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

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

Unit 1. ACCESS TO FILES

Part 1. Intro, Java review and basic file access. Non assessable exercises

Abelardo Martínez

Based and modified from Sergio Badal (www.sergiobadal.com)
Year 2023-2024

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.
- Remember that there are several ways to implement a solution, so choose the one you like best. **We strongly recommend using buffer-based solutions**.

1. Console mode. Basic file access

Activity (non assessable)

Try to solve these 10 exercises. Feel free to share your doubts at the UNIT forum.

1. Write a Java program to get a list of all file/directory names from the given.
2. Write a Java program to get specific files by extensions from a specified folder.
3. Write a Java program to check if a file or directory specified by pathname exists or not.
4. Write a Java program to get last modified time of a file.
5. Write a Java program to get file size in bytes, Kb, Mb.
6. Write a Java program to read a file content line by line.
7. Write a Java program to store text file content line by line into an array.
8. Write a Java program to write and read a plain text file.
9. Write a Java program to append text to an existing file.
10. Write a Java program to find the longest word in a text file. Choose the way to implement: Scanner or BufferedReader version.

2. Console mode. Complete file access. Shopping cart

Activity (non assessable)

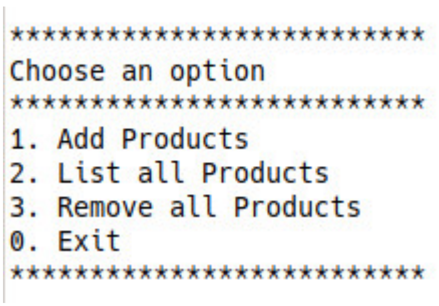
Create a program in Java to manage PRODUCTS in a shopping cart by printing and using a specific menu. After each option, the user should see the same menu until option zero is pressed. Feel free to share your doubts at the UNIT forum.

ATTENTION: Use the proper exceptions when accessing to files.

Menu options:

- **Press 1 to “Ask for products until user enters zero as Product name“**
  - For every product we need the Name (String), the Price (Double) and the Units (Integer), added to the ArrayList of products.
  - Once zero is entered as a Product name, all products will be saved in a “txt” file called “products.txt”, overwriting the whole file if exists.
  - One product is stored per line.
  - Afterwards, the menu will be printed again.
- **Press 2 to “List all the products stored”**
  - Just read the “txt” file and print every book.
- **Press 3 to “Remove all products”**
  - Just delete the “txt” file.
- **Press 0 to “Exit”**

Menu example:



```
**************************
Choose an option
**************************
1. Add Products
2. List all Products
3. Remove all Products
0. Exit
**************************
```

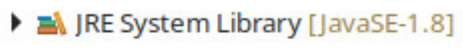
3. Graphical mode. Complete file access. Shopping cart

Activity (non assessable)

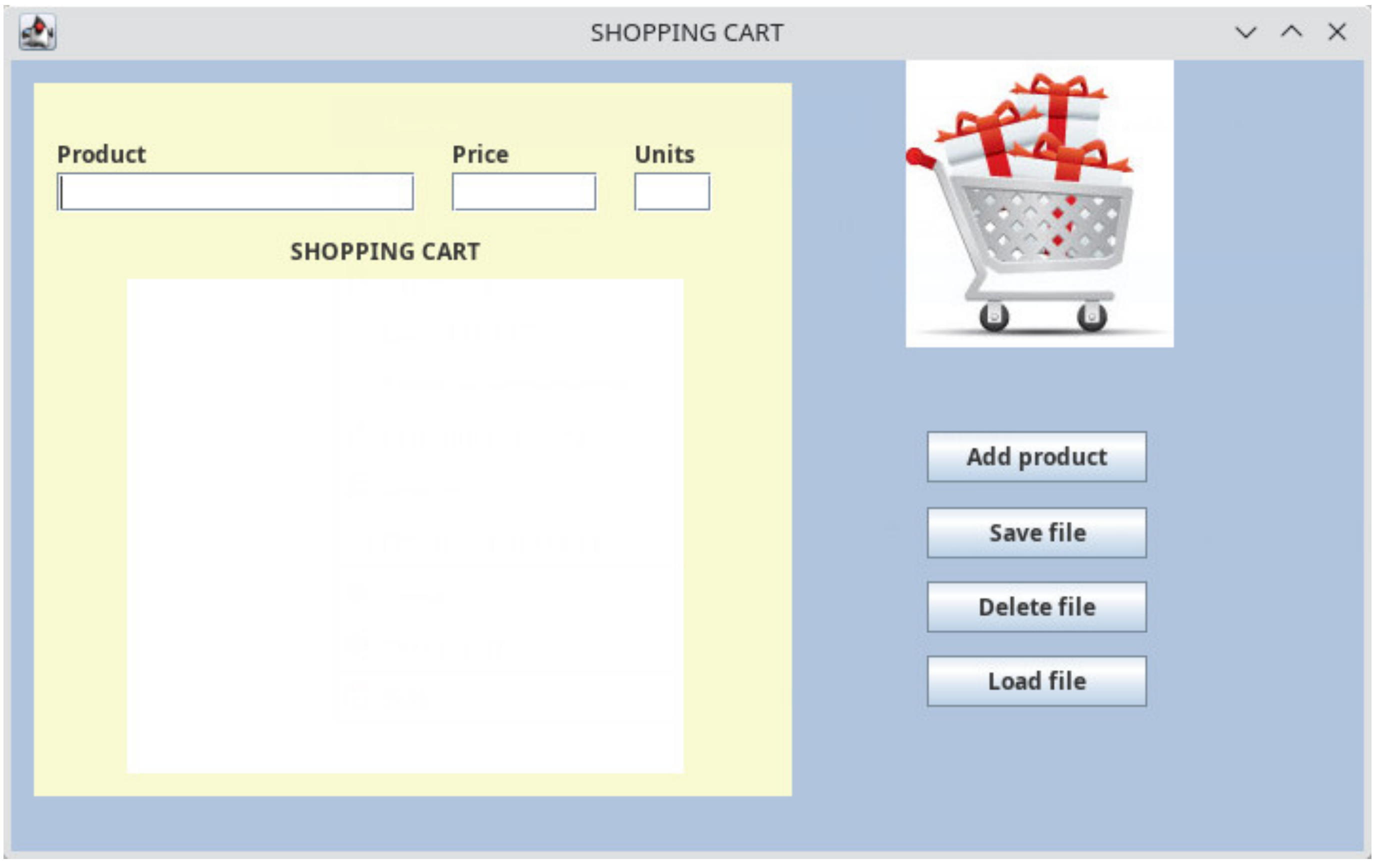
Convert the former program to a graphical environment using the Java GUI interface. Feel free to share your doubts at the UNIT forum.

