Lab #11
CSCI 201

Title
Form Validation with Servlets and JavaScript

Lecture Topics Emphasized
HTML5
JavaScript
Servlets
JSON

Introduction
Accepting form submissions is one of the primary ways of getting information from users through a web page. However, since a user can type anything at all into form fields, we need to validate the data to ensure accuracy. You should implement Validation using a combination of both JavaScript and HTML5.

Description
Start with creating a *Dynamic Web Project* in Eclipse. Remember that all the frontend files go into the *WebContent* directory, so create a HTML file in the *WebContent* directory. Don't worry about including CSS in your page unless you have time and want to make it look a little nicer.

The HTML file is going to display an HTML Form with several different form fields. The content of the form can be whatever you want (user registration, items to purchase, types and descriptions of animals, etc.). However, you must have all the following form fields:

```
text             textarea         select box
radio buttons    month            email
range            submit           reset
```

Additionally, you must use all the following HTML elements and attributes:

```
required          checked           placeholder
fieldset          pattern
```

Make sure you uniquely identify each input by setting up their *name* attributes.

Submit all those fields to a servlet. You can specify the address (i.e., URL) of the servlet through the *action* attribute in the *form* tag. Note that you will also need to specify the HTTP method through the *method* attribute. We will be using the *GET* method for this lab.

Now, it is the time for you to create the servlet in the backend. Please note, in a *Dynamic Web Project*, all the backend files go into the *src* directory under *Java Resources*. In the *src* directory, you may create as many packages as you need to group the files. However, for this lab, packages are not necessary, as there will only be one backend file, which is your servlet. Create the servlet in the *src* directory.

Note that the *@WebServlet()* annotation defines the URL of the servlet. Make sure the *action* attribute of the form has that URL.

You can retrieve the form fields in the servlet through *HttpServletRequest* (i.e. the *request* variable). Here's a method that you may find helpful: *request.getParameter()*.

You should retrieve all the value for each parameter and construct a JSON object with the correct syntax.

You can use a library like GSON to help construct the JSON object, or you can put it together manually using concatenated strings.

The JSON object should be returned to the browser and should display in the browser window. The resulting object must be a "valid" JSON object. You can validate the JSON object by using JSONLint, the JSON validator, available here:

https://jsonlint.com/

You should have two different files for this lab – the initial form page with appropriate validation, and the servlet.

Grading Criteria

Labs are graded based on your understanding of the course material. To receive full credit, you will need to **1)** complete the lab following the instructions above **AND 2)** show your understanding of the lab material by answering questions upon check-off.

If there is a discrepancy between your understanding of the material and your implementation (i.e., if your code is someone else's work), you will receive a grade of **0** for the lab. Please note, it is the professor's discretion to report the incident to SJACS.

Check-off Questions

Instructors, please randomly select one question from each section.

Question 1

a) Out of the two files that we just created, which files are on the frontend and which are on the backend? Please explain how the frontend differs from the backend.
b) Out of the two files that we just created, which files are on the server side and which are on the client side? What are the types of file that typically run on the server side? What about client side?
c) In this lab, is the form validation done on the server or the client? Please explain.
d) Draw a flow chart to illustrate the relationship among the two files.

Question 2

a) Why do we need to specify the HTTP method? What is the other method typically used for form submission and what is the difference between them?
b) What happens if we don't have the *name* attribute for input fields?
c) What is the value of the PATTERN attribute?

Question 3

a) In the implementation above, where do we generate the error messages for invalid inputs?
b) How do we specify the desired servlet when validating forms?