

# COMP 3095 – Lab 2

## Part I - Customize Servlet URL

---

This tutorial is an additional work of the previous lab. It allows for a Servlet to be accessed by different URLs.

1. Open the **servletlabs** project in the previous tutorial.
2. Create a file: **web.xml** in the **WebContent** → **WEB-INF** folder
3. Insert the following code into the **web.xml** file:

```
<web-app>
  <servlet>
    <servlet-name>HelloServ</servlet-name>
    <servlet-class>first.servlet.HelloWorld</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>HelloServ</servlet-name>
    <url-pattern>/helloworld</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
    <servlet-name>HelloServ</servlet-name>
    <url-pattern>/hello_world</url-pattern>
  </servlet-mapping>
</web-app>
```

4. The tag **<servlet-name>** defines your servlet; **<servlet-class>** points to the class that implements the servlet and **<servlet-mapping>** declares the mapping between your servlet with different URLs.
5. Save the file.
6. Right click on the **servletlabs** project, select **Run As** → **Run on Server**. **Finish**.
7. Select **Restart server**. **Finish**.
8. Now, the “**HelloWorld**” servlet can be accessed by different URLs:
  - <http://localhost:8080/servletlabs/HelloWorld>
  - <http://localhost:8080/servletlabs/helloworld>
  - [http://localhost:8080/servletlabs/hello\\_world](http://localhost:8080/servletlabs/hello_world)

## Exercises

---

1. Add another mapping to allow your servlet can be accessed by the URL: [http://localhost:8080/servletlabs/\[whatever you type here\]](http://localhost:8080/servletlabs/[whatever you type here]).

## Part II - Servlet: handling a POST request

---

In this second part of this tutorial, we will create a simple HTML page which sends a POST request, including a username and a password to the "HelloWorld" servlet. The "HelloWorld" servlet will printout the received information.

### Create a simple HTML page

---

1. Open the **servletlabs** project.
2. In the folder **WebContent**, create an **form.html** file with the following content:

```
<html>
<body>
  <form method="post" action="/HelloWorld">
    <input type="text" name="username" />
    <input type="password" name="password" />
    <input type="submit" />
  </form>
</body>
</html>
```

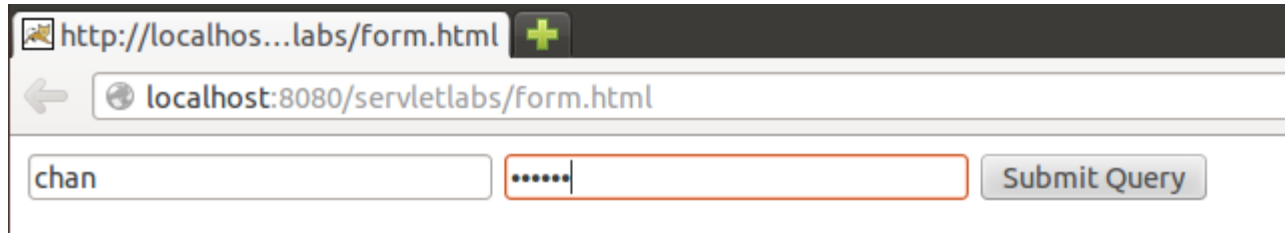
3. This form allows users to submit a username and a password to the **/HelloWorld** servlet.
4. Open the **HelloWorld.java** file, modify the **doPost(...)** function as the following:

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

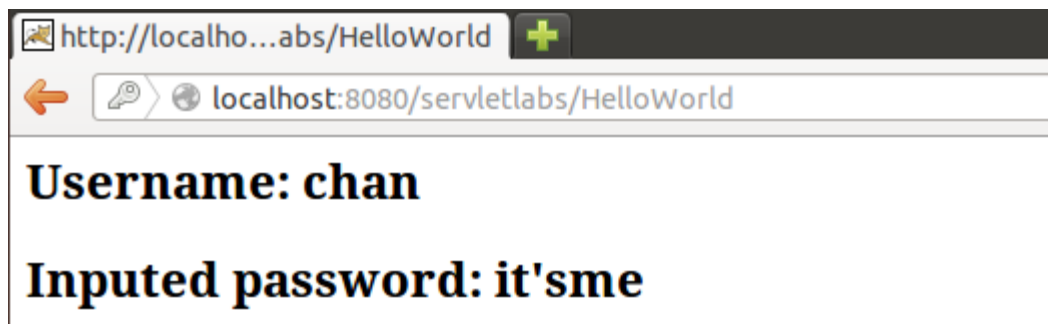
    String userName = request.getParameter("username");
    String password = request.getParameter("password");

    response.setContentType("text/html");
    PrintWriter pw = response.getWriter();
    pw.println("<h2>");
    pw.println("Username: "+userName);
    pw.println("</h2><h2>");
    pw.println("Inputed password: "+password);
    pw.println("</h2>");
    pw.println("");
}
```

5. This function handle user's POST requests.
6. Now, open <http://localhost:8080/servletlabs/form.html> on your web browser.
7. Type a **username** and a **password**. Then click on **SubmitQuery**.



8. The "HelloWorld" servlet will execute the **doPost** function and return the received username/password.



## Exercise

---

1. Inspired from this tutorial, create another form that send an integer number to a Servlet and receives back the factorial of this number. Use GET method instead of POST.
2. Instead of using the form as in exercise 1, how do we send a GET request to the server by using URL?
3. Create another form that contains two text fields and two radio buttons named "rectangle" and "eclipse". Then create a Java class that computes the area of a rectangle and an eclipse based on two input numbers. Finally, create a servlet that receives two numbers and an option from user, computes the corresponding area and returns the result to the user.
4. (*\*This is a bonus only*) Create a Java class that could interact with a database system (such as MySQL). Add to the database a list of student names. Implement a servlet and an HTML form that allows to list, add, update and delete a student name.