Lab: Static Methods

In this lab, you shall create a page named "methods.jsp" that has the same layout as your home page (index.jsp). The page shall perform field level validation (similar to last lab) then perform some calculation (also similar to a previous lab), but this time it shall do so by calling a method that you write. Your jsp page shall ask the user for two sets of inputs (so that you see the value of being able to call a method passing in two different sets of variables as input parameters). For each set of input that passes all validation, your page shall label and display the calculation performed by the method. Otherwise (for each set of inputs that do not pass all validation), the field level error messages shall be displayed and the form level message shall be like “Please try again”.

The Method

- You shall write a method that is in a class that is in a package. The appendix of this lab provides help for how you add a package to your web app, how you add a class to your package and how you add a method into the class.
  - Do NOT call your package “myPkg”, your class “MyClass”, and your method “myMax” as I have done in the example in the appendix – I used these names only for educational purposes. Typically you would name your package and your class and your method using good programming style (so, meaningful names). For this lab:
    - Name your method according to its function.
    - Try to name the class with a broader name that implies that might add other similar methods to this class later. Remember that your class names (by convention) start with a capital letter (so that we can easily tell a class, for example, from an object).
    - I recommend that you name your package staticMethodPkg (a name that is associated with this lab).
- Your method shall take numbers as input parameters – at least one double and at least one integer. Your method shall return a double or an integer. You may select any calculation you like, but try to make it related to the overall topic of your website. The calculation can be one that you have done for a previous lab or you can come up with a new calculation. However, the calculation CANNOT be the same or overly similar to examples given in lecture.

The JSP Page

- Your JSP page (methods.jsp) shall ask the user for two sets of numbers (so that would be at least 4 text boxes). It might be tough to make your form look realistic, but please try. For example, you could label your textboxes like this: “Today’s Dinner Bill”, “Today’s Tip Percentage”, “Yesterday’s Dinner Bill”, “Yesterday’s Tip Percentage”. As in previous labs:
  - Your page shall tell the user what to enter and explain what calculations will be performed (say in words what the calculations will be, so we can grade your homework).
  - The word “null” shall not appear anywhere on the page (neither on first rendering nor postback).
  - All user entered values shall persist (textbox values shall not disappear on submit).
  - The form shall post to itself (action attribute specifies the name of the jsp file you are editing).
  - The <form> tag shall have “get” specified for its method attribute (this is so that you can see the user entered values in the URL upon postback).
Your JSP page shall perform **field level validation** of all numeric input to be sure that the user did enter the type of input that was requested. As in previous labs, use `Double.parseDouble()` and `Integer.parseInt()` inside of try/catch blocks. If there is a problem with any numeric input, a field level message shall be displayed near where the problem is (and in red font so the user knows that they have made a mistake).

- To implement the red font, you can create a CSS style rule like this:
  ```
  .error { color: red; }
  ```
  then add a span tag around the field level message like this:
  ```
  <span class="error"> <%=strFldErrorMsg %> </span>
  ```
- Your page’s inputs shall be nicely aligned (use an HTML table, if necessary). The tags to make a table are: `<table><tr><td>hello</td></tr></table>` (This would create a single row, single column table with the word “hello” displaying inside that single cell. Remember you can only have input inside the `<td>`…`</td>` tags, no where else in `<table>…</table>```
- Your page’s input box shall not “move” when the user clicks submit (this might happen, for example, if your layout uses centering and an error message is displayed).

If the user’s **first set** of input passes validation (no exception is thrown) your method shall be invoked, passing in the numeric values that you just created from the first set of user input – and these results shall be displayed on the form. If the user’s **second set** of input passes validation (no exception is thrown) your method shall be invoked, passing in the numeric values that you just created from the second set of user input – and these results shall be displayed on the form. So, it’s possible for the first set to be OK and show good results but not the second set and vice versa (or both sets good, or both sets bad).

- The numeric result that is displayed on the form shall use formatting, as in the previous lab.

**Other Requirements**

- Add a blog to your labs page ("labs.jsp" has the same layout as your home page). Your blog shall describe the work you did this week and link to your methods.jsp. It shall list what concepts you thought were easy or hard. Note: methods.jsp will only be reachable from the labs.jsp page.
- Also remember to follow the requirements listed under **“Requirements for All Labs and Project”** in the 1056 labs page on my web site. It will have requirements like “HTML code reuse via JSP include statements”, “no (HTML/CSS) syntax errors when we view source from Firefox”, and “Good naming”.
- **Use the new instructions (“publishing web apps with classes”)** for publishing your web app. It can be tricky if you are not paying a lot of attention when you do it, so follow the instructions carefully and exactly.
- In addition to publishing your web application (and testing all links), attach a zip file (of your web app folder) into blackboard by the due date (check the lab schedule towards the top of blackboard – this is the single place where updated due dates are posted).
APPENDIX - How to Add a Static Method to a Class of a Package of a NetBeans Web App

In the Netbeans project pane (upper left of screen),

- right click on “source packages” and select “New – Java Package”. Call it what you like, but not “myPkg”. Give it a representative name.
- Then right click on the new package and select “New – Java Class” (again, give it a representative name, classes are supposed to start with a capital letter according to java naming convention).

- Then (inside your new class file), enter the code for your method. In the example below, my method’s name is “myMax”. This method takes 2 integers as input parameters and returns an integer.
• In your JSP page, you must use an import statement (as shown below and also in line 2 in the code example) or else you will not be able to reference any methods of your class.

```jsp
<%@page language="java" import="myPkg.MyClass" %>
```

• Now, in your JSP code, you can call the method by specifying the class name "dot" method name, as long as you provide the correct number and type of input parameters (see line 15 below).

```jsp
<%@page contentEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <title>Using a Static Method from a Class of a Package !</title>
</head>
<body>
  <h1>
    <%
      int first = 3;
      int second = 5;
      out.print ("Between " + first + " and " + second + ", the larger value is "+MyClass.myMax(first, second));
    %>
  </h1>
</body>
</html>
```

• When you right click the JSP page, you’ll see how the method call was evaluated.