Cuprins C 5 - VB

Afişarea şi Validarea datelor

- **`MsgBox` şi `InputBox`**
- **Textbox**
- **Validarea datelor**
- **Listbox**
- **Listview**
- **Treeview**

În cursul care urmează se vor examina mai multe tehnici de colectare a datelor de la utilizator, de validare a lor şi de afişare a unor date complexe într-un format cît mai prezentabil.

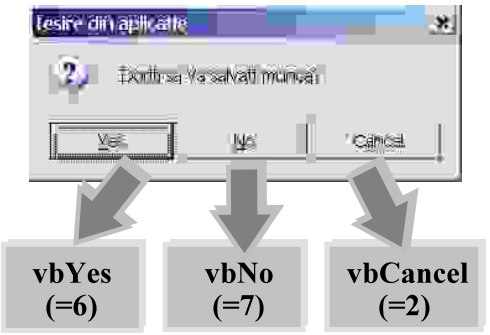
Funcţia **`MsgBox`** dă posibilitatea utilizatorului să ia decizii simple în timpul execuţiei programului, iar funcţia **`InputBox`** permite citirea unui şir de caractere introdus de utilizator.

Proprietăţile controlului **Textbox** vor fi examinate în detaliu pentru a vedea modalităţile cele mai eficiente de lucru împreună cu evenimentele care permit scrierea unor aplicaţii puternice de validare a datelor introduse. Controlul **Listbox** va fi discuta mai în detaliu pentru a putea oferi utilizatorului liste de opţiuni cît mai utile. Controalele **Listview** şi **Treeview** sînt obiecte avansate de afişare a datelor, se prezintă elementele de bază pentru utilizarea lor.

Funcția MsgBox

- **Funcția întoarce o valoare specifică butonului apăsat**
- **Permite controlul butoanelor afișate**
- **Permite controlul simbolurilor grafice (icon) afișate**

```
iRaspuns = MsgBox("Doriti sa Va salvati munca?", _
    vbQuestion + vbYesNoCancel, "Iesire din aplicatie")
```



1. Funcția MsgBox

`MsgBox` este un apel al unei funcții interne din Windows; ea întoarce o valoare în domeniul **`1-7`** în funcție de butonul care a fost apăsat.

Primul parametru este textul care va fi afișat în interiorul ferestrei.

Al doilea parametru este numeric; are utilizări multiple, se formează din adunarea unor valori numerice constante, predefinite, pentru a ajunge la efectul vizual dorit, cîteva dintre ele sînt:

- setarea butoanelor afișate (constantele **`vbOKOnly`**, **`vbOKCancel`**, **`vbAbortRetryIgnore`**, **`vbYesNoCancel`**, **`vbYesNo`**, **`vbRetryCancel`**).
- setarea butonului implicit (selectat automat la deschidere - **`vbDefaultButton1`** ... **`vbDefaultButton4`**);
- alegerea simbolului grafic afișat (**`vbCritical`**, **`vbQuestion`**, **`vbExclamation`**, **`vbInformation`**).

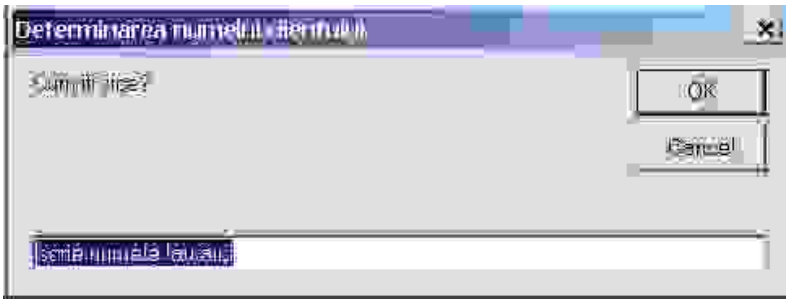
Al treilea parametru este textul afișat în bara ferestrei de mesaj.

Valoarea întoarsă depinde de butonul apăsat de utilizator.

Rutina implicită de tratare a erorilor

- **Obține un șir de la utilizator**
- **Poate fi preconfigurată**

```
strNume = InputBox("Cum iti zice?", _
    "Determinarea numelui clientului", _
    "scrie numele tau aici")
```



2. Funcția InputBox

Uneori dorim să obținem un șir sau o valoare numerică de la utilizator dar nu vrem să facem un formular nou numai pentru atît. Funcția **`InputBox`** este ideală în acest caz.

Primul argument (întrebarea) apare în interiorul ferestrei de intrare, le fel cu primul argument din **`MsgBox`**. Următorul argument (titlul) apare în bara de titlu a ferestrei de intrare, iar al treilea (text implicit) este plasat în cutia cu text din interiorul lui **`InputBox`**, selectată pentru a fi gata de suprascriere.

Conținutul lui **`TextBox`** este întors de funcție atunci cînd utilizatorul apasă butonul **OK.** Dacă utilizatorul apasă butonul **Cancel** se va întoarce un șir vid.

