BONUS CHAPTER 2

Using Visual Basic with Visual SourceSafe

Understanding source code control

Understanding source code control is required to understand why Visual SourceSafe is needed.

Introducing Visual SourceSafe

Take a look at the Visual SourceSafe functions that are used by most developers.

Adding Visual Basic Project to the SourceSafe Environment

See how to use Visual SourceSafe to control and track changes in your application code.

Controlling changes made to an application over a period of time by one or more programmers is one of the harder things to do. This type of control has changed over the years. First it was referred to as change control, meaning all that was controlled was who was using the program source code and what changes were being made. Most of the systems that did change control were stand-alone applications. Now this process has changed and it is just called *source code control*. Simply put, source code control is the capability to control the total access of any number of program files for a complex application. Several products are available that perform this function; however, if you are a Visual Basic programmer, the one you want to use is Visual SourceSafe.

Microsoft Visual SourceSafe is an easy-to-use tool that is completely integrated into the Visual Basic 5 Integrated Development Environment. It can be used by individuals on a single PC or by large development teams across a network to track who is working on the programs of an application and what changes are being made to those programs. Version 5 of Visual SourceSafe is a 32-bit-only application that comes with Visual Basic 5 (Enterprise Edition only), or can be purchased separately.

This chapter introduces the concepts of source code control, what is Visual SourceSafe, and how it is integrated into the Visual Basic environment. Finally, from within Visual Basic 5, you will see how to use the features of Visual SourceSafe to aid in maintaining your applications. ■

Understanding Source Code Control

Understanding the concept of source control is the first step to knowing why you need to practice it if you are, or plan to be, a serious developer of Windows applications. If you have ever done any programming, you know the problems involved in trying to protect your source code. What would you do if the changes you just saved to the project files are wrong and you have to go back to an earlier version of the code? Without source code control, you would need to save or back up the files to several different locations so that you could always go back to a previous copy of a file.

By using a source code control tool, such as Visual SourceSafe, you don't have to worry about which copy of the source is the working copy and which is the starting set. This type of tool also keeps track of your program changes. Source control enables you to create components that can be compiled into many different applications. This *sharing* handles the question, "Which version of the component am I currently using?" If you don't know which version is which, it is possible that you will compile an application that doesn't work correctly.

You may think that change control is exactly the same as source code control. In fact, they are very different—source control tracks applications, change control does not. They do both track and control changes made to source code and who is using the program files. However, change control is old technology. The concept of change control came about in the days of mainframe assembler programming. The best change control programs in those days were products such as Librarian. These programs were stand-alone products that programmers were required to use if they needed to control changes made to the applications on which they were working. The problem with these change control programs was that they did not track applications, just individual program files. In addition, programmers had to perform the following routine when using change control:

- Leave the editing environment
- Run the commands to check out a file
- Return to the editor
- Make the changes needed
- Save the file
- Exit the editor
- Run the commands to check the file back into the change control system

As you can see, this is a very tedious process if you are working on many different program files at one time. The switch from change control over to source code control happened at approximately the time that Windows programming was introduced. Initially, the process was about the same as with change control, but as Windows software became more complex and PCs became larger and faster, source code control started to take off.

Source code control became an integrated part of the Windows Development Toolbox that programmers can use to enhance their working environment. Source code control now means complete control over the application and any related files, text or binary, that may be included with the application.

Introducing Visual SourceSafe

Visual SourceSafe is a project-oriented source code control system for individual or team development of software applications, or any other work that benefits from using source code control. It stores and tracks changes to files of any type, text or binary. Visual SourceSafe supports team application development by enabling team members to share files, modify them independently, and later merge the changes back into a single copy of the file. Team members can review a file's history and, if necessary, recover to an earlier version of a file. Visual SourceSafe is flexible enough to support any size project and any number of users.

Unlike other version control programs, Visual SourceSafe provides these features in a project-oriented system that keeps track of relationships among files, so the developers don't have to. You can use Visual SourceSafe directly with any of the Microsoft Visual development tools, including Microsoft Visual Basic 5. The most commonly used Visual SourceSafe commands are available from within the Visual Basic 5 environment, enabling the programmer to remain within the development environment while accessing the Visual SourceSafe functions. The rest of the Visual SourceSafe command set is always available from the Visual SourceSafe Explorer interface.

What Visual SourceSafe Can Do for You

A typical software development team can include many programmers, writers, testers, and designers, or it can be a single programmer working in the comfort of his or her own home. Any development effort has many different changes occurring at once, so you would need to have the following functions performed to protect the application:

- Check files in and out, keep them where they belong, and store important documents
- Record all file activity
- Make sure files go only to those with the proper security
- Keep track of utilities and keep things running smoothly
- Merge changes back into the master copy
- Version control

Rather than rely on the programmers to keep track of the changes they make or worry about changes made by others, Visual SourceSafe can take over and perform these functions. This helps prevent one programmer from changing a section of code that another programmer is already modifying.

As its name suggests, the most important feature of Visual SourceSafe is that it keeps your source files safe. Using SourceSafe with your applications is like checking books in and out

of a library; the only difference is that these "books" can't get lost. The advantages of using Visual SourceSafe include the capability to do the following:

- Check out files to two or more programmers at the same time.
- Share common code files between projects without needing separate copies.
- Check files in and out and add comments to describe the changes made.
- Prevent program files from accidental deletion.

Visual SourceSafe has a history function that keeps the "books" for your application project. By knowing the history of any given program file, you can determine what changes were made, who made them, and when they were made. This enables you to choose the correct version of the program file to use with the version you are working on. In addition, by using passwords on the SourceSafe user definitions, you can prevent unwanted usage of any program file that is protected by Visual SourceSafe.

Installing Visual SourceSafe

Installing Visual SourceSafe really includes installing two separate products. The first product to install is the Visual SourceSafe Administrator. This application creates the SourceSafe database to which all users will have access. The second installation is the Visual SourceSafe client program called the Explorer. Installing the client program not only gives the programmer access to the Visual SourceSafe functions, but it also registers Visual SourceSafe to enable direct integration into the Visual Basic environment (Enterprise and Professional Editions, versions 5.0). For single-computer environments, both the administrator and the client are installed on the same PC.

