KS. Lâm Hoài Bảo

GIÁO TRÌNH

Visual Basic

Ebook.moet.gov.vn, 2008

CHƯƠNG 1 TỔNG QUAN VỀ VISUAL BASIC 6.0

Mục tiêu:

Chương này giới thiệu về môi trường phát triển tích hợp (IDE) Microsoft Visual Basic 6.0; cũng như giúp sinh viên có cái nhìn tổng quan về Visual Basic.

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

- Sử dụng môi trường phát triển tích hợp VB 6.0 để phát triển ứng dụng.
- Cách tạo dự án mới (New Project) trong VB 6.0.

Kiến thức có liên quan:

- Sử dụng hệ điều hành Windows.

Tài liệu tham khảo:

- **Visual Basic 6 Certification Exam Guide** - Chapter 1, Page 1 - **Dan Mezick & Scot Hillier** - **McGraw-Hill** - **1998.**

I. Giới thiệu về Visual Basic 6.0

Visual Basic 6.0 (VB6) là một phiên bản của bộ công cụ lập trình Visual Basic (VB), cho phép người dùng tiếp cận nhanh cách thức lập trình trên môi trường Windows. Những ai đã từng quen thuộc với VB thì tìm thấy ở VB6 những tính năng trợ giúp mới và các công cụ lập trình hiệu quả. Người dùng mới làm quen với VB cũng có thể làm chủ VB6 một cách dễ dàng.

Với VB6, chúng ta có thể :

- Khai thác thế mạnh của các điều khiển mở rộng.
- Làm việc với các điều khiển mới (ngày tháng với điều khiển MonthView và DataTimePicker, các thanh công cụ có thể di chuyển được CoolBar, sử dụng đồ họa với ImageCombo, thanh cuộn FlatScrollBar,…).
- Làm việc với các tính năng ngôn ngữ mới.
- Làm việc với DHTML.
- Làm việc với cơ sở dữ liệu.
- Các bổ sung về lập trình hướng đối tượng.

II. Cài đặt Visual Basic 6.0

Sử dụng chương trình Setup, người dùng có thể cài đặt VB6 lên máy tính của mình. Chương trình Setup này còn cài đặt các tập tin cần thiết để xem tài liệu trên đĩa CD MSDN (Microsoft Developer Network). Nếu cần, người dùng có thể cài đặt riêng phần tài liệu và ví dụ mẫu của Visual Basic lên máy tính.

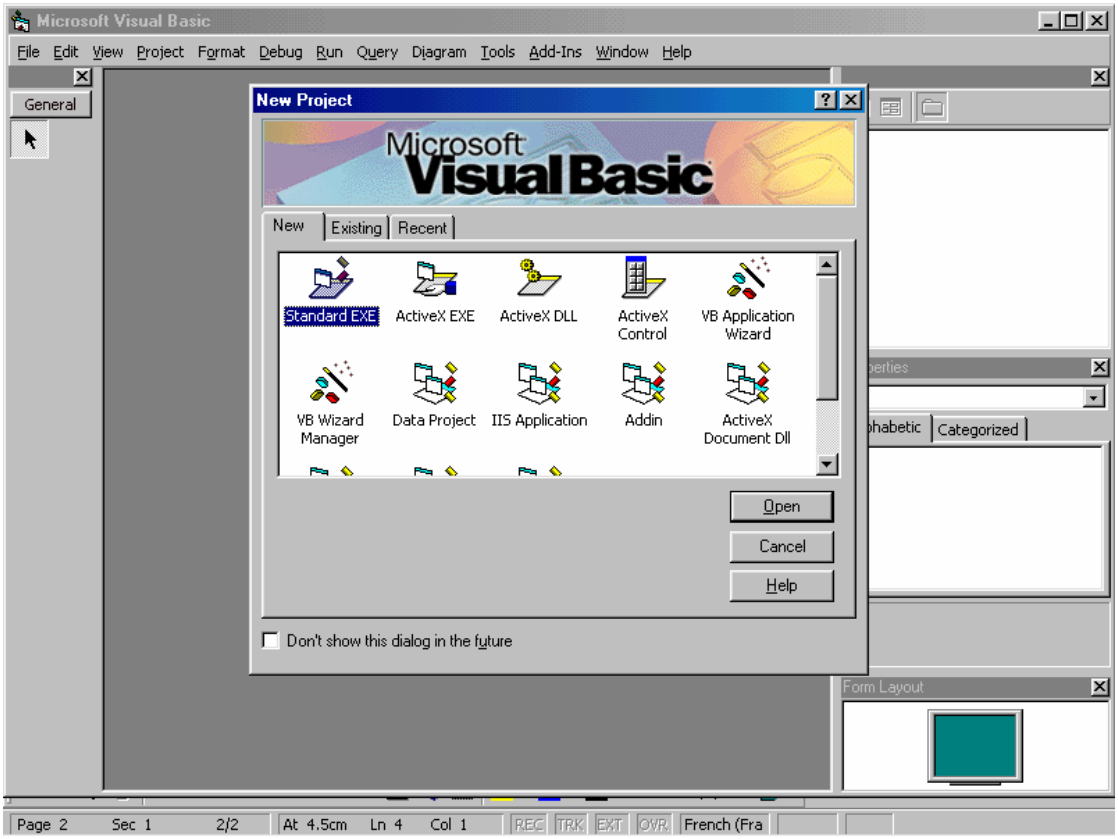
Để cài đặt VB6, người dùng nên kiểm tra máy tính của mình đảm bảo được cấu hình tối thiểu. Các yêu cầu hệ thống tối thiểu :

- Microsoft Windows 95 trở lên hoặc là Microsoft Windows NT Workstation 4.0 trở lên.
- Tốc độ CPU 66 MHz trở lên.
- Màn hình VGA hoặc màn hình có độ phân giải cao được hỗ trợ bởi Microsoft Windows.
- 16 MB RAM cho Microsoft Windows 95 hoặc 32MB RAM cho Microsoft Windows NT Workstation.

III. Làm quen với VB6

III.1 Bắt đầu một dự án mới với VB6

Từ menu Start chọn Programs, Microsoft Visual Basic 6.0. Khi đó bạn sẽ thấy màn hình đầu tiên như hình I.1 dưới đây.



