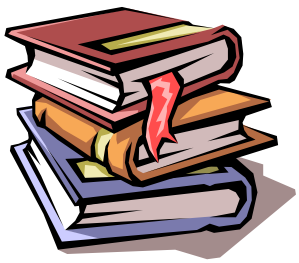
Thiết kế chương trình duyệt file âm thanh bằng Visual Basic



Thiết kế chương trình duyệt file âm thanh

bằng Visual Basic

(Sử dụng MediaPlayer 6.x của Windows)

MediaPlayer của Windows từ version 6.x trở đi có thể player được rất nhiều dạng thức tập tin Multimedia khác nhau như: .avi, .asf, .asx, .rmi, .wav ; .ra, .ram, .rm, .rmm ; .mpg, .mpeg, .m1v, .mp2, .mpa, .mpe ; .mid, .rmi ; .qt, .aif, .aifc, .aiff, .mov ; .au, .snd ... Chất lượng cũng được cải thiện rất rõ rệt so với các phiên bản trước.

Nếu bạn đang sử dụng Windows 98 thì MediaPlayer đã sẵn sàng, nếu dùng Windows 95, 97 bạn buộc phải cài đặt bổ sung để lên đời MediaPlayer của mình. Bạn có thể tìm bộ nâng cấp trên các CDROM phần mềm hay nằm chung trong bộ Internet Explorer 4.01 SP2.

Các file multimedia hiện này tràn ngập trên Internet, CDROM, rất nhiều. Đặc biệt là MP3 & Midi, 2 loại file này rất thịnh hành và đang được ưa chuộng.

Cái gì nhiều cũng gây nên ý tưởng (nói dúng hơn là sinh tật). Mặc dù chỉ cần double click lên file Mp3 hay Midi trong một trình quản lý file là có thể Play được một cách dễ dàng nhờ MediaPlayer của Windows nhưng cái gì của riêng mình mới khoái.

Chính vì vậy trong bài viết này tôi xin mạn phép hướng dẫn các bạn tự thiết kế một MediaPlayer rất tiện dụng và để dành làm của riêng. Tuy nhiên nói của riêng không phải là tự làm hết mà chúng ta phải dùng một bản sao của MediaPlayer trong chương trình.

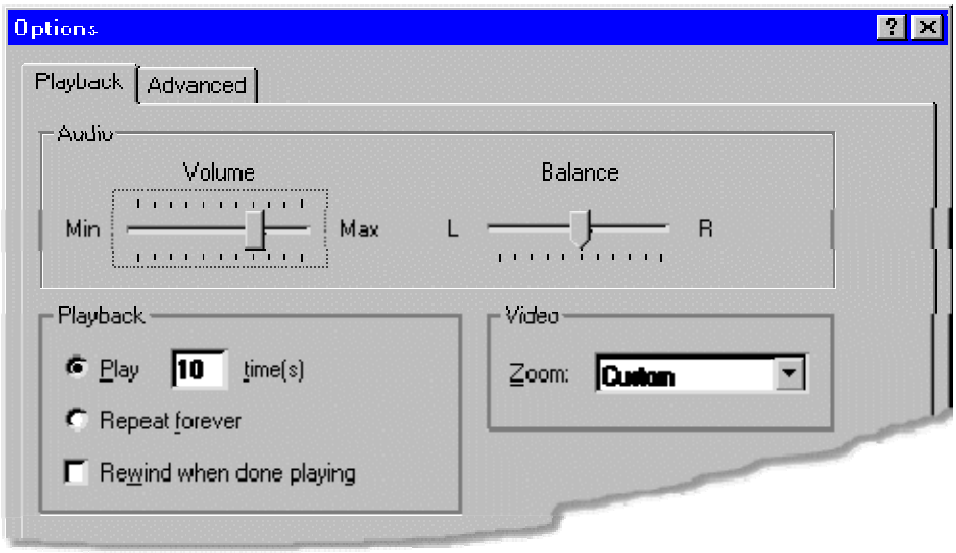
Khái quát về chương trình

Chúng ta sẽ thiết kế chương trình có giao diện như sau:

Đầu tiên người dùngười chọn ổ đĩa, thư mục có chứa các file Multimedia (thí dụ là file Midi). Kế đến nhấn nút Play hoặc double click trên tên file cần phát để nghe nhạc.

Ngoài ra còn có các nút Help, Author, Exit

Phía dưới là một MediaPlayer được nhúng vào chương trình, có thể điều chỉnh các chức năng như một chương trình riêng biệt (bạn có thể right click để mở menu tắt quen thuộc như khi dùng MediaPlayer), ở cuối của cửa sổ có dòng thông báo tên file & đường dẫn đang Play.



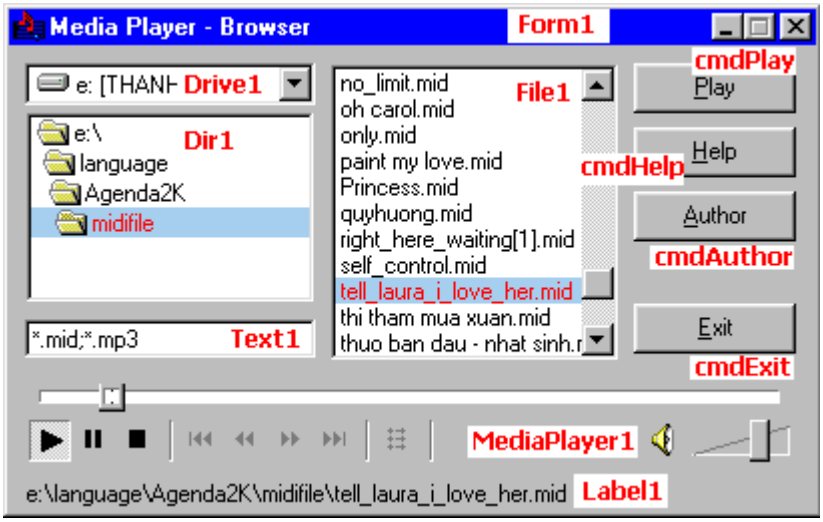
Các xác lập trong hộp thoại Options của MediaPlayer

Phía dưới của hộp chọn thư mục có một Text box dùng để lọc file. Các loại file này ngăn cách bởi dấu chấm phảy ";". Thí du như bạn muốn lọc các file MP3 & MIDI thì gõ vào: *.mp3;*.mid

Cũng lưu ý thêm là: nếu như trong hộp liệt kê tên file không có file nào, thì nút Play bị vô hiệu hoá (Enabled=False). Chỉ khi nào có file nút Play mới có tác dụng.

Thiết kế giao diện

Bạn hãy khởi động Visual Basic và bắt tay vào việc tạo dáng cho ứng dụng của mình. Cách bố trí các Control trên form tùy theo ý mỗi người, riêng tôi, tôi trình bày như sau:



Các thuộc tính & Caption của các Control trong chương trình:

FORM

Form1.caption = "MediaPlayer - Browser"

Form1.BorderStyle = 1-Fixed Single

Form1.Minbutton=True

TEXTBOX/LABELBOX

Text1.text="*.mid;*.mp3"

Label1.caption=""

COMMAND BUTTON

cmdPlay.caption="&Play"

cmdPlay.enabled=False

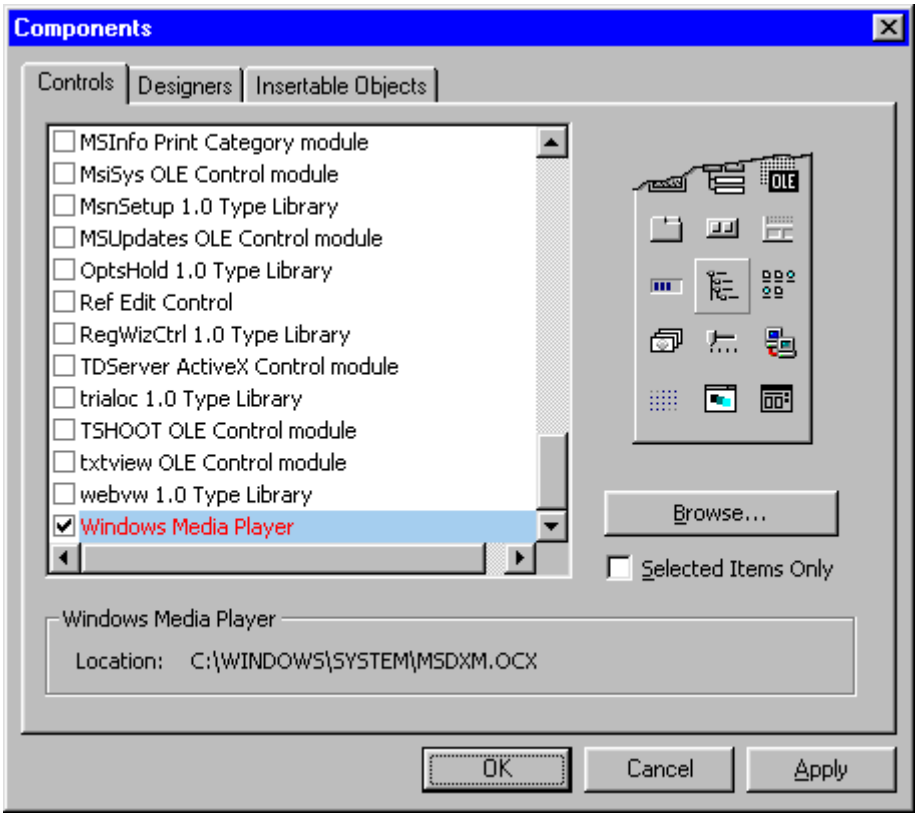
cmdHelp.caption="&Help"

cmdAuthor.caption="&Author"

cmdExit.caption="&Exit"

Trên thanh Toolbox của Visual Basic không có đối tượng MediaPlayer. Bạn phải dùng một Custom Control để thêm đối tượng đó vào.

Nhấn CTRL - T. Trong hộp thoại Components chọn Windows MediaPlayer (thường ở cuối danh sách), Click nút OK



Đối tượng MediaPlayer sẽ được thêm vào Toolbox, việc còn lại, chỉ cần vẽ nó lên form, đặt ở vị trí thích hợp (nó có tên mặc nhiên là MediaPlayer1)

Viết Code

Đầu tiên bạn cần cho bộ 3 control: Drive1, Dir1, File1 hoạt động. Hãy gõ đoạn Code sau đây để cho chúng "hiểu nhau"

```
Private Sub Dir1_Change()

   File1.Path = Dir1.Path

   If File1.ListCount = 0 Then
   'Kiểm tra xem có file nào trong listbox File1 chưa

      cmdPlay.Enabled = False
      'Nếu chưa có thì vô hiệu nút Play

   Else

      cmdPlay.Enabled = True
      'Nếu có rồi thì cho hiệu lực nút Play

   End If

End Sub
```

```
Private Sub Drive1_Change()

   Dir1.Path = Drive1.Drive

End Sub
```

Double click lên nút Play và viết

```
Private Sub Command1_Click()

   MediaPlayer1.filename = Dir1.Path & "\" _
      & File1.List(File1.ListIndex)

   Label1.Caption = MediaPlayer1.filename

End Sub
```

Nếu thuộc tính AutoStart của MediaPlayer được gán bằng True. MediaPlayer sẽ tự động Play nếu bạn truyền cho thuộc tính FileName của nó một chuỗi là đường dẫn đến file cần Play. Khi thuộc tính FileName là rỗng, nó sẽ ngừng.

ở đoạn Code trên tôi đã ghép nối các thuộc tính của Drive1, Dir1 & File1 để chỉ ra file cần Play. Đoạn code sẽ gặp lỗi khi các file cần Play nằm ngoài thư mục gốc, bạn hãy tự hoàn chỉnh lấy bằng hàm IIF() hay câu lệnh IF

Dòng thứ 2 dùng để hiển thị đường dẫn file đang Play trong Labelbox ở cuối form.

Nếu muốn khi người dùng Double Click lên tên file trong danh sách file thì MediaPlayer sẽ Play file đó, bạn chỉ cần làm như sau:

```
Private Sub File1_DblClick()

   cmdPlay_Click

End Sub
```

Để khả năng lọc (Pattern) của File1 hoạt động theo nội dung trong Textbox (Text1). Bạn cần gán các chuỗi trong Textbox do người dùng gõ vào mỗi khi có sự thay đổi (thuộc tính Change của Textbox).

```
Private Sub Text1_Change()

   File1.Pattern = Trim(Text1)

End Sub
```

Đồng thời lúc chương trình khởi động bạn cũng phải gán nội dung trong Textbox cho thuộc tính Pattern của File1

```
Private Sub Form_Load()
```

Text1_Change

End Sub

MediaPlayer còn có một thuộc tính tên là PlayCount - Số lần phát lại một file nhạc, bạn hãy gán cho nó một số thích hợp trong khi thiết kế chương trình.

Khả năng của MediaPlayer còn tùy thuộc vào MediaPlayer đang sử dụng trong Windows của bạn.

Vậy là xong, một chương trình duyệt file âm thanh, thật là quá đơn giản phải không bạn :-)

Thay lời kết

Bây giờ bạn có thể dịch ra file exe, đem tặng cho bạn bè "làm kỷ niệm". Nhớ chép thêm các file cần thiết cho chương trình nhé. MSDXM.OCX là file chứa Custom Control MediaPlayer đã sử dụng trong chương trình. Hãy nén lại cho chúng thật mi nhon trước khi chép ra đĩa mềm hay gởi kèm theo E-mail.

Trên đây chỉ là một chương trình rất đơn giản, nhưng tính năng có nó thì đáng khâm phục phải không bạn. Còn lại vài chi tiết khác bạn có thể tự mình làm lấy theo ý thích. Bạn có thể thêm vài tính năng nữa cho chương trình trở nên đa dụng, thí dụ như: Play các file Video, tự động Play một loạt các file...

Chúc bạn thành công.

Viết ứng dụng
INDEXER

[Thiết kế giao diện] [Viết Code]

Viết chương trình tạo trang Web chứa các Link đến các tập tin trong một thư mục được người dùng chỉ định.

Chương trình này có các chức năng và hoạt động tổng quát như sau:

Chọn thư mục

Lọc file

Cho người dùng chọn file

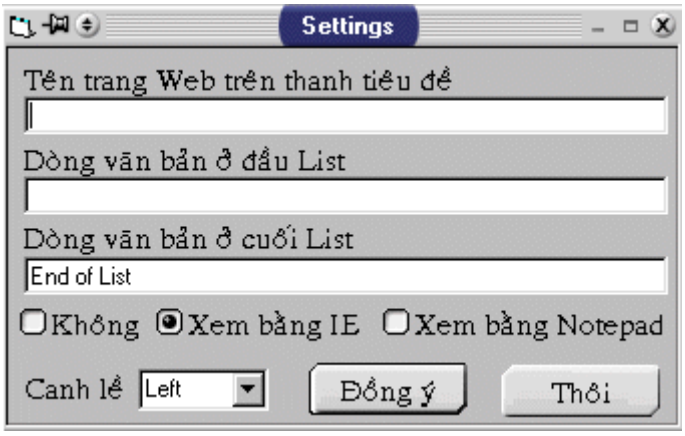
Đặt tên tiêu đề cho trang Web

Đặt dòng văn bản ở đầu danh sách

Đặt dòng văn bản ở cuối danh sách

Sau khi tạo xong cho phép xem bằng IE hay Notepad

Chọn canh lề: Trái, phải , giữa.



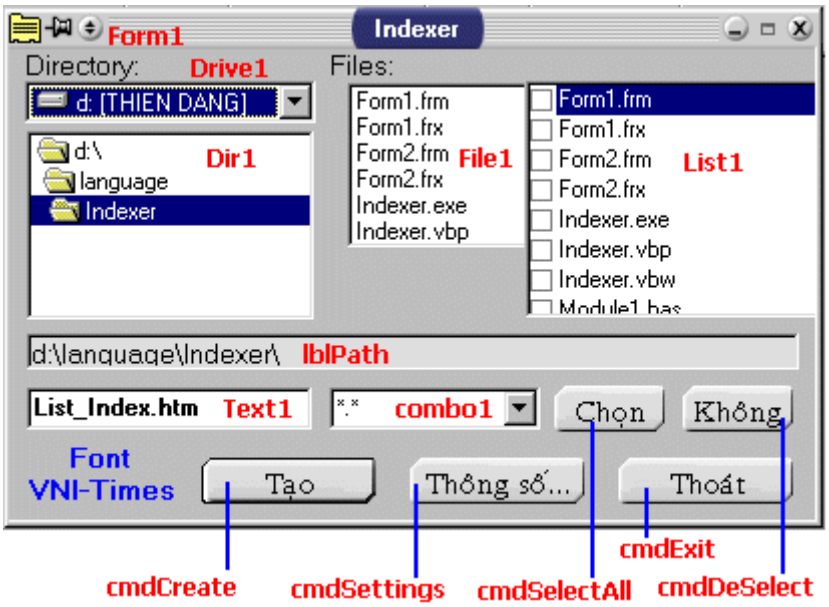
Khi bạn nhấn nút "Tạo" trong Form chính (Form1) chương trình sẽ tạo một trang Web chứa các link đến các file trong thư mục, trang Web này được lưu vào cùng thư mục mà bạn chỉ định.

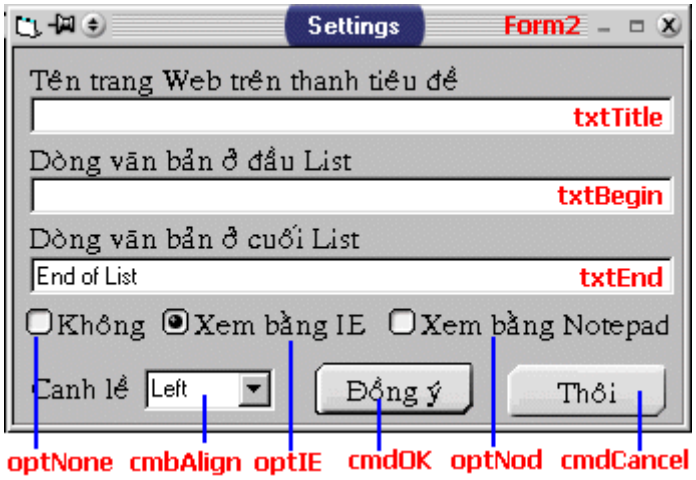
Mỗi lần người dùng chỉ định thư mục, chương trình sẽ tự động điền đường dẫn và tên file (mặc nhiên là List_index.htm) vào hộp chọn file name (Text1)

Để dễ dàng trong việc chọn lựa ta dùng thêm một ListBox (List1) thế cho FileListBox (File1). Bạn nên cho ListBox nằm đè lên đối tượng File1 (hoặc cho File1.Visible=False) vì ta chỉ cần File1 để lấy tên các tập tin Add vào List1 chớ không dùng đến.

Một ComboBox (Combo1) để lọc file theo từng loại file hoặc tất cả (do người dùng tự chọn hay gõ vào).

Đồng thời cung cấp thêm các nút lệnh: "Chọn" chọn tất cả các tập tin trong Listbox, "Không" bỏ chọn tất cả các tập tin trong Listbox (bạn cũng có thể chọn bằng cách Check vào từng tên file tương ứng), "Tạo" nhấn nút này để bắt đầu tạo trang Web, "Thông số" nhấn nút này để xác lập thêm các tùy chọn cho trang Web, "Thoát" Thoát khỏi chương trình.

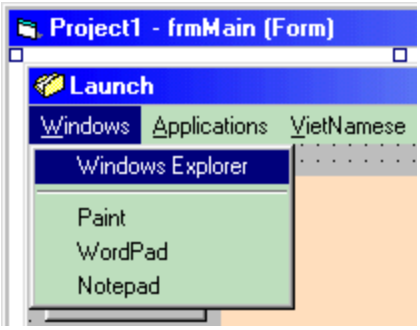


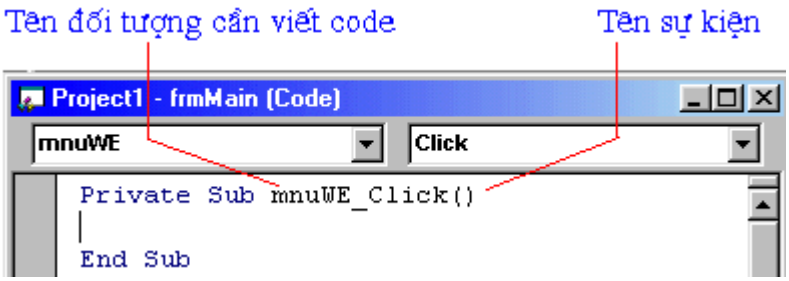


Viết Code cho menu

Ta chỉ cần viết code cho menu, sau đó dùng các nút lệnh để gọi menu tương ứng.

Bây giờ chúng ta viết code cho mục Windows Explorer trong menu Windows. Vào Windows chọn Windows Explorer để viết code cho mục chọn menu này.





Bạn gõ vào đoạn sau:

```
Private Sub mnuWE_Click() ' dòng này có sẵn

        Dim P
        P = Shell("explorer", vbNormalFocus)

End Sub ' dòng này có sẵn
```

Giải thích:

* Dim P

Khai báo 1 biến kiểu variant để chứa trị trả về của hàm Shell. Đây là kiểu dữ liệu bao trùm tất cả các kiểu dữ liệu trong Visual Basic.

* P=Shell("explorer",vbNormalFocus)

Hàm Shell dùng để gọi một chương trình khác thi hành

Cú pháp Shell(pathname[,windowstyle])

pathname: là đường dẫn và file thực thi của chương trình cần gọi. Đây là 1 xâu cho nên khi viết bạn phải đặt chúng trong cặp dấu " " mới đúng.

windowstyle: là hằng số qui định phong cách khi khởi động của chương trình cần chạy. Thí dụ: sau khi gọi chương trình bạn cần Maximize, Minimize chương trình đó ... các hằng có giá trị và ý nghĩa như sau:

Tên hằng	Giá trị	ý nghĩa

vbHide	0	Window is hidden and focus is passed to the hidden window.
vbNormalFocus	1	Window has focus and is restored to its original size and position.
vbMinimizedFocus	2	Window is displayed as an icon with focus.
vbMaximizedFocus	3	Window is maximized with focus.
vbNormalNoFocus	4	Window is restored to its most recent size and position. The currently active window remains active.
vbMinimizedNoFocus	5	Window is displayed as an icon. The currently active window remains active.

Vậy có thể viết lại hàm Shell như sau Shell("explorer",1) cho gọn

Lưu ý: Trong phần pathname của hàm shell lý ra phải ghi đầy đủ đường dẫn, thí dụ "C:\Windows\Explorer.exe" (giả sử thư mục windows là c:\windows) thay vì "explorer.exe". Sở dĩ ta có thể ghi gọn như vậy là vì Windows tự động đặt dường dẫn path đến các thư mục như: Windows; Windows\system. Do đó chỉ cần ghi explorer.exe cho tổng quát (khỏi sợ sai đường dẫn khi đem chạy trên máy khác).

Bây giờ nhấn F5 để chạy chương trình, vào menu Windows chọn Windows Explorer, lập tức chương trình Windows Explorer được khởi động.

Tương tự như vậy bạn có viết code cho tất cả các menu con còn lại của menu Windows.

Notepad.exe (Windows/Notepad)

Write.exe (Windows/WordPad)

Pbrush.exe (Windows/Paint)

Đối với Paint và WordPad ta phải dùng 2 file write.exe & pbrush.exe trong thư mục Windows để khởi động. Thực ra 2 file này chỉ có chức năng gọi WordPad.exe và MSPaint.exe (trong thư mục \Program Files\Accessories\) chứ không phải là file chương trình chính. Microsoft phải làm vậy để tương thích với các chương trình cũ.

Còn các mục chọn khác bạn cũng viết hàm Shell tương tự nhưng đường dẫn phải cụ thể và chính xác. Thí dụ để viết code cho menu "Lac Viet td". Vào VietNamese / Lac Viet td, gõ vào

```
Private Sub mnuLV_Click()

        Dim F
        F=Shell("d:\tools\lvtd\lvtd.exe",1)

End Sub
```

Do file lvtd.exe của máy tôi nằm trong thư mục d:\tools\lvtd

Nhấn F5 chạy thử xem có vừa ý hay không ?

Viết code cho các Command Button

Bây giờ ta viết lệnh cho các CommandButton tương ứng. Yêu cầu là viết code sao cho khi nhấn vào nút Windows thì menu Windows tương ứng sẽ hiện ra như hình minh họa

Vậy phải viết lệnh cho nút

+ Windows (cmdWin) gọi menu Windows (mnuWin)
+ Application (cmdApp) ---> mnuApp
+ VietNamese (cmdVN) ---> mnuVN

Double click vào cmdWin (hay Right click chọn View code từ menu popup), gõ vào

```
Private Sub cmdWin_Click()

        popupmenu mnuWin

End Sub
```

Giải thích:

popupmenu mnuWin hành vi (method) popupmenu dùng để hiển thị menu có tên mnuWin

Xem cú pháp popupmenu

Tương tự cho 2 nút lệnh còn lại. Khi chạy thử chương trình bạn click vào nút lệnh nào sẽ xuất hiện menu tương ứng. Từ đây người dùng có thể chọn lệnh từ menu popup hay menu pulldown (menu kéo xuống) đều được.

Viết lệnh cho nút Exit như sau:

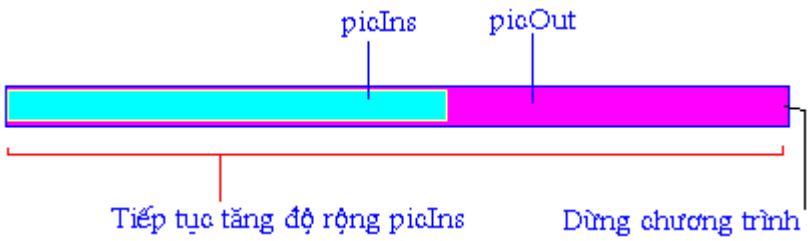
```
Private Sub cmdExit_Click()
   End
End Sub
```

Làm cho chương trình tự động thoát

Đối tượng Timer

Nếu đang ở chế động tự động thoát (mục Unload after 20 Sec được chọn) sau 20 giây chương trình sẽ tự động thoát, không cần chúng ta can thiệp. Để làm được việc này ta phải dùng Timer và Picture box (picIns, picOut) đã tạo từ trước.

Sau khi chương trình khởi động hoặc khi check vào checkbox. Mỗi giây độ rộng hiện tại của picIns cộng với độ rộng của picOut/20 (vì 20 giây), cho đến khi độ rông của picIns = picOut thì dừng chương trình. Nếu không check chức năng tự động thoát không hoạt động.



Chúng ta tiến hành viết code cho các đối tượng như sau

Tình huống Form_Load() sẽ được kích hoạt khi chương trình khởi động, timer hoạt động với trị interval = 1000 (tương đương 1 giây), độ rộng picIns ban đầu là 0.

```
Private Sub Form_Load()
   Timer1.Interval = 1000
   PicIns.Width = 0
End Sub
```

Khi người dùng Click vào check box. Nếu có chọn sẽ làm cho timer hoạt động tương tự như Form_Load(), nếu không chọn thì cho timer ngừng.

```
Private Sub chkUnload_Click()
If chkUnload.Value = 1 Then
   PicIns.Visible = True
   Timer1.Interval = 1000
Else
   Timer1.Interval = 0
   PicIns.Visible = False
End If
PicIns.Width = 0
End Sub
```

Kiểm tra xem độ rộng picIns >= picOut hay không. Nếu có, kết thúc chương trình (End), nếu không tiếp tục tăng độ rộng picIns theo chu kỳ mỗi giây 1 lần.

```
Private Sub Timer1_Timer()
If PicIns.Width >= PicOut.Width Then
   End
Else
   PicIns.Width = PicIns.Width + PicOut.Width / 20
End If
End Sub
```

Chạy thử chương trình xem nó có tự động thoát không. Thử click vào check box xem có hoạt động như mong muốn chưa.

Tô son điểm phấn

Thêm vài lời nhắc nhỡ

Chúng ta còn sót 1 đối tượng là lblMsg (Label box) chưa sử dụng đến. Label box này ta dùng để in câu thông báo hướng dẫn mỗi khi người dùng rê Mouse qua các Command Button.

Thí dụ như: Khi rê mouse trên nút Windows thì câu thông báo sẽ là "Các ứng dụng chuẩn của Windows" chẳng hạn. Để làm được điều này ta hãy khảo sát tình huống MouseMove của đối tượng, cụ thể là của Command Button và Form. Right click vào nút Windows, chọn View code, chọn tình huống MouseMove.

```
Private Sub cmdWin_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    lblMsg.Caption = "Run Windows Utilities (Accessories group)"
End Sub
```

Hiển thị câu thông báo Run Windows Utilities (Accessories group) trong lblMsg khi mouse di chuyển trên nút Windows. Một cách tương tự bạn có thể làm cho các button còn lại.

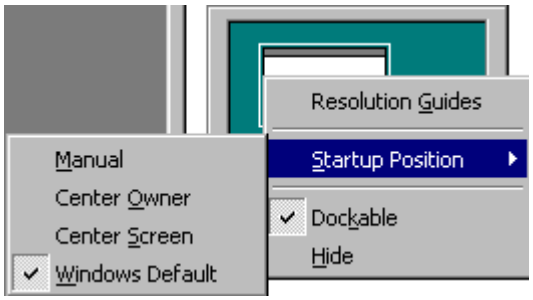
```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    lblMsg.Caption = "Please, select a program to start your work."
End Sub
```

Hiển thị câu thông báo Please, select a program to start your work. trong lblMsg khi mouse di chuyển phía trên form

Khi chạy chương trình, bạn thử rê mouse lên các button sẽ thấy nội dung của lblMsg thay đổi liên tục (hiển thị các câu thông báo của chính bạn).

Làm sao để form khởi động ở giữa màn hình

Đối với Visual Basic version 5 & 6, thì chuyện này rất dễ nhưng có vẻ bí hiểm. Bạn chỉ cần right click lên cửa sổ Form Layout (nếu chưa hiển thị hãy bật lên bằng cách View\Form Layout Window) chọn Startup Position, chọn Center Screen là xong ngay.



Không những thế, bạn còn có thể tự hiệu chỉnh vị trí form sẽ hiển thị trên màn hình khi chạy một cách rất trực quan. Còn nếu bạn khoái thủ công, hãy thêm dòng lệnh này vào tình huống FormLoad của form cần canh giữa màn hình như sau.

```
Private Sub Form_Load()
    Me.Move (Screen.Width - Me.Width) \ 2, (Screen.Height - Me.Height) \ 2
End Sub
```

Cách này áp dụng cho mọi phiên bản của Visual Basic.

Cuối cùng bạn chỉ dịch ra file EXE để chạy.

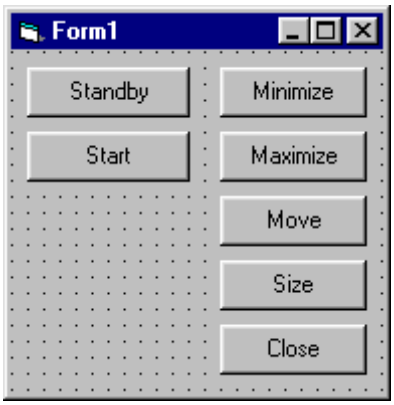
Chúc bạn thành công !

