Assignment 2

INF 5750 – Open Source Software development – autumn 2007

Technologies involved

Use of Spring, Hibernate, JUnit, Subversion, and Maven.

Overview

In this assignment you will be implementing the functionality for a simple student system. The system consists of three layers: The persistence layer, the service layer, and presentation layer. Your task is to implement the functionality of the two first-mentioned layers. The functionality of the two layers is defined in interfaces which are provided for you. You need to implement the interfaces and create unit tests which prove that your implementations are correct according to the interfaces. The graphical user interface (presentation layer) is provided, and can be configured to use your implementations to get a fully working system.

Case description

The case is a simple student system for keeping track of students, courses and degrees. Students can attend courses and get degrees, courses have attendants, and degrees have required courses. In order for a student to get a degree the student must attend all courses which are required by the degree. To keep the model simple there is no time aspect in the system. The system does not claim to be a realistic student system.

Provided source code

Note: You're not allowed to change any of the provided code except the root pom.xml and the pom.xml of the GUI project.

assignment2-api

The API project contains the model and the interfaces you will be implementing. The model consists of three classes: Student, Course, and Degree. The properties of these classes are kept to a minimum to make life simpler. The Student has a Set of Courses and a Set of Degrees. The Course has a Set of Students (attendants), so the Student-Course relation is bidirectional. The Degree has a Set of required Courses. (Tip: Make a drawing of all the relations). For each of these classes you will find a corresponding DAO interface (Data Access Object) with simple methods to persist ("store") and to retrieve persisted objects. The main functionality of the system is defined by the StudentSystem service interface. Your implementation of this interface will use the DAOs to persist objects.

assignment2-support-hibernate

The Hibernate support project contains one class with methods intended for facilitating use of Hibernate. Methods for starting and ending a transaction as well as retrieving the current Hibernate session are provided. HSQLDB is the recommended DBMS and currently added as a dependency, but any preferred database can be used.

assignment2-gui

The GUI project contains a fully working GUI based on Swing. The GUI will use your implementation of the API when you run it.

Requirements

- All your code must be in a separate Maven 2 project (assignment2-<username>) alongside the provided modules. You must put all modules in a subdirectory called assignment2 in the same Subversion repository as in assignment 1.
- All DAO interfaces must be implemented according to the description in the interfaces and in the assignment using Hibernate for persistence. Usage of the Hibernate support project is mandatory.
- The service interface (StudentSystem) must be implemented, and must use the DAO interfaces for storing and retrieving the model objects.
- *All methods* must be unit tested according to the defined behaviour of the implementations. The tests must run successfully with Maven and use Spring to instantiate the components which are going to be tested.
- The components of the system must be wired together using the Spring container and the dependency injection principle.
- The code must follow the DHIS 2.0 code style.

Hints and detailed description

- First create a new directory in your inf5750 repository from assignment 1 called assignment2.
- Download the provided source code and unzip the three modules within it into your working copy.
- You must create a new project alongside the other three modules from the zip file called assignment2-<username> in which you will write your implementations (from now on referred to as “your project”). Inside your project directory you will need a pom.xml file (use one of the pom.xml files in assignment2-api or assignment2-gui as a template) and the usual source directories (src/main/java etc).
- In order to implement the API in your project, you need a dependency to assignment2-api.
- In order to implement and run unit tests you need a dependency to JUnit (junit) and a dependency to Spring (org.springframework).
- Opening up the project in Eclipse:
    - Run *mvn install* from the root of the project (where pom.xml, assignment2-api, assignment2-gui, etc. is located)

    - Run *mvn eclipse:eclipse* from this directory
    - Import (do not choose "new project") this location into Eclipse and select all the projects. This will create separate projects in Eclipse.
- In order to get the GUI working later on, you have to add your project as a dependency to the pom.xml in the assignment2-gui project.
- Use the repository actively! It will prevent lost source code.
- Your Hibernate implementations of the DAO interfaces should be named *HibernateCourseDAO* (and so on) and be placed in a package called no.uio.inf5750.assignment2.dao.hibernate. Your implementation of the StudentSystem interface should be named *DefaultStudentSystem* and be placed in package called no.uio.inf5750.assignment2.service.impl.
- Your DAO test classes should be placed in a package called no.uio.inf5750.assignment2.dao. Your StudentSystem test class should be placed in a package called no.uio.inf5750.assignment2.service.
- The *beans.xml* file must go into a directory called src/main/resources/META-INF/assignment2/ for the main class in the GUI to find it.
- To run the GUI, locate the main class in the GUI project in Eclipse and run it as a Java Application in Eclipse (right click the file in the package explorer). By default, Eclipse uses the dependencies from the local Maven 2 repository. Remember to install the artefacts/projects you change into the repository before you run the GUI. You can set up Eclipse to depend directly on the other projects in Eclipse, so that you don't have to manually install them in the repository every time you wish the changes to take effect in the GUI.
- Remember that each time you alter a pom.xml, you must rebuild the Eclipse files and refresh the corresponding project for the changes to take effect in Eclipse.
- When you're done, commit all the source code to your Subversion repository and send your group teacher an e-mail with subject "Assignment 2 - <username>".