CHƯƠNG 6: LẬP TRÌNH XỬ LÝ GIAO DIỆN & ĐỒ HỌA

Mục tiêu:

Chương này giới thiệu về cách tạo menu cũng như một số hàm xử lý đồ họa, những thành phần quan trọng trong các ứng dụng chạy trên Windows.

Học xong chương này, sinh viên phải nắm bắt được các vấn đề sau:

- Sử dụng menu trong thiết kế giao diện.
- Sử dụng các hộp thoại trong thiết kế ứng dụng.
- Sử dụng các phương thức đồ họa cùng với cách xử lý một số sự kiện để tạo các ứng dụng đồ họa.

Kiến thức có liên quan:

- Các cấu trúc lập trình trong VB.
- Cách thức xử lý sự kiện.

Tài liệu tham khảo:

- **Microsoft Visual Basic 6.0 và Lập trình Cơ sở dữ liệu** - Chương 7, trang 99 - **Nguyễn Thị Ngọc Mai** (chủ biên), **Nhà xuất bản Giáo dục** - **2000**.

I. Menu

I.1. Khái niệm

Menu là một loại điều khiển trong đó người sử dụng có thể lựa chọn các mục từ một danh sách cho trước.

Có 2 loại menu:

- Menu thả xuống (Drop-Down Menu): là dạng menu thông dụng nhất.
- Menu bật ra (Pop-Up Menu): thường hiển thị khi ta ấn nút phải chuột.

I.2. Các thuộc tính của Menu

Các thuộc tính của Menu không chứa trong cửa sổ Properties mà chứa trong Menu Editor.

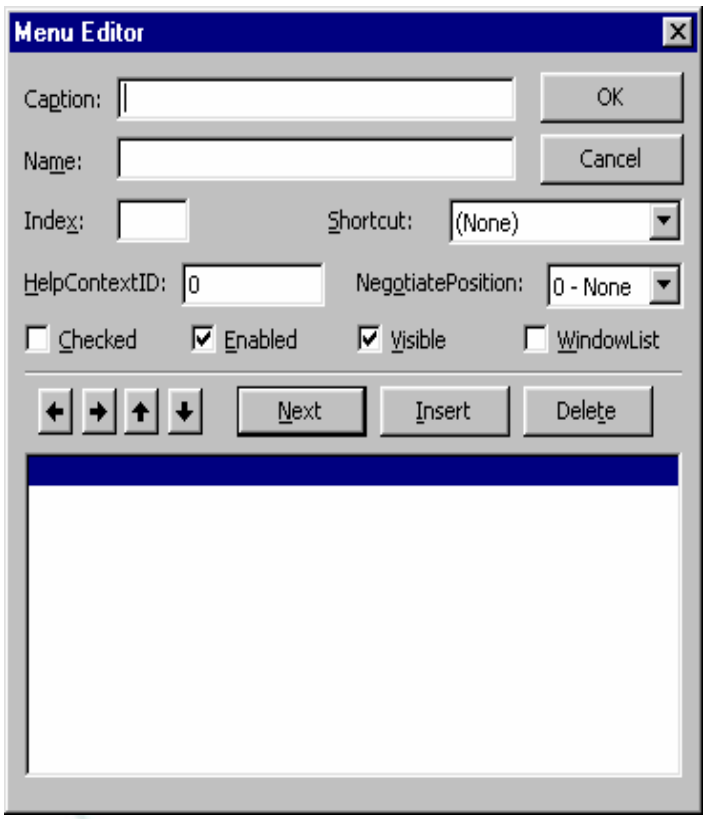
- **Caption**: Là chuỗi hiển thị trên menu.
- **Name**: Phải duy nhất và dễ nhớ. Đây là tên để nhận biết thành phần nào của Menu được chọn.
- **Shortcut**: dùng để thiết lập các phím tắt (Shortcut key).
- **WindowList**: Dùng trong các ứng dụng MDI (Multipe Document Interface). Đây là ứng dụng có một biểu mẫu chính và nhiều biểu mẫu con. Thuộc tính này ra lệnh cho VB hiển thị tiêu đề của các biểu mẫu con trên menu.
- **Checked**: Nếu chọn thuộc tính này thì sẽ có một dấu hiển thị bên cạnh trái, nhưng thuộc tính này không được áp dụng cho những mục menu có chứa menu con.
- **Enabled**: Nếu thuộc tính này không được chọn thì mục này sẽ bị xám đi và người dùng không thể chọn.
- **Visible**: Nếu thuộc tính này không được chọn thì mục này sẽ không được hiển thị.
- **NegotiatePosition**: quản lý vị trí gắn menu trong trường hợp sử dụng các đối tượng ActiveX.

I.3. Các sự kiện

- **Click**: Xảy ra khi người sử dụng chương trình nhấp chuột vào một mục nào đó của Menu.

I.4. Cách tạo Menu

Menu cũng là một loại điều khiển, nhưng Windows sẽ kiểm soát việc vẽ menu. Lập trình viên chỉ quản lý việc điều hành các sự kiện mà thôi.

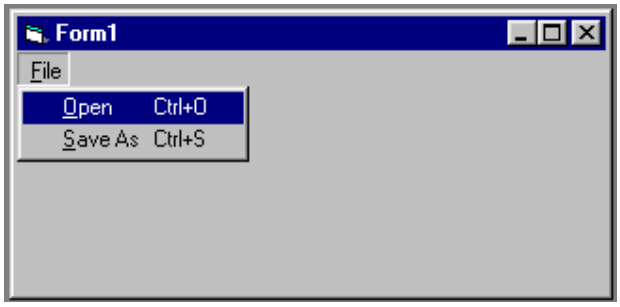


Hình VI.1 Trình soan thảo Menu

Menu không chứa trong hộp công cụ mà được thiết kế từ trình soạn thảo menu. Trong Visual Basic 6.0 IDE, chọn *Tools*, rồi *Menu Editor* để mở chương trình này.

Ví dụ: *Tạo một Drop-Down Menu*.

- Tạo một đề án mới.
- Ấn Ctrl-E để mở Menu Editor
- Ta sẽ tạo một Menu File với Menu con là Open và Save As.
- Trước tiên ta nhập vào &File trong ô Caption và nhập một tên bất kỳ vào ô Name (chẳng hạn là mfile). Ký tự & trước chữ F cho biết chữ F sẽ là phím tắt (ấn Ctrl-F) coi như chọn menu File.
- Tiếp theo ta nhập &Open và &Save As. Để Open và Save As là Sub menu của File, ta chọn Open rồi ấn mũi tên sang trái. Tương tự đối với Save As.

