Jaipur Engineering College and Research Center, Jaipur

Department of Information Technology

Rajasthan Technical University, KOTA)

LAB MANUAL

Lab Name	**:** Advanced JAVA Lab
Lab Code	**:** 5IT4-24
Branch	**:** Information Technology
Year	**:** 3rd Year

Prepared by:

Ms. Shweta Saxena

(V SEM)

Department of Information Technology
Session: 2020-21
LAB MANUAL

Subject Code: 5IT4-24
Subject: Advanced Java Lab

INDEX

JAIPUR ENGINEERING COLLEGE AND RESEARCH CENTER

VISION & MISSION OF INSTITUTE

VISION:

To become a renowned centre of outcome based learning, and work towards academic, professional, cultural and social enrichment of the lives of individuals and communities.

MISSION:

M1: Focus on evaluation of learning outcomes and motivate students to inculcate research aptitude by project based learning.

M2: Identify, based on informed perception of Indian, regional and global needs, areas of focus and provide platform to gain knowledge and solutions.

M3: Offer opportunities for interaction between academia and industry.

M4: Develop human potential to its fullest extent so that intellectually capable and imaginatively gifted leaders can emerge in a range of professions.

JAIPUR ENGINEERING COLLEGE AND RESEARCH CENTER

Department of Information Technology

1. VISION & MISSION

VISION:

To established outcomes based excellence in teaching, learning and commitment to support IT industry.

MISION:

M1: To provide outcome based education

M2: To provide fundamental & intellectual knowledge with essential skills to meet current and future need of IT Industry across the globe

M3: To inculcate the philosophy of continues learning, ethical values & Social Responsibility.

PEO

1. To enrich students with fundamental knowledge, effective computing, problem solving and communication skills enable them to have successful career in Information Technology.
2. To enable students in acquiring Information Technology's latest tools, technologies and management principles to give them an ability to solve multidisciplinary engineering problems.
3. To impart students with ethical values and commitment towards sustainable development in collaborative mode.
4. To imbibe students with research oriented and innovative approaches which help them to identify, analyze, formulate and solve real life problems and motivates them for lifelong learning.
5. To empower students with leadership quality and team building skills that prepare them for employment, entrepreneurship and to become competent professionals to serve societies and global needs.

PROGRAM OUTCOMES

1. **Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems in IT.
2. **Problem analysis:** Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences in IT.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations using IT.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions using IT.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations in IT.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice using IT.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development in IT.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice using IT.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings in IT.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project Management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage IT projects and in multidisciplinary environments.
12. **Life –long Learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological changes needed in IT.

PROGRAM SPECIFIC OUTCOMES

PSO1: Graduates of the program would be able to develop mobile and web based IT solutions for real time problems.

PSO2: Graduates of the program would be able to apply the concepts of artificial intelligence, machine learning and deep learning.

5IT4-24: Advanced Java Lab

S.No.	List of Experiments
1	Introduction To Swing, MVC Architecture, Applets, Applications and Pluggable Look and Feel, Basic swing components : Text Fields, Buttons, Toggle Buttons, Checkboxes, and Radio Buttons
2	Java database Programming, java.sql Package, JDBC driver, Network Programming With java.net Package, Client and Server Programs, Content And Protocol Handlers
3	RMI architecture, RMI registry, Writing distributed application with RMI, Naming services, Naming And Directory Services, Overview of JNDI, Object serialization and Internationalization
4	J2EE architecture, Enterprise application concepts, n-tier application concepts, J2EE platform, HTTP protocol, web application, Web containers and Application servers
5	Server side programming with Java Servlet, HTTP and Servlet, Servlet API, life cycle, configuration and context, Request and Response objects, Session handling and event handling, Introduction to filters with writing simple filter application
6	JSP architecture, JSP page life cycle, JSP elements, Expression Language, Tag Extensions, Tag Extension API, Tag handlers, JSP Fragments, Tag Files, JSTL, Core Tag library, overview of XML Tag library, SQL Tag library and Functions Tag library

Software Used:

1)- JDK 8.0

2)- MS access Database

Course Outcomes

Subject and Code: Advanced JAVA Lab (5IT4-24) [L/T/P – 0/0/2]
Semester: V

Course Outcomes (COs): Graduates will be able

1. To develop a deep understanding of JAVA programming and development.
2. To design and develop GUI applications using SWING and event handling.
3. To identify how to access database through Java programs, using java database connectivity (JDBC).
4. To design and create WebPages using JAVA servlets and JSP.

