Rapport IHM Projet 1: Interface d'optimisation du choix de couleurs

François Lepan Benjamin Van Ryseghem

25 mars 2013

Introduction

Ce rapport fait état de l'utilisation ainsi que de la description de l'interface. Nous verrons comment l'utiliser, quels procédés ont été mis en oeuvre ainsi qu'un UML résumant les classes principales ainsi que leur fonctionnalités.

1 L'interface

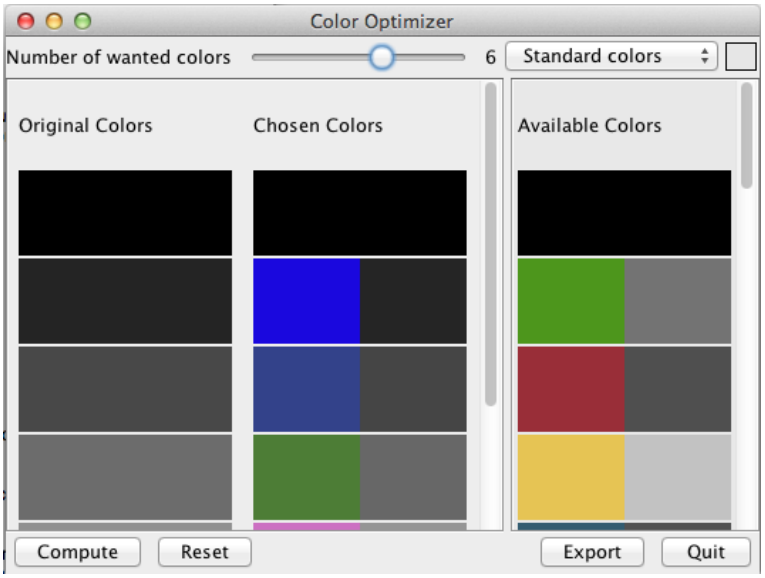

Figure 1 – L'interface d'optimisation du choix de couleurs

L'interface (*cf.* Fig. 1)choisie est très basique il y a un curseur et une liste en haut, trois colonnes au centre et quatre boutons en bas.

Le curseur en haut permet de choisir le nombre de nuance de gris que l'application doit générer et la liste permet de choisir le type de couleur à générer.

Au centre les deux colonnes à gauche représente les couleurs que l'utilisateur va choisir. La colonne de gauche contient les nuances de gris générées par l'application et la colonne au centre contient le choix de l'utilisateur.

Pour choisir de nouvelles couleurs l'utilisateur n'aura qu'a prendre les couleurs du pool de couleur proposé. Ce pool se situe dans la colonne à droite de l'interface.

En bas se trouve les différents boutons. Les deux boutons à gauche servent à modifier la colonne du centre. Le bouton *Compute* permet de calculer des couleurs ayant une nuance assez marquée. Le bouton *reset* permet de vider cette colonne.

Ensuite le bouton *Export* copie dans le presse-papier les valeurs des couleurs pour la colonne du centre.

Et enfin le bouton *Quit* permet de quitter l'application.

2 L'utilisation

L'interface est très simple d'utilisation. L'utilisateur choisie le nombre de couleur dont il a besoin via le curseur, un pool de couleur est créé (colonne centrale).

Si le choix calculé par l'application ne lui plait pas il peut à tout moment choisir de nouvelles couleurs via de simple cliqué glissé ou copier/coller parmi le pool de choix (colonne de droite).

Il peut grâce aux bouton *Compute* et *reset* recalculer ou remettre à zéro le choix des couleurs. Si il a fini exporter les couleurs dans le presse-papier grâce au bouton *export*.

Et enfin si il veut quitter l'application il n'a qu'a cliquer sur le bouton *quit*.

3 Les procédés utilisés

Nous avons mis en place un cliqué glissé, un copier/coller ainsi que la création d'un composant *DualColorComponent*.

Le cliqué glissé nous paraissais essentiel pour que cette interface soit la plus facile à utiliser, plus intuitif. En effet l'utilisateur pourra prendre des couleurs parmi le pool de couleur pour les mettre dans la colonne de choix par un simple cliqué glissé.

Le copier coller est un plus pour les utilisateur qui préfère utiliser leur clavier.

Nous avons aussi créer un composant nommé *DualColorComponent*. C'est ce composant qui alimente la colonne centrale ainsi que la colonne de droite. Cette Classe est constitué de deux JPanel l'un contenant la couleur et l'autre son niveau de gris. Cette élément est très intéressant car il permet a l'utilisateur de toujours avoir une vue direct sur la couleur et son niveau de gris sans effort. Cela lui permettra de faire des choix plus rapide et précis.

Conclusion

Cette interface répond bien au problème posé qui est d'aider l'utilisateur dans son choix de couleur pour le passage en niveau de gris. L'interface permet de le faire sans effort (ou presque) et en un temps minimum grâce aux procédé mis en place.