Proprietăți importante ale lui Texbox

`Text`	**Conținutul șirului scris în Textbox**
`MaxLength`	**Numărul maxim de caractere**
`MultiLine`	**Da sau Nu pentru linii multiple**
`PasswordChar`	**Caracterul de afișat în cazul de parolă**
`Scrollbars`	**Tipul barelor de defilare folosite**
`Locked`	**Blocare împotriva editării**
`DataChanged`	**Modificarea textului**

3. Proprietăți importante ale lui Textbox

Proprietățile prezentate sînt folosite de programatorii VB pentru furnizarea funcțiilor din **Textbox** dorite de ei.

Text

Stochează valoarea conținutul curent de **Textbox**. Proprietatea implicită a lui **Textbox** este **`Text`**, din acest motiv în loc de forma corectă **`strNume=txtNume.Text`** se poate scrie **`strNume=txtNume`**.

MaxLength

Proprietatea restricționează numărul caracterelor ce pot fi introduse în **Textbox**. Dacă **`MaxLength`** este 0 nu există restricții. Se foloseşte pentru a preveni introducerea unor date mai lungi decît lungimea de cîmp corespunzătoare dintr-o bază de date în care se va înscrie valoarea respectivă.

MultiLine

Implicit, o singură linie de text poate fi introdusă în **Textbox**. La setarea pe **`True`** se validează introducerea de linii multiple.

PasswordChar

Dacă **Textbox**-ul se foloseşte pentru introducerea unei parole atribuind lui **`PasswordChar`** o valoare care va fi afşată pentru fiecare caracter introdus (parola adevărată este mascată). Obligatoriu **`MultiLine`** trebuie să fie **`False`**.

ScrollBars

Bara pentru defilare poate fi orizontală, vericală sau şi în pentru a permite ca în cazul unui

text pe mai multe linii introducerea să se poată face într-un spaţiu limitat. Obligatoriu, **`MultiLine`** trebuie să fie **`True`**.

Locked
Conţinutul unui **Textbox** poate fi blocat la modificări, dar conţinutul poate fi copiat prin Clippboard. **`Locked`** este de preferat în locul lui **`Enable`** pentru păstrarea datelor.

DataChanged
Este o proprietate cu valoarea **`Boolean`** disponibilă numai în timpul execuţiei aplicaţiei care spune dacă valoarea din **Textbox** a fost modificată sau nu. În condiţii normale trebuie restată manual. Se foloseşte pentru a detecta dacă utilizatorul a modificat sau nu conţinutul controlului.

Evenimente Textbox importante

`Change`	**Conţinutul Textbox-ului s-a modificat**
`KeyPressed`	**Utilizatorul a apăsat o tastă, data nu a fost încă acceptată**
`GotFocus`	**Din acest moment Textbox-ul poate primi caractere**
`LostFocus`	**Din acest moment Textbox-ul NU mai poate primi caractere**
`Validate`	**Textbox-ul va pierde focus-ul**

4. Evenimente Textbox mai importante

Evenimentele prezentate sînt folosite de programatorii VB pentru validarea datelor şi în scopul îmbunătăţirii afişării datelor.

Change
Evenimetul **`Change`** apare atunci cînd conţinutul unui **Textbox** se modifică. Modificarea apare deoarece:

- utilizatorul foloseşte tastatura pentru introducerea de taste;
- utilizatorul foloseşte taie (cut) sau inserează (paste) date în control;
- codul atribuie o valoare nouă proprietăţii **`Text`**;
- controlul este legat la datele dintr-o bază de date şi înregistrarea s-a schimbat.

Evenimetul este tipic folsit pentru actualizarea altor controale ale formularului.

KeyPress

Apare atunci cînd utilizatorul apasă o tastă, dar înainte ca tasta să fie acceptată de **Textbox**. Transferat într-un parametru **ByRef** evenimentului este codul ASCII a tastei pentru a putea fi modificată din cod (setarea valorii la 0 descarcă tasta apăsată).

Tipic, evenimentul se foloseşte pentru validarea caracter cu caracter. În exemplu care vine **Textbox**-ul **txtNumar1** poate accepta numai valori numerice.

```
Private Sub txtNumar1_KeyPress(KeyAscii As Integer)
    Debug.Print KeyAscii
    Select Case KeyAscii
        Case vbKey0 To vbKey9, vbKeyBack
        Case Else
            KeyAscii = 0
    End Select
End Sub
```

Constantele de la **vbKey0** la **vbKey9** corespund la codurile ASCII ale tastelor numerice, iar **vbKeyBack** permite ca tasta **BackSpace** să poată şterge.

