HW0: Getting Started!

Introduction

This homework will serve as a warm-up for the team project (hence the zero in *HW0*). In it, you will practice several skills:

- Editing, running, and testing a Java program
- Creating HTML documentation
- Editing code/mark-up collaboratively with your team

HW0 will introduce you to several tools that you will be using for the project: Eclipse, Subversion (SVN), JUnit, and Komodo Edit. These are professional-quality tools commonly used in practice.

A goal of HW0 is to have you setup the development environment that you will use for the remainder of the course. Because of the diversity of possible platforms that you may be working on (e.g., Windows, Mac, Linux), HW0 does not provide specific setup instructions for the various tools. It will be your responsibility to figure out those details. However, fear not! You can collaborate with the other members of your team to debug any tricky setup issues.

Part 1

The first part of the homework focuses on Java development with Eclipse and JUnit.

Step 1. Download and run Eclipse

Download and install Eclipse on your machine. Eclipse is the most widely used Integrated Development Environment (IDE) for Java. It includes facilities for editing, running, and debugging Java code. Here is the URL for downloading Eclipse:

http://www.eclipse.org/downloads/

You will want the version called "Eclipse IDE for Java Developers."

Once you've unpacked/installed the software (exactly what you do here will vary by platform), fire up Eclipse for the first time. You will be asked to create a workspace directory (aka a *folder*). This workspace directory is where all the files associated with your Java projects will be stored. Once Eclipse has launched, you will probably want to open the Workbench (see the arrow icon).

Step 2. Install the Eclipse Subversion plugin

In Eclipse, follow these steps to install the Subversive SVN Team Provider plugin:

1. Click **Help → Install New Software…**
2. In the **Work with** dropdown, select **--All Available Sites--**.
3. In the filter field (initially contains the text "type filter text"), enter "Subversive". Note: Eclipse may be *very slow* in updating the list. Be patient and wait for it!

4. You may need to expand the Collaboration item by clicking the triangle icon.
5. Check the box next to **Subversive SVN Team Provider**, and click the **Next** button.
6. Click through the installation screens by clicking **Next**/**Finish**/etc., and restart Eclipse when prompted.
7. Once Eclipse has started, return to the Workbench.
8. In the upper right, next to the **Java perspective** button, is an **Open Perspective** icon; click it, and select **Other…**
9. From the list, select **SVN Repository Exploring** and click **OK**. This should bring up an **SVN Repositories** view along the left side of Eclipse and cause an **Install Connectors** window to appear.
10. From the Install Connectors window, select the newest version of **SVN Kit** (*do not use* **JavaHL**), and click **Finish**.
11. Click through the installation windows by clicking **Next**/**Finish**/etc.
12. Restart Eclipse when prompted. When Eclipse restarts, you should be in the Workbench and the **SVN Repositories** view should be at the left side of the main Eclipse window.

Note: If you mess up on the **Install Connectors** bit and can't get that window to pop up again, try removing your workspace folder (I assume you have no valuable work in there at this point).

Step 3. Checkout the project skeleton

Next, you'll need to get a working copy of the Java source code to work on.

1. At the top of the **SVN Repositories** view is a **New Repository Location** button—click it. This should open a **New Repository Location** window with fields for you to fill in.
2. Fill in the fields to access your team's SVN repository and click **Finish**. (See the appendix below regarding details, such as URL, etc.) You may optionally check the **Save authentication**… box.
3. The repository should be listed in the **SVN Repositories** view. Inspect the contents of the repository by clicking the triangle.
4. Select the **trunk** folder within **Eclipse_Project**, right click on **trunk**, and select **Check Out**. This should download your team's Eclipse project into your local Eclipse workspace.
5. Switch to the Java perspective (button in upper right of Eclipse window), and confirm that the project is there (hint: look in the Package Explorer).

Step 4. Run the program and the unit test

Since running the application and running JUnit tests are basic operations in Eclipse, I will not give detailed instructions here. (Hint: Run → Run As → …)

1. Run the application (MyMain). You should see the output in the Console view at the bottom of the Eclipse window.
2. Run the JUnit test. You should see the results in the JUnit view at the left side of the Eclipse window.

Step 5. Make/commit changes to the code

Make the following changes to the project:

1. Add a method to `MyMain` that is your own version of `drFlemingSaysHi`. For example, if a student named *Kelly* were doing HW0, he/she would add a method `kellySaysHi`. Your method may return whatever String you want.
2. Edit the method `main` so it calls your method.
3. Add a unit test for your method. (See `MyMainTest` for an example.)

Once you've made your changes, commit them to the repository.

Be aware: All members of your team will be working on the same files. Therefore, your team will need to have a plan for merging changes done in parallel.

Part 2

The second part of the homework is a warm up for the HTML documentation that you'll need to produce in future homeworks.