4 Annexe

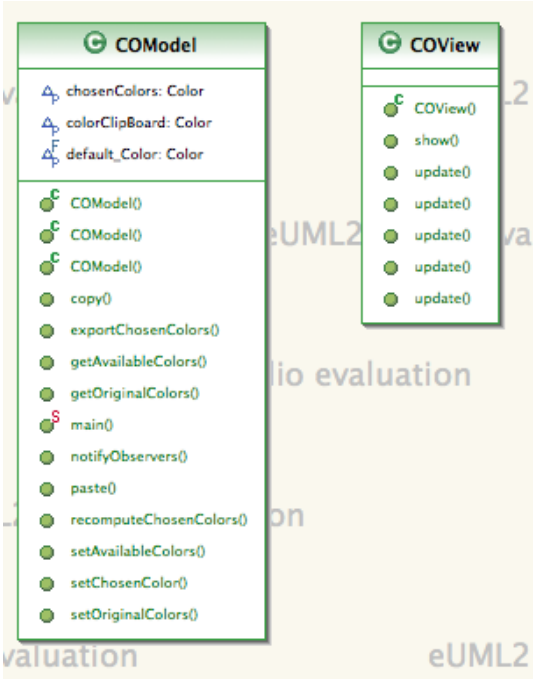

FIGURE 2 – UML du package core : le modèle et la vue

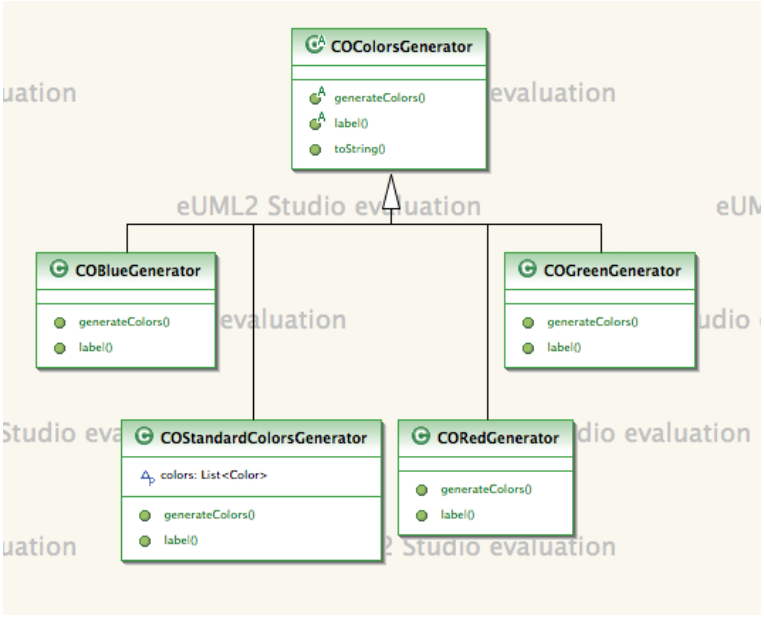

FIGURE 3 – UML du package color.generator : les classes qui s'occupent de générer les nuances de gris avec leurs couleurs respective

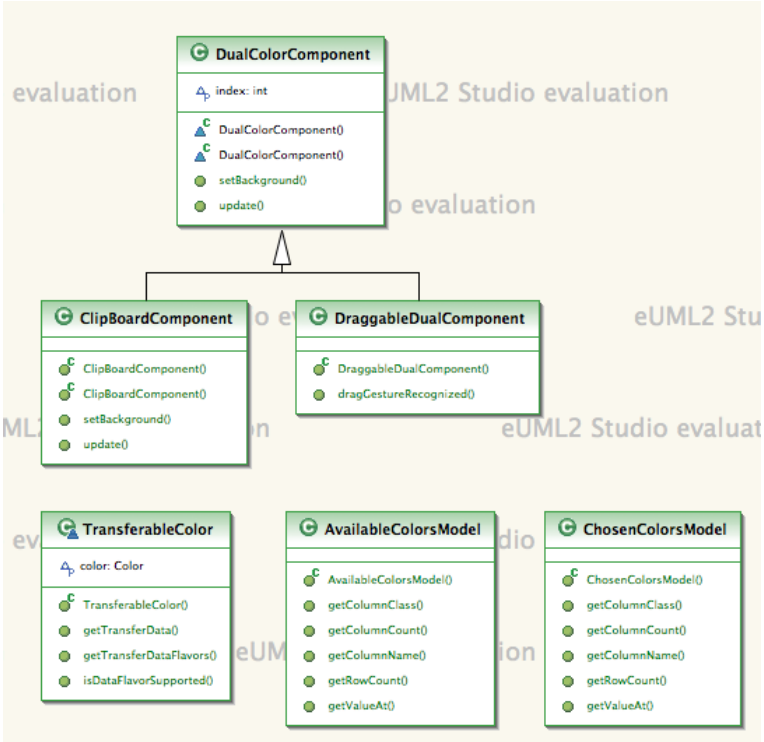

FIGURE 4 – UML du package support : les classes qui permettent le cliquer/glisser et copier/coller des couleurs ainsi que la classe DualColorComponent (toutes les classes ne sont pas présentes justes les essentiels)

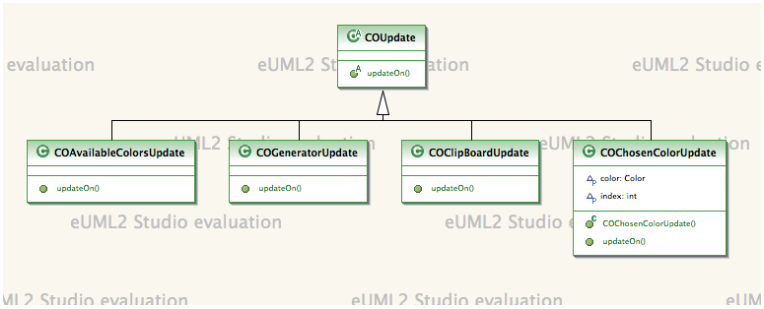

FIGURE 5 – UML du package update : classes permettant la mise à jour des composants