Chapter 1
INTRODUCTION TO VISUAL BASIC 6, AND WRITING YOUR FIRST PROGRAM

Sohaib Khan, Tariq Jadoon

This chapter is based on the content of the lectures delivered by Tariq Jadoon on Thu 30 Aug 2007

It is assumed that you already have access to a PC which has Visual Basic 6.0 installed. VB 6 is installed on all computers in Labs 1 and 2. If you wish to install VB 6 on your personal machine, please ask ITSC for help.

PROGRAMMING USING VISUAL BASIC

Visual Basic is a programming language, which provides us with an easy way to write computer programs.

Key Concept: A Computer Program
A computer program is a set of instructions or code to be executed by the computer, and is designed to help users solve problems or perform tasks. Computer Programs are also called software. In simple words, a program tells a computer what to do, in precise and unambiguous terms. You are already familiar with a number of computer programs, such as the Calculator application in Windows, MS Word, and for that matter Windows itself. You also may be familiar with computer programs on the Internet, such as Gmail, Facebook and Google.

Note that when you run any computer program, such as the Windows Calculator, you do not see the set of instructions or code behind it. This is because the program is being executed when you run it, and the code is hidden from the user.

Key Concept: A Programming Language
A programming language provides us a means to write computer programs. A programming language consists of instructions (also termed as commands, statements, code) which specify to the processor what task it should do.

Thus, whoever wrote the Windows Calculator application for the first time must have used a programming language to provide the computer with the complete set of instructions which specify the behavior of the Calculator application, such as the numerical value and mathematical operations associated with each button. Remember that the computer is a totally dumb machine and hence, every minor detail has to be precisely communicated to it, for it to do anything useful. We use programming languages to communicate such details to the computer, much in the same way as we use English or Urdu to communicate with each other.

There are a number of programming languages, including Visual Basic, C, Pascal and Assembly Language, that you can use to write your computer programs. In a sense, these languages are like different spoken languages, each one with its own grammar, sets of words, strengths and weaknesses. Many professional programmers write their programs using the C language because of its power and flexibility. C, however, tends to be difficult to learn, and writing Windows
programs in C takes a lot of hard work. Of all the languages that we have worked with, Visual Basic is the easiest to learn and use.

**GETTING STARTED WITH VISUAL BASIC 6.0**

Visual Basic 6.0 (from here on referred to simply as VB) provides us with an easy way to write programs for Windows. We will now show you how to write your first Windows program.

From the Start Menu of your computer, find the Microsoft Visual Basic 6.0 shortcut. Most likely, this will be under the Microsoft Visual Studio 6.0 link. Visual Studio is a software development environment which contains other programming languages, in addition to VB. However, we will only be using VB in this course.

When you launch VB, you will see a screen similar to the one below.

What you are seeing is the *New Project* dialog. Select the *Standard EXE* project and click *Open*. You will now see a screen similar to the one below.
There are a number of windows on this screen. Each one of them is useful in developing programs. Each window can be closed and then opened again, by clicking on the appropriate toolbar button or selecting the menu bar and then clicking on appropriate sub-menu item. The following table shows the icons associated with each window.

<table>
<thead>
<tr>
<th>Window Name</th>
<th>Icon</th>
<th>Screen Shot</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form Window</td>
<td><img src="image" alt="Icon" /></td>
<td><img src="image" alt="Screen Shot" /></td>
<td>The Form Window displays the Form. A Form is the canvas through which users interact with Windows based applications. The Form window displays a Form hinged close to the top left corner of the Form Window. The Form can be resized by selecting and dragging one of the three guides at the right, bottom and bottom right of the Form, horizontally, vertically or diagonally.</td>
</tr>
<tr>
<td>Window</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Form Layout Window</strong></td>
<td>This window describes how the Form will appear relative to the entire screen once we run the program. If Form1 is moved outside the Screen, it will not appear on screen when the application is run. Resizing the Form1 in the Form Window will dynamically resize Form1 in the Form Layout Window.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Project Explorer</strong></td>
<td>This window shows the components of the current project. For example, we are currently seeing one object (Form1) under the project named Project 1. We will talk about Forms shortly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Toolbox</strong></td>
<td>This window contains number components such as buttons and scroll bars which you can place on your form. We will not discuss the functionality of this window currently, and will come back to its use after a few lectures.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Properties Window</strong></td>
<td>This window contains a listing of the properties (attributes or characteristics) of the current form. For example, the property which is currently highlighted in the figure on the left is named “DrawWidth” has a value of “1” written next to it. Also note that the small help tip at the bottom of the window contains a description of the currently highlighted property. We will talk about properties in the next sections.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Exiting Visual Basic 6.0

One can exit VB similar to most Windows based applications by pulling down the File Menu (Alt+F) and then selecting Exit (X).

More about Forms & Event-Driven Programming

Forms are the canvas through which users interact with Windows based applications. For example consider the Paint Application which is part of Accessories and can be accessed by pressing the Windows key ⊞ All Programs, Accessories and then selecting Paint.

