LECTURE NOTES

Prepared by: Mazhar Javed Awan (mazharjaved2001@yaho.com)

PROGRAMMING WITH VISUAL BASIC 6.0

CHAPTER 1

“WRITING YOUR FIRST PROGRAM”

INTRODUCTION

Visual Basic is an extremely flexible programming product designed for a variety of applications. Students, managers, and people in various technical fields use Visual Basic to learn how to write practical, Windows-based programs; business professionals use Visual Basic to write macros that leverage the documents and capabilities of their Microsoft Office applications; and experienced software developers use Visual Basic to build powerful commercial applications and corporate productivity tools.

Whether you fit into one of these categories or you are just curious about programming, Visual Basic 6.0 has features designed specifically for you. When you complete this programming course, you will have the skills necessary to write useful and interesting applications for the Windows and Windows NT operating systems.

CHAPTER 1: WRITING YOUR FIRST PROGRAM

The Visual_Basic_development_environment contains all the resources you need to build powerful Windows-based programs quickly and efficiently. This chapter introduces you to the features and capabilities of the Visual Basic 6.0 program development system, helps you get started with Visual Basic, and describes the various Visual Basic tools and windows that are available to you.

In this chapter, you'll learn how to:

- Explore and configure the Visual Basic development environment.
- Build your first program.
- Create an **executable (.EXE) file.**

1.1: Getting Started

This section helps you get Visual Basic loaded and running, and shows you how to control the development environment elements.

The following topics are included in this section:

1.1.1: Starting Visual Basic

The first step in using Visual Basic is launching it and opening existing files or creating new ones.

▶ **to start Visual Basic 6.0**

1. In Windows, click **Start**, point to Programs, and point to the Microsoft Visual Basic 6.0 folder.

   The icons in the folder appear in a list.

2. Click the **Microsoft Visual Basic 6.0** program icon.

   The **New Project** dialog box appears. This dialog box prompts you for the type of programming project you want to create.

   See the picture of **New Project** window below.

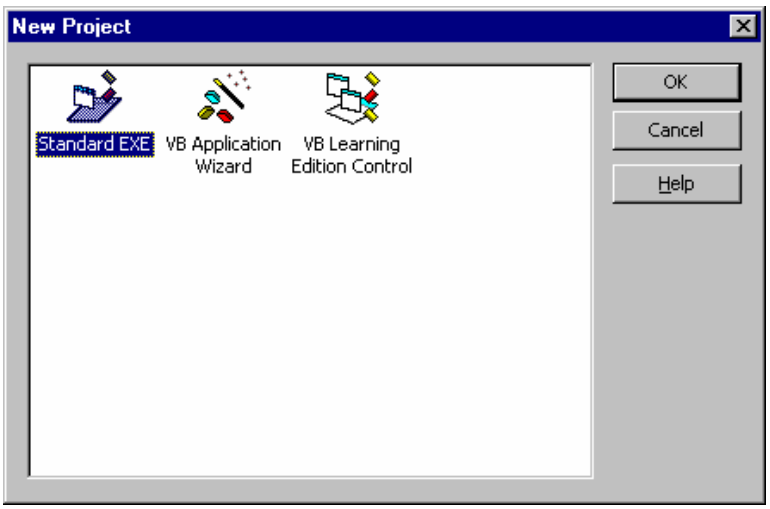


Figure 1: New Project Window

3. To accept the default new project, click **OK**.

In the Visual Basic development environment, a new project (a standard, 32-bit Visual Basic application) and the related windows and tools open.

See the picture below.