Step 1. Download and install a Subversion client

Eclipse isn't a good environment for editing HTML. Thus, you will want to check out your team's HTML document skeleton using another Subversion client.

The following are recommended subversion clients:

- For Windows: TortoiseSVN (http://tortoisesvn.net/)
- For Mac: SCPlugin (http://scplugin.tigris.org/)
- For Linux: The subversion package (command-line tool)

You may need to read the tool's documentation to figure out how to use it.

Step 2. Download and install Komodo Edit

Komodo Edit is an Eclipse-like IDE that is good for editing HTML/CSS/Javascript/etc.:

http://www.activestate.com/komodo-edit

Step 3. Checkout the HTML document skeleton

Use your Subversion client to checkout the trunk of the design document, which contains an HTML skeleton. (See the appendix below for details about the repository URL, etc.) Since Subversion clients vary, it's up to you to figure out how to do the checkout. (Feel free to ask teammates for help—that's what they're there for.)

Step 4. Make/commit changes to HTML

Launch Komodo and open the Komodo project file **doc.komodoproject**. You should see the contents of the HTML document skeleton in Komodo's **Places** view (left side of Komodo window). You can edit the **index.html** file in Komodo Edit and view the file in your web browser of choice (e.g., Firefox, Google Chrome).

Looking at the document skeleton, you will see there is a title area and a section with Dr. Fleming's biography (including a picture and course history).

Your job is to make the following changes to the file **index.html**:

1. Change XXXX to the name of your team (only 1 team member should do this)
2. Add a your name to the author list such that the list is sorted alphabetically by last name. (Also, remove the dummy names from the author list.)
3. Add a section for yourself that includes, like Dr. Fleming's section, a heading, a biographical sketch, a picture of yourself, and a table listing the courses you've taken. Here are some additional constraints:
    a. Use the HTML tags in the same way that Dr. Fleming's section uses them.
    b. Include at least one link (using an **a** tag) in your biography.
    c. Sort sections by last name.
    d. Remove Dr. Fleming's section.

Once you've made your changes, commit them to the repository. As in Part 1, your team will need to have a plan for merging changes done in parallel.

Submitting

To submit your team's work, one team member must tag your submission by copying your trunk directories into the tags directories as follows:

- Within **Eclipse_Project**, make a copy of **trunk**, place the copy in **tags**, and name the copy **HW0**.
- Do the same for **Design_Document**.

Of course, the team member who does the tagging should wait until everyone has committed, or until time runs out—whichever comes first.

To give you an idea of how tagging is done, here is an example of how team Puma would tag their Eclipse project using the command line:

```
svn copy \
https://wd378pc3.memphis.edu/group/COMP_Whatever/Team_Puma/Eclipse_Project/trunk \
https://wd378pc3.memphis.edu/group/COMP_Whatever/Team_Puma/Eclipse_Project/tags/HW0
```

The above three lines are actually all one command (the \'s allow the insertion of line breaks within a command). (You must replace *Whatever* as described in the appendix below.)

To evaluate productivity and provide quality feedback, I will checkout and inspect the tagged versions. You should get used to tagging like this because your team will submit this way for each homework.

Productivity

Regular productivity

You must complete all of the above tasks to earn the regular productivity points. (There will be no negotiating or renegotiating regular-productivity work this HW.)

Above-and-beyond productivity

Ideas for above-and-beyond productivity:

- Create a step-by-step tutorial (including screenshots) for one of the trickier tasks from above.
- Document a merging protocol for your team, including an example scenario.
- Create a step-by-step tutorial (including screenshots) that explains how to use the Eclipse Java debugger.
- Create a step-by-step tutorial (including screenshots) that explains how to use Eclipse's navigation features (e.g., Open Declaration, Open Call Hierarchy, Open Type Hierarchy).
- Create a step-by-step tutorial (including screenshots) that explains how to use Eclipse's refactoring features.

Note: You must negotiate the specifics of any above-and-beyond work with me. You may also propose an idea of your own.

Appendix: Repository Information

The URL for your team repository is as follows:

https://wd378pc3.memphis.edu/group/COMP_4882_2012_Spring/Team_XXXX/

where XXX is the name of your team (e.g., Puma, Falcon, etc.).

You will use your UofM username (e.g., sdflming) and the password that you provide me in class to access the repository.

The repository is organized into two directories:

- **Eclipse_Project** – Where your source code goes
- **Design_Document** – Where your HTML goes

Each of these directories is treated as a separate project, and thus, each is organized using the trunk/branches/tags schema. That is, within each of these two project directories are the following subdirectories:

- **trunk** – Where the current main version of your source code goes
- **branches** – Where experimental variants of your source code go (may not be used in this course)
- **tags** – Where tagged versions of your source go (see the Submitting section above)