The Paint application allows the user to select a particular drawing tool by clicking on a brush, pencil, rubber, etc. Clicking on the appropriate tool in the left panel triggers an event, i.e. the selection of the corresponding tool. In this sense the order in which instructions are executed is dictated by the events that take place such as clicking on a certain tool etc. Each event, such as clicking, double clicking (clicking twice in quick succession), pressing a key, moving the mouse, bringing the mouse over a certain object, etc. executes a block or set of instructions or code. Similarly once a certain tool such as a brush has been selected, further events dictate what color is chosen for the brush. Similarly the drawWidth of the brush can be changes by a single click etc. Event-Driven programming and the use of Forms containing appropriate Controls is at the heart of all windows based applications. This makes up the Graphical User Interface (GUI) through which the user interacts.

Designing, Testing and Executing Programs

An Integrated Development Environment (IDE) such as the VB 6.0 environment allows users to:

- Design
- Test
- Execute or Run

their programs through an integrated system. Thus, the development of VB programs goes through three phases.

Developers first design programs by designing the Form, selecting appropriate controls if necessary and specifying what code should execute when a particular event takes place. The code associated with an event is typed in by selecting the view code tab in the Project Explorer Window or by double clicking the Form in the Form Window. The view can be switched back to Form by clicking on the view object tab in the Project Explorer Window. While viewing code by clicking on the view code tab in the Project Explorer, one can choose between (General) and (Form) through a drop down menu. All events associated with a Form can be selected by clicking on the Form drop down menu on the top left of the Form in code view. VB Code is written in a block in between the Private Sub and End Sub keywords. The corresponding event that triggers this particular block of instructions is identified by the title of the event such as Form_Load, Form_Click, Form_Dbl_Click etc.

Once code has been written, the program can be executed or run by clicking on the Start button on the Toolbar. Notice that the menu title of the Form changes from:

Project1-Microsoft Visual Basic [design] to Project1-Microsoft Visual Basic [run]
As soon as the program begins execution. In the [run] mode, the application executes the application program and behaves as instructed during the [design] mode. If the program is not behaving as intended, one would stop the execution of the program and clicking on the End button on the toolbar. This would immediately return control to the design mode, and further modifications to the code can be tried out. Developing computer programs is usually an iterative process between design and run modes. However, in order to precisely figure out what is happening in the run mode it is extremely useful to debug (fix) a program by going into the break mode, more on this later.

UNITS

By default, the Form has dimensions of Twips. A Twip is one twentieth of a Postscript point, which in turn is one seventy second of an inch. This makes 1440 twips to an inch, and thus, a twip is a suitably small unit to work with.

A Form can be resized by selecting and dragging the guides in the Form Window or by directly entering in the Width or Height of the Form in Properties Window.

The properties of a Form may also be assigned by writing code in events. As an example consider the following lines of code, written in the Form_Load event.

```vbnet
Private Sub Form_Load()
    Form1.Width = 6000
    Form1.Height = 6000
End Sub
```

The Form_Load event is one of the first events that take place when a VB program having a Form executes. Thus, when this Form is loaded it is resized to the dimensions dictated by executing these two instructions in the Form_Load event. Notice, that the values assigned during the design mode are reassigned by executing these instructions during the run mode.

Similarly, if the same lines of code were associated with the Form_Click Event, instead:

```vbnet
Private Sub Form_Click()
    Form1.Width = 6000
    Form1.Height = 6000
End Sub
```

When the Form would load, it would have the same dimensions as specified during the design mode. However, clicking on the Form would result in the Form being resized to 6000x6000 twips as a consequence of the execution or firing of the Form_Click event.

Similarly, if the following lines of code were written in the Form_DblClick event:

```vbnet
Private Sub Form_DblClick()
```

Double clicking on the Form would result in the Form being resized to 3000x3000 twips. If the Form was clicked on, once again it would resize to 6000x6000 twips as a result of the corresponding code being executed in the Form_Click event.

**WORKING WITH GRAPHICS**

When working with graphics, always set the AutoRedraw property True in the design mode. Trust us on this, for the moment and we’ll explain this in greater detail at a later stage.

The simplest thing to draw with a pen or pencil is a dot. The equivalent in VB is drawing a dot on the Form using the PSET (pixel set) command. A **pixel** is a picture element and is a single point in a graphics image. The size of the dot is determined by the DrawWidth property of the Form, which can be assigned a value either through the Properties Window in the design mode or through code by specifying the drawWidth = *value* where *value* is a natural number in an Event.

The PSET command requires that the center of the dot be specified, i.e. PSET *(x, y)* where *(x, y)* is where the dot will be drawn in twips counted from the top left corner of the Form. The *x* co-ordinate increases from left to right while the *y* from top to bottom.

The Color of the **dot** can be specified by a third parameter. VB recognizes the following 8 default colors:
Thus, to draw a yellow dot at (3000, 3000) one would use the command:

PSET (3000, 3000), vbYellow

Finally, another important command when working with graphics is the CLS command, which clears the screen to erase all drawing. Thus if I have the following code in my program, what do you think the output will be on the screen (Hint: Commands are executed one after the other by a computer, and are executed very quickly)

Private Sub Form_Load()
    Form1.Width = 6000
    Form1.Height = 6000
    Form1.DrawWidth = 30
    Form1.PSet (2000, 2000), vbYellow
    Form1.Cls
End Sub

Now you can practice with some of the exercises given below. Make sure that you have the AutoRedraw property set to True, otherwise none of the drawings will appear on the screen.

EXERCISES
1. Write code to draw each of the following drawings.

2. Modify your code above so that the first drawing appears when the program runs, the second appears when the form is clicked, and the third appears when the form is double-clicked.