Lập trình với hàm API bằng

Visual Basic & Delphi

Bạn có thể thực hiện các chức năng với một cửa sổ như Phóng to, Thu nhỏ, Gửi xuống Taskbar, Di chuyển, Chỉnh kích thước hoặc bật nút **Start** của Windows hay đặt chế độ **Standby**, chạy **Screen Saver** thậm chí **tắt màn hình** máy tính của mình bằng cách gọi hàm API. Chương trình VB dưới đây mô phỏng những việc này.

Bạn thiết kế giao diện và các đối tượng như hình dưới đây



Caption	Name
Standby	cmdStandby
Start	cmdStart
Minimize	cmdMinimize
Maximize	cmdMaximize
Move	cmdMove
Size	cmdSize
Close	cmdClose

Copy đoạn code này và dán vào chương trình của bạn

```
Private Const WM_SYSCOMMAND = &H112
Private Const SC_SCREENSAVE = &HF140&
Private Const SC_MINIMIZE = &HF020&
Private Const SC_MAXIMIZE = &HF030&
Private Const SC_RESTORE = &HF120&
Private Const SC_TASKLIST = &HF130&
Private Const SC_MOVE = &HF010&
Private Const SC_SIZE = &HF000&

Private Declare Function SendMessage Lib "user32" Alias "SendMessageA" _
(ByVal hwnd As Long, ByVal wMsg As Long, ByVal wParam As Long, lParam As Any)
```

```
As Long

Dim WDMax As Boolean

Private Sub cmdMinimize_Click()
SendMessage Form1.hwnd, WM_SYSCOMMAND, SC_MINIMIZE, 0
End Sub

Private Sub cmdMaximize_Click()
If WDMax = True Then
SendMessage Form1.hwnd, WM_SYSCOMMAND, SC_RESTORE, 0
WDMax = False
Else
SendMessage Form1.hwnd, WM_SYSCOMMAND, SC_MAXIMIZE, 0
WDMax = True
End If
End Sub

Private Sub CmdClose_Click()
End
End Sub

Private Sub cmdMove_Click()
SendMessage Form1.hwnd, WM_SYSCOMMAND, SC_MOVE, 0
End Sub

Private Sub cmdSize_Click()
SendMessage Form1.hwnd, WM_SYSCOMMAND, SC_SIZE, 0
End Sub

Private Sub cmdStandby_Click()
SendMessage Form1.hwnd, WM_SYSCOMMAND, SC_SCREENSAVE, 20
End Sub

Private Sub CmdStart_Click()
' Start menu
SendMessage Form1.hwnd, WM_SYSCOMMAND, SC_TASKLIST, 0
End Sub

o=========)================= END
==============================>
```

Với chương trình này bạn có thể làm được nhiều việc khá thú vị, nhưng tiếc là tôi không tìm ra cách để tắt màn hình và gọi trình bảo vệ màn hình (Screen Saver) bằng VB, do đó tôi sử dụng Borland Delphi 6.0 để thực hiện. Dưới đây là đoạn code bằng Delphi có thể tắt màn hình và chạy Screen Saver.
Nếu có thể bạn nên viết chương trình có chức năng đặt biểu tượng vào Systray, sau đó bật một Popup menu để chọn các chức năng như Đóng mở CD-ROM, Tắt màn hình, Chạy Screen Saver.... đó quả là một chương trình có ích.

Delphi

Bạn tự thiết kế giao diện, và trên đó bạn đặt 2 Button với Name là Button1 và Button2, Caption tuỳ ý, sau đó click đúp vào một Button để hiện ra cửa sổ soạn thảo và gõ đoạn lệnh sau vào.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
SendMessage(application.Handle,WM_syscommand,SC_MonitorPower,1);
{bạn có thấy số 1 ở gần cuối dòng lệnh trên không ? nó có nghĩa là Tắt màn hình,
bạn thay bằng số 0 (không) thì sẽ chuyển về chế độ Text }
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
{Tương tự như trên}
{1: Standby}
{0: Screen Save (chỉ có hiệu lực khi bạn đang sử dụng 1 trình Screen Saver)}
SendMessage(application.Handle,WM_syscommand,SC_ScreenSave,0);
```

Tự tạo chương trình nghe nhạc bằng VB 6.0

Các điều khiển của VB thật dồi dào, và vẫn liên tục phát triển, điều này giúp cho người lập trình nhanh chóng cho ra lò một sản phẩm không đến nỗi nào, mà chỉ trong một thời gian rất ngắn. Bài viết này trình bày về chương trình nghe nhạc số (MP3,WAV,MID) sử dụng điều khiển Windows Media Player, chương trình có khả năng phát tuần tự từng bài trong danh sách, save danh sách bài hát vào một file, cho phép Browse để chọn các bài hát và thêm vào danh sách, có chức năng ghi các thông tin cấu hình vào Registry để lưu giữ, khi chạy chiếm rất ít tài nguyên hệ thống, khởi động tức thì. Giao diện đơn giản dễ sử dụng, có các chức năng tối thiểu của một trình nghe nhạc, có mã nguồn hoàn chỉnh đi kèm

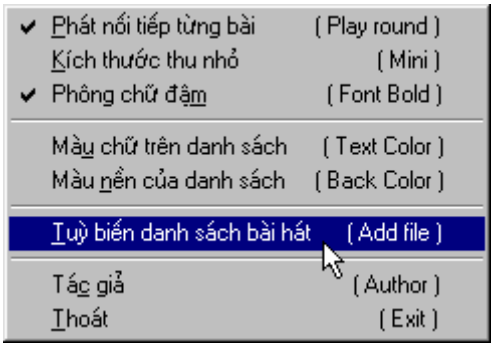
Chương trình này sử dụng file danh sách là một file kiểu bản ghi, điều này có lợi thế là truy xuất nhanh, thêm xoá sửa cũng dễ dàng hơn, nhưng bù lại kích thước file khá lớn.

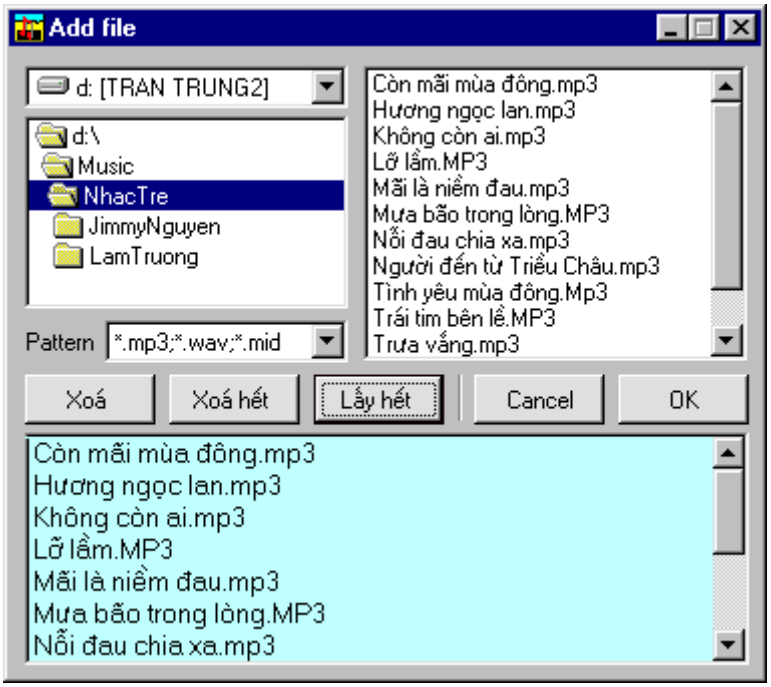
Với chương trình này bạn đã sở hữu trong tay một máy nghe nhạc, và với một chút kiến thức lập trình bạn có thể làm cho giao diện cũng như hoạt động của nó chuyên nghiệp hơn, chương trình còn nhiều hạn chế, tôi rất mong các bạn cải tiến cho nó mạnh hơn nữa.

Giao diện chương trình


My Media Player
Go
Playing
01:30 / 05:19
Hương ngọc lan.mp3
Không còn ai.mp3
Lỡ lầm.MP3
Mãi là niềm đau.mp3
Mưa bão trong lòng.MP3
Nỗi đau chia xa.mp3
Người đến từ Triều Châu.mp3
Tình yêu mùa đông.Mp3
Trái tim bên lề.MP3
Trưa vắng.mp3
Trở về dòng sông tuổi thơ.mp3
Trở về phố cũ.MP3


Author
Media Player
DEMO Version
Copyright (c) 2001 Tran Trung
Add: 23 Ba Trieu, Hoan Kiem, Hanoi
Email: tran_trung@lovemail.com
OK


Phát nối tiếp từng bài (Play round)
Kích thước thu nhỏ (Mini)
Phông chữ đậm (Font Bold)
Màu chữ trên danh sách (Text Color)
Màu nền của danh sách (Back Color)
Tuỳ biến danh sách bài hát (Add file)
Tác giả (Author)
Thoát (Exit)



Mã nguồn của chương trình.

Tôi không liệt kê thuộc tính của các control được sử dụng trong chương trình vì đã có mã nguồn hoàn chỉnh đi kèm, bạn chỉ việc download project này về ổ cứng, giải nén và mở nó bằng Visual Basic là xong. Tôi sử dụng Visual Basic 6.0, Windows 98 SE, nếu bạn dùng các phiên bản cũ hơn có thể chương trình không chạy.

1. Tạo một Project mới

Thêm vào Project một Modul với tên là Modul1

- Nội dung:

```
Option Explicit
'Kiểu bản ghi của file danh sách, chỉ gồm 2 trường
Type Media
Path As String * 250
Name As String * 100
'Tên file bài hát không dài quá 250 ký tự
'Đường dẫn không dài quá 100 ký tự
End Type
```

2. Đặt tên cho Form hiện hành là frmMedia

- Nội dung:

```
Dim Song As Media
Dim DATAfile As String
Dim RecEnd
```

```
Dim i, Filenum, Sogia As Integer
Dim p

'Hàm kiểm tra sự tồn tại của 1 file
Function FileExists(FileName) As Boolean
Dim Msg As String
On Error GoTo CheckError
FileExists = (Dir(FileName) <> "")
Exit Function
CheckError:
Const mnErrDiskNotReady = 71, mnErrDeviceUnavailable = 68
If (Err.Number = mnErrDiskNotReady) Then
Msg = "Put a floppy disk in the drive."
If MsgBox(Msg, vbExclamation & vbOKCancel) = vbOK Then
Resume
Else
Resume Next
End If
ElseIf Err.Number = mnErrDeviceUnavailable Then
Msg = "This drive or path does not exist: " & FileName
MsgBox Msg, vbExclamation
Resume Next
Else
Msg = "Unexpected error #" & Str(Err.Number) & " occurred: " _
& Err.Description
MsgBox Msg, vbCritical
Stop
End If
Resume
End Function

Private Sub cmdCapNhat_Click()
Capnhat
End Sub

Private Sub Command1_Click()
PopupMenu mnuSetting
End Sub

Private Sub Capnhat()
Filenum = FreeFile
Open DATAfile For Random As #Filenum Len = Len(Song)
RecEnd = FileLen(DATAfile) / Len(Song)
For i = 1 To RecEnd
Get #Filenum, i, Song
List1.AddItem (Trim(Song.Name))
List2.AddItem (Trim(Song.Path))
Next i
Close #Filenum
End Sub

Private Sub Form_Load()
Volume1.Value = 10 'Giá trị mặc định của Volume khi khởi động
```

```
'Mở file danh sách
If Len(App.Path) > 3 Then
DATAfile = App.Path & "\TMedia.lst"
Else
DATAfile = App.Path & "TMedia.lst"
End If
mnuRepeat.Checked = True
mnuMini.Checked = False
On Error Resume Next
mnuMini.Checked = GetSetting("FastRun 1.0", "Media", "Check Mini")
mnuRepeat.Checked = GetSetting("FastRun 1.0", "Media", "Check Repeat")
frmMedia.Top = GetSetting("FastRun 1.0", "Media", "Media Top")
frmMedia.Left = GetSetting("FastRun 1.0", "Media", "Media Left")
List1.BackColor = GetSetting("FastRun 1.0", "Media", "Back Color")
List1.ForeColor = GetSetting("FastRun 1.0", "Media", "Text Color")
mnuDam.Checked = GetSetting("FastRun 1.0", "Media", "Font Bold")
Hengio = GetSetting("FastRun 1.0", "Media", "Time Song")
Volume1.Value = GetSetting("FastRun 1.0", "Media", "Volume")
CheckDefaultList = GetSetting("FastRun 1.0", "Media", "DefaultList")
Capnhat
Mini
Dam
Volume1_Scroll
End Sub

Private Sub SaveReg()

'Ghi cấu hình vào Registry
On Error Resume Next
SaveSetting "FastRun 1.0", "Media", "Check Mini", mnuMini.Checked
SaveSetting "FastRun 1.0", "Media", "Check Repeat", mnuRepeat.Checked
SaveSetting "FastRun 1.0", "Media", "Media Top", frmMedia.Top
SaveSetting "FastRun 1.0", "Media", "Media Left", frmMedia.Left
SaveSetting "FastRun 1.0", "Media", "Volume", Volume1.Value
SaveSetting "FastRun 1.0", "Media", "Font Bold", mnuDam.Checked
SaveSetting "FastRun 1.0", "Media", "Back Color", List1.BackColor
SaveSetting "FastRun 1.0", "Media", "Text Color", List1.ForeColor
DeleteSetting "FastRun 1.0", "Media", "Time Song"
End Sub

Private Sub KetThuc()
SaveReg
Unload frmMedia
Unload frmAuthor
Unload frmOpen
End Sub

Private Sub Form_Unload(Cancel As Integer)
KetThuc
End Sub

Private Sub List1_DblClick()
If FileExists(List2.List(List1.ListIndex)) = True Then
```

```
MediaPlayer1.FileName = List2.List(List1.ListIndex)
ThanhCong = True
Else
If List1.ListIndex = List1.ListCount - 1 And ThanhCong = False Then
MsgBox "TÊt c¶ c¸c bµi trong danh s¸ch ®Òu sai ®êng dÉn hoÆc tªn file." + vbCrLf
+ "B¹n cÇn n¹p l¹i danh s¸ch !", vbCritical, "Media - Warning"
Else
HetBai
End If
End If
End Sub


Private Sub HetBai()
If mnuRepeat.Checked = True And List1.ListCount > 0 Then
If List1.ListIndex + 1 < List1.ListCount Then
List1.ListIndex = List1.ListIndex + 1
Else
List1.ListIndex = 0
ThanhCong = False
End If
On Error Resume Next
List1_DblClick
End If
End Sub

Private Sub List1_KeyPress(KeyAscii As Integer)
If Keyascii = 13 Then
List1_DblClick
End End Sub

Private Sub List1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As
Single)
If List1.ListIndex >= 0 Then
List1.ToolTipText = Left(List1.List(List1.ListIndex), Len(List1.List(List1.ListIndex)) -
3)
End If
End Sub

Private Sub MediaPlayer1_EndOfStream(ByVal Result As Long)
'Hành động khi hết một bài

HetBai
End Sub

Private Sub mnuAdd_Click()
frmOpen.Show vbModal
End Sub

Private Sub mnuAuthor_Click()
frmAuthor.Show
End Sub
```

```
Private Sub mnuDelete_Click()
frmListEdit.Show
End Sub

Private Sub mnuChu_Click()
CommonDialog1.Color = List1.ForeColor
CommonDialog1.Action = 3
List1.ForeColor = CommonDialog1.Color
End Sub

Private Sub mnuDam_Click()
If mnuDam.Checked = False Then
List1.FontBold = False
mnuDam.Checked = True
Else
List1.FontBold = True
mnuDam.Checked = False
End If
Dam
End Sub

Private Sub Dam()
If mnuDam.Checked = False Then
List1.FontBold = False
Else
List1.FontBold = True
End If
End Sub

Private Sub mnuExit_Click()
KetThuc
End Sub

Private Sub mnuMini_Click()
If mnuMini.Checked = True Then
mnuMini.Checked = False
Else
mnuMini.Checked = True
End If
Mini
End Sub

Private Sub Mini()
If mnuMini.Checked = True Then
List1.Height = 255
frmMedia.Height = 1740
List1.ListIndex = List1.ListIndex
Else
List1.Height = 2400
frmMedia.Height = 3885
End If
End Sub
```

```
Private Sub mnuNumber_Click()
If mnuNumber.Checked = True Then
mnuNumber.Checked = False
Else
mnuNumber.Checked = True
End If
End Sub

Private Sub mnuNen_Click()
CommonDialog1.Color = List1.BackColor
CommonDialog1.Action = 3
List1.BackColor = CommonDialog1.Color
End Sub

Private Sub mnuRepeat_Click()
If mnuRepeat.Checked = True Then
mnuRepeat.Checked = False
Else
mnuRepeat.Checked = True
End If
End Sub

Private Sub Text1_Click()
Text1.Text = Str(MediaPlayer1.Volume)
End Sub

Private Sub Volume1_Scroll()
Select Case Volume1.Value
Case 13: Sogia = 0
Case 12: Sogia = -40
Case 11: Sogia = -90
Case 10: Sogia = -180
Case 9: Sogia = -280
Case 8: Sogia = -410
Case 7: Sogia = -500
Case 6: Sogia = -650
Case 5: Sogia = -860
Case 4: Sogia = -1100
Case 3: Sogia = -1350
Case 2: Sogia = -1900
Case 1: Sogia = -2600
Case 0: Sogia = -9640
End Select
MediaPlayer1.Volume = Sogia
End Sub
```

3. Tạo một form mới đặt tên là frmOpen

-Nội dung:

```
Option Explicit
Dim SongOpen As Media
```

```
Dim i, CurrentSong, Filenum As Integer
Dim PathSong As String
Dim DATAfile As String
Dim RecEnd

Function FileExists(FileName) As Boolean
Dim Msg As String
On Error GoTo CheckError
FileExists = (Dir(FileName) <> "")
Exit Function
CheckError:
Const mnErrDiskNotReady = 71, mnErrDeviceUnavailable = 68
If (Err.Number = mnErrDiskNotReady) Then
Msg = "Put a floppy disk in the drive."
If MsgBox(Msg, vbExclamation & vbOKCancel) = vbOK Then
Resume
Else
Resume Next
End If
Else If Err.Number = mnErrDeviceUnavailable Then
Msg = "This drive or path does not exist: " & FileName
MsgBox Msg, vbExclamation
Resume Next
Else
Msg = "Unexpected error #" & Str(Err.Number) & " occurred: " _
& Err.Description
MsgBox Msg, vbCritical
Stop
End If
Resume
End Function

Private Sub cmdAddAll_Click()
If Len(Dir1.Path) = 3 Then
PathSong = Dir1.Path
Else
PathSong = Dir1.Path + "\"
End If
For i = 0 To File1.ListCount - 1
List1.AddItem (File1.List(i))
List2.AddItem (PathSong + File1.List(i))
Next i
If cmdClear.Enabled = False Then
cmdClear.Enabled = True
End If
KTnutClear
End Sub

Private Sub cmdCancel_Click()
Unload frmOpen
End Sub

Private Sub cmdClear_Click()
```

```
KTnutClear
If cmdClear.Enabled = True Then
If List1.ListIndex < 0 And List1.ListCount > 0 Then
List1.ListIndex = 0
End If
CurrentSong = List1.ListIndex
List1.RemoveItem (CurrentSong)
List2.RemoveItem (CurrentSong)
If List1.ListCount < 0 Then
List1.ListIndex = List1.ListCount - 1
End If
If List1.ListCount = 0 Then
cmdClear.Enabled = False
End If
End If
End Sub

Private Sub cmdClearAll_Click()
KTnutClear
If cmdClearAll.Enabled = True Then
List1.Clear
List2.Clear
End If
End Sub

Private Sub cmdOK_Click()
'save in file
If Len(App.Path) > 3 Then
DATAfile = App.Path + "\TMedia.lst"
Else
DATAfile = App.Path + "TMedia.lst"
End If
If FileExists(DATAfile) = True Then
Kill DATAfile
End If
frmMedia.List1.Clear
frmMedia.List2.Clear
If List1.ListCount > 0 Then
Filenum = FreeFile
Open DATAfile For Random As #Filenum Len = Len(SongOpen)
If List1.ListCount > 0 Then
For i = 0 To List1.ListCount - 1
SongOpen.Name = List1.List(i)
SongOpen.Path = List2.List(i)
Put #Filenum, i + 1, SongOpen
Next i
End If
Close #Filenum
frmMedia.cmdCapNhat.Value = True
End If
Unload frmOpen
frmMedia.SetFocus
End Sub
```

```
Private Sub Combo1_Click()
File1.Pattern = Combo1.Text
If Combo1.ListIndex = 1 Then
cmdAddAll.Enabled = False
MsgBox "NÕu b¹n chän kiÓu file lµ '' *.* '', b¹n sÏ kh«ng thªm ®îc file vµo danh
s¸ch", vbCritical, "Warning"
Else
cmdAddAll.Enabled = True
End If
End Sub

Private Sub Dir1_Change()
File1.Path = Dir1.Path
KTnutAddAll
End Sub

Private Sub Dir1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
Dir1.Path = Dir1.List(Dir1.ListIndex)
'File1_DblClick
End If
End Sub

Private Sub Drive1_Change()
On Error Resume Next
Dir1.Path = Drive1.Drive
If Err Then
MsgBox "Kh«ng t×m thÊy ®Üa", vbCritical, "Media - Warning"
Drive1.Drive = Dir1.Path
End If
End Sub

Private Sub File1_DblClick()
If File1.Pattern <> "*.*" Then
If Len(Dir1.Path) = 3 Then
PathSong = Dir1.Path + File1.FileName
Else
PathSong = Dir1.Path + "\" + File1.FileName
End If
List1.AddItem (File1.FileName)
List2.AddItem (PathSong)
If cmdClear.Enabled = False Then
cmdClear.Enabled = True
End If
KTnutClear
Else
MsgBox "B¹n cÇn ®Æt kiÓu file trong hép Pattern lµ ''*.mp3;*.wav;*.mid''",
vbCritical, "Media - Warning"
End If
End Sub
```

```
Private Sub File1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
File1_DblClick
End If
End Sub

Private Sub Form_Load()
For i = 0 To frmMedia.List1.ListCount - 1
List1.AddItem (frmMedia.List1.List(i))
List2.AddItem (frmMedia.List2.List(i))
Next i
KTnutAddAll
KTnutClear
Combo1.ListIndex = 0
File1.Pattern = Combo1.Text
File1.Hidden = True
File1.ReadOnly = True
File1.System = True
End Sub

Private Sub KTnutAddAll()
If File1.ListCount > 0 And File1.Pattern <> "*.*" Then
cmdAddAll.Enabled = True
Else
cmdAddAll.Enabled = False
End If
End Sub

