Philatel

TP1

Objectif

- Amélioration d'un programme existant
- Implémentation et/ou de plusieurs patrons : Fabrique, Commande, Itérateur, Singleton

Remise

- La remise se fait via Léa
- Corrigé selon le barème sur 5.
- N'oubliez pas d'inclure le nom de votre coéquipier (un readme.txt, dans Program.cs, dans les commentaires de remise, …)

Description

- À faire seul ou en équipe de 2.

A. Intégrez votre pile limitée dans le programme pour les commandes annulables en tant que DLL.
B. Faites un nouveau projet de type DLL pour ajouter un type de produit philatélique : « planche non coupée » (*uncut sheet*) de timbres (un gros bloc de timbre), il faut conserver le nombre de timbres, le nombre de timbres différents, le nom du designer, le nombre de lignes et de colonnes, etc. Voir le site de Postes Canada, par exemple https://www.canadapost.ca/shop/stamps/collectible-stamps/uncut-press-sheets.jsf?execution=e1s1 ou https://www.canadapost.ca/web/fr/search.page?query=planche+non+coup%C3%A9e&segment=Boutique&LOCALE=fr
   Il faut tout ajouter ce qu'il faut (boîte de dialogue, fabriques, etc.) selon la structure courante du programme. Copiez le DLL dans le dossier de la solution afin de l'activer dans le programme.
C. Ajoutez le contraire de l'*Annuler* (*Undo*) qui peut s'appeler *Rétablir* ou *Annuler Annuler* (raccourci clavier *Ctrl+Y*).
   Il vous faudra une autre pile limitée : les commandes effectuées s'empilent dans la pile existante et les commandes annulées sont transférées dans une autre pile de commandes qu'on peut rétablir. Et vice-versa (une commande rétablie retourne dans la pile des commandes qu'on peut annuler).
   Dès qu'il y a une nouvelle opération, les commandes rétablissables disparaissent, car on ne pourrait pas nécessairement les refaire.
   Si vous avez des doutes sur le comportement à implémenter, essayez l'éditeur de Visual Studio ou Word (mais n'essayez de reproduire *Répéter*, qui est tout autre chose et n'a pas vraiment de sens dans un programme comme Philatel).
D. Ajoutez la sérialisation des commandes annulables (pas celles pouvant être rétablies). On pourra donc annuler des commandes même si le programme est fermé puis rouvert.
E. Modifiez la saisie des articles afin que le Motif soit saisi dans un combo déroulable où seront présentés les motifs déjà inscrits dans le système (on peut aussi en taper des nouveaux).
F. Ajoutez une opération « Effacer tout » dans le menu. Après une confirmation (simple Message Box), elle fait enlever tous les articles philatéliques (évidemment, on peut l'annuler, etc.).
G. Vous allez modifier le type de base ArticlePhilatélique et la boîte de dialogue de base (tout le reste du programme devrait automatiquement s'ajuster ou presque…).

a. La valeur pour le prix payé doit être facultative, on utilisera un type annulable (Nullable). Le champ dans la boîte de dialogue peut donc rester vide. Vous devrez probablement modifier certains opérations d'affichage pour en tenir compte.
b. Ajoutez un champ texte TailleEtForme : on pourra y écrire des choses comme "11 mm par 20 mm", "Rond, 2,4 cm", "Variés", etc. (sans autre validation que d'être requis). Ce champ de saisie sera placé sous le motif dans la boîte de dialogue.

H. Créez une fonction EnOrdreAlphabétique qui implémentera l'itérabilité dans LesFabriques. Vous pourrez ainsi enlever la fonction CompléterLeMenu de cette classe (on n'aura plus besoin du using System.Windows.Forms;, enlevez-le). Le code pour compléter le menu ira où ça devrait être, dans FormPrincipal. Le parcours de LesFabriques devrait renvoyer des fabriques (IFabriqueCommande) seulement, pas des KeyValuePair et il devrait renvoyer les fabriques dans l'ordre alphabétiques des descriptions (pour menu).

