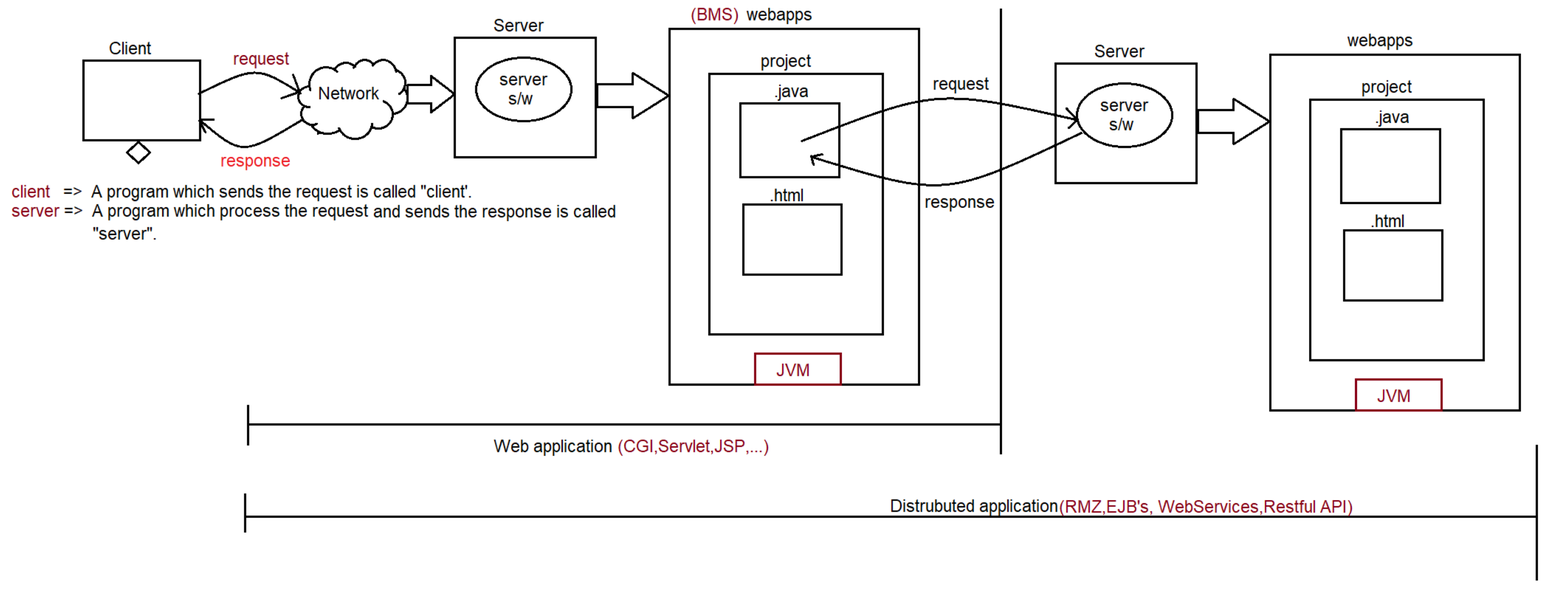
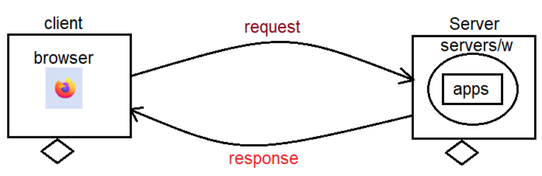
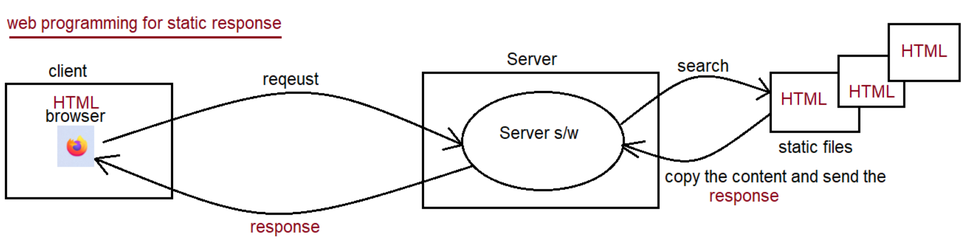