Hình I.1 Cửa sổ khi kích hoạt VB6

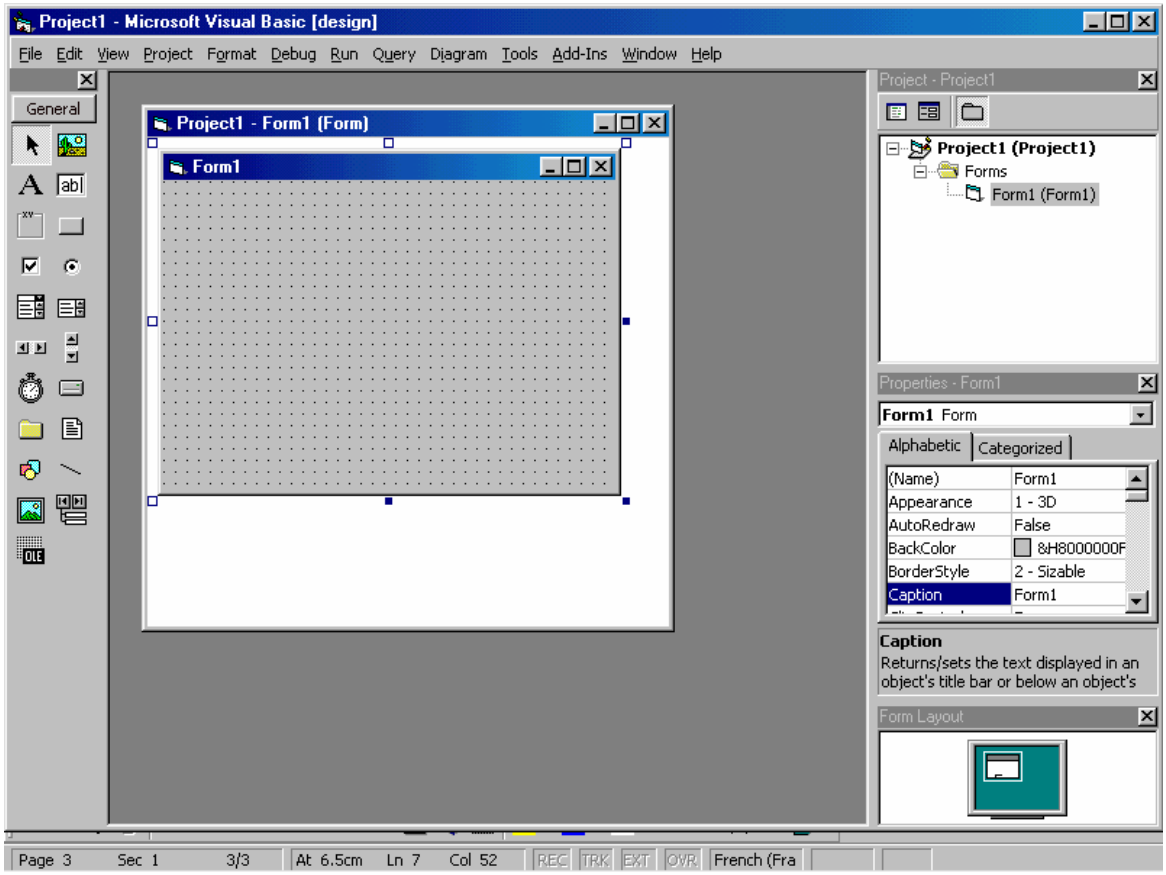
Ở đây, người dùng có thể chọn tạo mới một dự án thực thi được bằng cách chọn Standard EXE rồi nhấp Open (Hình I.2).

Tiếp theo là cửa sổ làm việc chính của VB6, gọi tắt là IDE (Integrated Development Environment) sẽ được giới thiệu chi tiết trong phần sau.

III.2 Tìm hiểu các thành phần của IDE

IDE là tên tắt của môi trường phát triển tích hợp (Integrated Development Environment), đây là nơi tạo ra các chương trình Visual Basic.

IDE của Visual Basic là nơi tập trung các menu, thanh công cụ và cửa sổ để tạo ra chương trình. Mỗi một thành phần của IDE có các tính năng ảnh hưởng đến các hoạt động lập trình khác nhau.



Hình I.3 Cửa sổ IDE của VB6

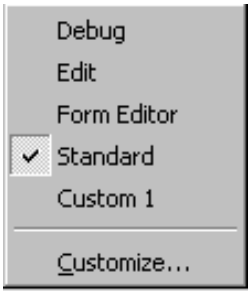
Thanh menu cho phép bạn tác động cũng như quản lý trực tiếp trên toàn bộ ứng dụng. Bên cạnh đó thanh công cụ cho phép truy cập các chức năng của thanh menu thông qua các nút trên thanh công cụ.

Các biểu mẫu (Form) - **khối xây dựng chương trình chính của VB** - xuất hiện trong cửa sổ Form. Hộp công cụ để thêm các điều khiển vào các biểu mẫu của đề án. Cửa sổ Project Explorer hiển thị các đề án khác nhau mà người dùng đang làm cũng như các phần của đề án. Người dùng duyệt và cài đặt các thuộc tính của điều khiển, biểu mẫu và module trong cửa sổ Properties. Sau cùng, người dùng sẽ xem xét và bố trí một hoặc nhiều biểu mẫu trên màn hình thông qua cửa sổ Form Layout.

III.3 Sử dụng thanh công cụ trong IDE của VB

Thanh công cụ là tập hợp các nút bấm mang biểu tượng thường đặt dưới thanh menu. Các nút này đảm nhận các chức năng thông dụng của thanh menu (New, Open, Save ...).

Hình I.4 Thanh công cụ ở dạng standard



Hình I.5 Popup menu thêm, xóa công cụ

Hơn nữa, người dùng có thể kéo rê thanh công cụ trên IDE đến vị trí bất kỳ nào đó thuận tiện cho việc sử dụng.

Người dùng có thể thêm hay xóa thanh công cụ trên IDE:

* Chọn Toolbars từ menu View hoặc ấn chuột phải vào điểm bất kỳ nào trên thanh menu, một popup menu bật ra.