CO-PO Mapping

SEM	LAB SUBJECT WITH CODE	L/P/T	COs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
VI	Advanced Java Lab (5IT4-24)	2/0/0	1. To develop a deep understanding of JAVA programming and development.	H	H	H	M	H	M	M	L	H	L	H	M
			2. To design and develop GUI applications using SWING and event handling.	H	M	H	M	M	M	L	L	H	L	M	M
			3. To identify how to access database through Java programs, using java database connectivity (JDBC).	H	H	H	M	M	M	L	L	H	L	M	M
			4. To design and create WebPages using JAVA servlets and JSP.	H	H	H	M	M	M	L	L	H	L	M	M

CO/PSO Mapping

Course Outcome	PSO1	PSO2
To develop a deep understanding of JAVA programming and development.	H	L
To design and develop GUI applications using SWING and event handling.	H	L
To identify how to access database through Java programs, using java database connectivity (JDBC).	H	L
To design and create WebPages using JAVA servlets and JSP.	H	L

Introduction about Lab: Brief Description, Applications

Brief Description: If technology touches life, chances are, so does Java technology. Invented by Sun Microsystems in 1995, Java technology has become the essential ingredient of the digital experience for hundreds of millions of people in all walks of life, all over the planet.

Java software powers the onboard computers in toys, cars, planes, rockets, and even the NASA Mars Rover. It brings interactivity to the Internet, real- time graphics to television, instant imaging to cameras, and multi-player games to mobile phones and desktop PCs. It connects the largest enterprises and smallest businesses to their employees, customers, and data. And it secures the vast majority of electronic transactions in retail, finance, government, science, and medicine. In short, Java technology goes everywhere you go.

It's no wonder that Java technology has become the most powerful force in software and the most prevalent software in technology. In fact, it is the software of choice for more software engineers than any other brand of software.

Java is the most efficient, fast, secure, animated, compatible, and reliable software.

Top 11 Applications of Java with Real-world Examples

Below is the Java applications list:
- Desktop GUI Applications
- Mobile Applications
- Enterprise Applications

- Scientific Applications
- Web-based Applications
- Embedded Systems
- Big Data Technologies
- Distributed Applications
- Cloud-based Applications
- Web servers and Application servers
- Software Tools
- Gaming Applications

List of Experiments :

Exp:- 1	Swapping program in Java. WAP to print patterns in Java.		
Exp:- 2	Write programs to implement concepts of JAVA SWING.		
Exp:-3	A Program to execute select query using JDBC		
Exp:-4	A simple servlet that just generates plain text		
Exp:-5	A Program which displays cookie id suing servlets.		
Exp:-6	A program for basic arithmetic functions using JSP		
Exp:-7	A Program to display a String		
Exp:-8	A program to generates plain text using Java Beans		
Exp:-9	Write programs to implement concepts of RMI architecture.		

Experiment No.: 1

What is Java?

Java is a popular programming language, created in 1995.

It is owned by Oracle, and more than **3 billion** devices run Java.

It is used for:

- Mobile applications (specially Android apps)
- Desktop applications
- Web applications
- Web servers and application servers
- Games
- Database connection
- And much, much more!

Advanced Java" is everything that goes beyond that - most importantly the APIs defined in Java Enterprise Edition, i.e. Servlet programming, Web Services, the Java Persistence API, etc

Difference between Java & Advanced Java

CORE JAVA	ADVANCED JAVA
To develop general purpose applications.	To develop online application and mobile application.
Without Core Java no one can develop any advanced java applications.	Whereas advanced java only deals with some specialization like Database, DOM(web), networking etc.
OOP, data types, operators, functions, loops, exception handling, threading etc.	Apart from the core java parts it has some specific sections like database connectivity, web services, servlets etc.

