Module 3

INTRODUCTION TO VISUAL BASIC

Introduction

Unlike structured (traditional) programming where program design is based on flowcharting, top-down design etc, VB is object oriented i.e., it resolves around ready made objects, and it is event driven i.e., all the activities in a program are triggered by one event or another.

Objectives

At the end of this module, you should be able to

- Explain Visual basic concepts
- Get started with Visual basic
- Describe the visual basic program structure

What is Event Driven Programming?

All windows applications have a common graphic user interface (GUI) and program execution is based on events -Event driven programming i.e., instead of writing a program that executes from top to bottom, you design an application that responds to events, such as key press etc.

In this type of programming, the programmer's job is to create objects and then determine its ***properties*** and ***behaviours***. Each object has its own ***event-handling*** procedures and the system knows all about this. For example it knows what a button is and how it works. As such, the programmer does not have to write code to trap these events-The system does that automatically.

In VB program design, the programmers begins by creating the screen layout (the user-interface in the jargon), and work outwards from here, adding first the code that will run in response to specific events and then any necessary code to co-ordinate the whole program.

1.0 Getting started with VB

There are 3 main steps to creating an application in VB namely:
1. Create the interface
2. Set properties of the controls
3. Write the code to handle the events

- The following is the procedure in developing VB programming

1. *Plan the program by*
   - *Identify your objectives i.e., what you want your program to do*
   - *Identify the inputs*
   - *Identify the processing*
   - *Identify the output*

2. *Build the program by*
   - *Create the controls (GUI)*
   - *Set the control properties*
   - *Write the codes (to process data, test for conditions or change the order of carrying out instructions)*

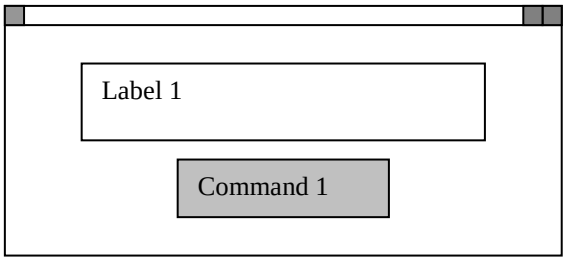
3. *Test, Compile and Run the program*
   - *Run the program*
   - *Test and Debug the program*
   - *Save the program/Create an Executable file*

- Follow the procedure below to get started with VB

Procedure
1. Click on start, select programs, select MS visual studio, select VB 6.0
2. Click the open button to start a new project, select *standard.exe project*
3. Drag and drop the objects that will make up the user interface
4. Set the properties for each object, where necessary, to change its appearance and behaviours
5. Attach a VB code to each object as necessary
6. Run the program
7. Save form (using save as)
8. Save project (Use save project as)

Activity 3.1

1. List and explain the THREE aspects of an object
2. Assemble the following objects on the form. A label and a command button. Adjust their positions and sizes appropriately with the pointer so that your layout looks like in the figure below.



3. Explore the properties of the controls by
   (a) Change the labels caption to read "Your Name", font 18, and change the fore and back colours from palette to any other
   (b) Change the buttons caption to read "Click"

2.0 VB Concepts

(a) The working screen

When you load up VB, the initial screen will look like something in Fig. 3.1. The main component window has 3 main windows: the form window, project window, the properties window and control/toolbox window

- The VB desktop also features the menu bar. The menu bar gives access to the commands (together with keyboard shortcuts and icon equivalents) used at the design stage

(a) Form window

It is a window, initially blank, on which you paste controls to create your screen display or print display

- Forms can be any size, color and you can attach to it code that will run when the form is loaded, closed, or when a mouse is clinked or moves over it.
- A program may use one or several forms -each of which will handle a different part of the program
- Each form is saved on disk as a separate file, with a .FRM extension (see Fig. 3.1)

(b) Project window

- The project holds together the various forms and modules that make up a program. Its purpose is primarily one of convenience. When you want to start work on a program, you only have to open the one project file-marked by a .MARK extension - rather than a whole set of forms and modules

  Used to move between forms and modules and between form and code views. The VB eXtension program are also shown here with .VBX files.

