**Java**

Object Oriented Development

Module 7 – Dependencies

© FDM Group Ltd 2020. All Rights Reserved.

Any unauthorised reproduction or distribution in part  
or in whole will constitute an infringement of copyright.

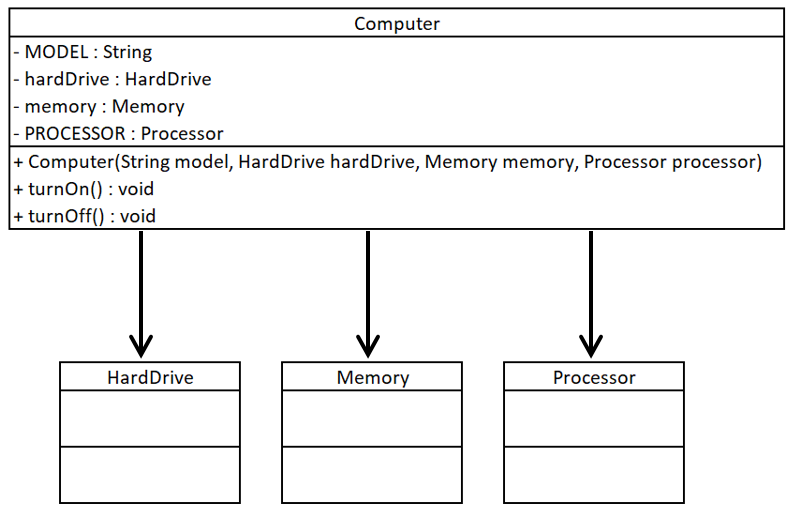
|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 26 / 10 / 20 | Nick Lawton | First draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Setting up

Create a package called com.fdmgroup.dependenciesExercises. Copy everything from the classesAndObjectsExercises package.

## Creating associations

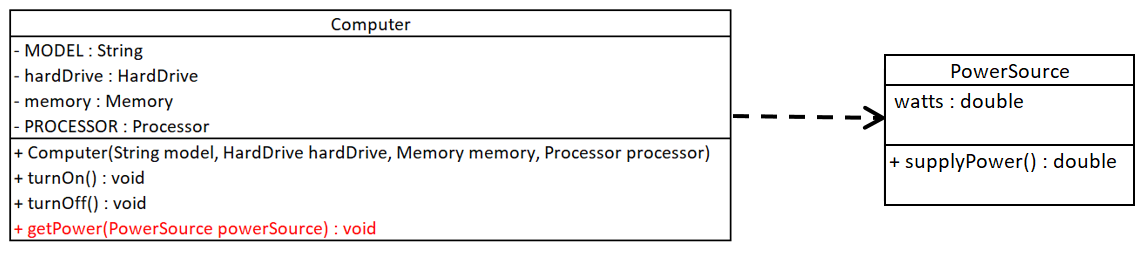
1. Create the Computer class shown in the UML below.
   1. Create getters and setters for all attribures.
   2. The two void methods should just print a simple message.



1. In the main method of your Runner class, create the following Computer objects:
   1. computer1
      1. model : Dell
      2. hardDrive1, processor1, memory1
   2. computer2
      1. model : Lenovo
      2. hardDrive2, processor2, memory2

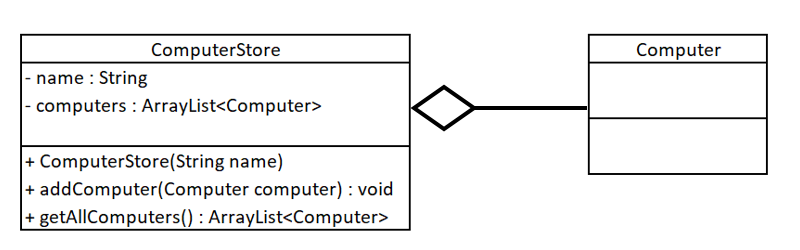
## Creating a simple dependency

1. Create the PowerSource class shown in the UML below. Create a setter method for the ‘watts’ variable. The supplyPower() method should return the value of the ‘watts’ variable.
2. Create the getPower() method in the Computer class. It should call the supplyPower() method of the PowerSource object that it receives as an argument. It should then print out a simple message such as watts+”watts received from power source”.
3. In the main method of your Runner class, create a PowerSource object. Give its ‘watts’ variable a value. Call computer1’s getPower() method passing in your PowerSource object. Verify that it prints a message containing the correct number of watts.



## Creating an aggregation

1. Create the ComputerStore class shown in the UML below:
   1. Create a getter and setter for the ‘name’ variable.
   2. Don’t create a getter and setter for the ‘computers’ ArrayList
   3. Instantiate the ‘computers’ ArrayList within the ComputerStore class.



1. Write the code in the ComputerStore’s methods:
   1. addComputer() should simply call the ‘computers’ ArrayList’s add() method passing in the Computer argument.
   2. getAllComputers() should return the ‘computers’ ArrayList.
2. In the main method of your Runner class. Create a ComputerStore object. Call addComputer() once for each of the computer objects you created in the previous section. Call getAllComputers() to verify that the computer objects have been added.