Java Exercices

fc - dec 2010

1. Java installation

- Install jdk1.6.x
- Add jdk1.6.x/bin at the beginning of the PATH
- Editor Textpad / Notepad++ (Windows) / Scite (Linux)
- Integrated compilation in the editor / javac in the terminal
- Create a directory for these exercices, e.g. **java/**

2. Exercices

a. Write the Training application

- See the Java training, page 10
- Type in an editor
- Save in correct directory / File.java

Help:

- ✔ compile from **java/**
- ✔ javac directory/File.java
- ✔ java package.ClassName

b. Write the two classes Tree and SpatializedTree

- Two classes -> two files
- Also in the training package

Help:

- ✔ See the Java training, page 24 and page 31

c. Add properties to these trees

- Add the following properties to the trees
- **id**: unique identifier - int
- **age**: age of the tree (years) - int
- **height**: height of the tree (m) - double
- **crownBaseHeight**: height of the crown base (m) - double
- **crownRadius**: max radius of the crown (m) - double

Help:

- ✔ In which class should you add these properties ?
- ✔ Add the accessors to get / set the values for these properties
- ✔ Organise your classes: keep variables order for accessors, mind indentation, white lines...

d. Add / update the constructors in the two tree classes

- All trees must be built with all their properties in the constructor

Help:
- ✔ Mind constructor chaining
- ✔ The properties must be managed at the level of their class
- ✔ Let the one who knows do (*Qui sait fait*) e.g. dbh is managed in Tree, x is managed in SpatializedTree

e. In the Training class, add a method to create a list of trees

- Takes 3 parameters: **numberOfTrees**, **xSize** and **ySize** (terrain size in m)
- **For each tree** to be created:
  - **id**: from 1 to n
  - **age**: ramdom between 1 and 25
  - **x**, **y**: random in the terrain, **z**: 0
  - **height**: age / 2
  - **dbh**: age
  - **crownBaseHeight**: height * 0.25
  - **crownRadius**: you choose
- Returns the tree list

Help:
- ✔ Use java.util.Random
- ✔ nextDouble (): a double in [0, 1[
- ✔ nextInt (n): an int in [0, n[
- ✔ Remember the for (...) {...} loop, see java training page 18

f. In the Training class, add a method to write a list of trees in a file

- Takes two parameters: **treeList** and **fileName**
- See Capsis Web Site > Documentation > 2.2 Developing modules > Tips and tricks > Write text to a file
- Write a line **for each tree**
- Variables separated by tabs
- id age x y z height dbh crownBaseHeight crownRadius

Help:
- ✔ Mind the exceptions handling, see java training page 20
- ✔ Create the line in a String
- ✔ Do not forget the newLine after each line

- ✔ Write a header line at the beginning: # id age x y...
- ✔ Do not forget to close the file
- ✔ In case of trouble, write to System.err (System.err.println ("Error, could not...");)

g. Create a list of trees and write them to a file

- Work in Training.main
- Use the two methods upper to create the list and write the file
- Work with local variables: int numberOfTrees = ... double xSize = ... double ySize = ...
- Write a file with a hardcoded name in the current directory

Help:
- ✔ This version works with hardcoded values for x, xSize, ySize, fileName
- ✔ Assign directly your values to local variables
- ✔ The next exercise is about passing the values to the program

h. Pass the values to the program on the command line

- e.g. java training.Training 25 50 30 trees.txt
- Read the parameters of the command line
- Order: **numberOfTrees**, **xSize**, **ySize**, **fileName**
- Use these parameters to create the file instead of the hardcoded values

Help:
- ✔ The parameters of the command line are passed to the main (String[] args) method
- ✔ This is an array of String
- ✔ Loop on the parameters to evaluate them
- ✔ Evaluate the 4 parameters and transform them into resp. **int**, **double**, **double**, **String**
- ✔ int i = new Integer (String).intValue ()
- ✔ double d = new Double (String).doubleValue ()
- ✔ In case of error, write a message on System.err and stop the program