateHandler()
findHandler()
loadHandler()

Steps for creating the MyDBHandler class like the Student class and its code can look like this:

```
public class MyDBHandler extends SQLiteOpenHelper {
 //information of database
 private static final int DATABASE_VERSION = 1;
 private static final String DATABASE_NAME = "studentDB.db";
 public static final String TABLE_NAME = "Student";
 public static final String COLUMN_ID = "StudentID";
 public static final String COLUMN_NAME = "StudentName";
 //initialize the database
 public MyDBHandler(Context context, Stringname, SQLiteDatabase.CursorFactoryfactory,
intversion) {
  super(context, DATABASE_NAME, factory, DATABASE_VERSION);
 }
 @Override
 public void onCreate(SQLiteDatabase db) {}
 @Override
 public void onUpgrade(SQLiteDatabase db, int i, int i1) {}
 public String loadHandler() {}
 public void addHandler(Student student) {}
 public Student findHandler(String studentname) {}
 public boolean deleteHandler(int ID) {}
 public boolean updateHandler(int ID, String name) {}
}
```

We must also use import statements, as follows:

```
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.content.Context;
import android.content.ContentValues;
import android.database.Cursor;
```

Initialize the Database

The database can be initialized in the constructor of the MyDBHandler class. The code of this constructor looks like this:

```
public MyDBHandler(Context context, Stringname,
 SQLiteDatabase.CursorFactoryfactory, intversion) {
 super(context, DATABASE_NAME, factory, DATABASE_VERSION);
}
```

Create the Student Table

The SQL statement creates a table:

```
CREATE TABLE table_name(
 column1 datatype,
 column2 datatype,
 column3 datatype,
 ....
);
```

We need to convert the SQL statement to a string and implement it by using the execSQL() method of a SQLiteDatabase object. All of these statements will be put inside the onCreate method of the handler class, as follows:

```
public void onCreate(SQLiteDatabase db) {
 String CREATE_TABLE = "CREATE TABLE" + TABLE_NAME + "(" + COLUMN_ID +
  "INTEGER PRIMARYKEY," + COLUMN_NAME + "TEXT )";
 db.execSQL(CREATE_TABLE);
}
```

Our application can load all of the students from the database, add a new student to the database, remove a student from the database, find a student from the database and modify the information of a student from the database. Therefore, we need to add corresponding methods to the handler class.

Load Data

To load data, we use a SQL query statement:

```
SELECT * FROM table_name;
```

The result of above SQL statement is a table. We use the rawQuery() method of a SQLiteDatabase object to implement SQL statement and display result via a Cursor object. The following code will demonstrate the loadHandler method:

```
public String loadHandler() {
 String result = "";
 String query = "Select*FROM" + TABLE_NAME;
 SQLiteDatabase db = this.getWritableDatabase();
 Cursor cursor = db.rawQuery(query, null);
 while (cursor.moveToNext()) {
  int result_0 = cursor.getInt(0);
  String result_1 = cursor.getString(1);
  result += String.valueOf(result_0) + " " + result_1 +
   System.getProperty("line.separator");
 }
 cursor.close();
 db.close();
 return result;
}
```

Add a New Record

To add a new record to the database, we must use the ContentValues object with the put() method that is used to assign data to ContentsValues object and then use insert() method of SQLiteDatabase object to insert data to the database. The addHandler method can look like this:

```
public void addHandler(Student student) {
 ContentValues values = new ContentValues();
 values.put(COLUMN_ID, student.getID());
 values.put(COLUMN_NAME, student.getStudentName());
 SQLiteDatabase db = this.getWritableDatabase();
 db.insert(TABLE_NAME, null, values);
 db.close();
}
```

Controls used in above UI include:

Controls	text atrribute	id attribute
EditText	Student ID	studentid
EditText	Student Name	studentname
Button	LOAD DATA	btnload
Button	ADD	btnadd
Button	FIND	btnfind
Button	DELETE	btndelete
Button	UPDATE	btnupdate
TextView		lst

19.3 Short Questions:

Q.No What is View group in Android?

