USING

Visual SourceSafe

by Kevin Gao
CEO, Architect of Dynamsoft

Dynamsoft™

Introduction

SourceSafe / VSS is a version control tool used by many Windows developers. I started to use SourceSafe in 1995. Before I started to work on SourceAnywhere for VSS (a VSS remote access tool) in 2003, I only used very basic features of SourceSafe and did not even use integration in Visual Studio. After successfully launching SourceAnywhere for VSS, our team started another product, SourceAnywhere, the SQL-based VSS replacement. During the development process of these products I gained a deeper and deeper understanding of SourceSafe.
In this SourceSafe How To series / tutorial, I will write some guides, How To and FAQs about SourceSafe and will also post resources I found on the internet. This series will cover some basic and intermediate topics of SourceSafe as well as some advanced topics.

Although this SourceSafe tutorial can be used for anyone who is interested in SourceSafe, it is mainly for software developers. As you may have been using SourceSafe for a long time and know many features of VSS, I will try to make the table of the content easy to navigate so hopefully you can easily find the information that you are interested in.

Should you have questions or comments about SourceSafe / VSS or this series, you can contact me at kgao@dynamsoft.com. But due to the limit of my time, there is no guarantee that you can get an answer. Nevertheless, I may publish new articles covering your feedback.
Thanks.

What is VSS?

Microsoft Visual SourceSafe, also called 'VSS', is a file-system-based source control tool from Microsoft. This tool was originally developed by One Tree Software Company and later taken over by Microsoft.
The following content is copied from Microsoft web site:

Microsoft Visual SourceSafe is a file-level version control system that permits many types of organizations to work on several project versions at the same time. This capability is particularly beneficial in a software development environment, where it is used in maintaining parallel code versions. However, the product can also be used to maintain files for any other type of team.
At a minimum, Visual SourceSafe does the following:

- Helps protect your team from accidental file loss.
- Allows back-tracking to earlier versions of a file.
- Supports branching, sharing, merging, and management of file releases.
- Tracks versions of entire projects.
- Tracks modular code (one file that is reused, or shared, by multiple projects).

Market position

As a member of Visual Studio product family, VSS is a very popular version control solution among Windows developers. An IDC August 2004 ALM market report indicates the revenue of Microsoft in the ALM market in 2003 is US$38.1M. Since Microsoft did not have any other ALM product at that time, we can assume all the Microsoft's ALM revenue is from SourceSafe.
SourceSafe comes with some editions of Visual Studio and MSDN subscription, which makes many developers think that SourceSafe is FREE. Actually, it is not. You paid for it. SourceSafe can also be purchased independently through a retail channel.

The list price of SourceSafe 2005 is US $549.

What can VSS do?

In a nutshell, you can use VSS to keep your files, including the previous versions, in a central repository/database. Although VSS is mainly used by software developers, it can be used by anyone working with computers. VSS can store any types of files, such as source code, project plans, specification documents, database objects, and your kitchen design blue print.

