

Practical No. 14

Aim :-

Write a program to demonstrate the use of URL and URLConnection class and its methods.

Theory :-

Java InetAddress class

Java InetAddress class represents an IP address. The `java.net.InetAddress` class provides methods to get the IP of any host name for example `www.javatpoint.com`, `www.google.com`, `www.facebook.com`, etc.

An IP address is represented by 32-bit or 128-bit unsigned number. An instance of `InetAddress` represents the IP address with its corresponding host name. There are two types of addresses: Unicast and Multicast. The Unicast is an identifier for a single interface whereas Multicast is an identifier for a set of interfaces.

Moreover, `InetAddress` has a cache mechanism to store successful and unsuccessful host name resolutions.

InetAddress Class Factory Methods :

Method	Description
<code>static InetAddress getLocalHost() throws UnknownHostException</code>	This method returns the instance of <code>InetAddress</code> containing the local hostname and address.
<code>public static InetAddress getByName (String host) throws UnknownHostException</code>	This method returns the instance of <code>InetAddress</code> containing LocalHost IP and Name.
<code>static InetAddress[] getAllByName (String hostName) throws UnknownHostException</code>	This method returns the array of the instance of <code>InetAddress</code> class which contains IP address.
<code>static InetAddress getByAddress (byte IPAddress []) throws UnknownHostException</code>	This method returns an <code>InetAddress</code> object created from the raw IP address.
<code>static InetAddress getByAddress (String hostName, byte IPAddress []) throws UnknownHostException</code>	This method creates and returns an <code>InetAddress</code> based on the provided hostname and IP address.

Code :-

```

import java.io.*;
import java.net.*;
import java.util.*;

class prac14 {
    public static void main(String[] args)
        throws UnknownHostException
    {
        // To get and print InetAddress of Local Host
        InetAddress address1 = InetAddress.getLocalHost();
        System.out.println("InetAddress of Local Host : "
            + address1);

        // To get and print InetAddress of Named Host
        InetAddress address2
            = InetAddress.getByName("45.22.30.39");
        System.out.println("InetAddress of Named Host : "
            + address2);

        //To get and print All InetAddresses of Named Host
        InetAddress address3[]
            = InetAddress.getAllByName("172.19.25.29");
        for (int i = 0; i < address3.length; i++) {
            System.out.println("ALL InetAddresses of Named
                Host : " + address3[i]);
        }

        // To get and print InetAddresses of Host with //specified IP
        Address
        byte IPAddress[] = { 125, 0, 0, 1 };
        InetAddress address4
            = InetAddress.getByAddress(IPAddress);
        System.out.println(
            "InetAddresses of Host with specified IP
                Address : " + address4);

        // To get and print InetAddresses of Host with
        // specified IP Address and hostname
        byte[] IPAddress2
            = { 105, 22, (byte)223, (byte)186 };
        InetAddress address5 = InetAddress.getByAddress(
            "abc.com", IPAddress2);
        System.out.println(
            "InetAddresses of Host with specified IP

```

```
        Address and hostname : " + address5);  
    }  
}
```

Output :-

PROBLEMS 17 OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

```
PS A:\yash.java> cd "a:\yash.java\" ; if ($?) { javac prac14.java } ; if ($?) { java prac14 }  
InetAddress of Local Host : yash/192.168.43.140  
InetAddress of Named Host : /45.22.30.39  
ALL InetAddresses of Named Host : /172.19.25.29  
InetAddresses of Host with specified IP Address : /125.0.0.1  
InetAddresses of Host with specified IP Address and hostname : abc.com/105.22.223.186  
PS A:\yash.java>
```

Practical No. 15

Aim :-

Write a Program to demonstrate the use of URL and URLConnection class and its methods.

Theory :-

URL Class

URL known as Uniform Resource Locator is simply a string of text that identifies all the resources on the Internet, telling us the address of the resource, how to communicate with it, and retrieve something from it.

The URL class is the gateway to any of the resources available on the internet. A Class URL represents a Uniform Resource Locator, which is a pointer to a “resource” on the World Wide Web.

URLConnection Class :-

URLConnection Class in Java is an abstract class that represents a connection of a resource as specified by the corresponding URL. It is imported by the *java.net* package. The URLConnection class is utilized for serving two different yet related purposes, Firstly it provides control on interaction with a server(especially an HTTP server) than URL class. Secondly, with a URLConnection we can check the header sent by the server and respond accordingly, we can configure header fields used in client requests. We can also download binary files by using URLConnection.