GotFocus

Evenimentul apare atunci cînd controlul primeşte focus-ul, adică devine capabil să accepte tastele apăsate de la tastatură. Numai un singur focus poate avea focus-ul. Una dintre aplicaţiile evenimentului este cea de colorare (în verde, de exemplu, prin constanta **vbGreen**) a controlului dînd o valoare proprietăţii **BackColor** ca şi în exemplul:

```
Private Sub txtNumar1_GotFocus()
    txtNumar1.BackColor = vbGreen
End Sub
```

LostFocus

Evenimetul apare atunci cînd un control pierde focus-ul din unul dintre următoarele motive:

- utilizatorul, prin apăsarea tastei **Tab**, se poziţionează pe un alt control;
- utilizatorul face clic pe un control capabil să primească focus-ul;
- utilizatorul foloseşte o tastă de acces pentru a trece pe un nou control;
- codul schimbă focus-ul prin apelul metodei **SetFocus** al unui alt control.

Dacă aţi folost **GotFocus** pentru modificarea culorii controlului, atunci **LostFoscus** se poate folosi la readucerea controlului la culoarea originală (**vbWhite** - alb) prin secvenţa:

```
Private Sub txtNumar1_LostFocus()
  txtNumar1.BackColor = vbWhite
End Sub
```

Una dintre aplicaţiile folclorice ale lui **LostFocus** este cea de validare a datelor. În secvenţa de cod care urmează utilizatorul introduce o valoare numerică apoi se mută pe un

alt control. Se va inspecta valoarea introdusă în control, iar dacă aceasta este în afara domeniului [0, 30] se foloseşte **SetFocus** pentru a face utilizatorul să revină asupra valorii greşit introduse.

```
Private Sub txtNumar1_LostFocus()
    If txtNumar1.Text < 0 Or txtNumar1.Text > 30 Then
        MsgBox "Valoarea trebuie sa fie in domeniul [0, 30]", _
        vbExclamation, "Eroare de validare a datelor"
        txtNumar1.SetFocus
    End If
End Sub
```

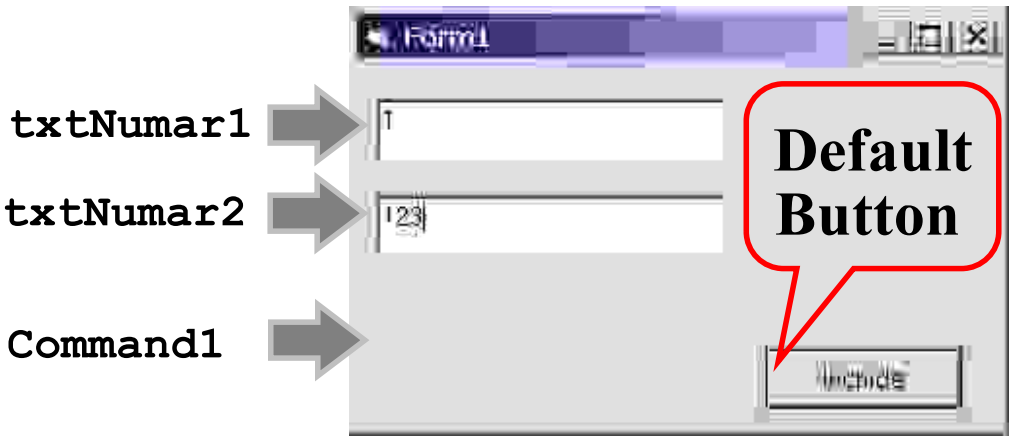
Această soluţie conduce însă la o problemă atunci cînd două controale rulează cod de validare în care modifică focus-ul. Soluţia este fie complexă fie prea puţin elegantă. În VB6 evenimentul **Validate** se foloseşte în acest caz.

Validate

Evenimentul apare atunci cînd focus-ul este pe cale să se transfere unui alt control. Parametrul **Cancel** se foloseşte pentru anularea schimbării focus-ului, utilizatorul fiind blocat în **Textbox** pînă la introducerea unei valori corecte dar fără competiţia pe care o generează evenimentul **LostFocus**.

Evenimentul **Validate** este declanşat numai dacă controlul care primeşte focus-ul are proprietatea **CausesValidation** pe **True**. Aceasta este starea implicită. Dacă doriţi să daţi utilizatorului o şansă de ieşire fără validarea datelor din **Textbox** înainte de a le introduce corect setaţi **CausesValidation** pe **False**.

Evenimentul **Validate** nu este declanşat dacă butonul implicit (Default button) este apelat prin apăsarea tastei **Enter**. Un buton implicit este un buton de comandă a cărui proprietate **Default** este **True**. Evenimentul nu este declanşat deoarece deşi codul evenimentului **Click** este lansat în execuţie focus-ul nu este transferat. Pentru a rezolva această problemă formularul are o metoda **ValidateControl** care ar putea fi apelată din **Click** cu **Me.Validatecontrol**.

