

EX NO:9
DATE:08.06.2022

INSTALLING AND CONFIGURING APACHE TOMCAT WEBSERVER

Aim:

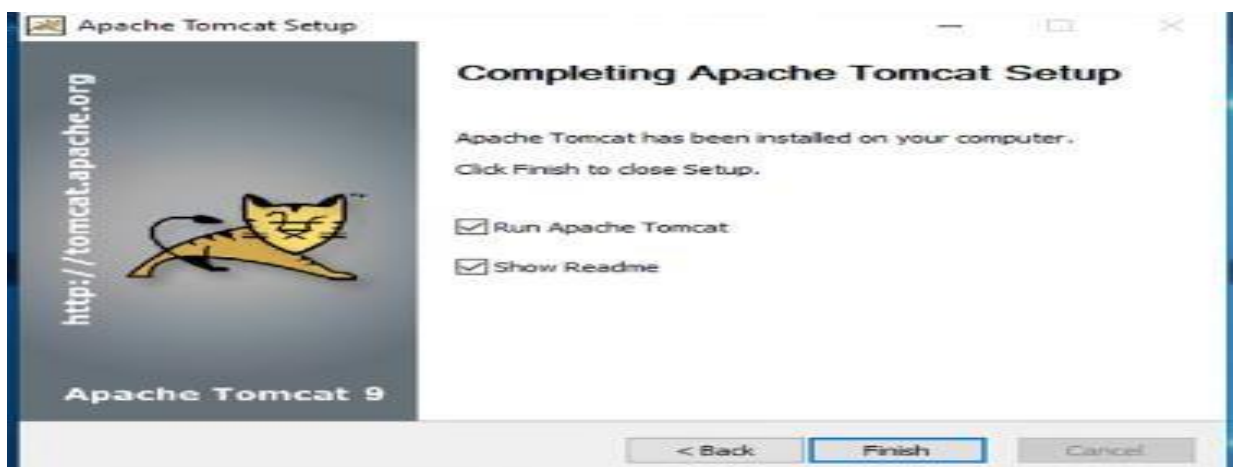
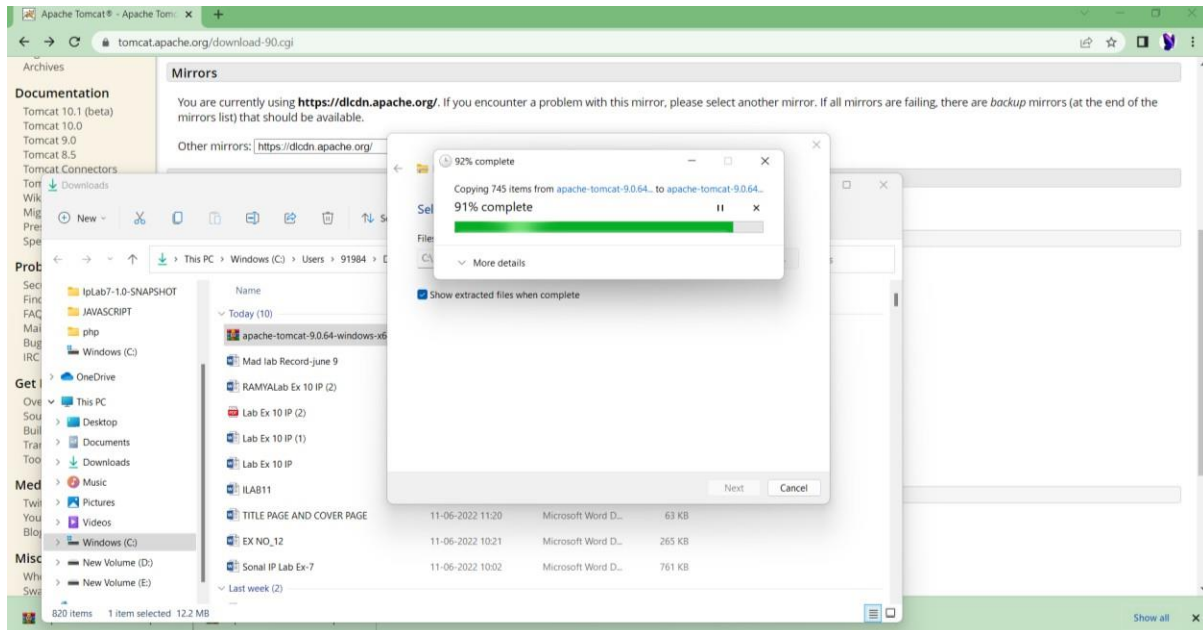
To install and configure apache tomcat webserver