Methods of URLConnection Class –

- **URL getURL ()**
Returns the value of this URLConnection’s URL field.
- **abstract void connect () throws IOException**
This method is used for establishing connection to the resource specified URL , if such connection has not already been established .
- **int getContentLength ()**
Returns the value of the content
- **String getContentType ()**
Returns the value of the Content-type header field

Code :-

```
import java.io.*;
import java.net.*;

// Main class
// URLConnectionExample
public class prac15 {

    // Main driver method
    public static void main(String[] args) throws Exception
    {
        // Try block to check for exceptions
        try {

            // Creating an object of URL class
            URL u = new URL("http://www.google.com");

            // Creating an object of URLConnection class to
            URLConnection urlconnect = u.openConnection();

            // Creating an object of InputStream class
            // for our application streams to be read
            InputStream stream
                = urlconnect.getInputStream();

            // Declaring an integer variable
            int i;
            int a = 0;
            // Till the time URL is being read
            while ((i = stream.read()) != -1) {

                // Continue printing the stream
                System.out.print((char)i);
                a++;
                if (a==1000){
                    break;
                }
            }

            //getURL() method displaying URL
            System.out.println("\n\nURLConnection's URL Field - " +
                urlconnect.getURL());

            //getContentLength() method displaying Length
            System.out.println("\nContent Length - " +
                urlconnect.getContentLength());
        }
    }
}
```

```

        //getContentType() method displaying Type
        System.out.println("\nContent Type - " + urlconnect.getContentType());

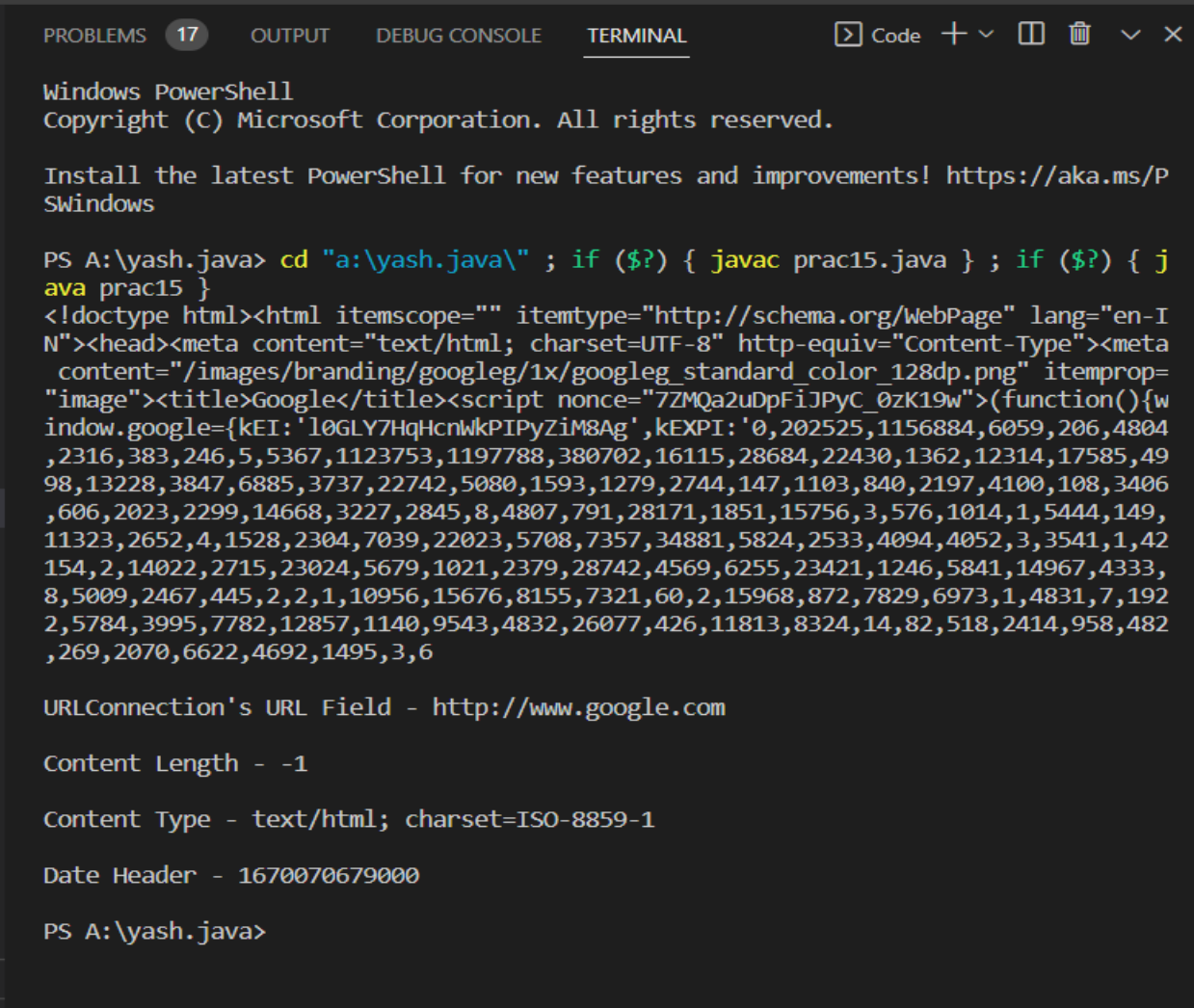
        //getDate() method displaying Date
        System.out.println("\nDate Header - " + urlconnect.getDate() + "\n");
    }

    // Catch block to handle the exception
    catch (Exception e) {

        // Print the exception on the console
        System.out.println(e);
    }
}
}
}