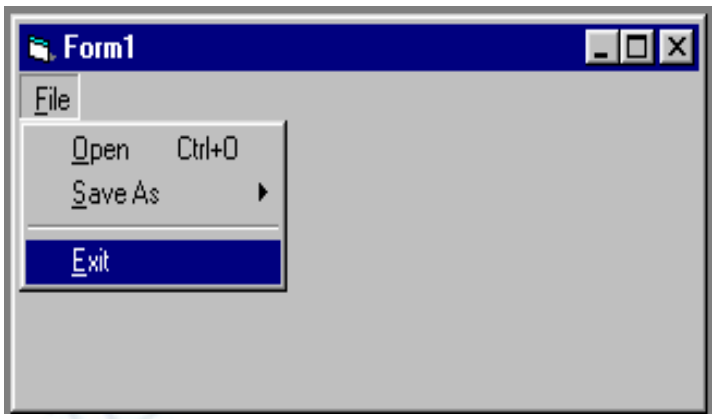


Hình VI.2 Ví dụ tạo Popup Menu

Tách nhóm menu:

Trong trường hợp Menu có nhiều mục, ta sẽ tách nhóm Menu để tiện theo dõi. Chẳng hạn ta thêm vào Menu File mục Exit và tách riêng ra với Open và Save As.

Ta sẽ xen vào giữa hai mục Save As và Exit một mục mới có Caption là "-". Ta có thể theo dõi qua hình VI.3.



Hình VI.3 Ví dụ tách nhóm các

Ví dụ: *Tạo Pop-up menu*

- Sử dụng lại menu đã dùng ở ví dụ trước, nhưng ta sẽ tắt thuộc tính Visible của menu File.
- Sau đó, mở cửa sổ Code của ứng dụng, dùng sự kiện MousUp, nhập vào đoạn lệnh sau:

```
Private Sub Form_MouseUp (Button As Integer, Shift As _
      Integer, X As Single, Y As Single)
            If Button = vbRightButton Then
                  PopupMenu mfile, vbPopupMenuLeftAlign
            End If
End Sub
```

- Chạy thử ứng dụng, khi ta ấn chuột phải, một menu sẽ bật ra.
- Lệnh PopupMenu cho biết tên menu cần bật ra, đó là *tên mà ta đã đặt trong trình soạn thảo MenuEditor*, ở đây là mfile.
- Kế đến, đó là tham số xác định cách hiển thị menu: vbPopupMenuLeftAlign, vbPopupMenuRightAlign, vbPopupMenuCenterAlign.

Sau khi đã thiết kế xong menu, ta sẽ viết các đoạn mã để VB sẽ thi hành một công việc nào đó tương ứng với mục được chọn. *Công việc thi hành sẽ được viết bên trong sự kiện Click của mục đó*.

II. Hộp thoại

II.1. Khái niệm

Hộp thoại (Dialog Box) là một trong những cách VB dùng để giao tiếp với người dùng. Có 4 loại hộp thoại:

- Hộp thông điệp (Message Box).
- Hộp nhập (Input Box).
- Các hộp thoại thông dụng (Common Dialog)
- Hộp thoại hiệu chỉnh (Custom Dialog).

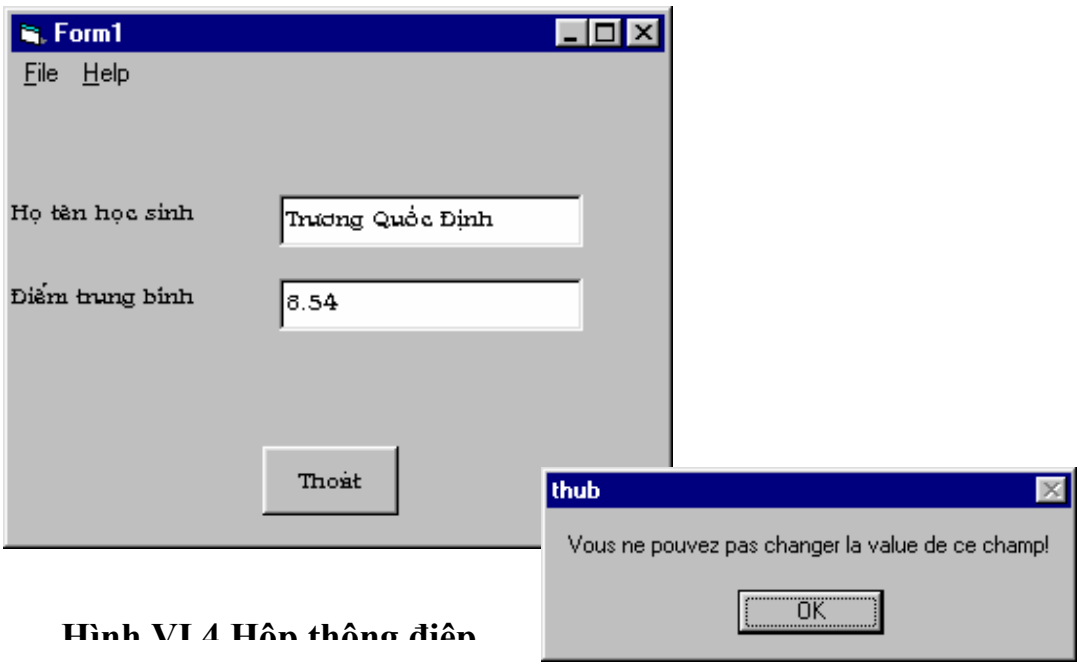
II.2. Hộp thông điệp

Hộp thông điệp cũng có 2 loại: Loại chỉ xuất thông báo, loại có tương tác với người dùng.

II.2.1 Loại chỉ xuất thông báo

- Lúc này ta dùng MsgBox như là một thủ tục.
- Cú pháp: `MsgBox Prompt, Button, Title`. Trong đó:
  Prompt: Chuỗi thông báo sẽ hiển thị.
  Button: Các nút nhấn sẽ được hiển thị trên hộp thông báo.
  Title: Chuỗi hiển thị trên thanh tiêu đề của hộp thông báo.

- Ví dụ:



Hình VI.4 Hộp thông điệp

Chẳng hạn ta xây dựng một biểu mẫu dùng để hiển thị tên và điểm trung bình cuối năm của một học sinh khối lớp 12. Do đó giá trị điểm trung bình cũng như họ tên học sinh là không thể thay đổi. Do đó khi người dùng Click vào một ô Text nào đó, ta sẽ xuất thông báo rằng giá trị này không thể thay đổi.

```
Private Sub Text2_Click()
     MsgBox "Vous ne pouvez pas changer la value de ce champ!"
End Sub
```

Sau khi xuất thông báo, VB sẽ đợi ta ấn vào nút OK hoặc Enter. Sau đó VB sẽ thi hành dòng lệnh ngay sau dòng lệnh MsgBox.

Đôi khi dòng thông báo quá dài, VB sẽ tự động cắt để đưa xuống dòng khác, tuy nhiên có khi sẽ không như mong muốn của lập trình viên. Ta có thể thực hiện công việc này như sau:

```
MsgBox "This is a multi-line " & chr$(10) & " message"
```

Tùy theo thông số truyền vào MsgBox mà có nhiều loại hộp thoại thông điệp khác nhau.