Q.No 2 What is a Content Provider in Android?

Q.No 3 What is Container in Android?

Q.No 4 What is a ADB and ANR in Android?

20. Experiment 11 (Beyond Curricula)

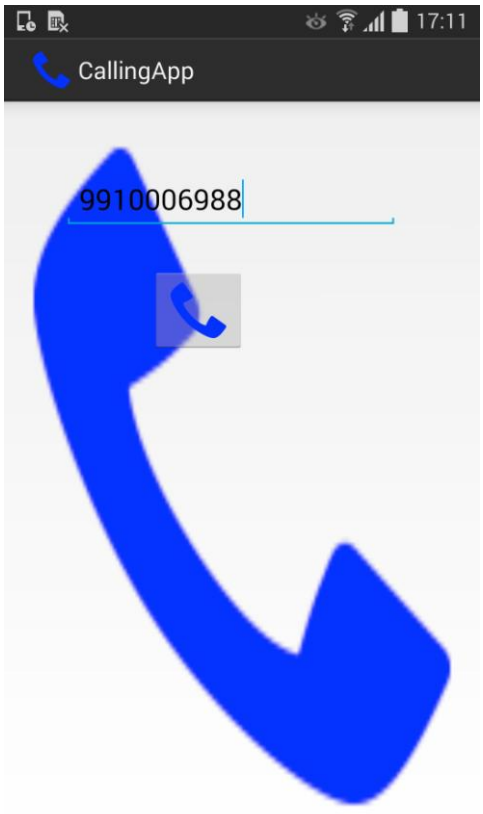
AIM: Write a program to design a calling application.

20.1 Source Code:

```
package com.example.callingapp;
import android.net.Uri;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.View;
import
android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import
android.widget.ImageButton;

public class CallActivity extends Activity implements

        OnClickListener { EditText et1;
        ImageButton bt1;

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

                et1 = (EditText) findViewById(R.id.editText1);
                bt1 = (ImageButton) findViewById(R.id.imageButton1);
                bt1.setOnClickListener(this);

        }

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

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

```
            // Call a Number

            String number = et1.getText().toString();

            Intent call = new Intent(Intent.ACTION_CALL);
            call.setData(Uri.parse("tel:" + number));
            startActivity(call);


        }

}
```

Output:



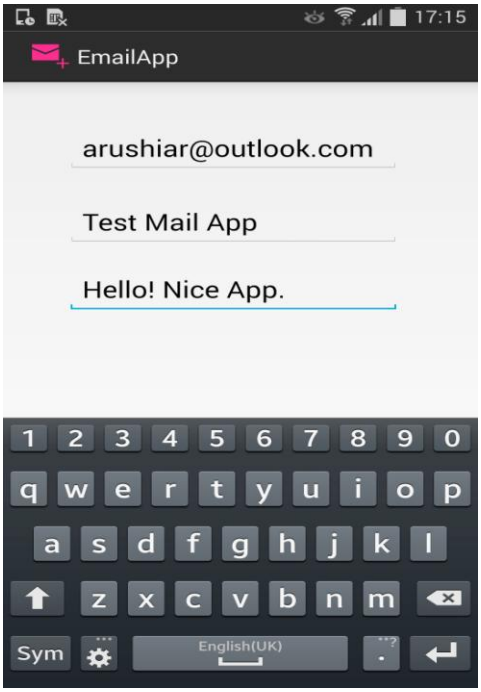
21. Experiment 12 (Beyond Curricula)

AIM: Write a program to design a Mailing application.

a. Source Code:

