# Java and Object Oriented Software Engineering
# Semester 2 Coursework Laboratory 4
# JAgora - An Electronic Stock Exchange

Friday 4th March, 2016

## 1 Coursework Overview and Working Arrangements

During the course of semester 2 you will work on the development of a single software project. Each laboratory will concentrate on a different aspect of the software development process:

| Laboratory | Week | Beginning | Topic | Marks |
|---|---|---|---|---|
| 1 | 19 | 25th Jan | Warm up | 0 |
| 2 | 21 | 8th Feb | Problem domain analysis | 25 |
| 3 | 23 | 22nd Feb | Unit and acceptance test harness | 25 |
| 4 | 25 | 7th Mar | Design and implementation | 25 |
| 5 | 27 | 21st Mar | Applying design patterns | 25 |

During semester 2, we will be using *pair programming* to improve the quality of software produced during the laboratories. You should prepare for your scheduled laboratory as normal, by reading the laboratory sheet problem description and undertaking background research. At the start of the scheduled laboratory, your tutor will pair you with a different student to work with in each of the five JOOSE laboratories. You must work with your assigned partner throughout the scheduled laboratory. Your tutor will monitor how you work as a development team as part of your assessment. Following the laboratory each partner should take a copy of the solution developed thus far and make further independent improvements as desired before submission.
**Note:** As pair programming is part of the assessment for each laboratory, non-attendance without good cause will result in a loss of all 5 marks available in this category. You must note down the name and matriculation number of your pair programming partner on your submission.

## 2 Task

You have been provided with a *three* related software projects in the laboratory handout.

- `jagora-api` contains a set of interfaces that define the application programming interface of the JAgora application. Each of the interfaces is accompanied by JavaDoc documentation describing the semantics of the operational signatures. *You must not make any modifications to this project.*
- `jagora-test` contains a suite of JUnit test cases for the `jagora-impl` project. You may add your own test cases from the previous laboratory to this project if you wish. *However, you must not make any modifications to the existing test cases*.
- `jagora-impl` contains a set of (unimplemented) classes that realise the interfaces in the `jagora-api` project. *You will work exclusively in this project*.

Working with your lab partner, you should complete an implementation of JAgora by completing the unimplemented classes.

Follow these steps to get started:

1. Start by creating *three* new projects in Eclipse for each of the three projects supplied in the laboratory archive. For example, for the project called `jagora-api` you should create a project called `jagora-api` with a root directory of `lab4-distrib/jagora-api`. You may find it convenient to change the Eclipse workspace to the `lab4-distrib` directory first.

2. The source code for the `jagora-api` and `jagora-impl` projects is contained in the directory `src/main/java`. Make sure that this directory is used as a source folder in both projects so that the code can be compiled in Eclipse. Eclipse may do this for you automatically, but if not you should: *Right click on the source directory* (e.g. `src/main/java`) the choose *Build Path* and *Use as Source Folder*.

3. The source code for tests in the `jagora-test` project is contained in a directory called `src/test/java`, so that we can separate test and application code. Make sure that this directory is also used as a source folder following a similar step to above.

4. You may also need to add the JUnit libraries in the `lib` directory to the `jagora-test` project build path. To do this, select the three libraries then *Right click*, *Build Path* and *Add to Build Path*.

5. Finally, you need to create the dependencies between projects. These are as follows:

   - `jagora-impl` depends on `jagora-api`
   - `jagora-test` depends on `jagora-api` and `jagora-impl`

   To configure the dependencies for`jagora-impl` right click on the project, then *Build Path* and *Configure Build Path....* Select the *Projects* tab and click on *Add*. Make sure that the `jagora-api` project is checked, then click *Ok* and *Ok*.

   Repeat this process for the `jagora-test` project.

6. At this point your code should compile. Try running the test suite contained in the `jagora-test` project by right clicking on the project and choosing *Run As* then *JUnit Test*. Notice that almost all the tests will fail or result in errors, as the project is largely unimplemented.

7. Now proceed to implement your own first class. Start by choosing a test case for a relatively simple interface and implementation class from the `jagora-api` project, such as the combination of `StockTest` and `DefaultStockTest` for the interface `Stock` and implementation, e.g. `DefaultStock`.

8. Run the test, noting that it should fail. Now implement the `DefaultStock` class methods, i.e.:

```
public class DefaultStock implements Stock{

 private String name;

public DefaultStock(String name){
 this.name = name;
}

@Override
public String getName() {
 return name;
}
```

```
    @Override
    public String toString(){
     return name;
    }

   }
```

9. Run the test again, this time noting that it should pass - congratulation, you've got a green test!

Proceed through each of the test classes, implementing the incomplete implementation classes until all test methods pass. You may find the documentation for the domain and clearing algorithms for the previous laboratory helpful in guiding your implementation of the `jagora-impl` project.

# 3 Assessment

Submissions for this exercise are due by 10am two days after your lab during week 25 (week beginning 8th Mar). You should submit your solution on Moodle in the appropriate upload slot for the laboratory. You must *only* submit a zip compressed archive containing the `jagora-impl` project that you have implemented. *Do not submit either of the other two projects: these will not be considered in your assessment*. The root of this zip archive must contain a single directory called `jagora-impl`, with the source code and other files for the project below this node.

Call your submission file `<matriculation>.zip`, where `<matriculation>` should be replaced with your matriculation number, for example `2001234.zip`.

Your submission will be marked on a basis of 25 marks for the assignment. Credit will be awarded for:

| section | marks |
|---|---|
| **Pair programming** | 5 |
| **Test case passes** | 10 |
| **Implementation quality** | 10 |

Note that attendance at the laboratory is mandatory: any student who does not attend the laboratory and work in a pair without good cause will suffer a 5 mark penalty.

Test case coverage will be assessed on the percentage of test cases passed, rounded up to the nearest mark. Code quality will be assessed by your tutor based on the criteria for program quality discussed during lectures.

As per the Code of Assessment policy regarding late submissions, submissions will be accepted for up to 5 working days beyond this due date. Any late submissions will be marked as if submitted on time, yielding a band value between 0 and 22; for each working day the submission is late, the band value will be reduced by 2. Submissions received more than 5 working days after the due date will receive an H (band value of 0).