Exercises

Design Patterns: Factory Method

Java SE and Java EE patterns and best practices

João Miguel Pereira – http://jpereira.eu

0 Prerequisites, assumptions and notes

- Have Maven 2 installed in your computer
- Have Eclipse installed in your computer (Recommended: Indigo Version)
- I'm assuming you're running the exercises in Ubuntu
- It's recommended that you place all Design Patterna exercises under a common directory. For example:
  `${user.home}/javatraining/designpatterns`

  During the exercises I will refer this directory as
  `${designpatterns.exercises.folder}`
- In every exercise I will refer the directory where you are working as `${project.dir}`.

1 Quick Start Exercise

You will put your hands on a small program and apply the factory method to improve the design of the application. Currently, the design violates two design principles.

1.1 Checkout code and create eclipse project

In this step you will checkout the code to `${project.dir}`.

Complete the following tasks.↓

1. Go to the `${project.dir}` directory

```
cd ${project.dir}
```

2. Checkout the code from code.google.com

```
svn checkout
http://javatrainings.googlecode.com/svn/designpat
terns/trunk/factorymethod
```

3. Enter the created directory and run the tests to check that everything is ok.

```
mvn test
```

4. Enter the created folder and generate the eclipse project

```
mvn eclipse:eclipse
```

5. Import project into eclipse

✓ *you're done! You have now the project ready to refactor.*

1.2 Refactor code with Factory Method Pattern

First, the program it's not closed for modification but open for extension. Whenever you need to add a new type of report, you don't have any extension hook, so you will have to modify the `ReportGenerator` code.

The other principle this program is violating is the Dependency Inversion Principle. This principle states that higher modules in abstraction should not depend on concrete implementations of lower modules in abstraction. This is violated because `ReportGenerator` depends on concrete implementations of Report, which are lower modules in abstraction.

Complete the following tasks.↓

1. Change the program in order to use the Factory Method Pattern
2. Run and change the tests to see if changes are not affecting the expected behavior.

✓ *You're done.*