History preserving

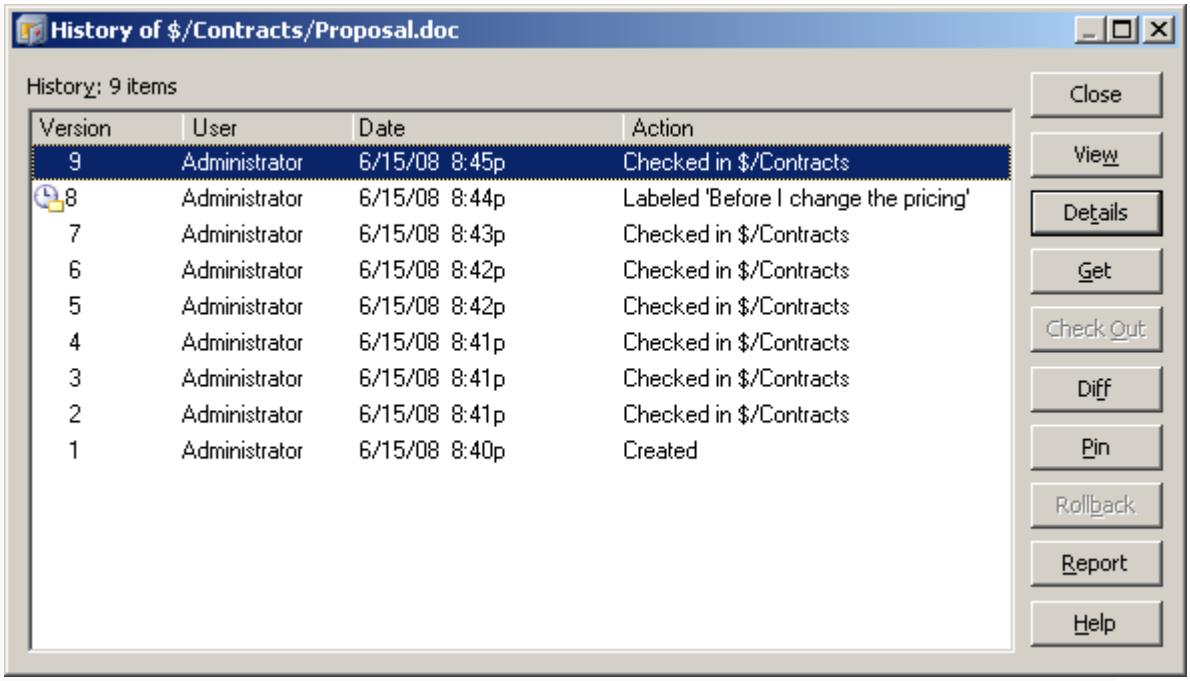
Let's say you have a file, c:\work\proposal.doc. Do you have a bunch of related files that look like:

Proposal_2008.1.1.9PM.doc
Proposal_backup_before_changing_pricing11.doc
Proposal.doc.back2

Or several related folders that look like:

C:\Work_Back1
C:\work_back2
C:\work_back_2008.1.1

A version control system, such as VSS, can completely eliminate the need of backing up your files in the above mentioned way.
SourceSafe keeps all your versions you checked in (if "Store only latest version" option is not selected). And best of all, under normal circumstances, you see only the latest version, which make your file management much easier. Whenever you need previous versions, you can use the "Show History..." feature to access all the previous versions as shown in the following figure:



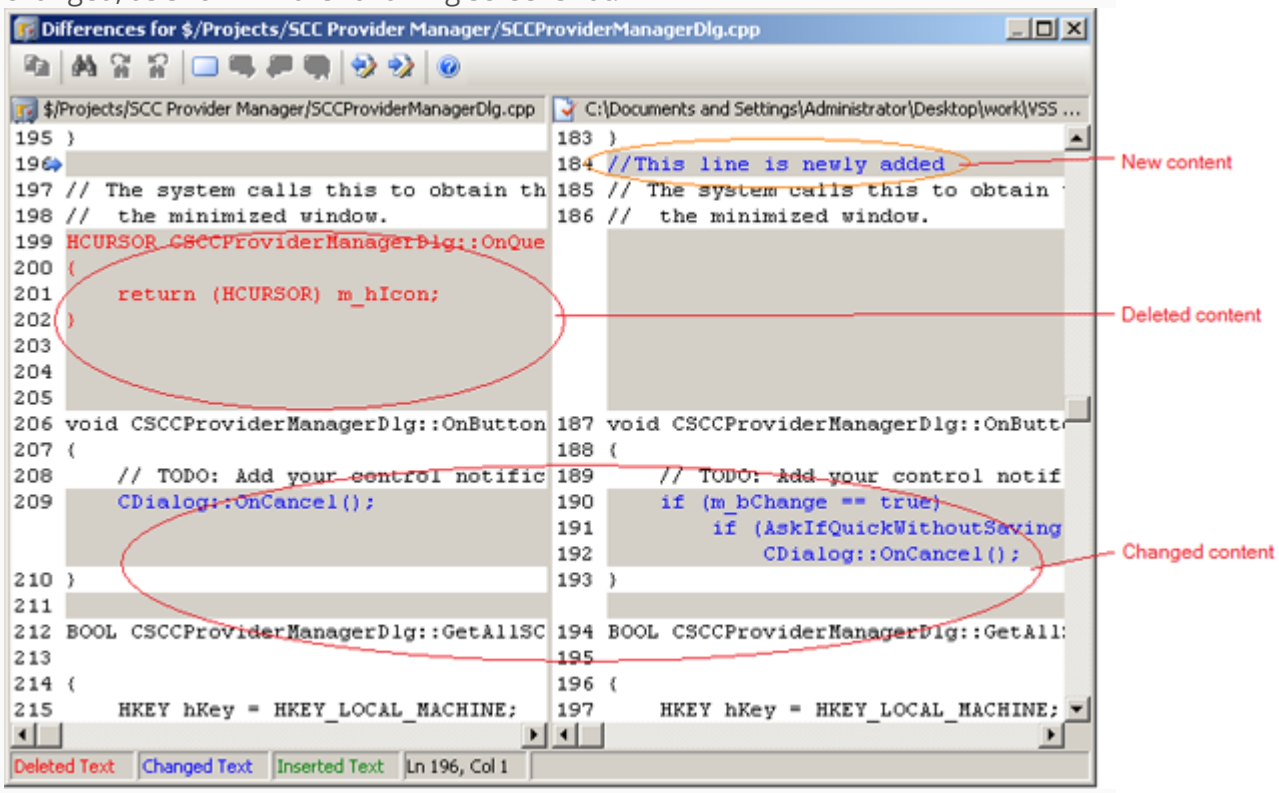
(History of Visual SourceSafe / VSS)

