Laborator 4 – Citirea și scrierea fișierelor

Obiective

După completarea acestui laborator veți dobândi următoarele cunoștințe:

- Să folosiți clasele **FileInfo** și D**irectoryInfo** pentru a copia fișiere și foldere
- Să folosiți locații izolate pentru date specifice utilizatorilor

Scenariul de laborator

Vi s-a cerut să scrieți o aplicație numită BackUp ce permite backup-ul fișierelor și a folderelor. Deoarece aplicația va fi necesar să se integreze cu o serie de scripturi, aplicația va avea forma unei aplicații pentru consolă.

Aplicația va primi 3 argumente linie de comandă, directoarele sursă și destinație, un flag ce specifică informații despre vechimea fișierelor pt a le selecta pe cele ce trebuie arhivate, și încă un flag ce va specifica dacă fișierele trebuie și comprimate odată ce sunt arhivate. Dacă folderul sursă conține și sub-foldere, acestea vor fi de asemenea luate în considerare.

Toate argumentele sunt obligatorii, un apel al programului făcându-se de forma:

```
BackUp <source folder> <destination folder> <age>
```

Un exemplu concret: un apel ce va face backup tuturor fișierelor mai vechi de 10 zile dintr-un folder:

```
BackUp E:\SourceFiles E:\Destination 10
```

➢ Exercițiul 1- Copierea fișierelor

În acest exercițiu, veți copia fișierele dintr-o locație către destinație. Doar fișierele ce respectă criteriile de vechime vor fi copiate. Laboratorul nu are un startup, se va pleca de la un proiect nou, Principalii pași sunt:

- Parsarea argumentelor linie de comandă
- Verificarea validității directoarelor sursă și destinație
- Arhivarea fișierelor ce respectă criteriile

Etapa 1: Parsarea argumentelor linie de comandă

- Faceți verificări privind numărul și tipul argumentelor furnizate
- Realizați parsarea argumentelor și aruncați o excepție de tipul **ArgumentException** dacă nu au fost furnizate destule argumente sau dacă formatul acestora nu este corect

Etapa 2: Verificarea valabilității directoarelor

- Creați metoda statică privată ce întoarce void numită **CheckDirectories î**n clasa Program. Această metodă primește doi parametri de tipul string, sursa și destinația.
- Verificați, folosind metoda **Exists** a clasei **Directory**, dacă folderele sursă și destinație există. Dacă folderul sursă nu există, aruncați o excepție de tipul **ArgumentException**. Dacă folderul destinație nu există, creați-l.

- Apelați metoda în metoda Main, trimițând ca parametri cele două directoare primite ca argumente.

Etapa 3: Copierea fișierelor ce respectă criteriile

- În clasa Program, adăugați o metodă statică privată ce întoarce bool numită **BackUpFiles**. Această metodă primește ca parametri 2 string-uri, un folder sursă și unul destinație.
- În această metodă, instanțiați clasa **DirectoryInfo** furnizând folderul sursă ca parametru constructorului. Numiți variabila **sourceInfo**; asemănător, pentru destinație, variabila se va numit **destInfo**
- Apelați metoda **GetFiles** pentru **sourceInfo** pentru a obține o listă a tuturor fișierelor din folder sub forma unor obiecte **FileInfo**. Numiți această listă **fileList**.
- Iterați pe vectorul **fileList**. Dacă un anumit fișier îndeplinește condiția de vechime (primită ca argumet linie de comandă), folosiți metoda **CopyTo** a obiectului **FileInfo** pentru a copia fișierul în folderul specificat de obiectul **destInfo**. Suprascrieți un fișier ce exsită în folderul destianție dacă are același nume.
- La finalul metodei **BackUpFiles**, apelați metoda **GetDirectories** pentru a obține o listă a tuturor subdirectoarelor din sursă. Salvați directoarele în lista **childFolderList**
- Iterați prin elementele din listă, pentru fiecare dintre acestea:
    - Extrageți numele directorului și construiți apoi numele destinație plecând de la numele folderului destinație; folosiți clasa **Path**
    - Creați un subdirector nou în folderul destinație furnizând numele construit anterior
    - Apelați recursiv metoda **ArchiveFiles** oferind ca parametri subdirectoarele din sursă și din destianție
- Apelați metoda **BackUpFiles** în metoda Main
- Compilați și testați aplicația.

OPTIONSFILE

➢ Exerciţiul 3 – salvarea și citirea preferințelor utilizatorilor

În acest exercițiu, veți modifica aplicația BackUp pentru ca aceasta să rețină cel mai recent set de argumente linie de comandă într-o locație izolată. Acest lucru va permite utilizatorilor să ruleze aplicația folosind doar argumentul „replay” furnizat aplicației.

Principalele puncte ale acestui exercițiu sunt:

- Salvarea argumentelor într-o locație izolată
- Citirea acestora din locația izolată

Etapa 1: Salvarea argumentelor într-o locație izolată

- Adăugați folosind instrucțiunea **using** namespace-ul System.IO.IsolatedStorage în fișierul Program
- Definiți în clasa Program o constantă cu numele **OPTIONSFOLDER** cu valoarea **„BackUpOptionsFolder”** și una nu numele **OPTIONSFILE** și valoarea **„BackUpOptions”**
- Definiți o metodă privată, statică ce întoarce void cu numele **StoreOptionsInIsolatedStorage.**
- În această metodă, executați operațiile:
    - Creați un obiect de tipul **IsolatedStorageFile**, ce reprezintă store-ul utilizatorului ce folosește aplicația, apelând metoda **GetUserStoreForDomain** a clasei **IsolatedStorageFile**
    - Creați un folder în locația izolată apelând metoda **CreateDirectory** a obiectului **IsolatedStorageFile**. Oferiți ca nume al folderului valoarea constantei **OPTIONSFOLDER**
    - Creați un obiect de tipul **IsolatedStorageFileStream** pentru scrierea și citirea fișierului al cărui nume se găsește în constanta **OPTIONSFILE**

- Creați un obiect **StreamWriter** pentru a lucra cu acest stream de fișier
- Folosiți obiectul **StreamWriter** pentru a scrie valorile argumentelor linie de comandă.
- Închideți obiectul **StreamWriter** și **IsolatedStorageFile**

- În metoda **Main**, după parsarea argumentelor linie de comandă, adăugați o instrucțiune ce apelează metoda de mai sus.

Etapa 2: citirea valorilor stocate în fișier

- În clasa Program, adăugați o metodă privată statică ce întoarce void numită **PopulateFieldsFromIsolatedStorage.** Metoda nu primește parametri
- În această metodă, executați următorii pași:
  - Creați un obiect **IsolatedStorageFile** ce reprezintă store-ul utilizatorului curent apelând metoda **GetUserStoreForDomain** a clasei **IsolatedStorageFile**
  - Creați un obiect de tipul **IsolatedStorageFileStream** pentru citirea din fișierul al cărui nume se află în constanta **OPTIONSFILE** în folderul specificat de constanta **OPTIONSFOLDER**.
  - Creați un obiect de tipul **StreamReader** ce citește date din streamul deschis.
  - Citiți folosind acest obiect **StreamReader** argumentele linie de comandă folosite la ultima rulare.
- În metoda Main, modificați instrucțiunile ce verifică numărul de parametri pentru a permite fie specificarea unui parametru („replay”), fie 4
- Adăugați linii de cod ce, dacă s-a oferit parametrul „replay”, să se apeleze metoda **PopulateFieldsFromIsolatedStorage**.

Compilați și rulați aplicația.