NOTE The Visual SourceSafe integration into Visual Basic will not work until the user has done a network or client setup.

Visual SourceSafe creates an empty database during the installation process. The Visual SourceSafe Administrator's program needs to run in order to add users to the Visual SourceSafe environment.

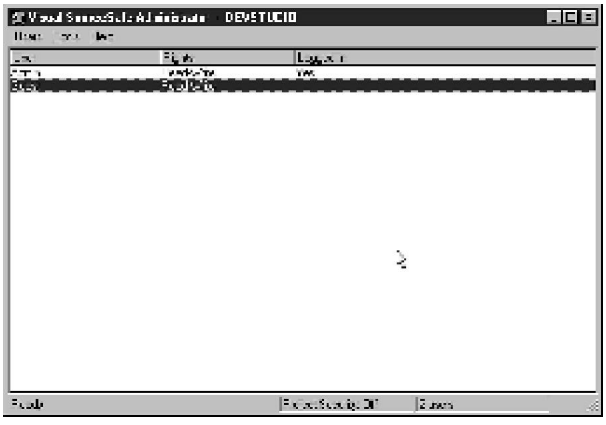
Administrating the Visual SourceSafe Environment

The administrator of the SourceSafe environment controls the list of users who have access to the Visual SourceSafe database. If you are the administrator or the only user, you are responsible for maintaining the user list and the associated working parameters for each user. In addition, you are responsible for keeping the database working smoothly, backing up the database periodically, and fixing any problems that may occur.

The key to Visual SourceSafe is the database. This stores all the master copies of the program file, the history records, and the project structures. A project is always contained within one database, but multiple projects can be stored in one database, or multiple databases can exist to store multiple projects.

Before using Visual SourceSafe, you must set up the environment for your particular installation by starting the Visual SourceSafe Administrator. Both the Administrator and the Client program icons were added to the Programs list of the Start menu during the installation process. When starting Visual SourceSafe Administrator, you will see the Administrator's visual interface (see Figure B2.1), which lists the users defined in the SourceSafe database.

FIG. B2.1
Maintaining the Visual SourceSafe's User list to access any project files.



The user list shows every user who has access rights to the Visual SourceSafe environment. Any user that doesn't appear in this list cannot access the database. By default, when you install Visual SourceSafe, this list has only two entries: Admin and Guest, which are defined as follows:

- *Admin User* There can be only one Admin user. This user cannot be deleted or have its user name changed. The Admin user has full access rights and the right to undo the checkout of a file that another user has checked out—neither function can be changed. Finally, Admin is the only user who can run the Visual SourceSafe Administrator and modify the user list
- *Guest user* This user is used as a default template to create other users. Also, it is used to provide access to the Visual SourceSafe database for occasional or first-time users. You can delete the Guest, as well as change its access rights.

By selecting a user from the list and then choosing Tools, Options from the menu, the Options dialog box displays (see Figure B2.2).

The tabs on the Options dialog box cover several different areas of the Visual SourceSafe environment that can be modified to uniquely define each user and their capabilities when making use of Visual SourceSafe. These are the following:

Tab	Description
General Options	Settings that affect all the users defined in the user list.
Project Security	Controls whether security is on or off for a project. If security is on, you can then set default access rights for the users.

continues

Tab	Description
Shadow Folders	Used to set the shadow folder for a particular project. A shadow folder is a central folder that contains current versions of all the files in a project.
Web Projects	Sets the information that is applicable to a single Web project.
Web	Sets options for all Web projects at once rather than for an individual Web project.
File Types	Sets the types of files that the user can store in Visual SourceSafe.

FIG. B2.2
Project and security options for a user are maintained on the Options dialog box.

After you install Visual SourceSafe and set up the user list, it is advisable to back up your database. Thereafter, you should continue to back up the database on a very regular basis. Make sure that these backups are *full backups*, not incremental or differential. In addition to regular backups, it is recommended that you run the ANALYZE program to check your Visual SourceSafe database for corruption. If any corruption is found, the ANALYZE program often can repair the problem.

ON THE WEB

www.microsoft.com/ssafe Updates to the Analyze program are periodically posted to this Visual SourceSafe Web site. You should check this site periodically to keep this utility up-to-date.

As a single user, you do not need to assign passwords to the user IDs that you create. You might be wondering why, as a single user, you would even need more than one user ID. If you work on more than one project or application, you might want different user properties set for each application—for example, a different Shadow Folder and Working Folder for each application.

Project-Oriented Programming

Project-oriented programming is not as mysterious as it sounds. If you're using a development tool such as Visual Basic 5, you are already familiar with the concept of project-oriented programming, even though you may not realize it. When you do any type of program development, you make changes to only a few of the modules in the application at a time. When the application contains many different files that are then compiled together, it is easier to keep all these files together in a project. Visual Basic does this by creating a single project file (.vbp) that keeps track of all other files included in that project. Visual SourceSafe provides the additional tools necessary to get the files you need from the project, make the changes you want to make, and then return the files to Visual SourceSafe for safe keeping.

Planning a Visual SourceSafe Project

Because Visual SourceSafe is a project-oriented tool, you must have a project in which to place the files before you can do anything with them. When you begin working with Visual SourceSafe, you first must create a project for your application. The organization of projects is similar to the organization of folders on your PC. Projects contain subprojects in the same way that folders contain subfolders.

When designing Visual SourceSafe projects, it is a good idea to have the project folders match the directory structure that you created for the application.

When you begin a development project, you often start with a design to determine what and how you will build it. In the same way, when you start working with a Visual SourceSafe project, you can apply the same design considerations. Organizing your files in a Visual SourceSafe project is similar to organizing files in folders on your hard drive. The files included in the project can be source code, bitmaps, word documents, ASCII text files, or any other file type you want to use in the project. The big difference in Visual SourceSafe is that a file can be shared by many projects at the same time. When you modify a file in one project, the change is applied automatically to all the projects sharing the file. A list can be produced to show the projects that will be affected by a change.