You can see who and when did what change and do Get and View operations on the versions.

File Difference

VSS can visually diff non-binary files, such as your C#, ASP.NET or Java source code. Word and Excel files are binary files and cannot be diffed in SourceSafe. You can view the differences between local file and any version in the VSS database, two previous versions in VSS, any two local files or any two files in the VSS database.

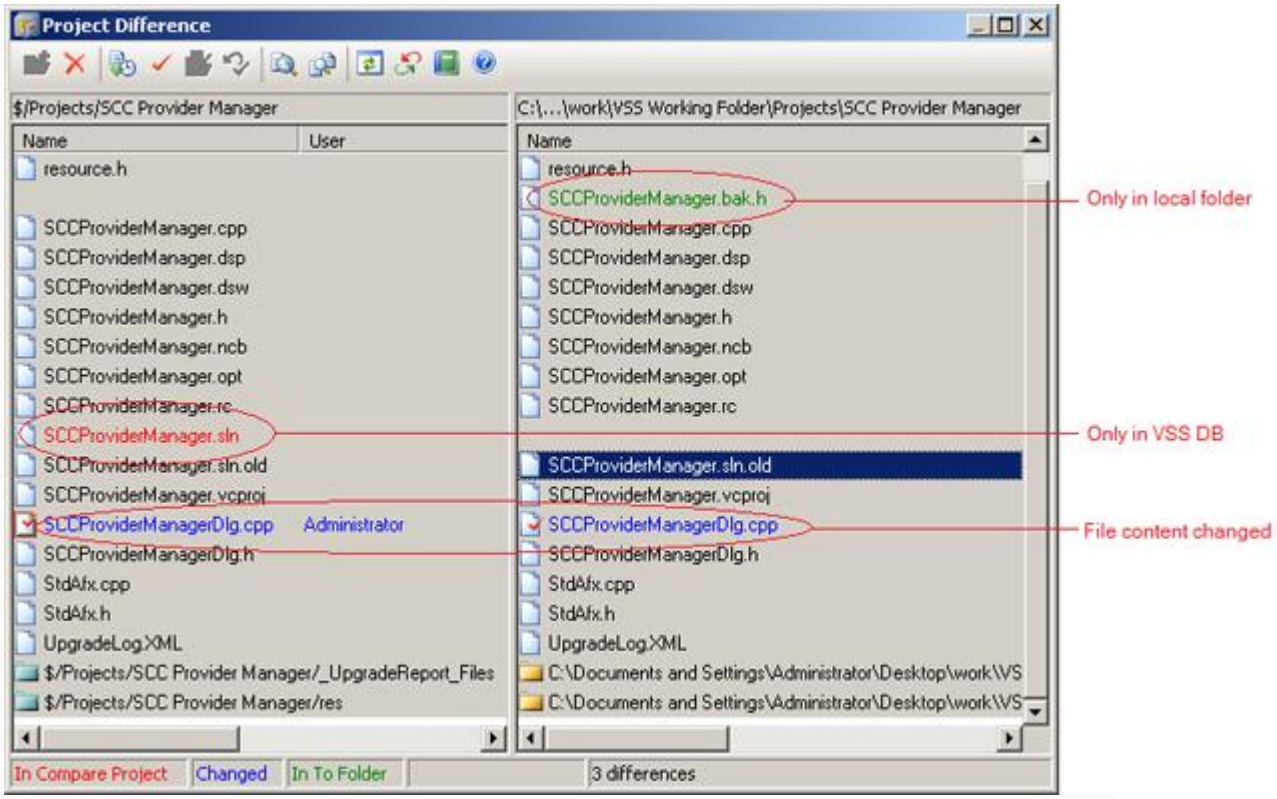
You can see which lines are newly added, which lines are deleted and which lines are changed, as shown in the following screenshot:



(File Difference in Visual SourceSafe / VSS)

Project/Folder Difference

Project difference feature allows you to see the differences between your local folder and VSS project, two local folders or two VSS projects:



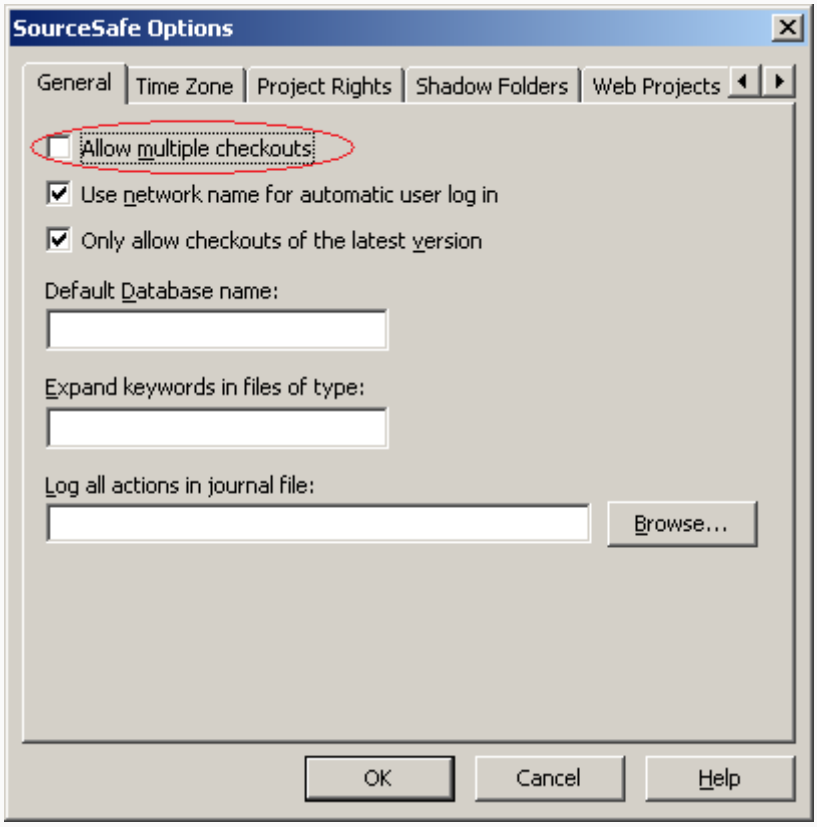
(Project Difference in Visual SourceSafe / VSS)

The diff result shows you which files are only in the local folder, which files are only in the VSS database and which files are different. You can do version control operations, like check out, get and view, directly in this interface. There is also a powerful feature called Reconcile All, which can synchronize your whole local folder and VSS database.

Easy Collaboration

VSS makes team collaboration easy and intuitive. When you add/check in a file to the VSS database, the file is available to other users. The team members, if they have sufficient permission, can see the latest or the previous versions of the file and can also make changes to the file. When you have multiple team members, this is really useful. Everyone can access the latest files and make changes without worrying about working on the outdated files or overwriting the changes made by other members.

