# Chap 3 – What are Servlets?

* What is a Servlet?

A servlet is a Java program that runs on a Web Server. It is executed by the user of the Application by clicking on a Submit button in an HTML Web Page. Any java code can be added to a Servlet.

The Web Server executes a Servlet by calling 3 methods(called the life-cycle of a Servlet): init(), service() and destroy(). These 3 methods are called by the Web Server. So as a programmer you put your code in these methods. Usually you put your code in the service() method. Init() is used to initialize any needed data for this Servlet. And destroy() is used to close down any resources used by this Servlet.

In NetBeans you actually put your code in a method named processRequest(). This method is called by the service() method that you inherit from HttpServlet, a super class that is used to build servlets.

There are actually 2 arguments passed into the service() method. They are request and response. Data come in in the request object and your answer goes out through the response object. service(request, response)……

* How do you build a Servlet?

In NetBeans you right-click on the name of your Project located on the left hand side of NetBeans in the Project Manager window. Once you right-click, choose add Servlet. It will ask you to name your Servlet, then it will create you a servlet template. You add your code to this code template. Change whatever you need to.

If you were to build a servlet without Netbeans, you would have 3 choices:

1. class MyServlet implements Servlet {

Then you would have to build the 3 methods for a servlet init()…service()…and destroy(). Put your code you want to run in service().

1. Class MyServlet extends GenericServlet {

GenericServlet give you the 3 servlet methods init(), service() and destroy(). You just build one method in your class service() and add the code you want to run.

1. Class MyServlet extends HttpServlet {

HttpServlet give you the 3 servlet methods init(), service() and destroy(). The service() method calls doGet() and doPost() to retrieve the data from the end user. So you put your code in one of these 2 methods, whichever you want to use, or you can have one of the methods call the other so either method would then work.

doPost() { doget(); {

doGet() { …..servlet code goes here………}

* How do you call a Servlet?

There are 3 ways to execute a Servlet.

1. In NetBeans you can simply right-click anywhere in the Servlet code and select run-file.
2. You can go to a Browser and type in a URL:

<http://localhost:8080/projname/servletname>

Where 8080 is the web-server that you are using. I think 8080 is Tomcat and 8084 is glassfish.

1. From an HTML page, you can add a Submit button to a Web Form and then in the Form tag you add an ‘action=’ attribute.

Ex: <FORM action=”LoginServlet” method = “post’ >

* Complete Servlet Example:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Java III - Servlet Example

\* Ron Enz - Fall 2019

\* MyServlet.java

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

@WebServlet(name = "MyServlet", urlPatterns = {"/MyServlet"})

public class MyServlet extends HttpServlet {

/\*===================== processRequest() Method======================\*/

protected void processRequest(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

PrintWriter out = response.getWriter(); //out is used to send html page back to user

String id;

try {

id = request.getParameter("idTB"); // read data from previous HTML page

if (id.equals("admin")) {

// Sending html string to client browser

out.println("<html>");

out.println("<head>");

out.println("<title>Admin Logging in............</title>");

out.println("</head>");

out.println("<body>");

out.println("<h1>Valid</h1>");

out.println("</body>");

out.println("</html>");

}

else{

// Sending html string to client browser

out.println("<html>");

out.println("<head>");

out.println("<title>Unknown login attempt.......</title>");

out.println("</head>");

out.println("<body>");

out.println("<h1>InValid</h1>");

out.println("</body>");

out.println("</html>");

}

}

finally {

out.close();

} //end finally/try

} //end processRequest()

/\*============================= doGet() Method===================\*/

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

// doGet() calls processRequest()

processRequest(request, response);

}

/\*============================== doPost() Method===================\*/

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// doPost() calls processRequest()

processRequest(request, response);

}

public String getServletInfo() {

return "Short description";

}

}

***Servlet Tutorials:***

<https://www.geeksforgeeks.org/introduction-java-servlets/>

<https://www.baeldung.com/intro-to-servlets>

<https://www.javatpoint.com/servlet-tutorial>