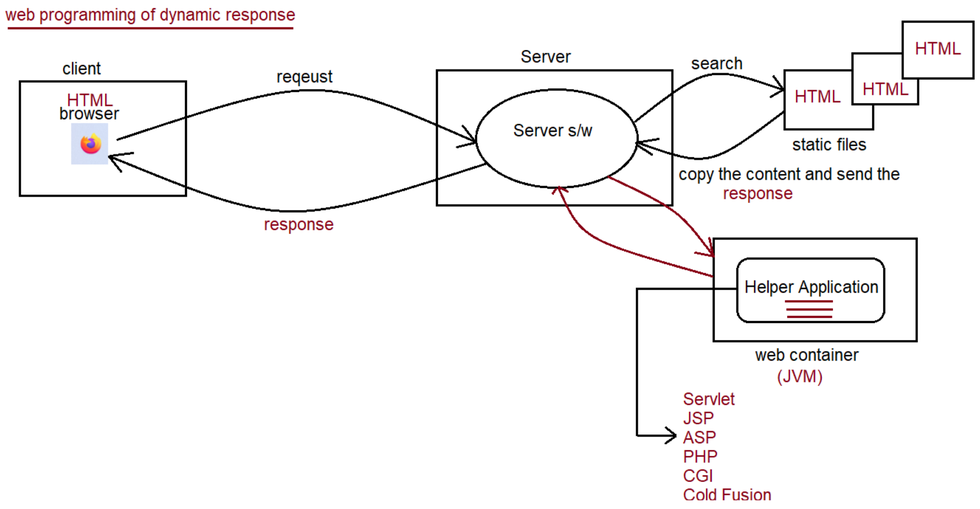
Client-Server Archietcture



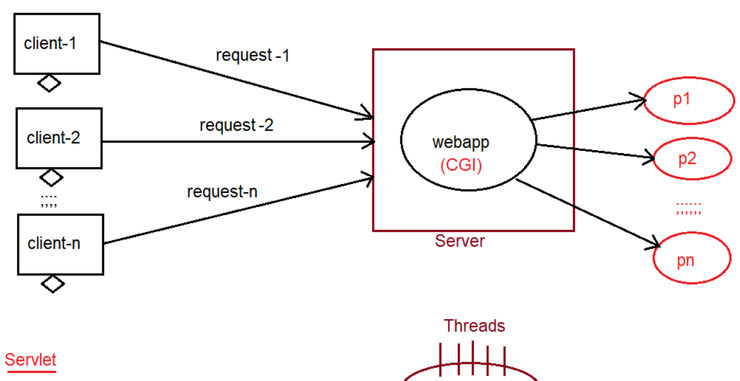
web programming for static response



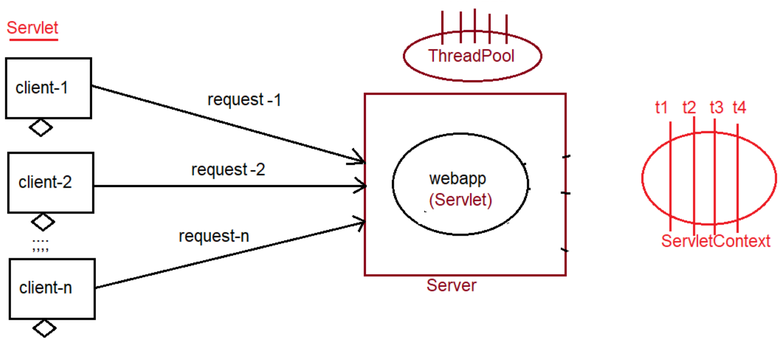
web programming of dynamic response



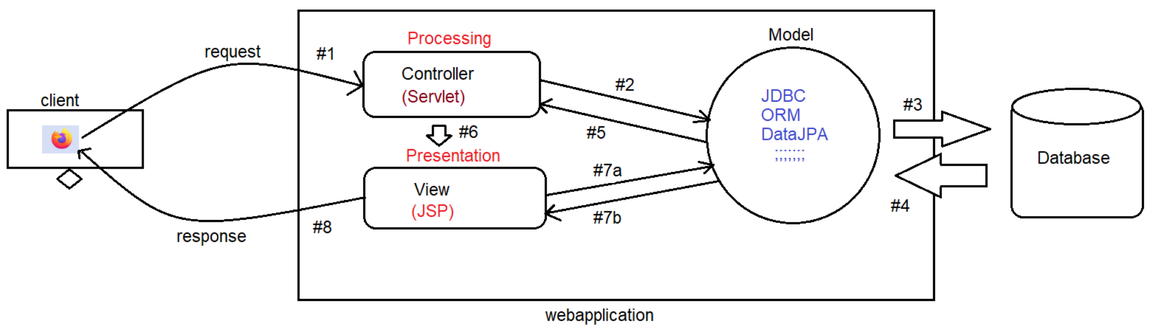
CGI



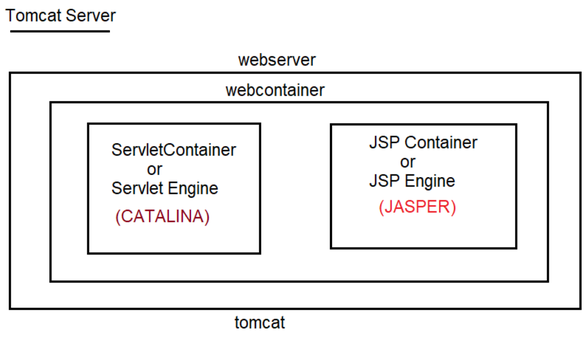
Servlet

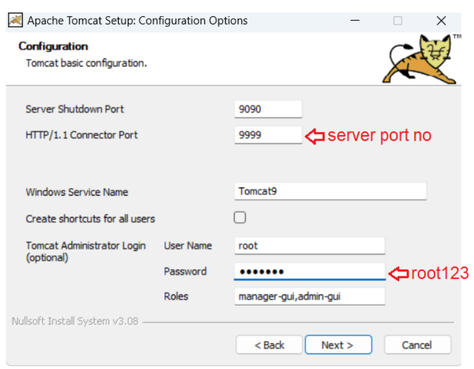


MVC



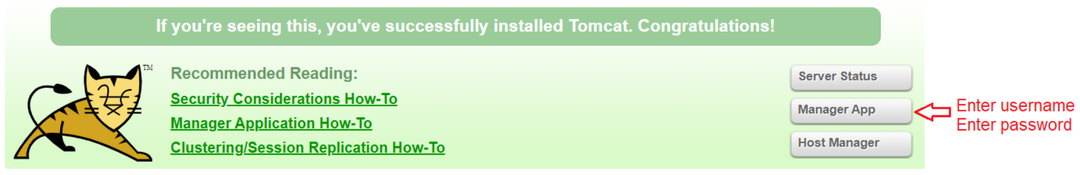
Tomcat Server

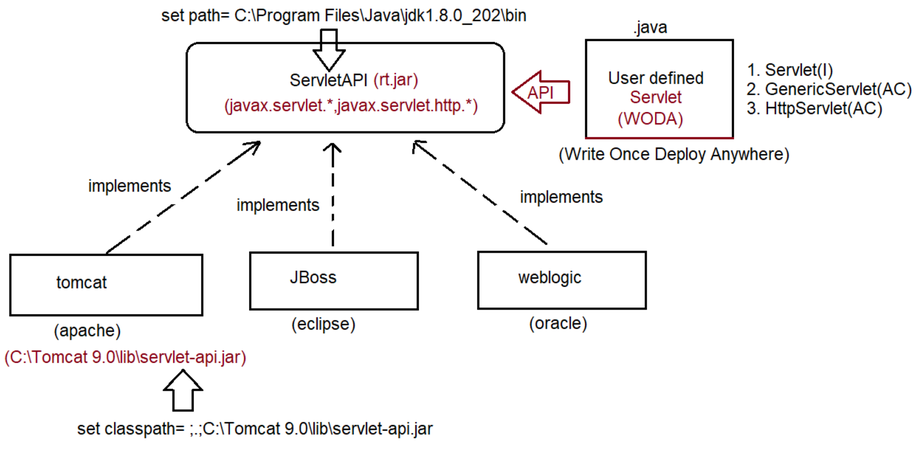




folder structure of tomcat

- bin ⇦ all executable files(.exe)
- conf ⇦ configruation files
- lib ⇦ .jars(related to technology)
- logs ⇦ (request,response details on the user)
- temp ⇦ (temporary files)
- webapps ⇦ Deployment folder(all our projects should be saved here)
- work ⇦ (JSP-SERVLET execution folder)





Tommorrow Session(25/01/2023) => From 8.00PM to 11.00PM

Different types of application built using java
==========
 a. Standalone applications(JSE)
            1. CUI applications
            2. GUI applications
 b. Enterprise applications(JEE and frameworks)
            1. Web applications
            2. Distrubuted applications

Difference b/w Web applications vs Distrubuted applications?

webapplication
==========
webapplication is a server side application, it will be designed without distrubuting application logic over multiple jvm.
To build webapplication we need to use technologies like CGI,Servlet,JSP and so on...
The main purpose of webapplication is to generate dynamic response from server machine.
webapplication provides services to webclients only(using the browser we need to send the request)
            eg: BMS(webapp) -------> to send the request to this application u need browser + internet
                                                mobile based apps(internet)
                  Webapplication => client to server model
In case of webapplications to execute the program we need "webservers".