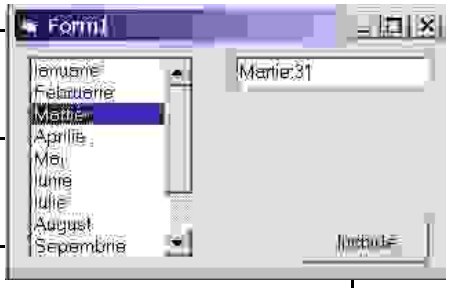


```
Private Sub txtNumar2_Validate(Cancel As Boolean)
    If txtNumar2.Text < 0 Or txtNumar2.Text > 30 Then
        MsgBox "Valoarea trebuie sa fie in domeniul [0, 30]", _
        vbExclamation, "Eroare de validare a datelor"
        Cancel = True
    End If
End Sub
```

O eroare va fi generată dacă se va executa linia **`Cancel=True`**. Tratarea erorii din butonul de comandă va trebui să anuleze operația care se încerca.

```
Private Sub Command1_Click()
    On Error Resume Next
    Me.ValidateControls
    If Err.Number = 0 Then
        Unload Me
    End If
End Sub
```

Proprietățile cheie ale controlului Listbox



`Text`	**Textul corespunzător articolului selectat - Martie**
`ListCount`	**Numărul total al articolelor din listă - 12**
`ListIndex`	**Indicele (index-ul) articolului selectat - 2**
`Sorted`	**Sortarea (True) sau nu (False) a articolelor - False**
`Multiselect`	**Tipul multiselecției - 0**
`List(n)`	**Textul corespunzător celui de a n-elea articol - List(3) = "Aprilie"**
`ItemData(n)`	**Număr ascuns asociat celui de al n-elea articol din listă - 31**
`Selected(n)`	**Este al n-elea articol selectat Selected(2) - True**
`SelCount`	**Numărul de articole selectate - 1**

5. Proprietățile controlului Listbox

Controlul **Listbox** realizează afișarea unei liste de articole ce pot fi selectate. Dacă spațiul alocat listei pe suprafața formularului nu este suficient de mare pentru afișarea tuturor articolelor atunci se va afișa automat o bară pentru defilare (**scrollbar**).

Proprietățile modului de proiectare

Proprietatea **`Sorted`** se foloseşte pentru a asigura, în timpul execuție aplicației, afișarea articolelor în ordine alfabetică crescătoare. Proprietatea nu poate fi schimbată din cod ci doar în timpul proiectării.

`List` poate fi populat, în timpul proiectării, cu articole; **Ctrl+Enter** face trecerea la un articol nou, iar **Enter** indică terminarea adăugării atricolelor în listă.

`Multiselect` determină dacă **Listbox**-ul permite selectarea mai multor articole; dacă da selecția va fi Simple multi-selection (clic pentru selectare/deselectare) sau Extended (clic + folosirea tastelor **Shift/Ctrl**). Proprietatea nu se poate modifica din cod, ea poate fi setată numai în timpul proiectării.

`IntegralHeight` determină dacă **Listbox**-ul se redimensionează singur pe verticală pentru a putea afișa întreaga listă. Valoarea implicită este **`True`**.

`Style` permite afișarea de checkbox-uri pentru fiecare articol de listă. În acest caz este posibil ca o listă cu selecție simplă (nu este **`Multiselect`**) să aibă mai multe articole selectate.

`ItemData` se discută la proprietățile din timpul exexcuției.

Proprietățile din timpul execuției

Proprietatea **`Text`** conține textul articolului curent selectat; dacă nu este selectat nici un articol proprietatea întoarce un şir vid (**`""`**).

Avertisment

Nu se foloseşte proprietatea **`Text`** dacă **Listbox**-ul este **`Multiselect`** deoarece va conține textul corespunzător ultimului articol selectat indiferent dacă el a fost selectat sau deselectat.

`ListCount` conține numărul articolelor totale din Listbox (nu numai a celor afișate).

`ListIndex` conține deplasamentul articolului curent selectat, dacă nu este selectat nici un articol are valoarea -1. Acest index pleacă de la valoarea 0.

Avertisment

`ListIndex` nu se foloeşte dacă Listbox-ul este **`Multiselect`** deoarece va conține index-ul ultimului articole care a fost selectat sau deselectat.

List() este un tablou care poate întoarce textul oricărui articol din listă pe baza index-ului. Şi acest index pleacă de la valoarea 0, de exemplu pentru a extrage textul corespunzător celui de al 3-lea articol se va scrie **lstLuni.List(2)**.

ItemData() este un tablou de valori de tipul **Long** invizibile asociate fiecărui element din listă. De exemplu, se pot stoca numărul zilelor dintr-o lună aici. **ItemData()** poate fi populat în timpul proiectării, însă în timpul execuţiei aceasta se poate face doar imediat după setarea valorii corespunzătoare afişate în **Listbox**. Un exemplu va fi prezentat imediat.