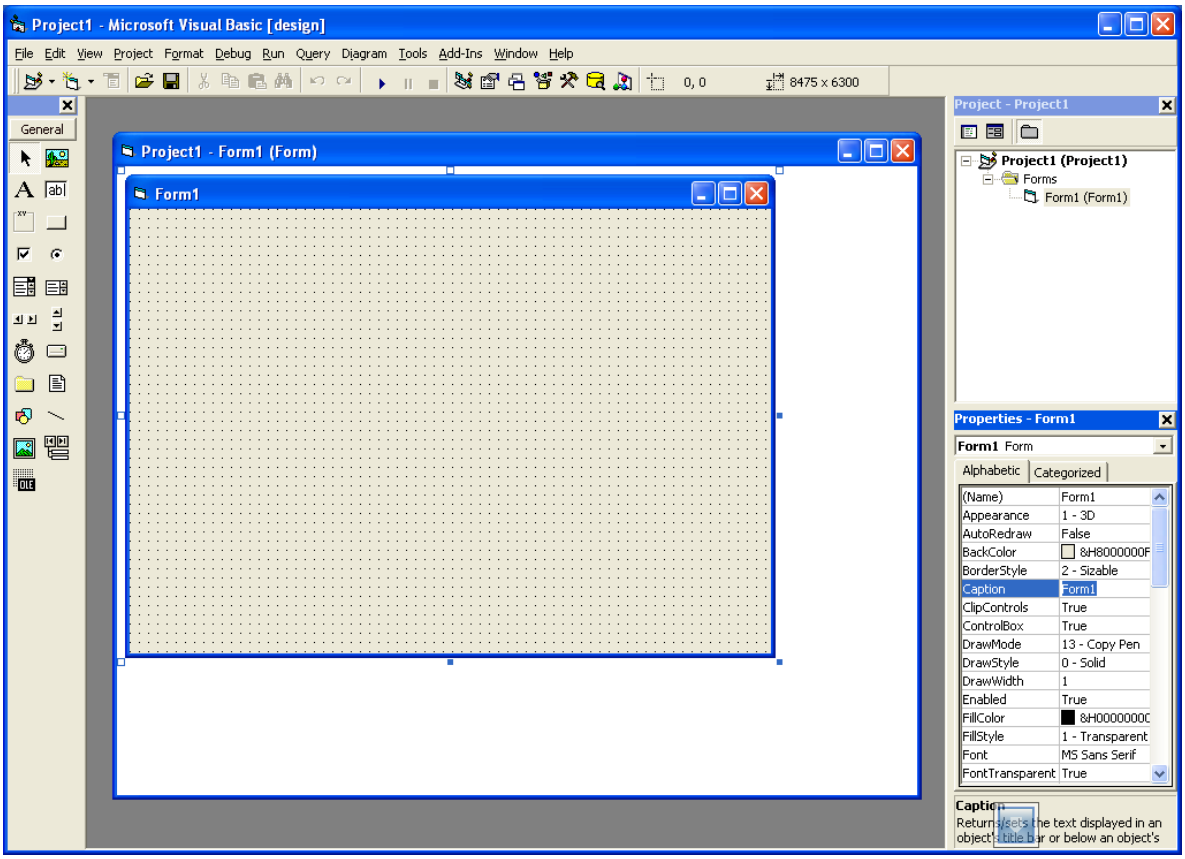


Fig. 3.1. The VB desktop showing the main component windows

(c) Properties window

This is where you set properties of objects

To set properties for objects

⇒ Click on the object

⇒ Select properties to be set from the list (see example Fig. 3.1). Properties that can be set include caption, font, colour, name etc.

NB:

- You can set object properties when you are designing your program (at design time) by selecting an object, clicking the properties window, and changing its property settings.
- You can also set object properties while your program runs (at run time) by modifying object properties with the VB program code
- Always change the controls' name from the default name to something meaningful. As prog gets more complex, memorable names become more useful

(d) Toolbox

It holds the standard controls -the basic building blocks used to create the user interface (see Fig. 3.2). Table 1 shows the functions of major controls

Controls are objects which can be pasted onto a form and range from labels which display text, through picture boxes for graphic images, to buttons, check boxes, lists and other menus, to file management utilities and spreadsheet-style grids. Each control may have code attached to it

- NB each control is an object and has
  (a) properties which determine its position, size, colour, appearance and nature of its text etc.
  (b) behaviours and event-handling procedures i.e., an object is able to react when an event occurs.

