

Laboratorul 4

Lucrul cu fișiere în Visual Basic .NET

Ce ne propunem astăzi?



În acest laborator ne propunem crearea în Visual Basic .NET a unei aplicații pentru evidența datelor angajaților dintr-o companie și memorarea lor într-un fișier text (vezi Figura 1).

Misc	
CNP	1730816144587
Domiciliu	Str. Garii, nr. 234
Funcție	inginer
Nume	Cristescu
Prenume	Emil
Salariu	3240
Varsta	36
Vechime	12

Figura 1. Interfața aplicației

Mai pe larg, vom proceda astfel...

Pentru afișarea datelor unui angajat vom utiliza un control de tip PropertyGrid. Acesta permite afișarea tuturor proprietăților unui obiect atașat acestuia. O proprietate a unei clase se definește cu ajutorul cuvântului cheie *Property* și, cel puțin în cazul aplicației curente, este de preferat în locul variabilelor membre ale clasei, din două motive:

- ne permite crearea unor valori care pot fi accesate numai în citire (read-only);
- ne permite afișare acestora în interiorul unui control de tip PropertyGrid.

Din punct de vedere al programării orientate pe obiecte, ca o extensie a principiului încapsulării, este de preferat folosirea proprietăților publice pentru accesarea câmpurilor (variabilelor membre) private ale unei clase. Același principiu era aplicat adeseori și în limbajul C++, atunci

când câmpurile private ale unei clase erau accesate din exterior prin intermediul metodelor publice.

Înainte de crearea unei proprietăți se va declara o variabilă membru (asupra căreia va putea opera proprietatea), apoi va fi declarată proprietatea împreună cu procedurile Get și Set, care permit returnarea unei valori, respectiv preluarea unei valori. Iată un exemplu de proprietate read-only:

```
ReadOnly Property Salariu() As Integer
    Get
        Return 1000 * _functie + _vechime * 20           ' valoare returnata
    End Get
End Property
```

Și un exemplu de proprietate care poate fi accesată atât în citire cât și în scriere:

```
Property Nume() As String
    Get
        ' proprietatea opereaza cu variabila membru _nume
        Return _nume ' valoare returnata
    End Get
    Set(ByVal value As String)
        _nume = value           ' valoarea primita este inscrisa in variabila _nume
    End Set
End Property
```

În continuare va fi adăugat un modul nou proiectului curent, iar în interiorul acestuia vor fi declarate două clase: clasa *Persoana* și clasa *Angajat* (derivată din clasa *Persoana*) și un tip enumerare (pentru stocarea funcției pe care o ocupă un angajat).

```
Enum Functii
    muncitor = 1
    maistru = 2
    inginer = 3
    economist = 4
    director = 5
End Enum

Class Persoana
    Protected _Nume As String
    Protected _Prenume As String
    Protected _CNP As String
End Class

Class Angajat : Inherits Persoana
    Private _Functie As Functii
    Private _Vechime As Integer
    Private _Domiciliu As String

    Sub New()
        ' constructor fara parametri
    End Sub

    Sub New(ByVal nume As String, ByVal prenume As String, ByVal cnp As String, _
        ByVal domiciliu As String, ByVal functie As Functii, ByVal vechime As Integer)
        ' constructor cu parametri
    End Sub

    ReadOnly Property Salariu() As Integer
        Get
            Return 1000 * _Functie + _Vechime * 20
        End Get
    End Property
```

```

ReadOnly Property Varsta() As Integer
    Get
        ' aici se va extrage varsta din CNP
    End Get
End Property

' Aici se vor mai defini proprietatile:
' Nume() As String
' Prenume() As String
' CNP() As String
' Functie() As Functii
' Vechime() As String
' Domiciliu() As String
End Class

```

Obiectele de tip *Angajat* vor fi stocate într-o listă de tip *List(Of Angajat)*, iar apoi afișate în controlul *PropertyGrid*:

```

Dim lista As New List(Of Angajat)
Dim ang As New Angajat(...) ' apelare a constructorului cu sau fara parametri
lista.Add(ang)
PropertyGrid1.SelectedObject = ang ' afisarea in PropertyGrid

```

Parcurgerea listei se va face prin intermediul unei bare de derulare orizontale (controlul *HScrollBar*). La fiecare adăugare a unui angajat în listă vom modifica proprietatea *Maximum* a barei de derulare, astfel încât să fie egală cu indexul ultimului element din lista de angajați (implicit *Minimum* are valoarea 0, iar *Maximum* are valoarea 100). Acest lucru permite ajustarea automată a dimensiunii barei de derulare în funcție de numărul de angajați existenți în listă (în Figura 1 se observă dimensiunea ajustată a barei de derulare pentru un număr de trei angajați). Se recomandă ca valorile proprietăților *SmallChange* și *LargeChange* să fie setate la 1.