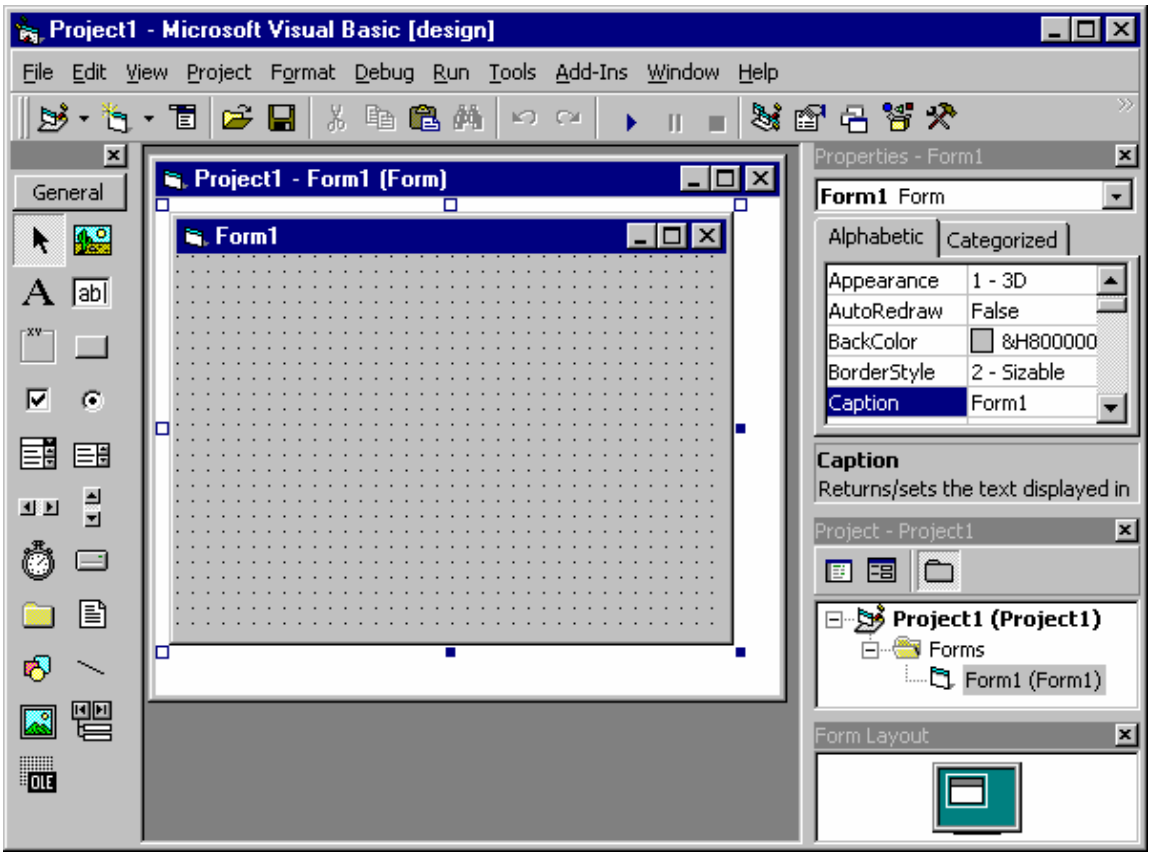


Figure 2: Visual Basic Environment

The Visual Basic development environment contains these programming tools and windows, with which you construct your Visual Basic programs:

- Menu bar
- Toolbars
- Visual Basic toolbox
- Form window
- Properties window
- Project Explorer
- Immediate window
- Form Layout window

The exact size and shape of the windows depends on how your system has been configured. In Visual Basic 6.0, you can align and attach (dock) the windows to make all the elements of the programming system visible and accessible. You'll learn how to customize your development environment in Moving, Docking, and Resizing Windows.

1.1.2: Loading and Running a Program

Before you can work with a Visual Basic program, you need to load the program into memory, just as you would load a word processing document in a word processor for editing.

▶ **To load a Visual Basic program into memory and run it**

1. On the **File** menu, click **Open Project**.

   The **Open Project** dialog box appears. With this dialog box, you can open any existing Visual Basic program on your hard disk, attached network drive, CD-ROM, or floppy disk.

2. If necessary, use the **Look In** drop-down list box and the **Up One Level** button to locate the program you want to load. Then, double-click the program name.