ATTENTION: Use the proper exceptions when accessing to files.

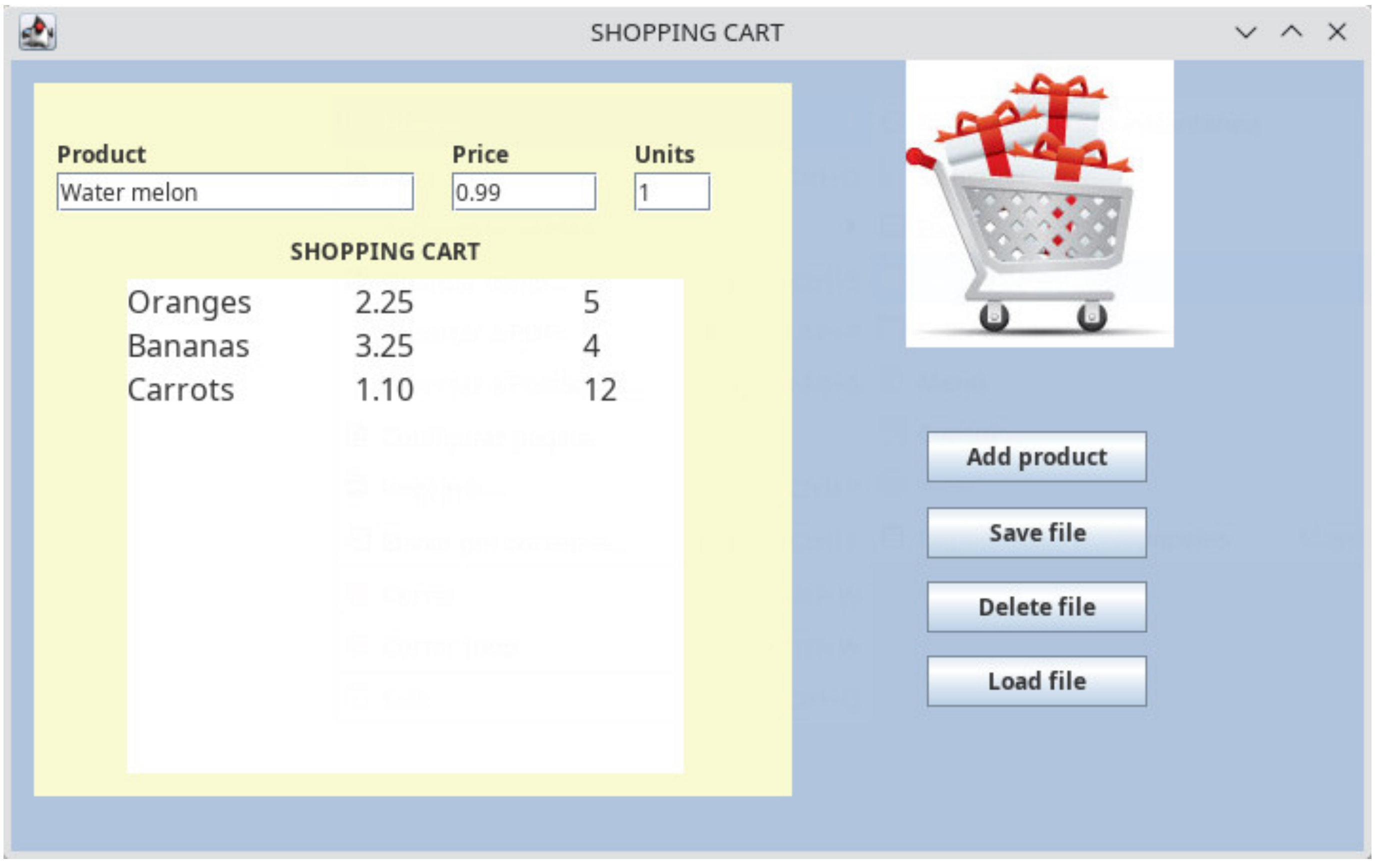
Remember to choose this **Java version** when creating a GUI project:


JRE System Library [JavaSE-1.8]

- **Create the graphical objects: labels, text fields, buttons, images. Customize the elements with your own design.**
  - Create project with Java JRE 1.8 machine.
  - To distribute the objects in a free way, use: JFrame → property Layout → Absolute layout.
  - For example:



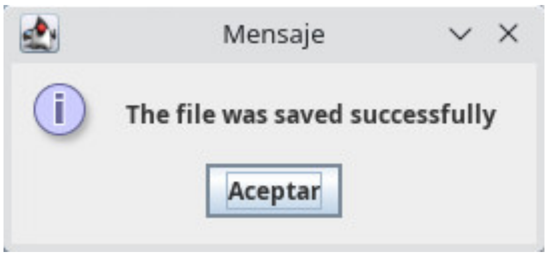
- **Button Add Product.**
  - Adds the product information (name -string-, price -double- and units -integer-) to the shopping cart (textArea).
  - Every field must be separated by a tab.
  - Every product in a different line.
  - For example:



Water melon 0.99 1

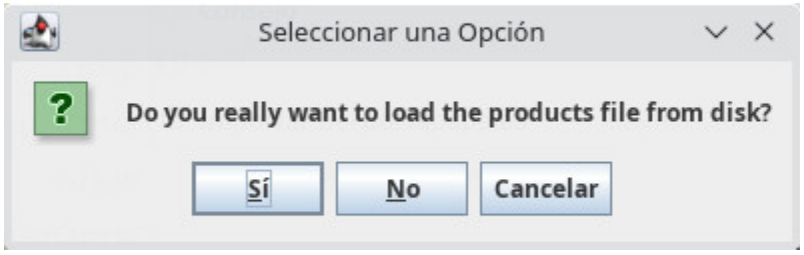
Oranges	2.25	5
Bananas	3.25	4
Carrots	1.10	12

- **Button Save file.**
  - All products will be saved in a text file called “products.txt”, overwriting the whole file if exists.
  - Every product in a different line.
  - Show a message dialogue with the result. For example:



The file was saved successfully

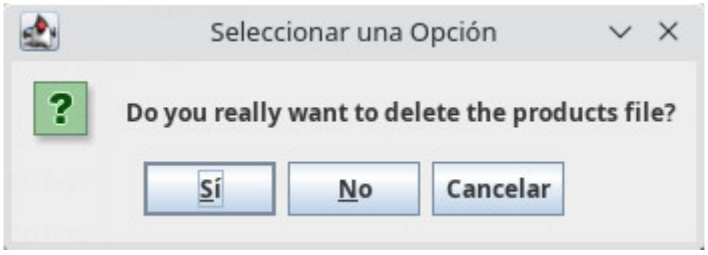
- **Button Load file.**
  - Ask the user if he/she is sure to do it. For example:


Seleccionar una Opción

Do you really want to load the products file from disk?

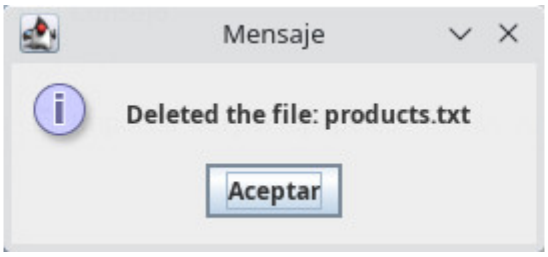
  - If yes, then clear the text area, read the text file line by line and fill in the text area with every product.

- **Button Delete file.**
  - Ask the user if he/she is sure to do it. For example:


Seleccionar una Opción

Do you really want to delete the products file?

  - If yes, delete the text file. For example:


Mensaje

Deleted the file: products.txt

- **Exit the programme/program.**
  - Quit by default; that is, when you click on the “x” you close the application.