Pentru parcurgerea listei de angajați veți avea nevoie să scrieți cod în handler-ul evenimentului *Scroll* asociat controlului *HScrollBar*.

După introducerea angajaților în listă, se dorește salvarea acestora sub formă de fișier text și deschiderea fișierului cu editorul implicit de text al sistemului de operare (vezi Figura 2).

Pentru crearea unui fișier text și scrierea de date în acesta este nevoie de un obiect de tip *StreamWriter* (tipul este definit în spațiul de nume *System.IO*). Pentru scrierea unei linii de text se apelează metoda *WriteLine* a obiectului de tip *StreamWriter*. Funcția *Shell* permite deschiderea unei aplicații cu transmiterea parametrilor pentru apelul din linie de comandă a acesteia.

```

Dim f As New System.IO.StreamWriter("c:\work\angajati.txt", False)
' ...
f.WriteLine(a.Nume + " " + a.Prenume)

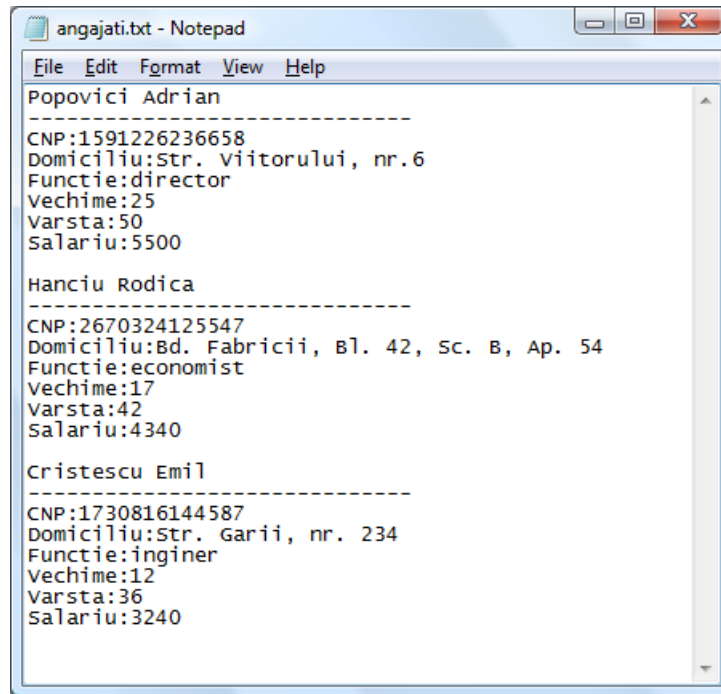
f.Close()
Shell("notepad d:\work\angajati.txt", AppWinStyle.NormalFocus, False)

```

În continuare vor fi implementate următoarele funcții:

- Ștergerea unui angajat
- Modificarea datelor unui angajat.
- Încărcarea listei de angajați dintr-un fișier salvat anterior.

Pentru aceste funcții vor fi adăugate noi butoane de comandă pe fereastra principală a aplicației.



```

angajati.txt - Notepad
File Edit Format View Help
-----
Popovici Adrian
CNP:1591226236658
Domiciliu:Str. Viitorului, nr.6
Functie:director
Vechime:25
Varsta:50
Salariu:5500
-----
Hanciu Rodica
CNP:2670324125547
Domiciliu:Bd. Fabricii, Bl. 42, Sc. B, Ap. 54
Functie:economist
Vechime:17
Varsta:42
Salariu:4340
-----
Cristescu Emil
CNP:1730816144587
Domiciliu:Str. Garii, nr. 234
Functie:inginer
Vechime:12
Varsta:36
Salariu:3240

```

Figura 2. Afișarea datelor salvate în fișier

Pentru operația de scriere în fișier nu veți avea nevoie de indecșii elementelor din enumerarea Functii, ci de numele acestora. Determinarea denumirii unui element dintr-o enumerare se face pe baza valorii acestuia prin următoarea sintaxă:

```
[Enum].GetName (GetType (Tip_enumerare), valoare)
'exemplu:
[Enum].GetName (GetType (Functii), 2) ' maistru
```

Cu ce ne-am ales?



Prin aplicația dezvoltată în cadrul laboratorului de astăzi am învățat pe de o parte să lucrăm cu controalele PropertyGrid și HScrollBar, iar pe de altă parte să efectuăm operații de scriere și citire asupra fișierelor text.

Bibliografie



[1] <http://msdn.microsoft.com/en-us/vbasic/default.aspx>