Concepteur Développeur en Informatique

Module 1 Développer l'interface d'une application informatique

| Programmation Objet P2 | Séance | S04 | Activité | A-002 |
|---|---|---|---|---|

Cette activité vous permet de poursuivre l'apprentissage des concepts objets en mettant en œuvre des techniques permettant d'assurer la persistance de ces derniers :

- Assurer la persistance des objets en utilisant les techniques de base de manipulation de fichiers.
- Assurer la persistance des objets en recourant aux mécanismes de sérialisation.

Vous aurez l'occasion de plus d'approfondir le traitement de chaines de caractères.

Sommaire de l'activité proposée :

# 1 Manipuler des fichiers texte

Dans ce premier exercice, vous devez mettre en œuvre les mécanismes de base de manipulation de fichiers.

Votre programme met en œuvre deux fonctionnalités.

Une fonctionnalité qui permet l'acquisition de mots et le stockage de ceux-ci dans un fichier texte. Les données sont encodées en UTF8. Le programme stocke un mot par ligne d'enregistrement.

Une fonctionnalité qui permet d'extraire les mots et les afficher à la console.

# 2 Assurer la persistance des états de vos objets

Vous devez mettre en place les mécanismes qui permettront de rendre persistant l'état de vos objets métier de type Salarie (et commercial)

## 2.1 Implémentez les méthodes SaveText() et LoadText()

Ces méthodes sont des méthodes d'instance de votre type Salaries qui permet de gérer un ensemble de salariés (collection).
Ces méthodes prennent en argument le chemin physique du fichier où seront conservés vos enregistrements de salariés.
Par convention le fichier porte le nom du type qu'il représente, ici Salaries, avec une extension (csv pour texte avec séparateur).
Elles sont implémentées avec des objets FileStream et les mécanismes spécialisés pour l'écriture et la lecture de texte vus précédemment (TextReader et TextWriter)

La méthode SaveText() doit permettre de sauvegarder les propriétés de vos objets dans un enregistrement (ligne). Une ligne de votre fichier correspond à un objet de type Salarié.

La méthode ToSTring() programmée au niveau de vos types Salarie et Commercial doit vous permettre de vous faciliter la tâche. Ecrivez une ligne par objet dans votre fichier.

Pour mettre en place la méthode LoadText() et charger les propriétés d'un objet à partir d'un enregistrement, utilisez la méthode Split().
La méthode Split permet de découper une chaîne et stocker chaque portion de chaîne dans un tableau. Voir support sur le langage C# ou la documentation de la classe String.

## 2.2 Implémentez les méthodes SaveBinary(), SaveXML(), LoadBinary() et LoadXML()

Ces méthodes prennent elles aussi en argument le chemin physique du fichier où seront conservés vos enregistrements de salariés. Par convention le fichier porte le nom du type qu'il représente, ici Salaries, et une extension (XML pour les fichiers XML, dat pour les fichiers binaires).

Préparez vos types en vue de leur sérialisation :

Définissez les attributs de vos classes Salaries, Commercial et Salarie afin qu'elles puissent être sérialisées. Attribut [Serializable()].

```
[Serializable()]
public class Commercial : Salarie
{
```

Précisez que vous souhaitez inclure dans la sérialisation XML des salariés, classe Salaries, le type Commercial. C'est obligatoire en XML.

```
[Serializable()]
[XmlInclude(typeof(Commercial))]
public class Salaries : List<Salarie>
{
```

## 2.3 Tester opérations sérialiser / désérialiser

Créez un programme interface utilisateur de type console afin de tester le bon fonctionnement des mécanismes de sérialisation et dé-sérialisation.

Ce programme doit autoriser l'ajout d'un nouveau Salarié et la modification ou la suppression d'un salarié existant dont on précisera le matricule.

Vous devez vous assurer que les états de vos objets ont correctement été sauvegardés.

**Attention :** Pour les mécanismes de sérialisation et de dé-sérialisation, le système doit recourir à un constructeur par défaut, le constructeur vide. Si vous avez défini un constructeur avec une autre implémentation que celle par défaut, il vous faut définir explicitement un constructeur par défaut.

Vous devez aussi prendre garde à la génération d'exceptions qui peut survenir du fait des contrôles implémentés au niveau des méthodes des accesseurs set et qui peuvent interrompre le processus de désérialisation...
**Attention donc** à ne stocker que des salariés dont les propriétés ont été vérifiées conformes aux règles programmées.