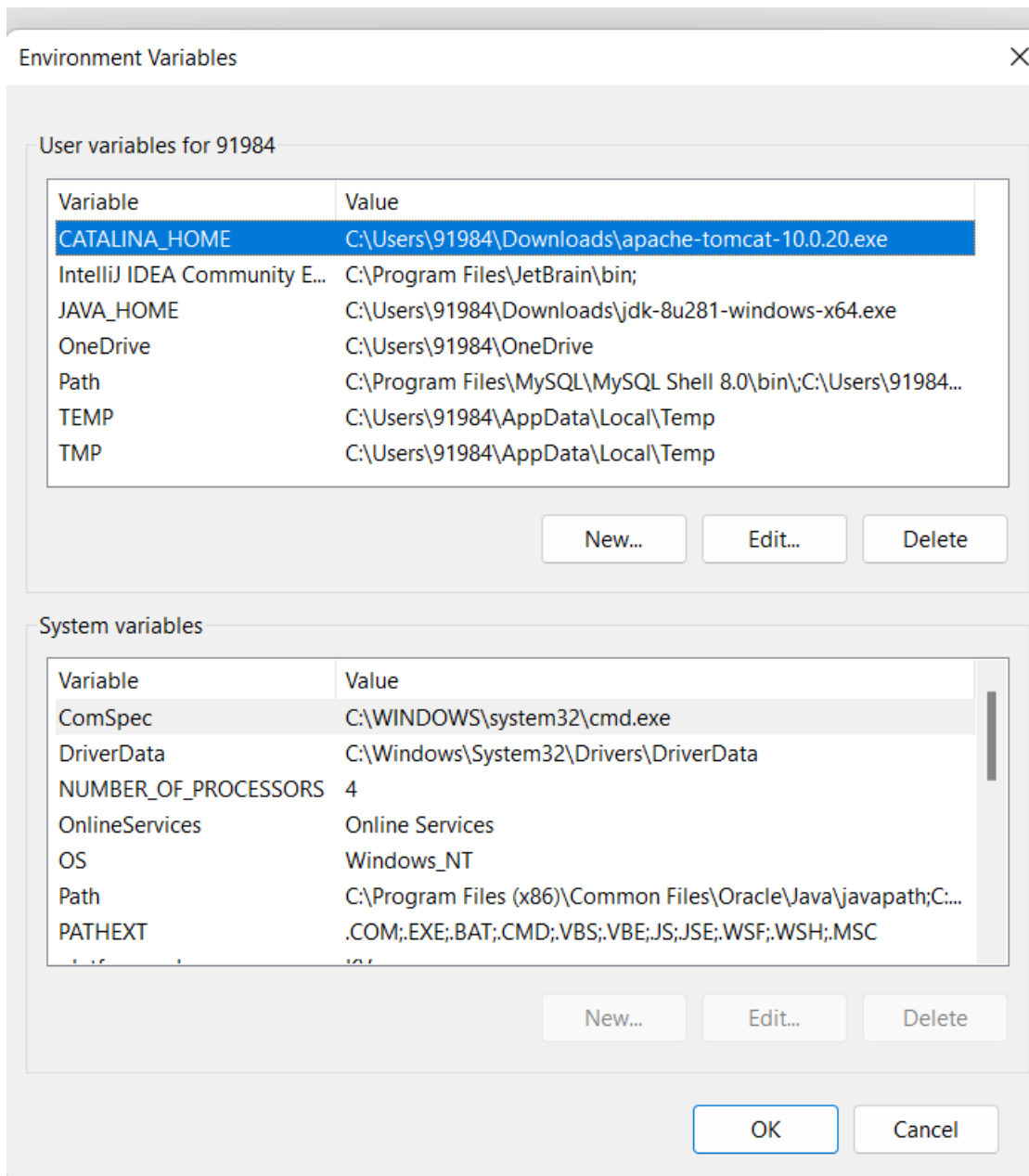
Procedure:

The first screenshot shows the Apache Tomcat 9.0.64 download page. The left sidebar contains links for Home, Taglibs, Maven Plugin, Download, Documentation, Problems?, Get Involved, Media, and Misc. The main content area includes a search bar, a welcome message, quick navigation links (KEYS, 9.0.64, Browse, Archives), release integrity information, and a list of mirrors. The '9.0.64' section is highlighted, showing a link to the README file. The 'Binary Distributions' section is also visible.