It uses only one tier architecture that is why it is called as ‘stand alone’ application.	It uses two tier architecture i.e. client side architecture and server side or backend architecture.
Core java programming covers the swings, socket, awt, thread concept, collection object and classes.	Advance java is used for web based application and enterprise application.

Swapping program in Java

```
import java.util.*;   //import whole package

import java.util.Scanner;          //import the scanner class

class Swap_With {

   public static void main(String[] args)  // Main Function


 {

      int x, y, t;// x and y are to swap

      Scanner sc = new Scanner (System.in);    //Create A Scanner object


      System.out.println ("Enter the value of X and Y");


      x = sc.nextInt();  // Taking Input from user in integer

      y = sc.nextInt();

      System.out.println ("before swapping numbers: "+x +"  "+ y);

      /*swapping */

      t = x;

      x = y;

      y = t;

      System.out.println ("After swapping: "+x +"   ”+ y);

      System.out.println ( );
```

```
  }
```

```
Enter the value of X and Y
2
3
Before swapping numbers: 2   3
After swapping: 3   2
```

1) public String nextLine(): For getting input String
2)public int nextInt(): For integer input
3) public float nextFloat(): For float input

WAP to print pattern:

```
1
1 0
1 0 1
1 0 1 0
1 0 1 0 1
1 0 1 0 1 0
```

```
class Pattern1
{
public static void main(String args[])
{
int i, j;
for(i=0; i<6; i++)
{
 for(j=0; j<i; j++)
 {
        if(j%2==0)
        System.out.println(1);
        else
        System.out.println(0);
}
System.out.println();
}
}
```

Experiment No.: 2

What is Swing in Java?

SWING IN JAVA is a Graphical User Interface (GUI) toolkit that includes a rich set of widgets. It is a part of Java Foundation Classes(JFC), which is an API for Java programs that provide GUI. **Swing** includes packages that let you make a sophisticated set of GUI components for your Java applications and it is platform-independent.

The Swing library is built on top of the Java Abstract Widget Toolkit (**AWT**), an older, platform dependent GUI toolkit. You can use the Java GUI components like button, textbox, etc. from the library and do not have to create the components from scratch.

Java Swing class Hierarchy Diagram

https://www.guru99.com/images/uploads/2012/06/java-swing-class-hierarchy.jpg

What is a container class?

Container classes are classes that can have other components on it. So for creating a GUI, we need at least one container object. There are 3 types of containers.

1. **Panel**: It is a pure container and is not a window in itself. The sole purpose of a Panel is to organize the components on to a window.
2. **Frame**: It is a fully functioning window with its title and icons.
3. **Dialog**: It can be thought of like a pop-up window that pops out when a message has to be displayed. It is not a fully functioning window like the Frame.

What is GUI in Java?

GUI (Graphical User Interface) IN JAVA gives programmers an easy-to-use visual experience to build Java applications. It is mainly made of graphical components like buttons, labels, windows, etc. through which the user can interact with the applications. Swing GUI in Java plays an important role in building easy interfaces.

1.1.1

1.1.2 Difference between AWT and Swing

There are many differences between java awt and swing that are given below.

No.	Java AWT	Java Swing
1)	AWT components are **platform-dependent**.	Java swing components are **platform-independent**.
2)	AWT components are **heavyweight**.	Swing components are **lightweight**.
3)	AWT **doesn't support pluggable look and feel**.	Swing **supports pluggable look and feel**.
4)	AWT provides **less components** than Swing.	Swing provides **more powerful components** such as tables, lists, scrollpanes, colorchooser, tabbedpane etc.
5)	AWT **doesn't follows MVC** (Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view.	Swing **follows MVC**.

