Exercises

Design Patterns: Adapter

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 Pattern exercises under a common directory. For example: `${user.home}/javatraining/designpatterns`

  During the exercises I will refer this directory as `${designpatterns.exercises.folder}`

1. 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 Adapter Design Pattern.

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/adapter
```

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 Implement the Class Adapter for ThirdPartyDoor

***Complete the following tasks.*↓**

1. Open the project **adapter** with eclipse
2. Under the package `eu.jpereira.trainings.designpatterns.structural.adapter.thirdparty`, create a new class for the Third Party Class door adapter. Name the file `ThirdPartyDoorAdaper.java`. The class should extends from `ThirdPartyDoor` and implement `Door` interface
3. Open the test `ThirdPartyDoorAdapterTest.class` and change the code to the following:

```
@Override
protected Door createDoorUnderTest() {
	return new ThirdPartyDoorAdaper();
}
@Override
protected String getDefaultDoorCode() {
	return ThirdPartyDoor.DEFAULT_CODE;
}
```

4. Run the test `ThirdPartyDoorAdapterTest`
5. Implement the `ThidrPartyDoorAdaper.java` until you have all tests green.

✓*You're done.*

1.3 Implement the Object Adapter for ThirdPartyDoor

***Complete the following tasks.*↓**

1. Open the project **<u>adapter</u>** with eclipse
2. Under the package `eu.jpereira.trainings.designpatterns.structural.adapter.thirdparty`, create a new class for the Third Party Object door adapter. Name the file `ThirdPartyDoorObjectAdapter.java`. The class should implement `Door` interface only.
3. Open the test `ThirdPartyDoorObjectAdapterTest.class` and change the code to the following:

```
@Override
protected Door createDoorUnderTest() {
	return new ThirdPartyDoorObjectAdapter ();
}
@Override
protected String getDefaultDoorCode() {
	return ThirdPartyDoor.DEFAULT_CODE;
}
```

4. Run the test `ThirdPartyDoorObjectAdapterTest`
5. Implement the `ThirdPartyDoorObjectAdapter.java` until you have all tests green.

✓*You're done.*