NOTE Visual SourceSafe does not impose any rules of its own on project design. It is up to you to set these rules.

When you create projects, you should consider the following design guidelines:

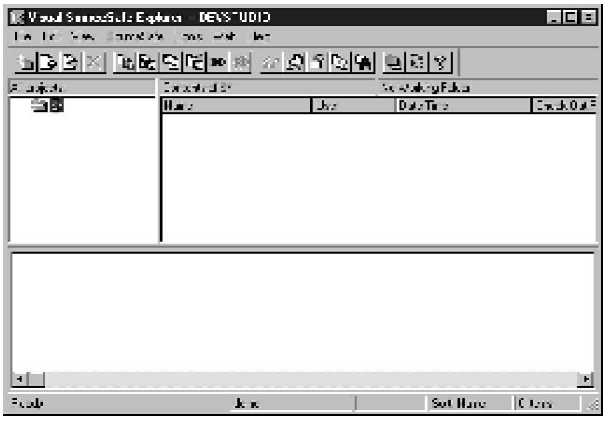
- If you have files in existing folders, make each folder a project in Visual SourceSafe. Each subfolder should be a subproject.
- Divide your files among projects and subprojects. Visual SourceSafe can handle more than 8,000 files in a project, but breaking large projects into subprojects tends to make them easier to manage.

- Don't add the same file separately to multiple projects. Instead, share the file between projects.
- Don't make a new project when moving to a new version of the project. You should label the project to note the version.
- Place all the files needed to create one application in one project. Such files are generally program code but can also include libraries, bitmaps, and dynamic-link libraries.
- Don't keep executable files in Visual SourceSafe projects unless absolutely necessary.
- The project should contain only tested code that compiles and runs. This sometimes is not possible, especially at the start of a project. A good rule, however, is that a program should not be checked in unless it compiles and runs.
- In any multiuser project, a file should not be checked out for longer than it takes to make and test the changes. If a file is checked out for several days, other programmers can't receive the benefit of changes that have been made to it.
- In addition to keeping your source code in Visual SourceSafe, you also can store your documentation files, icons, graphics files, and files of any other type in Visual SourceSafe projects.

Implementing a Visual SourceSafe Project with the Explorer

To set up the project structures, you must start the Visual SourceSafe client application to use the Explorer. The Visual SourceSafe Explorer provides the graphical representation of the relationships of projects and subprojects (see Figure B2.3).

FIG. B2.3
The Visual SourceSafe Explorer interface allows users to design the control project structure to match the application's directory structure on the hard drive.



The Visual SourceSafe Explorer is the main user interface that you use to navigate your project tree, select files, and execute the commands that act on those projects and files.

Managing Files and Folders in Your Project

All files in your project are stored in the Visual SourceSafe database. You should never work with the master copy of any file that is stored in Visual SourceSafe, except to compare another copy to it. Visual SourceSafe provides each user with a copy of the file to read or change. Each time someone checks in a file, Visual SourceSafe stores not only the changed file, but a history of the changes as well.

When using Visual SourceSafe, you cannot actually work with the files from within Visual SourceSafe. If you want to work on a file, you must get a copy of it from Visual SourceSafe and must have a working folder in which to put it. Every project is associated with a working folder on the PC. This working folder is a directory on your local hard disk, or on a network drive, if one exists. It is used to work on the files you have obtained from Visual SourceSafe.

NOTE You must specify a working folder before you start work on any files that are stored in a Visual SourceSafe project. ■

Another folder you can create is a *shadow folder*, which is used by SourceSafe to hold the current versions of all the files in a single project. This folder does not contain the master copy or the local copy of a file. It does, however, provide a central location from which to look at the overall structure of the project with which you are working. Additionally, it serves as a good place from which to build or compile the project.

NOTE It is not necessary to use shadow folders with Visual SourceSafe. Using a shadow folder at least doubles the amount of disk space that a project needs when checked into SourceSafe. ■

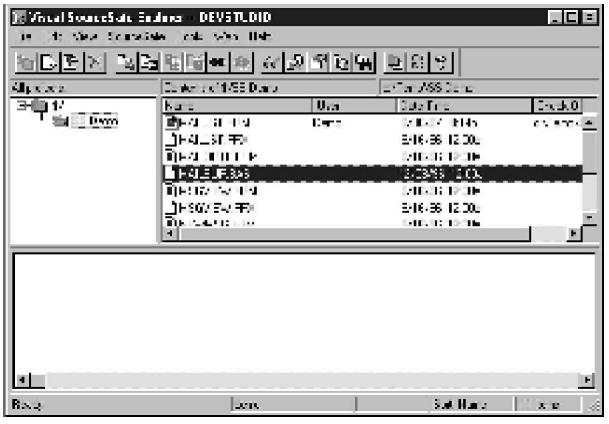
To start working with a file, you first must retrieve the file from the Visual SourceSafe project. To retrieve a file from Visual SourceSafe, you check the file out of the SourceSafe project. It is then copied into your working folder for that project. This provides a local copy for you to work on. If you have not yet specified a working folder for the project, Visual SourceSafe prompts you to do so. After you have made changes to the file and tested those changes, you can check in the file to Visual SourceSafe. This copies the file from your working folder back into your current project.

CAUTION

Working folders are associated with projects, not with files. You cannot choose a different working folder for each file in a project. However, a separate working folder should be used for each project.

Visual SourceSafe has made it very easy to check out a file from its associated project. Start Visual SourceSafe and log in with the user name assigned to you for using Visual SourceSafe (see Figure B2.4).