   The project file loads the Visual Basic user interface form, properties, and program code. (**Visual Basic project files are distinguished by the .VBP file name extension**.)

3. If the program user interface does not appear, open the Forms folder in the Project window, select the first form, and then click **View Object** in the Project window.

   This is an optional but useful step, especially if you want to look at the program user interface in the Form window before you run it.

4. On the Visual Basic Standard toolbar, click **Start** to run the program.

   The toolbox and several of the other windows disappear, and the Visual Basic program starts to run.

5. On the toolbar, click **End** when you want to exit the program.

1.2: Visual Basic Resources

1.2.1: Programming Tools

The location and purpose of the Visual Basic 6.0 programming tools are described in the following.

Menu bar Located at the top of the screen, the menu bar provides access to the commands that control the Visual Basic programming environment. Menus and commands work according to standard conventions used in all Windows-based programs. You can use these menus and commands by using keyboard commands or the mouse.

Toolbars Located below the menu bar, toolbars are collections of buttons that serve as shortcuts for executing commands and controlling the Visual Basic development environment. You can open special-purpose toolbars by using the **View** menu **Toolbars** command.

Windows taskbar This taskbar is located along the bottom of the screen. You can use the taskbar to switch between Visual Basic forms as your program runs and to activate other Windows-based programs.

1.2.2: Toolbox Controls

You use special tools, called controls, to add elements of a program user interface to a form. You can find these resources in the toolbox, which is typically located along the left side of the screen. (If the toolbox is not open, display it by using the **Toolbox** command on the **View** menu.) By using toolbox controls, you can add these elements to the user interface:

- Artwork
- Labels and text boxes
- Buttons
- List boxes
- Scroll bars
- File system controls
- Timers
- Geometric shapes
- Data and **OLE** controls.

See below to view an illustration of the standard contents of the toolbox.

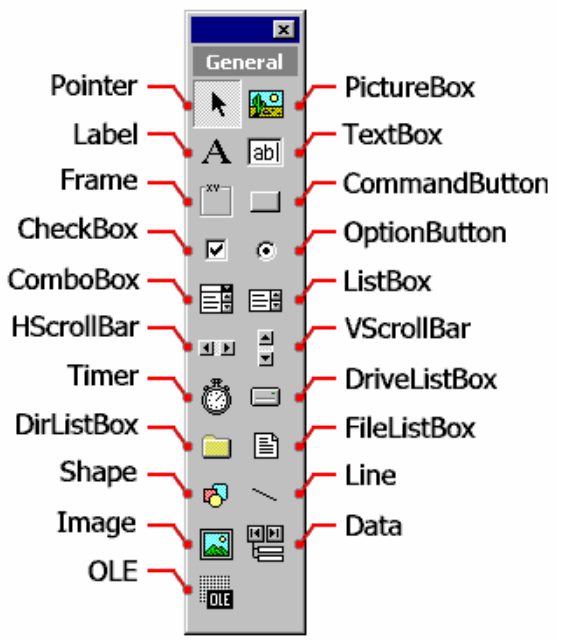


Figure 3: Toolbox Controls

Visible and Invisible Controls

When a Visual Basic program runs, most toolbox controls operate like the standard objects in any Windows-based application — and they will be visible to the user. However, the toolbox also contains controls that can be used to perform special, behind-the-scenes operations in a Visual Basic program. The powerful objects you create with these controls do useful work but can be made invisible to the user when the program runs. These objects can be used for:

- Manipulating database information.
- Working with Windows-based applications.
- tracking the passage of time in your programs.

1.2.3: Form Window

When you start Visual Basic, a default form (Form1) with a standard grid (a window consisting of regularly spaced dots) appears in a pane called the Form window. You can use the Form window grid to **create the user interface** and to line up interface elements.

