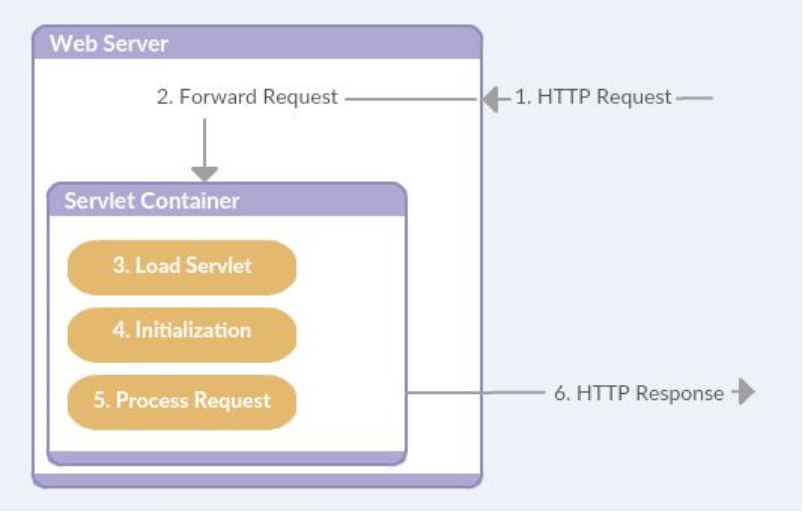
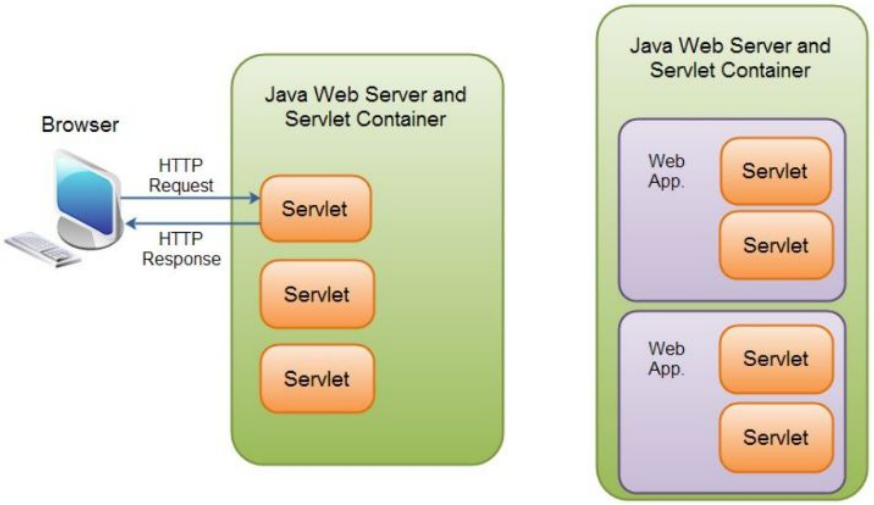
Laborator 13

Java Servlets

- Un servlet Java este o clasa care se ocupă cu procesarea și răspunsul cererilor de la un client
- Preia cereri HTTP de la browser-ul Web şi generează dinamic un răspuns (HTML sau XML).
- Servleturile rulează în interiorul unui servlet container, care reprezinta punctul intermediar dintre requesturile de la client și servleturi



Un servlet face parte dintr-o aplicație web. Un servlet container poate sa ruleze mai multe aplicații web în același timp, fiecare aplicație avand mai multe servleturi rulând în interior.



Pe langa Servlets, o aplicație java poate sa contina si alte componente: Java Serve Pages (JSP) și servicii web.

Http Request / Response

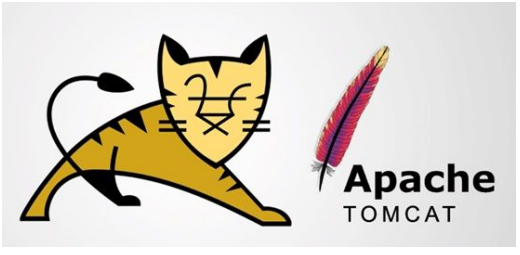
Serverul web poate recunoaște dacă o cerere corespunde unei aplicații web din containerul web.
Dacă cererea este pentru containerul web aceasta este direcționată către acesta. Cand containerul web a primit cererea trebuie să decidă ce aplicație web trebuie sa o trateze.

O cerere HTTP poate fi asociată unui servlet, unei pagini JSP (Java Server Pages) sau oricărei resurse statice (resursele statice includ pagini HTML/XML, imagini și applet-uri)

În baza informației de asociere din cererea HTTP containerul web determina dacă cererea trebuie prelucrata de un servlet. Dacă da se creaza o instanță a acestuia și i se transmite parametrii cererii.
Răspunsul de la server o sa vine sunt forma unui obiect de tipul HttpResponse.