Hằng số	Giá trị	Diễn giải
vbOKOnly	**0**	Chỉ hiển thị nút OK .
vbOKCancel	1	Hiển thị 2 nút OK và Cancel.
vbAbortRetryIgnore	2	Hiển thị các nút Abort, Retry, và Ignore.
vbYesNoCancel	3	Hiển thị các nút Yes, No, và Cancel.
vbYesNo	4	Hiển thị 2 nút Yes và No.
vbRetryCancel	5	Hiển thị 2 nút Retry và Cancel.

Các loại biểu tượng trên hộp công cụ

Hằng số	Diễn giải
vbCritical	Dùng cho những thông báo lỗi thất bại khi thi hành công việc nào đó.
vbQuestion	Dùng cho những câu hỏi yêu cầu người dùng chọn lựa.
vbExclamation	Dùng cho các thông báo của chương trình.
vbInformation	Dùng cho các thông báo cung cấp thêm thông tin.

II.2.2 Loại tương tác với người dùng

Lúc này MsgBox được dùng như một hàm, khi một nút nào đó trên hộp thông báo được ấn, VB sẽ trả về giá trị của nút ấn đó.

Cú pháp:

```
MsgBox (Prompt, Button, Title) As Integer
```

Hằng số	Giá trị	Nút
vbOK	**1**	OK
vbCancel	2	Cancel
vbAbort	3	Abort
vbRetry	4	Retry
vbIgnore	5	Ignore
vbYes	6	Yes

vbNo	7	No

Với những thông điệp quan trọng, ta mong muốn người dùng phải chọn lựa một trong các đề xuất mà ta đưa ra trước khi chuyển qua ứng dụng khác, ta sẽ dùng thông số vbSystemModal.

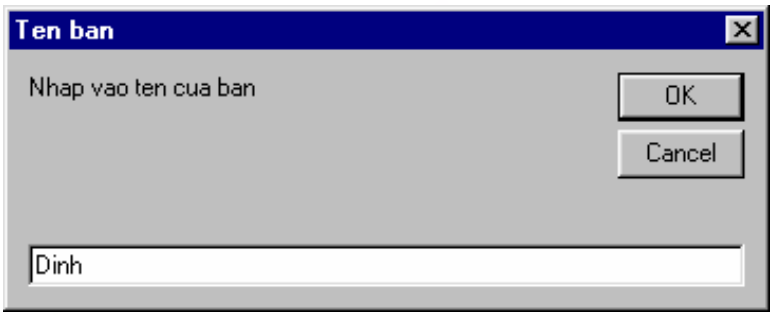
II.3. Hộp nhập

Đây là loại hộp thông điệp cho phép nhận thông tin từ phía người sử dụng. Tuy nhiên trong các ứng dụng, hộp nhập rất ít khi được dùng do:

- Ta không có cách nào để kiểm tra thông tin do người dùng nhập vào khi mà Enter chưa được ấn.
- Thông tin được nhập là rất ít.

Sau đây là một ví dụ về hộp nhập:

```
Public Sub Main ()
     Dim   ReturnString    As    String
     ReturnString = InputBox("Nhap vao ten cua ban")
End  Sub
```



Giá trị trả về của hộp nhập là một chuỗi.

Hình: Một dạng nhập tên

II.4. Các hộp thoại thông dụng

Có 6 loại hộp thoại thông dụng:

- Mở tập tin
- Lưu tập tin
- Chọn màu
- Chọn Font
- In ấn
- Trợ giúp

Tuy có 6 loại, nhưng khi thiết kế biểu mẫu, ta chỉ thấy một công cụ duy nhất đó là CommonDialog. Muốn đưa Common Dialog vào dự án, ta chọn: *Project/Components…/Controls/Microsoft Common Dialog Control 6.0*. Sau đó, Common Dialog sẽ xuất hiện trong hộp công cụ ToolBox.

II.4.1 Hộp thoại mở và lưu tập tin

Hai hộp thoại này có chức năng và thể hiện như nhau. Cả hai hộp thoại đều hiển thị danh sách các tập tin, người dùng có thể duyệt qua các ổ đĩa để tìm các tập tin. Chúng chỉ khác nhau phần tiêu đề và nút nhấn.

Các thuộc tính quan trọng:

- Name: tên của Common Dialog.
- Filter: đây là một chuỗi xác định phần mở rộng của tên các tập tin mà hộp thoại có thể mở hay lưu.

- FilterIndex: nếu có nhiều phần mở rộng của tên tập tin được mô tả trong thuộc tính Filter thì thuộc tính này xác định mặc định loại tập tin nào được chọn (là một số nguyên).
- FileName: trả về tên tập tin sau khi người sử dụng hộp thoại chọn một tập tin nào đó.
- CancelError: nếu TRUE thì trả về giá trị lỗi khi người dùng chọn nút Cancel, mặc nhiên giá trị này là False.

Phương thức:

- ShowOpen: mở ra hộp thoại mở tập tin.
- ShowSave: mở ra hộp thoại lưu tập tin.

Ví dụ:

```
Private Sub Form_Load()
      On Error GoTo ErrHandler
      dlgFile.Filter = "All Files (*.*)|*.*|Text Files " & _
            "(*.txt)|*.txt|Batch Files (*.bat)|*.bat"
      dlgFile.FilterIndex = 2
      dlgFile.ShowOpen
      Exit Sub

   ErrHandler:
      MsgBox Err.Description
End Sub
```

Ở ví dụ trên, ta thiết kế một hộp thoại mở tập tin, trong đó các tập tin được hiển thị theo 3 nhóm tập tin đó là:

- All Files: (*.*)
- Text Files: (*.txt)
- Batch Files: (*.bat)