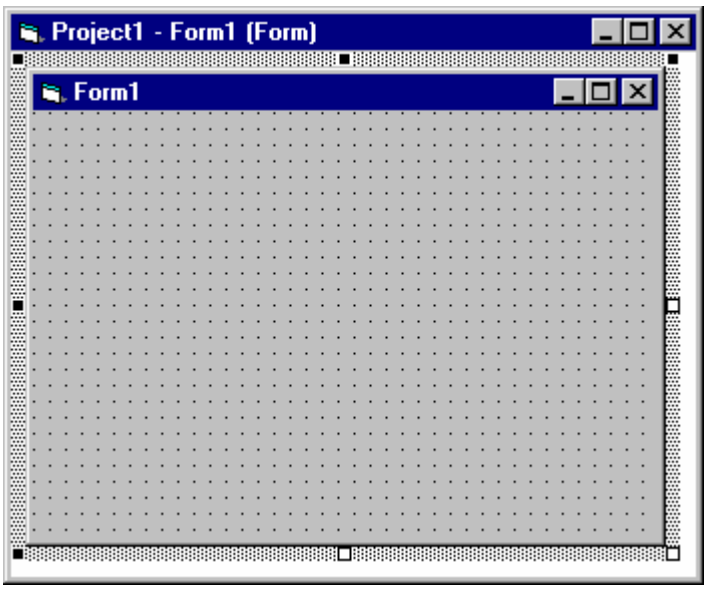


Figure 4: Form Window

Building Interface Elements

To build the interface elements, you click an interface control in the Visual Basic toolbox, and then you draw the user interface element on your form by using the mouse. This process is usually a simple matter of clicking to position one corner of the element and then dragging to create a rectangle the size you want. After you create the element — say, a text box — you can refine it by setting properties for the element. In a text box, for example, you can set properties to make the text boldface, italic, or underlined.

Adjusting Form Size

You can adjust the size of the form by using the mouse — the form can take up part or the entire screen.

Controlling Form Placement

To control the placement of the form when you run the program, adjust the placement of the form in the **Form Layout** window.

1.2.4: Properties Window

With the Properties window, you change the characteristics (property settings) of the user interface elements on a form. A property setting is a characteristic of a user interface object. For example, you can change the text displayed by a text box control to a different font, point size, or alignment. (With Visual Basic, you can display text in any font installed on your system, just as you can in Microsoft Excel or Microsoft Word.)

Displaying the Properties Window

To display the Properties window, click the **Properties Window** button on the toolbar. If the window is currently docked, you can enlarge it by double-clicking the title bar. To redock the Properties window, double-click its title bar again.

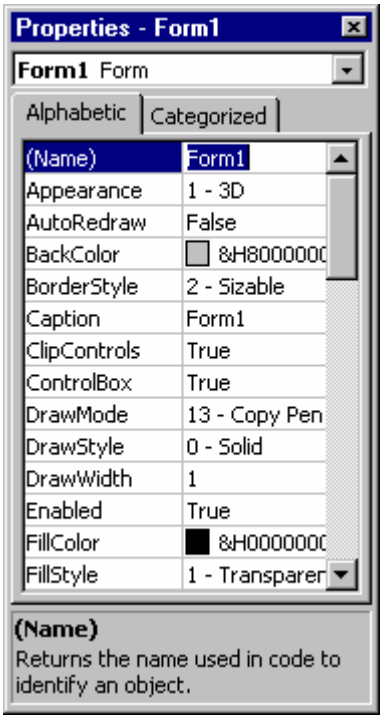


Figure 5: Properties Window

Properties Window Elements

The Properties window contains the following elements:

- A drop-down list box at the top of the window, from which you select the object whose properties you want to view or set.
- Two tabs, which list the properties either alphabetically or by category.
- A description pane that shows the name of the selected property and a short description of it.

Changing Property Settings

You can change property settings by using the Properties window while you design the user interface or by using program code to make changes while the program runs.

1.2.5: Project Window

A Visual Basic program consists of several files that are linked together to make the program run. The Visual Basic 6.0 development environment includes a Project window to help you switch back and forth between these components as you work on a project.