When “Allow multiple checkouts” option is selected, VSS also supports parallel development, which allows individual team members to work on different parts of a file at the same time.



(Allow Multiple-Checkouts in Visual SourceSafe / VSS)

Several Important Concepts

VSS Database/Repository

VSS database/repository is the central place where all files, history, project structures, permission and user information are stored. VSS database consists of hundreds of or even thousands of individual files with some strange names like aaaaaaa.a or baaaaaaa,a. The format of the VSS database is unknown to the public since Microsoft never published the format/specification.

Working Folder

Working folder is the specified corresponding folders on a user's local computer used to store files when working with VSS projects. By default, the files are retrieved to the corresponding working folder when a user does Get or Check Out. A user can make changes to files in the working folder and then checks the modified files back into the VSS database for version tracking.
Working folder is required for many VSS operations. For example, if a working folder is not set, when you do Get Latest Version, VSS will prompt you to set a working folder for the current VSS project or else the Get operation cannot continue.

Master Copy and Local Copy

A master copy is a copy of a file stored in a VSS database. A local copy is a copy of a file stored in your working folder on your local computer. The reason why the copy in VSS database is called a master copy is that the VSS database is the central point for file synchronization. After you make changes to your local copy, you need to check the file into the VSS database so that your team member can access the file content changed by you. You can do Get Latest Version to retrieve the latest version. More importantly, before making any changes, you need to Check Out the file first. The Check Out operation marks the file in the VSS database as checked out and retrieves the latest version to your local hard disk.

Most version control systems have similar concepts with the ones in VSS.

Basic operations

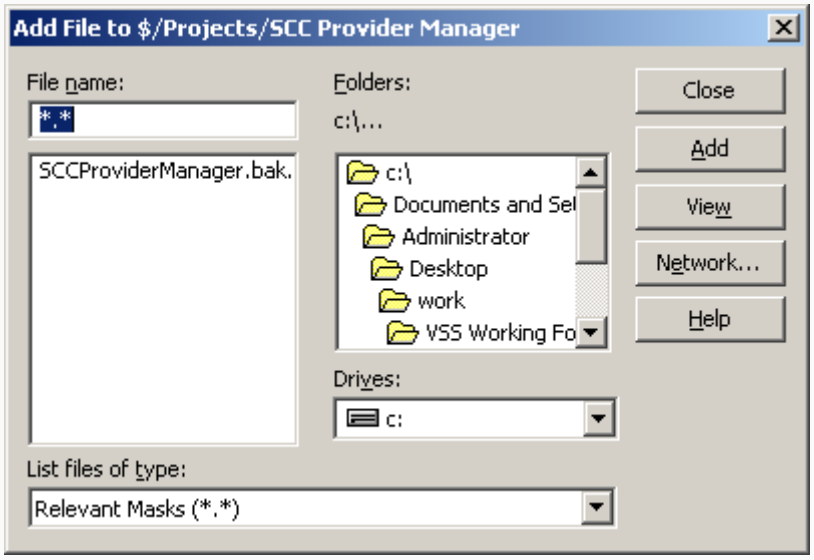
I only put basic introduction of SourceSafe / VSS version control operations in this post. For detailed information about SourceSafe operations, like Add, Get, Check In and Undo, please refer to the upcoming posts.

