Weekly Assignments Object Oriented Programming

A.H. Boode

February 18, 2014

Contents

Introduction

The assignments are based on the book Data Structures & Problem Solving Using Java by Mark Allen Weiss [1]. Each week consists of a presentation covering the topics of the week and hands-on practice. The assignments are a preparation for the final practical examination. The final examination consists of the implementation of UML diagrams, like a class diagram, a sequence diagram and a state transition diagram.

Week 1 | Basic Java

1.1 Create project

- Create a project Test in a java IDE (e.g. Eclipse, NedBeans, etc.)
- For NedBeans select the java application option, with creation of a main class.
- In the class Test, create in main a new instantiation of Test.

1.2 Class name

- Add the class DavinciRobot to your project sources.

1.3 Class attributes

- read section 1.3, the primitive types in Weiss [1].
- define in the attribute part of the class DavinciRobot the variables x, y, z of respectively type $boolean, int, byte$
- initialise the variables x, y, z to respectively $\{true, 12345678, 127\}$, using a constructor.

1.4 Class methods

- Create a method convert which has x, y, z as parameters and returns a $boolean$.
- The method convert returns $true$ if $x = true$ or $y < z$, otherwise $false$.
- call method convert from object Test.

1.5 Primitive types

Use your imagination, while creating methods that use all primitive types as given on page 43 of [1].

- byte
- short
- int
- long
- float
- double
- char
- boolean

1.6 Overloading

- Create a convert method that uses other types of parameters and demonstrate, using System.out.println(" that your implementation works.

1.7 Private, public and protected

- Change your implementation is such a manner that the attributes can be read or written only by get and set methods and by objects from the same package or inheritance tree.
- Change your implementation is such a manner that the attributes can be read or written only by get and set methods.
- Create a method updateBalance(int value) that adds value (value can be negative) to the private attribute balance only if the result will be greater or equal to zero.

1.8 Scope of a variable

- assign the parameter values of x, y, z of convert to the attributes x, y, z of the object of DavinciRobot.

1.9 Exercises Weiss

- Do the exercises 1.1 through 1.10
- Create a method in the DavinciRobot that implements 1.21

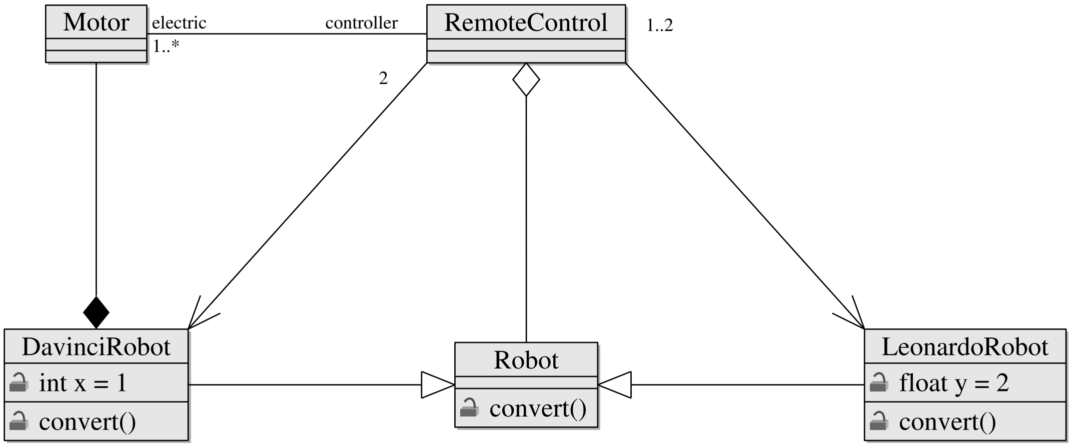

Figure 1: The Davinci Robot Class Diagram

Week 2 | Relations

In Figure 1 several types of relations are given.

- Implement the classes Robot, Motor, RemoteControl and DavinciRobot without their relations.

2.1 Association

- Implement the (default multiplicity) association between the classes RemoteControl and DavinciRobot

2.1.1 Navigation

- Add navigation to the association between the classes RemoteControl and DavinciRobot

2.1.2 Multiplicity

- Add multiplicity to relations where multiplicity is defined.

2.2 Inheritance

- Implement the inheritance association.

2.3 Aggregation

- Implement the aggregation association.

2.4 Composition

- Implement the composition association.

2.5 Polymorphism

- Create class LeonardoRobot according to Figure 1.
- Implement the associations from RemoteControl to DavinciRobot and from RemoteControl to LeonardoRobot. Store these objects in the array robots in RemoteControl.
- Modify convert in DavinciRobot, so that it displays the text ""This is the DavinciRobot".
- Modify convert in LeonardoRobot, so that it displays the text "But this is the LeonardoRobot"
- Implement a for loop that iterates over the array robots in RemoteControl with stored objects from the classes DavinciRobot and LeonardoRobot and call convert().

2.6 Exception handling

References

[1] Mark Allen Weiss. *Data Structures and Problem Solving Using Java.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 4th edition, 2010.