```
package
com.example.emailapp;
import android.os.Bundle;
import android.app.Activity;
import
android.content.Intent;
import android.view.Menu;
import android.view.View;
import
android.view.View.OnClickListener;
import android.widget.EditText;
import
android.widget.ImageButton;
import android.widget.Toast;

public class EmailActivity extends Activity implements

        OnClickListener { EditText et1, et2, et3;
        ImageButton ib1;

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

                et1                =                (EditText)
                findViewById(R.id.editText1);    et2    =
                (EditText)
                findViewById(R.id.editText2);    et3    =
                (EditText)
                findViewById(R.id.editText3);

                ib1 = (ImageButton) findViewById(R.id.imageButton1);

                ib1.setOnClickListener(this);

        }

        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
```

```
            // Inflate the menu; this adds items to the action bar if it is present.
            getMenuInflater().inflate(R.menu.email,    menu);
            return true;
        }
        @Override
       public void onClick(View v) {
            // TODO Auto-generated method stub

            String emailid = et1.getText().toString();
            String[] emails = { emailid };
            String subject = et2.getText().toString();
            String message = et3.getText().toString();

            Intent email = new Intent(Intent.ACTION_SEND);
            email.setType("*/*");
            email.putExtra(Intent.EXTRA_EMAIL, emails);
            email.putExtra(Intent.EXTRA_SUBJECT, subject);
            email.putExtra(Intent.EXTRA_TEXT, message);
            startActivity(email);
            Toast.makeText(this, "Mail Sent to Email Provider",
Toast.LENGTH_SHORT).show();
        }
}
```



22. Experiment 13 (Beyond Curricula)

AIM: Write a program to display the latitude and longitude of a user

Program

MainActivity.java

```
package com.gpslocationtracker;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import
android.view.MenuItem;
import
android.widget.TextView;

public class MainActivity extends ActionBarActivity

        { TextView textview;

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

                 // check if GPS enabled
    GPSTracker gpsTracker = new GPSTracker(this);

    if (gpsTracker.canGetLocation())
    {
      String stringLatitude = String.valueOf(gpsTracker.latitude);
      textview = (TextView)findViewById(R.id.fieldLatitude);
      textview.setText(stringLatitude);


      String stringLongitude =
      String.valueOf(gpsTracker.longitude); textview =
      (TextView)findViewById(R.id.fieldLongitude);
      textview.setText(stringLongitude);
    }
    else
    {
      // can't get location
      // GPS or Network is not enabled
      // Ask user to enable GPS/network in settings
```

```
            gpsTracker.showSettingsAlert();
        }

         }
        @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;
        }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
                // Handle action bar item clicks here. The action bar will
                // automatically handle clicks on the Home/Up button, so long
                // as you specify a parent activity in
                AndroidManifest.xml. int id = item.getItemId();
                if (id == R.id.action_settings) {
                        return true;
                }
                return super.onOptionsItemSelected(item);
        }
}
```

a. GPSTracker.java

```
package com.gpslocationtracker;
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 {

        @Override
        public void onLocationChanged(Location location) {
                // TODO Auto-generated method stub
```

```
        }

        private final Context mContext;

    //flag for GPS Status
    boolean isGPSEnabled = false;

    //flag for network status
    boolean isNetworkEnabled = false;

    boolean canGetLocation = false;

    Location location;
    double latitude;
    double longitude;

    //The minimum distance to change updates in metters
    private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES = 10; //10
    metters

    //The minimum time beetwen 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;
```

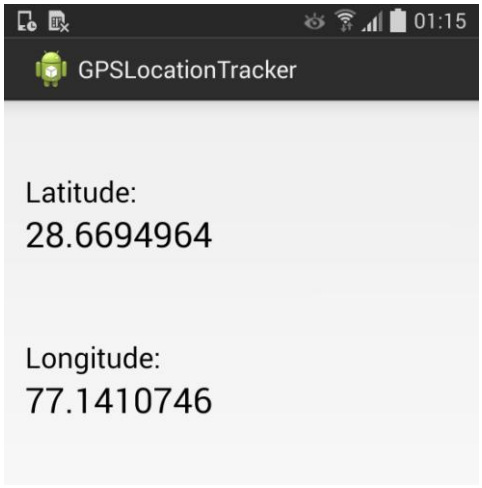
```
        //First get location from Network Provider if
        (isNetworkEnabled)
        {
          locationManager.requestLocationUpdates(
              LocationManager.NETWORK_PROVIDER,
              MIN_TIME_BW_UPDATES,
              MIN_DISTANCE_CHANGE_FOR_UPDATES,
              this);

