

Bio 595

**Computers in
Biomedical Research**

Class notes set 3

Fall 2003

VB Programming

Basic steps in the VB Programming Process:

- 1. Start Visual Basic.**
- 2. Create a new application or load an existing application.**
- 3. Use debugging tools to help locate and eliminate program errors (bugs).**
- 4. Revise your program as needed into a final version.**
- 5. Quit Visual Basic.**
- 6. Submit the application to your instructor.**

Mastering VB Development Environment

Starting Visual Basic from Windows Start menu

Select Visual Basic 6.0 from submenu: Visual Basic loads and appears on your screen

Dialog box appears on startup

Start VB Application Wizard, edit an existing VB project, or select from a list of recent projects

Close the dialog box and the regular Visual Basic screen appears

From this Visual Basic development environment you create your VB Windows programs

Stopping Visual Basic

Select File / Exit command; or

Click VB's main window close button; or

Press Alt+F4; or

Double-click VB's Control menu icon in upper left corner of the screen.

Visual Basic gives you a last chance to save work before quitting.

Don't power off the computer without completely exiting Visual Basic.

Menu Bar, Toolbar

Many of the menu options have shortcut (or accelerator) keys like Ctrl+S for the File/Save.

The toolbar provides one-button access to common menu commands. Visual Basic supports a wide range of toolbars. (See View/Toolbars for a list of available toolbars)

Form location and form size coordinates shown at the left of the toolbar buttons.

Measurements in “twips” (= 1,440th inch, the smallest screen measurement) Coordinates determine where Form window appears, and what its size is. Twip values are in x,y pairs.

The Form Window (1)

This is the primary work area and the "background" screen for your application.

The programs you write consist of a form window and objects tied together with code.

The Form window holds the program's interactive objects, such as command buttons, labels, text boxes, scrollbars, and other controls.

The code appears elsewhere in a special window called the Code window.

Select View/Code to see the Code window.

The Form Window (2)

An application has a different appearance in its “design-time” state than in its “run-time” state.

During design time you design, create, edit, and correct the application.

When you run the application, the results of your work appear in the run-time state.

The program you will produce will consist of code, forms, menus, graphics, and help files that you create and edit to form the project (also called source code).

The Form Window (3)

The parts of the application that you create, such as the forms, the code, and the graphics that you prepare for output, comprise the “source” program.

When you compile or run the source program, VB translates the program into an “executable” or “object” program. You cannot make changes directly to an executable program. If you see bugs when you run the program, you must change the source application and rerun or recompile the source.

The Toolbox (1)

Contains the control types that you can place on the Form window.

All of the controls used in a typical application already appear in the toolbox and are called the “intrinsic” controls - they are found in all four versions of VB..

Extra tools come with all editions of VB, but these do not appear on the Toolbox window until you add them through the Project/Components menu option.

The Toolbox (2)

Controls on the left half of the dialog box, from the top down, include: pointer, label, frame, check box, combo box, horizontal scrollbar, timer, directory list box, shape, image, and OLE.

Controls on the right half of the dialog box, from the top down, include: picture box, text box, command button, option button, list box, vertical scrollbar, drive list box, file list box, line, and data.

The Form Layout Window

Displays the initial position and relative size of the current form shown in the Form window.

You can move the form inside the Form Layout window to move the form's location when the application is run.

Note that the form location indicators, to the right of the toolbar buttons, change when you move the form in the Form layout window.

You can display the Form Layout window from the View menu; you can hide the Form Layout window by clicking its window close button.

The Project Explorer Window

This Project window gives a tree-structured view of all the files in your application. The window displays forms, modules (files that hold supporting code for the application), classes (advanced modules), and more.

When you want to work with a particular part of the loaded application, double-click the component in the Project window to bring that component into view.

The Properties Window

“Properties” are detailed descriptive information about a control.

A different list appears in the Properties window for each different Form window tool.

The Properties window gives descriptive and functional information about the controls on the form.

Visual Basic's Online Help System

Press F1 to get help with a control, window, tool, or command. Visual Basic analyzes what you are doing and offers contextual help.

In addition, Visual Basic supports a help resource called Books Online. (electronic books about VB)

Selecting Books Online from the Help menu gives a tree-structured view of books about Visual Basic that you can search and read.

Online help also extends to the Internet, if you have a connection and a licensed copy of VB.