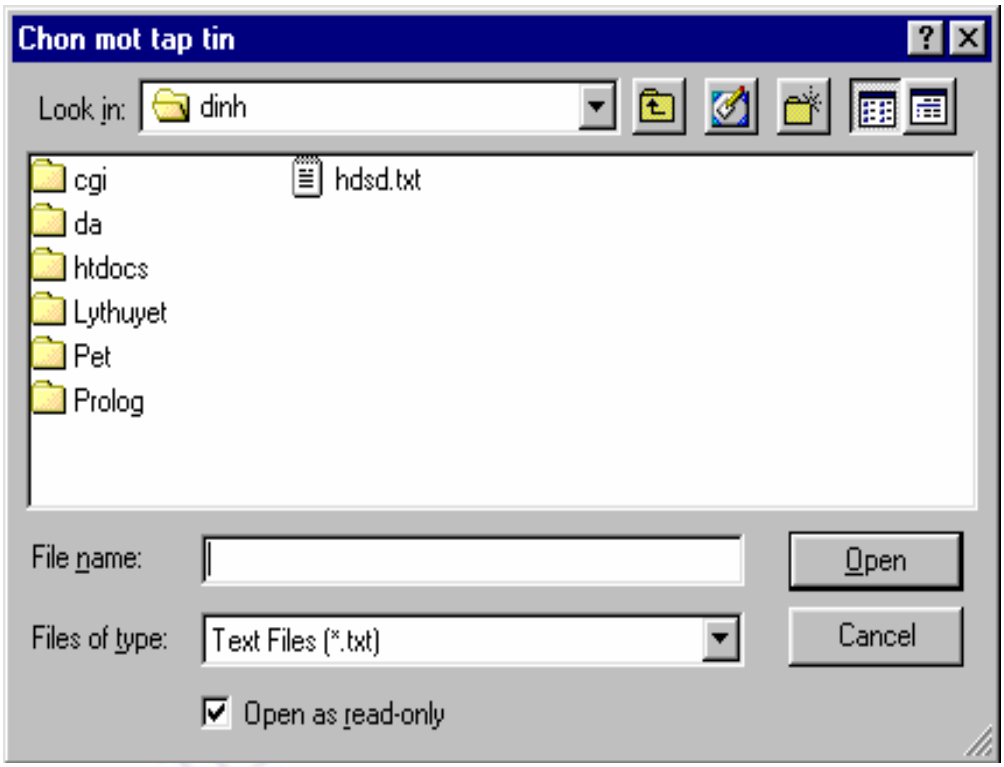
Các nhóm tập tin được thể hiện trong thuộc tính Filter. Mỗi nhóm tập tin cách nhau bởi dấu phân cách |.

Thuộc tính FilterIndex = 2 tức là khi hộp thoại Open được mở lên, thì loại tập tin hiển thị mặc định là Text Files.

Sau khi đã chọn một tập tin và nhấn nút Open, ta sử dụng thuộc tính FileName để nhận về tên tập tin đã chọn.

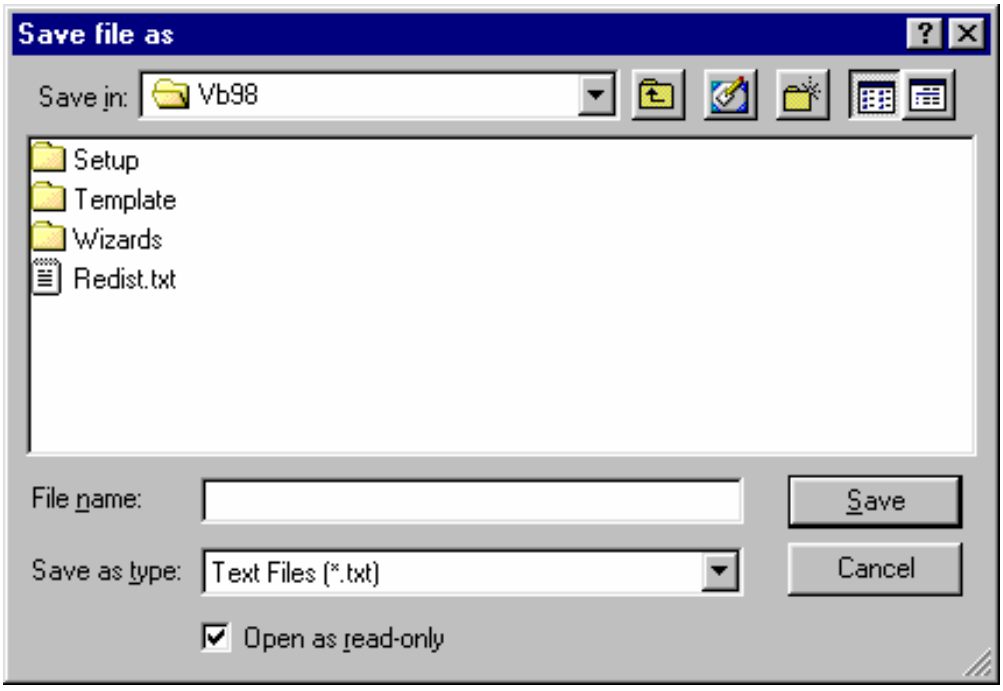
Khi ta chọn thuộc tính CancelError là TRUE, thì khi người dùng ấn nút Cancel trên hộp thoại, ta sẽ nhận được một lỗi và sẽ có cách xử lý lỗi này.

Ta chọn phương thức ShowOpen để hiển thị hộp thoại mở tập tin.



Hình VI.6 Hộp thoại mở tập tin

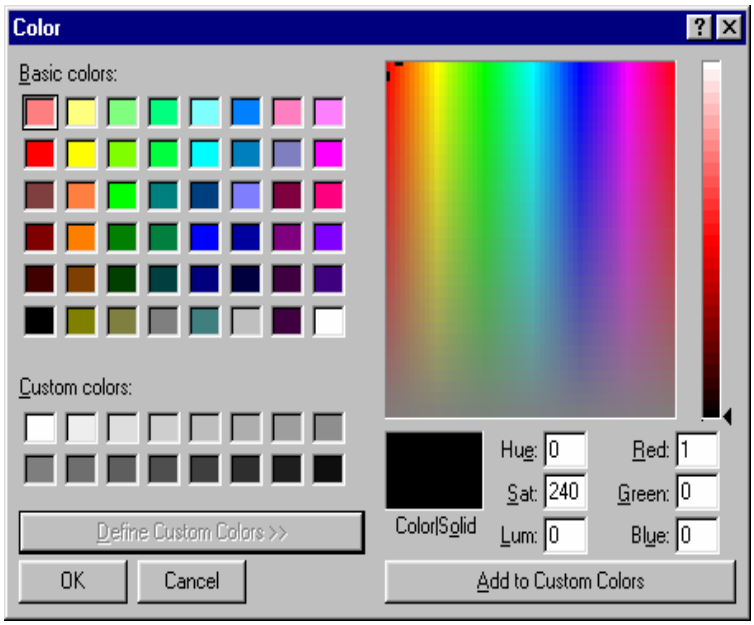
Các thuộc tính cũng tương tự đối với hộp thoại lưu tập tin, ta chỉ cần thay đổi tiêu đề của Dialog và dùng phương thức ShowSave. Dưới đây là minh họa cho hộp thoại lưu tập tin.



