

# Lab 3 – JavaScript and Servlets

**Note:** This lab follows the work you have done in the previous lab. Therefore, before starting this lab, please ensure that you have done as much as possible of “Lab 2 – Sockets and HTML”.

## 1. Hosting an HTML page

Install Tomcat and host the web page shown in **Figure 2** of **Lab 2** (you can find this at the end of **Lab 2** sheet). Use the instructions given in the **EBU5042** course area in QMplus, under the **BLOCK 3: BUPT WEEKS 12-14** topic:

- *EBU5042\_TB3\_Core Activity: Setting up Tomcat*

## 2. Servlets and JavaScript

Write the *Servlet* `HoroscopeServlet` that simply outputs,

`Hello <name>`

where `<name>` is the name input, and then outputs the *horoscope* according to the rules:

`male and Zodiac sign ≤6 → You will have a long life.`

`male and Zodiac sign >6 → You will have a rich life.`

`female and Zodiac sign ≤6 → You will find a tall handsome stranger.`

`female and Zodiac sign >6 → You will have six children.`

- a) Before generating an HTTP response, the *Servlet* must first check whether the name parameter is empty. If it is empty, then the *Servlet* should simply output the message:

`Please identify yourself (by indicating your name), so that  
your horoscope can be given!`

- b) Write the *Servlet*'s Deployment Descriptor, such that:

- The *Servlet*'s public URL name is `Horoscope`.
- The *Servlet*'s internal name is `Give Horoscope Servlet`.

- c) Modify the HTML (written by you previously for **Lab 2 – Q2.a)**) with some JavaScript that performs validation of the inputs. A user must:
- Enter his/her name (so it must not be left empty) and the name field can only be made up of letters and must start with a capital letter.
  - Select (i.e., check) a radio button for his/her gender.
- d) Instead of selecting the horoscope in the *servlet* `HoroscopeServlet`, make the *servlet* write a JavaScript function called `displayHoroscope(gender, sign)` that takes the user's gender and the sign of the Zodiac as parameters and can display the horoscope back to the client. The *servlet* simply then makes sure that the JavaScript function is invoked.
- e) **Questions:**
- What code is compiled and/or executed on the client side and on the server side of the web application?
  - How can validation of user data inputs be done with JavaScript?
  - After input validation is included in the HTML code, what happens if the user enters either nothing as a name, or enters invalid data (e.g., the text contains characters that are not letters)? And what happens if no radio button has been selected?
  - How does the *servlet* ensure that the response object is in the appropriate format when sending it to the client browser?
  - Briefly describe what changes you would have to make to the HTML page (and where) such that a user is now required to enter his/her *student College username* of the form `jp2020xxxxxx` (instead of the name) and validation of the user input data checks that the student College username field:
    1. Is not empty and,
    2. Starts with the sequence "jp2019" and
    3. Ends with a digit.

**Note:** The next page contains a list of typical errors that you may encounter when developing web applications, together with some suggestions as to the cause of the errors and thus, how to fix them.

**END of Lab 3: "JavaScript and Servlets"**

## Appendix: Typical Error Messages

**Error 1:**      `error: cannot read: src\com\example\web\PhoneSelect.java`

---

Possible causes of error:

- a) Your java file has a different name (e.g., `Phone.java`). This is common in web apps where you have one name for the HTML files, plus hidden names due to *servlet* mapping.
- b) You are compiling in the wrong directory. The compiler will look in the current directory for the directory `src`. In the standard deployment model, the `src` directory will be inside the **WEB-INF** directory.

**Error 2:**      `cannot find symbol`  
                 `symbol: class Iterator/List`

---

Your class does not **import** `java.util.*`;

**Error 3:**      `cannot find symbol`  
                 `symbol: class NameOfBusinessClass (e.g., PhoneExpert.java)`

---

Your class does not import the package e.g., **import** `com.example.model.*`;

**Error 4:**      `cannot find symbol`  
                 `symbol: class HttpServlet`

---

This class is inside the **servlet-api.jar** provided with Tomcat. Possible causes of error:

- a) You have not installed Tomcat.
- b) There is a typo (typing/spelling error) in your **classpath** instructions.
- c) You have not provided the complete path in your **classpath** instructions. For example, on a home computer this may look like:  
`C:\Program Files\Apache Software Foundation\Tomcat 6.0\lib\servlet-api.jar`  
**or**  
`C:\Program Files\Apache Software Foundation\Tomcat 6.0\lib\servlet-api.jar;classes;`

**Error 5:**      `Your browser cannot find your Servlet.do file`

---

Possible causes of error:

- a) Incorrect class path provided in the HTML file. It should be the full local path e.g., `localhost:8080/Phone/SelectPhone.do`.
- b) There is a typo in your deployment descriptor.

To test the validity of your deployment descriptor, double click on the XML file. This should open in a browser and tell you if there are any syntax errors.