**Université Cheikh Anta Diop** **Année Universitaire 2021/2022**
**Faculté Sciences et Techniques** **Enseignement .NET C#**
**M2TDSI** **TP Accès au Système de Fichiers**
**Chargé de Cours E. H. Ousmane Diallo**

**23/04/2022**

## Réalisation d'un éditeur de texte simple

### Tâche 1: Ouvrir le projet EditeurSimple

1. Ouvrir la solution ***EditeurSimple*** contenu dans le dossier ***Souche 51***.

### Tâche 2: Afficher une boîte de dialogue pour lire le nom de fichier saisi par l'utilisateur

1. Afficher la fenêtre ***MainWindow.xaml***. Elle implémente un simple éditeur de texte. La fenêtre principale contient une zone de texte qu'un utilisateur peut utiliser pour Afficher et modifier le texte. Le bouton **Ouvrir** permet à l'utilisateur d'Ouvrir un fichier, et le bouton **Enregistrer** d'Enregistrer les modifications du texte. Vous allez ajouter le code qui implémente la logique pour ces deux boutons.
   **TODO - Implémenter une méthode pour lire le nom de fichier**.
2. Cette tâche est située dans la classe ***MainWindow.xaml.cs***. Définir une nouvelle méthode nommée ***LireNomFichier*** sans paramètres et qui renvoie une chaîne valeur qui contient le nom de fichier que l'utilisateur a indiqué.
3. Dans le corps de la méthode, déclarer un nouveau membre chaîne nommé ***nomFichier***, initialisée à la valeur ***String.Empty***.
4. A la suite de la série des instructions **using** du début du fichier, ajouter une instruction pour rendre l'espace de nom ***Microsoft.Win32*** visible.
5. Dans la méthode ***LireNomFichier***, après l'instruction qui déclare ***fnom***, ajouter le code suivant :
   a. Créer une nouvelle instance de ***OpenFileDialog***, nommée ***dlgOuvrirFichier***.
   b. Fixer la propriété ***InitialDirectory*** de ***dlgOuvrirFichier*** à ***@"C:\Souche51"***.
   c. Fixer la propriété ***Defaultext*** de ***dlgOuvrirFichier*** à ".txt";
   d. Fixer la propriété ***Filter*** de ***dlgOuvrirFichier*** à **"Texte Documents (.txt)|*.txt"**.
6. Ajouter le code pour les tâches suivantes :
   a. Appeler la méthode ***ShowDialog*** de ***dlgOuvrirFichier***, et enregistrer le résultat qui.est un Booléen ***nullable*** dans une variable.
   b. Si le résultat est ***true***, affecter la valeur de la propriété ***FileName*** de ***dlgOuvrirFichier*** à la variable ***fnom***.
   c. A la fin de la méthode, renvoyer la valeur dans la variable ***fnom***.

### Tâche 3: Implémenter un nouvelle classe pour lire et écrire le fichier texte.

1) Ajouter une nouvelle classe nommée ***EntreesSortiesTexte*** dans le projet ***EditeurFichier***.
   a) A la fin de la collection des instructions **using** au début du fichier, ajouter une instruction pour rendre l'espace de nom **System.IO** visible.

1. Dans la classe, ***EntreesSortiesTexte*** ajouter une méthode publique statique nommée ***LireTexte***. La méthode accepte un paramètre chaîne nommée ***nomFichier***, et renvoie un objet chaîne.
2. Dans la méthode ***LireTexte***, ajouter le code pour renvoyer le contenu entier du texte dont le chemin est indiqué dans le paramètre ***nomFichier***. Utiliser la méthode statique ***ReadAllText*** de la classe ***File***.
3. Après la méthode ***LireTexte***, ajouter une méthode publique statique nommée ***EcrireTexte***. La méthode ne renvoie aucune valeur. Elle accepte 2 paramètres: ***nomFichier*** et ***contenuFichier*** de type chaîne.
4. Dans la méthode ***EcrireTexte***, ajouter le code pour écrire le texte qui est contenu dans le paramètre ***contenuFichier*** dans le fichier qui est indiqué dans le paramètre ***nomFichier.*** Utiliser la méthode statique ***WriteAllText*** de la classe ***File***. 2) Construire et exécuter la solution.