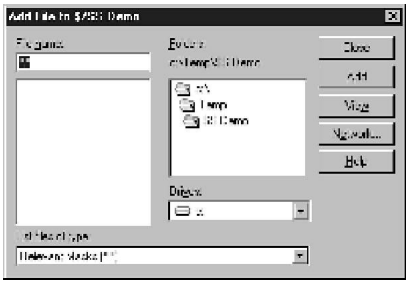
FIG. B2.4
Opening Visual SourceSafe for a specific project lists all the available files in the project and also shows if any files are already checked out by the user.



Using Visual SourceSafe

When you create a project in Visual SourceSafe, you need to add the files you want to track to the SourceSafe project. You can add these files to the project either by using the Add File dialog box (see Figure B2.5) or by using Windows drag-and-drop from the Windows Explorer to the SourceSafe Explorer.

FIG. B2.5
Adding files or folders to the SourceSafe project is as easy as saving a file to your hard disk.



If you have an entire folder that you want to put in the project, you can add it the same way you would add a single file to the project. Using Visual SourceSafe gives you the capability to support projects of different sizes and degrees of complexity. The best way to see how this is accomplished is to take a look at two examples that show different uses of the product.

Example 1 A programmer is working on a one-person project that consists of a set of Visual Basic files. Visual SourceSafe is used to maintain separate versions of each file and the complete set of files. As each set of files reaches a milestone, the programmer labels that version with a unique name so that any version of the program can be easily built by returning to a specific named version.

Example 2 Two teams of developers are working in parallel on different versions of the same application. One version is a "bug fix" release for the current product; the other version will become the next major release of the application. Both teams start by sharing the same code

base but branch off onto different development paths. The bug fix team completes its project, checks the code back into Visual SourceSafe, and delivers its version. The major release team then incorporates all the relevant bug fixes into its version of the application by merging them into the major release.

Checking Out Files To request files from Visual SourceSafe, start the Visual SourceSafe Explorer, select the files that you want to use, and then choose SourceSafe, Check Out to have the files copied to your working folder. When you check out a file from Visual SourceSafe, you have several options from which to choose. First, you need to indicate whether you are checking out the file to modify it or just to look at or use it. If you only want to see what's in the file, you can use the Get Latest Version option to retrieve the file. This option retrieves the most recent version of a file from Visual SourceSafe and creates read-only copies for viewing or compiling in your working folder.

NOTE Getting a file does not replace the file that you currently have in your working folder, unless you set the Replace option. ■

When you use the Get Latest Version command, the following four possibilities can occur:

- When the file you are retrieving is not in your working folder, it is then simply copied to your folder.
- When the file in your working folder is identical to the file you are retrieving, no action is performed.
- When the file in your working folder is different from the master copy, and your local copy is read-only, your local copy is replaced.
- When the file in your working folder is not read-only, Visual SourceSafe assumes that you have the file checked out and doesn't replace the file.

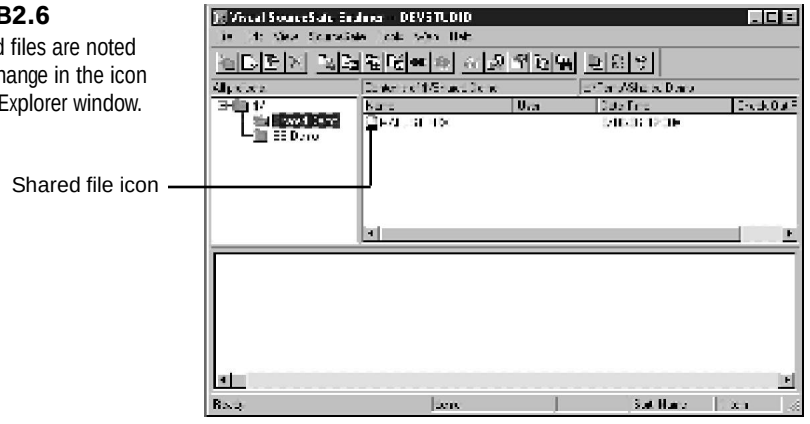
If you are retrieving the file to modify it, you then use the Check Out command. The Check Out command retrieves updatable files from Visual SourceSafe and places them in your working folder so that you can modify them. There is an important difference between the Check Out and Get Latest Version commands. The Get Latest Version command places a read-only local copy of a file in your working folder, whereas the Check Out command places an updatable copy of the file in your working folder.

CAUTION

A file checkout is generally exclusive; that is, no one else can check out a file that you currently have checked out. Your installation may be set up to allow multiple checkouts.

Sharing Files Across Projects Another feature of Visual SourceSafe is the capability to mark a file or files as Shared. This allows one file to be part of two or more projects at the same time. You can share any file from any project with any other project to which you have access. When a file is shared, the file's icon, which is displayed in the Explorer file pane, changes from a single file to overlapping files, as shown in Figure B2.6.

FIG. B2.6
Shared files are noted by a change in the icon in the Explorer window.

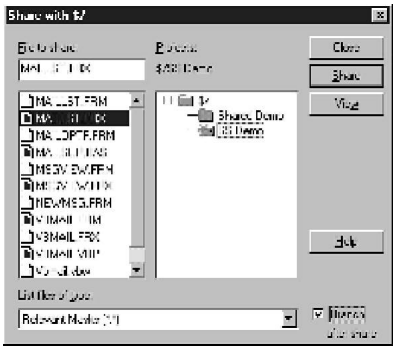


There is only one master copy of a shared file; therefore, any changes that are checked in trickle down to all the projects that share the file. The Links command can be used to list all projects that are sharing a particular file. An offshoot of sharing files is a function called *branching*.

Branching is the process of taking a file or project in two separate directions at once. To do this, they must be separated into two copies that are not linked. Up to the point of branching, the files share a common history because they are the same copy. After that point, they each have a separate development path. Visual SourceSafe tracks branches by making each development path a different project. When you use this technique, each branch has its own project, which keeps it unique. Using the Paths command shows the history of a file in the form of a tree diagram, which shows both the current and former links that have been separated by the branching process.