```

Output :-



```

PROBLEMS 17 OUTPUT DEBUG CONSOLE TERMINAL Code + - [ ] [X] [V] [X]
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS A:\yash.java> cd "a:\yash.java\" ; if ($?) { javac prac15.java } ; if ($?) { java prac15 }
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="en-IN"><head><meta content="text/html; charset=UTF-8" http-equiv="Content-Type"><meta content="/images/branding/googleg/1x/googleg_standard_color_128dp.png" itemprop="image"><title>Google</title><script nonce="7ZMQa2uDpFiJPYC_0zK19w">(function(){window.google={kEI:'l0GLY7HqHcnWkPIPyZiM8Ag',kEXPI:'0,202525,1156884,6059,206,4804,2316,383,246,5,5367,1123753,1197788,380702,16115,28684,22430,1362,12314,17585,4998,13228,3847,6885,3737,22742,5080,1593,1279,2744,147,1103,840,2197,4100,108,3406,606,2023,2299,14668,3227,2845,8,4807,791,28171,1851,15756,3,576,1014,1,5444,149,11323,2652,4,1528,2304,7039,22023,5708,7357,34881,5824,2533,4094,4052,3,3541,1,42154,2,14022,2715,23024,5679,1021,2379,28742,4569,6255,23421,1246,5841,14967,4333,8,5009,2467,445,2,2,1,10956,15676,8155,7321,60,2,15968,872,7829,6973,1,4831,7,1922,5784,3995,7782,12857,1140,9543,4832,26077,426,11813,8324,14,82,518,2414,958,482,269,2070,6622,4692,1495,3,6

URLConnection's URL Field - http://www.google.com

Content Length - -1

Content Type - text/html; charset=ISO-8859-1

Date Header - 1670070679000

PS A:\yash.java>

```

Practical No. 18

Aim :-

Write a program to insert and retrieve data from database using JDBC.

Theory :-

JDBC

JDBC or Java Database Connectivity is a Java API to connect and execute the query with the database. It is a specification from Sun microsystems that provides a standard abstraction(API or Protocol) for java applications to communicate with various databases. It provides the language with java database connectivity standards. It is used to write programs required to access databases. JDBC, along with the database driver, can access databases and spreadsheets. The enterprise data stored in a relational database(RDB) can be accessed with the help of JDBC APIs.

Purpose of JDBC

Enterprise applications created using the JAVA EE technology need to interact with databases to store application-specific information. So, interacting with a database requires efficient database connectivity, which can be achieved by using the [ODBC](#)(Open database connectivity) driver. This driver is used with JDBC to interact or communicate with various kinds of databases such as Oracle, MS Access, Mysql, and SQL server database.

Code :-

```
// Program to Insert Data into Database
```

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
import java.sql.SQLException;

public class InsertStaticOracle
{
    public static void main(String args[])
    {
        Statement st = null;
        Connection connection = null;
        try{
            oracle.jdbc.OracleDriver driverObj = new oracle.jdbc.OracleDriver();

            String url = "jdbc:oracle:thin:@localhost:1521:XE", username = "System" ,password
= "pass";

            connection =DriverManager.getConnection(url,username,password);

            if(connection!=null)
```

```
System.out.println("Connection established successfully");