Add

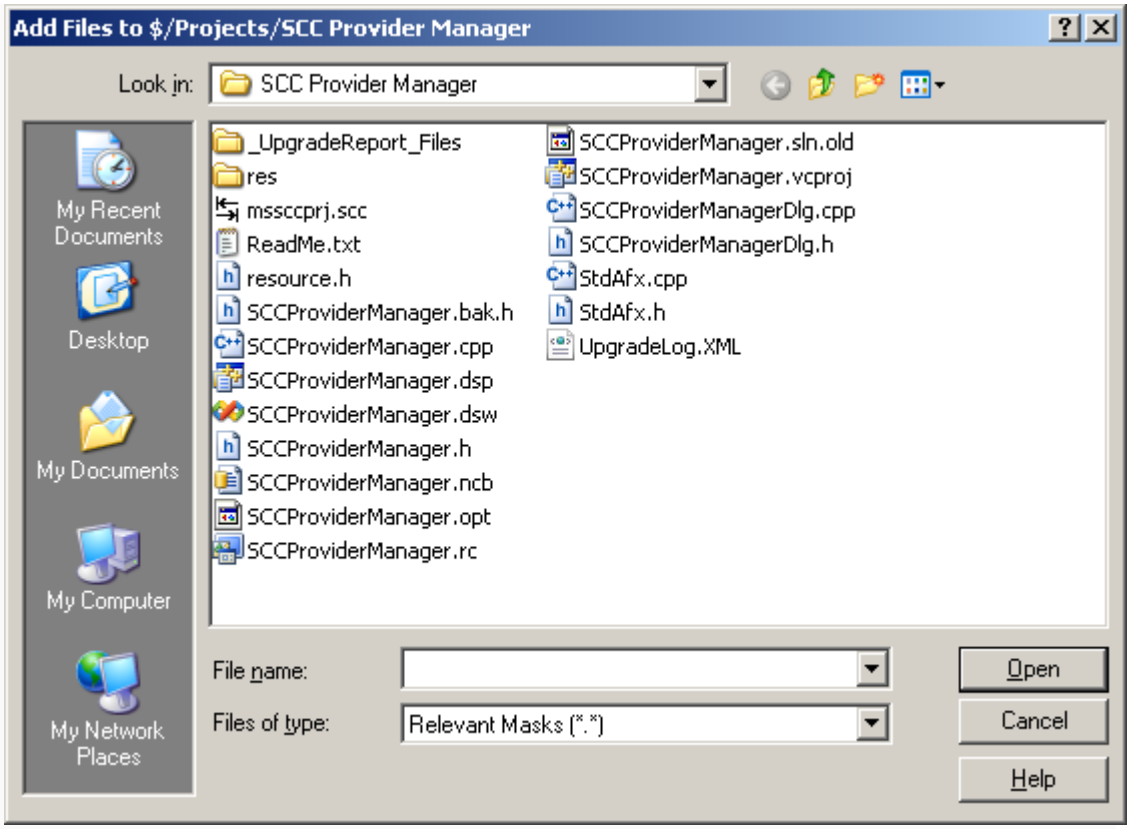
Add is the starting point of all VSS file operations. Before you can do Get, Check In, Check Out or other operations, the file needs to be under the control of VSS first.

You can add individual files and folders into VSS database.

Screen shots of Add File:



(Screen shot: Add File in VSS 6)



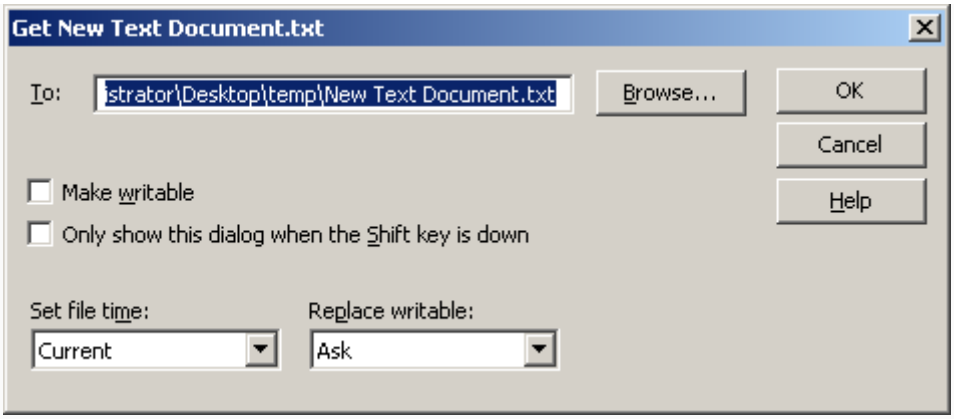
(Screen shot: Add File in VSS 2005)

Get

When you want to view/get a file or project, but not modify it, you can use the Get or View command. Get copies the file or project from the current project into your working folder.

Get Latest Version, which retrieves the most recent version of a file or a project to your working folder, is the most commonly used command in VSS.

Screen Shot of Get:



(Screen shot: Get Latest)

Check Out

To make changes to a file you must first check it out of the VSS database. When you Check Out an item, VSS retrieves the latest copy of the file to your working folder and make it writable.

