INFO-F-202
Langages de programmation 2
Université libre de Bruxelles
Laboratoire 12

Justin Dallant John Iacono
Yannick Molinghen Alexis Reynouard

1 But

Plus de pratique sur la copie, le déplacement et la mémoire dynamique.

2 Sommaire

On vous donne un code qui vous permet de dessiner dans quatre rectangles. Chaque rectangle ne se souvient que des quatre derniers traits et les colore en noir, bleu, rouge et vert. Le noir est le trait le plus récent et le vert le moins récent des quatre.

Lorsque vous appuyez sur 1, le contenu du premier rectangle est copié dans les autres rectangles. Lorsque vous appuyez sur 2, le contenu de chaque rectangle est déplacé vers le rectangle de droite et le rectangle le plus à gauche est maintenant vide.

Cependant, il y a plusieurs problèmes avec le code que vous devez corriger :

- Lorsque vous dessinez quelque chose dans le rectangle de gauche, appuyez sur 1, puis dessinez davantage dans le rectangle de gauche, les couleurs des autres rectangle changent ! Quelque chose est cassé avec la copie. Répare le.
- Lorsque vous appuyez sur 2, pour des raisons d'efficacité, vous devez DÉPLACER le contenu d'un rectangle à l'autre, et non le COPIER. Corrigez le code pour que le déplacement soit correctement implémenté.
- Il y a deux fuites de mémoire dans le code. Trouvez-les et réparez-les.
- Le code pour appuyer sur 2 ressemble à :

```
        case '1': // THIS IS THE COPY
          for (int i=1;i<drawCanvases.size();i++)
              drawCanvases[i]=drawCanvases[0];
          break;
```

  Si vous démarrez la boucle `for` à 0 au lieu de 1, cela ne devrait rien changer. Vérifiez et voyez si cela casse votre code.

Pour vous assurer que votre solution fonctionne, ajoutez les `cout<<`s aux méthodes de copie et de déplacement que vous écrivez. Assurez-vous que lorsque vous appuyez sur 1, les méthodes de copie sont appelées et lorsque vous appuyez sur 2, seules les méthodes de déplacement sont appelées.

3 Conseils

— Commencez par comprendre comment les données sont stockées dans DrawCanvas. Il a un tableau de taille 4 de pointeurs vers des objets de type ColoredStroke. Chaque ColoredStroke est une structure qui contient une couleur et un vecteur avec tous les points du trait. Les quatre pointeurs sont initialement à zéro, mais dans `DrawCanvas::processEvent` de nouveaux traits sont créés et ceux existants sont déplacés et recolorés.
— Vous avez découvert les constructeurs de copie, les constructeurs de déplacement, l'assignation de copie, l'assignation de déplacement, les destructeurs. En voyez-vous dans ce code ? Où doivent-ils aller et que doivent-ils faire ?
— Comment déplacez-vous les choses ? Qu'est-ce que `std::move` faire ?
— Rappelez-vous que les objets STL (tels que `std:vector`) implémentent généralement la copie et le déplacement correctement et efficacement.
— Vous pouvez ajouter uniquement des méthodes à DrawCanvas. **Vous n'avez pas besoin de modifier le code existant dans DrawCanvas.** Vous devrez également apporter une très petite modification dans `mainWindow::handle()` pour que le mouvement fonctionne correctement.