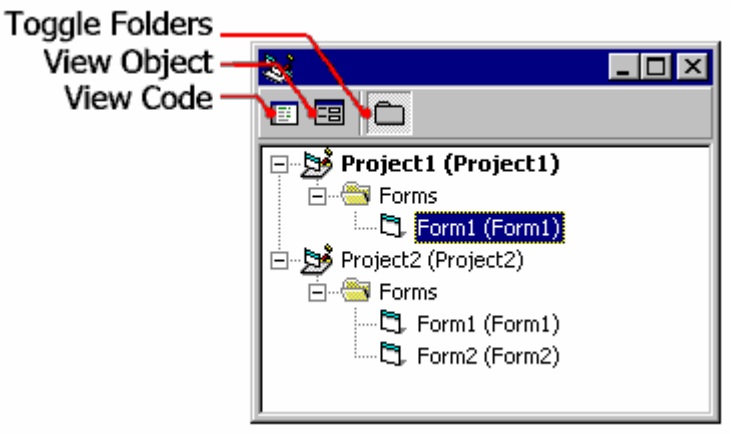


Figure 6: Project Explorer Window

Project Window Components

The Project window lists all the files used in the programming process and provides access to them with two special buttons: **View Code** and **View Object**.

Displaying the Project Window

To display the Project window, click the **Project Explorer** button on the Visual Basic toolbar. If the window is currently docked, you can enlarge it by double-clicking the title bar. To re-dock the Project window, double-click its title bar again.

Adding and Removing Files

The project file maintains a list of all the supporting files in a Visual Basic programming project. You can recognize project files by their .vbp file name extension.

You can add individual files to and remove them from a project by using commands on the Project menu. The changes that you make will be reflected in the Project window.

Note In Visual Basic versions 1 through 3, project files had the .mak file name extension. In Visual Basic versions 4, 5, and 6.0, project files have the .vbp file name extension.

Adding Projects

If you load additional projects into Visual Basic with the **File** menu **Add Project** command, outlining symbols appear in the Project window to help you organize and switch between projects.

1.2.6: Code Window

You can create much of your program by using controls and setting properties. However, most Visual Basic programs require additional program code that acts as the brains behind the user interface that you create. This computing logic is created using program statements — keywords, identifiers, and arguments — that clearly spell out what the program should do each step of the way.

You enter program statements in the Code window, a special text editing window designed specifically for Visual Basic program code. You can display the Code window in either of two ways:

- By clicking **View Code** in the Project window.
- By clicking the **View** menu **Code** command.

1.2.7: Form Layout Window

The Form Layout window is a visual design tool. With it, you can control the placement of the forms in the Windows environment when they are executed. When you have more than one form in your program, the Form Layout window is especially useful — you can arrange the forms onscreen exactly the way you want.

To position a form in the Form Layout window, simply drag the miniature form to the desired location in the window.

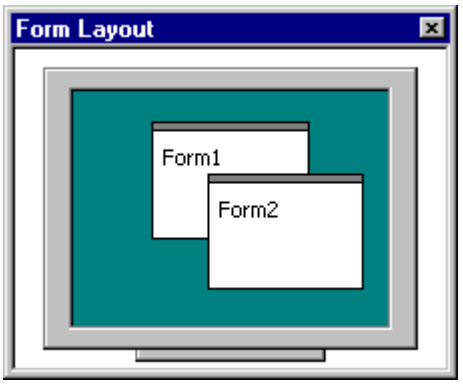


Figure 7: Form Layout Window

1.3: Developing Visual Basic Programs

1.3.1: Developing Visual Basic Programs

If you haven't written a program before, you might wonder just what a program is and how to create one in Visual Basic. This section provides an overview of the Visual Basic programming process.

A program is a set of instructions that collectively cause a computer to perform a useful task, such as processing electronic artwork or managing files on a network. A program can be quite small — something designed to calculate a home mortgage — or it can be a large application, such as Microsoft Excel.