VB is Event-Driven

**Programs offer choices of what to do next:
menu options, some controls, data-entry, etc.**

**“Events” include a key-press, mouse movement,
etc. VB programs consist of a group of coded
routines to handle these events.**

**When the program runs, specific event
procedures execute only if that particular
event occurs.**

**Every control placed on a form window supports
one or more events. The written code “under”
that object executes as soon as the event
occurs.**

Analyzing a sample application

Check out the sample application “Chap2example.vbp”, a simple program calculating average value of a laboratory measurement taken five times.

This "project" consists of several files shown in the Project window.

VB will assign default names to objects you use in the program you write: "Command1" and "Label2" are assigned automatically by VB but can be renamed to suggest the object's purpose.

The six textboxes are Text1 through Text6; the seven labels are called Label1 through Label7. The two command buttons are Command1 and Command2.

VB Code (1)

In the Design Time state, select a control by clicking on it once to make that object's property list appear in the Properties window.

Program code for this project can be found in the Code window.

This window behaves like a simple text editor, with standard functions like insert, delete, copy, cut and paste mouse-selected text.

Look at the code executed when a particular object is selected by double-clicking it during Design Time.

Code accompanying an object like a command button are "wrapper lines" of code - the first and last lines of code.

VB Code (2)

Wrapper lines are automatically generated when a new object is placed on a form of an application project you are writing. Typical event wrapper lines are "Private Sub controlName_eventName()", ending with "End Sub"; the specific code you enter is to be sandwiched between these two lines.

Event procedures (code specifying how an object is to respond to being activated) always consist of the control name, an underscore, and the procedure's event name.

The top of a code window has a drop-down menu with all possible events listed.

Sample Program (1)

Examine features of the simple biosciences laboratory program by selecting objects on the form window and examining their properties.

Select the “Exit Program” button (in Design Time mod to see how simple coding a command button event can be: this one quits the program and returns VB to its Design Time mode, using the BASIC keyword “End”).

Sample Program (2)

Look at code behind the “Find result” button: a simple calculation to find the average of the five values entered in textboxes 1 through 5.

The 5 variables, X1 through X5, are set equal to the text entered in Textboxes 1 through 5.

Text in a textbox is not a number; the function VAL converts each into a numeric value that VB can manipulate in a calculation.

The code performing the calculation is written like a simple math formula: The variable average_X is set equal to the sum of X1 to X5, divided by 5.

The final line of code places this numeric value into the text property of textbox 6 (“Text6.text”)

Generating a program from scratch

Try using the VB Application Wizard to generate a skeleton (or "shell") of a program.

Respond to a series of dialog boxes.

Select File/New Project.

Double-click the Application Wizard icon.

Answer the questions asked by the Wizard; use default (computer-assigned) values.

Examine the program the Wizard has generated.

Try its controls and menus.

Creating a new application

- Place controls on form window by double-clicking a control to put it in the center of the forms window where you can drag it to any desired location, or**
- Click the control, then click the crosshair cursor where you want the control located on the form. Size the control by dragging one of its eight “sizing handles”**
- Move multiple commands by drawing a rectangle around them**
- Now create your program's interface: scroll the Properties window.**
- Note that the form itself has its own properties**
- Use an appropriate 3-letter prefix in naming the controls (optional, not mandatory).**

Standard object name prefixes

| | | | |
|------------|-----------------------------|------------|---------------------------|
| cbo | Combo box | lbl | Label |
| chk | Check box | lin | Line |
| cmd | Command button | lst | List box |
| dir | Directory list box | mnu | Menu |
| drv | Drive list box | ole | OLE client |
| fil | File list box | opt | Option button |
| fra | Frame | pic | Picture box |
| frm | Form | shp | Shape |
| grd | Grid | tmr | Timer |
| hsb | Horizontal scrollbar | txt | Text box |
| img | Image | vsb | Vertical scrollbar |

Built-in "help"

“Tooltips” are clues about the function of a control entered as text in the control’s “ToolTipText” property.

The tip appears as the mouse is run over the button.

Ellipses (...) following choices indicate that a dialog box will be displayed if choice is selected

Create your first application from scratch

**Create a new project (File | New Project, .EXE option
Select the Label control and drag a Label control to top
of Form window**

**Change the label's Name property to "lblFirst" and its
Caption to "Hello, World!"**

Increase font size (Font property) to 24 Bold

Add a Command button

Set the button's Name to cmdExit

**Note the displayed caption has the "x" underlined,
permitting the Alt+x keyboard command as a way to
activate the button in addition to clicking on it with the
mouse**

Your first application...