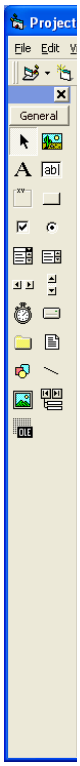


Fig. 3.2 Standard controls

Table 1: Controls and their functions

Control Name	Function
Pointer	Used for selecting objects that have been placed on a form
Picture box	Holds pictures created with paint brush or similar art packages. They must be in acceptable format ->BMP, .WMF, .ICO, or .DIP
Labels and text boxes	Both hold text but only textbox is capable of receiving input from user. Use labels for messages and prompts (text that can only be read e.g. display instructions to the user. Labels are also used to identify controls, such as text boxes and scroll bars, that do not have their own caption property Use text boxes for inputting data and replies e.g., password box etc.
Frame	Holds things together. NB once a control is placed on a frame it cannot be moved out onto the form and vice versa.
Command Buttons	Are one method of selection for limited options e.g., OK or CANCEL, START or QUIT.
Check boxes	Act as toggle switches, turning options on or off and allows user to choose one or more options
Option buttons	Used to select only one from a set of mutually exclusive options.
List boxes	Display a scrollable list of items from which the user can choose
Combo box	Combine a drop-down list with a slot in which users can enter their own data when the program is running
Horizontal/vertical scroll bars	Used on forms to give a very flexible means of setting values
Timer	Controls actions that must take place at or after set intervals
Shape (circle, line etc)	Used to display a single graphical element
Image	Limited version of a picture box
Data control	Allows you to read records in from a database
Common Dialog & Grid	Provides file management facilities. Grid gives you a simple way to create tables or even to write your own spreadsheets
OLE (object Linking and Embedding)	Allows you to make a two way link with an object in another windows application

(e) Module

- A set of program code attached to a form
- Modules are saved as files with a **.BAS** extension

(f) Procedures/subroutines

- All codes are written in procedures. Most procedures are attached to a control-or more accurately to an event belonging to a control
- All start with keyword **sub** and close with **Endsub** (see VB program structure for detail).

(g) Using focus

- Focus is the ability of an object to receive input through the mouse or keyboard. In Microsoft windows interface, several applications can be running at any time, but only the application with the focus will have an active tittle bar and can receive user inputs.

  Forms and controls support these events. For example, when a command button has the focus, it appears with a highlighted border around its caption.

- An object receives focus only when it is enabled and its ***visible property set to True***
- The enabled property allows the object to respond to user-generated events such as keyboard and mouse events
- The visible property determines whether an object is visible on the screen

NOTE:
A form can receive focus only if it doesn't contain any controls that can receive the focus.

3.0 VB program structure and features

The overall VB program structure shall mostly include the following features

(i) Procedures

- Each control has its own set of events. By double-clinking on a control, a **code window** opens up. It is here that you write the code to be attached to events
- Each event has its own procedure. General format for linking objects to events is

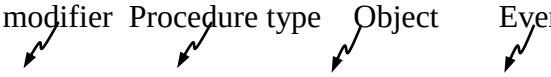
Modifier Procedure type Object type_Event

{ Code/statements to be executed in response to an event. Codes are concerned with changing property and take the format:
Object/Control Name.Property = New Value

End of procedure

NOTE:
Object = Name of object you are customizing
Value = New property settings
Property = A value or characteristic held by a VB object such as a caption or forecolour.

Example: modifier Procedure type Object Event



```
Private Sub    Command1_Click()
        Label1.Caption ="Jambo"
End Sub
```

Or

```
Sub Text1_Keypress(KeyAscii As Integer)
    .................
    Add code here
    .................
End Sub
```

In this case, variable names are used to carry information from the event through to the code. When this is triggered, KeyAscii will hold the Ascii code of the key that was pressed. The code might well check this to see if the user had pressed a specific key, or one from an acceptable range

Activity 3.2

You are going to add code to your controls in activity 3.1 so that when button is clicked, 3 messages appear on the screen. Add the following code

```
Label1.Caption = "Ouch"
MsgBox "Not that hard"
Form1.Print "Use Run I End to stop"
```

Run the program by selecting **Run/Start.** To stop the program use **Run/End.**

In activity 3.2, the first line changes value of a property. It assigns the text "Ouch" to the caption property of the label. The second uses the MsgBox statement. The third uses the print method to display a message on the form.

(ii) General declaration

- If the program will involve storing data temporarily, then declare when they are to be stored (variables) and the type of data to be stored. Variables are a special container that hold data temporarily in a program e.g., calculated results, file names etc. A variable must have a name and a data type associated with it.

- Declare all variables before use (in a procedure) using a ***Dim statement***. General format is

	Dimension (modifier)	variable Name	As	Variable Type
e.g.	DIM	NUM1	As	Integer
Or	DIM	SurName	As	String

Compare the example of page 24 and activity 3.4 for illustration

- By default, VB assigns a *variant data type* for variable names with undeclared data types (see section 4.0)

- Some numeric data types include
  Integer: whole numbers from -32, 768 to +32, 767
  Long:- Whole numbers range + or - 2 billion
  Single:- Floating point numbers to 7 digits
  Double:- As single, but held to 15 digit accurately
  String:- Text with up to 65,000 characters
  Currency:- Numbers in range + or - 900,000 billion, held to 19 -digit accuracy
- Variable and data type are covered in detail in Section 5.0

(iii) Statements

- Statements are a combination of keywords, identifiers, and arguments in the code that does the work of the program i.e., line of code e.g.,

```
For n = 1 to 10
Msg Box "Welcome to VB"
NEXT n
```

Note:

1. A long statement may be broken into multiple lines in the code window using the line-continuation character (a space followed by an underscore) as shown in example below. However, there are some limitations as to where the line continuation character.

```
MsgBox "Wellcome to VB"_
& "Please do not eat in class"
```

2. Two or more statements can be placed on one line using a colon (**:**) to separate them e.g.,

```
Text1.text = "HELLO":Red = 255
```

(iv) Comment lines

- Comment lines are added using the symbol (`).

