Experiment No. 04

Learning Objective: Implement Apache Tomcat

Tools:

Apache Tomcat,Eclipse IDE,Maven,JDK (Java Development Kit)

Theory:

Introduction to Apache Tomcat

Apache Tomcat is an open-source implementation of the Java Servlet, JavaServer Pages (JSP), and WebSocket technologies. It acts as a web server and servlet container, providing a "pure Java" HTTP web server environment in which Java code can run. Tomcat is widely used for developing and deploying web applications in Java and serves as a foundational technology for more complex enterprise-level systems.

Servlets and Their Role in Web Applications

Servlets are Java programs that run on a server and handle client requests. They are a core component of Java's server-side programming model. When a client (usually a web browser) makes a request to a web server, the server can pass that request to a servlet for processing. The servlet can then generate a dynamic response, such as an HTML page, which is sent back to the client. This allows for the creation of dynamic web content, in contrast to static content like simple HTML files.

Servlet Lifecycle

The lifecycle of a servlet is managed by the servlet container (in this case, Apache Tomcat). It includes the following stages:
Loading and Instantiation: The servlet class is loaded and instantiated when it is first requested or during the startup of the server.
Initialization (init method): The servlet container calls the init method to initialize the servlet. Request Handling (service method): The service method is called to handle each client request. For HTTP servlets, service dispatches requests to the doGet, doPost, doPut, etc., methods depending on the request type. Destruction (destroy method): The destroy method is called before the servlet is removed from service, allowing it to clean up any resources.

Web Application Structure

A typical Java web application follows a specific directory structure to ensure that the application can be easily managed and deployed. The key components include:

WEB-INF Directory: This is a special directory that contains configuration files, such as web.xml, and other resources that should not be directly accessible by clients.
web.xml: The deployment descriptor file that configures servlets, servlet mappings, context parameters, and other application-wide settings.

Servlets: Java classes that handle requests and generate dynamic content.

Code

Code for HelloWorldServelet.java

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

public class MyServlet extends HttpServlet{
    public void doGet(HttpServletRequest req,HttpServletResponse res)throws IOException,ServletException{

        //To print hello world
            PrintWriter out=res.getWriter();
            out.println("<h1>Hello World!</h1>");
    }
}
```

```
C:\Program Files\Apache Software Foundation\Tomcat 9.0\webapps\myapp\WEB-INF\cla
sses>javac MyServlet.java

C:\Program Files\Apache Software Foundation\Tomcat 9.0\webapps\myapp\WEB-INF\cla
sses>_
```

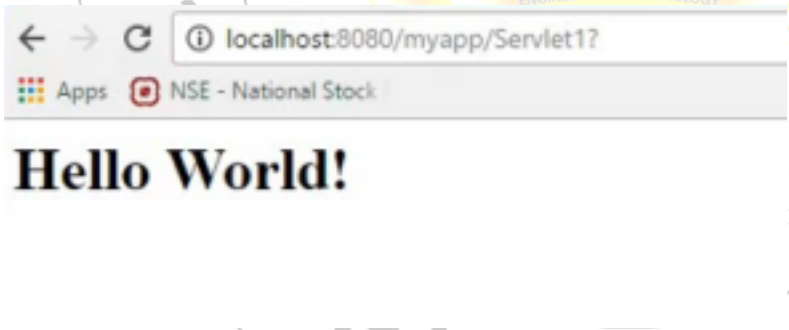
```html
<html>
<head>
    <title>Home page</title>
</head>
<body>
        <form action="Servlet1">
            <input type="submit" value="Invoke servlet"/>
        </form>
</body>
</html>
```

Code of web.xml

```
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
                      http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
  version="4.0"
  metadata-complete="true">

  <display-name>Welcome to Tomcat</display-name>
  <description>
     Welcome to Tomcat
  </description>
  <servlet>
    <servlet-class>MyServlet</servlet-class>
    <servlet-name>Servlet1</servlet-name>
  </servlet>
  <servlet-mapping>
    <servlet-name>Servlet1</servlet-name>
    <url-pattern>/Servlet1</url-pattern>
  </servlet-mapping>
</web-app>
```

Output:



Result and discussion:

Students will understand how to:

1. Set up a Java web application project.
2. Write and configure a basic Servlet.
3. Deploy and test the servlet in Apache Tomcat.
4. Identify and use the necessary tools for Java web development.

Learning Outcomes: Students should have the ability to

LO1: Identify various free available open-source tools for Java web development with Apache Tomcat.

LO2: Describe the pros and cons of each tool in the context of servlet-based web application development.

Course Outcomes:

1. Develop a foundational understanding of Java web application architecture.
2. Gain practical experience in deploying Java applications on an open-source web server.

Conclusion:

For Faculty Use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**
Marks Obtained			