Experiment-4

Create Java project

↓

Right click on src → new → name of class (JDBc.java)

↓

Write logic for the experiment to interact with database

↓

Configure JDBC driver → Right click on project → Build path
→ Configure Build path → Libraries → classpath
→ Add External Jars → provide path to Jarfile.

↓

Open MySQL → Create corresponding Database & table

↓

In JDBC code use same database & table names.

↓

keep MySQL open

↓

Run JDBC program

↓

Check changes at console & database table.

Experiment-5

* Steps

1] create 2 Java project

i) RMI client

ii) RMI server

2] RMI client → (2 file contain)

i) PalindromeInterface

ii) clientclass

3] RMIServer → (3 file contain)

i) PalindromeInterface

ii) palindromeImplementclass

iii) serverclass

- palindromeInterface in RMIclient & RMIServer is same.

- Run palindromeImplement class. [O/P generated]

* Right click on project
↓
properties
↓
Location
↓
path of project directory.
↓
open above path

open above path

↓

open bin folder

↓

In address bar type

↓

Cmd

↓

Enter

↓

open command prompt

↓

(If you open command prompt manually then set path to directory) – bin folder of RMIServer project

↓

Execute on command prompt

→ rmic palindromeImplementclass

↓

Warning will be displayed.

↓

again Execute on Command prompt

→ start rmiregistry

↓

New command prompt like registry window open

↓

open bin folder of RMIServer project

↓

palindromeImplementclass_stub file will be created palindromeImplementClass_stub file will be created

↓

Copy this file into bin folder of RMIClient project

↓

goto Eclipse → run serverclass

↓

"Server Ready" message will be displayed

↓

Run client class

↓

Result of palindrome or Not palindrome will be displayed.

Experiment-7

Create Dynamic Web project

↓

Enter project name

↓

Download and configure Apache-tomcat-server

Target Runtime → New Runtime → Apache → Select Version

↓

Tomcat Installation directory ← Next ← Local server ← Create new

↓

Set path to folder where Apache-Tomcat is available

↓

finish

↓

click next

↓

Check on Generate web.xml deployment descriptor

↓

finish

↓

Right click on WebContent/WebApp folder

↓

new

↓

html

↓

Name this file as index.html

Name this file as index.html

↓

Create a form which accepts username & password in html file.

↓

Right click on Java Resource

↓

New → servlet → NameofServletClass (Login.java)

↓

Link html form with the servlet using action attribute of html & call servlet after submitting form

↓

servlet will read username & password from html & display it.

↓

Create JDBC program Java Resources

↓

New

↓

Class

↓

class Name (UserDao.java)

↓

Write logic for reading database table with given Username

↓

Configure JDBC driver

↓

Configure JDBC driver

↓

Right click on project

↓

Build path

↓

configure Build path

↓

Libraries

↓

classpath

↓

Add External Jars

↓

provide path to Jar file.

↓

Ensure JDBC connector is available in Webapp

↓

Web-INF

↓

lib folder

↓

If not copy paste JDBC connector manually to above folder

↓

Check & verify username & password from database

↓

Display username & password from html as well as database

↓

Run program on server

Run program on server

↓

Select approprite browser in window

↓

preferences of Eclipse

↓

If html file is not rendered automatically then open it manually by appending index.html to localhost path.

Experiment -8

Create Java project

↓

Right click on src → new → Name of class (JDBC.java)

↓

Write logic for the experiment to interact with database

↓

Use switch case statement to implement operations
CRUD

↓

(create, Read, update, delete)

↓

Configure JDBC driver.

↓

Right click on project → Build path → configure Build path

Libraries ←

↓

class path

↓

Add external Jars

↓

provide path to Jar file

↓

open MySQL

↓

Create corresponding Database & table

↓

In JDBC code use same database & table names

↓

Keep MYSQL open

↓

Run JDBC program

↓

Check changes at console & database table.

Experiment - 9

9 -①

Create Dynamic Web project

↓

Enter project Name.

↓

Downbad & configure Apache-Tomcat Server —

Target Runtime

↓

New Runtime

↓

Apache → Select version

↓

Create new local server → Next

↓

Tomcat Installation Directory

↓

Set path to the folder where Apache - Tomcat is available

↓

finish.

↓

click Next → Check on generate web.xml deployment descriptor → finish

↓

Right click on Webcontent / WebApp folder

↓

New

↓

html (index. html)

Create a form which accepts number in html file.

↓

Link html form with the JSP using action attribute
of html and call JSP after submitting form

↓

Right click on WebContent/WebApp folder

↓

New

↓

JSP (index.jsp)

↓

In jsp file write java code to read number
from html file & display cube of it.