**Tâche 4: Mettre à jour les gestionnaires d'événement de la classe *MainWindow* pour utiliser la classe *EntreesSortiesTexte*.**

1. **TODO - Mettre à jour la méthode *BoutonOuvrir_Click*** Effectuer les Tâches suivantes :
    a. Appelez la méthode ***LireNomFichier***. Stockez le résultat de la méthode dans ***nomFichier***.
    b. Si ***nomFichier*** n'est pas une chaîne vide, Appeler la méthode ***LireTexte*** de la classe ***EntreesSortiesTexte***, et passer le paramètre ***nomFichier*** :
    c. . Stockez le résultat dans la propriété ***Text*** de la zone de texte ***editeur*** de la fenêtre ***WPF***.
2. **TODO - Mettre à jour la méthode *BoutonEnregistrer_Click***.
    a. Si ***nomFichier*** n'est pas une chaîne vide, appeler la méthode ***EcrireTexte*** de la classe ***EntreesSortiesTexte***. Passer les paramètres **nomFichier** et le texte de la zone de texte ***editeur***.
3. Construire et exécuter la solution.
4. Démarrer l'application sans débogage.
    a. Dans la fenêtre principale, cliquer sur **Ouvrir**. Dans la boîte de dialogue ***Ouvrir***, sélectionner le fichier ***Commands.txt*** **et** cliquer sur ***Ouvrir***.
    b. Changer le "**Store 30"** en **"Enregistrer 50"**, et cliquer sur ***Enregistrer***.
5. Ouvrir Notepad et vérifier la modification.
6. Fermer ***Visual Studio***.

**Exercice 2: Permettre à l'Editeur de reconnaitre le code XML**

Par exemple, si un fichier texte contient les instructions suivantes :

Move x, 10

**23/04/2022**

```
Move y, 20
If x << y Add x, y
If x > y & x < 20 Sub x, y
Console.WriteLine($"Valeur de x : {x}’ Valeur de y {y}")
Store 30
```

Vous allez modifier l’application ***WPF*** pour rechercher les tags XML dans le fichier texte et les remplacer par des séquences d’échappement.

**Tâche 1: Ajouter une nouvelle méthode pour filtrer les caractères XML.**

1) Ajouter un nouvelle méthode publique statique nommée ***FiltrerTexte*** dans la classe ***EntreesSortiesTexte*** . La méthode accepte un paramètre nommé ***nomFichier***, et renvoie une chaîne.
2) Dans ***FiltrerTexte :***
   a) instancier un objet ***contenuFichier*** de type ***StringBuilder ,***
   b) déclarer une variable locale ***codeCaractere*** de type ***int.***
   c) Instancier un objet ***fluxFichier*** de type ***StreamReader*** avec le paramètre ***nomFichier***.
   d) Ajouter un **while** qui lit chaque caractere de l’objet ***StreamReader*** jusqu’à la fin du fichier.
3) Dans le bloc ***while***, ajouter une instruction **switch** qui évalue ***charCode***, puis appelle la méthode ***Append*** de ***contenuFichier*** pour remplacer :
   - " par "
   - & par &
   - ' par '
   - < par <
   - > par >
   - { par &acg
   - } par &acd
   - $ par &dol
   - ( par parg
   - ) par pard
   - Ajouter un **case *default*** qui conserve le caractère lu si son code est différent des valeurs cidessus (un ***cast*** est obligatoire pour changer le type du caractère lu de ***int*** à ***char***)

1. A la fin de la méthode renvoyer ***contenuFichier*** converti en chaine.

**Tâche 2: Mettre à jour l’interface utilisateur pour appeler la nouvelle méthode**

1. Dans la méthode **BoutonOuvrir_Click** modifier la ligne de code qui appelle ***LireTexte.*** Appelez la méthode ***FiltrerTexte*** à la place en transmettant le paramètre ***nomFichier*** et enregistrer le résultat dans la propriété ***Text*** de la zone de texte ***editeur*** de la fenêtre ***WPF***.
2. Testez l’application, en ouvrant le fichier ***C*:\Souche51\Commands.txt**.

7. Fermer la fenêtre principale et quitter Visual Studio.