Private Sub KTnutClear()
If List1.ListCount > 0 Then
cmdClear.Enabled = True
cmdClearAll.Enabled = True
Else
cmdClear.Enabled = False
cmdClearAll.Enabled = False
End If
End Sub
```

4.Tạo thêm một form đặt tên là frmAuthor

-Nội dung:

Đây là form ghi thông tin về tác giả chương trình, tuỳ ý bạn

Sử dụng Visual Data Manager của Visual Basic 5.0

Khi sử dụng Visual Basic (VBasic), điều khiển bạn hài lòng có lẽ là số lượng "đồ nghề" dồi dào của nó trong Tool Box. Bạn đang thiết kế một form cho ứng dụng có xử lý đến cơ sở dữ liệu và

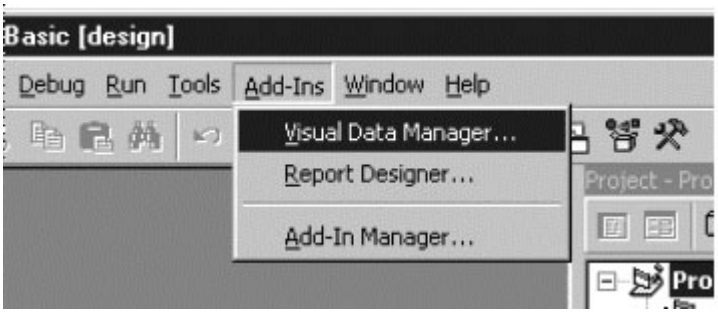
đang phân vân sẽ đặt vào đấy Data control hay DBGrid? Chưa hết, bạn cần những thao tác nào trên form để quyết định sẽ có bao nhiêu nút lệnh? Muốn có nhiều thao tác, ắt phải cần càng nhiều nút lệnh mà việc lập trình cho các nút lệnh có khi không đơn giản. VBasic có tiện ích hỗ trợ trong trường hợp bạn cần một form có khả năng duyệt từng record trong một Table cùng với những thao tác thêm, xóa, sửa, di chuyển, dò tìm và xếp thứ tự. Dĩ nhiên, tiện ích ấy cũng tự động phát sinh mã lệnh cho các thao tác. Đó là trình Data Manager vốn được xem là "trị giá gia tăng" có sẵn trong VBasic 4.0. Nhưng kể từ phiên bản VBasic 5.0, trình tiện ích ấy được "lột xác" hoàn toàn và gọi bằng tên mới là Visual Data Manager (VisData).

Với chức năng nói trên, tiện ích này khác với hệ Microsoft Access ở chỗ nó cho phép bạn tạo bất cứ CSDL nào: Access, dBase, FoxPro hay Paradox. Và vì đó là trình quản trị dữ liệu nên VisData sẵn lòng chấp nhận những chỉ thị SQL nếu bạn muốn có các queries trong CSDL. Nếu đó vẫn chưa phải là lý do để bạn sử dụng VisData, thì hãy tưởng tượng bạn đang viết một ứng dụng bằng VBasic, nhưng cũng cần đến những thông tin về một CSDL nào đó bằng cách sử dụng MS-Access ở chế độ Inactive Window, bạn có tin rằng tốc độ máy của bạn sẽ chậm lại không? Và nếu VBasic của bạn cũng đang cần đến CSDL mà MS-Access đang mở thì sao? Hẳn bạn phải đóng CSDL đó trong MS-Access. Duy trì một cửa sổ MS-Access trống rỗng trong chế độ Inactive là một sự phí phạm tài nguyên của máy và thời gian của bạn. Bạn có sẵn lòng "xài sang" những thứ đó không?

VisData còn có những chức năng liên quan đến việc sử dụng CSDL phân tán. Nghĩa là một CSDL dùng trong môi trường mạng. Tuy nhiên trong bài này, chúng ta chỉ xét đến những chức năng của nó trên máy đơn.

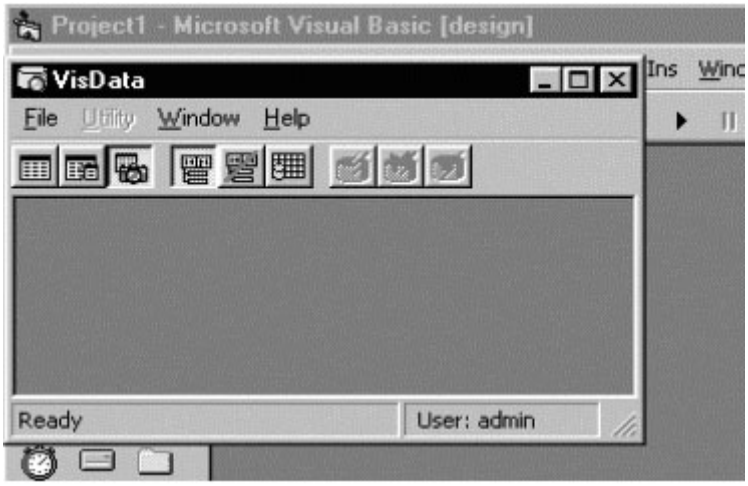
Gọi sử dụng VisData

Từ trong menu chính của VBasic, bạn có thể gọi sử dụng VisData bằng lệnh: Add-Ins, Visual Data Manager. Cây lệnh mô tả qua Hình 1.



Hình 1. LƯnh gäi VisData

Sau đó cửa sổ VisData xuất hiện trên mặt trước của cửa sổ VBasic như Hình 2.



Hình 2. Cưa sỉ VisData

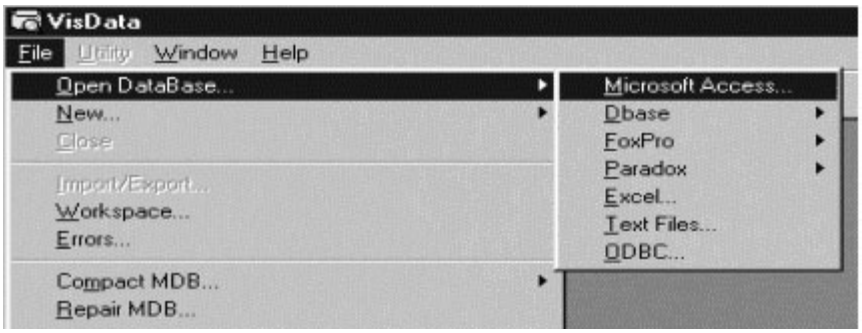
Menu và thanh công cụ của VisData tương đối đơn giản. Menu File gồm những lệnh liên quan đến mở/tạo lập, bảo trì CSDL. Menu Utilities phản ánh hai tiện ích quan trọng là trình Query Builder giúp xây dựng SQL trực quan và bộ Data Form Designer giúp tự động tạo form nếu bạn đang dùng VBasic và muốn VisData hỗ trợ tạo form để xử lý dữ liệu trong table. Các nút công cụ chia thành ba nhóm. Nhóm thứ nhất giúp người sử dụng xác định xem mình sẽ mở/tạo một CSDL theo kiểu nào trong những kiểu Tableset, Dynaset, hay Snapshot. Nhóm nút công cụ thứ hai được dùng khi bạn muốn tạo một form và trong form đó sẽ đặt một Data control hay DBGrid control. Nhóm nút công cụ thứ ba dành cho những trường hợp dùng đến CSDL phân tán. Trong VisData có thể hiển thị hai cửa sổ Database Windows và SQL Statement.

Mở/Tạo lập CSDL

Đối với một CSDL đã có sẵn, trước khi mở bạn hãy xác định VisData sẽ mở nó theo kiểu nào trong những kiểu Tableset, Dynaset, hay Snapshot. Những nút công cụ cho trong Hình 3 cho biết nút nào giúp bạn làm việc đó. Bạn sẽ dùng lệnh File.Open Database để mở CSDL hay lệnh File.New nếu muốn tạo một CSDL mới.

Hình 3. Quy ®Þnh kiÎu CSDL sÏ më

Hình 4 mô tả cây lệnh cho thao tác này.

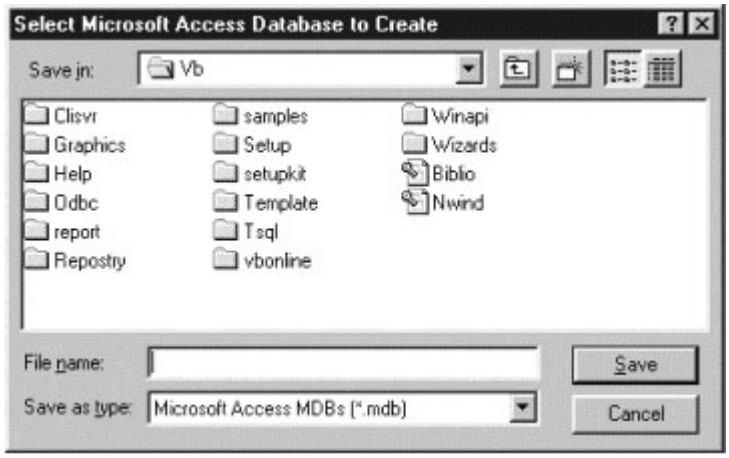


Hình 4. C©y lƯnh më/tạo CSDL

Trường hợp muốn tạo CSDL FoxPro, bạn phải khai báo đường dẫn cho CSDL sẽ tạo trong hộp thoại như VisData trình bày ở Hình 5.

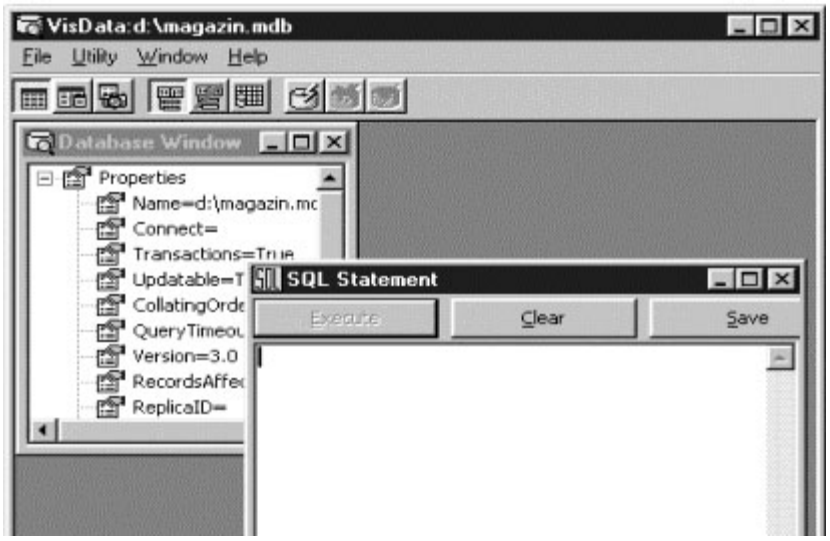
Hình 5. Khai báo ®ưêng dÉn khi tạo CSDL FoxPro

Nếu muốn tạo CSDL .MDB của MS-Access, bạn sẽ khai báo tên của CSDL trong hộp thoại như Hình 6.



Hình 6. Tạo CSDL .MDB cđa MS-Access

Giả sử bạn muốn tạo một CSDL dưới format của Access 7.0 có tên magazin.mdb trong đĩa D:\, lúc đó vùng làm việc của VisData sẽ có hai cửa sổ Database Window và SQL Statement. Cửa sổ Database Window là nơi thể hiện về các properties của bản thân CSDL này và của những đối tượng trong CSDL như Table, Query, ... Những properties này được thể hiện dưới dạng cây thư mục. Cửa sổ SQL Statement là nơi để phát những chỉ thị SQL với những nút lệnh Execute thi hành chỉ thị, nút lệnh Clear để xóa chỉ thị và nút lệnh Save để lưu chỉ thị SQL hiện hành dưới một tên, tên đó được gọi là QueryDef. Hình 7 minh họa những mô tả trên.



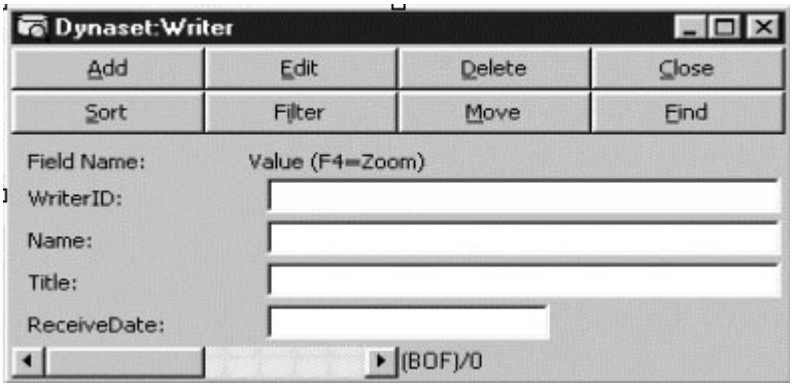
Hình 7. Cưa sỉ Database và SQL Statement

Tạo một Table

Bạn có thể tạo một table bằng lệnh SQL phát ra trong cửa sổ SQL Statement theo dạng lệnh: CREATE TABLE Table (Field1, Type(Size),...) và sau đó bấm vào nút lệnh Execute để thi hành. Ví dụ muốn tạo Table Writer cho CSDL magazin.mdb, bạn có thể nhập vào dòng lệnh: Create Table Writer (writerid Text(5), Name Text(25), Title Text(32), ReceiveDate Date time)

Bây giờ trên màn hình sẽ hỏi bạn đây có phải là một chỉ thị PassThrough SQL không, bạn nhớ trả lời "No" vì một chỉ thị PassThrough SQL sẽ dành riêng cho ODBC xử lý. ở đây chúng ta chưa bàn đến ODBC là gì.

Với một table vừa khai báo xong, bạn có muốn xem những properties của bảng này? Hãy bấm vào dấu Aà+' kế bên tên của bảng trong cửa sổ Database Window. Bạn muốn nhập liệu vào bảng? Hãy nhấn đúp vào tên của bảng. Lúc đó một bộ duyệt nội dung sẽ thể hiện dưới dạng form như trong Hình 8.



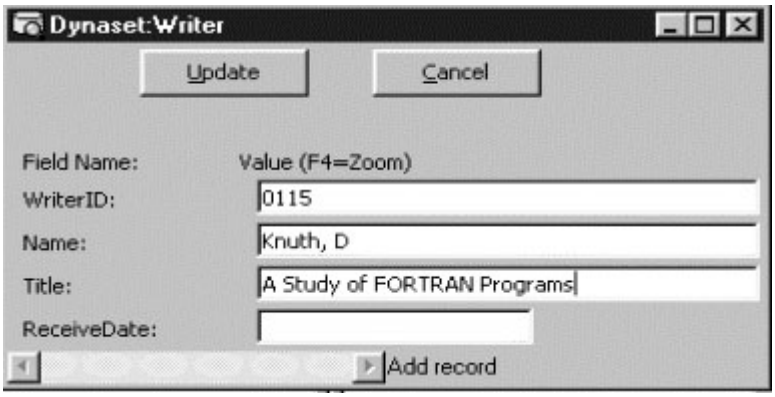
Hình 8. Form nhËp liƯu và duyƯt bảng Writer

Các nút lệnh liên quan đến những thao tác trên record; các hộp Text box để nhập nội dung cho từng trường và thanh trượt để duyệt nội dung từng record.

Xin lưu ý tùy bạn đang mở bảng theo kiểu nào trong những kiểu TableSet, DynaSet hay SnapShot và có đặt Data Control vào form hay không mà hình thức của form nhập liệu có thay đổi đôi chút về các nút lệnh trong form. Nhưng điều đó sẽ không làm bạn lúng túng khi sử dụng form. Dưới đây, chúng tôi sẽ mô tả đôi nét về việc sử dụng form trong trường hợp mở bảng theo kiểu DynaSet và trong form không có Data Control.

Sử dụng form nhập liệu

Sau khi đã làm xuất hiện form nhập liệu như Hình 8, hẳn bạn cũng muốn nhập một số record đầu tiên cho bảng. Form không cho phép bạn nhập nội dung các trường vào hộp text box. Bạn hãy bấm chuột vào nút lệnh Add để bắt đầu. Hình 9 minh họa form nhập liệu.

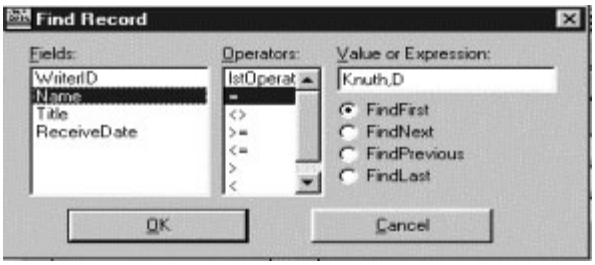


Hình 9. NhËp néi dung record

Khi nhập xong, bạn bấm chuột vào nút lệnh Update để ghi nhận hoặc Cancel trong trường hợp ngược lại. Để chỉnh sửa nội dung record, dùng Edit; để loại bỏ record hiện hành trong form, dùng Delete; để thay đổi record hiện hành, dùng Move và cung cấp độ dời tính từ record hiện hành. Độ dời là một số nguyên dương hay nguyên âm tùy theo bạn muốn dời về hướng đầu bảng hay cuối bảng. Bạn có muốn xếp thứ tự bảng theo một trường nào đó không? Chỉ cần nhấn chuột vào nút lệnh Sort và khai báo tên trường mà bạn muốn dùng làm khoá. Hình 10 là hộp thoại mà bạn sẽ cung cấp tên khóa sắp xếp.

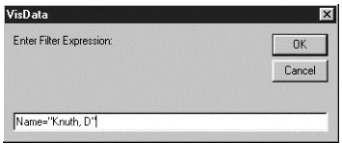
Hình 10. Khai báo khóa sắp xếp cho bảng.

Nếu muốn dò tìm một record nào đó, thay vì sử dụng thanh trượt, bạn dùng nút lệnh Find và chọn khóa dò tìm, chọn toán tử và chọn giá trị cho khóa dò tìm. Với những record trùng khóa dò tìm thì sao? Liệu VisData chỉ có khả năng tìm duy nhất một record đầu tiên trùng khóa? VisData chấp nhận dò tìm cả những record trùng khóa khác nếu ở lần dò tìm sau bạn chọn nhiệm ý Find Next. Hình 11 minh họa cách dùng lệnh Find để dò tìm record liên quan đến tác giả Knuth,D.



Hình 11. Khai báo biÎu thøc dß tìm.

Cuối cùng là thao tác lọc những record thỏa mãn một điều kiện cho trước mà với những bạn quen dùng Fox thì đó là chỉ thị SET FILTER. Chẳng hạn, bạn muốn lọc tất cả những record về tác giả Knuth,D., hãy nhấn chuột vào nút lệnh Filter rồi cung cấp điều kiện lọc record vào hộp thoại như Hình 12 chỉ ra.

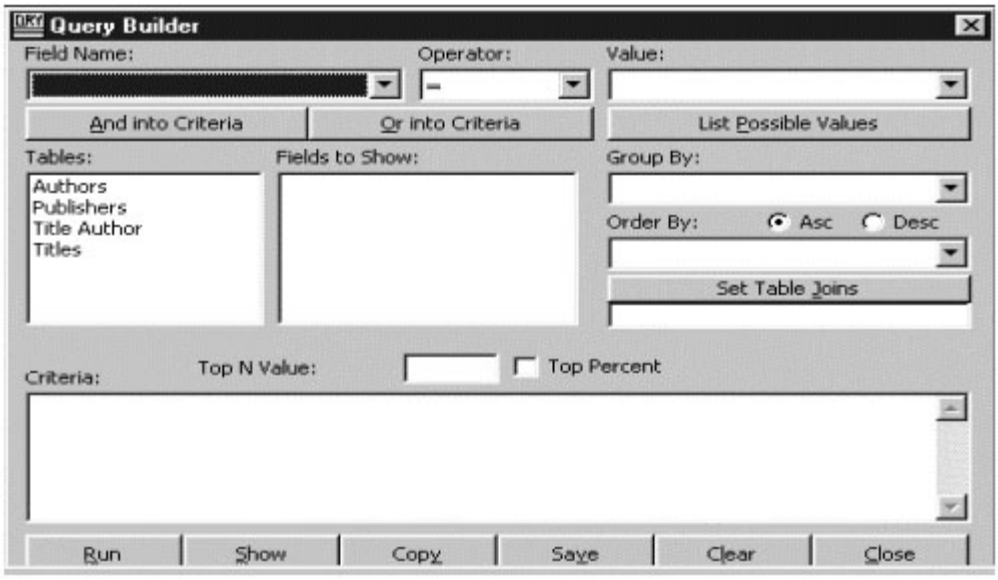


Hình 12. Läc những record theo mét ®iịu kiƯn

Sử dụng chỉ thị SQL

VisData cũng có khả năng chạy những chỉ thị SQL nếu bạn nhập dòng lệnh vào cửa sổ SQL Statement rồi bấm chuột vào nút lệnh Execute để thi hành. Khi có hộp thoại hỏi bạn chỉ thị SQL sắp thi hành có phải là một SQLPassThrough hay không, bạn nhớ chọn "No". Nếu không bị bắt lỗi, Queries thu được sẽ là một form. Để làm ví dụ, bạn thử mở CSDL Biblio.mdb (là CSDL cài đặt theo VBasic) bằng lệnh File.Open và phát chỉ thị chọn tên, địa chỉ, mã của những nhà xuất bản của thành phố New York. Bạn có nhập dòng lệnh SELECT Name, Address, PubID FROM Publishers WHERE City="New York" hay không?

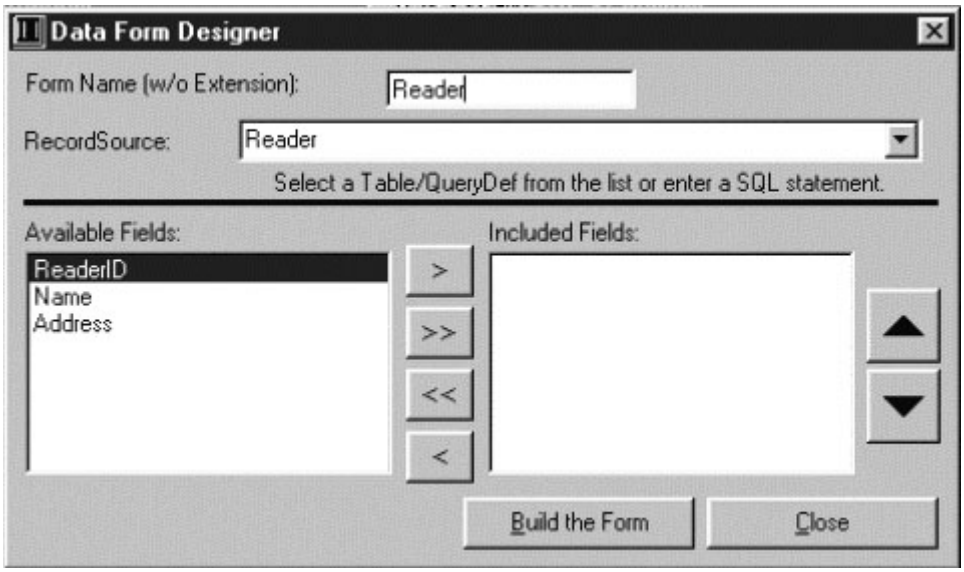
Nếu muốn thử công cụ tạo Query, chẳng hạn như wizard, hãy dùng lệnh Utility.Query Builder. Bạn muốn chọn các bảng và trường cần cho Query, dĩ nhiên phải chọn trong hai hộp liệt kê Tables và Fields to Show. Để cung cấp điều kiện cần truy vấn, bạn có thể dùng các hộp combo Field Name, Operator, Value. Nếu trong điều kiện cần những toán tử And, Or bạn dùng hai nút And into Criteria, và Or into Criteria.



Hình 13. Tạo Query bằng wizard thông qua Query Builder

Tạo form và sinh mã cho ứng dụng

Giả sử trong một CSDL mới, bạn đã tạo một bảng có tên Reader với các trường ReaderID TEXT(5), Name TEXT(25), Address TEXT(255) và đang định tạo form nhập record cho bảng này với mã lệnh bằng VBasic. Từ VisData, bạn mở CSDL đó, quyết định sẽ mở CSDL theo kiểu nào trong những kiểu TableSet, DynaSet hay SnapShot. Và bạn sẽ đặt đối tượng Data Control hay DBGrid vào trong form tương lai. Bạn chọn những ấn định đó bằng nút công cụ như mô tả ở Hình 2. Sau đó bạn sẽ lấy lệnh Utility. Data Form Designer. Hình 14 cho thấy những gì bạn phải khai báo để tạo lập một form: nhập tên của table vào hộp RecordSource, nhập tên form vào hộp Form Name w/o Extension, và chọn các trường mà bạn muốn thể hiện trong form. Các trường này được chọn từ hộp list box Available Fields để chuyển vào Include Fields. Công việc chấm dứt bằng nút lệnh Build the Form. Bây giờ có thể đóng VisData bằng lệnh File.Exit để trở về VBasic. Hẳn bạn sẽ ngạc nhiên với form mới tạo với những chi tiết khá chuyên nghiệp. Thích thú hơn nữa khi chuyển sang màn hình Code, bạn sẽ thấy mã lệnh đã có sẵn.



Hình 14. Hép thoại cđa Data Form Designer

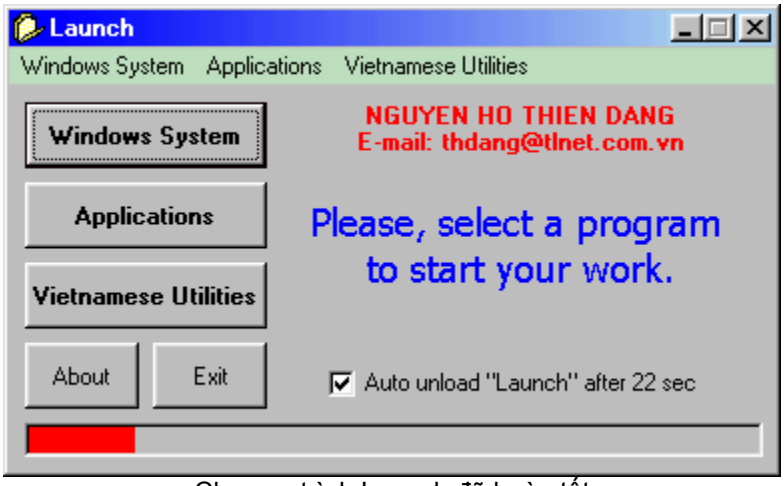
Đến đây bạn có thể nói gì về phiên bản mới của VBasic? Đáng để nâng cấp phải không?

Xây dựng chương trình để khởi động các chương trình khác

Nhập Dẫn

Để bày vẻ và quảng cáo thêm cho máy tính có rất nhiều cách. Nhưng nếu bạn là người chuyên ráp máy hay một cửa hàng tin học thì có lẻ bạn nên viết 1 chương trình tự động chạy lúc khởi động (Start Up) để trưng bày các phần mềm đã cài đặt trên máy, luôn tiện giới thiệu chút ít về mình hay cửa hàng. Ta tạm đặt tên cho chương trình đầu tiên này là "Launch" nhé.

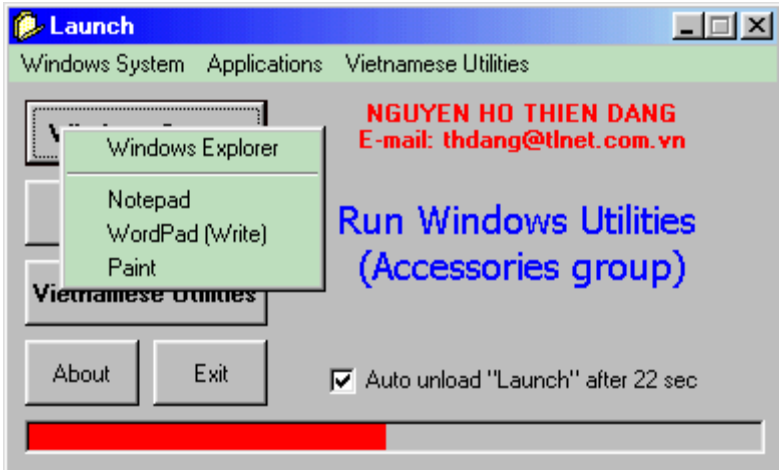
Viết chương trình, có rất nhiều ngôn ngữ lập trình dư sức để viết một chương trình như vậy. Nhưng dễ dàng hơn hết, có lẻ và Visual Basic. Viết chương trình này bằng Visual Basic không đòi hỏi gì bạn nhiều, chỉ cần chút khéo léo và áp dụng các công cụ của Visual Basic một cách thích hợp cộng thêm chút ít sáng tạo mà thôi.



Chương trình Launch đã hoàn tất

Có rất nhiều cách để phân loại các phần mềm cài đặt trên máy. Chương trình thí dụ này phân nhóm các ứng dụng cài đặt trong Windows thành 3 nhóm: Windows System (các công cụ chuẩn của Windows), Applications (các ứng dụng của người dùng cài thêm), Vietnamese Utilities (các tiện ích về tiếng việt).

Cơ chế hoạt động của chương trình này là "mồi lửa" cho người sử dụng khởi động chương trình mà mình cần bằng menu hay các button của chương trình, sau đó chương trình này tự động "biến".



Để mồi lửa cho người sử dụng, ta buộc phải biết chính xác vị trí của các file chương trình mà người dùng cần.

Đầu tiên là các tiện ích kèm theo Windows như WordPad, Notepad, Paint, ... (tôi chỉ xin thí dụ 3 chương trình thôi). Các file thực thi của chương trình này chủ yếu nằm trong thư mục Windows, vậy chỉ cần tìm ra thư mục Windows là ta có thể giải quyết được vấn đề.

Chuyện này cũng rất dễ dàng và vô cùng may mắn là Windows có khả năng tự động tìm kiếm các file thực thi trong thư mục Windows và Windows\System cho nên chúng ta khỏe (ta có thể hiểu là nó tự đặt dường dẫn đến thư mục Windows và Windows\System).

Chú ý: Khi muốn gọi WordPad bạn phải gọi file write.exe và Paint phải gọi file Pbrush.exe. 2 file này không phải là file chương trình chính, chúng chỉ có nhiệm vụ gọi file thực thi của Paint (MSPaint.exe) và WordPad (WordPad.exe) nằm trong Program files\Accessories mới thực sự là file chương trình chính.
Microsoft phải làm như vậy để tương thích với các chương trình cũ của Windows 3.x. Sau đây là tên file của một số chương trình có sẵn trong Windows (với điều kiện bạn phải cho cài đặt khi setup Windows).

Tên file	Thư mục	Chương trình
notepad.exe	Windows	Notepad
write.exe	Windows	WordPad
Pbrush.exe	Windows	Paint
Cleanmgr.exe	Windows	Disk Cleanup (W98)
Defrag.exe	Windows	Disk Defragmenter
Scandskw.exe	Windows	Scan Disk
Sndvol32.exe	Windows	Volume Control
Winfile.exe	Windows	File manager
Msconfig.exe	System	System Configuration Utility
Sfc.exe	System	System file checker
Sysedit.exe	System	System Configuration Editor
....	....	....

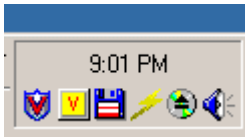
Vậy là chuyện gọi 1 chương trình của Windows không có gì khó khăn, chỉ cần biết tên file là xong, mọi chuyện còn lại là của Windows.

Còn các ứng dụng khác do người dùng hay người lắp máy cài đặt thêm chỉ có cách là gọi theo đường dẫn chính xác vì mỗi máy mỗi khác, công sức cho việc tìm kiếm khá gian nan, chưa hợp với các tay nghiệp dư như chúng ta. Khuyết điểm của chương trình là phải viết riêng cho từng máy, chúng ta sẽ giải quyết vấn đề này ở bài viết sau "Nâng cấp chương trình Launch".

Bạn cũng có thể viết chương trình này cho chức năng Auto Run của CD chương trình, lúc nay mọi chuyện lại càng dễ dàng vì các đường dẫn và chương trình trên CD đều nằm trong tay ta, cứ đi từ thư mục gốc vào là xong chuyện.

Dùng ActiveX (.OCX)
để đưa chương trình vào System Tray

Để đưa được chương trình của mình vào System Tray chúng ta cần phải lập trình, cũng không có gì phức tạp lắm. Tuy nhiên cách nhanh nhất là dùng một ActiveX (tập tin .OCX) để giúp chúng ta dễ dàng đưa chương trình của mình vào System tray mà không hề tốn một giọt mồ hôi. Tôi xin giới thiệu với các bạn một ActiveX tên là ZTray dùng trong các phiên bản Visual Basic 32 bit (hoàn toàn free). Bạn chỉ cần tạo đối tượng này vào chương trình & đặt các thuộc tính thích hợp cho nó tức thì chương trình của bạn bay cái vèo vào System Tray, thiệt hết ý.



Bắt đầu với ZTray control

()) Đặc điểm

- Chỉ cần tạo một đối tượng duy nhất , nó sẽ hoá phép cho chương trình của bạn bay vào System Tray.

- Biểu tượng (Icon) của chương trình trong System Tray phải là file biểu tượng (*.ico).

- Icon này phải được đặt trong một ImageList, kích thước không thành vấn đề.

- Nếu bạn không chỉ định Icon, nó sẽ tự động lấy Icon mặc định của nó (là quả địa cầu, trông xấu tệ).

- & còn nhiều thứ nữa ....

()) Một số thuộc tính & sự kiện

Ngoài những thuộc tính, sự kiện bình thường của một đối tượng trong môi trường VB. ZTray còn có các thuộc tính đặc sau.

** ImageList Property*

Thuộc tính này để bạn có thể gán cho nó một ImageList. Trong ImageList này chứa (các) Icon mà nó dùng làm biểu tượng chương trình trong System Tray. Chỉ có thể thay đổi lúc Design, lúc chương trình đang chạy bạn không thay đổi được thuộc tính này đâu.

Cú pháp: ZTray.ImageList [=value]

[value] Chính là tên của ImageList (kiểu String)

- Nếu không có ImageList nó sẽ tự động xài cái Icon mặc định của mình.

- Nếu bạn có thay đổi biểu tượng lúc chương trình thực thi bạn phải chủ động để nó biết bằng cách thay đổi chuộc tính ImageNumber hoặc gán ShowInTray = True (ngay cả khi nó đang là True).

- Nhắc lại nữa: Biểu tượng phải là icon file.

**** ShowInTray Property***

Nhận giá trị Boolean (TRUE/FALSE). True nghĩa là cho hiện Icon trong System Tray. Là False thì ngược lại.

Cú pháp: ZTray.ShowInTray [=value]

[value] là True hay False

- ShowInTray sẽ có hiệu lực ngay khi chương trình bắt đầu Run.

- Có thể gán bạn True để Update cho Icon trong System Tray nếu có thay đổi.

**** ImageNumber Property***

Thuộc tính này dùng để gán hoặc truy xuất thứ tự của Image mà ZTray dùng làm Icon cho chương trình. Giá trị này là chỉ số của Image trong một ImageList.

Cú pháp: ZTray.ImageNumber [=value]

[value] Chỉ số của image trong ImageList mà ZTray dùng làm biểu tượng (làm một Integer).

Biểu tượng tự động cập nhật khi thuộc tính này có sự thay đổi.

**** TipText Property***

Dùng để gán hoặc truy xuất đến ToolTip của đối tượng, ToolTip này sẽ tự động xuất hiện khi bạn rê mouse đến trên biểu tượng trong System Tray.

Cú pháp: ZTray.TipText [=value]

[value] Là một String. Độ dài tối đa là 64 ký tự, nếu bạn cố tình cho một string quá dài, nó tự động cắt bớt.

- Sẽ có tác dụng ngay lập tức nếu thay đổi.

**** Click Event***

Xẩy ra khi người dùng click nút (trái hay phải) chuột vào Icon trong System Tray.

Cú pháp: Private Sub ZTray_Click (Button as integer)

[Button] cho biết nút nào được nhấn.

1 là Left Mouse Button

2 là Right Mouse Button

** DblClick Property*

Xẩy ra khi người dùng Double click vào Icon trong System Tray (cả trái lẫn phải đều được công nhận một cách rõ ràng).

Cú pháp:

Private Sub ZTray_DblClick (Button as integer)

[Button] cho biết nút nào được nhấn.

1 là Left Mouse Button

2 là Right Mouse Button

()) Minh họa cách sử dụng

1. Chuẩn bị:

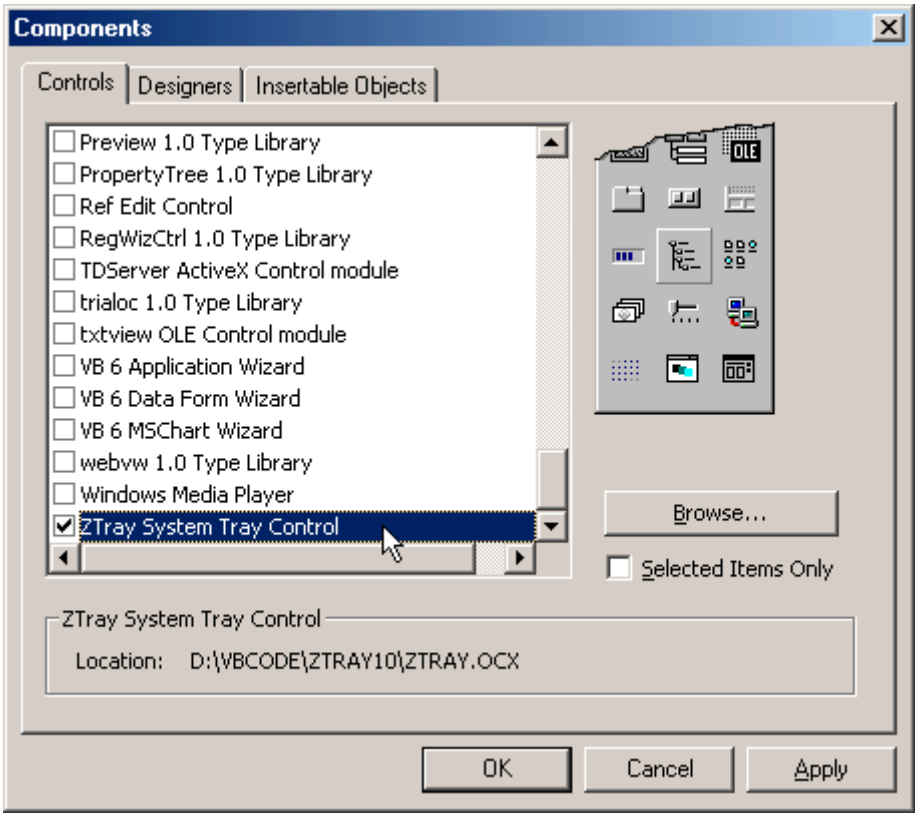
Sau đây là một chương trình thí dụ minh họa cách sử dụng đối tượng ZTray.

Đầu tiên bạn hãy chuẩn bị tập tin ZTray.ocx, nếu chưa có hãy vào WebLH tải về, mở nén vào thư mục System của Windows.

Bạn hãy khởi động VB, tạo một Project mới để bắt đầu cuộc thử nghiệm.

2. Đưa ZTray vào đề án:

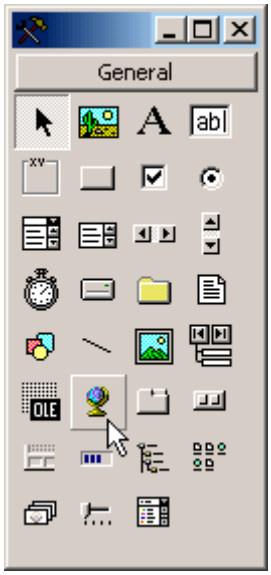
Project / Components hoặc dùng tổ hợp phím Ctrl - T để mở cửa sổ Components. Cick chọn ActiveX tên ZTray System Tray Control.



Nếu chưa có trong Danh sách bạn có thể Click nút Browse để chọn tập tin ZTray.ocx từ một thư mục nào đó

Vì ZTray đòi hỏi có một ImageList nên bạn phải click chọn thêm "Microsoft Windows Common Controls 6.0".

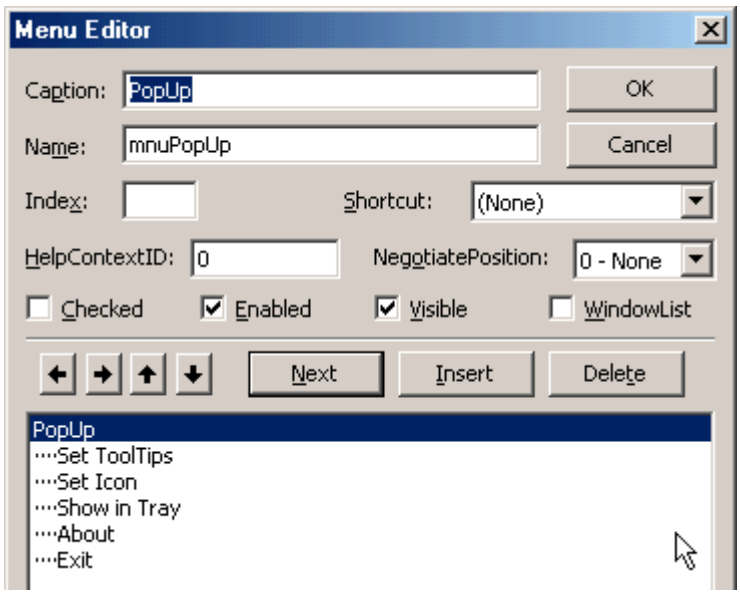
Click OK bạn sẽ thấy ZTray Control xuất hiện trên hộp ToolBox của VB.



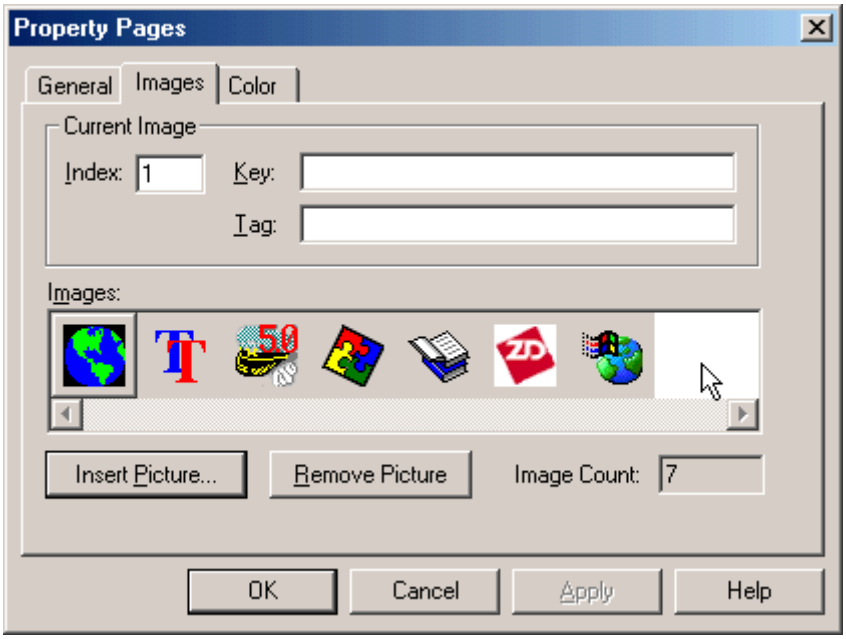
3. Thiết kế:

Trên Form1, bạn tạo một ImageList tên là ImageList1 & một ZTray tên là ZTray1. Và tạo các menu có tên tương ứng như sau:

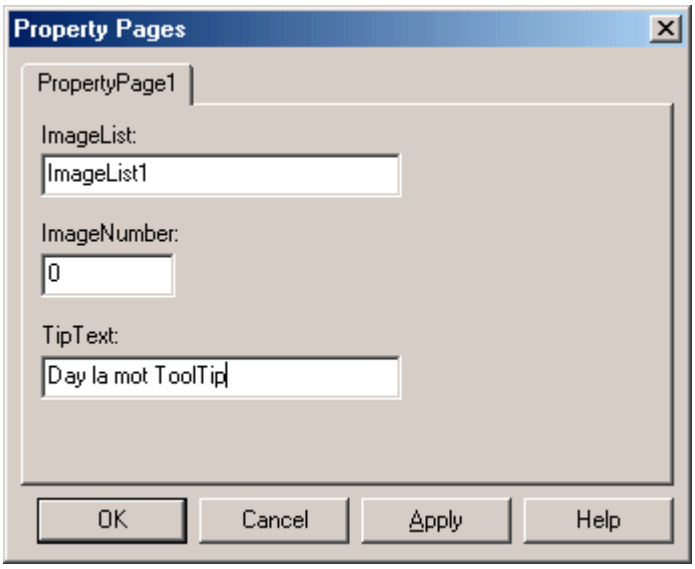
Caption	Name	Checked
PopUp	mnuPopUp	
Set ToolTips	mnuTips	
Show In Tray	mnuShow	True
About	mnuAbout	
Exit	mnuExit	



Click phải chuột lên ImageList1 vừa tạo khi nãy, chọn Properties, chọn tiếp thẻ Images, dùng nút Insert Picture để thêm vào một số Image (nhớ là phải dùng Icon file). Đại loại như sau, vậy là ta có 7 hình (tứ 1 đến 7). Click OK để đóng hộp thoại này lại.



Lại click phải lên ZTray1 vừa tạo. Nhập vào khung ImageList là ImageList1, ImageNumber là 1 (ảnh đầu tiên trong ImageList1), TipText: nhập đại một vài chữ, xong click OK.



Sau đó tiến hành viết code cho chương trình như sau

Option Explicit

Private Sub Form_Load()

ZTray1.ImageNumber = 1

ZTray1.ShowInTray = True

End Sub

Ngay lúc form được nạp, đặt chỉ số cho ImageNumber & cho hiện biểu tượng trong System Tray.

Private Sub Form_Unload(Cancel As Integer)