Distrubuted application
==========
Distrubutedapplication is a server side application, it will be designed by distrubuting application logic over multiple jvm.
To build distrubuted application we need to use technolgies like RMI(remote method invocation), EJB's, WebServices,.....
The main purpose of distrubuted application is to establish the communication b/w local machine and remote machine
to access the remote services.
Distrubuted applications will provide service to any type of clients.
                  Distrubutedapplication => buisness to buisness model.
In case of distrubutedapplications to execute the program we need "application servers".

webapplication
==========
      The application which is developed only using web based technologies like html,css,javascript,servlet,jsp etc is
      called "webapplication"

application ===> collection of many programs

webserver
==========
if we want webapplications to execute, we need one special software that special software only we call it as "webserver".
webserve provides environment to run webapplications
            eg: tomcat,resin,jetty,glassfish,jboss,oracleweblogic server,.....

Deployment and UnDeployment

==========
The process of placing the webapplication inside webserver is called "Deployment".

The process of removing webapplication from the webserver is called "UnDeployment".

webclient
========
To send the request from the user we need to have special software installed in the client machine.
The special software is only called as "browser".
eg: mozilla,chrome,safari,.....

web programming for static response
==========
static response
The response which won't be changed from person to person and time to time, such type of response is called as
"static response".
eg: login page of gmail
home page of icici bank

dynamic response
The response which is varied from person to person and time to time, such type of response is called as
"dynamic response".
eg: inbox page of gmail
balance information of icici bank.

Flow diagram of static response
==========
1. client send the request for static files to the server
2. Server searches wheter the requested resource(html file) is available or not.
3. If the request resource(html file) is available then server will provide that file as response.
4. If the requested resource(html file)is not available then we will get 404 status code saying reqeusted resource not available.

Note: To server static files, no processing is required at the server side, hence webserver always loves to serve static files.

web programming for dynamic response
==========
1. client sends the request for webserver
2. webserver will check whether the request is for static resource or dynamic resource(based on url)
3. if it is a static resource, then webserver only will search for static resource,if it is available server will provide the static file
contents(copy and paste) as the response the client.
if it is not available, then 404 status code will be sent as the response to the client saying the requested resource is not
available.

4. if the requested resource is for dynamic information, then webserver will forward the request to webcontainer.
5. webcontainer will search for the helper application, which needs to be executed.
6. if it is not available, then 404 status code will be sent as the response to the client saying the requested resource is not

available.
6. if it is available, then the requested helper application will be executed and it will be sent as the response to the webserver
and webserver inturns will send as the response to the end user.
7. During the execution if any problem occurs then it would reuslt in exception and status code 500 would be sent as a
response to the end user by the server.

Note: To generate the dynamic response at the server side we need some helper applications. To build these applications
which are capable of generating dynamic response we need to learn technologies like

a. CGI
b. Servlet
c. JSP

To design a webapplication, we already have CGI then what is the need to go for Servlet?

CGI => it stands for Common Gateway Interface.
CGI is a server side webtechnology which is built on top of 'c' language, c language is a process based language, which inturn
make CGI as "process based technology".
if we deploy any CGI application,then container will create a seperate process for every request.
Process is heavy weight component, to handle single process system has to consume a lot of memory and execution time.
Due to the above reason, more the request comes server would be getting a burden of creating a process which inturn reduce
the system performance and increase the response time for the client.
To reduce the burden on server and to increase the performance we need to use server side technology called "Servlet".

Servlet is a server side technology which is built on top on "Java language".
Java language is Thread based technology.
if we deploy a Servlet application at the server side then for every request servlet container will generate a seperate thread
on the respecitve Servlet Object.
In the above context, if we increase the no of requests container will create a separte thread instead of process.
When compared to process,threads are light weight,since it is light weight, server would not be burdened.
server would provide quick response for client requests which increase the peformance of the application.

To design a webapplication, we already have Servlet then what is the need to go for JSP?

Servlet
1. To build web applications using Servlet, we need to have knowledge of Java properly.
2. Servlet is mainly meant for Processing logic(pick the request and process the request).
3. Any modification done in the Servlet, then we need to perform compilation and reloading on the server explicitly.
4. If we build webapps using MVC design pattern, then Servlet will placed inside Controller logic.
5. In case of Servlet, we are unable to seperate both presentation logic and buisness logic.