* Chọn loại thanh công cụ mà ta muốn thêm vào hoặc xóa đi. Nếu có đánh dấu check ở bên trái thì loại công cụ đó đang được chọn.

Sử dụng thanh công cụ gỡ rối (debug)

Hình I.6 Thanh công cụ gỡ rối

Với thanh công cụ gỡ rối, người dùng có thể thực thi, tạm ngưng hoặc dừng một đề án. Với thanh công cụ Debug, người dùng có thể kiểm tra chương trình và giải quyết các lỗi có thể xảy ra. Khi gỡ rối chương trình, người dùng có thể chạy từng dòng lệnh, kiểm tra giá trị các biến, dừng chương trình tại một điểm nào đó hoặc với một điều kiện nào đó.

Sử dụng thanh công cụ Edit

Hình I.7 Thanh công cụ Edit

Thanh công cụ Edit được dùng để viết chương trình trong cửa sổ Code, thanh công cụ Edit có đầy đủ các tính năng của menu Edit. Ngoài ra người sử dụng có thể sử dụng chức năng viết chương trình tự động như là Quick Info.

Thanh công cụ Edit của VB6 có tính năng lý thú đó là tự hoàn tất các từ khóa. Tính năng này rất hữu dụng giúp cho người dùng tránh các lỗi mắc phải do gõ sai từ khóa.

Sử dụng thanh công cụ Form Editor

Hình I.8 Thanh công cụ thiết kế biểu mẫu

Thanh công cụ Form Editor có chức năng giống như menu Format dùng để di chuyển và sắp xếp các điều khiển trên biểu mẫu.

Trong quá trình thiết kế biểu mẫu, đôi khi chúng ta phải sử dụng thuộc tính ZOrder để cho phép một điều khiển có thể thay thế một điều khiển khác hay không hoặc là xuất hiện bên trên một điều khiển khác hay không.

Sử dụng hộp công cụ (Toolbox)

Hộp công cụ là nơi chứa các điều khiển được dùng trong quá trình thiết kế biểu mẫu. Các điều khiển được chia làm hai loại: Điều khiển có sẵn trong VB và các điều khiển được chứa trong tập tin với phần mở rộng là .OCX.

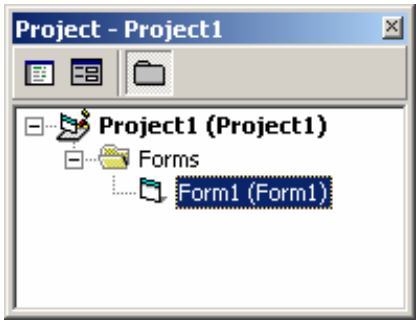
Đối với các điều khiển có sẵn trong VB thì ta không thể gỡ bỏ khỏi hộp công cụ, trong khi đó đối với điều khiển nằm ngoài ta có thêm hoặc xóa bỏ khỏi hộp công cụ.

Một điều khiển có thể được đưa vào biểu mẫu bằng cách chọn điều khiển đó và đưa vào biểu mẫu. Chúng ta sẽ trở lại phần này trong chương tiếp theo khi thiết kế các biểu mẫu.

Hình I.9 Hộp công cụ của Visual Basic

III.4 Quản lý ứng dụng với Project Explorer

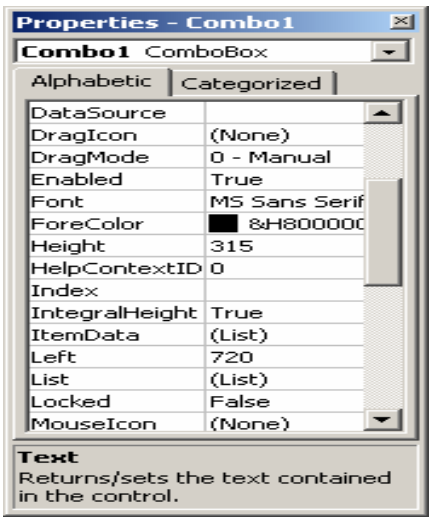
Project Explorer trong VB6 giúp quản lý và định hướng nhiều đề án.VB cho phép nhóm nhiều đề án trong cùng một nhóm. Người dùng có thể lưu tập hợp các đề án trong VB thành một tập tin nhóm đề án với phần mở rộng **.vbp**.



Hình I.10 Cửa sổ Project Explorer

Project Explorer có cấu trúc cây phân cấp như cây thư mục trong cửa sổ Explorer của hệ điều hành. Các đề án có thể được coi là gốc của cây, các thành phần của đề án như biểu mẫu, module ... là các nút của cây. Khi muốn làm việc với thành phần nào thì ta có thể nhấn đúp lên thành phần đó trên cửa sổ Project Explorer để vào cửa sổ viết code cho thành phần đó.

Khi làm việc với một dự án lớn, chúng ta sẽ thấy Project Explorer cực kỳ hữu ích cho việc tổ chức và quản lý một dự án lớn.



III.5 Cửa sổ Properties

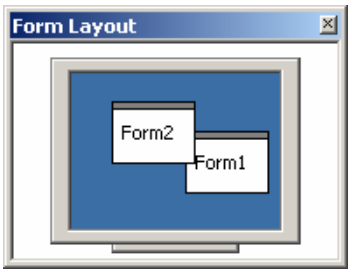
Mỗi một thành phần, điều khiển điều có nhiều thuộc tính. Mỗi một thuộc tính lại có một hoặc nhiều giá trị.

Cửa sổ Properties cho phép người dùng xem, sửa đổi giá trị các thuộc tính của điều khiển nhằm giúp điều khiển hoạt động theo đúng ý đồ của người sử dụng.

III.6 Cửa sổ Form Layout

Đây chính là cửa sổ trình bày biểu mẫu cho phép định vị trí của một hoặc nhiều biểu mẫu trên màn hình khi chương trình ứng dụng được thi hành.

Ta định vị một biểu mẫu trên màn hình bằng cách dùng chuột di chuyển biểu mẫu trong cửa sổ Form Layout.



Hình I.12 Cửa sổ Form

Sử dụng cửa sổ Form Layout không đơn giản như các cửa sổ khác vì nó không được kích hoạt sẵn, người dùng cần phải chạy ứng dụng sau đó mới có thể bố trí được các biểu mẫu thông qua Form Layout.

