



Cégep de Saint-Hyacinthe
Département d'informatique

Programmation orientée objet (420-2DP-HY)

Travail Pratique #3 (Mobilier de la Ville de Montréal)

Préparé par
Martin Lalancette
bureau B-2230

DESCRIPTION

But :	LINQ/Lambda, classes statiques et formulaires non modal.
Objectifs :	<ul style="list-style-type: none"> • Appliquer les principes logiques algorithmiques avec une bonne syntaxe en suivant vigoureusement les normes de programmation • Création et utilisation de classes statiques • Création et utilisation de collections • Applications multi-formulaires (non modal) • Utilisation de requêtes LINQ/Lambda • Apprentissage de nouvelles composantes visuelles • Être capable d'utiliser l'outil MS Visual Studio
Durée :	9 h 00
Pondération :	Sur 15
Remise :	À la fin de la semaine 12.
Contenu général :	<ul style="list-style-type: none"> • Remettre vos documents d'analyse et votre solution contenant le projet dans un document .ZIP via LÉA.
Notes	<ul style="list-style-type: none"> • Conserver une copie de sécurité. Il est de <u>votre responsabilité</u> de conserver une copie de sécurité dans l'éventualité où la lecture des données serait impossible. Cette copie doit <u>être disponible sur demande</u>.

Spécifications du travail



La ville de Montréal sollicite vos compétences pour concevoir une application de type WPF servant à gérer l'inventaire de son mobilier installé sur son territoire. Elle met à votre disposition ses données recueillies sur le terrain concernant le positionnement et les caractéristiques de chaque mobilier installé un peu partout dans sa ville. **Vous devez accéder au site suivant pour télécharger les données de type GEOJSON :** <http://donnees.ville.montreal.qc.ca/dataset/mobilierurbaingp>. Vous devrez l'analyser et bâtir vos classes en conséquence.

PARTIE #1 : Les interfaces de base exigées

Cette partie consiste à programmer les interfaces ici-bas demandées. L'apparence visuelle est **laissée à votre discrétion (et doit être améliorée)**, mais les fonctionnalités doivent demeurer les mêmes. Voici les 2 fenêtres à faire :

ListView

Nombre: 6797

Charger Filtrer

Filtres...

Options de tri:

- ☒ Aucun ☐ Descendant
- ☐ ID
- ☐ Type
- ☐ Modèle
- ☐ Description
- ☐ Parc
- ☐ Installation
- ☐ Inspection

Options de filtre:

Description:

Type:

Parc:

Dates:

Début: Sélectionner ur

Fin: Sélectionner ur

Filtrer

Ouverture et gestion non modales (Boîte à outils)

La liste est mise à jour suite au clic.

Liste contenant tous les types possibles de mobilier. À bâtir à partir de la collection avec une requête et avec le champ *ÉlémentType*.

Contient :

- Installation
- Inspection

Une fois les options de tri et/ou filtre sélectionnées, un clic doit filtrer le contenu en LINQ/Lambda.

Consignes :

1. Créer un projet de type **WPF App .Net Core**. Récupérer le fichier GEOSON du site de la [ville de Montréal](#). Ajouter cette solution/projet au projet Azure DevOps (GIT).
2. **Bouton Charger** : Charger le fichier JSON dans une collection (Créer la classe et la liste). Utiliser la composante **System.Text.Json** pour charger le fichier. **Créer une classe statique pour gérer la liste des mobiliers (collection)** afin de la rendre disponible pour l'ensemble de l'application.

3. **Bouton Filtrer** : Doit ouvrir la fenêtre Filtre... en mode non modal. Les items dans un combobox doivent être chargés à partir de la liste des mobilier (ÉlémentType). Un clic sur Filtrer doit aviser la fenêtre principale via un événement.
4. **Case Tout sélectionner** : Lorsque l'utilisateur coche cette case, tous les éléments affichés dans la liste doivent être cochés. Inversement si décoché.
5. **Améliorer l'aspect visuel des fenêtres.**
6. Ajouter les commentaires selon les normes.
7. Le tout doit compiler sans erreur.
8. **Produire un diagramme de classe.**

Barème d'évaluation

Partie 1 : Base (GIT avec 5 commits/dév., Normes, MVVM, diagramme de classes, etc.)	/2.0
Partie 1 : Conception des interfaces visuelles (avec personnalisation)	/4.0
Partie 1 : Programmation des filtres avec LINQ/Lambda	/4.0
Partie 1 : Fonctionnement général des 2 fenêtres.....	/5.0
Note totale*	/15.0

* Tout travail plagié en partie ou en totalité se verra attribuer une note totale de 0 %.

PDEA #1: Lors d'activités d'évaluation sommative en classe ou hors classe (documentation, rapport de laboratoire, rapport de stage, examen), une **pénalité maximale de 10 %** peut être retranchée de la note finale de ladite évaluation (le barème étant de **0,5%/erreur incluant les fautes répétitives**). Pour les commentaires dans les programmes informatiques, le barème est de **0,5% par faute**. Pour les interfaces utilisateurs, le barème est de **1% par faute**.

PDEA #4 : Toute évaluation sommative remise **après la date d'échéance fixée** se voit **attribuer la note zéro.**