          Log.d("Network", "Network"); if

          (locationManager != null)
          {
            location =
locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
            updateGPSCoordinates();
          }
        }

        //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);
              updateGPSCoordinates();
            }
          }
        }
      }
    }
    catch (Exception e)
    {
      //e.printStackTrace();
      Log.e("Error : Location", "Impossible to connect to LocationManager", e);
```

```
    }

    return location;
}

public void updateGPSCoordinates()
{
  if (location != null)
  {
    latitude = location.getLatitude(); longitude =
    location.getLongitude();
  }
}

public void stopUsingGPS()
{
  if (locationManager != null)
  {
    locationManager.removeUpdates(GPSTracker.this);
  }
}

public double getLatitude()
{
  if (location != null)
  {
    latitude = location.getLatitude();
  }

  return latitude;
}

public double getLongitude()
{
  if (location != null)
  {
    longitude = location.getLongitude();
  }

  return longitude;
}

public boolean canGetLocation()
{
  return this.canGetLocation;
```

```
 }

 public void showSettingsAlert()
 {
   AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);
   //Setting Dialog Title
   alertDialog.setTitle("GPSAlertDialogTitle");

   //Setting Dialog Message
   alertDialog.setMessage("GPSAlertDialogMessage");

   //On Pressing Setting button
   alertDialog.setPositiveButton("Settings", new DialogInterface.OnClickListener()
   {
     @Override
     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()
   {
     @Override
     public void onClick(DialogInterface dialog, int which)
     {
       dialog.cancel();
     }
   });

   alertDialog.show();
 }

       @Override
       public void onStatusChanged(String provider, int status, Bundle extras) {
               // TODO Auto-generated method stub

       }

       @Override
       public void onProviderEnabled(String provider) {
               // TODO Auto-generated method stub
```

```
	}

	@Override
	public void onProviderDisabled(String provider) {
		// TODO Auto-generated method stub
	  }


	  @Override

	  public IBinder onBind(Intent intent) {

			// TODO Auto-generated method stub

			return null;

	  }

}
```

OUTPUT



23. Experiment 14 (Beyond Curricula)

AIM: Write a program to design Multiple Activity Class.

23.1 Source Code:

FirstActivity.java

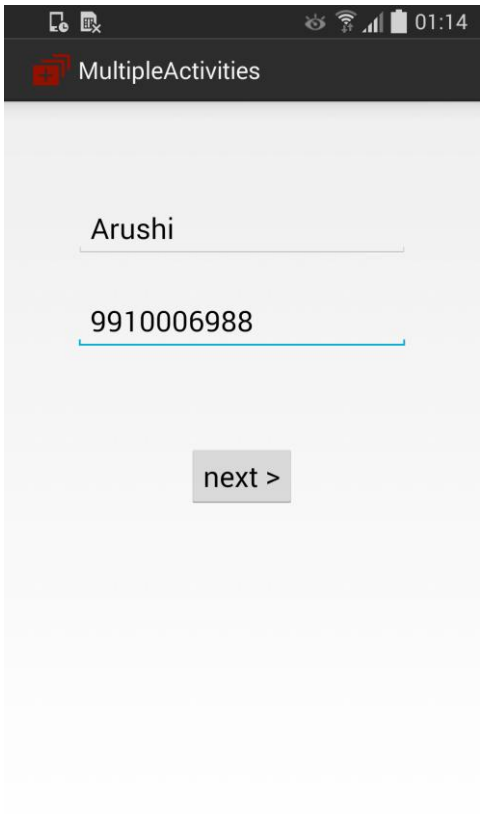
```
package com.example.multipleactivities;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class FirstActivity extends Activity implements OnClickListener {