```
ZTray1.ShowTray = True

Visible = False

Cancel = 1
```

End Sub

Khi người dùng click nút close trên thanh Title bar, chương trình sẽ không thoát mà chỉ ẩn form đi thôi. Nhưng trước khi giấu Form nó lại cho hiện Icon nếu lúc đó ẩn để tránh trường hợp cả Form lẫn Icon đều biến mất.

Private Sub mnuAbout_Click()

```
'About

MsgBox "ZTray Demo by Thien Dang 30/07/2000"
```

End Sub

Private Sub mnuExit_Click()

```
'Exit

End
```

End Sub

Thoát khi chương trình bằng lịnh Exit trong menu.

Private Sub mnuIcon_Click()

```
' Set Icon

ZTray1.ImageNumber = InputBox("Image (1 - 7) ?", "Icon", 1)
```

End Sub

Thay đổi Icon cho ZTray. Ta có thể nhập vào số từ 1 đến 7 do có 7 Image như đã nói ở trên.

Private Sub mnuShow_Click()

```
'Show In Tray

mnuShow.Checked = Not (mnuShow.Checked)
```

```
    ZTray1.ShowInTray = mnuShow.Checked

    If mnuShow.Checked = False Then Visible = True

End Sub
```

Chức năng này có nhiệm vụ bật tắt cái Icon của chương trình. Khi bạn tắt nó sẽ tự hiển thị form lên để tránh trường hợp cả Icon lẫn form đều mất tích.

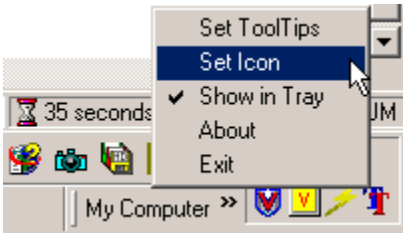
```
Private Sub mnuTips_Click()

    ' Set Tooltips

    ZTray1.TipText = InputBox("Your text here", "Enter TipText", "Text")

End Sub
```

Thay đổi ToolTip. Độ dài tối đa 64 ký tự.

```
Private Sub ZTray1_Click(button As Integer)

If button = 1 Then

    Me.Visible = True

    SetFocus

Else

    PopupMenu mnuPopUp

End If

End Sub
```

Nếu click nút trái thì hiện Form, nút phải thì hiện menu.

Bây giờ bạn có thể chạy thử chương trình của mình rồi đấy. Bạn có thể click phải chuột trên Icon trong System Tray để truy xuất menu.



Chúc bạn thành công.

Những câu hỏi nhỏ

Sau đây là một số câu hỏi nhỏ của các bạn tôi, bạn có thể dùng chúng để tô điểm thêm cho ứng dụng Visual Basic của mình. Đây không phải là những điều cao siêu, lạ lẫm và cũng không phải là cách giải thích tối ưu nhưng nó giúp chúng ta hiểu thấu đáo được một số vấn đề. Hy vọng sau khi đọc xong bạn sẽ "à thì ra là vậy ! ..."

Làm sao để có được những dòng chữ chạy liên tục trên màn hình ?

Thật ra chuyện này cũng dễ hiểu, bạn chỉ cần cắt chữ ở đầu đoạn văn bản và gắn nó vào cuối đoạn văn bản, làm liên tục như vậy sẽ tạo cho người dùng có cảm giác là dòng chữ đang chạy. Bạn hãy mở 1 form mới, trên đó tạo 1 textbox (Text1), gán 1 dòng văn bản vào thuộc tính text của textbox, tạo 1 timer (timer1).

Khi form load sẽ khởi động Timer với trị Interval = 100

```
Private Sub Form_Load()
   Timer1.Interval = 100
End Sub
```

Và timer sẽ xử lý các lệnh theo yêu cầu của bạn mỗi khi nó phát sinh 1 sự kiện thời gian.

```
Private Sub Timer1_Timer()
   Dim x As String
   Dim y As String
'gán x = 1 ký tự đầu dòng văn bản
   x = Left(Text1.Text, 1)
'gán y là phần còn lại
   y = Right(Text1.Text, Len(Text1.Text) - 1)
'Hiển thị trở lại textbox theo thứ tự ngược lại.
   Text1.Text = y + x
End Sub
```

Bạn có thể thay textbox bằng labelbox để người dùng không thể can thiệp vào dòng văn bản đang chạy, lúc này bạn phải thay đổi thuộc tính caption thay vì text của textbox.

Canh form giữa màn hình khi hiển thị ?

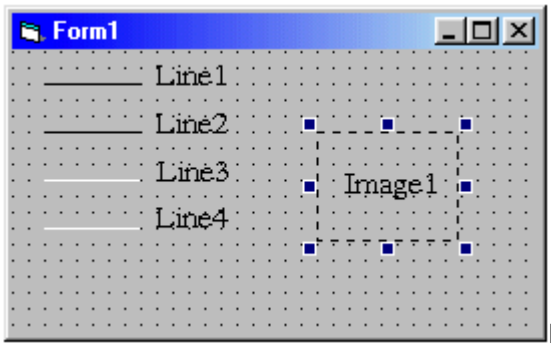
Chỉ việc thêm đính lệnh này vào thủ tục tình huống FormLoad của form tương ứng.

```
Me.Move (Screen.Width - Me.Width)\2, Screen.Height.Height - Me.Height)\2
```

Làm thế nào để tạo hiệu ứng 3D ?

Bạn hãy mở 1 form trống, trên đó tạo 1 image, vẻ 4 đối tượng Line. Khảo sát tình huống MouseMove của Image, khi rê mouse trên Image lập tức 4 đối tượng line sẽ hiển thị xung quanh

Image. Còn 3D ư ? Bạn chỉ việc cho 2 line của cạnh phải và dưới màu đen (Line1, Line2), còn 2 line của cạnh trái và trên màu trắng (Line3, Line4).

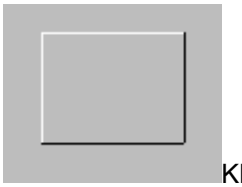


Form khi thiết kế

Khi di chuyển Mouse trên form, mọi chuyện đều bình thường, 4 đối tượng Line không xuất hiện (thuộc tính Visible=False)

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Line1.Visible = False
    Line2.Visible = False
    Line3.Visible = False
    Line4.Visible = False
End Sub
```

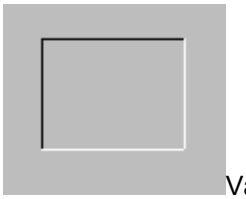
Khi di chuyển mouse trên đối tượng Image, 4 line sẽ được xếp xung quanh và hiển thị lại bằng cách thay đổi các thuộc tính X1, Y1, X2, Y2 của line. Nhờ có màu sắc thích hợp nên ta có cảm giác Image nổi lên khi rê mouse đến.


Khi rê mouse đến

```
Private Sub Image1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
'Cạnh Phải
    Line1.X1 = Image1.Left + Image1.Width
    Line1.Y1 = Image1.Top
    Line1.X2 = Image1.Left + Image1.Width
    Line1.Y2 = Image1.Top + Image1.Height
'Cạnh dưới
    Line2.X1 = Image1.Left
    Line2.Y1 = Image1.Top + Image1.Height
    Line2.X2 = Image1.Left + Image1.Width
    Line2.Y2 = Image1.Top + Image1.Height
'Cạnh trái
    Line3.X1 = Image1.Left
    Line3.Y1 = Image1.Top
    Line3.X2 = Image1.Left
    Line3.Y2 = Image1.Top + Image1.Height
'Phía trên
    Line4.X1 = Image1.Left
```

```
    Line4.Y1 = Image1.Top
    Line4.X2 = Image1.Left + Image1.Width
    Line4.Y2 = Image1.Top
'Cho hiện lại 4 đối tượng Line
    Line1.Visible = True
    Line2.Visible = True
    Line3.Visible = True
    Line4.Visible = True
End Sub
```

Khi nhấn Mouse trên Image, sự kiện MouseDown (nhấn mouse) phát sinh và đảo màu của 4 đối tượng line tạo cảm giác Image bị lõm xuống.

 Và khi nhấn mouse

```
Private Sub Image1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Line1.BorderColor = QBColor(7) 'Màu trắng
    Line2.BorderColor = QBColor(7)
    Line3.BorderColor = QBColor(0) 'Màu đen
    Line4.BorderColor = QBColor(0)
End Sub
```

Sau khi nhấn, thả mouse ra làm phát sinh sự kiện MouseUp, các lệnh cần xử lý trong sự kiện này là trả màu sắc lại như ban đầu.

```
Private Sub Image1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Line1.BorderColor = QBColor(0)
    Line2.BorderColor = QBColor(0)
    Line3.BorderColor = QBColor(7)
    Line4.BorderColor = QBColor(7)
End Sub
```

Thấy có vẻ hơi cực khổ quá các bạn nhỉ ?

Custom Control trong Visual Basic 5.0

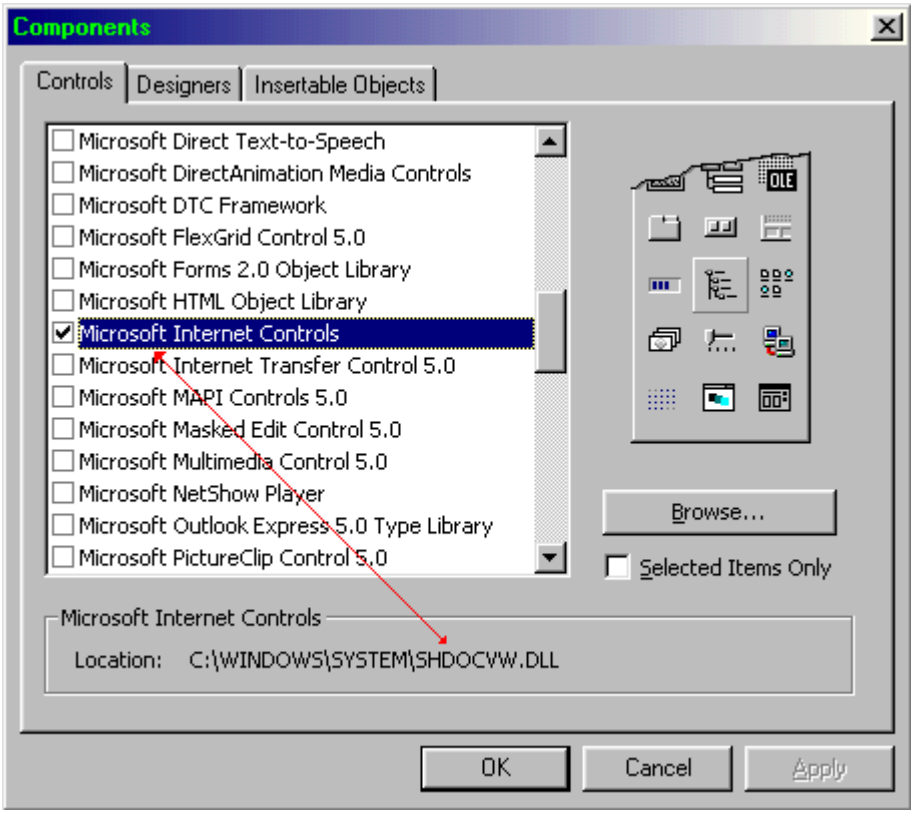
Khi viết một ứng dụng trong VB nếu không sử dụng thêm bất cứ một Custom Control nào ngoài các Control chuẩn của VB. Sau khi dịch thành file EXE, muốn chép sang máy khác bạn phải chép kèm theo 2 file thư viện chuẩn VB nằm trong thư mục System.

MSVBVM50.DLL (1.355.776 bytes)

CTL3D32.DLL (45.056 bytes)

Đây là 2 file thư viện cần thiết cho bất cứ ứng dụng nào viết bằng Visual Basic. Bởi vậy có nhiều khi file chương trình EXE của bạn chỉ vài ba chục Kb mà phải vác theo 2 file này quả là hơi bất tiện, tuy nhiên bạn có thể nén chúng lại cho nhỏ bớt, tôi đã thử và sau khi nén chỉ còn 655.557 bytes thay vì 1.400.832 bytes như lúc đầu.

Còn nếu trong ứng dụng của bạn có xài thêm các Custom Control thì nên lưu ý phải chép thêm các file tương ứng, có như vậy khi đem qua máy khác chương trình của bạn mới chạy được.



Khi bạn chọn một Custom Control trong hộp thoại Components thì file tương ứng sẽ được hiển thị ở phần Location, bạn hãy căn cứ vào đây mà tìm chép cho đúng. Các file này thường có phần mở rộng là DLL hay OCX nằm trong thư mục System (có thể mở bằng Visual C++).

Hãy chép chúng vào thư mục Windows, System, các thư mục đã được đặt đường dẫn PATH, hay cho chung vào cùng thư mục với file EXE của máy cần chạy chương trình của bạn.

Có thể dùng một chương trình tạo bộ đĩa Setup và chỉ định cho chúng chép thêm các file này, ví dụ như Create Install chẳng hạn. Đồng thời nếu có trình Setup, chương trình của bạn trông có vẻ đàng hoàng và chuyên nghiệp hơn (có thể tin cậy được).

Phiên bản của ứng dụng Visual Basic

Khi bạn viết một chương trình bằng Visual Basic, trong ứng dụng của bạn luôn xuất hiện một đối tượng tên là App, trong các thuộc tính của đối tượng App có 3 thuộc tính lưu giữ số phiên bản (Version). Ta có thể dễ dàng truy cập 3 thuộc tính này để biết được version của chương trình.

App.Major: Con số chính

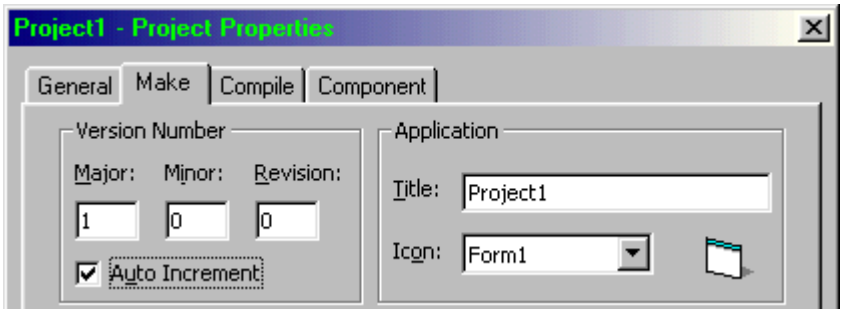
App.Minor: Con số phụ

App.Revision: Con số này cho biết số lần bạn hiệu chỉnh và dịch lại chương trình.

Bạn có thể dùng một Msgbox để thể hiện Version của chương trình:

Msgbox "Version: " & App.Major & "." & App.Minor & App.Revision

Tuy nhiên bạn có thể để cho con số Revision tự động tăng mỗi lần dịch chương trình, vào Project \ Properties, chọn tab Make và click chọn mục Auto Increment trong khung Version Number

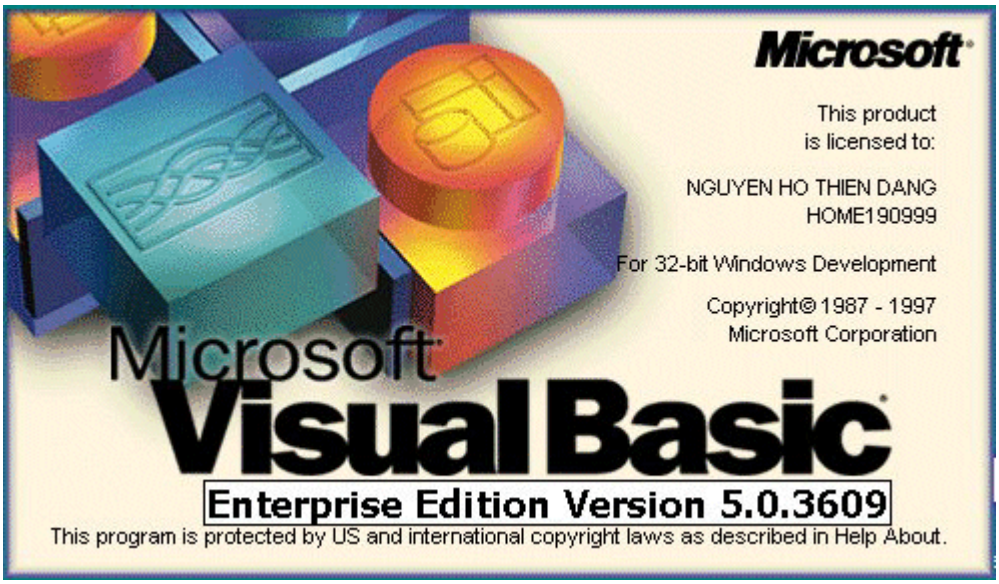


Hoặc cho chương trình thể hiện phiên bản lên Caption của Form khi Load

```
Private Sub Form_Load()
   Me.Caption = Me.Caption & " - Ver " & App.Major & "." & App.Minor & App.Revision
End Sub
```

Tạo màn hình Splash Screen

Bạn có để ý khi khởi động Visual Basic không ? Ttrước khi chương trình Visual Basic được khởi động thường có một khung như sau hiện ra trong chốc lát lại biến mất và VB sẵn sàng cho bạn làm việc. Không chỉ Visual Basic không đâu, hầu như tất cả các phần mềm hiện nay đều có màn hình này.



Khung đó ta gọi là màn hình Splash Screen, được cho hiển thị trong lúc khởi động nhằm tránh cho người dùng đỡ sốt ruột trong khi chương trình nạp dữ liệu hoặc thực hiện các lệnh cần thiết, trên đó thông báo các vấn đề về bản quyền, phiên bản, logo ... Khi chương trình đã sẵn sàng làm việc màn hình này tự động biến mất.

Trong VB màn hình Splash này thật ra cũng là một Form nhưng không hiển thị thanh tiêu đề, được nạp lên màn hình từ thủ tục tình huống FormLoad của 1 form nào đó trong chương trình (thường là Form chính - form sẽ luôn luôn được hiện dịch trong suốt quá trình làm việc).

```
Private Sub Form_Load()

    Me.Show
    frmSplash.Show
    DoEvents

    <Các lệnh cần thực hiện khi khởi động chương trình>

    Unload frmSplash

End Sub
```

Me.Show: bắt chương trình vẽ form chính lên màn hình. Bạn có thể ghi Show cũng được vì lệnh Show mặc nhiên tác động lên form hiện hành khi không dược chỉ rõ đối tượng.

frmSplash.Show: Cho hiển thị màn hình Splash (bạn lưu ý: cho đến lúc này frmSplash vẫn chưa hiện lên mặc dù đã gọi)

DoEvents: Chờ cho Windows hiển thị hoàn tất frmSplash lên màn hình.

Sau lệnh DoEvents là tập hợp các lệnh cần xử lý trong khi chương trình khởi động.

Cuối cùng là lệnh Unload frmSplash có tác dụng đóng màn hình Splash Screen lại. Lúc này chương trình đã sẵn sàng cho người dùng.

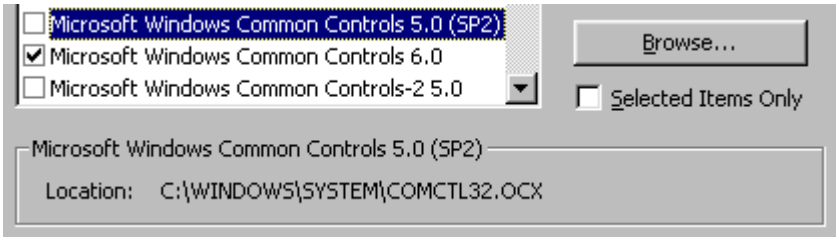
Ngôn ngữ lập trình Visual Basic ngày càng trở nên phổ biến, được rất nhiều người quan tâm tới. Trong vòng 2 năm trở lại đây bắt đầu xuất hiện khá nhiều sách viết về ngôn ngữ này tạo điều kiện cho chúng ta tìm hiểu, theo tôi dự đoán trong vài năm tới nó sẽ thông dụng như Word, Excel vậy. Cách thức lập trình trong ngôn ngữ này rất gần gũi với Windows (bạn sẽ hiểu Windows sâu sắc hơn khi lập trình bằng ngôn ngữ này). Nếu bạn đã học Microsoft Access thì đừng nên bỏ qua ngôn ngữ lập trình "thần tốc" này. Mặc dù chương trình được viết ra chạy không hiệu quả bằng những ngôn ngữ khác nhưng với nó bạn có thể tạo ra một ứng dụng Windows nhanh và dễ dàng như ... "nấu một gói mì ăn liền" lúc này vấn đề hiệu quả có thể tạm cho qua.

Nếu bạn muốn tìm hiểu căn bản về Visual Basic thì nên chọn quyển "Tự học lập trình Visual Basic 5" của tác giả Phạm Thùy Nhân, còn nếu chịu khó thì hãy tìm những quyển sách của Samis, thậm chí trong lúc đi xem sách tôi thấy có quyển ghi là: "chưa biết gì, đọc sách, gấp sách lại, thành chuyên gia" nữa đấy.

Tự tạo Progress Bar cho ứng dụng Visual Basic

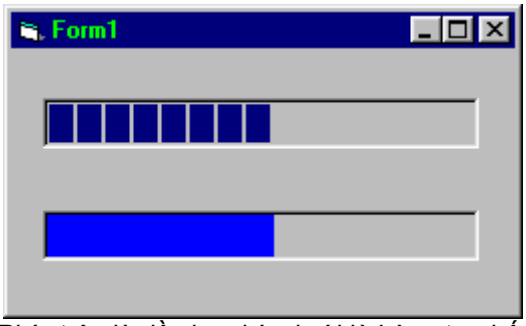
[Combo box tự hiện danh sách] [Form Layout]

Nếu các bạn dùng các phiên bản Enterprise - 32bit của Visual Basic, thì có sẵn 1 Custom control cho phép bạn tạo nhanh một Progress bar theo mốt của Windows 9.x trông rất đơn sơ nhưng chuẩn mực



Còn ở đây tôi xin trình bày cách tự chế Progress bar bằng các control chuẩn của Visual Basic (Picture box), cách này đặc biệt hữu ích cho các bạn còn dùng phiên bản vb 16bit hay không muốn vác theo file COMCTL.OCX (MSCOMCTL.OCX) kích thước trên dưới 1Mb kèm theo ứng dụng của mình.

Bạn hãy tạo 2 Picture box lồng vào nhau, di chuyển vị trí Picture box bên trong, chọn màu sắc cho thích hợp. Còn để nó chạy được ư ? bạn hãy tăng (hoặc giãm) độ rộng của Picture box bên trong, cực đại khi độ rộng Picture box bên trong = bên ngoài, cực tiểu khi độ rộng Picture box bên trong = 0.



Phía trên là đồ xịn, phía dưới là hàng tự chế.

Bạn thấy không cũng ngang ngữa chớ bộ, không những thế hàng tự tạo còn cho phép chúng ta thay đổi màu sắc một cách vô tư, muốn xanh đỏ tím vàng đều được cả, về khả năng này thì đồ xịn rất khó thực hiện nếu không muốn nói là không được.

Để minh họa tôi đã dùng một Timer (Timer1) và 2 Picture box (Picture1 và Picture2), Picture2 màu xanh nằm trong Picture1 màu xám. Cùng với đoạn mã sau:

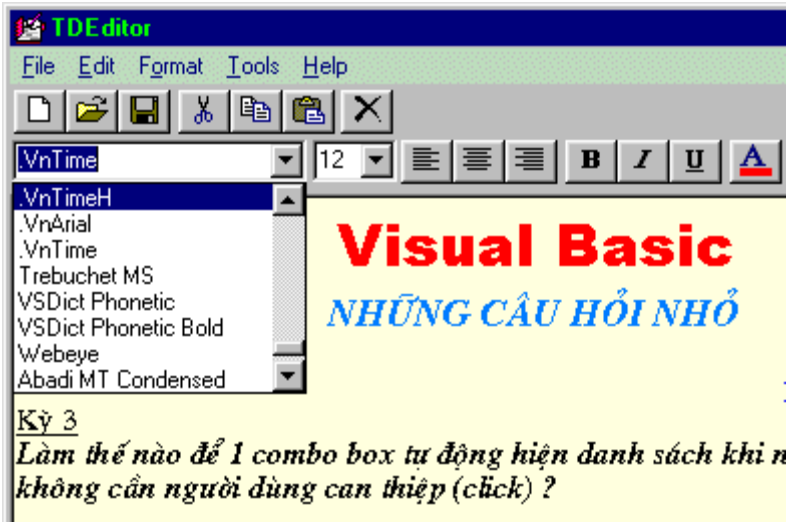
```
Option Explicit

Private Sub Form_Load()

    Picture2.Width = 0
    Timer1.Interval = 1000

End Sub

Private Sub Timer1_Timer()

If Picture2.Width < Picture1.Width Then

    Picture2.Width = Picture2.Width + Picture1.Width \ 20

Else

    Timer1.Interval = 0
    MsgBox "Đã 20 giây trôi qua rồi đấy !"
    End

End If

End Sub
```

Khi chạy thử chương trình bạn sẽ thấy Progress bar của chúng ta thay đổi mỗi giây 1 lần, cho đến hết 20 giây, thông báo bằng hàm MsgBox nếu bạn click OK thì nó goodbye bạn luôn.

