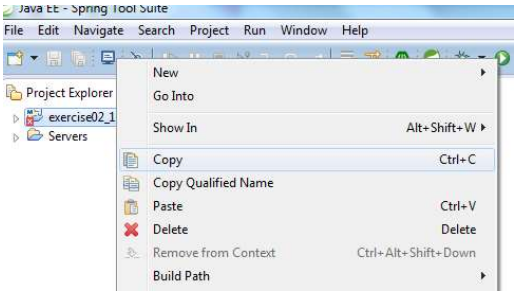
Module 03: Association Mapping

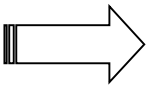
Exercise 03.1 – Adding an Association

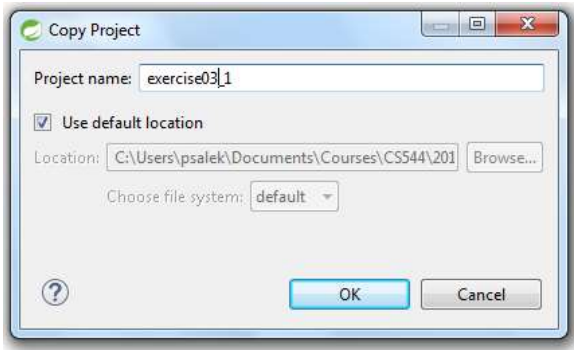
The Setup:

In this exercise we will be extending the code from **Exercise-02-1** to create a basic entity relationship between the **Car** class and a new **Owner** class.

Copy the exercise-02-1 project by right clicking on it, and selecting **Copy and then Paste**







Change the project name to exercise-03-1 and click the **Ok** button. Unfortunately, because the pom.xml of the copied project is still configured with the project name exercise-02-1 you will now have 2 projects called “exercise-02-1”

Open the pom.xml file for the second project, and change both the **artifactId** and the **name** to **exercise-03-1**

```
<groupId>cs544</groupId>
<artifactId>exercise-03-1</artifactId>
<version>0.0.1-SNAPSHOT</version>
<packaging>jar</packaging>

<name>exercise-03-1</name>
<url>http://maven.apache.org</url>
```

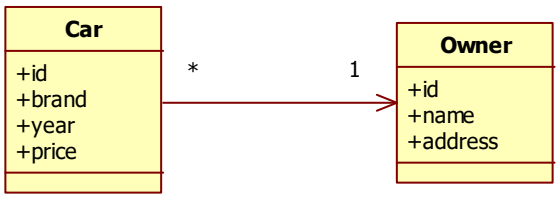
Once you save the changes to the pom.xml the name of the project in Eclipse will automatically update.

The Exercise:

Create a new class **Owner** with the following attributes:

```
public class Owner {
	private int id;
	private String name;
	private String address;
}
```

Start by generate the needed getters, setters, and constructors, and adding the persistence annotations. If you want, you can test your class by creating an Owner object and persisting it to the database.



Then add a uni-directional ManyToOne association from Car to Owner that cascades on persistence In other words, when a Car is persisted, Hibernate should automatically also persist its Owner.

Update the main method in **App.java** in so that it creates two cars and associates an owner with each one before persisting it. Then when retrieving the cars also print the details of each owner