MVC architecture:

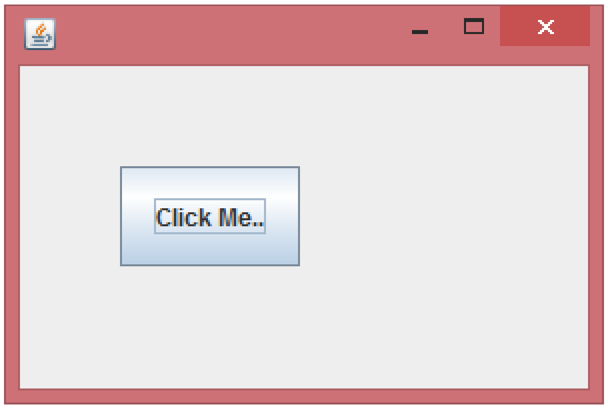
Java Swing MVC, The MVC pattern is a model of how a user interface can be structured. Therefore it defines the following 3 elements:

- Model- that represents the data for the application.
- View- that is the visual representation of that data.
- Controller- that takes user input on the view and translates that to changes in the model.

Program : To make a GUI platform for showing button using swing .

```
 import java.awt.event.*;
import javax.swing.*;
public class Button1{
public static void main(String[] args) {
    JFrame f=new JFrame("Button Example");
```

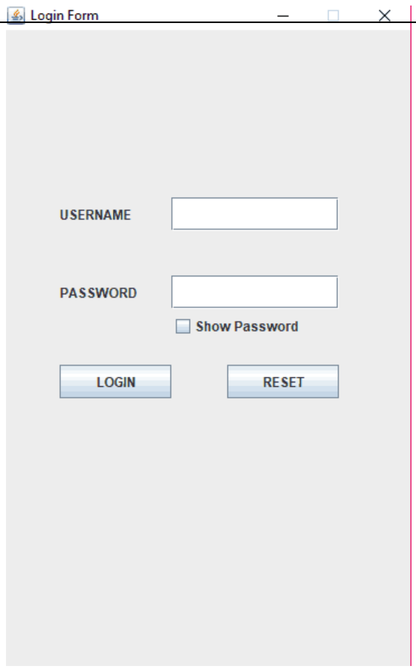
```
        final JTextField tf=new JTextField();
        tf.setBounds(50,50, 150,20);
        JButton b=new JButton("Click Me");
        b.setBounds(50,100,95,30);
        b.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e){
            tf.setText("Welcome to Javatpoint.");
          }
        });
        f.add(b);f.add(tf);
        f.setSize(400,400);
        f.setLayout(null);
        f.setVisible(true);
    }
    }
```



Experiment No.: 3

Program : Aim to provide the login form using swing

```
import javax.swing.*;
public class LoginView {
	public static void main(String[] args)
	 {
		JFrame frame = new JFrame("Demo application");
		frame.setSize(300, 150);
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		JPanel panel = new JPanel();
		frame.add(panel);
		placeComponents(panel);
		frame.setVisible(true);
	}

	public static void placeComponents(JPanel panel) {

		panel.setLayout(null);

		JLabel userLabel = new JLabel("Username");
		userLabel.setBounds(10, 10, 80, 25);
		panel.add(userLabel);

		JTextField userText = new JTextField(20);
		userText.setBounds(100, 10, 160, 25);
		panel.add(userText);

		JLabel passwordLabel = new JLabel("Password");
		passwordLabel.setBounds(10, 40, 80, 25);
		panel.add(passwordLabel);

		JPasswordField passwordText = new JPasswordField(20);
		passwordText.setBounds(100, 40, 160, 25);
		panel.add(passwordText);

		JButton loginButton = new JButton("login");
		loginButton.setBounds(10, 80, 80, 25);
		panel.add(loginButton);

		JButton registerButton = new JButton("register");
		registerButton.setBounds(180, 80, 80, 25);
		panel.add(registerButton);
	}

}
```


Login Form
USERNAME
PASSWORD
Show Password
LOGIN
RESET

Experiment No.: 4

JDBC

Call-level interfaces such as JDBC are programming interfaces allowing external access to SQL database manipulation and update commands. They allow the integration of SQL calls into a general programming environment by providing library routines which interface with the database. In particular, Java based JDBC has a rich collection of routines which make such an interface extremely simple and intuitive.
what happens in a call level interface: You are writing a normal Java program. Somewhere in the program, you need to interact with a database. Using standard library routines, you open a connection to the database. You then use JDBC to send your SQL code to the database, and process the results that are returned. When you are done, you close the connection.

