Homework. 3: Color image class in C++

Igor Bogoslavskyi, E-Mail igor.bogoslavskyi@uni-bonn.de

Handout : Wed, 25.04.2018
Handin: Wed, 09.05.2018

In this exercise you will change and extend your previous version of the `igg::Image` class. You can use the project sceleton provided here. Feel free to copy relevant parts from the previous homework where appropriate.

A What you will implement

Unpack the provided archive into the `homework_3` folder to get started (no need for `task_x` folders)

After performing all the tasks in this exercise you will have an implementation of a class that will be able to switch between reading/writing from and to `*.png` and `*.ppm` files by picking a different IO strategy.

This time you will need to implement reading and writing to and from files into the `*.ppm` format. We will use the human-readable ASCII PPM format. See `readme.md` in the provided archive for more information on this format.

We will also be using a library `libpng++` for reading and writing `*.png` files. Install `libpng++` by calling `sudo apt install libpng++-dev`. This library provides data structures and methods to read png images.

You can find example images in `data/` folder of the provided project.

B Color image initialization

1. **(2 points)** Modify the image class to hold objects of type `Image::Pixel` with `int` data members `red`, `green` and `blue` in this order. Create a library with the name `image` with the following functionality:

   - Add a const reference to `IoStrategy` as a member to your class. You can find the declaration of `IoStrategy` in `io_strategies/png_strategy.h`
   - Adapt your constructors to take a const reference of `IoStrategy` as an input and store it in your class:
     - `Image(const IoStrategy& io_strategy);`
     - `Image(int rows, int cols, const IoStrategy& io_strategy);`

     Make sure to use initializer list to be able to store the constant reference.
   - You must initialize this reference with a `DummyIoStrategy` instance for now when testing your code
   - Make sure your image stores pixel data in row-major order just like in previous homework: `https://en.wikipedia.org/wiki/Row-_and_column-major_order`
   - Size of an image can be accessed with getter functions `rows()` and `cols()` for variables `rows_` and `cols_`
   - Pixel values can be accessed and modified through the function `at(int row, int col)`

   This class will look very similar to the one you have implemented in the previous exercise.

2. **(2 points)** Resizing the color image:

   - `void DownScale(int scale);`
   - `void UpScale(int scale);`

   When upscaling some pixels will not have a value. Fill these pixels using the nearest neighbor algorithm.
   Hint: you can use the same implementation as your old simple image class. You should need very few modifications (if any) to it.

B.1 Implementing a strategy

The core idea of this exercise is to familirize yourself with the concept of strategy. You will implement and use a number of strategies to read and write to and from the hard drive. The idea is that you will be able to write and read `*.ppm` or `*.png` files by storing a `const` reference to an `IoStrategy` and using its `Read` and `Write` methods.

3. **(2 points)** Add I/O functionality to your class.
   - Make sure your class implements functions:
     - `bool ReadFromDisk(const std::string& file_name);`
     - `void WriteToDisk(const std::string& file_name);`
   - These functions must convert data from your class format to `ImageData` declared in `io_strategies/strategy.h` and call the appropriate functions in the stored strategy object
   - Make sure your library `image` links to the library that contains all available strategies
4. **(2 points)** Make sure you can use the provided class `PngIoStrategy` for reading/writing PNG files.
   - Make sure that class `PngIoStrategy` is part of `strategies` library
   - Use `find_package(PNG REQUIRED)` in one of your `CMakeLists.txt` to find png package and use correct variable to link against the `strategies` library
   - Test that you can read and write png files using this I/O strategy in your image class
   - *Optional:* read documentation on how to work with `png++` library: `https://www.nongnu.org/pngpp/doc/0.2.9/`
5. **(2 points)** Create a new strategy class `PpmIoStrategy` for reading/writing PPM files. You can use the provided `PngIoStrategy` class as a reference on how to implement your own strategy.
   - Create a new class `PpmIoStrategy` in file `io_strategies/ppm_strategy.h` file that can read ppm files
   - Use `fstream` to read and write images, see `readme.md` in `igg_image` folder for details on PPM format
   - Test that you can read and write PPM files using this I/O strategy in your image class

IMPORTANT: The interfaces provided above are stripped from `const` modifiers. It is part of this exercise to think where `const` is appropriate and add it where needed.
IMPORTANT: Use Google Tests to evaluate your work. The evaluation script will inject our custom tests into your framework and will run those tests against your code. Do not remove `tests` folder from the project.