The second screenshot shows the 'Archives' page of the Apache Tomcat website. The left sidebar is the same as the first screenshot. The main content area shows the 'Mirrors' section, followed by the '9.0.64' section, and then the 'Binary Distributions' section. The 'Binary Distributions' section lists various download options for the 9.0.64 version, including Core, Full documentation, Deployer, Embedded, and Source Code Distributions. The 'Source Code Distributions' section lists links for tar.gz and zip files.

G.Ghulam Dasthageer
19CSE041






```
Tomcat
INFO: Initialization processed in 1780 ms
Jun 12, 2013 4:37:39 PM org.apache.catalina.core.StandardService startInternal
INFO: Starting service Catalina
Jun 12, 2013 4:37:39 PM org.apache.catalina.core.StandardEngine startInternal
INFO: Starting Servlet Engine: Apache Tomcat/7.0.40
Jun 12, 2013 4:37:39 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory E:\myserver\tomcat7.0.40\webapps\docs
Jun 12, 2013 4:37:40 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory E:\myserver\tomcat7.0.40\webapps\examples
Jun 12, 2013 4:37:40 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory E:\myserver\tomcat7.0.40\webapps\host-manager
Jun 12, 2013 4:37:40 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory E:\myserver\tomcat7.0.40\webapps\manager
Jun 12, 2013 4:37:40 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory E:\myserver\tomcat7.0.40\webapps\ROOT
Jun 12, 2013 4:37:40 PM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler ["http-bio-9999"]
Jun 12, 2013 4:37:40 PM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler ["ajp-bio-8009"]
Jun 12, 2013 4:37:40 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 1081 ms
```

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

Apache Tomcat/9.0.58

If you're seeing this, you've successfully installed Tomcat. Congratulations!

 Recommended Reading:
[Security Considerations How-To](#)
[Manager Application How-To](#)
[Clustering/Session Replication How-To](#)

[Server Status](#)
[Manager App](#)
[Host Manager](#)

Developer Quick Start

[Tomcat Setup](#) [Realms & AAA](#) [Examples](#) [Servlet Specifications](#)
[First Web Application](#) [JDBC DataSources](#) [Tomcat Versions](#)

Managing Tomcat

For security, access to the `manager.webapp` is restricted. Users are defined in `$CATALINA_HOME/conf/tomcat-users.xml`. In Tomcat 9.0 access to the manager application is split between different users. [Read more...](#)

[Release Notes](#)
[Changelog](#)
[Migration Guide](#)
[Security Notices](#)

Documentation

[Tomcat 9.0 Documentation](#)
[Tomcat 9.0 Configuration](#)
[Tomcat Wiki](#)

Find additional important configuration information in:
`$CATALINA_HOME/README.txt`
Developers may be interested in:
[Tomcat 9.0 Java Database](#)
[Tomcat 9.0 JavaDoc](#)
[Tomcat 9.0 Git Repository at GitHub](#)

Getting Help

FAQ and Mailing Lists

The following mailing lists are available:

- [tomcat-announce](#)
Important announcements, releases, security vulnerability notifications. (Low volume).
- [tomcat-users](#)
User support and discussion.
- [tomcat-dev](#)
User support and discussion for [Apache Tomcat](#).
- [tomcat-dev](#)
Development mailing list, including commit messages.

Other Downloads

[Tomcat Connectors](#)
[Tomcat Native](#)
[Tomcat Native](#)
[Tomcat Native](#)

Other Documentation

[Tomcat Connectors](#)
[Tomcat Native](#)
[Tomcat Native](#)
[Tomcat Native](#)

Get Involved

[Overview](#)
[Source Repositories](#)
[Mailing Lists](#)
[YML](#)