A Visual Basic program is a Windows-based application that you create in the Visual Basic development environment. This section includes the following topics:

- Planning the Program
- Building the Program
- Testing, Compiling, and Distributing the Program

1.3.1.1: Planning the Program

The first step in programming is determining exactly what you want your program to accomplish. This sounds simple (isn't it obvious?), but without a mission statement, even the best programmer can have trouble building an application he or she is happy with.

Planning a program is a little like planning a barbecue. For a barbecue to go off smoothly, you need to prepare for it ahead of time. You need to organize the menu, invite your friends, buy the food, and (most likely) clean your house. But a barbecue can be entertaining if friends just happen to drop by and bring stuff. Programs, though, usually don't turn out the best if they're built with the stone-soup approach.

Identify your Objectives

Long before you sit down in front of your computer, you should spend some time thinking about the programming problem you are trying to solve. Up-front planning will save you development time down the road, and you'll probably be much happier with the result. One part of the planning process might be creating an ordered list of programming steps, called an **algorithm.**

Ask Yourself Questions

When you plan Visual Basic programming projects, you might find it useful to ask yourself the following questions about your program:

- **What** is the **goal (mission)** of the program I am writing?
- **Who** will use the program?
- **What** will the program **look like** when it starts?
- **What information** will the user enter in the program?
- **How** will the program process the input?
- **What** information **(output)** will the program produce?

When you finish this preliminary work, you'll be ready to start building the program with Visual Basic.

1.3.1.2: Building the Program

Building a Windows-based application with Visual Basic involves three programming steps: creating the user interface, setting the properties, and writing the code. And, of course, your project must be saved.

These steps are described in the following topics:

- Creating the User Interface
- Setting the Properties
- Writing Program Code
- Saving a Project

Creating the User Interface

After you have established a clear goal for your program, it's important to think about how it will look and how it will process information. The complete set of forms and controls used in a program is called the program user interface. The user interface includes all the menus, dialog boxes, buttons, objects, and pictures that users see when they operate the program. In the Visual Basic development environment, you can create all the components of a Windows-based application quickly and efficiently.

Setting the Properties

Properties are programmable characteristics associated with forms and their controls. You can set these properties either as you design your program (at design time) or while you run it (at run time). You change properties at design time by selecting an object, clicking the Properties window, and changing one or more of the property settings. To set properties at run time, you use Visual Basic program code.

Writing the Program Code

You finish building your program by typing program code for one or more user interface elements. Writing program code gives you more control over how your program works than you can get by just setting properties of user interface elements at design time. By using program code, you completely express your thoughts about how your application:

- Processes data
- Tests for conditions
- Changes the order in which Visual Basic carries out instructions.

The Visual Basic Programming Language

The Visual Basic programming language contains several hundred statements, functions, and special characters. However, most of your programming tasks will be handled by a few dozen, easy-to-remember keywords.

In this course, you'll spend a lot of time exploring the subtleties of writing useful program code that you can adapt to a variety of situations. For now, though, just keep these points in mind:

- Program code follows a particular form (syntax) required by the Visual Basic compiler.
- You enter and edit program code in the Code window, a special text editor designed to track and correct (debug) program statement errors.

Saving a Project

After you complete a program or find a good stopping point, you should save the project to disk with the **Save Project As** command on the **File** menu.

Saving to Disk

Saving a Visual Basic project to disk is a little more complicated than saving a Word or Excel document. In addition to the project (.vbp) file, Visual Basic also creates a separate file for each form (.frm file) and (if necessary) for each standard code module (.bas file) in your project. You must save each of these files, one by one.

When you choose the **Save Project As** command, Visual Basic prompts you for each file name, one at a time. The process concludes when the project assembly instructions (which serve as the program's packing list) are saved in a project file. The file names included in this packing list are the components listed in the Project window.