Hình VI.7 Hộp thoại lưu tập tin

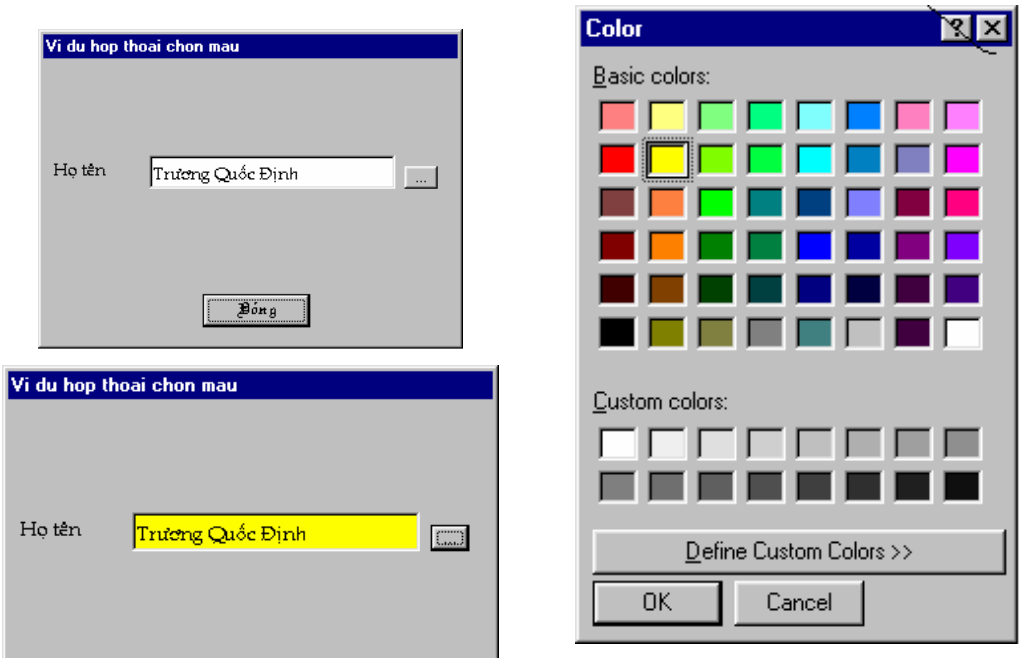
II.4.2 Hộp thoại chọn màu

Đây là hộp thoại cho phép người dùng chọn và hiển thị các màu có sẵn trong bảng màu của Windows cũng như thiết lập thêm nhiều màu mới. Một thuộc tính quan trọng đối với hộp thoại chọn màu đó là thuộc tính Color, thuộc tính này trả về giá trị của màu đã được chọn. Ta sẽ dùng phương thức ShowColor để hiển thị hộp thoại chọn màu.



Hình VI.8 Hộp thoại chọn màu

Trong một số ứng dụng, ta sẽ dùng hộp thoại chọn màu để thay đổi giá trị màu của các điều khiển trong một số trường hợp nào đó. Ví dụ thay đổi màu nền của điều khiển TextBox trong ví dụ dưới đây:



Hình VI.9 Ví dụ sử dụng hộp thoại chọn mầu

Ta sẽ thiết kế một nút nhấn nhỏ bên cạnh điều khiển TextBox, nút nhấn này cho phép người sử dụng chọn màu nền của TextBox. Ta có đoạn mã lệnh sau:

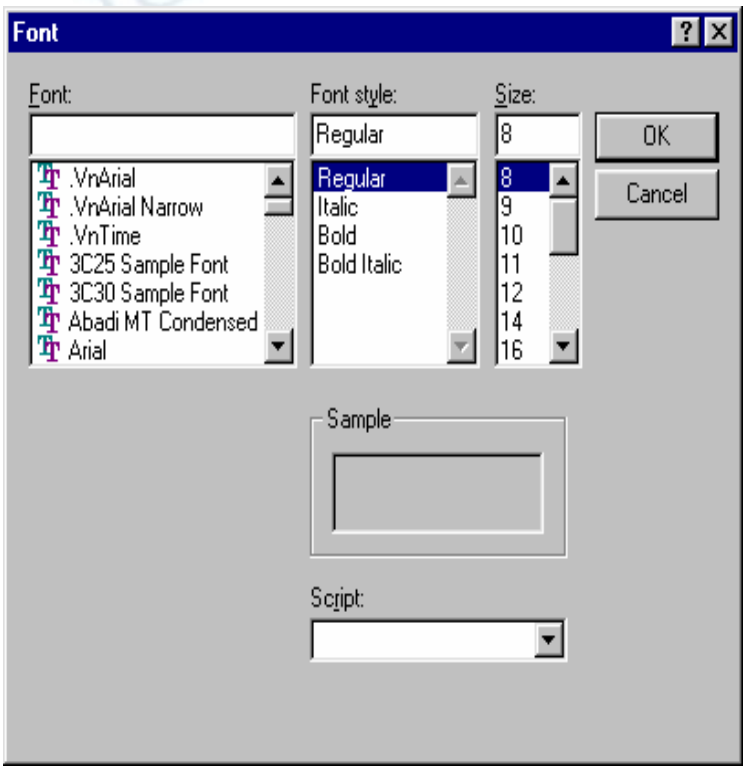
```
Private Sub Command2_Click()
      On Error GoTo ErrHandler
      CommonDialog1.ShowColor
      Text1.BackColor = CommonDialog1.Color

ErrHandler:
      CommonDialog1.ShowColor
End Sub
```

Trước khi chạy chương trình cần xác định thuộc tính CancelError = TRUE.

II.4.3 Hộp thoại chọn Font chữ

Cho phép người dùng chọn Font màn hình, máy in hay cả hai. Khi dùng hộp thoại chọn Font ta phải dùng thuộc tính Flags quy định loại Font nào sẽ được hiển thị.



Hình VI.10 Hộp thoại Font

Thuộc tính	Giải thích
Color	Lưu giữ giá trị của màu được chọn
FontBold	TRUE nếu người dùng chọn chế độ đậm (Bold) và FALSE nếu

	ngược lại.
FontItalic	TRUE nếu người dùng chọn chế độ nghiêng (Italic) và FALSE nếu ngược lại.
FontStrikeThru	TRUE nếu chọn chế độ gạch ngang các ký tự.
FontUnderLine	TRUE nếu chọn chế độ gạch dưới
FontName	Tùy ý
Max	Kích cỡ lớn nhất của Font được hiển thị
Min	Kích cỡ nhỏ nhất của font được hiển thị
FontSize	Kích cỡ của Font được chọn

Các giá trị của thuộc tính Flags:

Hằng	Giá trị	Hiệu quả
cdlCFPrinterFonts	&H2	Chỉ hiển thị font máy in
cdlCFScreenFonts	&H1	Chỉ hiển thị font màn hình
cdlCFBoth	&H3	Chỉ hiển thị font màn hình và font máy in
cdlCFScalableOnly	&H20000	Hiển thị font tỷ lệ như là fonts TrueType

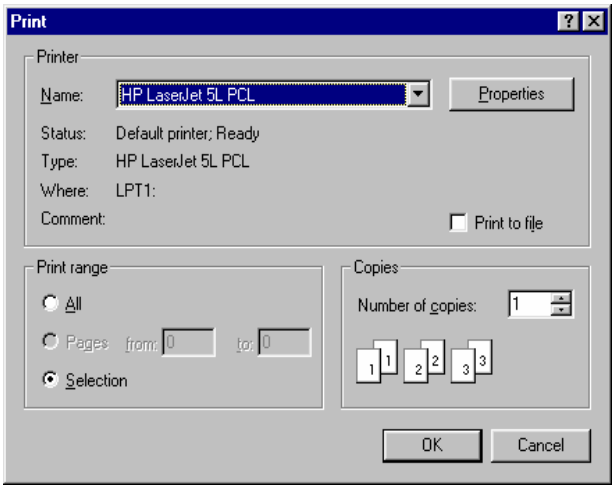
Nếu muốn chọn màu cho Font, ta thêm 256 vào giá trị của thuộc tính Flags. Nếu không có điều này, ta chỉ thấy tên Font, kiểu Font và kích cỡ Font mà thôi.