Purpose: A Program to execute select query using JDBC import

```
java.sql.*;

class SelectFromPer {

public static void main(String argv[]) { try {

// Load the JDBC-ODBC bridge
Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");

// specify the ODBC data source's URL String
url = "jdbc:odbc:SSPer";

// connect
Connection con = DriverManager.getConnection(url,"North","Ken");

// create and execute a SELECT Statement stmt =
con.createStatement();
ResultSet rs = stmt.executeQuery
("SELECT Surname,FirstName,Category FROM Per");

System.out.println("Class is SelectFromPer\n");
// traverse through results
System.out.println("Found row:");

while (rs.next()) {

// get current row values
String Surname = rs.getString(1); String
FirstName = rs.getString(2); int Category =
rs.getInt(3);
// print values
System.out.print (" Surname=" + Surname);
System.out.print (" FirstName=" + FirstName);
System.out.print (" Category=" + Category);
System.out.print(" \n");
}
// close statement and connection
stmt.close();
con.close();
} catch (java.lang.Exception ex) {

ex.printStackTrace();
}

}

}
```

Experiment No.: 5

SERVLETS

Servlets are the Java platform technology of choice for extending and enhancing Web servers. Servlets provide a component-based, platform-independent method for building Web-based applications, without the performance limitations of CGI programs. And unlike proprietary server extension mechanisms (such as the Netscape Server API or Apache modules), servlets are server- and platform-independent. This leaves you free to select a "best of breed" strategy for your servers, platforms, and tools.
Servlets have access to the entire family of Java APIs, including the JDBC API to access enterprise databases. Servlets can also access a library of HTTP-specific calls and receive all the benefits of the mature Java language, including portability, performance, reusability, and crash protection.

Today servlets are a popular choice for building interactive Web applications. Third-party servlet containers are available for Apache Web Server, Microsoft IIS, and others. Servlet containers are usually a component of Web and application servers, such as BEA WebLogic Application Server, IBM WebSphere, Sun Java System Web Server, Sun Java System Application Server, and others.

PROGRAMS

PROGRAM 1

Purpose: A simple servlet that just generates plain text package hall;

```
import java.io.*; import
javax.servlet.*;
import javax.servlet.http.*;

public class HelloWorld extends HttpServlet { public void
  doGet(HttpServletRequest request,
                    HttpServletResponse response) throws
      ServletException, IOException {
    PrintWriter out = response.getWriter();
    out.println("Hello World");
  }
}
```

PROGRAM 2

Purpose: A Program which displays cookie id

```
import java.io.*; import
javax.servlet.*;
import javax.servlet.http.*;

public class CookieExample extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws IOException, ServletException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        // print out cookies

        Cookie[] cookies = request.getCookies(); for (int i =
        0; i < cookies.length; i++) {
            Cookie c =  cookies[i]; String name
            = c.getName(); String value =
            c.getValue();
            out.println(name + " = " + value);
        }

        // set a cookie

        String name = request.getParameter("cookieName"); if (name !=
        null && name.length() > 0) {
            String value = request.getParameter("cookieValue"); Cookie c =
            new Cookie(name, value); response.addCookie(c);
        }
    }
}
```

Experiment No.: 6

JAVA SERVER PAGES

JavaServer Pages (JSP) is a Java technology that allows software developers to dynamically generate HTML, XML or other types of documents in response to a Web client request. The technology allows Java code and certain pre-defined actions to be embedded into static content.