Note: **You can recognize PROJECT files by their .vbp file name extension, and FORM files as .frm file name extension.**

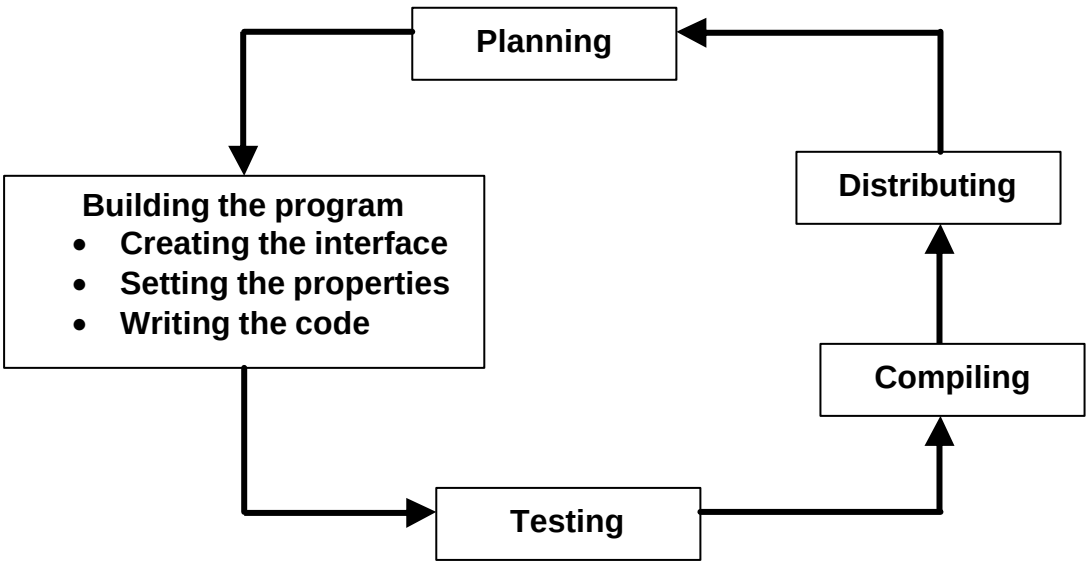
Reusing Component Files

You can use component files associated with a project in future programming projects by using the **Project** menu **Add File** command. This command merges forms and standard code modules into the current programming project and makes them appear in the current program Project window.

1.3.1.3: Testing, Compiling and Distributing the Program

After you create a working version of your program, you need to test it carefully to verify that it works correctly. If you wish to distribute your program, you also need to compile it into an executable program (a stand-alone Windows-based program) and give it to your users. If you decide to revise the program later, you repeat the process, beginning with planning and an analysis of any new goals. These steps complete the software development life cycle.

See the illustration below for a summary of the software development life cycle



Testing and Debugging

Testing a program involves checking it against a variety of real-life operating conditions to determine whether it works correctly. A problem that stops the program from running or from producing the expected results is called a software defect (bug).

If you've used software for any length of time, you've probably run into your share of computer glitches caused by faulty software. Now that *you* are the programmer, it's your job to stamp out these problems before they reach the end user. Fortunately for all of us, Visual Basic provides some excellent debugging tools to catch bugs. You'll learn how to find and correct bugs in Chapter 5: Controlling Flow and Debugging.

Compiling

When you've finished creating and testing your program, you can compile it into an executable (.exe) file that will run under Windows or Windows NT. Creating an executable file with Visual Basic is as simple as clicking the **Make Project1.exe** command on the **File** menu.

If you plan to run your new Visual Basic application only on your own system, creating the executable file will be your final step. When you want to run your new program, simply double-click the program's .exe file in Windows Explorer. Or, use the file to create a shortcut icon that you can place on your Windows desktop.

Distributing

You might want to share a Visual Basic executable file with friends or colleagues or sell your program. If so, you need to put the program and a few necessary support files on a disk (or the network), where your users or customers can get at them. All Visual Basic programs require one or more **dynamic-link library (DLL)** files to run. To get your users up and running, you need to copy the necessary files from your hard disk to a distribution disk. The exact DLL files you need depends on these factors:

- The operating system that your users run.
- The Visual Basic features that your program uses.