Waukesha County Technical Institute
152-198 Distributed Java

Class 4 Plan and Assignments

Discussion Activities:

- **Due Today:**
  1. Your "Class2Lab" lab activities #2 and #3 are due today on GitHub. Remember, activity #4 is not due today, but extra credit will be awarded for those who made a decent attempt.
  2. Your research topic for JEE features is due today. You will make a brief (no more than 5 minutes) presentation to the class. Use any support tools you want, including PowerPoint, etc., or nothing at all.

- **Q&A & Review of your homework assignments.**

- **Using the CSS Editor in Netbeans**
  - ✓ There is a wizard for creating a new CSS file from the web category in Netbeans.
    - This will create a blank css file for you to edit, and
    - You will have some intellisence assistance in the editor, and
    - From the Window Menu in Netbeans you can select Web > CSS Styles to see a pane that displays your current CSS rules. And add rule icon helps you create new rules. You'll see this icon in the lower, properties panel.
    - You can drag and drop a CSS file from the projects tab into your web page to automatically create the link tag.

- **Introduction to Maven**
  - ✓ Maven is a Java build, deploy, test, report and dependency management system that can automate many of these tasks.
  - ✓ The two main advantages of using Maven to develop Java applications is that it manages your dependent libraries for you and it is a universal format that works in any development tool or IDE that suppots Maven. This means, for example, that you can create and work on a Maven project in Netbeans one day, and then load it into a different IDE – say Eclipse or IntelliJ – and continue working on it there the next day. The Maven project format is the same for all development tools, so it's portable.
  - ✓ To create a Maven project go to the New Project > Maven category and select Maven Web Application.
  - ✓ Read more about Maven here: **https://books.sonatype.com/mvnref-book/reference/public-book.html**
  - ✓ And here's a video of using Maven in Netbeans: **https://youtu.be/XMbTz_NQpUc**

- ✓ **Starting with uour "JstlExperiments" assignment (see below), from now on all projects in this class should be based on Maven.**

- **Today's Lab work: Continue "Class2Lab" Activity #4**
  - In this activity you are going to use the MVC design pattern to build an app that has a two-page view (a form and a response page), a single controller servlet, and a single model class. You will be using the same project as above.
  - Under "Source Packages" create a new package named "model". In that package create a regular Java class named "WelcomeService"
  - In that service class create a private property for the current date using the Calendar data type. Then create two methods:
  - The first method should be written to determine whether it is morning, afternoon or evening. Hint: you can use the before or after methods in the Calendar class to help you do this. Have it return a String containing "morning", "afternoon" or "evening". Come on, it's not that hard, you just have to think! You made it through Advanced Java, you can do this too!
  - Next, create a method that takes a String value for a person's name, and constructor a return String that combines that name plus the date result above into a welcome message. Something like this: "Good afternoon, Jim. Welcome!"
  - Next, create an web page named "welcome". Create an appropriate headline and form that takes a person's name and submits the form to a controller that has the alias "greeter". Use what you've learned so far to create the servlet for this alias. In your form don't forget to reference the alias like this:
    ```
    <form id="form1" name="form1" method="POST" action="greeter">
    ```
  - Next, create your controller with an appropriate name and the alias mentioned above. In the "processRequest" method remove the boilerplate code provided by Netbeans and replace it with your own. You will need to retrieve the form parameter and pass it to an instance of your model class. The model class will return a String value that you must forward to a result page that is a JSP. In that page you will retrieve that String from the request object and display it on that page. If you don't remember how to do these things, take a look at the sample BeerAdvice app provided by your instructor and use similar techniques. Learn by learning from someone else's code. Again, you are required to do critical thinking to be successful. Bear down and get it done!
  - When done, do a Git add/commit/push. We'll discuss all the lab activities soon, and look at samples of student work.
- Also, try to use some CSS to dress things up.

Textbook Chapters (and other resources) covered:

- Java EE v1.7 tutorial: http://docs.oracle.com/javaee/7/tutorial/doc/home.htm
- Java SE API (v1.8): http://docs.oracle.com/javase/8/docs/api/
- Java EE API (v1.7): http://docs.oracle.com/javaee/7/api/
- Online tutorials for client-side: http://w2schools.com

- Netbeans web development tutorials: https://netbeans.org/kb/trails/java-ee.html
- Netbeans Git User Guide: http://netbeans.org/kb/docs/ide/git.html
  (don't use SSH – we'll be using the modern HTTPS approach)

Homework for Next Class:

1. Complete any unfinished lab challenges and post changes to GitHub (activity #4 for example)
2. Research the use of JSTL (Java Standard Tag Libraries) and EL (Java Expression Language) to simplify your code

    - **JSTL** Reference #1 (no download needed):
      http://docs.oracle.com/javaee/5/tutorial/doc/bnake.html
      (Note: all the JSTL libraries are summarized in Table 7-1. Of these we will not use the Database tags because it violates MVC). For now, concentrate on the "core (c)", "format (fmt)" and "function (fn)" tags.

    - **JSTL** Reference #2:
      http://www.tutorialspoint.com/jsp/jsp_standard_tag_library.htm

    - **EL** Reference #1: http://docs.oracle.com/javaee/6/tutorial/doc/bnahq.html
    - **EL** Reference #2: http://www.journaldev.com/2064/jsp-expression-language-el-example-tutorial

2. Create a new sample web project (topic of your choosing) that demonstrates at least 3 different JSTL tags and uses EL in at least two different ways. You may not copy any code off the Internet or from books. All code must be of your own creation. Call this project "JstlExperiments". Remember, from now on you must create Maven web projects.