Để mở hộp thoại chọn Font, ta sử dụng phương thức ShowFont.

II.4.4 Hộp thoại in ấn

Đây là hộp thoại cho phép xác lập các thông tin về máy in chẳng hạn như bao nhiêu dữ liệu được in, máy in sẽ hoạt động như thế nào…

Hộp thoại in ấn, nó trả về 3 thuộc tính thông dụng: Copies, FromPage và ToPage.



Hình VI.11 Hộp thoại in ấn

Thuộc tính	Giải thích
Copies	Số bản in
FromPage	Số thứ tự của trang bắt đầu
Max	Số bản in tối đa cho phép

Min	Số bản in tối thiểu cho phép
PrinterDefault	Nếu gán thành TRUE, mọi thay đổi mà người dùng thực hiện sẽ được ghi lại thành các thay đổi trên hệ thống và có ảnh hưởng đến các ứng dụng khác nếu có sử dụng máy in.
ToPage	Số thứ tự của trang in cuối cùng

Để mở hộp thoại in ấn, ta sử dụng phương thức ShowPrinter.

III. Xử lý các sự kiện chuột và bàn phím

III.1 Sự kiện chuột

Biểu mẫu hoặc điều khiển có thể nhận biết sự kiện chuột khi có con trỏ chuột đi ngang qua.

Có 3 sự kiện chuột chủ yếu, đó là

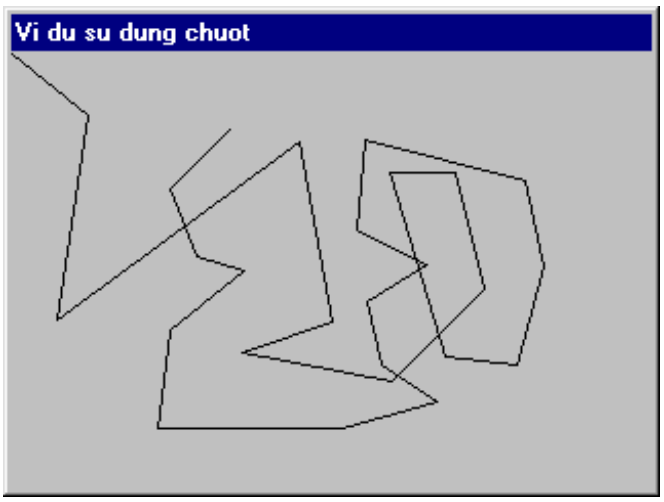
Sự kiện	Giải thích
MouseDown	Xảy ra khi người sử dụng ấn chuột (chuột trái hoặc phải)
MouseUp	Xảy ra khi người sử dụng thả một nút chuột bất kỳ
MouseMove	Xảy ra khi con trỏ chuột di chuyển đến một điểm mới trên màn hình.

Các tham số

Tham số	Giải thích
Button	Cho biết phím chuột nào được ấn
Shift	Cho biết SHIFT hay CTRL hay ALT được ấn
X, Y	Xác định vị trí của con trỏ chuột đối với hệ tọa độ của điều khiển

Ví dụ 1: Sử dụng sự kiện MouseDown để vẽ các đoạn thẳng nối tiếp nhau mỗi khi ta dùng chuột chấm một điểm trên biểu mẫu. Ta có thể thực hiện điều đó với đoạn mã lệnh xử lý sự kiện Form_MouseDown như sau:

```
Private Sub Form_MouseDown(Button As Integer, & _
     Shift As Integer, X As Single, Y As Single)
          Line -(X, Y)
End Sub
```

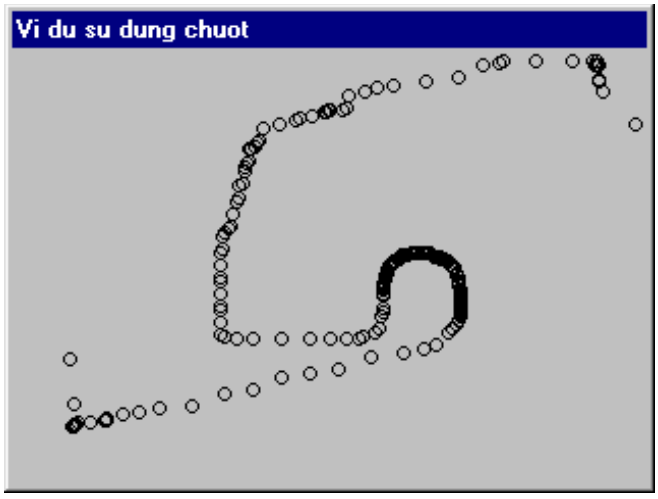


Ví dụ 2: Sử dụng sự kiện MouseUp để hiển thị một thông điệp cho biết nút chuột nào vừa được thả. Sự kiện Form_MouseUp được xử lý:

```
Private Sub Form_MouseUp (Button As Integer,  & _
     Shift As Integer, X As Single, Y As Single)
          If  Button  =  1  Then
               Print  "Ban vua tha phim chuot trai"
          End If
          If  Button  =  2  Then
               Print  "Ban vua tha phim chuot phai"
          End If
          If  Button  =  4  Then
               Print  "Ban vua tha phim chuot giua"
          End If
End Sub
```

Ví dụ 3: Sử dụng sự kiện MouseMove để vẽ các đường tròn liên tục trên biểu mẫu. Sự kiện Form_MouseMove được xử lý:

```
Private Sub Form_MouseMove(Button As Integer, & _
     Shift As Integer, X As Single, Y As Single)
          Circle (X, Y), 50
End Sub
```



Hình VI.13 Ví dụ về MouseMove

Với ví dụ 3 ta nhận thấy rằng: sự kiện MouseMove không nhất thiết phải xảy ra ứng với mỗi Pixel khi con trỏ chuột đi qua. Thực ra mỗi đơn vị thời gian nào đó, hệ điều hành phát ra một số thông điệp. Ở đây, ta vẽ đường tròn ứng với sự kiện MouseMove, nếu người dùng di chuyển chuột chậm, thì các đường tròn sẽ được vẽ sát nhau và ngược lại nếu chuột được di chuyển nhanh.