The JSP syntax adds additional XML-like tags, called JSP actions, to be used to invoke built-in functionality. Additionally, the technology allows for the creation of JSP tag libraries that act as extensions to the standard HTML or XML tags.
Tag libraries provide a platform independent way of extending the capabilities of a Web server.JSPs are compiled into Java Servlets by a JSP compiler. A JSP compiler may generate a servlet in Java code that is then compiled by the Java compiler, orit may generate byte code for the servlet directly.

JSP technology enables Web developers and designers to rapidly develop and easily maintain, information-rich, dynamic Web pages that leverage existing business systems. As part of the Java technology family, JSP technology enables rapid development of Web-based applications that are platform independent.
JSP technology separates the user interface from content generation, enabling designers to change the overall page layout without altering the underlying dynamic content

PROGRAMS

PROGRAM 1

Purpose A program for basic arithmetic functions

```
<html>
  <head>
    <title>JSP 2.0 Expression Language - Basic Arithmetic</title>
  </head>
  <body>
    <h1>JSP 2.0 Expression Language - Basic Arithmetic</h1>
    <hr>
    This example illustrates basic Expression Language arithmetic. Addition (+),
    subtraction (-), multiplication (*), division (/ or div), and modulus (% or mod) are
    all supported. Error conditions, like division by zero, are handled gracefully.
    <br>
    <blockquote>
      <code>
        <table border="1">
          <thead>
               <td><b>EL Expression</b></td>
               <td><b>Result</b></td>
```

```
</thead>
<tr>
  <td>\${1}</td>
```

```
                  <td>${1}</td>
                </tr>
                <tr>
                  <td>\${1 + 2}</td>
                  <td>${1 + 2}</td>
                </tr>
                <tr>
                  <td>\${1.2 + 2.3}</td>
                  <td>${1.2 + 2.3}</td>
                </tr>
                <tr>
                  <td>\${1.2E4 + 1.4}</td>
                  <td>${1.2E4 + 1.4}</td>
                </tr>
                <tr>
                  <td>\${-4 - 2}</td>
                  <td>${-4 - 2}</td>
                </tr>
                <tr>
                  <td>\${21 * 2}</td>
                  <td>${21 * 2}</td>
                </tr>
                <tr>
                  <td>\${3/4}</td>
                  <td>${3/4}</td>
                </tr>
                <tr>
                  <td>\${3 div 4}</td>
                  <td>${3 div 4}</td>
                </tr>
                <tr>
                  <td>\${3/0}</td>
                  <td>${3/0}</td>
                </tr>
                <tr>
                  <td>\${10%4}</td>
                  <td>${10%4}</td>
                </tr>
                <tr>
                  <td>\${10 mod 4}</td>
                  <td>${10 mod 4}</td>
                </tr>
        <tr>
          <td>\${(1==2) ? 3 : 4}</td>
          <td>${(1==2) ? 3 : 4}</td>
```

```
    </tr>
          </table>
      </code>
    </blockquote>
  </body>
</html>
```

Experiment No.: 7

A Program to display a String using JSP
Purpose A Program to display a String

```
<html>
  <head>
    <title>JSP 2.0 Examples - Hello World SimpleTag Handler</title>
  </head>
  <body>
    <h1>JSP 2.0 Examples - Hello World SimpleTag Handler</h1>
    <hr>
    <p>This tag handler simply echos "Hello, World!" It's an example of a very basic
    SimpleTag handler with no body.</p>
    <br>
    <b><u>Result:</u></b>
    <mytag:helloWorld/>
  </body>
</html>
```

Experiment No.: 8

JAVA BEANS

A Java Bean is the name trademarked by Sun and given to a Java class that adheres to a specific and well-defined set of interface specifications.

According to JavaSoft,"A Java Bean is a reusable software component that can be manipulated visually in a builder tool."

Beans are "capsules" of code, each designed for a specific purpose. The advantage of Java Beans over standard programming controls is that Beans are independent. They are not specific to operating systems or development environments. A Bean created in one development environment can be easily copied and modified by another. This allows Java Beans greater flexibility in enterprise computing, as components are easily shared between developers

The following five attributes are common to Beans.