NB. Comment lines are ignored by the VB compiler and only help guide the reader e.g.,

```
`This Program calculates the Net Pay of Employees
```

(v) Methods

- They link statements, perform actions (i.e., change property), but they can only be used with suitable objects.
- *A method is the action you want the object to perform. Value is an optional argument to be used by the method.*
- They have a general format:

Object . Method Value

E.g., Form.Print "Hello Mum"

Or Picture1.circle (x,y) radius

The first prints a greeting on form 1. You can only print on objects that can handle text, or draw circles in those that can handle graphics

(vi) Syntax

- It is a set of rules that govern how command lines must be written. They consist of keywords, essential and optional parameters and the punctuation. For example, the syntax for circle method may read:

[*Object.*]**Circle**[**Step**](*x,y*),*radius*[,[*colour*][,[*start*][,[*end*][,*aspect*]]]]

Note;

* Items in bold are reserved words
* Items in italics are parameters (include names of objects, variable names, number of string variables etc)
* Those enclosed in [ ] are optional, others are essential. In the above case, the object name is part of a routine attached to the object; step is an optional keyword that modifies the way the method works; in the parameters , x, y and radius are all essential; colour, start, end and aspect are all optional

(vi) Operations

Table A: Common arithmetic and logical operations of VB

Operator	***Operand***	***Description***	***sample expression***	***result***
=	Any	Assignment	x = 7, y = 3;	7 in x and 3 in y
+, -	Numbers	Addition, subtraction	3.0 + x	10
*, /	Numbers	Multiplication, division	7/2	3
<=		Less than or equal to		
>=		Greater than or equal to		
<>		Greater or less than		
NOT	Values	Reverse its values		
AND				
OR				
XOR				
EQU				

Naming the controls

- The **Name prefix** shown below is the standard prefix that you should use when naming controls
- The **Main property** is the one that can be omitted when collecting a value from a control. E.g., Text is the main property of a text box and the line *answer = Text1* has the same effect as *answer - Text1.text.*

Control	Name Prefix	Example	Main property	Default event
Check box	Chk	chkBold	Value	Click
Combo box	Cbo		Text	Change
Command button	Cmd		Value*	Click
Directory List box	Dir		Path*	Change
Drive list box	Drv		Drive*	Change
File list box	Fil		FileName*	Change
Form Frame	Frm		N/A	Load
Frame	Fra		Caption	DragDrop
Horizontal scroll bar	Hsb		Value	Change
Image	Img		Picture	Click
Label	Lbl		Caption	Click
List box	Lst		List(ListIndex)*	Click
Picture box	Pic		Picture	Click
Text box	Txt		Text	Change
Timer	Tmr		Enabled	Timer
Vertical scroll bar	Hsb		Value	Change
Option Button	Opt			
Ole	Ole			
Shape	Shp			
Line	Lin			
Vertical scroll bar				

- The **Default event** is the one that the code window will open onto when you first attach code to a control
- Properties marked * are only available during run-time.

Creating an Executable file

This is a stand-alone program that can run without starting the VB programming system

1. On the **File** menu, click **Make ProjectXXX.exe.**
2. To adjust the default settings, click **Options** in the **Make Project** dialog box that opens and make your choices
3. Specify a folder and file name for your executable file with the **Save** In drop-down box and file Name text box and click **OK** to compile your executable file.

Activity 3.3

1. Outline the 3 main steps followed in developing an application program in Visual basic

2. Set up a form containing a textBox, a label and a button. Write a caption on the label asking the user to enter a name. Edit the textBox's Text property to leave it blank.
   (a) Attach code to the button so that when the button is clicked, the label displays the user's name.
   (b) Add a button hat will end the program when clicked [NB. the only word needed is **End**].

3. Set up a new form containing one label and three command buttons. Edit the button captions to read "Stop", "Go" and "End". Attach code to these buttons so that the first makes the label display "Stop" in red, the second makes it display "Go" in green, and the third ends the program.
   [Hint: Colour values can be copied from the properties list using copy and paste and attached to your code]