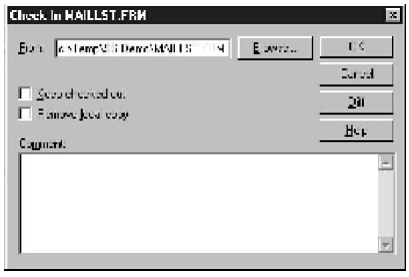
You can instruct Visual SourceSafe that you want to create a branch at the same time that you share a file or project. When you use the Share With dialog box (see Figure B2.7), you can click the Branch After Share check box to create a new branch of the selected file(s) or project.

FIG. B2.7
Branches are created when a file or project is shared by two or more programmers and they make unique changes to the code.



Checking In Files After you finish making necessary changes to a file in your working folder, you should return the file to the Visual SourceSafe project from which it came. You use the Check In command on the SourceSafe menu to replace the file into the project. The Check In dialog box gives you several options when you return the file to the project (see Figure B2.8).

FIG. B2.8
You can choose from several actions when you are checking a modified file into the project depending on what you want the final outcome to be.



The options in the Check In dialog box enable you to make the following choices:

- Keep the file checked out to continue working on it after you check in your changes.
- Remove the copy of the file from your working folder after the check in is complete. By default, a read-only copy of the file is left in your working folder.
- Display the differences between the version of the file you are checking in and the version you checked out.
- Browse folders in search of other files that you have checked out and would like to check in.

In addition to these settings, you can also enter a comment that describes the changes you made to the file. When you are working in a team environment, with shared files, you should make it a practice to list the differences and history of the file before checking it back in to the SourceSafe project. This way, you can see if anyone else was working with the file while you were and, if so, you can check for differences. This list lets you see if your changes will be affected by changes made by the other programmer.

Other Features of Visual SourceSafe

Several other features of Visual SourceSafe merit mentioning in this chapter. These features are as follows:

- Project history reporting
- Comparison processing
- Project text search
- Version tracking of your project

Each feature helps you control your application development project.

Project History Reporting During the lifetime of a Visual SourceSafe project, all changes to a file or project are tracked. You can view the history of changes to any file, or the entire project, by choosing Tools, Show History. From the History dialog box, you can:

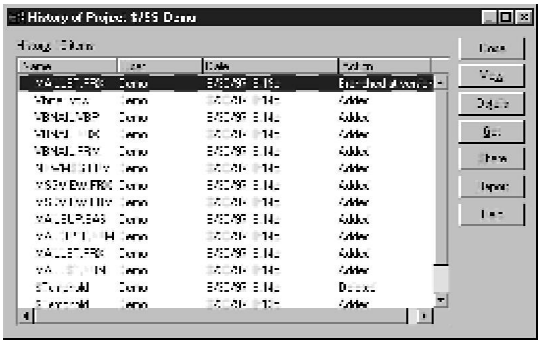
- View
- Get
- Share
- See Details
- Print Reports

All these functions can be performed against any file(s) or project(s). In addition, comments that have been added when an action was performed against a file (for example, check in) are included in the history records for a file or project. Visual SourceSafe provides space to enter as much information as an action requires, up to 32K characters. The History dialog boxes for files and for projects are very similar—the same procedures can be followed for both files and projects. To see the history of a project or file, start Visual SourceSafe, log in, and then select the project or file whose history you want to view. Then choose Tools, Show History from the menu. If you select a project, you will see the Project History Options dialog box (see Figure B2.9).

FIG. B2.9
Choose the options to display the History Information.

Notice on this form that you can choose to skip it in the future. Now, click the OK button to see the History display for the selected project or file (see Figure B2.10).

FIG. B2.10
Display the history of a file in Visual SourceSafe.



The History of Project dialog box provides several command buttons from which to choose that affect the file you are looking at. Table B2.1 lists and explains these commands.

Table B2.1 Project History Command Buttons

Button	Function
Close	Closes the dialog box.
View	For projects—displays an additional dialog box that lists the versions of files in the selected version of the project. For files—displays a read-only version of the selected file in an editor.
Details	Displays the History Details dialog box. Allows you to edit comments and labels for the selected project or file.
Get	Places a read-only copy of the selected file, or all the files in a project when you select a project, in your working folder.
Share	Shares a selected file in the current project with another project.
Diff	Displays differences between two selected versions of a single file (available for files only).
Pin	Freezes the selected file at a specific selected version. Becomes Unpin after a Pin action (available for files only).
Rollback	Returns a file to the previous version, discarding all versions of the file subsequent to the rolled back version (available for files only).

Select an item from the list and click the Details button. This will display the History details, as shown in Figure B2.11, for the selected entry. It is always a good idea to check the history of a file before making any major changes to that file.

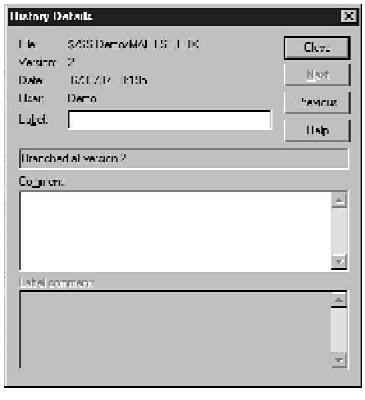
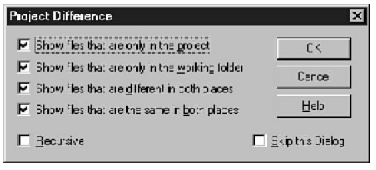


FIG. B2.11
History Details help determine where the file has been.

Comparison Processing Visual SourceSafe can display differences between text files, but not between binary files. You can use the Show Differences command to display differences between two files, and even to show differences between the contents of a selected project and the contents of your working folder for that project. You can select the options that Visual SourceSafe uses to display the differences. When you choose the Show Differences option from the Tools menu, the Project Difference dialog box is displayed for you to select the working options (see Figure B2.12).

FIG. B2.12
You can choose to display the differences only in the files you are using, or the differences in both your files and the master copies.