- Properties
- Customization
- Persistence
- Events
- Introspection

we are usually referring to the following three attributes of the class:

- Properties
- Methods
- Events

PROGRAM 1

Purpose: A program to generates plain text import

```
java.awt.Color;
import java.beans.XMLDecoder; import
javax.swing.JLabel; import java.io.Serializable;
public class SimpleBean extends JLabel implements Serializable
{
    public SimpleBean()
    {
        setText( "Hello world!" ); setOpaque(
        true );
        setBackground( Color.RED ); setForeground(
        Color.YELLOW ); setVerticalAlignment( CENTER
        ); setHorizontalAlignment( CENTER );
      }

}
```

Experiment No.: 9

Remote Method Invocation (RMI)

Remote Method Invocation (RMI) is the object equivalent of Remote Procedure Calls (RPC). While RPC allows you to call procedures over a network, RMI invokes an object's methods over a network.

In the RMI model, the server defines objects that the client can use remotely. The clients can now invoke methods of this remote object as if it were a local object running in the same virtual machine as the client. RMI hides the underlying mechanism of transporting method arguments and return values across the network. In Java-RMI, an argument or return value can be of any primitive Java type or any other Serializable Java object.

Purpose : A Program on Stock Market

1. Develop your Remote Interface

```
package SimpleStocks; import
java.util.*; import java.rmi.*;

public interface StockMarket extends java.rmi.Remote { float get_price( String
  symbol ) throws RemoteException;
}
```

2. Implement your Java/RMI Server

```
package SimpleStocks; import
java.rmi.*;
import java.rmi.server.UnicastRemoteObject;

public class StockMarketImpl
  extends UnicastRemoteObject implements StockMarket {

  public StockMarketImpl( String name ) throws RemoteException { try {
      Naming.rebind( name, this );
    }
    catch( Exception e ) {
      System.out.println( e );
    }
  }

  public float get_price( String symbol ) { float price = 0;
    for( int i = 0; i < symbol.length(); i++ ) { price += (int)
      symbol.charAt( i );
    }
    price /= 5; return price;
  }

}
```

Implement an Application that creates your Server

```
import java.rmi.*;
import java.rmi.server.UnicastRemoteObject; import
SimpleStocks.*;

public class StockMarketServer {

  public static void main(String[] args) throws Exception {
    if(System.getSecurityManager() == null) {
      System.setSecurityManager( new RMISecurityManager() );
    }
    StockMarketImpl myObject = new StockMarketImpl( "NASDAQ" );
    System.out.println( "RMI StockMarketServer ready..." );
  }
  }
```

VIVA QUESTIONS

JDBC

Q.1. Once I have the Java 2 SDK, Standard Edition, from Sun, what else do I need to connect to a database
Q.2. What's the JDBC 3.0 API?
Q.3. How do I start debugging problems related to the JDBC API?
Q.4. Are all the required JDBC drivers to establish connectivity to my database part of the JDK?

SERVLETS

Q.1.How do I get started with Servlets using Tomcat? How do Servlets compare to CGI?

Q.3.How do I upload a file to my

Servlet?

Q.4. How do I debug a Servlet?

JSP

Q.1.What is JavaServer Pages technology?

Q.2. How does the JavaServer Pages technology work? Which web servers support JSP

technology?

Q.3. How is a JSP page invoked and compiled?

JAVA BEANS

Q.1.What is a Bean? Why isn't a Bean an Applet?

Q.2.Is JavaBeans complete component architecture?

Q.3.What is the security implications for downloading Beans over the Internet?

Q.4.What is the relationship between Sun's JFCs and JavaBeans?

RMI

Q.1.Does RMI require using an HTTP server?

Q.2.Will there be debugging mechanisms built into RMI?

Q.3.Where can I find a list of system properties that might be useful for implementing and debugging RMI applications?

Q.4.How do RMI clients contact remote RMI servers?