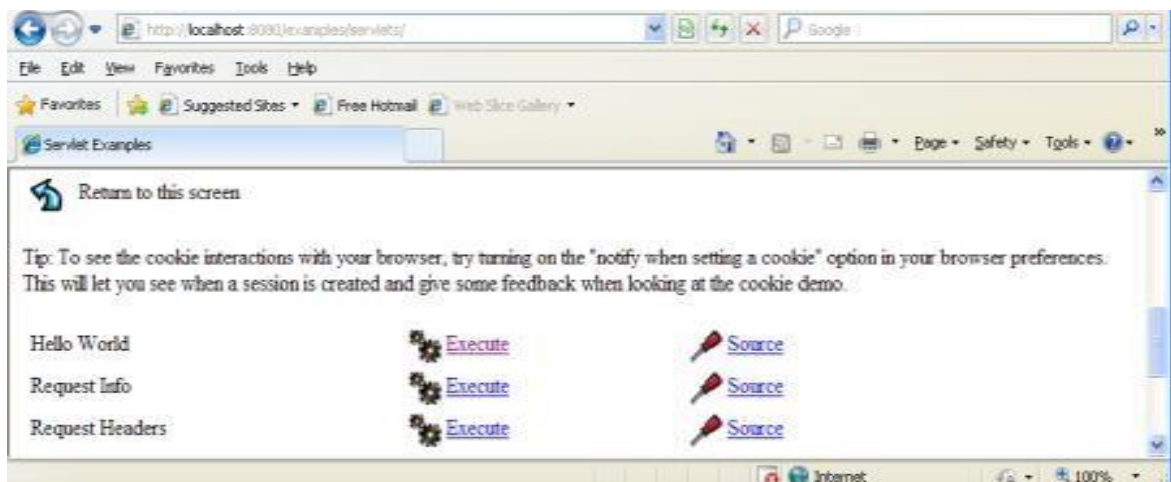
Miscellaneous

[Contact](#)
[Legal](#)
[Sponsorship](#)
[Thanks](#)

Apache Software Foundation

[Who We Are](#)
[Heritage](#)
[Apache Home](#)
[Resources](#)

Copyright ©1999-2022 Apache Software Foundation. All Rights Reserved.



```

1 // To save as "<CATALINA_HOME>\webapps\helloworld\WEB-INF\src\mypkg\HelloWorldExample.java"
2 package mypkg;
3
4 import java.io.*;
5 import java.util.*;
6 import javax.servlet.*;
7 import javax.servlet.http.*;
8
9 public class HelloWorldExample extends HttpServlet {
10     @Override
11     public void doGet(HttpServletRequest request, HttpServletResponse response)
12         throws IOException, ServletException {
13         // Set the response message's MIME type.
14         response.setContentType("text/html;charset=UTF-8");
15         // Allocate a output writer to write the response message into the network socket.
16         PrintWriter out = response.getWriter();
17
18         // Use a ResourceBundle for localized string in "LocalStrings_xx.properties" for i18n.
19         // The request.getLocale() sets the locale based on the "Accept-Language" request header.
20         ResourceBundle rb = ResourceBundle.getBundle("LocalStrings", request.getLocale());
21         // To test other locales.
22         //ResourceBundle rb = ResourceBundle.getBundle("LocalStrings", new Locale("fr"));
23
24         // Write the response message, in an HTML document.
25         try {
26             out.println("<!DOCTYPE html>"); // HTML 5
27             out.println("<html><head>");
28             out.println("<meta http-equiv='Content-Type' content='text/html; charset=UTF-8'>");
29             String title = rb.getString("helloworld.title");
30             out.println("<title>" + title + "</title></head>");
31             out.println("<body>");
32             out.println("<h1>" + title + "</h1>"); // Prints "Hello, world!"
33             // Set a hyperlink image to refresh this page
34             out.println("<a href='" + request.getRequestURI() + "'><img src='images/return.gif'></a>");
35             out.println("</body></html>");
36         } finally {
37             out.close(); // Always close the output writer
38         }
39     }
40 }

```




```

package mypkg;

import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import myutil.HtmlFilter;

public class SessionExample extends HttpServlet {
    @Override
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws IOException, ServletException {
        // Set the response message's MIME type
        response.setContentType("text/html;charset=UTF-8");
        // Allocate a output writer to write the response message into the network socket
        PrintWriter out = response.getWriter();

        // Use ResourceBundle to keep localized string in "LocalStrings_xx.properties"
        ResourceBundle rb = ResourceBundle.getBundle("LocalStrings", request.getLocale());

        // Write the response message, in an HTML page
        try {
            out.println("<!DOCTYPE html"); // HTML 5
            out.println("<html><head>");
            out.println("<meta http-equiv='Content-Type' content='text/html; charset=UTF-8'>");
            String title = rb.getString("sessions.title");
            out.println("<head><title>" + title + "</title></head>");
            out.println("<body>");
            out.println("<h3>" + title + "</h3>");

            // Return the existing session if there is one. Otherwise, create a new session
            HttpSession session = request.getSession();

            // Display session information
            out.println(rb.getString("sessions.id") + " " + session.getId() + "<br />");
            out.println(rb.getString("sessions.created") + " ");
            out.println(new Date(session.getCreationTime()) + "<br />");
            out.println(rb.getString("sessions.lastaccessed") + " ");
            out.println(new Date(session.getLastAccessedTime()) + "<br /><br />");

            // Set an attribute (name-value pair) if present in the request
            String attName = request.getParameter("attribute_name");
            if (attName != null) attName = attName.trim();
            String attValue = request.getParameter("attribute_value");
            if (attValue != null) attValue = attValue.trim();
            if (attName != null && !attName.equals(""))
                && attValue != null && !attValue.equals("") ) {