I. Le sous-menu de *Ajouter* est créé dynamiquement à partir des types dans le programme, et ceux trouvés dans les *dll*. Mais ceux qui créent les *dll* ne savent pas quels autres types existent ou sont dans le programme (ou, du moins, ne devraient pas avoir à le savoir). Ils ne peuvent donc pas créer avec certitude une description pour le menu qui ne sera pas en conflit pour le raccourci clavier. Vous allez régler ça en générant le raccourci directement à partir des textes du menu :
   a. Renommez DescriptionPourMenu en une propriété DescriptionDuType (ça ne devrait pas être une méthode) et enlevez les & des implémentations de la fonction.
   b. Lors de la création des entrées du menu, mettez le & en première position, sauf si ça créerait un conflit. Dans ce cas, essayez de le mettre juste avant la première lettre du prochain mot, ou du suivant, etc. Si ça ne fonctionne toujours pas, on devrait choisir une voyelle « libre » quelconque. Dans votre cas, ça donnera *&Bloc de coin*, *&Planche non coupée*, *Pli premier &jour officiel (PPJO)*, *&Timbre seul*. N'oubliez de tester les autres cas possibles…

J. La classe LesFabriques ne contient que des méthodes statiques. C'est une pratique qui peut rendre le code plus difficile à tester et qu'on essaie d'éviter. Implémentez le patron Singleton pour cette classe.

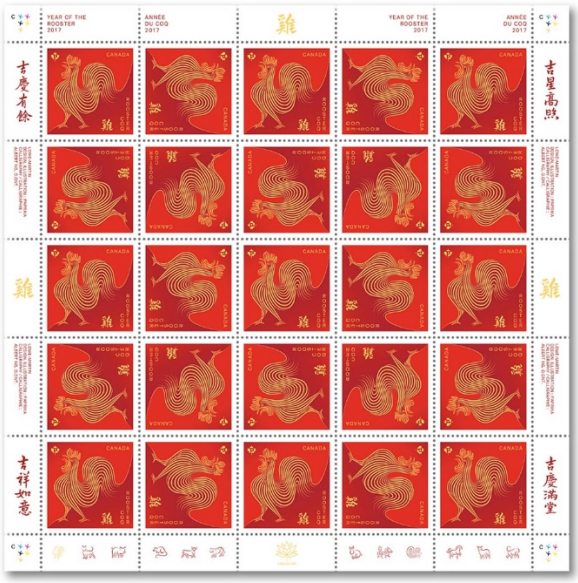
K. Les articles philatéliques étant souvent achetés en groupe lors de leur sortie (un timbre, avec des blocs de coin du même motif ou le pli premier jour), vous allez simplifier la création des objets de la même famille. Après l'ajout d'un nouvel article, vous ajouterez au menu *Ajouter* des choix pour ajouter des types d'articles (ceux de LesFabriques) avec le même motif. Par exemple, après l'ajout d'une feuille de timbres de motif *Superman*, le menu présentera les choix *Timbre seul Superman*, *Bloc de coin Superman*, *Feuille de timbre Superman*, *PPJO Superman*, etc. Ces choix font remplir les champs de base avant le début de la saisie par l'utilisateur, pas seulement le motif. Votre menu contiendra au maximum la possibilité de créer les trois derniers motifs (en plus des versions sans motif). Il devrait également se mettre à jour en temps réel (suivant une création ou annulation). N'oubliez pas de faire générer de bons raccourcis clavier. Faites attentions à ne pas avoir deux fois la même famille.

L. Afin de limiter les risques de perte de données, modifiez la sauvegarde des articles philatéliques et des commandes annulables afin qu'ils soient également enregistrés après qu'une commande soient exécutée avec succès.
   a. Fermer l'application : on enregistre
   b. Ajouter/Modifier/Supprimer un article : on enregistre.
   c. Annuler/Refaire : les fichiers restent dans l'état actuel, on attend avant d'enregistrer

Exemples d'articles philatéliques (pour vous aider à visualiser le TP !)

Planches non coupées (des feuilles de timbres qui seront coupées, avant la vente, en timbres individuels ou en petits blocs de plusieurs timbres)

25 fois le même timbre :



6 blocs de 5 timbres différents (et pouvant avoir différentes valeurs) :

Blocs de coin (les coins des feuilles… c'est les petits trucs écrits dans les marges qui intéressent les collectionneurs)

Coin supérieur droit

Coin inférieur gauche

Plis premier jour officiels

Des enveloppes spécialement faites pour le ou les timbres qui sont présentés. Les timbres sont oblitérés à la date de parution du timbre. Le verso de l'enveloppe comporte un texte explicatif sur les motifs des timbres, dont je n'ai malheureusement pas d'exemple ici

PPJO d'un bloc de 5 timbres de même valeur :

PPJO avec trois timbres différents et de différentes valeurs :

On ne traite pas ça directement dans le programme, mais voici un PPJO d'un bloc de coin (inférieur gauche)