Làm thế nào để 1 combo box tự động hiện danh sách khi nhận được focus, mà không cần người dùng click chuột ?

[Tự tạo Progress bar] [Form Layout]

Combo box dễ thấy nhất trong các thảo trình là hộp chọn font như hình dưới đây.



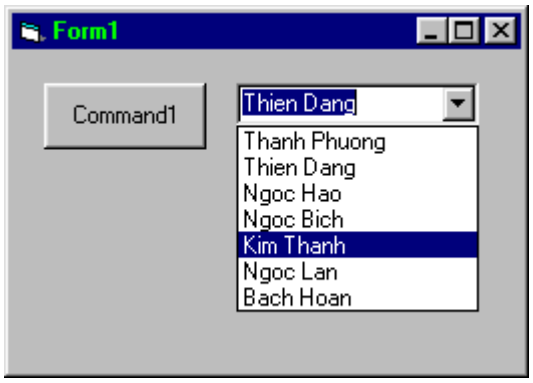
Để làm được việc này, bạn cần nhớ lại tổ hợp phím tắt để mở một combo box trong Windows là ALT - DownArrow (Mũi tên xuống). Do đó ta chỉ cần làm sao cho combo box nhận được tổ hợp phím ALT + DownArrow là êm chuyện.

Rất dễ dàng bạn hãy cho lệnh Sendkeys gởi 1 tổ hợp phím ALT - DownArrow từ thủ tục tình huống GotFocus của Combo box cần mở.

Hãy tạo 1 form mới trên đó tạo một control nào đó ví dụ như CommandButton, kế đến bạn hãy tạo 1 Combo box. Lý do bạn phải tạo Command button trước để cho nó có focus trước (TabIndex = 0), khi chạy chương trình bạn hãy nhấn phím Tab để chuyển focus sang cho combo box, lúc này bạn sẽ thấy rất rõ tác dụng của lệnh SendKeys.

```
Private Sub Combo1_GotFocus()

    SendKeys "%{DOWN}"

End Sub
```

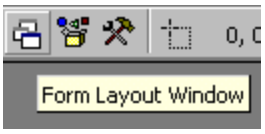
Câu lệnh SendKeys "%{DOWN}" dùng để gởi tổ hợp phím ALT - DownArrow lên Combo box khi chính nó nhận được Focus do người sử dụng dịch chuyển bằng phím Tab hay bằng Mouse. Sẽ làm cho combo box tự động mở ra (hiện danh sách).



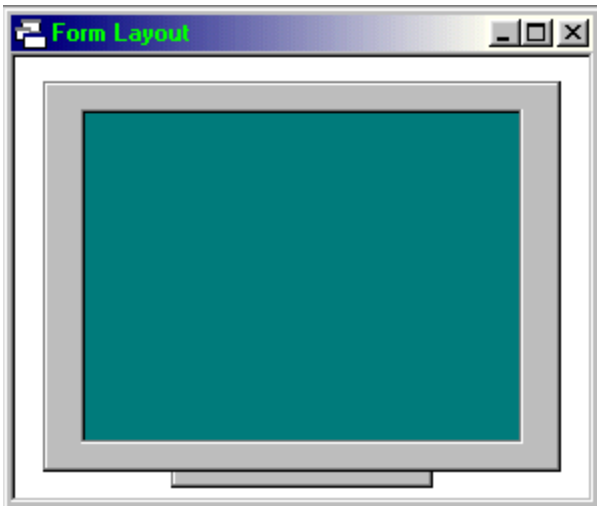
Thủ thuật này hữu ích khi bạn viết các ứng dụng có yêu cầu nhập dữ liệu, thường thì người dùng nhập dữ liệu bằng bàn phím, sau đó nhấn phím Tab đến một combo box, nó sẽ bung ra ngay để người dùng chọn lựa, rất tiện dụng, khiến cho người ta cảm thấy rất hài lòng về chương trình của mình.

Cửa sổ Form Layout dùng để làm gì ?

[Tự tạo Progress bar] [Combo box tự hiện danh sách]



Đây là một trong những tính năng mới của Visual Basic 5 & 6, giúp cho lập trình viên dễ dàng phân bố các form trên màn hình khi chương trình thực thi một cách rất trực quan. Vào View \ Form Layout Windows, hay click vào Icon trên toolbar để hiển thị cửa sổ Form Layout nếu chưa xuất hiện.



Để sử dụng các chức năng trên cửa sổ này, bạn hãy right click trên cửa sổ để bật menu Popup gồm các mục chọn sau:

Resolution Guides: Hiển thị độ phân giải màn hình (chỉ hiển thị các độ phân giải thấp hơn độ phân giải mà màn hình đang sử dụng).

Dockable: Hiển thị Form Layout bằng một cửa sổ riêng hay nằm chung với các cửa sổ Project, Properties.

Hide: đóng cửa sổ Form Layout.

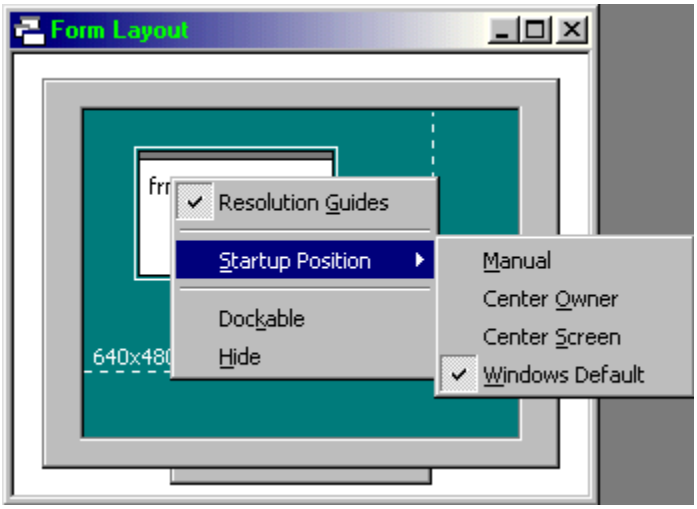
Startup Position: Vị trí form hiển thị trên màn hình

> Manual: Tự bạn thiết lập, bằng cách rê form có tên tương ứng, và dựa vào các Resolution Guides để biết rõ vị trí form sẽ xuất hiện trên màn hình khi chạy chương trình.

Center Owner: Nằm giữa form đã gọi nó.

Center Screen: Nằm giữa màn hình ở bất kỳ độ phân giải nào.

Windows Default: Theo chuẩn của Windows, nằm hơi chệch về góc trái trên của màn hình.



Mỗi form trong giai đoạn thiết kế, muốn hiển thị trên cửa sổ Form Layout bạn phải mở form đó lên, nó sẽ xuất hiện trên cửa sổ Form Layout bằng tên tương ứng, vị trí thực của nó sẽ xuất hiện trên màn hình lúc chạy được phản ảnh rất chính xác trong cửa sổ này.

Xưa nay người ta có quan niệm là ngôn ngữ lập trình Visual Basic "bị yếu" hơn các ngôn ngữ khác do không có các hàm, thủ tục can thiệp đến các thông tin cấp thấp của hệ thống như ngôn ngữ C. Nhưng cũng chính Visual Basic đã bù vào sự thiếu hụt đáng tiếc của mình bằng cách liên kết và sử dụng các hàm API (Application Programmer's Interface) có sẵn trong Windows qua câu lệnh Declare. Giúp cho ta làm được khối việc tưởng chừng như vô vọng. Thật ra các hàm API này được tạo ra bằng ngôn ngữ C do đó ta có thể tận dụng được các khả năng của C trong Visual Basic. Các hàm API tồn tại dưới dạng file có phần mở rộng là DLL hay EXE trong thư mục System. Ở các bài viết sau chúng ta sẽ cùng nhau tìm hiểu về các API tuyệt vời này. Một ứng dùng Visual Basic viết đàng hoàng, chạy ngon lành thì đố ai nhìn mà biết được bạn viết bằng ngôn ngữ nào (bạn lưu ý chỉ cho người ta nhìn thôi nhé).

Gọi Internet Explorer (IE)

Chức năng Wordwrap / Mẫu hàm API

Có rất nhiều cách gọi trình duyệt Internet Explorer của Windows. Ở đây tôi xin bày cho bạn 1 cách đơn giản nhất. Để chạy một chương trình khác từ Visual Basic, bạn buộc phải dùng hàm Shell(), đối số là gì, đó mới là điều quan trọng.

Thường IE nằm ở Program Files\Internet Explorer\IExplorer.exe thư mục Program Files nằm cùng ổ đĩa với thư mục Windows. Tóm lại, ta tìm thư mục Windows ở ổ đĩa nào, sau đó lấy 2 ký tự đầu tiên là tên ổ đĩa (Vi dụ C:), kế đến bạn hãy ghép với "Program Files\Internet Explorer\IExplorer.exe" để được đường dẫn đến IE rõ ràng. Tuy nhiên cách này sẽ "phá sản" nếu như người dùng "ngẫu hứng" cài IE ở một thư mục khác.

Có một cách khác hơi "mánh mung" một chút, nhưng đơn giản hơn và triệt để hơn rất nhiều. Bạn có thể "xí gạt" Windows để gọi IE thông qua Windows Explorer bằng cách chuyển cho nó một tham số là file htm, lúc này tự động Windows Explorer nhận biết và "alô" đến cho IE để vào thế chỗ. Vậy cũng xong chuyện, việc gọi Windows Explorer rất dễ dàng

Thí dụ: Cần gọi IE để hiển thị file "testfile.htm" bạn có thể dùng hàm Shell() như sau:

Shell("explorer.exe testfile.htm",vbNormalFocus)

Lưu ý: cách này đòi hỏi hệ thống bạn phải có Internet Explorer 4 trở lên.

Thật là đơn giản phải không bạn ?

Tao chức năng WordWrap bằng thuộc tính ScrollBars của TextBox

Gọi Internet Explorer / Mẫu hàm API

Trong các trình soạn thảo văn bản (như Notepad). Nếu chức năng Wordwrap được chọn, khi gõ văn bản đến giới hạn của cạnh phải của sổ, thì văn bản tự động rớt xuống dòng dưới. Ngược lại nếu không cho hiệu lực chức năng này, chỉ khi nào bạn nhấn phím Enter mới xuống dòng được.

Bạn có muốn tạo 1 tính năng Wordwrap như trong Notepad không ? Cũng dễ lắm. Chúng ta hãy lợi dụng thuộc tính ScrollBars của Textbox để làm tính năng Wordwrap.

Properties **ScrollBars**

Thuộc tính này dùng để quy định cho Textbox có các thanh cuộn hay không, nhưng thuộc tính này chỉ có tác dụng khi thuộc tính Multiline của Textbox = True.

0 - None: Textbox không có thanh cuộn.

1 - Horizontal: Chỉ có thanh cuộn ngang.

2 - Vertical: Chỉ có thanh cuộn đứng.

3 - Both: Có cả 2 thanh cuộn ngang và đứng.

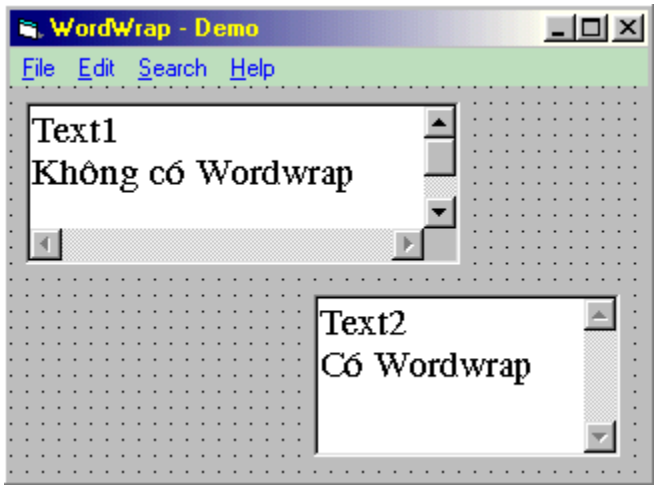
Tính năng Wordwrap chỉ có tác dụng khi thuộc tính ScrollBars là:

0 - None hoặc 2 - Vertical

Không có Wordwrap khi ScrollBars là:

1 - Horizontal hoặc 3 - Both

Nhưng thuộc tính này không thể thay đổi trong lúc chương trình thực thi. Chỉ cho phép thay đổi trong lúc thiết kế mà thôi. Do đó bạn phải tạo luôn 2 Textbox: Một Textbox không có Wordwrap (Text1), một Textbox có Wordwrap (Text2).



Trong lúc thiết kế bạn hãy cho 1 Textbox ẩn đi (Visible = False). Trong thí dụ minh họa này tôi cho Text2 ẩn.

Text1.Visible = True

Text2.Visible = False

Sau đó hãy tạo một Control nào đó để chuyển đổi qua lại giữa chế độ Wordwrap và không Wordwrap. Trong minh họa này tôi đã tạo luôn một trình Notepad, và đặt lệnh Wordwrap vào menu Edit của chương trình (Name: WWrap).

Sau đây là Code của một số Control cần thiết cho việc Demo chức năng Wordwrap.

Option Explicit

Private Sub Form_Resize()

Text1.Move 0, 0, ScaleWidth, ScaleHeight
Text2.Move 0, 0, ScaleWidth, ScaleHeight

End Sub

Điều chỉnh kích thước và vị trí của 2 Textbox, cho thích hợp với kích thước của Form mỗi khi người dùng thay đổi kích thước cửa sổ, hoặc chương trình khởi động. Xin bạn hãy nhớ một điều là: tình huống Form_Resize luôn luôn được triệu gọi mỗi khi Form_Load.

Private Sub mnuWWrap_Click()

```
mnuWWrap.Checked = Not (mnuWWrap.Checked)
If mnuWWrap.Checked = True Then
   Text1.Visible = False
   Text2.Visible = True
   Text2.SetFocus
Else
   Text1.Visible = True
   Text2.Visible = False
   Text1.SetFocus
End If
```

End Sub

Các lệnh cần xử lý khi người dùng chọn chức năng Wordwrap. Cho ẩn hiện Text1 hay Text2 tùy theo trạng thái (check hoặc không check) của menu WordWrap.

Private Sub Text1_Change()

```
    Text2.Text = Text1.Text
```

End Sub

Private Sub Text2_Change()

```
    Text1.Text = Text2.Text
```

End Sub

Tuy nhiên tại mỗi thời điểm, người dùng chỉ làm việc (gõ văn bản) trên một Textbox mà thôi, cho nên ta phải tiến hành cập nhật liên tục nội dung của 2 Textbox mỗi khi có 1 sự thay đổi nào đó trên bất cứ Textbox nào nhờ vào thủ tục tình huống Change.

Nhưng nếu chỉ bấy nhiêu đó thôi thì không ổn. Mỗi lần người dùng chuyển qua lại giữa Wordwrap và không Wordwrap tức là thay đổi Textbox, tự nhiên người dùng có cảm giác là lạ do con trỏ không nằm đúng vị trí quả thật là hơi "vô duyên". Để khắc phục nhược điểm trên bạn hãy thêm vài dòng Code sau đây vào chương trình đảm bảo người dùng không hề biết được là bạn đã đánh tráo Textbox của họ mỗi khi chọn chức năng Wordwrap.

Private Sub Text1_GotFocus()

```
    Text1.SelStart = Text2.SelStart
```

End Sub

Private Sub Text2_GotFocus()

Text2.SelStart = Text1.SelStart

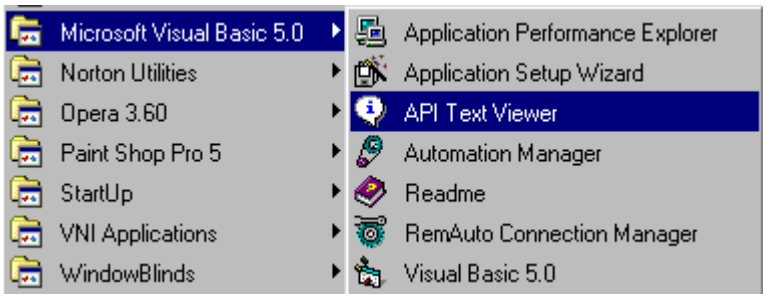
End Sub

Bây giờ bạn hãy chạy thử chương trình và gõ vào vài dòng văn bản, sau đó chọn lệnh Wordwrap vài lần xem, úi cha ! thật là tuyệt, chính bạn còn bị "lừa"nữa.

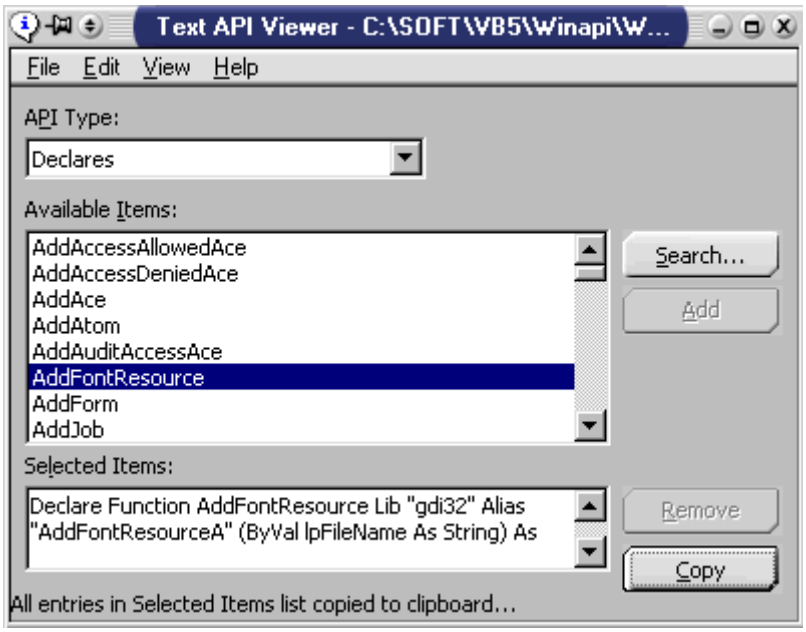
Lấy các mẫu khai báo hàm API

Gọi Internet Explorer / Chức năng Wordwrap

Để biết cách khai báo hàm API bạn cần khởi động tiện ích API Text Viewer được cung cấp kèm theo Visual Basic.



Nạp file Win32api.txt vào, chọn Declares trong phần API Type (có thể bạn phải chờ giây lát để chương trình load file này lên vì kích thước rất lớn). Hoặc bạn cũng có thể xem trực tiếp file này (dạng Text) bằng Wordpad.



Sau đó, bạn hãy chọn hàm trong danh sách Available Items, hàm nào cần dùng nhấn nút Add để thêm vào hộp văn bản Select Items phía đưới. Sau khi đã chọn xong tất cả các hàm cần

dùng, nhấn nút Copy để chép các hàm vừa chọn vào Clipboard. Kế đến bạn chỉ việc Paste vào Visual Basic để dùng mà thôi.

Các hàm API có 2 dạng: hàm (Function) có trị trả về và thủ tục (Sub) không có trị trả về.

Khai báo cho hàm có trị trả về như sau:

Declare Function <tên hàm API> Lib <Tên thư viện> [Alias <tên bí danh>]([danh sách các đối số])

Khai báo cho các thủ tục:

Declare Sub <tên hàm API> Lib <tên thư viện> [Alias <tên bí danh>]([danh sách các đối số])

<tên hàm API> là tên hàm trong các file thư viện DLL.

<tên thư viện> tên file thư viện DLL để Visual Basic tìm các hàm API. Các file thư viện này phải có đầy đủ tên cùng phần mở rộng, riêng đối với 3 thư viện USER, KERNEL, và GUI thì không cần phải có phần mở rộng. Tên này là một String nên cần phải bao trong dấu "".

[Alias <tên bí danh>] có thể có hay không cũng được. Bạn cần khai báo bí danh khi muốn triệu gọi hàm API với một cái tên khác do chính bạn đặt, hoặc trong tên hàm chuẩn có chứa ký tự bị cấm sử dụng trong Visual Basic, lúc này bạn hãy đặt bí danh cho nó để Visual Basic sử dụng được.

Ví dụ hàm API "AddfontResource " sau đây được đặt lại bí danh là AddFont cho ngắn gọn mỗi lần gọi hàm.

Declare Function AddFontResource Lib "gdi32" Alias "AddFont" (ByVal lpFileName As String) As Long

Phạm vi sử dụng của hàm API cũng phụ thuộc vào các vị trí khai báo nó như cách khai báo các biến trong Visual Basic

---

Thêm một đối tượng trong lúc chương trình thực thi

Ngoài các Control trong lúc thiết kế chương trình được bạn tạo ra, trong lúc chương trình đang thực thi (chạy) bạn vẫn có thể tạo thêm các Control một cách khá dễ dàng với điều kiện như sau để tạo nên một mảng các Control.

- Phải có tối thiểu 1 Control ban đầu

- Có thuộc tính Index = 0

Trong lúc chương trình chạy bạn có thể dùng câu lệnh sau đây để Load một Control lên. Khi Control được Load lên nó sẽ mang các thuộc tính giống hệt với cái ban đầu chỉ trừ chỉ số Index. Bạn hãy dùng lệnh sau để Load. Với Index là chỉ số của phần tử kế tiếp trong mảng

Load object(index)

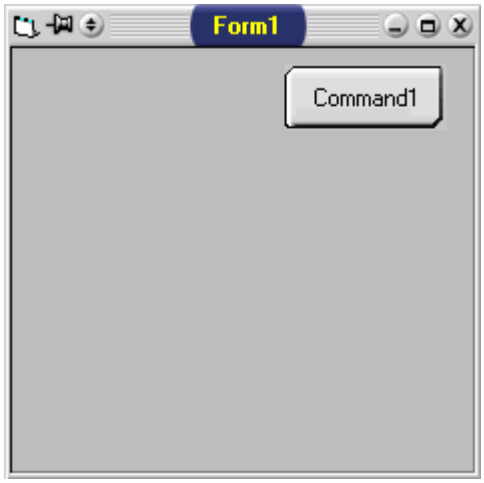
Visual Basic chỉ cho phép bạn Load đến 32767 (Giới hạn của Integer) phần tử trong một mảng mà thôi.

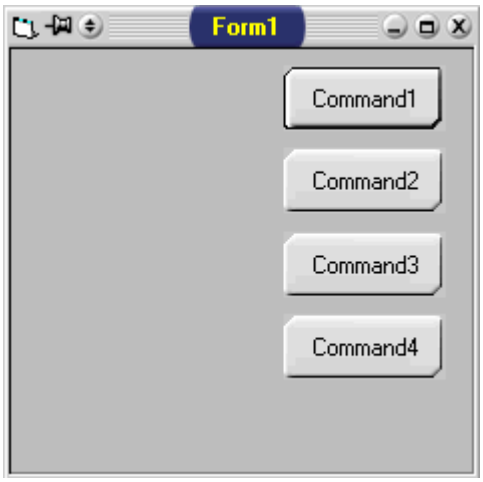
Sau đây là một thí dụ minh họa cách thêm CommandButton

- Bạn hãy tạo một CommandButton, đặt thuộc tính Name = cmdBtn và Index = 0 (zero). Khi bạn nhập vào thuộc tính Index của một con số cũng có nghĩa là bạn đã tạo một mảng các đối tượng đó.

- Paste vào đoạn Code sau:

```
Private Sub cmdBtn_Click(Index As Integer)

    Dim btn As CommandButton

    Dim iIndex As Integer

    iIndex = cmdBtn.Count

    If iIndex <= 32767 Then

       Load cmdBtn(iIndex)

       Set btn = cmdBtn(iIndex)

       With btn

          .Top = cmdBtn(iIndex - 1).Top + 620

          .Caption = "Command" & iIndex + 1

          .Visible = True

       End With

       Set btn = Nothing

    End If

End Sub
```



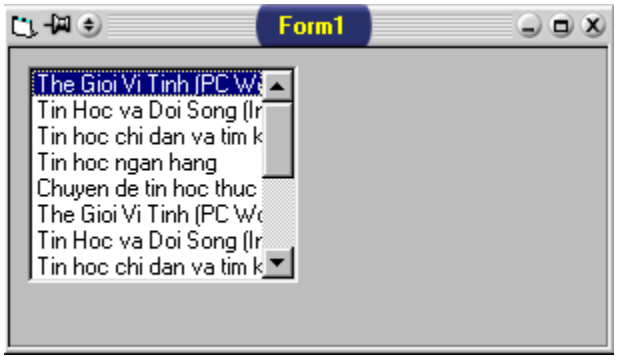


Nhấn F5 chạy thử chương trình, Click chuột vào CommandButton bạn sẽ thấy một CommandButton nữa xuất hiện phía dưới.

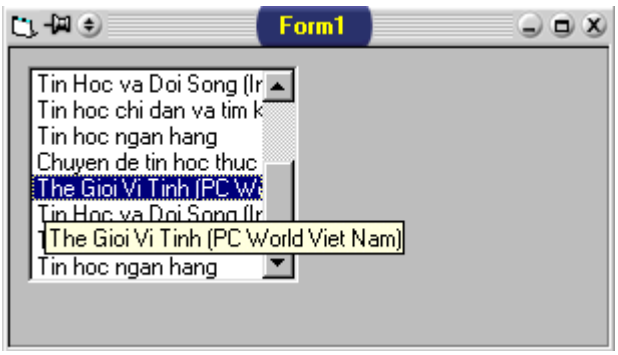
Nhờ ToolTip để hiển thị dữ liệu quá dài trong ListBox

Bạn có khi nào bạn gặp trường hợp Item cần hiển thị trong ListBox lại dài hơn bề rộng của Listox không ?

Trong rất dị hợm phải không ? Tự nhiên mất khúc đuôi của người ta.



Và đây là giải pháp. Bạn hãy dùng một ToolTip, nội dung của ToolTip chính là nội dung Item của ListBox mỗi khi bạn rê Mouse đến.



Hãy mở 1 form mới, trên đó tạo 1 ListBox (Name: List1). Tôi phải dùng đến hàm API SendMessage và hằng (LB_ITEMFROMPOINT = &H1A9) để làm việc này.

```
Option Explicit
Private Declare Function SendMessage Lib "user32" Alias "SendMessageA" _
(ByVal hwnd As Long,ByVal wMsg As Long, ByVal wParam As Long, _
lParam As Any) As Long
Private Const LB_ITEMFROMPOINT = &H1A9
```

Ở thủ tục tình huống Form_Load() bạn hãy thêm vào vài dòng lệnh để Add vào ListBox (Nhớ cho dòng dữ liệu dài dài một chút để thấy rõ tác dụng).

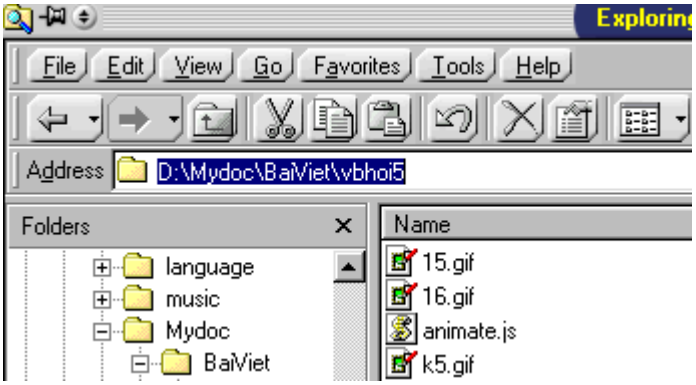
```
Private Sub Form_Load()
        List1.AddItem "Tao bo dia cai dat phan mem"
        List1.AddItem "Su dung Font Navigator"
        List1.AddItem "Tao man hinh Splash Screen"
        List1.AddItem "EditPlus (Text/HTML Editor for Windows)"
        List1.AddItem "Tu tao Progress bar cho ung dung"
        List1.AddItem "Combo box tu hien danh sach khi nhan focus"
        List1.AddItem "Cua so FormLayout dung de lam gi"
End Sub
```

'Khi rê mouse trên ListBox, thủ tục sau đây sẽ lấy nội dung của Item tại vị trí Mouse và cho hiện lên bằng ToolTip.

```
Private Sub List1_MouseMove(Button As Integer, Shift As Integer, _
X As Single, Y As Single)
   Dim lXPoint As Long
   Dim lYPoint As Long
   Dim lIndex As Long

   If Button = 0 Then 'Nếu không có nút nào được nhấn

      lXPoint = CLng(X / Screen.TwipsPerPixelX)
      lYPoint = CLng(Y / Screen.TwipsPerPixelY)

      With List1

         lIndex = SendMessage(.hwnd,LB_ITEMFROMPOINT,0, _
         ByVal ((lYPoint * 65536) + lXPoint))

        'Hiện ToolTip hoặc xóa cái trước đó

        If (lIndex >= 0) And (lIndex <= .ListCount) Then

          .ToolTipText = .List(lIndex)
        Else
          .ToolTipText = ""
        End If
      End With
   End If
End Sub
```

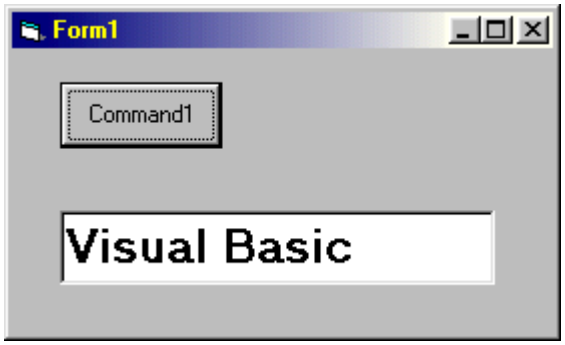
Làm sao để TextBox tự động Select văn bản mỗi khi nhận focus ?

Bạn có chú ý ở hầu hết các phần mềm, nếu người dùng cần nhập liệu vào một hộp văn bản (TextBox) nào đó, thì khi người dùng nhấn Mouse hoặc Tab để chuyển đến TextBox, tức thì toàn bộ dữ liệu đang hiện có trong TextBox sẽ được Select. Cách này giúp người dùng gõ dữ liệu khác đè lên dữ liệu hiện hữu, mà không phải mất công xóa đi rồi gõ lại cái khác.

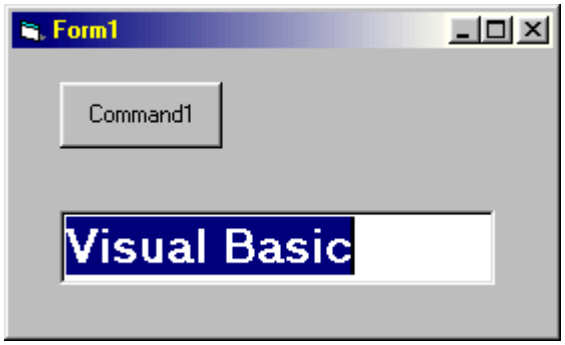


Viết một thủ tục riêng, thủ tục này nhận đối số là một TextBox

```
Public Sub FocusMe(ctlName As TextBox)
   With ctlName
      .SelStart = 0
      .SelLength = Len(ctlName)
   End With
End Sub
```




```
'Bây giờ bạn hãy gọi hàm này trong thủ tục tình huống GotFocus của một TextBox

Private Sub txtFocusMe_GotFocus()

   Call FocusMe(txtFocusMe)
End Sub
```



Hy vọng những câu hỏi nhỏ này sẽ giúp ích cho bạn trong lúc phát triển một ứng dụng bằng Visual Basic. Chúc bạn thành công.