Nếu ta không định vị các biểu mẫu thì vị trí của biểu mẫu trên màn hình lúc thiết kế cũng là vị trí khởi động của biểu mẫu khi thực thi.

III.7 Biên dịch đề án thành tập tin thực thi

Sau khi đề án đã hoàn thành, người dùng có thể biên dịch thành tập tin thực thi được. Cách tiến hành như sau:

- Trước tiên ta cần chỉ cho VB6 biết phần chương trình nào sẽ được thực thi trước bằng cách chọn Project Properties từ menu Project. Chọn tab General, chú ý phần Startup Object, đây là nơi quy định điểm khởi đầu của chương trình sau khi biên dịch kết thúc.

- Từ menu File, chọn Make ... EXE... Một hộp thoại xuất hiện cho phép bạn nhập vào tên của tập tin thực thi. Bạn chỉ cần gõ tên tập tin, VB sẽ tự động thêm phần mở rộng .EXE.

- Nhấn vào nút Options để mở hộp thoại Project Properties và điền tên của ứng dụng vào ô Title, ta có thể ghi chú thông tin cho từng phiên bản trong phần Version Information. Ta có thể chọn Auto Increment để VB tự động tăng số Revision mỗi lần ta tạo lại tập tin EXE cho dự án.

- Cuối cùng, nhấn OK để trở về hộp thoại Make Project.

CHƯƠNG 2 BIỂU MẪU VÀ MỘT SỐ ĐIỀU KHIỂN THÔNG DỤNG

Mục tiêu:

Chương này giới thiệu về một số điều khiển cơ bản để tạo nên giao diện cho các ứng dụng cũng như một số khái niệm trong lập trình với VB; những yêu cầu tối thiểu cần có trong việc "lập trình sự kiện" với VB.

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

- Khái niệm về điều khiển, thuộc tính, phương thức, sự kiện.
- Quy tắc đặt tên danh biểu trong VB.
- Sử dụng biểu mẫu trong thiết kế giao diện.
- Sử dụng điều khiển ô nhập liệu, nút nhấn, nhãn, khung.

Kiến thức có liên quan:

- Cách thức sử dụng môi trường phát triển VB.

Tài liệu tham khảo:

- **Microsoft Visual Basic 6.0 và Lập trình Cơ sở dữ liệu** - Chương 2, trang 26; Chương 3, trang 29 - **Nguyễn Thị Ngọc Mai** (chủ biên), **Nhà xuất bản Giáo dục** - **2000**.

I. Các khái niệm

✓ **Điều khiển**: Các thành phần có sẵn để người lập trình tạo giao diện tương tác với người dùng.

Mỗi điều khiển thực chất là một đối tượng, do vậy nó sẽ có một số điểm đặc trưng cho đối tượng, chẳng hạn như các thuộc tính, các phương thức & các sự kiện.

✓ **Thuộc tính**: Các đặc trưng của một điều khiển tạo nên dáng vẻ của điều khiển đó.

✓ **Phương thức**: Các điều khiển có thể thực thi một số tác vụ nào đó, các tác vụ này được định nghĩa sẵn bên trong các phương thức (còn gọi là chương trình con: hàm & thủ tục), người lập trình có thể gọi thực thi các phương thức này nếu cần.

✓ **Sự kiện**: là hành động của người dùng tác động lên ứng dụng đang thực thi.

Thí dụ: - Nhấn phím bất kỳ trên bàn phím.

- Nhấp chuột.

Các thành phần giao diện có khả năng đáp ứng lại sự kiện. Chẳng hạn khi chúng ta nhấp chuột vào button, lúc đó button nhận biết được sự kiện này; hay như textbox nhận biết được sự kiện bàn phím tác động lên nó.

Một ứng dụng trên Windows thường được thực hiện nhờ vào việc đáp ứng lại các sự kiện của người dùng.

✓ **Lập trình sự kiện**:

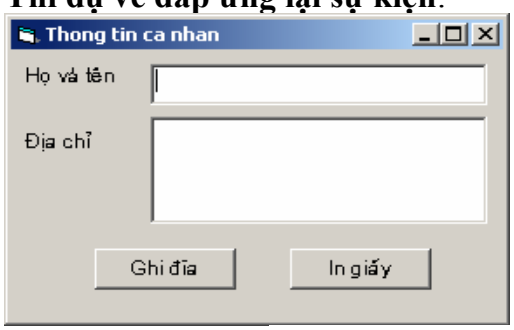
Các thành phần giao diện có khả năng nhận biết được các sự kiện từ phía người dùng. Tuy nhiên khả năng đáp ứng lại các sự kiện được thực hiện bởi người lập trình.

Khi một thành phần giao diện được sử dụng, người lập trình phải xác định chính xác hành động của thành phần giao diện đó để đáp ứng lại một sự kiện cụ thể. Lúc đó người lập trình phải viết đoạn mã lệnh mà đoạn mã lệnh này sẽ được thực thi khi sự kiện xảy ra.

Chẳng hạn, trong ứng dụng Paint của Windows; khi người sử dụng nhấp chuột vào nút vẽ hình elip sau đó dùng chuột vẽ nó trên cửa sổ vẽ, một hình elip được vẽ ra.

Trong lập trình sự kiện, một ứng dụng được xây dựng là một chuỗi các đáp ứng lại sự kiện. Tất cả các hành động của ứng dụng là đáp ứng lại các sự kiện. Do vậy người lập trình cần phải xác định các hành động cần thiết của ứng dụng; phân loại chúng; sau đó viết các đoạn mã lệnh tương ứng.

Thí dụ về đáp ứng lại sự kiện:



Mã lệnh

- Mã lệnh cho sự kiện Click của Ghi đĩa.

- Mã lệnh cho sự kiện Click của In giấy

Hình II.1: Thí dụ về đáp ứng sự kiện

- Khi người dùng không tác động vào ứng dụng, ứng dụng không làm gì cả.

- Khi người dùng nhập dữ liệu vào các ô nhập Họ và tên, Địa chỉ; sự kiện bàn phím xảy ra trên các ô nhập. Tuy nhiên, ứng dụng vẫn không làm gì cả vì không có đoạn mã lệnh nào đáp ứng các sự kiện này.