```

```

        // synchronized session object to prevent concurrent update
        synchronized(session) {
            session.setAttribute(attName, attValue);
        }
    }

    // Display the attributes (name-value pairs) stored in this session
    out.println(rb.getString("sessions.data") + "<br>");
    Enumeration names = session.getAttributeNames();
    while (names.hasMoreElements()) {
        String name = (String) names.nextElement();
        String value = session.getAttribute(name).toString();
        out.println(HtmlFilter.filter(name) + " = "
            + HtmlFilter.filter(value) + "<br>");
    }
    out.println("<br />");

    // Display a form to prompt user to create session attribute
    out.println("<form method='get'>");
    out.println(rb.getString("sessions.dataname"));
    out.println("<input type='text' name='attribute_name'><br />");
    out.println(rb.getString("sessions.datavalue"));
    out.println("<input type='text' name='attribute_value'><br />");
    out.println("<input type='submit' value='SEND'>");
    out.println("</form><br />");

    out.print("<a href=''");
    // Encode URL by including the session ID (URL-rewriting)
    out.print(response.encodeURL(request.getRequestURI() + "?attribute_name=foo&attribute_value=bar"));
    out.println(">Encode URL with session ID (URL re-writing)</a>");
    out.println("</body></html>");
} finally {
    out.close(); // Always close the output writer
}
}

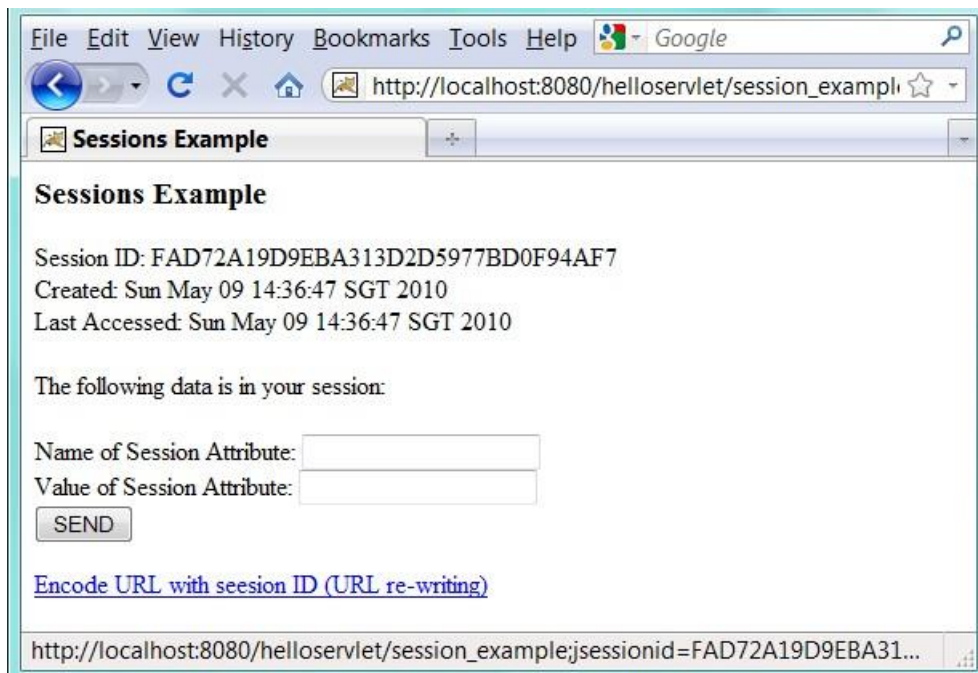
// Do the same thing for GET and POST requests
@Override
public void doPost(HttpServletRequest request, HttpServletResponse response)
    throws IOException, ServletException {
    doGet(request, response);
}
}

```

```

<servlet>
    <servlet-name>TomcatSessionExample</servlet-name>
    <servlet-class>mypkg.SessionExample</servlet-class>
</servlet>
.....
<servlet-mapping>
    <servlet-name>TomcatSessionExample</servlet-name>
    <url-pattern>/session_example</url-pattern>
</servlet-mapping>

```



Observation(20)	
Record(5)	
Total(25)	
Initial	

Result:

Thus the given task is completed successfully.

G.Ghulam Dasthageer
19CSE041

