Advanced Object Oriented Programming

DT4014

vt 2018

Project introduction

This project is a compulsory part of the course and encompasses the development of a software for image manipulation (transformations through filters). Students will work on the project in small groups and present project results during week 21.

Project objectives

The overall goal of the project is to design and implement an application for image manipulation, see Figure 1. The resulting image manipulation application allows end users to:

1. Load an image from a file and display it in a window;
2. Create a blank image and display it in a window;
3. Apply filters to the displayed image and then display the result.

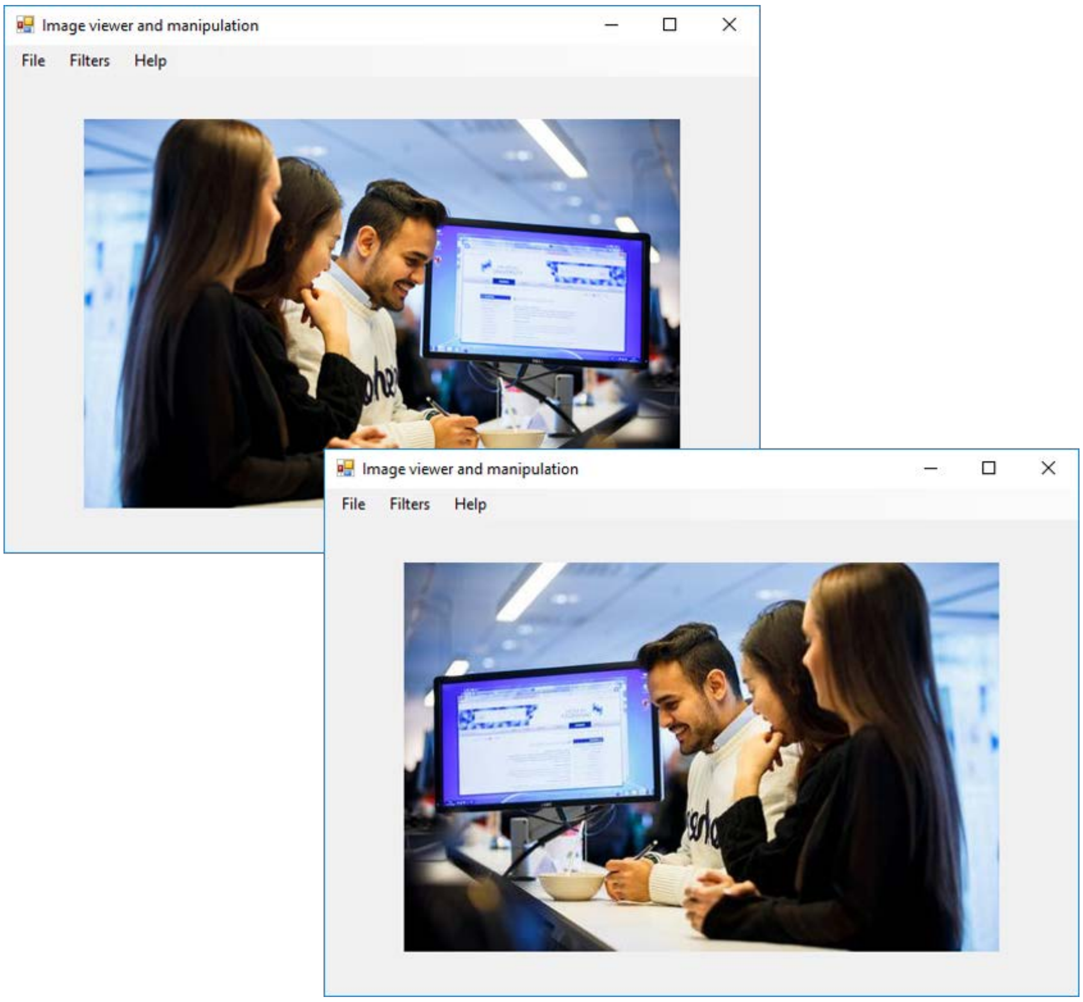

Figure 1. An example of a software for image manipulation illustrating its main components. The bottom image illustrates the result of a FlipX filter in the original image (top).

Requirement specifications

Main window

When executed, the image manipulation software should load a “Main window”, which must contain:

1. A menu in the top with the following items:
    - File: shall present the following options:
        - Open: to load an image from a file.
        - Save: to save into a file the current image.
        - Exit: to quit the application.
    - Filters: shall present the following options:
        - A list of possible filters. Please refer to section Filter for more information.
        - Undo/Restore: to restore the image to its previous state.
    - Help: shall present the following options:
        - Help: a simple help about how to use the system.
        - About: a simple information box about the authors and associated responsibilities.
2. An area in which the image will be displayed.

Filters

If you ever had the possibility to use an image manipulation software, such as GIMP or InkScape, you will know that there are different ways to manipulate, or filter, an image. In this project, students are expected to implement the following filters:

1. Swirl filter
2. Grayscale filter: converts the image to grayscale.
3. FlipX filter: flips the image horizontally, like a mirror effect.
4. Red filter: only the red component in each pixel.
5. Predefined image patterns: add a predefined image pattern:
    1. Vertical stripes: 10 vertical stripes that cover the image. The strips shall have the same width as the distance to the adjacent edge.
    2. Chess: A chess board that covers the image.
    3. Black circle: A black circle with a diameter equal to the width of the image in the center of the image.
6. Select and implement other two filters of your choice.
    - The goal with this requirement is NOT to select and implement a complex filter but to show how flexible and easy the system enables the addition of new filters.

Observations

Some filters, e.g. grayscale filter, should work only with source image, i.e. the image to be manipulated by the filter.

Some other filters, e.g. swirl filter, besides the image to be manipulated, will require additional parameters. For example, a slider component (JSlider) could be used by the user to interact with the system to indicate how much the filter should be applied.

The developed system should facilitate the addition of new filters.

Important

The application is expected to be designed in accordance with the advances OOP concepts presented in the course.

Project presentation

Project results will be presented during week 21 in room R1205. Project presentation shall have the following format:

- 5 minutes for a demonstration of the program,
- 5 minutes for an explanation of how the program is structured as explained in the project description, and
- 5 minutes for an explanation of the additional filter that where implemented and how new filters can be added to the system.

Important: Each group member should be able to present all these 3 items independently, i.e. without the help of his partner.

Project code submission

One student in the group will submit via Blackboard the source code for the software. The resulting code shall have the following characteristics:

- Source code should be properly indented and formatted.
- Source code should be documented: think of and consider using pre- and post-conditions and representation invariants. Explain the role of the parameters to methods and constructors.
- Source code should not include debugging statements.
- Create a single .zip file containing all the source code. Preferably provide a complete archived Eclipse project(s) with your work (use Export), if you are not using Eclipse just zip your project files, but please do not submit the .class files.

Grading

The minimum requirements to PASS (grade 3) are:

- The developed application has the basic functionalities according to the requirement specification.
- The developed application has the basic use of OO design patterns.

For better grade (4 or 5):

- Additional functionalities.
- Good design and use of patterns.
- Possibly a separate framework.