It is important to note that Visual SourceSafe stores the current version of a file and the previous changes that were made. This allows Visual SourceSafe to reverse any change to display a previous version of a file. This method uses much less disk space than it would take to store an entire new file with every check in. The similarity between the treatment of text and binary files ends with the Show Differences command, which displays file changes to the user. For a binary file, each change is stored as a small record of what the changes were. This record is excellent for reconstructing older versions, but it cannot be displayed to a user. When the differences are displayed, as shown in Figure B2.13, any lines that differ are displayed in contrasting colors, which can be set in the Difference tab of the Options dialog box (choose Tools, Options).

Project Text Search Visual SourceSafe lets you scan through text files for a specific character string by using the Find In Files command. All occurrences of a character string in the specified files are then displayed. This command makes no changes to the files or in the projects that contain them. You can scan through a file or through a project, in which case all the files in that project are scanned. The Find in Files dialog box, shown in Figure B2.14, works the same way as most Find functions in other Windows applications.

This command comes in very handy when you need to find text in labels, captions, or other string variables that are in the file or project.

Version Tracking of Your Project If you are working with an application that will grow over time, similar to the way in which Microsoft Word has grown and changed from version to version, you need to use version tracking to keep the changes straight. You see version tracking in almost every application that runs on a PC. When an application progresses from version 1 to version 2, someone must keep track of what changes were made for the new version. In addition, the older version must still be kept around for users who do not upgrade to the newer

version. Visual SourceSafe has the following three methods of tracking the different versions of files and projects with which you are working:

- *Version Numbers* These are internal numbers maintained by Visual SourceSafe. You, the user, have no control over these numbers. Every version of every file and project in Visual SourceSafe has a version number. The version number is always a whole number and always increases.
- *Labels* You can apply labels to any version of a project or file. A label is a free-form string of up to 31 characters. The following examples are all valid labels: 1.0, 2.01b, or Field Test 3.
- *Date/Time* These strings tell when a file was last modified, or when a file was checked in. Visual SourceSafe supports both 12-hour format (with *a* or *p* suffix) and 24-hour format.

FIG. B2.13
Differences are shown in a split screen, with contrasting colors and an arrow to highlight the lines that are different.

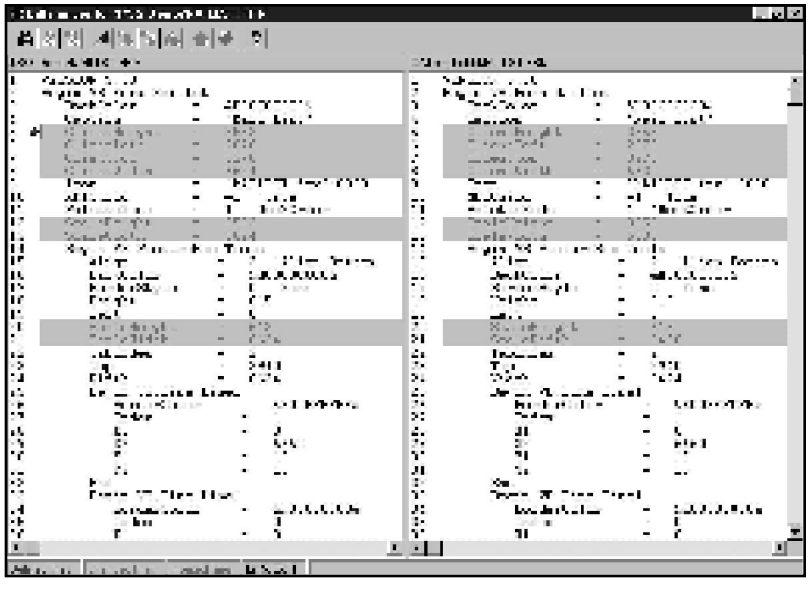
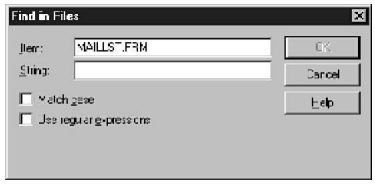


FIG. B2.14
Find text in any file within a Project by using the Find In Files function.



Version numbering and Labels both have good and bad points when you use them. Table B2.2 lists the ways these methods perform the same types of functions.

Table B2.2 Comparing the Way Version Numbering Works Against Version Labeling

Version Numbering	Version Labeling
Assigned automatically	Assigned by user, using the Label command.
Always a numeric value	Any combination of letters, numbers, symbols, and spaces, up to 31 characters.
Always increases to next whole	Anything user assigns number.
Increases each time an action that affects storage is taken on a file or project	Assigned when user feels that a significant milestone has been reached.
Displayed in history, paths, links, share, and file properties dialog boxes and in file pane of Visual SourceSafe Explorer	Displayed in history dialog boxes as a user-supplied string. Indicated by a label icon next to the project name in place of a version number. The user-supplied label string is displayed in the Action column of the History dialog box.
Does not create a new version, simply identifies a new version	Creates a new version of the file or project and the label is associated with the new version.
Cannot be edited or changed by user	Can be edited in the History Details dialog box.

As you can see, each has its own strong points. The one to use is the one that will do what you need for a given project.

Using Visual SourceSafe with Visual Basic

Now that you have seen what Visual SourceSafe is, how it works, and what it can do, look at how it's used with Visual Basic. When you use Visual Basic 5 with Visual SourceSafe, you receive all the features of SourceSafe, many of which are fully integrated into the Visual Basic development environment. Whether you are working with an existing Visual Basic project or starting a new project, Visual SourceSafe is present to make sure that the project will be controlled, if you want it to be.