Selected() este un tablou folosit în cazul **Listbox**-urilor cu **Multiselect**. Pentru fiecare articol din listă este o intrare corespunzătoare în **Selected**; ea va fi **True** dacă articolul este selectat sau **False** dacă nu este. Un exemplu va fi prezentat imediat.

SelCount conţine numărul articolelor curent selectate şi este, în particular, utilă pentru prelucrarea listelor lungi. Dacă au fot găsite deja **SelCount** articole selectate prelucrarea poate fi oprită.

Controlul Listbox - Metode şi Evenimente

Clear	**Goleşte conţinutul Listbox-ului**
AddItem	**Adaugă un articol în Listbox**

```
Dim luna(1 To 12) As String
Dim i As Integer
luna(1) = "Ianuarie"
 ...
luna(12) = "Decembrie"
lstLuna.Clear
For i = 1 To 12
   lstLuna.AddItem luna(i)
Next
```

RemoveItem	**Şterge articolul specificat prin index din Listbox**
Click	**Apare la clic pe Listă şi la deplasarea prin ea cu mouse-ul sau tastatura**
DblClick	**Apare numai cînd se face clic dublu pe un articol**

6. Controlul Listbox - metode şi evenimente

Listbox-urile pot fi populate în timpul proiectării, dar este mult mai obişnuită popularea lor în timpul execuţiei codului cînd, tipic, se foloseşte evenimentul **Load** al formularului din care fac parte sau după rularea unei interogări. Pentru aceasta trebuie înţelese metodele ce permit manipularea conţinutului unui **Listbox**.

`Clear` permite ştergerea întregului conţinut al **Listbox**-ului. Este bine ca înainte de adăugarea articolelor în **Listbox** acesta să fie şters pentru ca să fie sigur că el va conţine doar articolele dorite.

`AddItem` se foloseşte pentru adăugare de articole în **Listbox**. Primul argument este şirul ce va fi afişat şi, opţional, al doilea argument poate specifica poziţia în tabloul **`List`**. Implicit, adăugarea se face la sfîrşitul listei, dacă, de exemplu, se doreşte adăugarea articolului **`(neselectat)`** la capul liste atunci se va scrie:

```
lstLuna.AddItem "(neselectat)", 0
```

`RemoveItem` se foloseşte pentru şetergerea unui articol din listă. Necesită un parametru numeric care este index-ul elementului de şters. Prin cele două exemple care urmează se şterg primul şi utimul element ale listei:

```
lstLuna.RemoveItem 0
lstLuna.RemoveItem lstLuna.ListCount-1
```

Evenimente Listbox

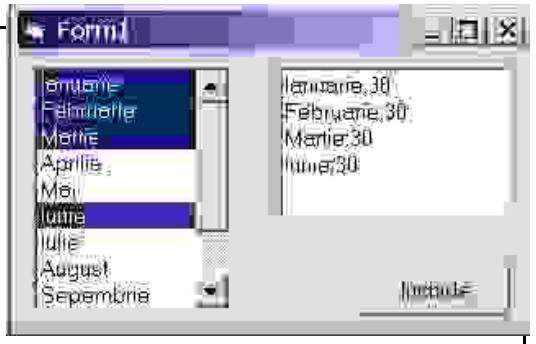
Evenimentele cel mai des folosite cu **Listbox**-urile sînt **`Click`** şi **`DblClick`**.

Evenimentul **`Click`** apare atunci cînd utilizatorul face clic cu butonul de mouse pe **Listbox** şi atunci cînd se foloseşte tastatura pentru deplasarea prin listă, astfel **`Click`** apare pentru orice selecţie realizată. Acesta este motivul pentru care evenimentul se foloseşte pentru suportul realizării deciziilor prin detalierea opţiunii variantei selectate.

Evenimentul clic dublu pe **Listbox** este un semnal mult mai pozitiv, în comparaţie cu un clic simplu, că utilizatorul s-a decis să aleagă respectiva variantă. **`DblClick`** se poate folosi pentru a trimite un e-mail promoţional unui client, a adăuga un rău platnic unei alte liste etc.

Exemple de cod cu Listbox

Folosirea lui ItemData




```
Private Sub Form_Load()
    Dim luna(1 To 12) As String
    Dim i As Integer
    luna(1) = "Ianuarie"
    ...
    luna(12) = "Decembrie"
    lstLuna.Clear
    For i = 1 To 12
        lstLuna.AddItem luna(i)
        Select Case i
            Case 1, 3, 5, 7, 8, 10, 12
                lstLuna.ItemData(lstLuna.NewIndex) = 31
            Case 4, 6, 9, 11
                lstLuna.ItemData(lstLuna.NewIndex) = 30
            Case 2
                If Year(Now) Mod 4 = 0 Then
                    lstLuna.ItemData(lstLuna.NewIndex) = 27
                Else
                    lstLuna.ItemData(lstLuna.NewIndex) = 28
                End If
        End Select
    Next
End Sub
```