Hiệu chỉnh con trỏ chuột

Ta có thể dùng thuộc tính MousePointer để hiển thị một biểu tượng, con trỏ màn hình hay con trỏ chuột đã được hiệu chỉnh. Dưới đây là các giá trị của thuộc tính MousePointer:

Hằng	Giá trị	Diễn giải
ccDefault	0	(Default) Shape determined by the object.
ccArrow	1	Arrow.
ccCross	2	Cross (cross-hair pointer).
ccIbeam	3	I Beam.
ccIcon	4	Icon (small square within a square).
ccSize	5	Size (four-pointed arrow pointing north, south, east, and west).
ccSizeNESW	6	Size NE SW (double arrow pointing northeast and southwest).
ccSizeNS	7	Size N S (double arrow pointing north and south).
ccSizeNWSE	8	Size NW, SE.
ccSizeEW	9	Size E W (double arrow pointing east and west).
ccUpArrow	10	Up Arrow.
ccHourglass	11	Hourglass (wait).
ccNoDrop	12	No Drop.
ccArrowHourglass	13	Arrow and hourglass.
ccArrowQuestion	14	Arrow and question mark.
ccSizeAll	15	Size all.

ccCustom	99	Custom icon specified by the MouseIcon property.

III.2 Sự kiện bàn phím

Bàn phím cũng có 3 sự kiện, đó là sự kiện KeyPress (khi một phím có mã ASCII bất kỳ được ấn), KeyDown (khi một phím bất kỳ được ấn), KeyUp (khi một phím bất kỳ được thả)

Chỉ có điều khiển đang có Focus mới bắt sự kiện bàn phím. Còn đối với *biểu mẫu, nó chỉ bắt được sự kiện bàn phím mỗi khi nó đã được kích hoạt và không có bất kỳ điều khiển nào trên nó có Focus. Tuy nhiên ta có khể khắc phục điều này nếu như gán giá trị thuộc tính KeyPreview của biểu mẫu là True*, biểu mẫu sẽ nhận mọi sự kiện bàn phím của mọi điều khiển đặt trên nó, điều này hữu ích khi ta muốn thực hiện cùng một công việc nào đó cho một phím được ấn mà không quan tâm rằng Focus đang thuộc điều khiển nào.

Các sự kiện KeyDown, KeyUp có thể phát hiện một số tình huống mà sự kiện KeyPress không phát hiện:

- Khi người dùng bấm một tổ hợp phím SHIFT, CTRL và ALT.
- Phím định hướng.
- PAGEUP và PAGEDOWN.
- Phân biệt được phím số ở bên phải bàn phím và phím số ở bên trái bàn phím.
- Đáp ứng khi thả phím.
- Phím chức năng không trùng với menu.

Các sự kiện bàn phím là không loại trừ nhau. Tức là một phím được ấn thì có thể là cả hai sự kiện KeyPress và KeyDown cùng được phát ra. Nhưng nếu là một phím mà KeyPress không phát hiện được thì chỉ có KeyDown và KeyUp xảy ra.

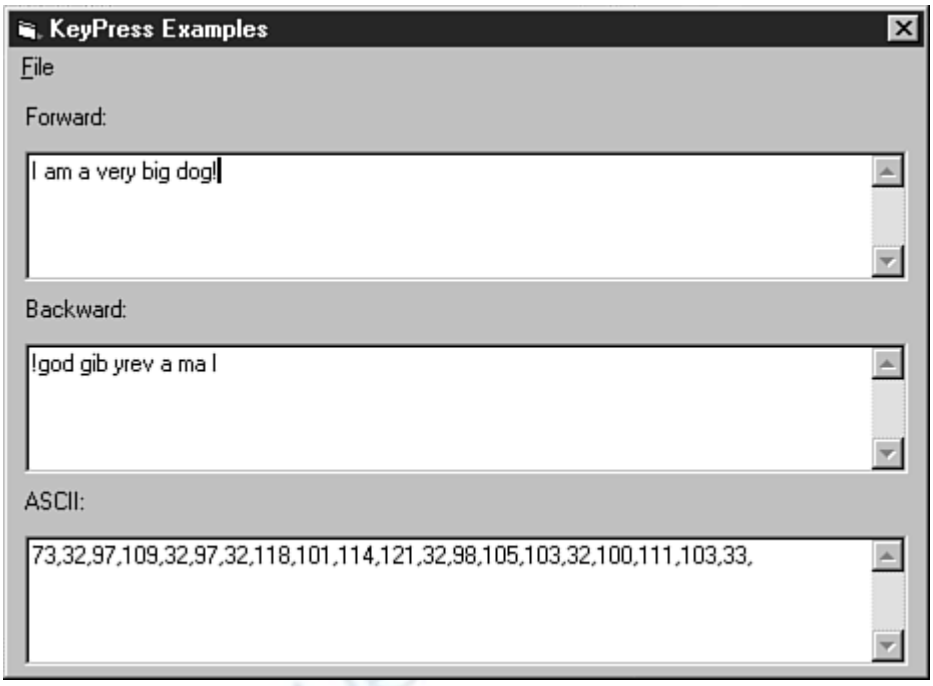
Thuộc tính KeyPreview

Đôi khi ta muốn tất cả các điều khiển trên Form nhận được sự kiện KeyPress chứ không phải chỉ có điều khiển đang nhận con trỏ (Focus), ta sẽ phải sử dụng thuộc tính KeyPreview.

Khi chúng ta thiết kế một Form, giá trị mặc định của thuộc tính này sẽ là False, khi đó bất kỳ một sự kiện bàn phím nào cũng đều được gởi đến điều khiển đang giữ quyền điều khiển. Tuy nhiên nếu giá trị của thuộc tính là True thì Form sẽ là nơi nhận mọi sự kiện bàn phím.

Sau đây là ví dụ về điều này:

```
Private   Sub   Form_KeyPress (KeyAscii   As   Integer)
       ' Gởi điều khiển đến textbox đầu tiên
       txtForward.SetFocus
       txtBackward.Text = Chr(KeyAscii) & txtBackward.Text
       txtAscii.Text = txtAscii.Text & CStr(KeyAscii) & ","
End Sub
```



Hình VI.15 Các sự kiện bàn phím

Trong ví dụ trên, nếu như giá trị của thuộc tính KeyPreview là False thì các TextBox Backward và Ascii không thể nhận được giá trị.

IV. Xử lý đồ họa và giao diện

IV.1 Hiển thị hình ảnh

Bởi vì hệ điều hành Windows là hệ điều hành sử dụng giao diện đồ họa, do đó một phần quan trọng trong ứng dụng của ta là cách thức để hiển thị hình ảnh trong ứng dụng của mình. VB cung cấp cho chúng ta 4 loại điều khiển để hiển thị và quản lý hình ảnh: Picture Box, Image, Shape và Line.