Write a command procedure for the command button

Double-click the button to open its Code window

Window will show default “wrapper” lines (first and last lines of code) as follows:

```
Private Sub cmdExit_Click()
```

```
End Sub
```

Enter the word “End” between these “wrapper lines”

Save your program

Run the program!

Control Focus

= the control active at any moment of program execution

Window or form currently highlighted (blue title bar) has the focus

Control with highlighted caption or outline has the focus

This determines what the next keystroke will activate

Command Buttons

To add a button to an application, locate and size the button; set its name and caption

Add code to the button's Click event procedure.

The click event is one out of a potential 15 different events

All properties can be set at design time; some of these can be set at runtime.

Command Button Properties (1)

BackColor Specifies the command button's background color. Click the BackColor's palette down arrow to see a list of colors and click Categorized to see a list of common Windows control colors. Before the command button displays the background color, you must change the style property from 0-Standard to 1-Graphical.

Cancel Determines whether the command button gets a Click event if the user presses the esc key.

Caption Holds the text that appears on the command button.

Default Determines if the command button responds to an Enter keypress even if another control has the focus.

Command Button Properties (2)

Enabled Determines whether the command button is active. You can change the Enabled property at runtime with code when a command button is no longer needed and you want to gray out the command button.

Font Produces a Font dialog box in which you can set the caption's font name, style, and size.

Height Holds the height of the command button in twips.

Left Holds the number of twips from the command button's left edge to the Form window's left edge.

MousePointer Determines the shape of the mouse cursor when the user moves the mouse over the command button.

Command Button Properties (3)

Picture Holds the name of an icon graphic image that appears on the command button as long as the **Style** property is set to 1-Graphical.

Style Determines whether the command button appears as a standard Windows command button (if set to 0 -standard) or a command button with a color and possible picture (if set to 1 -Graphical).

TabIndex Specifies the order of the command button in the focus order.

TabStop Determines whether the command button can receive the focus.

Command Button Properties (4)

ToolTipText Holds the text that appears as a tooltip at runtime.

Top Holds the number of twips from the command button's top edge to the Form window's top edge.

Visible Determines whether the command button appears or is hidden from the user. (Invisible controls cannot receive the focus until the running code changes the Visible property to True.)

Width Holds the width of the command button in twips.

Labels

These objects contain the text displayed on a form

If a label is given text too long (number of characters) or too large (font size) to fit within the label boundaries, the text may be truncated or the text field may grow downwards or to the right, depending on property settings.

Set `AutoSize` and `WordWrap` properties to `True` to account for this possibility. You don't need to worry about this if the label isn't going to change during program execution.

Common Label Properties (1)

Alignment Determines whether the label's caption appears left-justified, centered, or right-justified within the label's boundaries.

AutoSize Enlarges the label's size properties, when True, if you assign a caption that is too large to fit in the current label's boundaries at runtime.

BackColor Specifies the label's background color. Click the BackColor's palette down arrow to see a list of colors and click Categorized to see a list of common Windows control colors.

BackStyle Determines whether the background shows through the label or if the label covers up its background text, graphics, and color.

Common Label Properties (2)

BorderStyle Determines whether a single-line border appears around the label.

Caption Holds the text that appears on the label.

Enabled Determines whether the label is active. Often, you'll change the Enabled property at runtime with code when a label is no longer needed.

Font Produces a Font dialog box in which you can set the caption's font name, style, and size.

ForeColor Holds the color of the label's text.

Height Holds the height of the label's outline in twips.

Common Label Properties (3)

Left Holds the number of twips from the label's left edge to the Form window's left edge.

MousePointer Determines the shape of the mouse cursor when the user moves the mouse over the label.

TabIndex Specifies the order of the label in the focus order. Although the label cannot receive the direct focus, the label can be part of the focus order.

ToolTipText Holds the text that appears as a tooltip at runtime.

Top Holds the number of twips from the label's top edge to the Form window's top edge.

Visible Determines whether the label appears or is hidden from the user.

Width Holds the width of the label in twips.

Common Label Properties (4)

WordWrap Determines whether the label expands to fit whatever text appears in the caption.

Left Holds the number of twips from the label's left edge to the Form window's left edge.

MousePointer Determines the shape of the mouse cursor when the user moves the mouse over the label.