Folosirea unui Listbox `Multiselect`

```
Private Sub lstLuna_Click()
    Dim i As Integer
    If lstLuna.SelCount <= 0 Then Exit Sub
    txtLunaSel.Text = ""
    For i = 0 To lstLuna.ListCount - 1
        If lstLuna.Selected(i) Then
            txtLunaSel.Text = txtLunaSel.Text + lstLuna.List(i) +
";" + CStr(lstLuna.ItemData(lstLuna.ListIndex)) + vbCrLf
        End If
    Next
End Sub
```

7. Exemple cu Listbox

Folosirea proprietății ItemData

După cum am spus deja fiecărui articol din listă i se asociază şi o proprietate **`ItemData`** care poate stoca o valoare întreagă de tipul **`Long`**. Valoarea nu se afişează, dar poate fi folosită în cod. Deşi **`List`** şi **`ItemData`** ar putea fi completate în timpul proiectării este mult mai probabil ca valorile să fie generate din cod sau extrase din fişier sau dintr-o bază de date.

În codul anterior un articol (numele lunii) se adaugă în **Listbox**. Nu putem fi siguri că adăugarea se face la coada listei deoarece proprietatea `Sorted` ar putea fi **True**. **Listbox**-ul are o proprietate `NewIndex` care va stoca index-ul ultimului articol adăugat. Astfel, `NewIndex` se poate folosi pentru a indentifica care `ItemData` corespunde unltimei luni adăugate

Lucrul cu Listbox-uri Multiselect

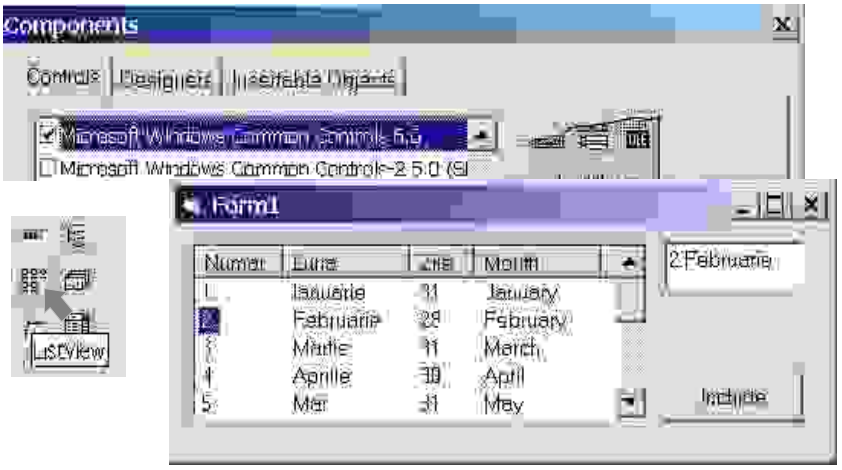
Odată ce **Listbox**-ul are activată selecția multiplă nu se mai pot folosi proproetățile `Text` şi `ListIndex` deoarece ambele sînt legate de ultimul articol pe care s-a făcut clic (care ar putea fi o deselectare).

Codul anterior prezintă cum se poate parcurge întregul conținut al **Listbox**-ului testînd proprietarea de tip tablou `Selected` pentru fiecare element individual. Nu există altă proprietate tablou care să permită identificarea din listă a articolelor selectate.

Codul testează prima oară proprietatea `SelCount` pentru a vedea dacă este ceva selectat. Ar mai putea fi adăugat un contor în ciclu pentru a stoca numărul articolelor selectate, dacă acesta devine egal cu `SelCount` ciclul ar putea fi terminat.

Controlul Listview- caracteristici de bază

- **Parte din Windows Common Controls**
- **proprietatea `View` trebuie setată !**
- **colecția `ColumnHeaders`**
- **colecția `ListItems` de obiecte `ListItem`**
- **`ListItem` are colecția `ListSubItems` de obiecte `ListSubItem`**
- **poate sorta după coloană**



8. Controlul Listview

Controlul **Listview** este util în aplicații datorită posibilității lui de afișare a datelor sub formă tabelară. Programarea controlului este ceva mai complicată decît cea a **Listbox**-ului, iar cele ce urmează vin să prezinte etapele de bază de urmat la folosire acestui control.

Listview nu apare în **Toolbox** deoarece nu este un control standard; el trebuie selectat din fereastra *Components* (prin apăsarea lui **Ctrl+T**) prin **Microsoft Windows Common Controls 6.0**; în urma selecției o mulțime de controale vor fi adăugate în **Toolbox**, **Listview** are simbolul grafic următor

Listview are 4 moduri de vizualizare a articolelor, cel care a fost prezentat este, în general, cel mai folosit fiind obținut dacă proprietății **`View`** i se dă valoarea **`lvwReport`**. Acesta este singurul caz în care datele articolelor sînt vizualizate într-o formă tabelară, iar proprietatea pooate fi setată în timpul proiectării sau în timpul execuței aplicației

Codul corespunzător descrierii ce urmează va fi prezentat imediat. Pentru a crea coloanele controlului împreună cu numele acestora se folosește colecția **`ColumnHeaders`**. Acțiunea se poate face în timpul proiectării sau, mai flexibil, în timpul execuției. Pentru a pune date în control se parcurg două etape pentru fiecare rînd:

- se creează un obiect **`ListItem`** reprezentînd rîndul, apoi se populează cu data primei coloane. Aceasta se face cu metoda **`Add`** a colecției **`ListItems`**;
- coloanele rămase se populează cu metoda **`Add`** a colecției **`ListSubItems`** a lui **`ListItem`**..

