Homework 5: Add Unit Tests

For this homework, you will practice writing unit tests to test your controller classes.

You will do this homework as a team; however, each member of your team will be responsible for the completion of particular tasks.

Step 1. Add some unit tests

Add black-box unit tests for each of the following servlet classes, dividing the work among your team by task:

- Task 1:
  - PlayersCreateFormServlet
  - PlayersCreateServlet
- Task 2:
  - PlayersDeleteFormServlet
  - PlayersDeleteServlet
- Task 3:
  - TeamsAddPlayerFormServlet
  - TeamsAddPlayerServlet
- Task 4:
  - TeamsUpdateFormServlet
  - TeamsUpdateServlet
- Task 5:
  - TeamsIndexServlet
  - TeamsDropPlayerServlet

Create three unit tests per servlet (for a total of six per person). Have one test of "typical" values and two additional tests that cover boundary cases (e.g., zero players and one player).

Because all of the above are servlets, you will need to use mocks (Mockito) for the servlet API classes and the DAOs.

Make sure that all your tests are contained in sub-folders of the **/src/test/** folder (not **/src/main/**).

Step 2. Submit (by tagging) your team's submission

The following instructions are essentially the same as last time; only the tag name has changed.

Attention! Before performing this step, you must make sure that all team members have committed their edits to the **trunk** in the repository.

Only one team member (the leader) performs the following.

First, you must fill out the **README.txt** file in your project's **trunk**. The file should list which team member performed each task (one team member per task).

To submit work in this course, you must tag it. Then, I will checkout the revision that you tagged and grade it. By tagging, you tell me that you are done, and this is the version you want me to grade.

The tag you must use for this homework is **hw5** (case sensitive, no spaces).

To tag the current revision of your trunk as **hw5**, do as follows:

1. Go to the **SVN Repository Exploring** perspective in Eclipse.
2. In the **SVN Repositories** view, find the **trunk** folder that you want to tag.
3. Right-click on the **trunk** folder, and click **Show History**. This should open the **History** view with a table listing the past commits to the **trunk**.
4. In the History table, right-click the newest revision (i.e., the one with the greatest revision number), and click **Tag from…** This should open a **Create Tag** dialog.
5. Enter **hw5** into the **Tag** field and optionally enter a log comment, then click **OK**. This should create the tag!

To verify that tagging was successful, open your team's repository URL in a web browser, and inspect what's in the repository.

New rules about documenting!

For all future assignments, you must provide the following documentation. Failure to follow these rules will result in deductions.

Class documentation. Each class that you create must have Javadoc comments that describe the class' purpose, and anything inobvious that a programmer would need to know in order to use it. Here is an example:

```
/**
 * A description of what the class MyClass does and how to use it.
 */
public class MyClass { ... }
```

Method documentation. Each method must have Javadoc comments that describe the method's purpose as well as annotations describing its parameters and return value (if not void). Here is an example:

```
/**
 * A description of what the method does and how to use it.
 *
 * @param myParam1 What the myParam1 String represents.
 * @param myParam2 What the myParam2 float represents.
 * @return What the returned int represents.
 */
public int myMethod(String myParam1, float myParam2) { ... }
```

Within-method comments. If your code within a method might be at all confusing to a reader (e.g., someone grading it), then you must include comments that explaining the code. You generally need not comment on common programming patterns that we use. For example, there is not need to explain how forwarding to a JSP works. However, application-specific logic should be strongly considered for commenting. The standard that the grader will follow is that if he/she can't easily understand what's going on in a method and there are no comments to help him/her understand, then there will be some deduction.

SVN log. You must include a log entry with <u>every</u> commit. The entry should describe what the nature of the change was. Was it a bug fix? Did you add a feature? Did you complete a task? Is it an intermediate commit for a task that is in progress?

For the current assignment, you need only add the above comments for the test classes/methods you create.