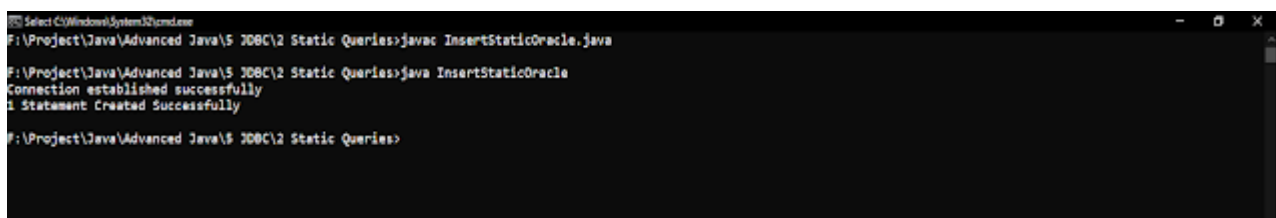
st = connection.createStatement();

String qry = "Insert into Student values(104 , 'Atharva Agrawal', 'Dhule')";

int count = st.executeUpdate(qry);

System.out.println(count+" Statement Created Successfully ");
}
catch(Exception e)
{
    e.printStackTrace();
}
finally{
    try
    {
        if(st != null)
            st.close();
        if(connection != null)
            connection.close();
    }
    catch(SQLException e){
        e.printStackTrace();
    }
}
}
```

Output:



```
C:\Windows\System32\cmd.exe
F:\Project\Java\Advanced Java\5 208C\2 Static Queries>javac InsertStaticOracle.java
F:\Project\Java\Advanced Java\5 208C\2 Static Queries>java InsertStaticOracle
Connection established successfully
1 Statement Created Successfully
F:\Project\Java\Advanced Java\5 208C\2 Static Queries>
```


Code :-

```
// Program retrieve Data into Database
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.DriverManager;
import java.sql.Statement;
import java.sql.ResultSet;

public class SelectOracle
{
    public static void main(String args[]) throws SQLException
    {
        System.out.println("Step 1: ");
        oracle.jdbc.driver.OracleDriver obj = new oracle.jdbc.driver.OracleDriver();
        // Class.forName(oracle.jdbc.driver.OracleDriver);
        System.out.println("Driver loaded successfully");

        System.out.println("Step 2: ");
        String url="jdbc:oracle:thin:@localhost:1521:XE",uname="SYSTEM" ,
password="Atharva007";
        Connection connection = DriverManager.getConnection(url,uname,password);
        if(connection!=null)
            System.out.println("Connection Established Successfully");
        else
            System.out.println("Connection Not Established Successfully");

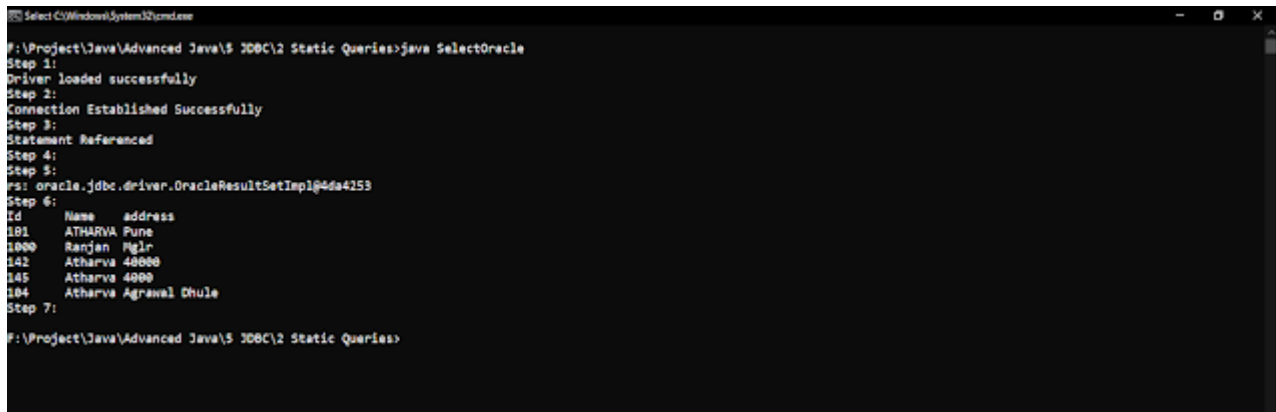
        System.out.println("Step 3: ");
        Statement st = connection.createStatement();
        System.out.println("Statement Referenced ");

        System.out.println("Step 4: ");
        System.out.println("Step 5: ");
        String qry = "select * from Student";
        ResultSet rs = st.executeQuery(qry);
        System.out.println("rs: "+rs);