TabIndex Specifies the order of the label in the focus order. Although the label cannot receive the direct focus, the label can be part of the focus order.

ToolTipText Holds the text that appears as a tooltip at runtime.

Common Label Properties (5)

Top Holds the number of twips from the label's top edge to the Form window's top edge.

Visible Determines whether the label appears or is hidden from the user.

Width Holds the width of the label in twips.

WordWrap Determines whether the label expands to fit whatever text appears in the caption.

Text Boxes (1)

These objects receive user keyboard input.

Note that text boxes don't have a "Caption" property.

(Text can be included in the box before the program runs, so captions aren't needed.)

Text Boxes (2)

Alignment Determines whether the text box's text appears left-justified, centered, or right-justified within the text box's boundaries.

BackColor Specifies the text box's background color. Click the BackColor property's palette down arrow to see a list of colors and click Categorized to see a list of common Windows control colors.

BorderStyle Determines whether a single-line border appears around the text box.

Enabled Determines whether the text box is active. You can change the Enabled property at runtime with code when a text box is no longer needed.

Font Produces a Font dialog box in which you can set the Text property's font name, style, and size.

Text Boxes (3)

ForeColor Holds the color of the text box's text.

Height Holds the height of the text box's outline in twips.

Left Holds the number of twips from the text box's left edge to the Form window's left edge.

Locked Determines whether the user can edit the text inside the text box that appears.

MaxLength Specifies the number of characters the user can type into the text box.

MousePointer Determines the shape of the mouse cursor when the user moves the mouse over the text box.

Text Boxes (4)

MultiLine Lets the text box hold multiple lines of text or sets the text box to hold only a single line of text. Add scrollbars if you wish to put text in a multiline text box so a user can scroll through the text.

PasswordChar Determines the character that appears in the text box when the user enters a password (keeps prying eyes from knowing what the user enters into a text box).

ScrollBars Determines whether scrollbars appear on the edges of a multiline text box.

TabIndex Specifies the order of the text box in the focus order.

TabStop Determines whether the text box can receive the focus

Text Boxes (5)

Text Holds the value of the text inside the text box. The Text property changes at runtime as the user types text into the text box. If you set an initial Text property value, that value becomes the default value that appears in the text box when the user first sees the text box.

ToolTipText Holds the text that appears as a tooltip at runtime.

Top Holds the number of twips from the text box's top edge to the Form window's top edge.

Visible Determines whether the text box appears or is hidden from the user.

Width Holds the width of the text box in twips

Form Properties (1)

BackColor Specifies the form's background color. Click the BackColor's palette down arrow to see a list of colors and click Categorized to see a list of common Windows control colors.

BorderStyle Determines how the Form window appears. The BorderStyle property specifies whether the user can resize the form and also determines the kind of form you wish to display.

Caption Displays text on the form's title bar at runtime.

ControlBox Determines whether the form appears with the Control menu icon. The Control menu appears when your application's user clicks the Control menu icon.

Form Properties (2)

Enabled Determines whether the form is active. Often, you'll change the Enabled property at runtime with code when a form is no longer needed. Generally, only multiform applications, such as MIDI applications, need to modify a form's Enabled property.

Font Produces a Font dialog box in which you can set the text's font name, style, and size.

ForeColor Holds the color of the form's text.

Height Holds the height of the form's outline in twips.

Form Properties (3)

Icon Describes the icon graphic image displayed on the taskbar when the user minimizes the form.

Left Holds the number of twips from the form's left edge to the screen's left edge.

MaxButton Specifies whether a maximize window button appears on the form.

MinButton Specifies whether a minimize window button appears on the form.

MousePointer Determines the shape of the mouse cursor when the user moves the mouse over the form.

Moveable Specifies whether the user can move the form at runtime.

Picture Determines a graphic image that appears on the form's back-ground at runtime.

Form Properties (4)

ScaleMode Determines whether the form's measurements appear in twips, pixels (the smallest graphic dot image possible), inches, centimeters, or other measurements.

ShowInTaskbar Determines whether the form appears on the Windows taskbar.

StartPosition Determines the state (centered or default) of the form at application startup.

Top Holds the number of twips from the form's top edge to the Form window's top edge.

Visible Determines whether the form appears or is hidden from the user.

Width Holds the width of the form in twips.

WindowState Determines the initial state (minimized, maximized, or normal) in which the window appears at runtime.