SIMON FRASER UNIVERSITY
Department of Computing Science
CMPT 276: Assignments

Assignment 2: Design Patterns (Composite + Observer)

1 Considerations

This is an individual assignment. Each person has to complete the assignment separately.

The TA will mark the uploaded files on Canvas on the specified deadline. See the Schedule page on Canvas for deadlines.

You should create a Java project using Maven. Tests should be written in JUnit format and be executable through the command line using Maven.

2 The File System

For this assignment, you want to build a simplified file system. Your file system contains nodes, which can be files or directories. (You will only create a simplistic representation of a file system, and not an actual file system.)

2.1 Composite Pattern

1. You want to add a `find()` method to the nodes in your file system. This method takes a string `s` and finds all files whose name contains `s` as a substring.
2. Design your file system using the composite design pattern. Identify the classes you need to implement and determine which participant in the composite pattern each class is. Draw a class diagram that represents the important classes in your design, their relations, and important fields/methods. Include the design in your report.

   Create the outline of your classes and methods in your code.
3. Write tests that specify the expected behaviour of your system.
4. Complete the implementation of your classes and their fields and methods, including the `find()` method. Rerun the tests as you complete your implementations. All tests should pass when your implementation is complete.

2.2 Observer Pattern

Next, you want to add *FileObserver*s to your file system. All FileObservers should be automatically notified when the contents of the observed file changes.
(For instance, if a file is open in different editors, all the editors should be notified if the contents of the file change.)

1. Add a method `write(String)` to class File to model operations that change a file's contents. Associate FileObservers with Files and notify them after each write to a file. FileObservers just print a warning message that the file has changed.
2. Extend your class diagram from the first part and add necessary classes and methods for implementing this feature to your design and your code. Update the report with the additions, and identify the new classes and how they participate in the observer pattern.
3. Extend your test suite with unit tests for verifying the new feature.
4. Complete your implementation of the new feature and rerun the tests.

3 Deliverables

You should upload a zip file to Assignment 2 on Canvas, named “<Student ID>.zip”, which contains:

- Your report, named “report.pdf,” containing the UML diagram that shows the design with both patterns. For each class, specify which design pattern it is used in, and as which participant in the pattern (a class can be used by one or both patterns).
- Your code, in src/ directory.
- Your POM file.