ÉCOLE POLYTECH 'NICE SOPHIA
SI4
2017-2018

Julien Deantoni

Programmation orientée objet : CPP

Créer son modèle UML et l'implémenter

Objectif

Le but de cette séance est que vous trouviez un modèle adéquate à un problème simple et que vous l'implémentiez en utilisant un maximum les éléments de la STL:

http://www-f9.ijs.si/~matevz/docs/www.sgi.com/tech/stl/table_of_contents.html.

Manipulation d'images PPM

Dans cet exercice, nous allons faire du traitement d'image, même si le but n'est pas du tout de savoir faire ce genre de manipulation simple.
Une image couleur (nous ne considérerons ici que des images couleurs) est un ensemble de pixel. Une image a pour caractéristique ses dimensions : largeur × hauteur. Un pixel est repéré dans l'image par sa position (numéro de colonnes, numéro de ligne). Un pixel couleur est un triplet d'entiers (généralement un triplet de unsigned char) Rouge, Vert, Bleu. Par exemple (R=255, V=0, B=0) pour rouge et (R=255, V=0, B=255) pour jaune.
Le format d'image que nous utiliserons dans ce TP est le format PPM et plus spécifiquement le sous-format P3 pour les images couleurs. Un fichier PPM (P3) sera définit comme ceci :

```
P3
# un commentaire sur l'image
nb_colonne nb_ligne 255
pixels en ASCII : suite de (nb_colonne * nb_ligne * 3 ) lignes
dans le fichier
```

Ouvrez l'exemple miniDamier.ppm avec votre éditeur de texte préféré. Ensuite ouvrez l'image avec un logiciel de visualisation d'images (GIMP, gwenview ou autre), n'hésitez pas à zoomer.

Maintenant que vous avez compris de quoi est composée une image, réalisez le modèle de class UML associé et implémentez le de manière à pouvoir faire des manipulations simples d'images telles que:

- charger l'image dans les objets C++ que vous avez défini et sérialiser les objets dans un nouveau fichier image.
- Dès que vous êtes capables de charger une image et la sérialiser , vous allez réaliser les traitements d'images définis ci dessous, après le main d'exemple.

Example de main (ne le recopiez pas ! Construisez le au fur et à mesure de vos développements):

```
#include "Image.h"
#include <iostream>
#include <fstream>

int main(){
    Image simpsons("../../pict/Simpsons.ppm");
    simpsons.makeItGrey();
    std::ofstream simpsons_Grey("Simpsons_Grey.ppm");
    simpsons_Grey << simpsons;
    simpsons_Grey.close();

    simpsons.returnIt();
    std::ofstream simpsons_Grey_returned("Simpsons_Grey_returned.ppm");
    simpsons_Grey_returned << simpsons;
    simpsons_Grey_returned.close();


    Image simpsonsBis("../../pict/Simpsons.ppm");
    simpsonsBis.makeItBW(150);
    std::ofstream simpsons_BW("Simpsons_BW.ppm");
    simpsons_BW << simpsonsBis;
    simpsons_BW.close();

    Image simpsonsTer("../../pict/Simpsons.ppm");
    simpsonsTer.sortIt();
    std::ofstream simpsons_sorted("Simpsons_sorted.ppm");
    simpsons_sorted << simpsonsTer;
    simpsons_sorted.close();

 return 0;
}
```

Remplacement de couleurs

Toujours à partir de l'image du damier, vous aller écrire une fonction membre qui change les carrés noir en carré rouges. Pour cela sachez que les couleurs principales sont codées comme ceci

```
 R   G   B
255 255 255  -> blanc
 0   0   0   -> noir
128 128 128  -> gris moyen
255  0   0   -> rouge
255 255  0   -> jaune
255  0  255  -> rouge/violet
 0  255 255  -> bleu clair
 0   0  255  -> bleu
```

Modifiez maintenant votre fonction membre afin que l'utilisateur puisse choisir la couleur à changer ainsi que sa nouvelle valeure. Afin de tester votre programme, utilisez l'image EU_propre_little.ppm et changer la couleur de la france en rouge. (Dans l'image d'origine, la couleur de la france est 0 0 128)

Négatif

Votre but est maintenant de faire un négatif de votre image. Pour cela vous devrez appliquer un traitement à chacune des valeurs qui encodent les couleurs. À vous de trouver le traitement à appliquer.

Noir et blanc

Maintenant que la manipulation d'image n'a plus de secret pour vous, vous allez faire une fonction membre dont le but est de mettre en noir et blanc une image. Le seuil qui distingue le blanc du noir doit pouvoir être choisi par l'utilisateur.

Retournement de situation

Après avoir joué sur les couleurs, nous allons maintenant faire des symètrie de l'image. Codez une fonction membre qui fait la symètrie de l'image de sorte que le coin en haut à gauche se retrouve en bas à doite.

Imaginez en d'autres ;)