Trong kỳ này:

---

1. Kiểm tra sự tồn tại của file ở bất kỳ đâu

2. Kiểm tra độ phân giải màn hình
3. Import file reg vào registry
4. Sự khác nhau khi load form ở chế độ MODAL & MODELESS
5. Khi nào, tại sao phải dùng Option Explicit
6. Tạo menu PopUp

Để giữ cho ứng dụng của bạn suôn sẻ khi chạy tránh trường hợp bị lỗi chương trình và ngừng một cách "bất hợp pháp", nếu người gặp lỗi là chính bạn thì không có gì, nhưng nếu người gặp lỗi không phải là bạn, bạn sẽ bị "mất mặt" vì chương trình của mình. Cụ thể hơn, trong quá trình làm việc với file, bạn cần phải luôn kiểm tra file có tồn tại hay không trước khi thực hiện một tác vụ ghi đọc nào đó. Dưới đây là một chương trình con dùng để kiểm tra, đối số duy nhất là đường dẫn file cần mở.

```
Public Sub VerifyFile(FileName As String)

    '

    On Error Resume Next

    'Mở file thử để kiểm tra

    Open FileName For I`165432nput As #1

    If Err Then
```

```
        MsgBox ("The file " & FileName & " cannot be found.")
        Exit Sub
    End If
    Close #1
'
End Sub
```

Bạn hãy đặt nó vào một module để sử dụng cho toàn chương trình.

Nếu quá trình kiểm tra diễn ra tốt đẹp thì không có gì, nếu có trục trặc bạn sẽ thấy một MsgBox xuất hiện, nhờ bẫy lỗi, chương trình của chúng ta vẫn hoạt động tiếp tục. Bây giờ bạn hãy tạo một form mới, thêm vào một CommandButton (Name: cmdVerify), gõ vào đoạn Code cho tình huống Click(), nhấn F5 chạy thử. Với chương trình con như vậy bạn có thể gọi nó để kiểm tra sự tồn tại của file trong chương trình.

```
Private Sub cmdVerify_Click()
    '
    Call VerifyFile("MyFile.txt")
    '
End Sub
```

---

2. Kiểm tra độ phân giải màn hình

Nếu chương trình của bạn có đòi hỏi phải chạy trong một độ phân giải màn hình nhất định nào đó, bạn có thể dùng cách sau đây để kiểm tra độ phân giải của màn hình có thích hợp với chương trình của mình không, rồi sau đó mới chạy.

Hàm CheckRez(rộng, cao) nhận vào 2 tham số là chiều rộng và cao của màn hình (tính bằng Pixel) trả về một giá trị kiểu Boolean (True/False). Nếu trả về True có nghĩa là độ phân giải màn hình đang sử dụng thích hợp với độ phân giải bạn cần kiểm tra.

Vd: CheckRez(800,600) = True

Do Visual Basic sử dụng đơn vị là Twip mà bạn lại sử dụng là Pixel cho nên phải có sự chuyển đổi đơn vị.

```
Public Function CheckRez(pixelWidth As Long, _
pixelHeight As Long) As Boolean

    '

    Dim lngTwipsX As Long

    Dim lngTwipsY As Long

    '

    ' chuyển đổi từ  pixels sang twips

    lngTwipsX = pixelWidth * 15

    lngTwipsY = pixelHeight * 15

    '

    ' kiểm tra lại độ rộng và cao của màn hình

    If lngTwipsX <> Screen.Width Then

        CheckRez = False

    Else

        If lngTwipsY <> Screen.Height Then

            CheckRez = False

        Else

            CheckRez = True

        End If

    End If

    '

End Function
```

Kế đến, bạn hãy dùng câu lệnh if cùng với điều kiện kiểm tra là hàm CheckRez ở đoạn mã bắt đầu của chương trình.

```
    If CheckRez(640, 480) = False Then

        MsgBox "Incorrect screen size!"

    Else
```

```
    MsgBox "Screen Resolution Matches!"

  End If
```

---

3. Inport một file Registry (.reg)*

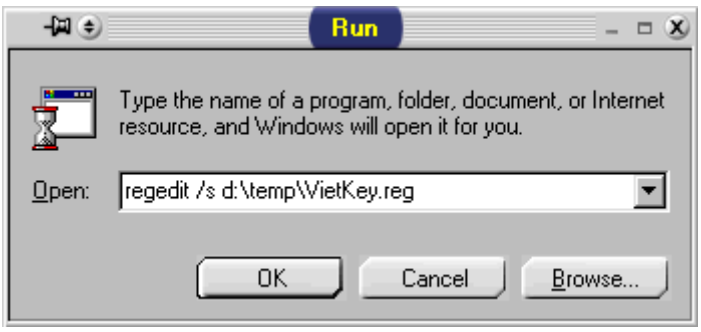
1. Kiểm tra sự tồn tại của file ở bất kỳ đâu
2. Kiểm tra độ phân giải màn hình
3. Sự khác nhau khi load form ở chế độ MODAL & MODELESS
4. Khi nào, tại sao phải dùng Option Explicit
5. Tạo menu PopUp

Nếu bạn cần Import một file reg vào Registry mà không muốn làm bận tâm đến người dùng. Bạn hãy chạy Regedit với thông số /s kèm theo sau là tên file và đường dẫn file reg cần Import vào Registry. Cụ thể như sau:

```
Dim strFile As String

Dim lngRet

strFile = App.Path & "\myreg.reg"

If Len(Dir$(strFile)) > 1 Then

  lngRet = Shell("Regedit.exe /s " & strFile, vbNormalFocus)

End If
```

Với cách này file reg của bạn sẽ được Merge vào Registry một cách âm thầm, không ai hay biết.

Và đương nhiên bạn cũng có thể gõ trực tiếp như vậy vào hộp thoại run của Windows.



---

4. Sự khác nhau khi load form ở chế độ MODAL và MODELESS

1. **Kiểm tra sự tồn tại của file ở bất kỳ đâu**
2. **Kiểm tra độ phân giải màn hình**

3. Import file reg vào registry
5. Khi nào, tại sao phải dùng Option Explicit
6. Tạo menu PopUp

Những Form được load ở chế độ MODAL thường có yêu cầu bắt buộc và chờ người dùng nhập dữ liệu trước khi thi hành các lệnh khác trong cùng một thủ tục (Sub / Function). Form loại này thường giữ focus của chương trình cho đến khi nó được người dùng "giải tán". Khi hiển thị form ở chế độ này, các lệnh không thuộc form đang hiển thị sẽ không được thực hiện, mà phải chờ cho đến khi form này được đóng lại. Các MsgBox, InputBox chính là một dạng của Modal form. Để hiển thị form ở chế độ này bạn dùng cú pháp lệnh sau:

MyForm.SHOW vbModal

Khi load form ở chế độ MODELESS các lệnh sẽ được lần lượt thực hiện một cách bình thường từ trên xuống dưới trong một thủ tục (không cần người dùng phải quan tâm, để từ từ rồi xem cũng được). các MDI child luôn ở dạng modeless. Để hiển thị form ở chế độ modeless bạn dùng lệnh:

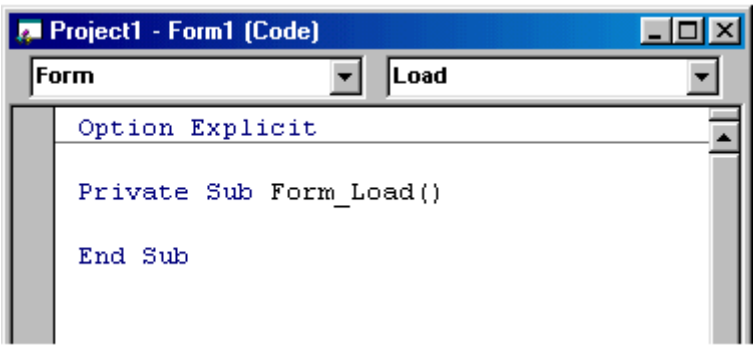
MyForm.SHOW

---

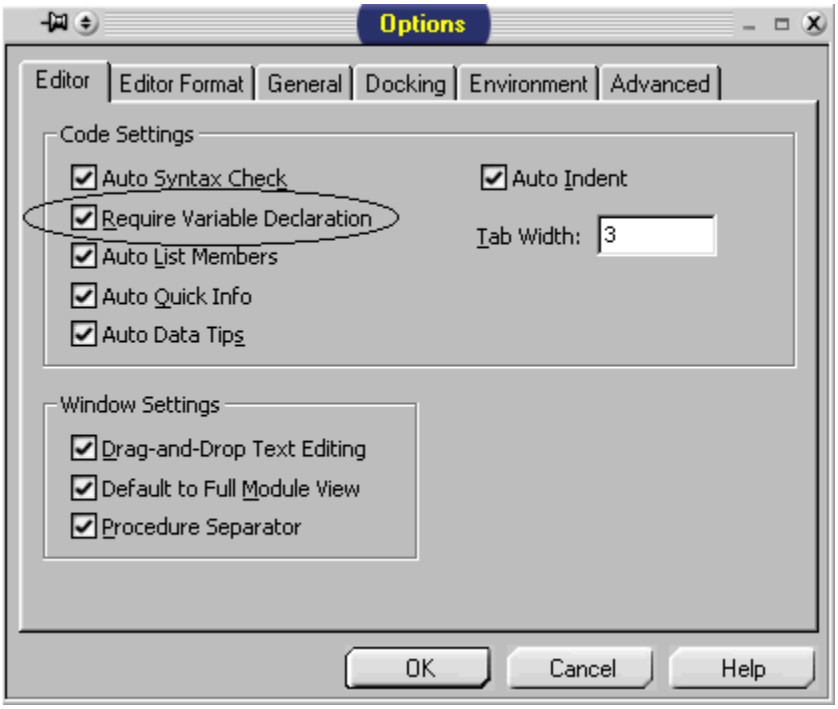
5. Tại sao phải dùng Option Explicit?

1. Kiểm tra sự tồn tại của file ở bất kỳ đâu
2. Kiểm tra độ phân giải màn hình
3. Import file reg vào registry
4. Sự khác nhau khi load form ở chế độ MODAL & MODELESS
5. Tạo menu PopUp

Nếu dòng Option Explicit xuất hiện trong cửa sổ code của chương trình cũng có nghĩa là Vb bắc buộc bạn phải khai báo biến một cách tường minh trước khi sử dụng chúng. Mệnh đề Option Explicit được đặt ở dòng đầu tiên trong cửa sổ Code (General Declaration). Quả thật việc bắt buộc phải khai báo biến sẽ làm cho chương trình rõ ràng, giảm thiểu tối đa các sai sót ngoài ý muốn (bugs) do khả năng tự phát sinh biến mới mà không cần khai báo của Visual Basic . Đây cũng là con dao 2 lưỡi, theo tôi tính năng này hại nhiều hơn là lợi.



Bạn có thể gõ trực tiếp mệnh đề Option Explicit hay nhờ VB tự động thêm giúp bằng cách vào Tools / Options. Check vào Require Variable Declaration trong thẻ (tab) Editor.



Option Explicit sẽ vô hiệu khả năng tự phát sinh biến của Vb. Như vậy, tất cả các biến muốn sử dụng đều phải khai báo bằng từ khóa DIM hay #800000IM, biến nào chưa được khai báo VB sẽ thông báo lỗi và bắt bạn phải khai báo mới chạy được chương trình. Kiểu dữ liệu mặc nhiên mà mỗi lần Visual Basic (Basic) tạo biến mới là Variant.

6. Tạo menu PopUp

(THIEN DANG)

1. Kiểm tra sự tồn tại của file ở bất kỳ đâu
2. Kiểm tra độ phân giải màn hình
3. Import file reg vào registry
4. Sự khác nhau khi load form ở chế độ MODAL & MODELESS
5. Khi nào, tại sao phải dùng Option Explicit

Trong ứng dụng ngoài loại menu kéo xuống (PullDown) còn một loại menu nữa khá linh động gọi là menu PopUp. Loại menu này bạn rất thường sử dụng trong Windows 9.x, được kích hoạt bằng phím phải chuột. Một ứng dụng Windows hoàn chỉnh chạy trong Windows 9.x không thể không có loại menu cấp tốc này.

Trong Visual Basic loại menu này thật ra cũng là một menu PullDown bình thường mà thôi. Khi nào cần hiện thành menu PopUp bạn chỉ việc gọi tên menu tương ứng (Name của Menu). Để cho thành menu PopUp thứ thiệt bạn hãy cho menu này ẩn đi (Visible = False), khi nào người dùng nhấn phím phải chuột bạn sẽ cho nó hiện ra bằng method **Popupmenu**

Trong chương trình ta có thể dùng hành vi (method) popupmenu để gọi một menu hiển thị, menu này phải được tạo trước (menu editor) và có ít nhất 1 mục chọn con tức là menu thứ cấp.

Cú pháp: **object.PopupMenu menuname, flags, x, y**

Trong đó:

object: tên form mà trình đơn sẽ xuất hiện. Nếu không ghi có nghĩa là dùng form hiện hành.

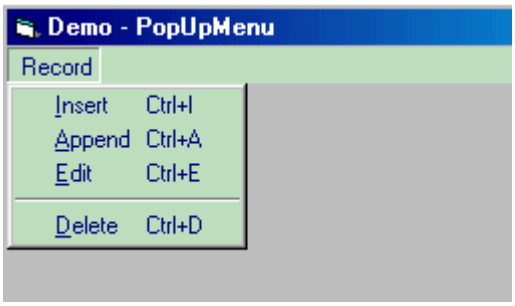
menuname: tên menu cần hiển thị.

flags: giá trị nguyên qui định vị trí xuất hiện của menupopup.

Tên Hằng	Giá Trị	ý nghĩa
vbPopupMenuLeftAlign	0	Trị mặc định, cạnh trái của trình đơn sẽ ở vị trí x.
vbPopupMenuCenterAlign	4	Trình đơn sẽ canh giữa so với vị trí x.
vbPopupMenuRightAlign	8	Cạnh phải của trình đơn sẽ ở vị trí x.
...	...	...
x,y		Tọa độ trình đơn sẽ xuất hiện. Nếu không ghi mặc nhiên trình đơn sẽ xuất hiện ở vị trí mouse

Sau đây là một thí dụ về PopUp menu:

Mở một form trống và tạo 1 hệ thống menu như hình sau. Trong hình menu chính là Record (Name: mnuRec) vào có một số menu con như: Insert, Append, Edit, Delete.



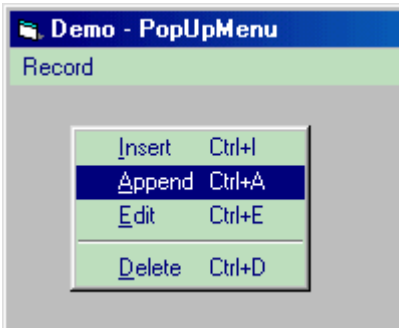
Để "chộp" được sự kiện nhấn Mouse bạn hãy khảo sát tình huống MouseDown của form này. Gõ vào đoạn Code sau:

Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)

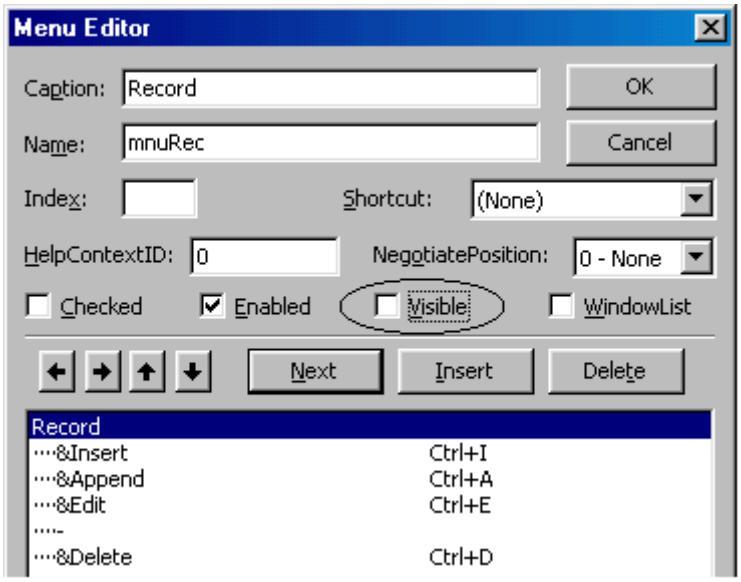
```
If Button = 2 Then
' Kiểm tra xem nút phải chuột có bị nhấn không

    PopupMenu mnuRec, 0
    ' Cho hiển thị menu mnuRec

End If
```

End Sub

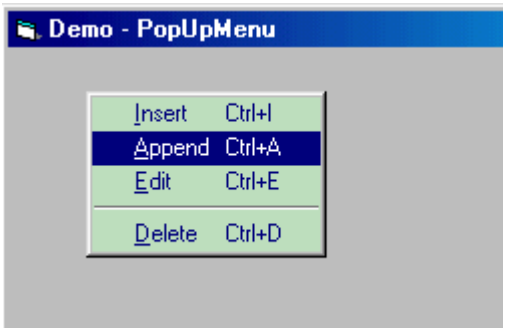
Nhấn F5 chạy thử bạn sẽ được như hình bên. Nhưng chúng ta lại không muốn cái menu này luôn xuất hiện sờ sờ trên form, hãy cho nó biến mất.



Bạn chỉ cần cho cái menu "đầu đàn" mnuRec biến mất là các menu con bên trong cũng mất tích theo. Hãy mở hộp thoại Menu Editor ra, click chọn menu Record, bỏ dấu check ở CheckBox Visible, click OK. Hà hà bây giờ nó biến mất tiêu rồi.



Nhấn F5, chạy thử chương trình, nhấn nút phải chuột... Bây giờ thì thành công rồi hén. Hãy áp dụng cách này vào ứng dụng Visual Basic của bạn để tiện cho người dùng.



Trong kỳ này:

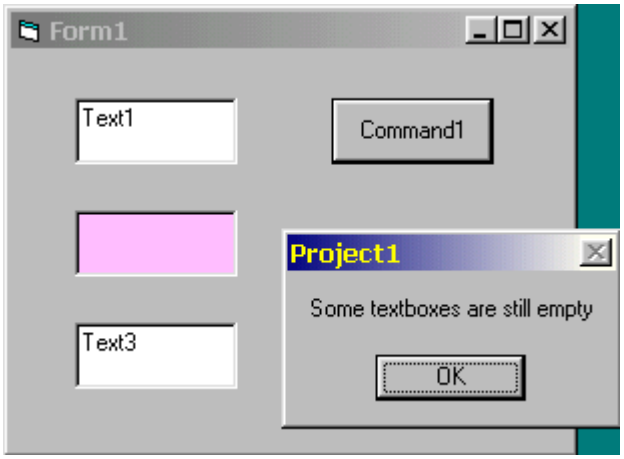
---

1. Kiểm tra việc nhập liệu vào TextBo

1. Kiểm tra việc nhập liệu vào TextBox

Trong chương trình nếu bạn có yêu cầu bắt người dùng phải nhập liệu vào mọi TextBox đang hiện hữu trên một form để thực hiện một việc nào đó, song không phải ai cũng "vâng lời". bạn cần kiểm tra xem người dùng có nhập đầy đủ thông tin vào các TextBox chưa. Để đỡ nhọc công phải kiểm tra từng cái một (thường kiểm tra khi có sự kiện LostFocus), với hàm này bạn chỉ việc chuyển cho nó một tham số là Form chứa các TextBox cần kiểm tra, nó có nhiệm vụ truy tìm những TextBox còn trống và phơi bày ra màn hình bằng bằng một biểu hiện khác thường nào đó.

Hàm này nhận vào một đối số duy nhất là Form (thường là form hiện hành "Me"), sau đó nó sẽ tìm đến tất cả các Textbox có trên form bằng cách sử dụng câu lệnh "For Each Control" nếu có Textbox còn trống thì đổi màu nền BackColor (màu hồng), đồng thời trả về một giá trị kiểu Boolean là True. Không những là TextBox, nếu là ComboBox vẫn bị vòng lặp này chiếu cố.



Để thử hàm này bạn hãy tạo một form, tạo vài chục cái TextBox (tên chi cũng được) và một CommandButton (name: cmdTextEmpty). Gõ vào đoạn Code sau:

```
Private Sub cmdTestEmpty_Click()
   If IsEmpty(Me) Then
      MsgBox "Some textboxes are still empty"
   End If
End Sub

Function IsEmpty(Frm As Form) As Boolean
Dim tmpControl As Control
On Error Resume Next
IsEmpty = False
For Each tmpControl In Frm.Controls
   If Trim(tmpControl.Text) = "" Then
      If Err.Number = 0 Then
         IsEmpty = True
         tmpControl.BackColor = &HFFC0FF 'Màu hồng
      End If
      Err.Clear
   Else
```

```
        If tmpControl.BackColor = &HFFC0FF Then
            tmpControl.BackColor = QBColor(15)'Màu trắng
        End If
    End If
Next tmpControl
End Function
```

Nhấn F5 để chạy chương trình. Bây giờ bạn hãy làm cho một TextBox trống (vài cái cũng được), sau đó click vào CommandButton, bạn sẽ thấy các TextBox trống bị đổi màu thành màu hồng. Hãy gõ văn bản vào các TextBox màu hồng đó, sau đó click lên CommandButton lần nữa, bây giờ thì êm rồi, các TextBox đã trở lại bình thường (nền trắng).

---

2. Chuyển đổi ký tự trong các đối tượng có khả năng hiệu chỉnh (Editbox) thành chữ hoa.

Các đối tượng loại này gồm có TextBox, Combobox. Sau đây là các đoạn code dùng để chuyển đổi văn bản hiện hữu trong các Control thành chữ hoa. Dùng cho một số đối tượng:

+ Cho combobox không có drop down

```
Dim hwndListbox As Integer
Dim childhWnd As Integer

hwndListbox = GetWindow(cbo1.hWnd, GW_CHILD)

childhWnd = GetWindow(hwndListbox, GW_HWNDNEXT)

lStyle = GetWindowLong(childhWnd, GWL_STYLE)

lStyle = lStyle Or ES_UPPERCASE

lRes = SetWindowLong(childhWnd, GWL_STYLE, lStyle)
```

+ Cho ComboBox có drop down

```
childhWnd = GetWindow(cbo1.hWnd, gw_child)

lStyle = GetWindowLong(childhWnd, GWL_STYLE)

lStyle = lStyle Or ES_UPPERCASE

lRes = SetWindowLong(childhWnd, GWL_STYLE, lStyle)
```

+ Cho TextBox

```
lStyle = GetWindowLong(Txt1.hWnd, GWL_STYLE)

lStyle = lStyle Or ES_UPPERCASE

lRes = SetWindowLong(Txt1.hWnd, GWL_STYLE, lStyle)
```

---

3. Bỏ qua một số ký tự khi gõ văn bản trong TextBox

Nếu bạn cần loại bỏ một số ký tự khi người dùng gõ văn bản vào trong một Textbox, thường là các ký tự đặc biệt như: "**!@#$%^&*()_+-=**" . Sau đây là một giải pháp. Bằng cách dùng liên tục hàm InStr mỗi khi có phím gõ (Sự kiện KeyPress)

```
Private Sub Text1_KeyPress(KeyAscii As Integer)

   Dim sTemplate As String

   sTemplate = "!@#$%^&*()_+-="

   If InStr(1, sTemplate, Chr(KeyAscii)) > 0 Then

      KeyAscii = 0

   End If

End Sub
```

---

4. Xóa các mục chọn (Item) trong Combo/List Box

Để xóa các Item trong Combo box hay List box, bạn cần phải duyệt ngược, tức là từ (ListCout-1) đến 0. Sau đây là một thí dụ minh họa. Bằng cách dùng vòng lặp For, kèm theo Step -1 để duyệt ngược.

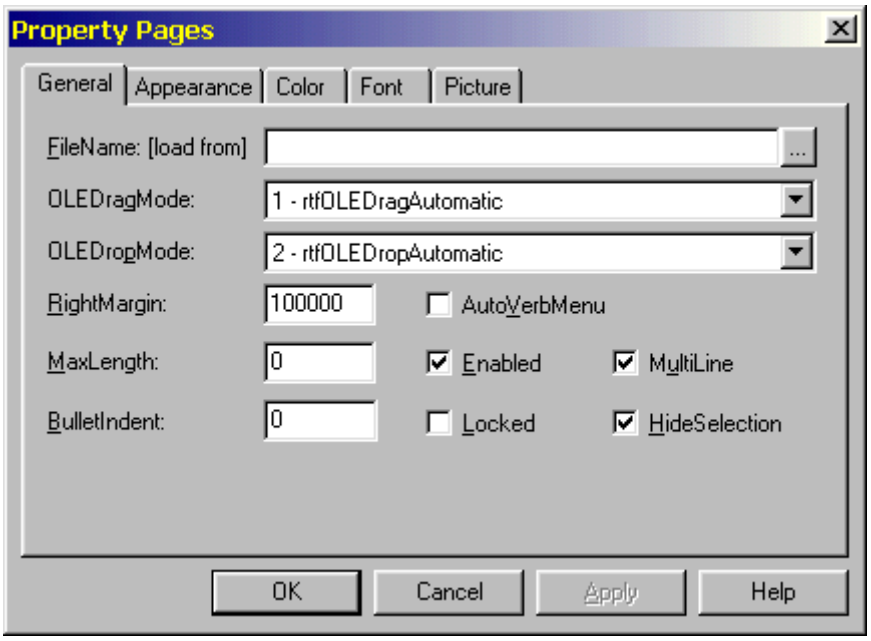
```
Sub cmdDeleteItems_Click ()

Dim i As Integer

   For i = List1.ListCount - 1 To 0 Step -1

      If List1.Selected(i) Then

         List1.RemoveItem i

      End If

   Next i

End Sub
```

---

5. Làm sao để tắt tính năng Wordwrap của Rich TextBox

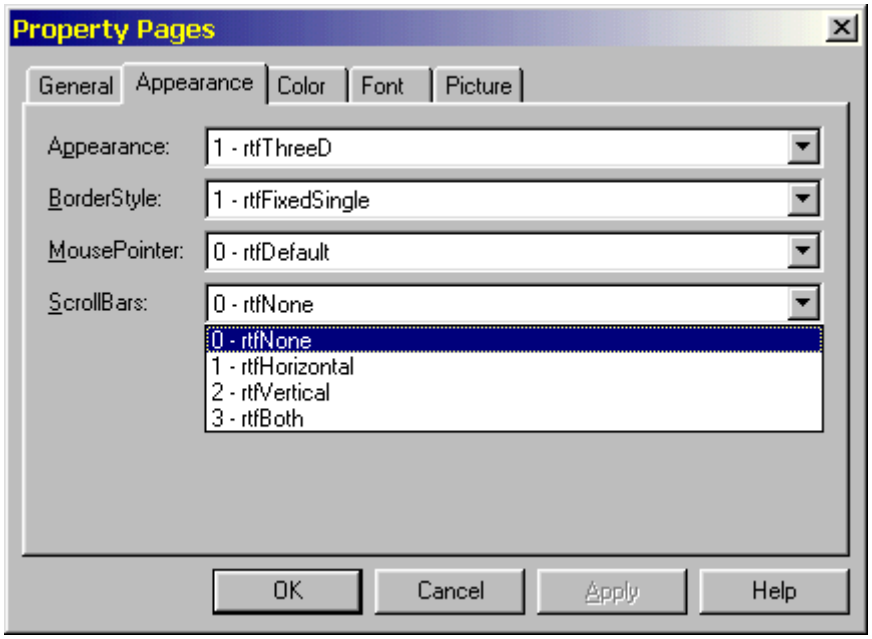
Có lần nào bạn bực mình vì không tìm được thuộc tính nào để tắt khả năng Wordwrap của RichTextBox không ? Tức là không cho nó tự cắt dòng văn bản của mình mỗi khi chạm cạnh phải. Tìm cũng không có, chắc Bill quên tạo rồi chăng?

Không hiểu tại sao khi sáng tác ra RichTextBox, Bill không chịu làm luôn cho nó chức năng Wordwrap để cho bà con có thể On/Off khi cần ? Không biết là Bill cố ý hay "lỡ quên" không tạo !?



Thật ra mà nói RichTextBox không có tính năng Wordwrap để dễ dàng sử dụng trên Internet đấy mà. Nhưng suy cho cùng chúng ta có thể nhanh chóng giải tỏa hạn chế này bằng cách gán cho thuộc tính **RightMargin** của RichTextBox một con số thiệt "bự", ôi chao đừng quá khiêm tốn với số 0 nữa ! Một con số chừng 100.000 cũng được mà, lúc đó thì khả năng Worwrap của nó bị đẩy đi tuốt luốt ở một nơi cách màn hình của bạn gần 400m về phía phải. Với mánh này bạn chỉ thật sự đại bại khi có dòng văn bản dài hơn cỡ đó thôi.

Bạn cũng đừng quên cho hiển thị ScrollBar để người dùng dễ dàng trong việc xem các dòng văn bản "cố ý dài" của bạn.



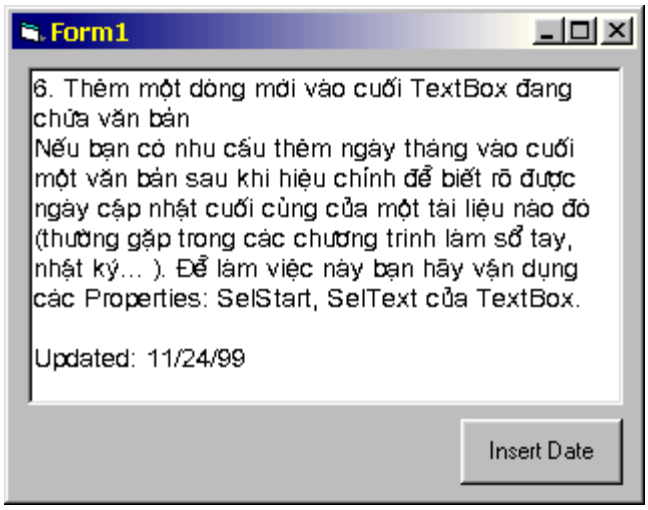
---

6. Thêm một dòng mới vào cuối TextBox đang chứa văn bản

Nếu bạn có nhu cầu thêm ngày tháng vào cuối một văn bản sau khi hiệu chỉnh, để biết rõ được ngày cập nhật cuối cùng của một tài liệu nào đó (thường gặp trong các chương trình làm sổ tay, nhật ký...). Để làm việc này bạn hãy vận dụng các Properties: SelStart, SelText của TextBox.

Sau đây là đoạn code thêm ngày tháng vào cuối văn bản trong TextBox

```
Dim strNewText As String

With Text1

   strNewText = "Updated: " & Date

   .SelStart = Len(.Text)

   .SelText = vbNewLine & strNewText

End With
```



---

7. Làm sao để Shut down hay Reboot lại Windows ?

Trong Visual Basic 32bit. Khai báo hàm API như sau:

```
Declare Function ExitWindowsEx Lib "user32" (ByVal uFlags As Long,
ByVal dwReserved As Long) As Boolean

Public Const EWX_SHUTDOWN = 1
```

Khi sử dụng bạn chỉ việc:

```
Dim success

success = ExitWindowsEx(EWX_SHUTDOWN, 0)
```

Nếu thành công, hàm này sẽ trả về True.