In this section, you will see how to use the Visual SourceSafe functions when you are doing the following:

- Creating a new Visual Basic application
- Working with an application that was not already controlled by Visual SourceSafe
- Creating a local copy of a Visual Basic project from a Visual SourceSafe-controlled project
- Working with a project to which you already have access from Visual SourceSafe

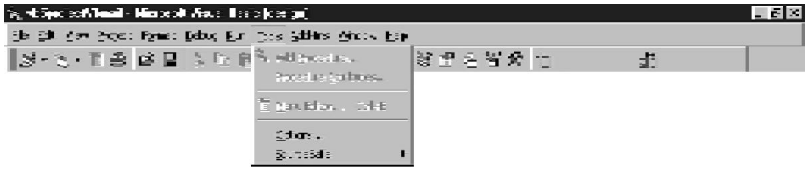
Except for a few minor differences, each of the uses just listed require the same basic process, which is as follows:

1. Log in to SourceSafe.
2. Open a project.
3. Check out the required files.
4. Make any needed changes.
5. Test the changes.
6. Check the files back into SourceSafe.

What's Included in Visual Basic

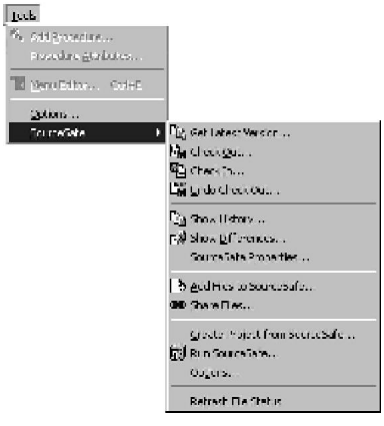
When Visual SourceSafe is installed and registered properly, a new selection appears on the Tools menu in Visual Basic (see Figure B2.15).

FIG. B2.15
A new SourceSafe menu option appears in Visual Basic.



Then, when you select SourceSafe, a secondary menu will appear (see Figure B2.16).

FIG. B2.16
Visual SourceSafe uses a submenu in Visual Basic.



The most important of these new menu choices is the Options command. When you select the Options item, the Source Code Control Options dialog box displays, as shown in Figure B2.17.

This dialog box enables you to customize how Visual Basic interacts with Visual SourceSafe automatically. For each question listed on the form, you can choose Yes, No, or Ask to have the program ask you whether to perform the action. Each question is fairly straightforward;

however, it is very important that you pick the correct settings. For most programmers, the default settings will work just fine. For more advanced interface options, click the Advanced button, which displays the SourceSafe Options dialog box shown in Figure B2.18.

FIG. B2.17
Set the options to control how SourceSafe performs.

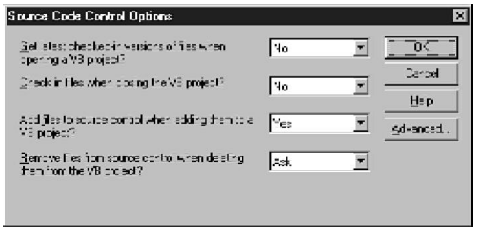
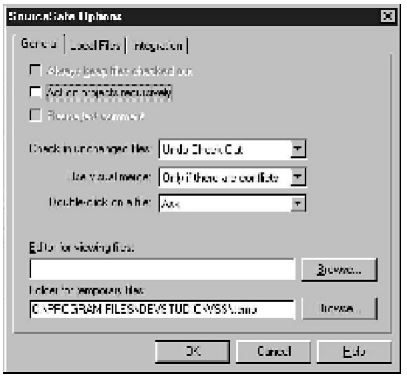


FIG. B2.18
Advanced SourceSafe options can be set on the tabbed Options dialog box.



This dialog box has three tabs that enable you to modify many setting that SourceSafe uses to interact with Visual Basic. The following list explains each tab:

Tab	Description
General	Enables the user to modify general configuration settings
Local	Enables the user to change how the files in the working folder are handled
Integration	Displays integration information about your project and the related Visual SourceSafe project

The remaining choices on the SourceSafe submenu are your gateway into SourceSafe (refer to Figure B2.16). You can check files in and out, display a form's history, even add new files to the project. In fact, if you right-click a form or module in the Project Explorer, the pop-up that displays includes the four main SourceSafe options at the bottom of the menu list (see Figure B2.19).

Creating a New Visual Basic Project

When you create a new Visual Basic application, you will not interact with Visual SourceSafe until you save your work for the first time. When you save your new project the first time, you

will be prompted by SourceSafe to add the project to the SourceSafe database (see Figure B2.20). If you click the Yes button, Visual SourceSafe prompts you for all the required information to create the SourceSafe project from your new Visual Basic project.

FIG. B2.19
SourceSafe options are accessible from the Visual Basic pop-up menu.

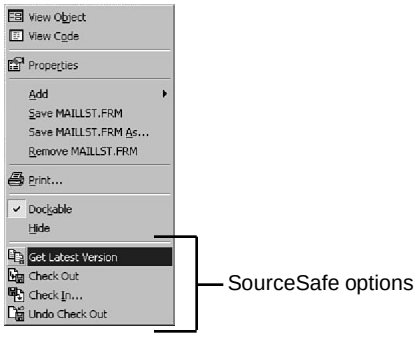
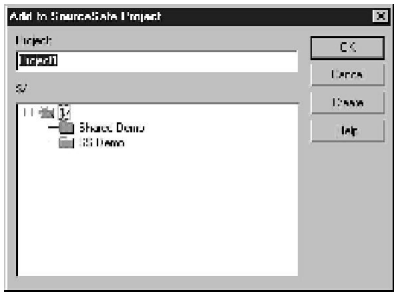
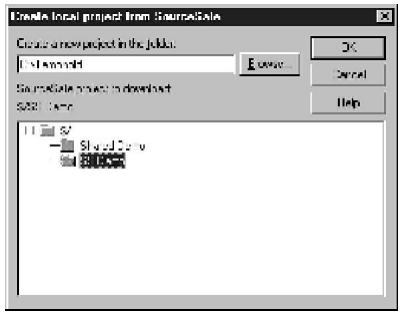


FIG. B2.20
Add a new project to SourceSafe when saving it the first time.