În final, controlul poate fi sortat după orice coloană prin folosirea combinată a următoarelor proprietăți: **`SortKey`** (coloana după care se face sortarea), **`SortOrder`** și **`Sorted`**, de obicei apelată ca rezultat al evenimentului **`ColumnClick`**.

```
Dim rind As MSComctlLib.ListItem
Dim luna(1 To 12) As String, lunae(1 To 12) As String
Dim i As Integer, nrZile As Integer

luna(1) = "Ianuarie"
...
luna(12) = "Decembrie"
lunae(1) = "January"
...
lunae(12) = "December"

With lvLuna
        ColumnHeaders.Clear
        ColumnHeaders.Add Text:="Numar", Width:="700"
        ColumnHeaders.Add Text:="Luna", Width:="1000"
        ColumnHeaders.Add Text:="Zile", Width:="500"
```

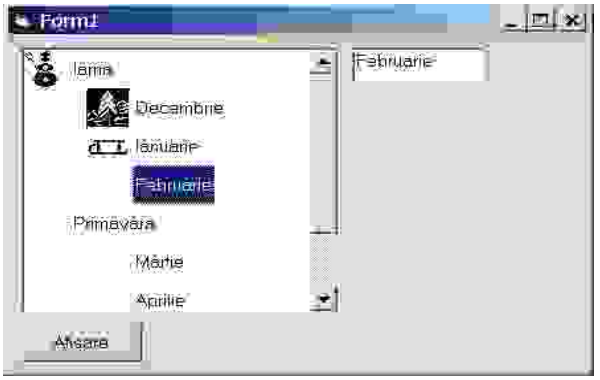
```
        ColumnHeaders.Add Text:="Month", Width:="1000"
        View = lvwReport
        ListItems.Clear
        For i = 1 To 12
            Set rind = lvLuna.ListItems.Add(Text:=i)
            rind.ListSubItems.Add Text:=luna(i)
            rind.ListSubItems.Add Text:=ZileLuna(i)
            rind.ListSubItems.Add Text:=lunae(i)
        Next
    End With
```

Codul prezentat este puțin scurtat pentru a evidenția etapele de bază:

- o variabilă obiect, mai sus **rind**, este declarată de un tip corespunzător pentru manipularea unui obiect **`ListItem`**. Pentru fiecare rînd va exista un astfel de obiect în control;
- colecția **`ColumnHeaders`** este golită de orice nume de coloane pe care le-ar putea conține;
- pentru fiecare coloană dorită se folosește metoda **`Add`** din colecția **`ColumnHeaders`**, transferînd cel puțin argumentul **`Text`** (deseori și argumentul **`Width`**);
- colecția **`ListItems`** este golită de orice date (rînduri) pe care le-ar putea conține;
- pentru fiecare rînd de date se creează un obiect **`ListItem`** prin folosirea metodei **`Add`** a colecției **`ListItems`**. **`Add`** întoarce un pointer la articolul de listă (**`ListItem`**) adăugat, iar **rind** se folosește pentru a stoca această referință. Din nou parametrul **`Text`** se folosește pentru definirea conținutului.
- **rind** se folosește pentru a crea și popula coloanele următoare prin folosirea metodei **`Add`** a colecției **`ListSubItems`**.

Controlul Treeview

- **Parte a lui Windows Common Controls**
- **colecția Nodes conține noduri între care există relații de legătură părinte-copil**



9. Controlul Treeview

Controlul **Treeview** este folosit pentru afişarea unor date între care există legături de tipul ierarhic. Nici el nu apare în **Toolbox** şi la fel ca şi la **Listview**, din fereastra de dialog *Components*, se va selecta Microsoft Windows Common Controls 6.0. Spre deosebire de controlul **Listview**, aici fiecare intrare este de acelaşi tip şi anume **`Node`**. Fiecare **`Node`** are proprietatea **`Text`**, iar dacă este cazul i se mai poate asocia şi o imagine. Deosebită în acest caz este stabilirea relaţiilor între nodurile controlului. Ceva mai tîrziu va fi prezentat un cod detaliat cu privire la această problemă.