+ EWX_REBOOT = 2 sẽ làm cho Windows 9.x Reboot

+ EWX_LOGOFF = 0 để Log off.

Với phiên bản 16bit

```
Declare Function ExitWindows Lib "user" (ByVal wReturnCode as
Long,ByVal dwReserved as Integer) as Integer
```

Exit Windows:

```
RetVal% = ExitWindows(0, 0)
```

Exit & restart Windows:

```
RetVal% = ExitWindows(&H42, 0)
```

Exit Windows & restart the system:

```
RetVal% = ExitWindows(&H43, 0)
```

---

8. Làm cho TEXTBOX trở thành read only (cấm người dùng thay đổi nội dung)

Nếu bạn sử dụng Visual Basic 5.0 trở lên thì có thuộc tính Locked, nếu gán là True thì người dùng không thể thay đổi nội dung của TextBox (Read Only).

Có một số ý kiến cho rằng, bạn cần chộp lấy sự kiện KeyPress và KeyDown làm cho chúng thành zero. Tuy nhiên, giải pháp tốt nhất vẫn là hàm Windows API SendMessage.

```
Global Const WM_USER = &H400
Global Const EM_SETREADONLY = (WM_USER + 31)

Declare Function SendMessage Lib "User" (ByVal hWnd As Integer ByVal wMsg
As Integer, ByVal wParam As Integer, lParam As Any) As Long

SendMessage(Text1.hWnd, EM_SETREADONLY, 1, 0)
```

Cách trên người sử dụng vẫn có thể Copy nội dung. Nếu cần bạn thực sự cần cấm người dùng Copy nội dung, bạn có thể loại bỏ tổ hợp phím Ctrl-C bằng cách "đón đánh" khi xảy ra sự kiện KeyPress.

---

9. Mở file với chương trình liên kết (associated program)

Hàm Shell() hơi bị bất tiện vì chỉ gọi được chương trình trực tiếp từ file exe. Không giống như Windows, thí dụ như khi double click lên file DOC sẽ mở Microsoft Word, file TXT sẽ mở Notepad. Bạn cũng có thể viết một Module tạo các mối liên kết như vậy trong Visual Basic để dễ dàng trong việc gọi các ứng dụng.

Các khai báo API

```
#IF WIN32 THEN
Private Declare Function ShellExecute Lib "shell32.dll" Alias _ "ShellExecuteA"
ByVal hwnd As Long, ByVal lpOperation As String, _
ByVal  lpFile As String, ByVal lpParameters As String, _
ByVal lpDirectory As String, ByVal nShowCmd As Long) As Long

Private Declare Function GetDesktopWindow Lib "user32" () As Long

#ELSE
Declare Function ShellExecute Lib "SHELL" (ByVal hwnd%, _
ByVal lpszOp$, ByVal lpszFile$, ByVal lpszParams$, _
ByVal lpszDir$, ByVal fsShowCmd%) As Integer

Declare Function GetDesktopWindow Lib "USER" () As Integer
#END IF

Private Const SW_SHOWNORMAL = 1
```

Hàm phục vụ việc khởi động

```
Function StartDoc(DocName As String) As Long

   Dim Scr_hDC As Long

   Scr_hDC = GetDesktopWindow()

   StartDoc = ShellExecute(Scr_hDC, "Open", _
   DocName, "", "C:\", SW_SHOWNORMAL)

End Function
```

Kiểm chứng chương trình

```
Private Sub Form_Click()
```

```
    Dim r As Long

    r = StartDoc("c:\mydoc\myletter.doc")

    Debug.Print "Return code from Startdoc: "; r

End Sub
```

10. Làm sao để Play các file: MID, WAV ?

10. Làm sao để play các file: MID, WAV ?

Bằng cách sử dụng MCI của Windows

```
Declare Function mciExecute Lib "MMSystem" (ByVal FileName as String)
As Integer

Private Sub Form1_Click ()

    iResult = mciExecute("Play c:\windows\mkmyday.wav")

End Sub
```

Tương tự như vậy, bạn có thể dùng cách này để Play một file MIDI. Còn đây là cách Play một file AVI

```
Private Declare Function mciExecute Lib "WINMM.DLL" (ByVal lpstrCommand as String)
as Long

Private Sub Form_Click()

    iResult = mciExecute("Play E:\Luu\AVI\Search.avi")

End Sub
```

Nếu muốn ngừng, đơn giản bạn chỉ cần thay từ Play thành Stop là được.

Chúc bạn thành công

Cùng học lập trình
Visual Basic

Những câu hỏi về Visual Basic

---

1. Ghi các thiết đặt vào Registry

Chúng ta sẽ dùng hàm GetSetting & lệnh SaveSetting để thực hiện nhiệm vụ này. Vị trí ghi đọc giá trị của chúng trong Registry nằm ở: HKEY_CURRENT_USER\Software\VB and VBA Program Settings

* Hàm GetSetting: Dùng để đọc một trị từ Registry. Có cú pháp như sau:

```
GetSetting(appname, section, key[, default])
```

Trong đó:

AppName: Nơi mà hàm này sẽ tìm đến để đọc, thường nên đặt là tên của ứng dụng. Bắt buộc có.

Section: Trong một AppName bạn có thể tạo nhiều Section khác nhau, mỗi Section lưu trữ một nhóm thông tin có liên quan với nhau để dễ quản lý (giống như tạo thư mục con vậy). Bắt buộc có.

Key: Trong mỗi Section bạn có thể tạo nhiều key, mỗi key lưu 1 trị. Bắt buộc có.

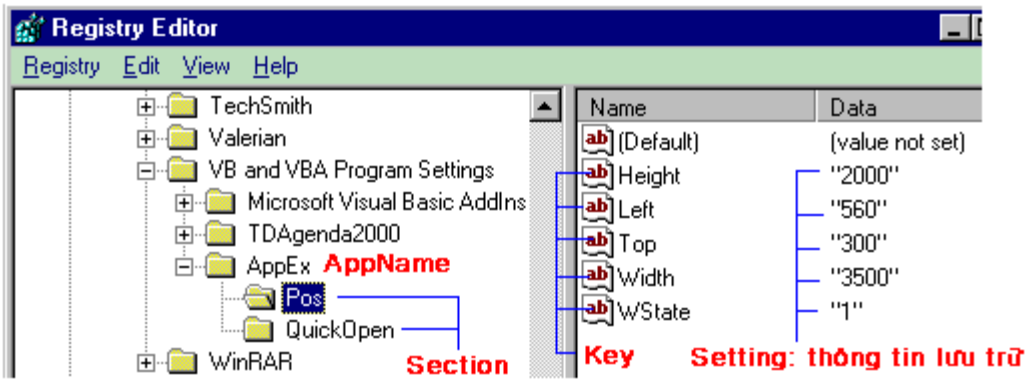
Default: Trị mặc nhiên mỗi khi không tìm thấy Key cần đọc trong Registry. Tham số này có thể có hay không cũng được. Nếu không có trị mặc nhiên sẽ là một chuỗi rỗng ("").

* Lệnh SaveSetting: Ghi thông tin vào Registry.

```
SaveSetting appname, section, key, setting
```

Các thông số tương tự như hàm GetSetting nhưng không có đối số Default. Nhưng lại có đối số Setting là trị sẽ được ghi vào Registry cho mỗi Key.

Bạn hãy xem hình để biết rõ hơn cách tổ chức các thông tin của Registry & cách ghi các thiết đặt của lệnh SaveSetting.



Các trị được lưu trong Registry

Thí dụ minh họa: Lưu trạng thái của form trước khi thoát

Trạng thái hiển thị của form bao gồm: Maximize, Minimize, Normal, Top, Left, Width, Height. Một vị trí thuận tiện nhất để lưu các thông số trên là Registry của Windows.

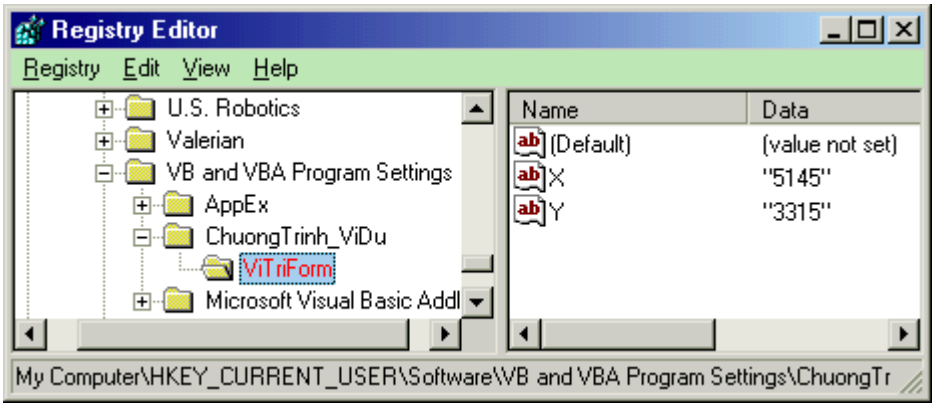
Vậy để giữ lại trạng thái của form thì khi form phát sinh sự kiện Unload bạn hãy ghi lại các thông số về trạng thái. Khi mở form bạn chỉ cần đọc lại các thông số đã lưu trữ. Vậy là ta có thể giữ lại được trạng thái form trước đó.

Để thí dụ tôi sẽ minh họa cách lưu lại vi trí của Form. Hãy khởi động VB & tạo 1 project mới, có 1 form tên là Form1, gõ vào code sau đây:

```
Option Explicit

Private Sub Form_Load()
   On Error Resume Next 'Dòng này hãy gõ sau.
   Me.Left = GetSetting("ChuongTrinh_ViDu", "ViTriForm", "X")
   Me.Top = GetSetting("ChuongTrinh_ViDu", "ViTriForm", "Y")
End Sub

Private Sub Form_Unload(Cancel As Integer)
   SaveSetting "ChuongTrinh_ViDu", "ViTriForm", "X", Me.Left
   SaveSetting "ChuongTrinh_ViDu", "ViTriForm", "Y", Me.Top
End Sub
```

Bây giờ bạn cho chạy thử chương trình, khi chương trình chạy bạn hãy thử di chuyển form đi nơi khác, dừng chương trình, sau đó cho chạy lại, bạn sẽ thấy form xuất hiện ngay đúng vị trí mà nó đã nằm trước khi thoát.

Vào Registry để kiểm nghiệm



Tuy nhiên khi thử chương trình bạn sẽ thấy báo lỗi, lý do là chưa có thông tin trong Registry để đọc vào 2 properties Left & Top khi chạy lần đầu.

Để khắc phục lỗi này bạn có thể dùng đến trị Default của hàm GetSetting. Riêng tôi, tôi chọn cách đặt thêm dòng On Error Resume Next vào dòng đầu tiên của sự kiện Form_Load.

2. Làm cho chương trình giống trang Web

Nếu bạn cảm thấy quá chán chường với mấy cái nút lệnh Command Button, xin hãy tạo cho chương trình của mình những siêu liên kết (Hyper Links) giống như trang Web. Cách làm cực kỳ dễ:

Hãy dùng một Label Box, đặt một Caption thích hợp, rồi sau đó mà tùy ý xử lý dựa trên cái sự kiện Mouse_Move của nó.

Ví du như khi rê mouse đến thì dòng Text trong Labelbox được gạch chân bạn làm như sau:

Tạo project mới, có Form tên là Form1, trên đó đặt một Labelbox tên là Label1. Mở cửa sổ code gõ vào đoạn code sau:

```
Option Explicit

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As      Single, Y As
Single)
    Label1.FontUnderline = False
End Sub

Private Sub Label1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As
Single)
    Label1.FontUnderline = True
End Sub
```

Qua đoạn code trên ta có nhận xét sau:

- Khi mouse được rê đến Label1 thì phát sinh sự kiện mouse_move trên Label này làm cho dòng Text của nó bị gạch chân (như là Hyper link thứ thiệt).

- Khi mouse còn trên form, luôn luôn xảy ra sự kiện mouse trên form và Label1 bị gán thuộc tính FontUnderline=False. Do đó ta sẽ thấy khi mouse rời khỏi Labelbox tức thì Text của nó hết bị gạch chân ngay.

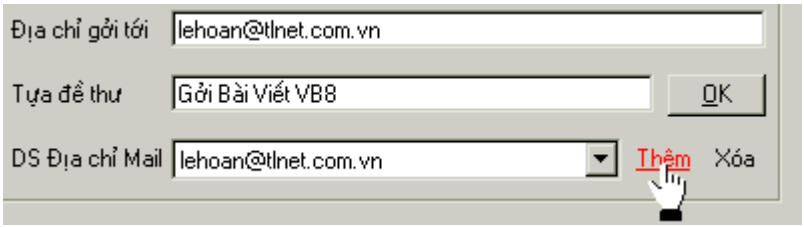
Khi mouse còn trên Form

Khi mouse được rê trên Labelbox

Tuy nhiên 1 Hyper link như vậy còn hơi nhạt nhẽo thiếu rất nhiều "hương sắc Web". Bạn có thể cho nó thêm vài thay đổi nữa như:

- Đậm lên (FontBold)
- In nghiêng (Font Italic)
- Đổi Font chữ (FontName)
- Đổi màu chữ (ForeColor)
- Đổi màu nền (BackColor)
- Đổi biểu tượng chuột (Mouse Icon). Nhớ đặt MousePointer là Custom - 99

Còn rất nhiều thứ xin mời bạn.



3. Làm sao để mở trình Browser & Mail mặc định của Windows ?

Bạn khai báo hàm API sau:

```
Private Declare Function ShellExecute Lib "shell32.dll" Alias _
"ShellExecuteA" (ByVal hwnd As Long, ByVal lpOperation As _
String, ByVal lpFile As String, ByVal lpParameters As String, _
ByVal lpDirectory As String, ByVal nShowCmd As Long) As Long
```

Khi nào cần dùng chỉ việc gọi hàm

```
Dim ret&
ret& = ShellExecute(Me.hwnd, "Open", "http://ilpweb.itgo.com", "", App.Path, 1)
```

Nếu muốn gọi Mail bạn chỉ cần thay địa chỉ Web thành "mailto:email@site.com" là được rồi

Chúc bạn thành công

4. Đường dẫn của đối tượng DirListbox và App

Hãy hãy lưu ý thuộc tính Path của 2 đối tượng này. Thí dụ như bạn cần tham chiếu đến một tập tin tên là Test.txt nằm trong thư mục của chương trình (giả sử App.path == c:\myapp).

```
app.path & "\Test.txt" ==> c:\myapp\Test.txt
```

Nhưng nếu App.path của bạn là một thư mục gốc của bất kỳ ổ đĩa nào, ví dụ như c:\ thì nó lại thành.

```
app.path & "\Test.txt" ==> c:\\Test.txt
```

Chương trình sẽ báo lỗi ngay lập tức. Tương tự như vậy đối tượng DirListBox cũng bị trường hợp này.

Lý do là 2 đối tượng này trả về cho thuộc tính Path các trường hợp sau:

- Nếu không phải là thư mục gốc thì: **Tên ổ đĩa:\Tên thư mục** - ví dụ **C:\TEMP**
- Nhưng nếu là thư mục gốc thì lại **Tên ổ đĩa:** - vi dụ **C:**

Khi lập trình tổng quát ta hay viết là

```
App.path & "\Test.txt"

Dir1.path & "\" & File1.Filename
```

Như vậy ta thấy dư 1 dấu \ khi là thư mục gốc. Để giải quyết lỗi này bạn có thể dùng hàm IIF() hoặc câu lệnh IF

Cụ thể tôi có thể viết như sau:

```
Dim F as String

F = Dir1.path & IIF(Len(Dir1.path)=3,""," ") & File1.FileName
```

Hay

```
If Len(Dir1=3) then

   F = Dir1.path & File1.FileName

else

   F = Dir1.path & "\" & File1.FileName

End If
```

Chúc bạn thành công.

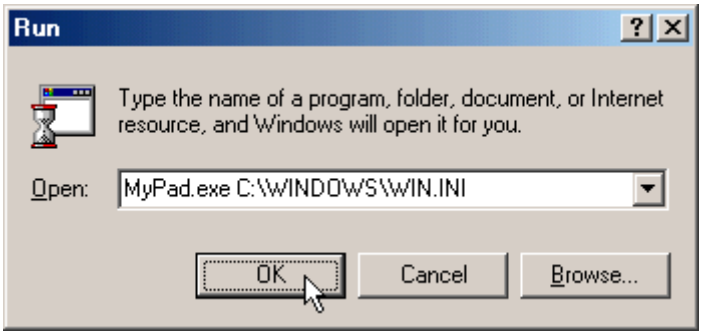
Những câu hỏi về Visual Basic

1. Lấy tham số truyền từ dòng lệnh

VB cung cấp cho ta hàm Command() hay Command$(). Hàm này trả về một String mà bạn truyền vào từ dòng lệnh khi gọi chương trình.

Lưu ý: Hàm Command$() mới thực sự trả về một String còn Command() lại trả về một Variant

Ví dụ: Hình bên dưới cho thấy ta khởi động chương trình MyPad.exe & truyền cho nó một tham số là C:\WINDOWS\WIN.INI



Bây giờ chúng ta hãy thử lấy tham số đó qua một chương trình viết bằng VB.

Ta sẽ viết một chương trình tên MYPAD, có một form, trên đó có một TextBox. Nếu khi chạy chương trình người dùng không truyền tham số thì để TextBox đó trống (hoặc chứa một thông báo). Còn ngược lại thì kiểm trả xem đó có phải là đường dẫn đến một tập tin hay không (tập tin văn bản), nếu đúng thì hiển thị nội dung file đó vào TextBox.

Đối với TextBox bạn phải đặt thuộc tính Multiline = TRUE. ScrollBars là 2-Vertical hay 3-Both

Double Click lên form để tạo code cho sự kiện FormLoad như sau:

```
Private Sub Form_Load()

    Dim sFile As String

    Dim nd As String, dong As String

    nd = "" : dong = ""

    sFile = Command$()

    If Dir$(sFile) <> "" And sFile <> "" Then
```

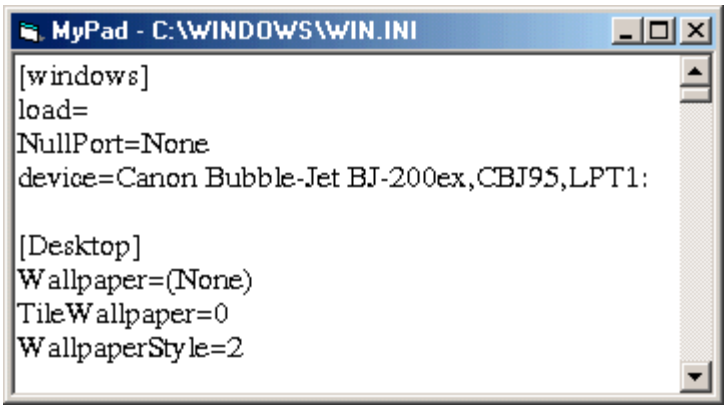
```
        Open sFile For Input As #1
        Do While Not EOF(1)
           Line Input #1, dong
           nd = nd + dong + vbCrLf
        Loop
        Close #1
        Text1 = nd
        Caption = "MyPad - " & sFile
    Else
        Text1 = "Tập tin: [" & sFile & "] không tìm thấy." & vbCrLf & "Hoặc không cho đường dẫn
file."
    End If
End Sub
```

Để cho Textbox (Text1) lúc nào cũng có kích thước bằng với form, bạn hãy gõ code sau vào sự kiện FormResize.

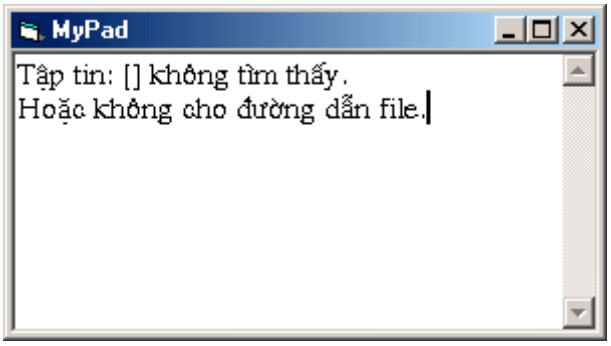
```
Private Sub Form_Resize()
    Text1.Left = 0 : Text1.Top = 0
    Text1.Width = ScaleWidth : Text1.Height = ScaleHeight
End Sub
```

Bây giờ bạn hãy cho chạy thử chương trình & truyền cho nó một tham số để mở file WIN.INI

Click Start / Run & gõ vào D:\MYPAD\MyPad.exe C:\WINDOWS\WIN.INI (Giả sử chương trình MyPad của tôi lưu trên D:\MYPAD), sau đó click OK. Chương trình MyPad của chúng ta chạy & hiển thị nội dung của WIN.INI trong Textbox.



Nếu bạn cho chạy MYPAD mà không cho tham số hoặc cho sai chương trình sẽ hiển thị như sau:



Vì đây chỉ là một chương trình thí dụ đơn giản minh cho cách nhận tham số từ dòng lệnh, cho nên bạn phải chú ý là file (đường dẫn file) mà bạn truyền cho nó không được có thuộc tính ẩn, nếu không nó sẽ báo là không tìm thấy.

Về đầu trang

2. Lấy các biểu tượng chuẩn của cửa sổ trong Windows

Nếu bạn có nhu cầu tự viết lấy một cửa sổ cho mình thì chắc chắn bạn sẽ cần các ký tự sau đây.

Windows dùng font tên là Marlett để thể hiện các ký tự này. Muốn thể hiện các ký tự này bạn gõ các ký tự từ 0 đến 9 & từ a đến y, sau đó chọn font là Marlett. Từ đó bạn có thể tìm thấy biểu tượng mình cần.

Chương trình sau đây sẽ thể hiện các biểu tượng đó:

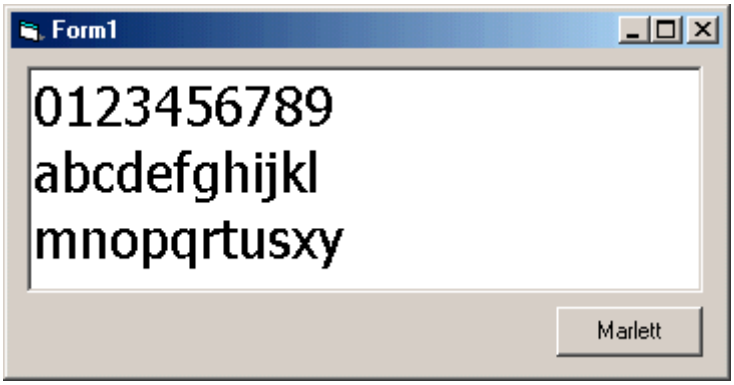
- Bạn hãy tạo 1 Project mới

- Tạo một Textbox (Text1) & một CommandButton (Command1) đặt Caption cho nút này là "Marlett"

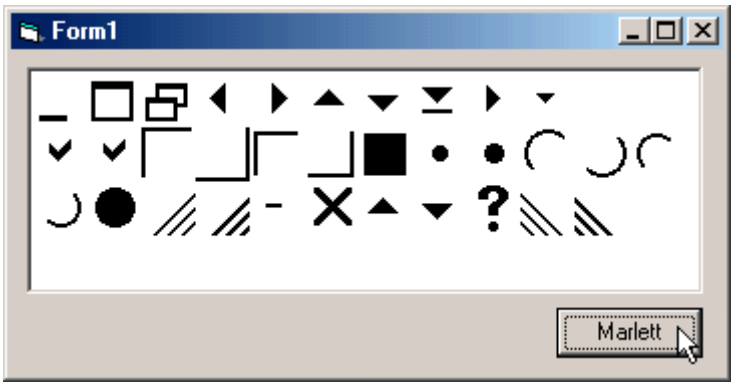
- Nhập đoạn Code sau đây vào Form1 rồi chạy thử

```
Option Explicit

Private Sub Form_Load()

    Text1.FontName= "Tahoma"

    Text1.FontSize = 20

    Text1.Text = "0123456789" & vbCrLf & _

    "abcdefghijkl" & vbCrLf & "mnopqrtusxy"

End Sub

Private Sub Command1_Click()

    Text1.FontName = "Marlett"

End Sub
```

Khi chương trình chạy bạn sẽ thấy các ký tự xuất hiện bình thường với font Tahoma.



Nhưng khi click vào nút Marlett thì kết quả như hình sau đây.



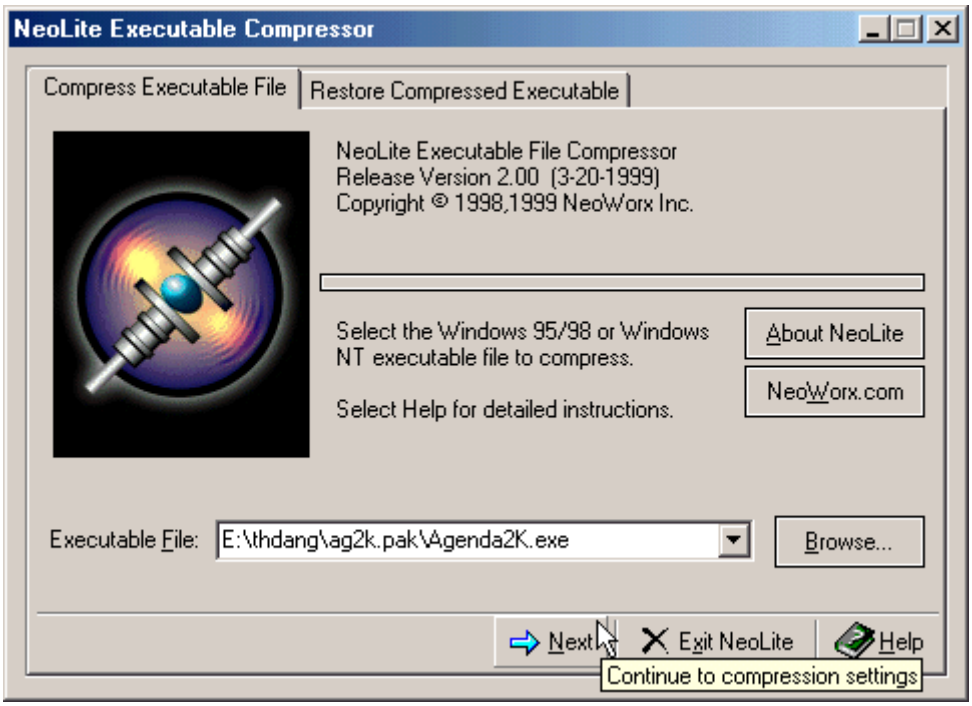
Rất tuyệt phải không bạn, như vậy nếu cần biểu tượng nào bạn có thể dùng ngay mà không phải mất công vẽ lại.

Về đầu trang

3. Thu nhỏ kích thước tập tin exe sau khi dịch chương trình.

Sau khi dịch chương trình thành file exe nếu thấy file này hơi lớn, bạn có thể dùng chương trình NeoLite để nén nó lại cho nhỏ bớt. Chương trình có thể nén các file sau:

NeoLite dùng để nén các chương trình 32 bit của Windows (.EXE, .DLL và .OCX). Sau khi đã nén chương trình này vẫn chạy bình thường như khi chưa nén, người dùng khó mà phát hiện được sự thay đổi do tốc độ của chương trình gần như không hề suy giãm. Chỉ có 1 sự khác biệt duy nhất là file đã nén nhỏ chỉ bằng 60% lúc đầu do đó tiết kiệm được một lượng đáng kể khoảng trống của chương trình trước khi phân phối đến người dùng.



- NeoLite có 2 mức nén: nén có khả năng phục hồi & nén vĩnh viễn (không phục hồi được, tùy chọn này cho tỷ lệ nén cao hơn).

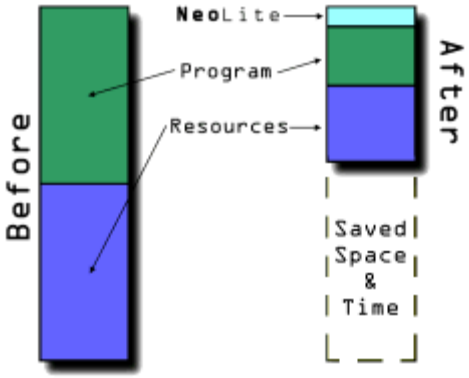
- Không những thu nhỏ kích thước mà việc nén có thể bảo vệ cho chương trình, tránh được việc dịch ngược (de-compilers, dis-assemblers).

- Không có gì đảm bảo file sau khi nén còn chạy tốt trăm phần trăm, cho nên bạn phải cẩn thận khi sử dụng, đặc biệt là phải sao lưu trước khi nén.

* Cơ chế nén của NeoLite:

Khi NeoLite nén một chương trình nó sẽ ghép một module đặc biệt của chính mình với module của file chương trình đã được nén, module đặc biệt đó của NeoLite được gọi là loader program (xem hình).

Khi file chương trình này được triệu gọi, loader program sẽ chạy trước và mở nén module của chương trình đã nén trực tiếp vào bộ nhớ và cho thực thi như một chương trình bình thường.



Mọi sự ta nhìn thì vẫn cứ tưởng diễn ra bình thường, nhưng thật ra đằng sau hậu trường êm ái ấy, chương trình còn phải tốn chút thời gian để tự mở nén chính nó trước khi nạp vào bộ nhớ, khoảng thời gian này lớn hay nhỏ là còn tùy thuộc vào kích thước thực của file chương trình & tốc độ xử lý của máy tính.

Cùng học lập trình VISUAL BASIC

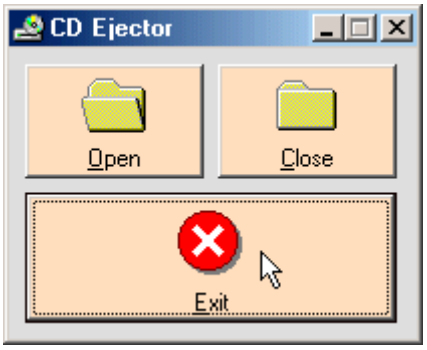
Bai viet su dung Font UNICODE

NỘI DUNG

Đóng mở khay đĩa CD-Rom

Bạn có muốn viết một tiện ích để đóng mở khay ổ đĩa CD-Rom cho riêng mình không ? Nếu bạn có ý đó thì vài dòng code sau đây sẽ giúp bạn toại nguyện.

Lưu ý: Chương trình này chỉ tác dụng tới ổ CD đầu tiên trên hệ thống của bạn (ổ có tên gần với tên Partition cuối cùng của máy).



- Tạo một Project mới.

BACK | FORWARD

- Khai báo hàm API sau trong Form1.

Option Explicit

Private Declare Function mciSendString Lib "winmm.dll" Alias "mciSendStringA" (ByVal lpstrCommand As String, ByVal lpstrReturnString As String, ByVal uReturnLength As Long, ByVal hWndCallback As Long) As Long

Tạo thêm hàm vbmciSendString() để nhận thông điệp đóng/mở khay CDROM. Hàm này trả về một String.

Function vbmciSendString(ByVal Command As String, ByVal hWnd As Long) As String

Dim Buffer As String

Dim dwRet As Long

Buffer = Space$(100)

dwRet = mciSendString(Command, ByVal Buffer, Len(Buffer),hWnd)

vbmciSendString = Buffer

End Function

Tạo lần lượt 2 Command Button và đặt Caption cho chúng là: "Open" & "Close". Sau đây là mã tương ứng cho 2 nút lệnh đó.