If you are joining a project team that is already using SourceSafe, you will have to create the working project on your PC. To do this, choose Create Project from SourceSafe on the Tools, SourceSafe menu. This will display the project folders from the SourceSafe database for you to select from (see Figure B2.21).

FIG. B2.21
Creating a new Visual Basic project from an existing SourceSafe project.



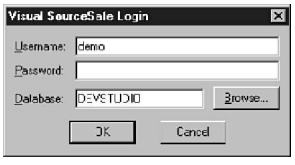
After you select a project folder, you will be prompted for the local directory in which to create the project. When you click OK, this directory will be created and will become the working

folder for the Visual Basic project on your PC. You now will be able to check files in and out of SourceSafe for this project.

Using an Existing Visual Basic Project

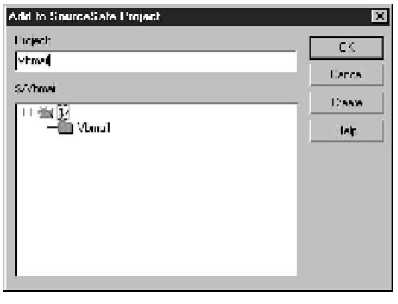
When opening a current project that is on your PC but not under SourceSafe's control, you will be asked if you want to add the project to SourceSafe. If you respond yes, you then are prompted to log into SourceSafe (see Figure B2.22).

FIG. B2.22
Logging into SourceSafe is required whenever you use a project that is protected.



Enter your SourceSafe user name and password, if any, and then click OK. Next, you are prompted to select the SourceSafe folder to add this project to (see Figure B2.23). The new project folder can be added either to the root level of the database or as a subfolder of an existing project folder.

FIG. B2.23
Adding the project folder is the first step in protecting your work.



If the folder you want to use doesn't exist, click the Create button to add the new folder to the database. After you have the folder you need, click OK to continue. Now that the project folder exists, you are shown a list of files that are in the Visual Basic project (see Figure B2.24). Select the files that you want to add to the SourceSafe project and click OK.

After you select the files, Visual SourceSafe finishes the process by adding these files to the project and copying them to the shadow folder that was created for you by the administrator of SourceSafe. While this is happening, you will see a status box showing SourceSafe at work. Don't expect to be able to start programming yet—display a form in Visual Basic, and you will see something new in the title bar (see Figure B2.25).

The label `(Read Only)` means just that. You must check out a form or module file before you can make any changes to it. To work with Visual SourceSafe, choose Tools, SourceSafe, and

then the appropriate submenu option. If you want to check out files, choose Check Out from the menu to get the Check Out dialog box (see Figure B2.26). Select the files you need and click OK.

FIG. B2.24
Choosing the files to include in the SourceSafe process.

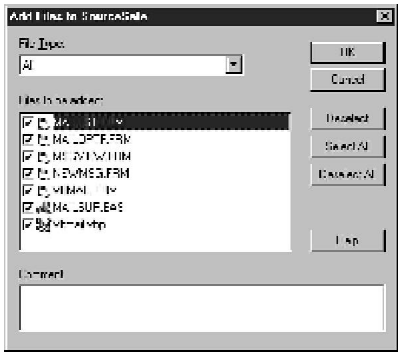


FIG. B2.25
The newly protected files in a project are READ-ONLY, preventing any changes from being made.

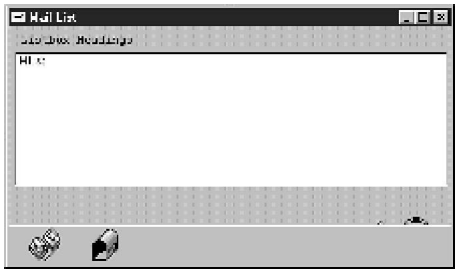
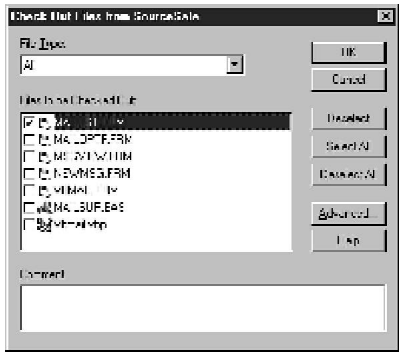
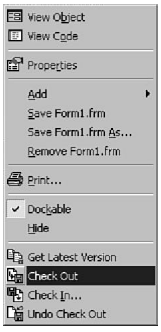


FIG. B2.26
Use the SourceSafe Check Out dialog box to select several files at the same time to work with.



However, if you only need to check out a single file, right-click the file in the Project Explorer to display a pop-up menu (see Figure B2.27).

FIG. B2.27
Use the SourceSafe options from the Visual Basic pop-up menu.

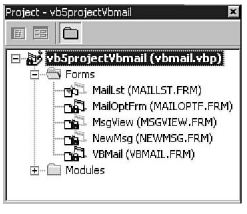


After you click the Check Out option, the file is checked out and ready to use. All the other Visual SourceSafe functions are available from these menus.

Opening a Protected Visual Basic Project

If you open a project that is protected by Visual SourceSafe, you will be prompted to log in to Visual SourceSafe before you can continue. After the project is open, you need to check out the files to work with. The easiest way to do this is to right-click the form or module that you need to modify and then select Check Out from the menu. After you do this, you will notice that the form icon has changed, as shown in Figure B2.28.

FIG. B2.28
Visual Basic file icons in the Project Explorer reflect the status of the file within SourceSafe.



From Here...

This chapter has taken a close look at source code control, and what is included in Visual SourceSafe to make it easier to control a Visual Basic project. Finally, you have seen that the integration of Visual SourceSafe into Visual Basic is not as complicated as you might have thought. The Visual Basic development process does not really change when using Visual SourceSafe to control the changes made to a Visual Basic application.

- Chapter 17, "Managing Your Projects," discusses how to manage all the pieces in a Visual Basic project.
- Chapter 41, "Using Visual Basic in a Client/Server Environment," looks at how the development process changes when working in a client/server environment.