        System.out.println("Step 6: ");
        System.out.println("Id\tName\taddress");
        while(rs.next())
        {
            int x = rs.getInt(1);
            String y = rs.getString("Name");
            String s = rs.getString(3);
            System.out.println(x+"\t"+y+"\t"+s);
        }
    }
}
```

```
System.out.println("Step 7: ");  
rs.close();  
st.close();  
connection.close();  
}  
}
```

Output:



```
Select C:\Windows\System32\cmd.exe  
F:\Project\Java\Advanced Java\S JDBC\2 Static Queries>java SelectOracle  
Step 1:  
Driver loaded successfully  
Step 2:  
Connection Established Successfully  
Step 3:  
Statement Referenced  
Step 4:  
Step 5:  
rs: oracle.jdbc.driver.OracleResultSetImpl@4da4253  
Step 6:  
id      Name      address  
181     ATHARVA  Pune  
1800    Ranjan   Mglr  
142     Atharva  48000  
145     Atharva  48000  
184     Atharva  Agrawal Dhule  
Step 7:  
F:\Project\Java\Advanced Java\S JDBC\2 Static Queries>
```

Practical No. 22

Aim :-

Write a Servlet Program to send username and password using HTML forms and authenticate user.

Theory :-

Today we all are aware of the need of creating dynamic web pages i.e the ones which have the capability to change the site contents according to the time or are able to generate the contents according to the request received by the client. If you like coding in Java, then you will be happy to know that using Java there also exists a way to generate dynamic web pages and that way is Java Servlet. But before we move forward with our topic let's first understand the need for server-side extensions.

Servlets are the Java programs that run on the Java-enabled web server or application server. They are used to handle the request obtained from the webserver, process the request, produce the response, then send a response back to the webserver.

Code :-

HTML Code –

```
<html>
  <head>
    <title>Servlet</title>
  </head>

  <body>
    <br /><br />

    <center>
      <h2><b> User's Authentication Page </b></h2>
    </center>

    <form action=" ../servlets/login" method="POST">
      <br /><br />

      <table
        border="0"
        cellpadding="5"
        cellspacing="0"
        width="400"
        align="Center"
      >
        <font color="#000000" size="2">
          <tr>
            <td align="Right">
```

```
        <b>Login Name:</b>
    </td>
    <td>
        <input type="Text" name="Loginid" size="20" />
    </td>
</tr>

<tr>
    <td align="Right">
        <b>Password:</b>
    </td>
    <td>
        <input type="Password" name="Password" size="20" />
    </td>
</tr>
</font>
</table>
<br /><br />

<p></p>
<p></p>
<center>
    <input type="Submit" value="Submit" name="Submit" />
    <input type="Reset" value="Reset" name="Reset" />
</center>
</form>
</body>
</html>
```

Java Servlet Code –

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

public class prac22 extends HttpServlet {
    Connection conn;

    public void service(HttpServletRequest request, HttpServletResponse response)
        throws IOException {
        ServletOutputStream out = response.getOutputStream();

        String url = "jdbc:odbc:MyDsn";

        try {
            String login = request.getParameter("Loginid");
            String pass = request.getParameter("Password");
```

```

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
conn = DriverManager.getConnection(url);
Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery("select * from login where loginname='"
    + login + "' and password= '" + pass + "'");

response.setContentType("text/html");
response.setStatus(HttpServletResponse.SC_OK);

if (rs.next() == false) {
    out.println("<BR><BR><BR><BR><BR>");
    out.println("<html><head><title>Login check</title></head><body>");
    out.println("<CENTER><B>Unknown User</B>");
    out.println("<BR><BR>");
    out.println("<h3> Access Denied</h3></CENTER>");
    out.println("<BR><BR><BR>");
    out.println("<CENTER><Input Type=Button Value=Back></CENTER>");
    out.println("</body></html>");
} else {
    out.println("<html><head><title>Login Check</title></head><body>");
    out.println("<BR><BR><BR><BR><BR>");
    out.println("<CENTER><B>Welcome " + " " + rs.getString("loginname")+
        "</B></CENTER>");
    out.println("<CENTER>");
    out.println("<h3> You have been Authenticated</h3></CENTER>");
    out.println("<BR><BR><BR>");
    out.println("<CENTER><Input Type=Button Value=HOME></CENTER>");
    out.println("</body></html>");
}
conn.close();

} catch (SQLException SQLExcp) {
    out.println("SQLException: " + SQLExcp);

} catch (ClassNotFoundException clsNotFndExcp) {
    System.out.println("Cannot find the class: " + clsNotFndExcp);
}
}
}

```

Output :-

User's Authentication Page

Login Name:

Password: 