- Khi người dùng nhấp nút chọn Ghi đĩa, ứng dụng tìm kiếm trong mã lệnh của mình thấy có đoạn mã lệnh đáp ứng lại sự kiện này; lúc đó đoạn mã lệnh được thực thi.

- Tương tự như vậy đối với nút chọn In giấy.

✓ **Cách xác lập các thuộc tính & các phương thức trong chương trình**

<*Thuộc tính Name của điều khiển***>.<Tên thuộc tính>**

<*Thuộc tính Name của điều khiển***>.<***Tên phương thức***>[(<*Các tham số*>)]**

✓ **Tên điều khiển (thuộc tính Name)**

Đây là thuộc tính xác định tên của điều khiển trong ứng dụng. Tên này được đặt theo quy tắc:

- Tên có thể dài từ 1 - 40 ký tự.
- Tên phải bắt đầu với ký tự chữ, có thể chữ hoa hay thường.
- Sau ký tự đầu tiên, tên có thể chứa ký tự, số hay dấu gạch dưới.

Ví dụ: Num, StudentCode, Class12A2 là những tên hợp lệ. 345, 7yu là những tên không hợp lệ.

II. Biểu mẫu (Form)

II.1. Khái niệm:

Chương trình ứng dụng giao tiếp với người dùng thông qua các biểu mẫu (hay còn gọi là cửa sổ, xuất phát từ chữ Form hay Windows); các điều khiển (Control) được đặt lên bên trên giúp cho biểu mẫu thực hiện được công việc đó.

Biểu mẫu là các cửa số được lập trình nhằm hiển thị dữ liệu và nhận thông tin từ phía người dùng.

II.2. Thuộc tính

- **Name**: thuộc tính này như là một định danh nhằm xác định tên của biểu mẫu là gì? Ta sẽ sử dụng thuộc tính này để truy xuất đến các thuộc tính khác cùng với phương thức có thể thao tác được trên biểu mẫu.
- **Caption**: chuỗi hiển thị trên thanh tiêu đề của biểu mẫu.
- **Icon**: hình icon được dùng trong thanh tiêu đề của biểu mẫu, nhất là khi biểu mẫu thu nhỏ lại.
- **WindowState**: xác định biểu mẫu sẽ có kích thước bình thường (Normal=0), hay Minimized (=1), Maximized =(2).

- **Font**: xác lập Font cho biểu mẫu. Thuộc tính này sẽ được các điều khiển nằm trên nó thừa kế. Tức là khi ta đặt một điều khiển lên biểu mẫu, thuộc tính Font của điều khiển ấy sẽ tự động trở nên giống y của biểu mẫu.

- **BorderStyle**: xác định dạng của biểu mẫu.

II.3. Phương thức

- **Move**: di chuyển biểu mẫu đến tọa độ X,Y: **Move X, Y**.

II.4. Sự kiện

- **Form_Initialize**: Sự kiện này xảy ra trước nhất và chỉ một lần thôi khi ta tạo ra thể hiện đầu tiên của biểu mẫu. Ta dùng sự kiện Form_Initialize để thực hiện những gì cần phải làm chung cho tất cả các thể hiện của biểu mẫu này.

- **Form_Load**: Sự kiện này xảy ra mỗi lần ta gọi thể hiện một biểu mẫu. Nếu ta chỉ dùng một thể hiện duy nhất của một biểu mẫu trong chương trình thì Form_Load coi như tương đương với Form_Initialize.

  Ta dùng sự kiện Form_Load để *khởi tạo các biến, điều khiển* cho các thể hiện của biểu mẫu này.

- **Form_Activate**: Mỗi lần một biểu mẫu được kích hoạt (active) thì một sự kiện Activate phát sinh. Ta thường dùng sự kiện này để cập nhật lại giá trị các điều khiển trên biểu mẫu.

- **Form_QueryUnload**: Khi người sử dụng chương trình nhấp chuột vào nút **X** phía trên bên phải để đóng biểu mẫu thì một sự kiện QueryUnload được sinh ra. Đoạn chương trình con dưới đây mô tả thủ tục xử lý sự kiện QueryUnload.

```
Private Sub Form_QueryUnload(Cancel As Integer, _
        UnloadMode As Integer)


End Sub
```

  Sự kiện này cho ta khả năng hủy bỏ hành động đóng biểu mẫu bằng cách đặt lại Cancel là 1.

- **Form_Resize**: Sự kiện này xảy ra mỗi khi biểu mẫu thay đổi kích thước.

III. Nhãn (Label)

III.1. Khái niệm:

Nhãn là điều khiển dạng đồ họa cho phép người sử dụng hiển thị chuỗi ký tự trên biểu mẫu nhưng họ không thể thay đổi chuỗi ký tự đó một cách trực tiếp.

Biểu tượng (shortcut) trên hộp công cụ: 

III.2. Thuộc tính:

- **Name**: Đây là một tên xác định một định danh, người lập trình có thể thay đổi tên này theo cách của mình để tiện sử dụng.

- **Caption**: Thuộc tính quy định chuỗi ký tự hiển thị khi ta tạo một điều khiển nhãn. Khi ta tạo mới một điều khiển thì thuộc tính Caption có giá trị mặc nhiên là "`Label…`".

Ví dụ: Ta muốn tạo một nhãn là "Chào mừng bạn đến với Visual Basic", ta thay đổi giá trị của thuộc tính Caption thành "Chào mừng bạn đến với Visual Basic".

Ta có thể thay đổi giá trị của thuộc tính Caption tại thời điểm ứng dụng đang chạy nhờ vào đoạn mã lệnh đơn giản như sau:

`L1.Caption = "Đã đổi giá trị Caption"` với L1 là tên của điều khiển nhãn mà ta muốn đổi.

- **Font, Fore Color**: Quy định kiểu chữ, kích thước, màu hiển thị.
- **BackStyle, BackColor**: BackStyle quy định là nhãn trong suốt hay không. BackColor quy định màu nền của nhãn trong trường hợp không trong suốt.