Fiecare nod este creat folosind metoda **`Add`** din colecţia **`Nodes`** a controlului. Mai pot fi specificate prin proprietatea **`Text`** un text care va fi afişat, de asemenea este bine ca să se atribuie proprietăţii **`Key`** un şir care să identifice unic nodul (asta pentru a putea extrage din colecţie nodul). Metoda **`Add`** mai permite specificarea relaţiei nodului cu alte noduri ale controlului.

Fiecărui nod i se poate asocia o imagine dintr-un control **Imagelist** (asta merge şi în cazul unui control **Listview**). Controlul are o proprietate **`Imagelist`** ce identifică controlul **Imagelist** din formular. Imaginea pentru un **`Nod`** (sau **`ListItem`**) este identificată printr-un index sau printr-o cheie.

Exemplu de cod cu Treeview:

```
Dim N As MSComctlLib.Node, N2 As MSComctlLib.Node
tvwLuni.ImageList = ImgList
tvwLuni.Nodes.Clear
...
Set N = tvwLuni.Nodes.Add(Text:="Iarna", Image:="iarna")
    Set N2 = tvwLuni.Nodes.Add(Relative:=N, Image:="dec", _
        Relationship:=tvwChild, Text:="Decembrie")
    N2.EnsureVisible

    Set N2 = tvwLuni.Nodes.Add(Relative:=N, Image:="ian", _
        Relationship:=tvwChild, Text:="Ianuarie")
    N2.EnsureVisible

    Set N2 = tvwLuni.Nodes.Add(Relative:=N, _
        Relationship:=tvwChild, Text:="Februarie")
    N2.EnsureVisible
...
```

Exemplul de cod prezentat este abreviat cu scopul evidenţierii etapelor de bază în lucrul cu obiectul **Treeview**:

- o variabilă obiect, mai sus **`N`**, este declarată de tipul corespunzător manipulării unui obiect **Node**, deoarece fiecare articol din **Treeview** este un nod de acest fel;
- un articol este adăugat la rădăcina lui **Treeview** prin metoda **`Add`** a colecţiei **`Nodes`**, fără a specifica o relaţie cu alte noduri. Metoda **`Add`** întoarce o referinţă la nodul

adăugat care se va stoca într-o variabilă nod;

- un articol aflat în relație cu nodul anterior este adăugat folosind metoda **Add** a colecției **Nodes**, cu specificarea nodului la care este relativ și a tipului de relație. Nodul la care este relativ se poate specifica prin variabila nod a părintelui (așa cum se vede în exemplu) sau prin folosirea valorii date lui **Key** atunci cînd a fost adăugat nodul părinte (nu apare în exemplu);
- asigurarea vizibilității unui nod adăugat se face prin metoda **EnsureVisible**.

Codul complet este prezentat în continuare:

```
Private Sub Command1_Click()
    Dim N As MSComctlLib.Node, N2 As MSComctlLib.Node
    tvwLuni.ImageList = ImgList
    'tvwLuni.LineStyle = tvwRootLines
    tvwLuni.Nodes.Clear

    Set N = tvwLuni.Nodes.Add(Text:="Iarna", Image:="iarna")
        Set N2 = tvwLuni.Nodes.Add(Relative:=N, Image:="dec", _
            Relationship:=tvwChild, Text:="Decembrie")
        N2.EnsureVisible

        Set N2 = tvwLuni.Nodes.Add(Relative:=N, Image:="ian", _
            Relationship:=tvwChild, Text:="Ianuarie")
        N2.EnsureVisible

        Set N2 = tvwLuni.Nodes.Add(Relative:=N, _
            Relationship:=tvwChild, Text:="Februarie")
        N2.EnsureVisible

    Set N = tvwLuni.Nodes.Add(Text:="Primavara")
        Set N2 = tvwLuni.Nodes.Add(Relative:=N, _
            Relationship:=tvwChild, Text:="Martie")
        N2.EnsureVisible

        Set N2 = tvwLuni.Nodes.Add(Relative:=N, _
            Relationship:=tvwChild, Text:="Aprilie")
        N2.EnsureVisible

        Set N2 = tvwLuni.Nodes.Add(Relative:=N, _
            Relationship:=tvwChild, Text:="Mai")
        N2.EnsureVisible
   tvwLuni.Nodes(1).Selected = True
End Sub

Private Sub Form_Load()
    Dim img As ListImage
       Set img = ImgList.ListImages.Add(, "iarna", LoadPicture("c:\Documents and
Settings\Antal Tiberiu\My Documents\My Pictures\icon1.ico"))
        Set img = ImgList.ListImages.Add(, "dec", LoadPicture("c:\Documents and
Settings\Antal Tiberiu\My Documents\My Pictures\icon2.ico"))
        Set img = ImgList.ListImages.Add(, "ian", LoadPicture("c:\Documents and
Settings\Antal Tiberiu\My Documents\My Pictures\icon3.ico"))
End Sub

Private Sub tvwLuni_Click()
    txtLuna.Text = tvwLuni.SelectedItem
End Sub
```