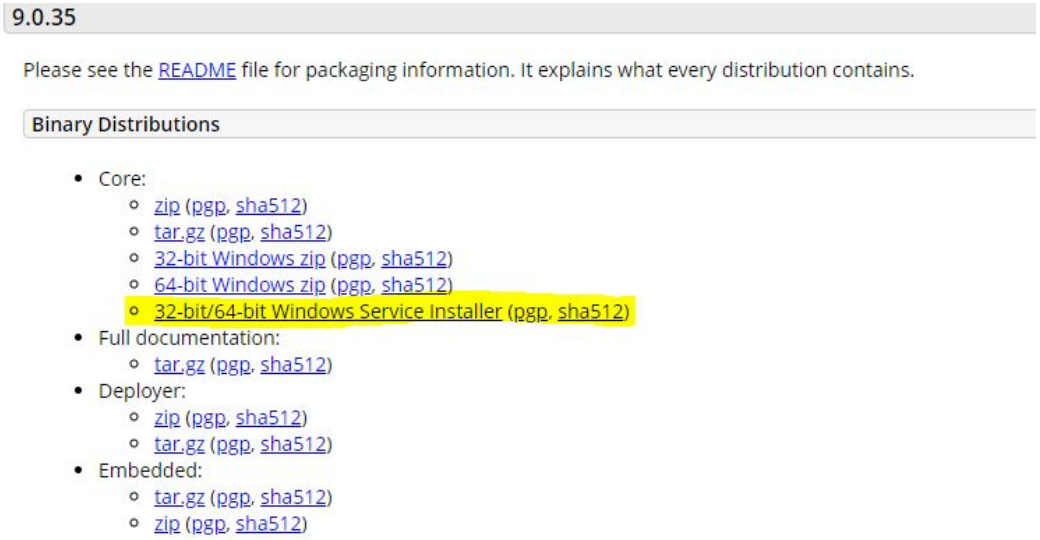
Servlet Containers

Un servlet container în general rulează în interiorul unui server web java, ca de exemplu : Tomcat, Jetty, GlassFish etc.

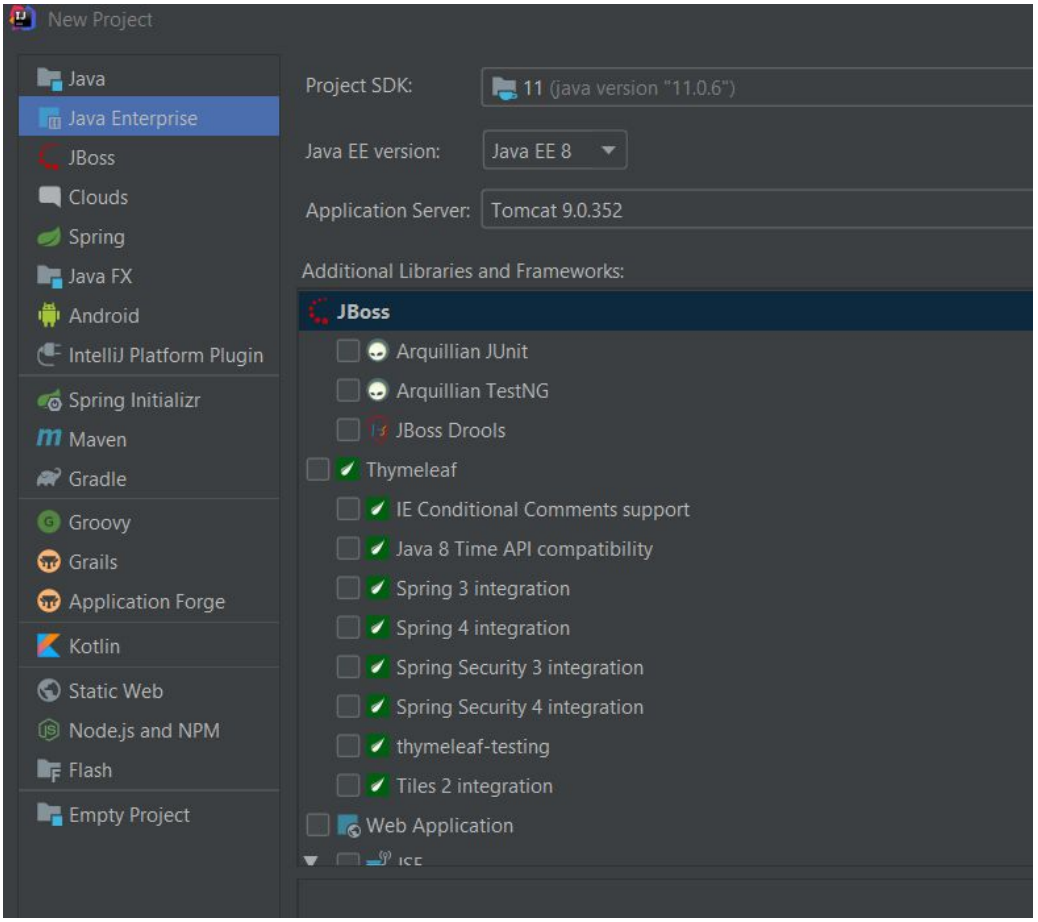


Pașii de instalare pentru Tomcat web server:

1. Accesati http://tomcat.apache.org/ si descarcati installerul pentru Tomcat 9.0



2. Instalați Tomcat local (sa nu lasati exit port pentru tomcat -1)
3. În IDE-ul vostru creați un proiect Java Enterprise și selectați application server-ul sa fie Tomcat-ul instalat de voi.



Un servlet are un life cycle specific, care este manageuit de servlet container:

1. Clasa Servlet este incarcata.
2. Se creaza o instanță a servletului.
3. Se apelează metoda init() a servletului.
4. Se apelează metoda service() a servletului.
5. Se apelează metoda destroy a servletului.

Un servlet Java este o clasa obișnuită care extinde clasa **HttpServlet**.

Clasa HttpServlet citește request-ul HTTP și determina dacă este un request de tipul GET, POST, PUT, DELETE etc și apelează metoda corespunzătoare. Ca sa răspunzi la un request de tipul HTTP GET, va trebui extinsă clasa HttpServlet și suprascrisa metoda doGet().

```
* The doPost takes two parameters (login and password) and displays them.
*/
@WebServlet(name = "Servlet", urlPatterns={"/Servlet"})
public class SimpleServlet extends HttpServlet {

   protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws IOException {
        response.setContentType("text/html");
        PrintWriter writer = response.getWriter();
        writer.println("This resource is not available directly.");
   }

   protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws IOException {
        PrintWriter out = response.getWriter();
        response.setContentType("text/html");
        out.println("<html><head></head><body>");
        String login = request.getParameter("login");
        String password = request.getParameter("password");
        out.println("<h1>Super Secret Login Information</h1>");
        out.println("<p>Login: " + login + "</p>");
        out.println("<p>Password: " + password + "</p>");
        out.println("</body></html>");
   }
}
```

```html
<!DOCTYPE html>
<html lang="en">
   <head>
       <meta charset="UTF-8">
       <title>Title</title>
   </head>
   <body>
       <form action="Servlet" method="post">
           <p>Login<input name="login" type="text"/></p>
           <p>Password<input name="password" type="password"/></p>
           <p><input type="submit" value="Login"/></p>
       </form>
   </body>
</html>
```

JavaServer Pages (JSP)

- JavaServer Pages (SP) facilitează crearea de conținut web atât static cât și dinamic.
- Ideea paginilor JSP este de a extinde paginile HTML cu fragmente de cod Java incluse între tag-uri delimitate de “<%” și “%>”.
- Paginile JSP sunt transformate (compilate) automat în servleti de către un motor JSP (JSP Engine).

Sintaxa JSP

Putem sa adaugam cod java într-un fișier .jsp în doua feluri:

1. Putem sa folosim sintaxa standard pentru Java Scriptlet care presupune sa adaugam cod java între doua tag-uri de scriptlet:

```
<% Java code here %>
```

2. A doua metoda este specific pentru XML:

```
<jsp:scriptlet>
    Java code here
</jsp:scriptlet>
```

```
<% if (condition) {%>
    <div>Hei!</div>
<% } else { %>
    <p>Hello!</p>
<% } %>
```

Resurse:

Pentru mai multe detalii puteti accesa link-ul urmator https://www.baeldung.com/jsp

```
package servlet;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

@WebServlet(name = "SecondServlet", description = "JSP Servlet With
Annotations", urlPatterns = {"/SecondServlet"})
public class SecondServlet extends HttpServlet {

   @Override
   protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        String message = request.getParameter("message");
        request.setAttribute("text", message);
        request.getRequestDispatcher("/secondpage.jsp").forward(request,
response);
   }
}
```

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
   <head>
       <title>Java Binding Example</title>
   </head>
   <body>
       <h1>Bound Value</h1>
       <p>You said: ${text}</p>
   </body>
```

```
</html>
```

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
   <head>
       <title>$Title$</title>
   </head>
   <body>
       Hello World
       <p>To invoke the java servlet click <a
href="${pageContext.request.contextPath}/Servlet">here</a></p>
       <p>Java in static page: <a href="firstpage.jsp"
target="_blank">here</a></p>
       <p>Java injected by Servlet: <a href="SecondServlet?message=hello!"
target="_blank">here</a></p>
   </body>
</html>
```