III.3. Phương thức:

- **Move**: di chuyển nhãn đến tọa độ X,Y: **Move X, Y**.

III.4. Sự kiện:

- **Change**: Xảy ra mỗi khi nhãn thay đổi giá trị.
- **Click**: Mỗi khi nhãn được chuột nhấp lên, sự kiện này xảy ra.
- **DblClick:** Xảy ra khi người sử dụng nhấp đúp chuột lên điều khiển nhãn.

IV. Khung (Frame)

IV.1. Khái niệm:

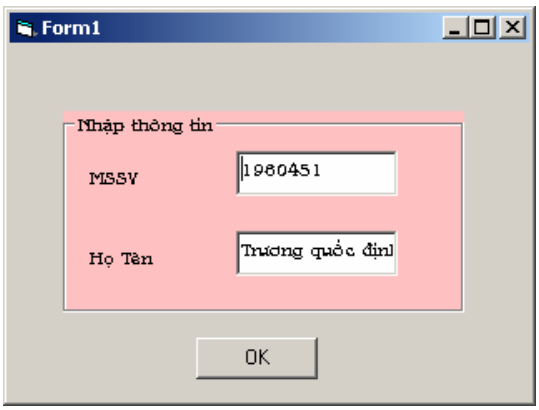
Khung là một điều khiển dùng trong việc bố trí giao diện của biểu mẫu một cách trong sáng và rõ nét. Thông thường các điều khiển cùng phục vụ cho một công việc nào đó sẽ được đặt trong một khung nhằm làm nổi bật vai trò của chúng.

Biểu tượng (shortcut) trên hộp công cụ: 

Khi chúng ta tạo mới một khung để chứa các điều khiển khác, ta có hai cách thực hiện:

- Tạo khung chứa trước, sau đó đưa các điều khiển vào trong khung chứa. Đây là cách đơn giản nhất.

- Tạo khung chứa sau khi đã tạo mới các điều khiển, khi đó khung chứa sẽ che mất các điều khiển, vì vậy ta cần phải đưa khung chứa ra sau các điều khiển bằng cách nhấp chuột phải và chọn **Send to Back**. Nhưng đối với cách này, các điều khiển khác không nằm trên khung chứa. Do vậy ta có thể giải quyết bằng cách *cắt (**Cut**) các điều khiển này đi, sau đó dán (**Paste**) vào trong khung chứa*.



Hình II.2 Ví dụ về khung chứa (Frame)

IV.2. Thuộc tính*:*

Khung cũng có các thuộc tính thông dụng như của điều khiển nhãn chẳng hạn như: Name, Caption,...

IV.3. Phương thức:

- **Move**: di chuyển khung đến tọa độ X,Y: **Move X, Y**.

IV.4. Sự kiện:

- **Click, DblClick**: xảy ra khi khung nhận được một thao tác nhấp (nhấp đúp) chuột.

V. Nút lệnh (Command Button)

V.1. Khái niệm:

Nút lệnh là một điều khiển dùng để bắt đầu, ngắt hoặc kết thúc một quá trình. Khi nút lệnh được chọn thì nó trông như được nhấn xuống, do đó nút lệnh còn được gọi là nút nhấn (Push Button). Người sử dụng luôn có thể chọn một nút lệnh nào đó bằng cách nhấn chuột trên nút lệnh đó.

Biểu tượng (shortcut) trên hộp công cụ: 

V.2. Thuộc tính:

- **Name**: sử dụng như một định danh nhằm xác định tên của nút lệnh.
- **Caption**: Dùng để hiển thị một chuỗi nào đó trên nút lệnh.
- **Default**: Nếu giá trị của thuộc tính này là True thì ta có thể chọn nút lệnh bằng cách nhấn phím Enter.
- **Cancel**: Nếu giá trị của thuộc tính này là True thì ta có thể chọn nút lệnh nào đó bằng cách nhấn phím ESC.
- **Enabled**: Trong một biểu mẫu, có thể có nhiều nút lệnh để thực hiện nhiều công việc khác nhau và tại một thời điểm nào đó ta chỉ được phép thực hiện một số công việc. Nếu giá trị thuộc tính Enabled là False thì nút lệnh đó không có tác dụng. Giá trị mặc định của thuộc tính này là True. Ta có thể thay đổi giá trị của thuộc tính tại thời điểm chạy ứng dụng.

- **ToolTipText**: cho phép hiển thị một đoạn văn bản chú thích công dụng của nút lệnh khi người sử dụng dùng chuột rê trên nút nhấn.
- **Font, Fore Color**: Quy định kiểu chữ, kích thước, màu hiển thị.

V.3. Phương thức

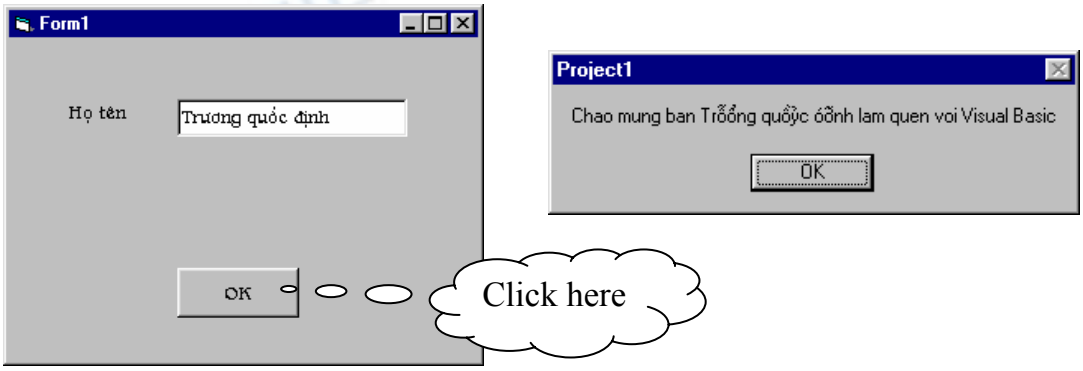
- **Move**: di chuyển nút lệnh đến tọa độ X,Y: **Move X, Y**.

V.4. Phương thức

- **Click**: đây là sự kiện thường xảy ra với nút lệnh. Mỗi khi một nút lệnh được chọn, sự kiện này được kích hoạt. Do đó, người sử dụng sẽ viết mã các lệnh để đáp ứng lại sự kiện này.