JSP

1. To build web applications using JSP, it is not required to have any java knowledge only presentation skills are enough.
2. JSP is mainly meant for providing dynamic response to the client with good look and feel (only presentation).
3. Any modification done in the JSP, then it is not required to do compilation and reloading becoz jsp pages are "Autocompiled".
4. If we build webapps using MVC design pattern, then JSP will placed inside View logic.
5. In case of JSP, there will be a clear cut seperation b/w presentation logic and buisness logic becoz presentation logic deals with html tags and buisness logic deals with "JSP tags".

Architecture of webserver(tomcat)
==========
Tomcat(https://tomcat.apache.org/download-90.cgi)
1. It is webserver provided by apache foundation.
2. Every webserver will have webcontainer
   a. webcontainer is responsible to manage and execute servlet and jsps.
3. Internally webcontainer consists of 2 components
   a. catalina container(servlet container)
   b. jasper container(jsp container)
4. ServletContainer
   It is also known as ServletEngine.
   It is responsible for managing and executing servlet components.
   Tomcat servlet container name is "CATALINA".
5. JSP container
   It is also known as JSP Engine
   It is responsible for managing and exeucting jsp componenets
   Tomcat jsp container name is "JASPER".

Note:
start the tomcat server by opening bin folder and select tomcat9.exe and double click on it.
now send the request by opening browser of ur choice and hit the request as shown below
   http://localhost:9999/

Servlet
======
   It is an API which helps the programmer to build webapplications.
   ServletAPI provides 2 packages
      a. javax.servlet.*
      b. javax.servlet.http.*

javax.servlet.*
==========
   1. Servlet(I)
   2. GenericServlet(AC)
   3. ServletConfig(I)
   4. ServletContext(I)
   5. RequestDispatcher(I)
   6. ServletRequest(I)
   7. ServletResponse(I)

javax.servlet.http.*
==========

1. HttpServletReqeust(I)
2. HttpServletResponse(I)
3. HttpSession
4. HttpServlet(AC)

Note:

```
Compiled from "Servlet.java"
public interface javax.servlet.Servlet {
  public abstract void init(javax.servlet.ServletConfig) throws
javax.servlet.ServletException;
  public abstract javax.servlet.ServletConfig getServletConfig();
  public abstract void service(javax.servlet.ServletRequest,
javax.servlet.ServletResponse) throws javax.servlet.ServletException,
java.io.IOException;
  public abstract java.lang.String getServletInfo();
  public abstract void destroy();
}

C:\Users\nitin>javap javax.servlet.GenericServlet
Compiled from "GenericServlet.java"
public abstract class javax.servlet.GenericServlet implements
javax.servlet.Servlet,javax.servlet.ServletConfig,java.io.Serializable {
  public javax.servlet.GenericServlet();
  public void destroy();
  public java.lang.String getInitParameter(java.lang.String);
  public java.util.Enumeration<java.lang.String> getInitParameterNames();
  public javax.servlet.ServletConfig getServletConfig();
  public javax.servlet.ServletContext getServletContext();
  public java.lang.String getServletInfo();
  public void init(javax.servlet.ServletConfig) throws
javax.servlet.ServletException;
  public void init() throws javax.servlet.ServletException;
  public void log(java.lang.String);
  public void log(java.lang.String, java.lang.Throwable);
  public abstract void service(javax.servlet.ServletRequest,
javax.servlet.ServletResponse) throws javax.servlet.ServletException,
java.io.IOException;
  public java.lang.String getServletName();
}

C:\Users\nitin>javap javax.servlet.http.HttpServlet
Compiled from "HttpServlet.java"
public abstract class javax.servlet.http.HttpServlet extends
javax.servlet.GenericServlet {
  public javax.servlet.http.HttpServlet();
  protected void doGet(javax.servlet.http.HttpServletRequest,
javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletException,
java.io.IOException;
  protected long getLastModified(javax.servlet.http.HttpServletRequest);
  protected void doHead(javax.servlet.http.HttpServletRequest,
javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletException,
java.io.IOException;
  protected void doPost(javax.servlet.http.HttpServletRequest,
javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletException,
java.io.IOException;
  protected void doPut(javax.servlet.http.HttpServletRequest,
```

```
javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletException,
java.io.IOException;
  protected void doDelete(javax.servlet.http.HttpServletRequest,
javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletException,
java.io.IOException;
  protected void doOptions(javax.servlet.http.HttpServletRequest,
javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletException,
java.io.IOException;
  protected void doTrace(javax.servlet.http.HttpServletRequest,
javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletException,
java.io.IOException;
  protected void service(javax.servlet.http.HttpServletRequest,
javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletException,
java.io.IOException;
  public void service(javax.servlet.ServletRequest, javax.servlet.ServletResponse)
throws javax.servlet.ServletException, java.io.IOException;
  static {};
}
```