	EditText et1, et2;
	Button bt1;
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_first);

		et1 = (EditText) findViewById(R.id.editText1);
		et2 = (EditText) findViewById(R.id.editText2);
		bt1 = (Button) findViewById(R.id.button_next);

		bt1.setOnClickListener(this);
	}
	@Override
	public boolean onCreateOptionsMenu(Menu menu) {
		// Inflate the menu; this adds items to the action bar if it is present.
		getMenuInflater().inflate(R.menu.first, menu);
		return true;
	}
	@Override
	public void onClick(View v) {
		// TODO Auto-generated method stub

		String name = et1.getText().toString();
		String number = et2.getText().toString();
```

```
		if(name.equals("") || number.equals("")){
			Toast.makeText(this, "Please Enter Values",
Toast.LENGTH_SHORT).show();

		}else {
			Intent next = new Intent(this, SecondActivity.class);
			next.putExtra("NAME", name);
			next.putExtra("NUMBER", number);
			startActivity(next);
			finish();
		}
	}
}
```

a. SecondActivity.java

```
package com.example.multipleactivities;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

public class SecondActivity extends Activity implements OnClickListener {

	TextView tv1, tv2;
	Button bt1, bt2;

	public void onCreate(Bundle savedInstanceState){

		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_second);

		tv1 = (TextView) findViewById(R.id.textView_name);
		tv2 = (TextView) findViewById(R.id.textView_number);

		bt1 = (Button) findViewById(R.id.button_main);
		bt2 = (Button) findViewById(R.id.button_exit);

		bt1.setOnClickListener(this);
		bt2.setOnClickListener(this);

		Intent receive = getIntent();
		String name = receive.getStringExtra("NAME");
```

```
            String number = receive.getStringExtra("NUMBER");
            tv1.setText(name);
            tv2.setText(number);

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

            case R.id.button_main:
                Intent main = new Intent(this, FirstActivity.class); startActivity(main);
                finish(); break;

            case R.id.button_exit:
                Toast.makeText(this, "Bye !", Toast.LENGTH_SHORT).show(); finish();
                break;
            }
        }
}
```


01:14
MultipleActivities
Arushi
9910006988
next >


01:15
MultipleActivities
9910006988
Arushi
< Main
Exit >

24. Experiment 15 (Beyond Curricula)

Aim: Write a program to design a contacts application.

a. Source Code:

DatabaseActivity.java

```
package com.example.contactsapp;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class DatabaseActivity extends Activity implements OnClickListener {

	EditText et1, et2;
	Button bt1, bt2, bt3;


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

		et1 = (EditText) findViewById(R.id.editText1);
		et2 = (EditText) findViewById(R.id.editText2);
		bt1 = (Button) findViewById(R.id.button_save);
		bt2 = (Button) findViewById(R.id.button_view);
		bt3 = (Button) findViewById(R.id.button_exit);


		bt1.setOnClickListener(this);
		bt2.setOnClickListener(this);
		bt3.setOnClickListener(this);

	}

	@Override
	public boolean onCreateOptionsMenu(Menu menu) {
		// Inflate the menu; this adds items to the action bar if it is present.
```

```
		getMenuInflater().inflate(R.menu.database, menu);
		return true;
	}

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

		switch(v.getId()){
		case R.id.button_save:

			String name = et1.getText().toString();
			String number = et2.getText().toString();

			DBHelper dbh = new DBHelper(this);
			dbh.open();
			dbh.insert(name, number);
			dbh.close();
			Toast.makeText(this, "Saved", Toast.LENGTH_SHORT).show();

			break;

		case R.id.button_view:

			Intent view = new Intent(this, ViewActivity.class);
			startActivity(view);
			finish();
			break;

		case R.id.button_exit:
			finish();
			break;
		}

	}

}
```

b. DBHelper.java

```
package com.example.contactsapp;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteDatabase.CursorFactory;
import android.database.sqlite.SQLiteOpenHelper;
```

```
public class DBHelper {

        public static final String DB_NAME= "Contacts.db"; public
        static final String DB_TABLE = "Contacts"; public static final
        int DB_VERSION = 2;
        public static final String DB_CNAME = "Name"; public static
        final String DB_CNUMBER = "Number";

        public static final String createQuery = "CREATE TABLE " + DB_TABLE + "(" + DB_CNAME + "
TEXT, " + DB_CNUMBER + " TEXT);" ;

        DBHelperInternal dbhI; Context

        context;
        private SQLiteDatabase db;

        public DBHelper(Context c){ context = c;
        }
        public DBHelper open() {
                // TODO Auto-generated method stub

                dbhI = new DBHelperInternal(context); db =
                dbhI.getWritableDatabase(); return this;
        }
        public void insert(String name, String number) {
                // TODO Auto-generated method stub

                ContentValues cv = new ContentValues();
                cv.put(DB_CNAME, name); cv.put(DB_CNUMBER,
                number); db.insert(DB_TABLE, null, cv);
        }
        public void close() {
                // TODO Auto-generated method stub db.close();
        }
        private class DBHelperInternal extends SQLiteOpenHelper{
                public DBHelperInternal(Context context) { super(context,
                        DB_NAME, null, DB_VERSION);
                        // TODO Auto-generated constructor stub
                }

                @Override
                public void onCreate(SQLiteDatabase db) {
                        // TODO Auto-generated method stub

                        db.execSQL(createQuery);
                }
                @Override
                public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
```

```
                // TODO Auto-generated method stub
                db.execSQL("DROP TBALE " + DB_TABLE + ";");
                db.execSQL(createQuery);
            }
        }

        public String read() {
            // TODO Auto-generated method stub String
            result = "";

            String[] columns = { DB_CNAME, DB_CNUMBER };
            Cursor c = db.query(DB_TABLE, columns, null, null, null, null, null); int
            nameI = c.getColumnIndex(DB_CNAME);
            int numberI = c.getColumnIndex(DB_CNUMBER);

            for(c.moveToFirst(); !c.isAfterLast(); c.moveToNext()){

                result = result + c.getString(nameI) + " - " + c.getString(numberI) + "\n";

            }
            return result;
        }
}
```

c. ViewActivity.java

```
package com.example.contactsapp;
import android.app.Activity; import
android.content.Intent; import
android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener; import
android.widget.Button;
import android.widget.TextView;

public class ViewActivity extends Activity implements OnClickListener {

        TextView tv;
        Button bt1, bt2;

        public void onCreate(Bundle savedInstanceState){

            super.onCreate(savedInstanceState); setContentView(R.layout.view_activity);

            tv = (TextView) findViewById(R.id.textView_view); bt1 =
            (Button) findViewById(R.id.button_home); bt2 = (Button)
            findViewById(R.id.button_exit1);
```

```
                bt1.setOnClickListener(this);
                bt2.setOnClickListener(this);

                // Receive Database Values DBHelper dbh =
                new DBHelper(this); dbh.open();
                String result = dbh.read();
                dbh.close();

                tv.setText(result);

        }
        @Override
        public void onClick(View v) {
                // TODO Auto-generated method stub
                switch(v.getId()){
                case R.id.button_home:

                        Intent home = new Intent(this, DatabaseActivity.class); startActivity(home);
                        finish(); break;

                case R.id.button_exit1:
                        finish(); break;

                }
        }
}
```