Ví dụ: Tạo một biểu mẫu có một ô nhập liệu với nhãn là họ tên và một nút lệnh cho phép đưa ra câu chào người dùng đó.

```
Private Sub Command1_Click()
      MsgBox "Chao mung ban " & Text1.Text & _
            " lam quen voi Visual Basic"
End Sub
```



Hình II.3 Sử dụng nút lệnh

VI. Ô nhập liệu (TextBox)

VI.1. Khái niệm:

Ô nhập liệu là một điều khiển cho phép nhận thông tin do người dùng nhập vào. Đối với ô nhập liệu ta cũng có thể dùng để hiển thị thông tin, thông tin này được đưa vào tại thời điểm thiết kế hay thậm chí ở thời điểm thực thi ứng dụng. Còn thao tác nhận thông tin do người dùng nhập vào dĩ nhiên là được thực hiện tại thời điểm chạy ứng dụng.

Biểu tượng (shortcut) trên hộp công cụ 

VI.2. Thuộc tính:

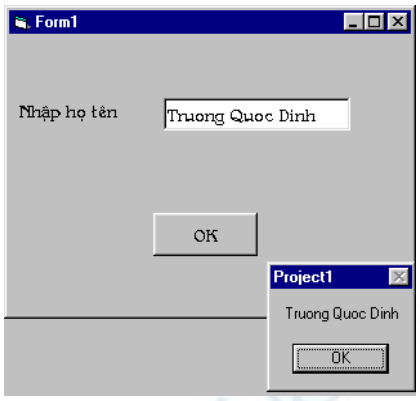
- **Name**: Đây là tên của ô nhập liệu, được sử dụng như một định danh.
- **MaxLength**: Thuộc tính quy định số ký tự tối đa có thể nhập vào ô nhập liệu. Nếu số ký tự nhập vào vượt quá số ký tự tối đa thì chỉ có đúng số ký tự tối đa được ghi nhận vào trong thuộc tính Text.

- **Text**: Dùng để nhập vào thông tin cần hiển thị trong Textbox tại thời điểm thiết kế hoặc nhận giá trị do người dùng nhập vào tại thời điểm chạy ứng dụng.

Ví dụ:

```
MsgBox Text1.Text
```

Đoạn mã này viết trong sự kiện Click của nút lệnh OK. Cho phép hộp thông báo hiển thị nội dung do người dùng nhập vào ô nhập liệu.



Hình II.3 Ví dụ về điều khiển ô nhập liệu

- **Locked**: Thuộc tính cho phép người dùng thay đổi nội dung của ô nhập liệu được hay không? Thuộc tính này có thể nhận 2 giá trị True hoặc False. Nếu False thì người dùng có thể thay đổi nội dung của ô nhập liệu & mặc định thì thuộc tính này có giá trị là False.

- **PasswordChar**: Thuộc tính này quy định cách hiển thị thông tin do người dùng nhập vào. Chẳng hạn, nếu ta nhập vào giá trị thuộc tính này là * thì các ký tự nhập vào điều hiển thị bởi dấu * . Thuộc tính này thường được dùng trong trường hợp thông tin nhập vào cần được che giấu (Ví dụ mật khẩu đăng nhập một chương trình ứng dụng nào đó mà trong đó các người dùng khác nhau thì có các quyền khác nhau).

- **Multiline**: Thuộc tính quy định ô nhập liệu có được hiển thị thông tin dưới dạng nhiều *hàng hay không*, nếu là TRUE thì ô nhập liệu cho phép nhiều hàng.

- **Font, Fore Color**: Quy định kiểu chữ, kích thước, màu hiển thị.

- **SelLength**:Cho phép trả về hoặc đặt trước số lượng ký tự được chọn trong ô nhập liệu.

- **SelStart**: Trả về hoặc xác định điểm bắt đầu của chuỗi được chọn. Đây là vị trí bắt đầu chèn một chuỗi mới trong trường hợp không có đánh dấu chọn chuỗi.

- **SelText**: Trả về hoặc xác định chuỗi ký tự được đánh dấu chọn, chỗi trả về sẽ là rỗng nếu như không đánh dấu chọn chuỗi nào.

Ba thuộc tính SelLength, SelStart, SelText chỉ có tác dụng tại thời điểm chạy ứng dụng.

VI.3. Phương thức

- **Move**: Di chuyển ô nhập liệu đến tọa độ X, Y: **Move X, Y**.

- **SetFocus**: Phương thức này nhằm mục đích thiết lập cho điều khiển ô nhập liệu nhận được Focus, nghĩa là nó sẵn sàng được tương tác bởi người sử dụng.

VI.4. Sự kiện:

- **KeyPress**: xảy ra khi người sử dụng chương trình nhấn một phím. Đối với điều khiển TextBox, ta thường dùng nó để lọc (filter out) các phím không chấp nhận. Sự kiện KeyPress cho ta một mã Ascii, một số có giá trị từ 0 đến 255, của phím vừa nhấn. Trong ví dụ dưới đây, TextBox Text1 sẽ chỉ nhận biết các phím là số (0 - 9), không nhận biết các phím khác:

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
 If KeyAscii < 48 Or KeyAscii > 57 Then ' Mã Ascii của  0 là 48, của 9 là 57
                KeyAscii = 0
 End If
End Sub
```

- **KeyDown, KeyUp**: mỗi sự kiện KeyPress lại cho ta một cặp sự kiện KeyDown/KeyUp. Sự kiện KeyDown/KeyUp có 2 tham số là KeyCode và Shift. Sự kiện này cho phép ta nhận biết được các phím đặc biệt trên bàn phím. Trong ví dụ dưới đây, ta hiển thị tên các phím chức năng mà người sử dụng chương trình nhấn vào:

```
Private Sub Text3_KeyDown(KeyCode As Integer, Shift As Integer)
        If (KeyCode >= 112) And (KeyCode <= 123) Then
                MsgBox "Ban vua nhan phim chuc nang: F" & _
                        Trim(Str(KeyCode - 111))
        End If
  End Sub
```

CHƯƠNG 3: LẬP TRÌNH CẤU TRÚC TRONG VISUAL BASIC

Mục tiêu:

Chương này giới thiệu về các cấu trúc lập trình trong VB; đây là các cấu trúc cốt lõi để xây dựng nên một chương trình VB.

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

- Sử dụng môi trường lập trình VB để viết mã lệnh.
- Các kiểu dữ liệu trong VB.
- Cách khai báo hằng, biến trong VB.
- Biểu thức trong VB.
- Các câu lệnh đơn cũng như các câu lệnh có cấu trúc.
- Chương trình con trong VB.
- Bẫy lỗi trong VB.

Kiến thức có liên quan:

- Cách sử dụng môi trường phát triển của VB.

Tài liệu tham khảo:

- **Microsoft Visual Basic 6.0 và Lập trình Cơ sở dữ liệu** - Chương 4, trang 49 - **Nguyễn Thị Ngọc Mai** (chủ biên), **Nhà xuất bản Giáo dục** - **2000**.

I. Môi trường lập trình

I.1. Soạn thảo chương trình:

Trong Visual Basic IDE, cửa sổ mã lệnh (Code) cho phép soạn thảo chương trình. Cửa sổ này có một số chức năng nổi bật:

- Đánh dấu (Bookmarks): Chức năng này cho phép đánh dấu các dòng lệnh của chương trình trong cửa sổ mã lệnh để dễ dàng xem lại về sau này. Để bật tắt khả năng này, chọn Bookmarks từ menu Edit, hoặc chọn từ thanh công cụ Edit.
- Các phím tắt trong cửa sổ mã lệnh:

Chức năng	Phím tắt
Xem cửa sổ Code	F7
Xem cửa sổ Object Browser	F2
Tìm kiếm	CTRL+F
Thay thế	CTRL+H
Tìm tiếp	SHIFT+F4
Tìm ngược	SHIFT+F3
Chuyển đến thủ tục kế tiếp	CTRL+DOWN ARROW
Chuyển đến thủ tục trước đó	CTRL+UP ARROW
Xem định nghĩa	SHIFT+F2
Cuộn xuống một màn hình	CTRL+PAGE DOWN
Cuộn lên một màn hình	CTRL+PAGE UP
Nhảy về vị trí trước đó	CTRL+SHIFT+F2
Trở về đầu của mô-đun	CTRL+HOME
Đến cuối mô-đun	CTRL+END

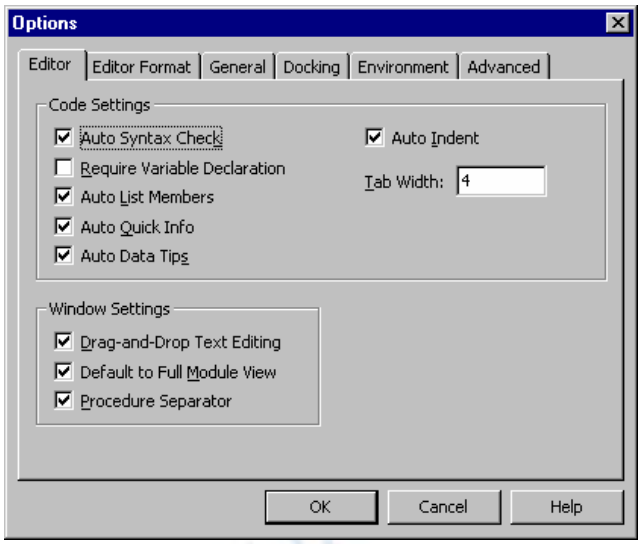
I.2. Các chức năng tự động:

- Tự động kiểm tra cú pháp (Auto Syntax Check)

Nếu chức năng này không được bật thì khi ta viết một dòng mã có chứa lỗi, VB chỉ hiển thị dòng chương trình sai với màu đỏ nhưng không kèm theo chú thích gì và tất nhiên ta có thể viết tiếp các dòng lệnh khác. Còn khi chức năng này được bật, VB sẽ cho ta biết một số thông tin về lỗi và hiển thị con trỏ ngay dòng chương trình lỗi để chờ ta sửa.

- Yêu cầu khai báo biến (Require Variable Declaration)

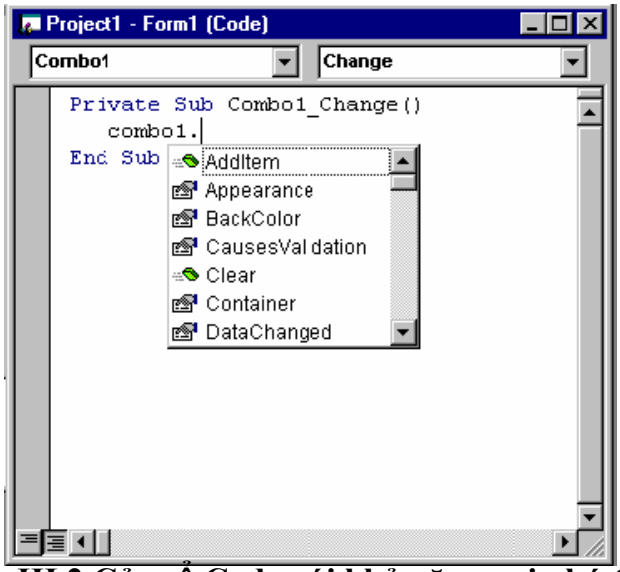
VB sẽ thông báo lỗi khi một biến được dùng mà không khai báo và sẽ chỉ ra vị trí của biến đó.



Hình III.1: Cửa sổ Options

- Gợi nhớ mã lệnh (Code):

Khả năng Auto List Members: Tự động hiển thị danh sách các thuộc tính và phương thức của 1 điều khiển hay một đối tượng khi ta gõ vào tên của chúng. Chọn thuộc tính hay phương thức cần thao tác và nhấn phím Tab hoặc Space để đưa nó vào chương trình.



Hình III.2 Cửa sổ Code với khả năng gợi nhớ Code

II. Kiểu dữ liệu

II.1. Khái niệm

Kiểu dữ liệu là một tập hợp các giá trị mà một biến của kiểu có thể nhận và một tập hợp các phép toán có thể áp dụng trên các giá trị đó.