Private Sub Command1_Click()

Dim Dummy As String

Dummy = vbmciSendString("set cdaudio door open", 0)

End Sub

Private Sub Command2_Click()

Dim Dummy As String

Dummy = vbmciSendString("set cdaudio door closed ", 0)

End Sub

Như vậy Command1 dùng để mở khay CD, còn Command2 thì đóng khay. Hãy save lại & cho chạy thử chương trình xem. Hãy áp dụng những thủ thuật của các bài trước thiết kế cho chương trình chạy thường trú trong Windows, thể hiện thành 1 Icon ở System Tray để tiện sử dụng.

Nhất định nó sẽ hữu ích khi bạn làm biếng thò tay nhấn nút Eject hoặc cái nút quái quỷ đó không còn tác dụng. Riêng tôi cái tiện ích này cực kỳ tiện, do cái thùng máy (CPU) tôi đặt dưới gầm bàn, muốn lấy CD ra phải khom người nhấn nút, rồi chờ (một hai giây thôi cũng đủ làm tôi nóng ruột). Với tiện ích chỉ việc click chuột khi nào nghe tiếng khay bung ra .... thật tiện lợi.

Đầu trang

Tùy biến Command Button

Command Button là 1 trong những control thông dụng nhất trong Windows nói chung & trong VB nói riêng. Một chương trình đơn giản nào hầu nhưng cũng có đối tượng này.

Tuy vậy nó cũng có những vướng mắt chứ chẳng phải đơn giản đâu. Sau đây là vài thắc mắc của một bạn tự học VB. Tôi muốn phổ biến cùng các bạn đang tự học VB khác.

Câu hỏi 1: Tại sao khi tôi thay đổi màu BackColor của Command Button thành một màu khác (màu mặc nhiên của Windows) nhưng nó vẫn trơ như đá, chẳng thấy thay đổi theo sự lựa chọn của tôi.

Trả lời: Command Button chỉ thay đổi màu Backcolor khi thuộc tính Style của nó được đặt là 1-Graphical mà thôi. Mặc nhiên là 0-Standard.

Điều này cũng tương tự như thuộc tính Picture, khi bạn gán một hình ảnh cũng phải gán cho thuộc tính Style là 1-Graphical thì bức ảnh mới hiện lên, nếu không thì chẳng thấy thay đổi gì.

Câu hỏi 2: Làm sao để thay đổi màu TextColor (Màu của dòng Text trên Command Button - Caption) của Command Button ?

Trả lời: Trong chế độ Design của VB không có thuộc tính nào cho phép ta thay đổi màu TextColor, và khi thi hành chương trình cũng vậy. Nói tóm lại trong môi trường phát triển của VB bạn không thể nào thay đổi màu TextColor của Command Button được.

Do vậy ta phải tìm một hướng khác. Tôi xin đề nghị 2 cách sau đây:

* Custom Control: Bạn có thể tự thiết kế, điều này đòi hỏi bạn phải có khả năng tạo ActiveX (cũng tạo bằng VB). Hay bạn có thể tìm những ActiveX được thiết kế sẵn, về bổ sung vào ứng dụng của mình (Internet là nơi chắc chắn bạn phải tìm đến, nếu không có điều kiện, hãy lục lọi trên các CD free code bán ở các cửa hàng).

Nhưng theo tôi biết, trong bộ VB 4 (Version 32bit) có một custom control tên là Sheridan 3D Controls (THREED32.OCX) bạn có thể dễ dàng tìm tìm thấy hơn. Bao gồm các đối tượng sau:

- SSCheck
- SSFrame
- SSPanel
- SSRibon
- SSCommand
- và SSOption

Bao gồm rất nhiều thuộc tính cho phép ta chọn đủ màu sắc, có cả thuộc tính cho phép gán hiệu ứng cho Text nữa. Thật lý thú.

Xem thử giao diện

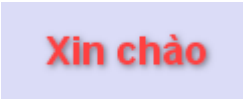
* Tự vẽ lấy Command Button: Cách này thì ai cũng có thể làm, nhưng đẹp hay xấu là do phụ thuộc vào khả năng vẽ của bạn. Cách làm như sau:

- Tự bạn vẽ một nút lệnh, hình dáng kích thước màu sắc tùy ý bạn. Bằng các chương trình đồ họa (hay dùng ngay Paint của Windows cũng

được).

- Lưu lại bức ảnh dưới dạng Bitmap (*.bmp). Thật ra với VB6 bạn có thể dùng format: bmp, jpg, gif ... Còn VB4 chỉ dùng bmp mà thôi.
- Gán vào thuộc tính Picture của command button. Lưu ý: Bạn phải đặt cho thuộc tính Style là 1-Graphical & Caption là trống (không đặt gì cả).

Tôi có làm một thí dụ đơn giản để minh họa



Bức ảnh gif được vẽ bằng các chương trình vẽ thông dụng.

Button thứ nhất là do tôi vẽ, button thứ hai là của VB (không thể thay đổi màu cho Text được). Bạn hãy so sánh.

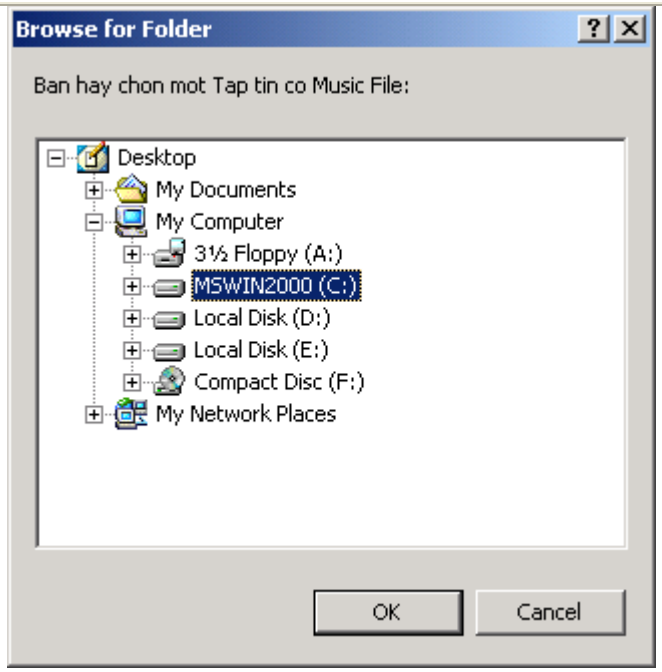
Có người cho rằng bạn có thể đặt một Picture vào Image control để giả làm button, nhưng theo riêng tôi cách này không hay, vì khi đặt hình ảnh trong Image lúc người dùng Click chuột sẽ không cảm thấy bị lún xuống như là cách đặt hình ảnh vào Command Button của tôi mới trình bày.

Đầu trang

Làm sao để hiển thị hộp thoại BROWSE FOR FOLDER ?

Trong lúc viết ứng dụng, có lúc bạn chỉ cần cho người dùng chọn thư mục (nếu chọn tập tin thì đã có Dialog Open rồi). Lúc này bạn nên cho hiển thị hộp thoại "Browse for Folder" là tiện nhất.

Sau đây là cách thực hiện:



Hãy khởi động VB. Tạo Project mới

Gõ đoạn Code sau đây vào một Module.

Private Type BrowseInfo

hwndOwner As Long

pIDLRoot As Long

pszDisplayName As Long

lpszTitle As Long

ulFlags As Long

lpfnCallback As Long

lParam As Long

iImage As Long

End Type

Private Const BIF_RETURNONLYFSDIRS = 1

Private Const MAX_PATH = 260

Private Declare Sub CoTaskMemFree Lib "ole32.dll" (ByVal hMem As Long)

Private Declare Function lstrcat Lib "kernel32" Alias "lstrcatA" (ByVal lpString1 As String, ByVal lpString2 As String) As Long

Private Declare Function SHBrowseForFolder Lib "shell32" (lpbi As BrowseInfo) As Long

Private Declare Function SHGetPathFromIDList Lib "shell32" (ByVal pidList As Long, ByVal lpBuffer As String) As Long

Public Function BrowseForFolder(hwndOwner As Long, sPrompt As String) As String

```
Dim iNull As Integer
Dim lpIDList As Long
Dim lResult As Long
Dim sPath As String
Dim udtBI As BrowseInfo

With udtBI
        .hwndOwner = hwndOwner
        .lpszTitle = lstrcat(sPrompt, "")
        .ulFlags = BIF_RETURNONLYFSDIRS
End With

lpIDList = SHBrowseForFolder(udtBI)
If lpIDList Then
        sPath = String$(MAX_PATH, 0)
        lResult = SHGetPathFromIDList(lpIDList, sPath)
        Call CoTaskMemFree(lpIDList)
        iNull = InStr(sPath, vbNullChar)
        If iNull Then sPath = Left$(sPath, iNull - 1)
End If
BrowseForFolder = sPath
```

End Function

Trên Form tạo một Command Button (Name: Command1), gõ đoạn code sau vào sự kiện Click.

```
Private Sub Command1_Click()
Dim strResFolder As String
strResFolder = BrowseForFolder(hWnd, "Select Folder.")
If strResFolder <> "" Then
        MsgBox strResFolder
End If
```

End Sub

Kết quả chọn của người dùng được đặt vào biến strResFolder. Bạn hãy áp dụng vào ứng dụng của mình khi có nhu cầu.

Chúc bạn thành công.

Những câu hỏi về VB

1. Kiểm tra sự tồn tại của tập tin

Đặt dòng code này vào tập tin .BAS (module) và có thể gọi nó từ bất kỳ đâu trong chương trình của bạn.

```
Function FileExists(Byval FileName As String)
        Dim Exist As Integer
        On Local Error Resume Next
        Exists = Len(Dir(FileName$))
        On Local Error Goto 0
        If Exists = 0 Then
                FileExists = False
        Else
                FileExists = True
        End If
End Function
```

Hàm này sẽ nhận tham số là một String (chính là đường dẫn đến file cần kiểm tra sự tồn tại). Trả về một trị kiểu Boolean (TRUE/FALSE), phản ánh kết quả kiểm tra của hàm.

2. Sử dụng phím Enter như là phím Tab để luân chuyển giữa các Control trên Form

```
Private Sub Form_KeyPress(KeyAscii As Integer)
If KeyAscii = vbKeyReturn Then 'Bắt phím Enter
        SendKeys "{TAB}" 'Giả như người dùng nhấn Tab
        KeyAscii = 0 'Huỷ phím Enter
End If
End Sub
```

Nhớ đặt thuộc tính **KeyPreview** của Form là **TRUE** bạn nhé (để cho Form đón các phím nhấn trước các Control trên nó, nếu không bạn sẽ nghe tiếng Beep trong loa chứ chẳng thấy tác dụng gì đâu).

3. Làm cho Form thành TopMost

Là form luôn nằm trên tất cả các cửa sổ khác mặc dù nó không bị kích hoạt (Active).

Khai báo hàm API **SetWindowPos** trong Module như sau:

```
#If Win32 Then
Declare Function SetWindowPos Lib "user32" (ByVal hWnd As Long, ByVal
hWndInsertAfter As Long, ByVal x As Long, ByVal Y As Long, ByVal cx As Long, ByVal cy
As Long, ByVal wFlags As Long) As Long
Global success As Long
#Else
```

Declare Function SetWindowPos Lib "user" (ByVal hWnd As Integer, _
ByVal hWndInsertAfter As Integer, ByVal x As Integer, ByVal Y As _
Integer, ByVal cx As Integer, ByVal cy As Integer, ByVal wFlags As _
Integer) As Integer

Global success As Integer

#End If

Khai báo thêm một số hằng cần thiết trong Module:

Global Const SWP_NOMOVE = 2

Global Const SWP_NOSIZE = 1

Global Const FLAGS = SWP_NOMOVE Or SWP_NOSIZE

Global Const HWND_TOPMOST = -1

Global Const HWND_NOTOPMOST = -2

Lúc nào muốn cho Form thành TopMost thì dùng đoạn code sau đây:

success = SetWindowPos(Me.hWnd, HWND_TOPMOST, 0, 0, 0, 0, FLAGS)

Như vậy khi muốn Form không TopMost nữa thì:

success = SetWindowPos(Me.hWnd, HWND_NOTOPMOST, 0, 0, 0, 0, FLAGS)

Để minh họa, bạn hãy tạo một Project mới.

- Trên Form1 tạo 2 command button là: Command1 & Command2, Add thêm 1 Module (Module1).

- Khai báo hàm API & các hằng trong Module1.

- Lần lượt đặt dòng code thứ nhất cho Command1 (TOPMOST)

success = SetWindowPos(Me.hWnd, HWND_TOPMOST, 0, 0, 0, 0, FLAGS)

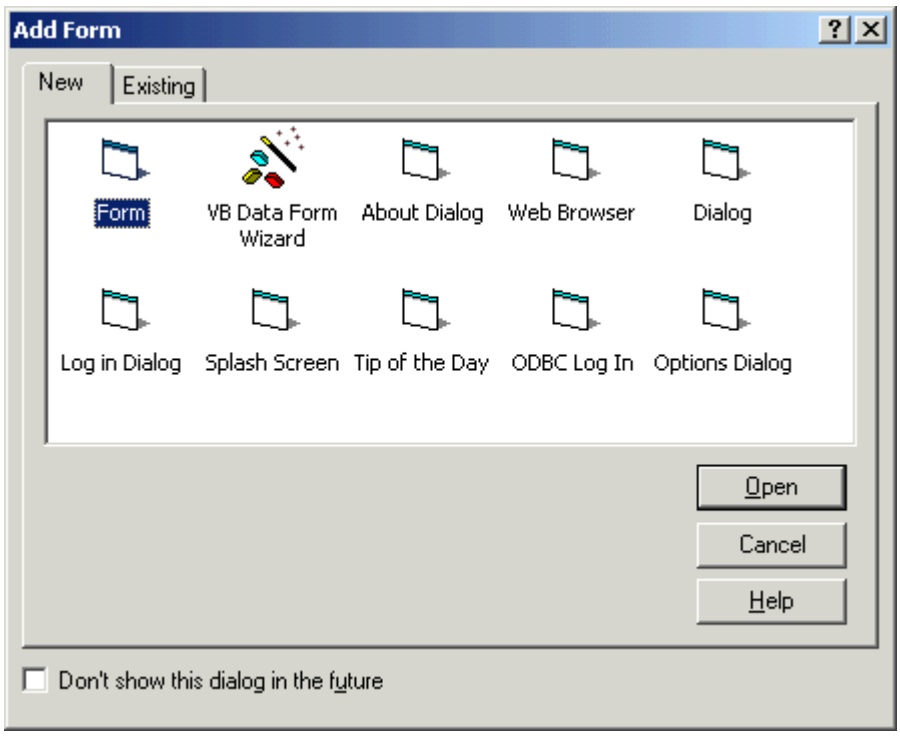
- Và dòng code thứ 2 cho Command2 (NOTOPMOST).

success = SetWindowPos(Me.hWnd, HWND_NOTOPMOST, 0, 0, 0, 0, FLAGS)

- Lúc này hãy ung dung mà nhấn F5 để chạy thử.

4. Hãy tạo Form mẫu cho riêng mình

Có bao giờ bạn tự hỏi: "Làm sao để tạo riêng cho mình một cái Form (của chính mình) và add nó vào Menu của VB để khi cần cho lệnh Add Form là có thể chèn vào ứng dụng của mình ngay lập tức ?". Giống như các mẫu có sẵn của Vb vậy đó.



Bạn thấy không ? About Dialog, Dialog, Splash Screen, Tip of the Day ... các thứ này bạn đều có thể tại riêng cho mình để chương trình mang một sắc thái riêng, độc đáo, và bất ngờ.

Hãy tạo ra những Form thật ưng ý bằng cách:

- Thay đổi các Properties

- Thêm một số Control cần thiết

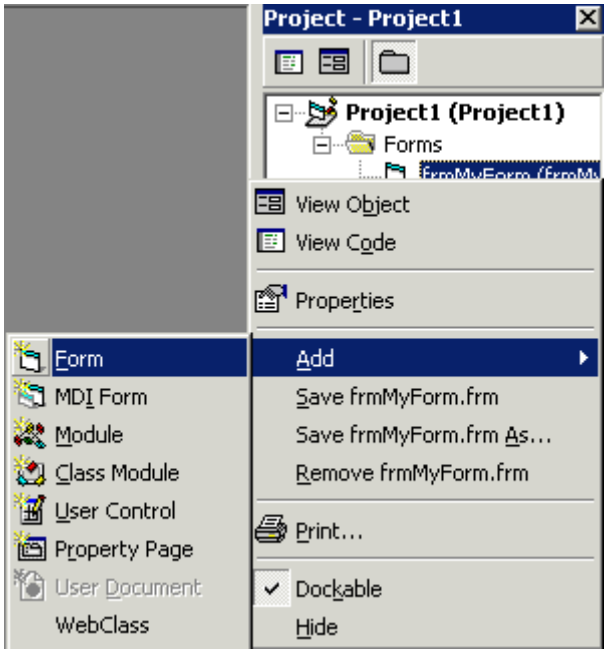
- và cái gì bạn thích hoặc hay dùng

- Gõ cả Source Code nếu cần.

Save Form vào thư mục **Template\Forms** của thư mục chứa chương trình VB (nhớ là chỉ cần Form thôi nhé, không cần tới Project đâu).

Ví dụ như: C:\soft\Microsoft Visual Studio\VB98\Template\Forms

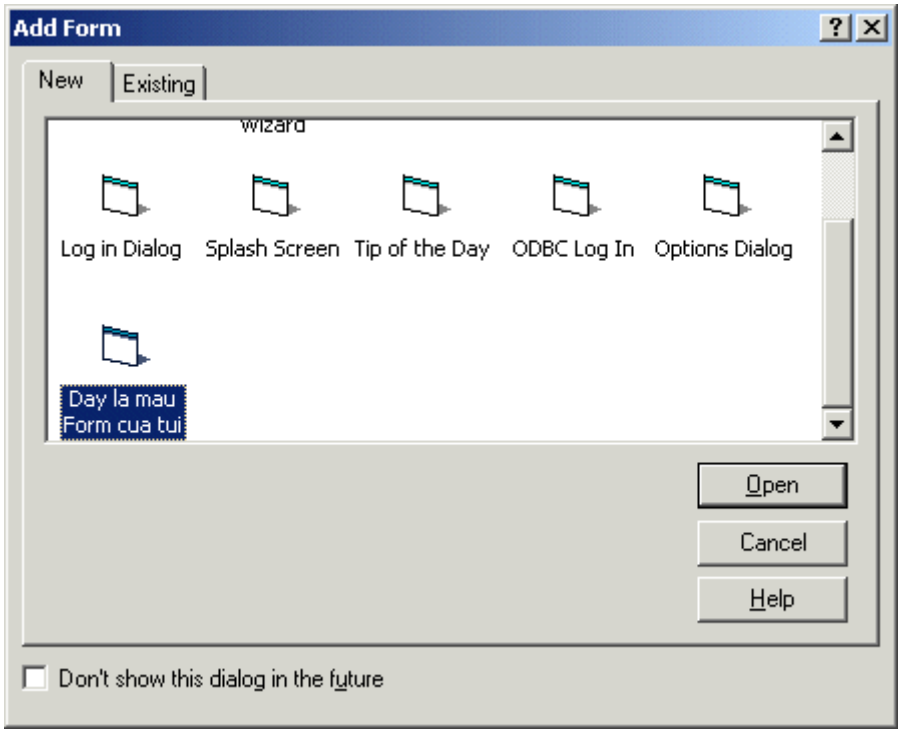
Sau đó bạn hãy vào mục Add Form xem



Lưu ý:

- Tên File chính là tên sẽ xuất hiện trong danh sách này. Ví dụ hình bên dưới là file "Day la mau Form cua tui.frm"
- Nếu bạn "dị ứng" với cái mẫu Form nào có sẵn của VB bạn có thể cho nó mất tích bằng cách Delete (hoặc Move đi nơi khác) tập tin tương ứng với tên là xong.

Và hãy nhìn xem



Đáng ghét ở chỗ là nó luôn cho Form của chúng ta vừa tạo vào cuối danh sách, dẫu bạn đặt tên có cả trăm chữ "a" đi nữa thì cũng "đội số" như thường. Thôi kệ, có cũng dzui rồi, đứng đâu cũng được miễn lợi hại hơn là OK hà (à không, Open mới chính xác).

NHỮNG CÂU HỎI VỀ VB

1. Cắt bớt các khoảng trống trong chuỗi

Khi gõ văn bản hoặc nhập liệu, không phải người dùng nào cũng "tốt lành" mà gõ chính xác từng câu chữ, và đặc biệt là các khoảng trống (space). Có khi trong 1 chương trình, nếu thừa một khoảng trống sẽ dẫn đến lỗi nghiêm trọng khiến chương trình treo ... "tòng teng". Để tránh rắc rối đó, ta cần viết một Module có nhiệm vụ rà soát trong chuỗi, hễ thNong chuỗi, hễ thN thừa khoảng trống thì bỏ ngay, sau đó mới đưa chuỗi đã xử lý xong cho chương trình thực hiện công việc chính. Có thế chương trình hoạt động mới trơn tru.

Hàm sau đây sẽ làm nhiệm vụ:

- Thay thế nơi nào có nhiều khoảng trống trong chuỗi thành 1 khoảng trống duy nhất.

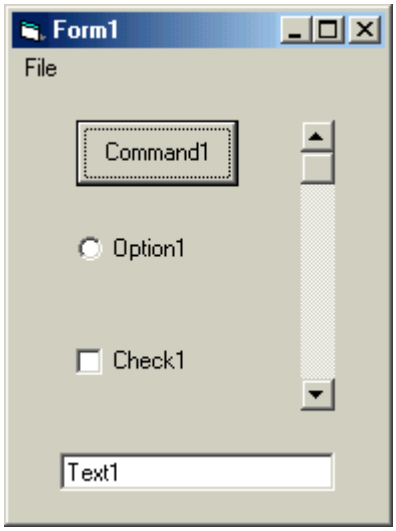
- Xoá bỏ luôn các khoảng trống ở đầu hoặc cuối chuỗi.

```
Private Function PreventDuplicateSpaces(Word)
    Dim i, WordLength, Character, LastCharacter, NewWord
    On Error GoTo ErrorHandler
    WordLength = Len(Word)
    For i = 1 To WordLength
        Character = Mid(Word, i, 1)
        If LastCharacter = " " And Character = " " Then
        Else
            NewWord = NewWord & Character
            LastCharacter = Character
        End If
    Next i
    PreventDuplicateSpaces = Trim(NewWord)
    Exit Function
    ErrorHandler:
    ' Chèn code cần xử lý khi xuất hiện lỗi.
End Function
```

Lưu ý:

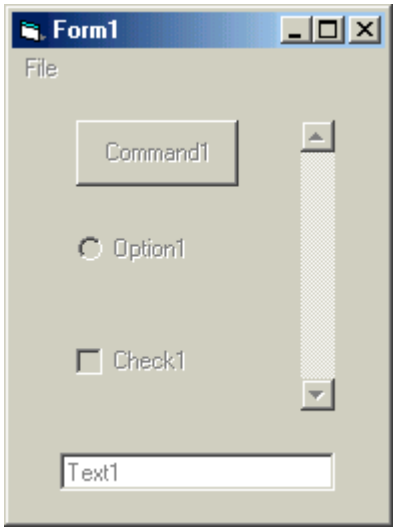
Đây chỉ là 1 cách để minh hoạ (mức độ hiệu quả trung bình), bạn có thể tự viết hàm cho riêng mình, hoặc mở rộng hàm trên làm thêm 1 số chức năng khác.

2. Vô hiệu (Disable) tất cả các Control trên Form




```
Dim i As Integer
For i = 0 To Form1.Controls.Count - 1
        Form1.Controls(i).Enabled = False
Next i
```

Dòng code trên sẽ làm cho tất cả các loại Control (kể cả menu) trên Form bị vô hiệu (thuộc tính Enable = False).



Thật tuyệt vời phải không bạn. Rất nhanh & cực kỳ gọn, nhưng bạn phải thận trọng khi sử dụng nhé. Như vậy việc cho Enable lại các đối tượng trên Form cũng rất là dễ phải không ?

^Top

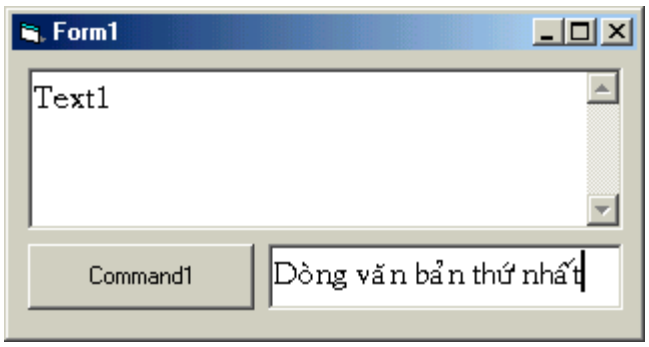
3. Giữ cho TextBox luôn bị cuộn xuống cuối văn bản

Khi số dòng hoặc chiều dài của văn bản lớn hơn phạm vi "thấy được" của Textbox. Lúc này nếu bạn có nhu cầu luôn giữ cho Textbox hiển thị các dòng cuối văn bản, hãy dùng mẹo sau:

```
Text1.SelStart = Len(Text1.Text)
```

Áp đặt thuộc tính SelStart của Textbox là chiều dài của văn bản đang chứa. Hãy triệu gọi dòng code trên bất cứ khi nào bạn cập nhật hoặc thay đổi nội dung của Textbox.

Để dễ hiểu, chúng ta sẽ cùng làm một thí dụ đơn giản sau:



- Hãy tạo một Project mới, trên Form1 có 2 Textbox (Name: Text1, & Text2) & 1 Command Button (Name: Command1).

- Đặt thuộc tính của Text1: MultiLine = True; ScrollBars = 2-Vertical

- Các control còn lại hãy giữ nguyên:

- Đặt vài dòng code vào Form:

Private Sub Command1_Click()

' Nối dòng văn bản trong Text2 vào cuối Text1 (có ghép thêm ký tự xuống dòng vbCrLf - bạn có thể dùng chr(13) & chr(10))

Text1 = Text1 & vbCrLf & Text2

End Sub

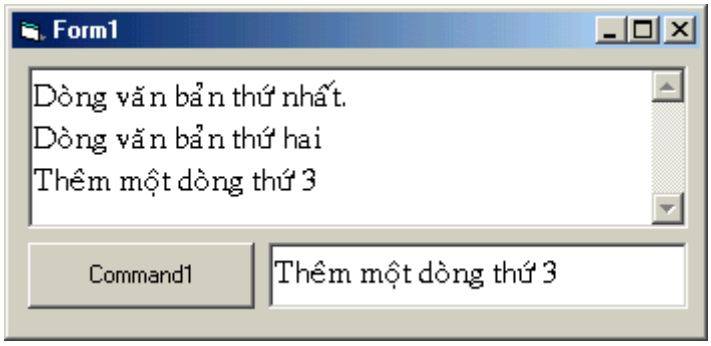
Private Sub Text1_Change()

'Cuộn Textbox khi có sự thay đổi

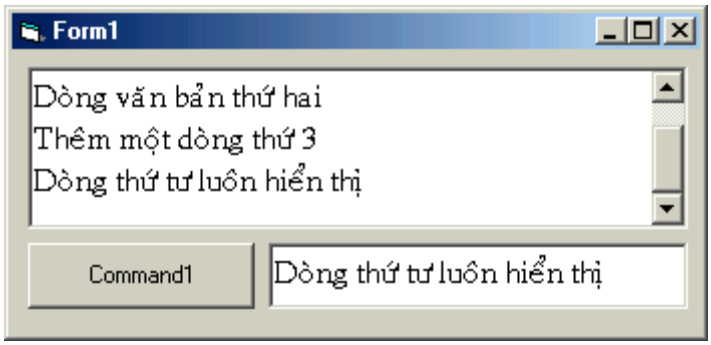
Text1.SelStart = Len(Text1)

End Sub

Còn cách thử như thế nào ư ? Quá dễ, hãy gõ gì đó vào Text2, nhấn Command1, dòng Text sẽ được cập nhật lên Text1, cứ thế từ dòng thứ tư trở đi bạn sẽ thấy ngay điều cần thấy.



Kể từ dòng thứ tư trở đi Text1 luôn được cuộn xuống phía dưới để người dùng có thể xem được thông tin vừa mới cập nhật.



Trên thực tế bạn chớ có đặt dòng

Text1.SelStart = Len(Text1)

vào tình huống Change của Textbox, vì nó sẽ làm cho bạn khó khăn khi định vị Cursor và sửa đổi ở một chỗ khác trong TextBox. Hãy sáng tạo & đặt vào những chỗ (lúc) thích hợp.

4. Đặt Picture vào Status Bar (thanh trạng thái)

Thanh trạng thái là một trong những đối tượng chuẩn mực có hầu hết trong các chương trình chuyên nghiệp. Thanh trạng thái có nhiệm vụ hướng dẫn cho người dùng bằng những giải thích ngắn gọn.

Click phải chuột lên đối tượng Status Bar, chọn mục Properties trong menu tắt. Click chọn thẻ Panels. Và thêm vào một panel mới. Trong khung Picture, click nút Browse, sau đó hãy chọn một bức ảnh mà bạn thích.

Xem hình minh hoạ

Các Format mà VB hỗ trợ là:

- Bimaps
- Icons & Cursors
- GIF Images
- JPEG Images

Tuy nhiên bạn nên dùng những bức ảnh nhỏ cho phù hợp với thanh Status.

5. Đặt Default Folder cho Visual Basic 5/6

Có khi nào bạn cảm thấy rất bực mình vì mỗi khi phát lệnh Save trong VB nó lại cho vào thư mục cài VB không ? Riêng tôi, bực mình lắm, tôi không bao giờ lưu dữ liệu trên cùng 1 ổ đĩa với các chương trình Windows cả, thiệt phát khùng mỗi khi phải làm cái thao tác chuyển đổi thư mục nhàm cháng ấy. Tôi cố gắng tìm kiếm để thay đổi thư mục mỗi khi phát lệnh Save. Nhưng vô cùng thất vọng, chẳng có ở bất kỳ một Version của nào cả.

Tuy nhiên nhờ Windows nên tôi có thể giải quyết được vấn đề này khá dễ dàng. Nếu đang sử dụng Windows từ 98 trở đi bạn có thể làm nhanh như sau:

- Click phải chuột lên Shortcut của VB trong menu Programs, chọn Properties.

Click here

Xem hình minh hoạ

- Sau đó bạn sẽ được hộp thoại Properties. Chú ý phần Start in chứa thư mục mặc định lúc ta cài VB.

Click here

Xem hình minh hoạ

- Vậy muốn như ý hãy đổi thư mục trong Start in lại như bạn muốn: Giả sử tôi đổi thành: H:\MYDOC\VBSOFT vậy là từ đây khi phát lịnh Save trong VB tôi sẽ được toại nguyện.

Click here

Xem hình minh hoạ

Bạn thấy đó một tình huống khá đơn giản, nếu không biết ta sẽ tốn rất nhiều thời gian. Tương tự như vậy, nếu bạn khởi động VB từ một Shortcut khác (ví dụ như trên Desktop) thì bạn vẫn làm y như vậy.