IV.1.1 Sử dụng Picture Box

Cách dùng chính của điều khiển Picture Box là hiển thị hình ảnh. Hình ảnh mặc định mà Picture Box hiển thị có tên được xác định bởi thuộc tính Picture (có thể bao gồm cả đường dẫn).

Ta cũng cần chú ý một điều đó là đối tượng Form cũng có thể hiển thị một hình ảnh xem như là ảnh nền thông qua thuộc tính Picture.

Thuộc tính AutoSize của điều khiển Picture Box quy định kích thước của điều khiển có thể thay đổi một cách tự động hay không? Nếu giá trị của thuộc tính này là True, thì kích thước của điều khiển sẽ thay đổi theo kích thước của hình ảnh mà nó chứa. Tuy nhiên sự thay đổi này có thể làm ứng dụng của chúng ta trở nên xấu đi do sự thay đổi kích thước của điều khiển Picture Box sẽ không quan tâm đến các vị trí của các điều khiển khác cùng có trên biểu mẫu. Tốt hơn hết là chúng ta nên thử qua tất cả các hình ảnh có thể hiển thị tại thời điểm thiết kế để quy định kích thước của điều khiển cho hợp lý.

Hơn thế nữa, có thể thay đổi hình ảnh hiển thị bên trong Picture Box bằng cách sử dụng phương thức LoadPicture để thay đổi giá trị của thuộc tính Picture.

Ngoài ra ta có thể dùng Picture Box như một vật chứa các điều khiển khác. Cũng như điều khiển Frame, ta có thể đặt các điều khiển khác bên trong Picture Box. Ta thường sử dụng Picture box chứa các điều khiển Label để hiển thị các thông tin và trạng thái của ứng dụng.

Một cách dùng khác của Picture box đó là xem như một khung vẽ trắng và ta dùng các phương thức Circle, Line, PSet hay Point để vẽ lên trên điều khiển này.

IV.1.2 Sử dụng Image Control

Image control cũng như điều khiển Picture Box nhưng chỉ dùng để hiển thị hình ảnh. Nó không thể dùng làm vật chứa và cũng không có một số thuộc tính như điều khiển Picture Box.

Các phương thức dùng để hiển thị, thay đổi hình ảnh cũng như điều khiển Picture Box, tuy nhiên thuộc tính quy định việc kích thước thay đổi một cách tự động là thuộc tính Stretch.

Một trong những ứng dụng chủ yếu của điều khiển Image Control đó là sử dụng như một nút lệnh, đây là một cách thức tiện lợi để thiết kế nút lệnh chứa hình ảnh thay vì là các câu văn bản.

Khi sử dụng Image Control như một nút lệnh, ta nên nhớ rằng điều khiển này sẽ không thể có trạng thái *ấn xuống* khi được Click, vì thế ta nên thay đổi hình ảnh hiển thị bởi Image Control để cho biết rằng nút lệnh đã được ấn.

IV.2 Xử lý đồ họa

IV.2.1 Tọa độ màn hình

Góc trái trên của màn hình có tọa độ là (0,0) có nghĩa là X = 0 và Y = 0. Như vậy tức là khi di chuyển sang phải màn hình thì X tăng lên cũng như di chuyển xuống dưới thì Y tăng lên.

Tuy nhiên VB chỉ cho phép ta vẽ trên biểu mẫu hay hộp hình (picture box). Khi đó hệ tọa độ sẽ được gắn với từng điều khiển.

Ta thường sử dụng 2 hệ tọa độ chủ yếu sau: Twips và Pixel.

Twips: Đây là hệ tọa độ mặc định dùng cho biểu mẫu. Mỗi điểm sẽ bằng 1/567 cm. Đây là hệ tọa độ không bị ảnh hưởng bởi thiết bị, kết quả vẽ sẽ như nhau trên màn hình VGA chuẩn, trên máy in hay trên màn hình có độ phân giải cao khác.

Pixel: Đây là hệ tọa độ phổ biến nhất, mỗi một điểm trên màn hình sẽ bằng chính xác với một Pixel, như vậy khi sử dụng hệ tọa độ này sẽ giúp cho các ứng dụng đồ họa thực hiện được nhanh hơn vì không phải thông qua quá trình đổi hệ tọa độ.

IV.2.2 Các phương thức đồ họa

Các điều khiển được vẽ lên biểu mẫu lúc thiết kế nhưng các phương thức đồ họa cho phép vẽ trực tiếp khi ứng dụng thi hành.

Phương thức PaintPicture

Phương thức PaintPicture cho phép sao chép nhanh các hình ảnh từ biểu mẫu, hộp hình và máy in.

Cú pháp:

object.PaintPicture picture, x1, y1, width1, height1, x2, y2, width2, height2, opcode

Object	Object là đối tượng mà phương thức sẽ làm việc, nó có thể là biểu mẫu, hộp hình hay đối tượng máy in.
Picture	Hình ảnh nguồn sẽ được vẽ lên đối tượng phải được chỉ rõ bởi thuộc tính Picture của biểu mẫu hoặc hộp hình.
x1, y1	Giá trị chỉ định vị trí của hình ảnh trên đối tượng. Thuộc tính ScaleMode xác định hệ tọa độ nào được sử dụng.
Width1	Giá trị xác định độ rộng của hình ảnh, nếu bỏ qua thì mặc định là độ rộng của ảnh nguồn.
Height1	Giá trị xác định độ cao của hình ảnh, nếu bỏ qua thì mặc định là độ cao của ảnh nguồn.
x2, y2	Các giá trị xác định hình ảnh sẽ được vẽ lại từ vị trí nào. Nếu bỏ qua thì giá trị mặc định là 0, tức toàn bộ hình ảnh được vẽ lại.
Width2	Tương tự như Width1, nhưng ở đây là tác động đến ảnh nguồn.
Height2	Tương tự như Height1, nhưng ở đây là tác động đến ảnh nguồn.
Opcode	Đây là tùy chọn và chỉ có tác dụng với ảnh Bitmap.

Ví dụ: Thiết kế chương trình sao cho khi người sử dụng vừa di chuyển vừa nắm giữ phím chuột thì một hình ảnh sẽ được vẽ lại ở tọa độ mới.

```
Dim re

Private Sub Form_Load()
      re = False
End Sub

Private Sub Form_MouseDown(Button As Integer, & _
      Shift As Integer, X As Single, Y As Single)
      re = True
End Sub

Private Sub Form_MouseMove(Button As Integer, & _
      Shift As Integer, X As Single, Y As Single)


      If re Then
            Form1.PaintPicture Image1.Picture, X, Y, & _
                  Image1.Width, Image1.Height
      End If
End Sub

Private Sub Form_MouseUp(Button As Integer, & _
      Shift As Integer, X As Single, Y As Single)
      re = False
End Sub
```