DAM. UNIT 1. ACCESS TO FILES. PART 2. NON ASSESSABLE EXERCISES

DAM. Acceso a Datos (ADA) (a distancia en inglés)

Unit 1. ACCESS TO FILES

Part 2. Files. XML & XSL. Non assessable exercises

Abelardo Martínez

Based and modified from Sergio Badal (www.sergiobadal.com)

Year 2024-2025

Aspects to bear in mind

Important

> **If you look for the solutions surfing the Internet or asking the oracle of ChatGPT you will be fooling yourself**. Keep in mind that **ChatGPT is not infallible or all-powerful**.
>
> It is a great tool to speed up your work once you have mastered a subject, but using it as a shortcut when acquiring basic skills and knowledge seriously undermines your learning. If you use it to get solutions or advice on your own, check the proposed solutions carefully as well. Try to solve the activities using the resources we have seen and the extended documentation you will find in the "Virtual Classroom".

Tips for programming

We advice to follow the next coding standards:

- One instruction per line.
- Add comments to make your code clearer and more readable.
- Use the Hungarian notation to recognise the type of variables at first sight.
- If necessary, we strongly recommend using buffer-based solutions.
- Remember that there are several ways to implement a solution, so choose the one you like best.

1. Console mode. Reading XML files with SAX

Activity (non assessable)

Create a program in Java to manage PIZZAS in an Italian restaurant by printing and using a specific menu. After each option, the user should see the same menu until option zero is pressed. Feel free to share your doubts at the UNIT forum.

ATTENTION: Use the proper exceptions when accessing to files.

Menu options:

- **Press 0 to "Exit"**
- **Press 1 to "Ask for pizzas until user enters zero as ID"**
  - For every pizza we need the ID (Integer), the Name (String with spaces) and the Price (Double), added to the ArrayList of pizzas.
  - Check if the pizza ID already exists in the array list. If yes, you must display a message on the screen. You must ask for each value (in loop) until the user enters a valid ID.
  - Once zero is entered as ID, all pizzas will be saved in an XML file called "pizzas.xml" (using DOM), with a proper format, overwriting the whole file if exists.
  - **ATTENTION**: ID must be integer! For every pizza, you must ask for an integer (in loop) until the user enters an integer or zero to print again the menu.
  - **ATTENTION**: Price must be double! For every pizza, you must ask for a double (in loop) until the user enters a double (floating point number).
- **Press 2 to "List all the pizzas (using SAX)"**
  - Just read the XML file using SAX and print every pizza information.
- **Press 3 to "Remove all pizzas"**
  - Just delete the XML file.

Menu example:

```
*****
MENU
*****
================================================
  0. Exit
  1. Add pizzas
  2. List all pizzas
  3. Remove all pizzas
  4. Generate HTML report using XSL
================================================
    Select an option:
```

2. Console mode. Managing XML files with DOM

Activity (non assessable)

Create a program in Java to manage ANIMALS in a veterinary clinic by printing and using a specific menu. After each option, the user should see the same menu until option zero is pressed. Feel free to share your doubts at the UNIT forum.

ATTENTION: Use the proper exceptions when accessing to files.

Menu options:

- **Press 0 to "Exit"**
- **Press 1 to "Ask for animals (patients) until user enters zero as ID"**
  - For every animal we need the ID (String), Name (String with spaces), Species (String with spaces), Breed (String with spaces), CurrentAge (Integer) and Sterilised (String), saved in an ArrayList of animals. If an animal is sterilised, we type "yes" and "no" otherwise.
  - Check if the animal ID already exists in the array list. If yes, you must display a message on the screen. You must ask for each value (in loop) until the user enters a valid ID.
  - Once zero is entered as ID, all animals will be saved in an XML file with a proper format.
  - **ATTENTION**: Age must be integer!
- **Press 2 to "List all the animals (using DOM)"**
  - Just read the XML file using DOM and print every animal information.
- **Press 3 to "Sterilise a specific breed"**
  - After asking for the Breed (example: DOG), we walk through the DOM and change the Sterilised to "yes" for each animal whose breed matches the former introduced breed.
  - Then we (over)write the XML again.
- **Press 4 to "Sterilise non-adults"**
  - We walk through the DOM and change the field Sterilised to "yes" for every animal

(age < 2) with Sterilised “no”.

- Then we (over)write the XML again.

- **Press 5 to “Remove all animals”**
  - Just delete the XML file.

Menu example:

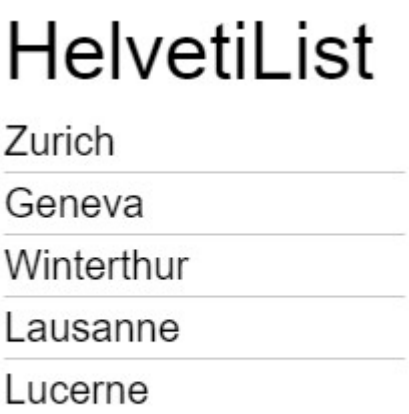
```
*****
MENU
*****
================================================
  0. Exit
  1. Add animals
  2. List all animals
  3. Sterilise a specific breed
  4. Sterilise non-adults
  5. Remove all animals
  6. Generate HTML report using XSL
================================================
    Select an option:
```

3. Console mode. Convert XML to HTML

Activity (non assessable)

Add a new option to the menus of exercises 2 (pizzas) and 3 (animals) to “Generate HTML report using XSL”. Feel free to share your doubts at the UNIT forum.

- Look for a predefined template on the Internet to show a list of items.
- Just create the necessary XSL to generate this HTML with this CSS.
- For example, you can use this one:
  - https://codepen.io/secondfret/pen/nxVKOD



Here you have more templates:

- https://designshack.net/articles/css/5-simple-and-practical-css-list-styles-you-can-copy-and-paste/
- https://speckyboy.com/html-lists-style/
- https://dev.to/hadrysmateusz/custom-list-styles-using-marker-before-4gi4