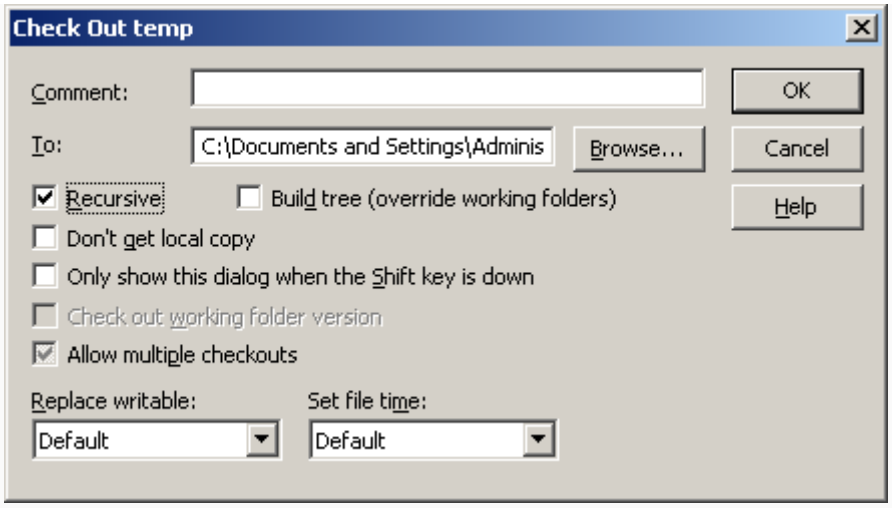
There are exclusive check out and non-exclusive check out. If a file is checked out exclusively, it cannot be checked out by other users.

Please always check out a file first before making any changes for the following reasons:

1. If a file is not checked out, it cannot be checked in;
2. If a file is not checked out, your local copy may not be the latest copy. Not working on a latest copy may cause extra work, overwriting other's changes, confusion and many other problems;
3. If we change a file without checking it out first, other team members can check out the file and make changes to it, which can cause huge problem if parallel changes to a file is not desired.

This is the classical lock-modify-unlock model of VSS. Other tools like CVS and SubVersion have very different models. The latest VSS 2005 also supports the Copy-Modify-Merge model. We will discuss the details later in the coming articles.

Screen shot of Check Out:

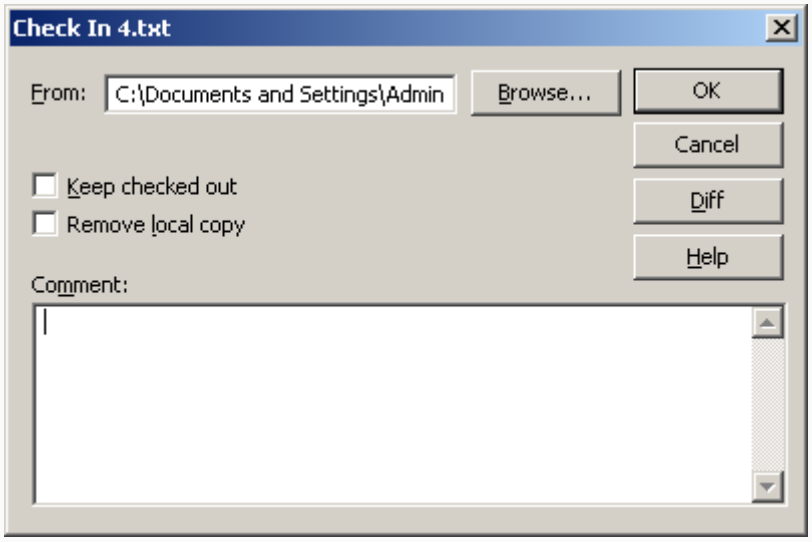


(Screen shot: Check Out)

Check In

After you have finished the changes, you need to Check In the updated file back into VSS. When we say VSS keeps all your previous changes, we are not saying that VSS keeps every change you make, we are saying that VSS keeps all the versions you checked in. If you do not check in your file, there is no way that VSS knows that you have changed the file and it should keep it.

Screen shot of Check In:

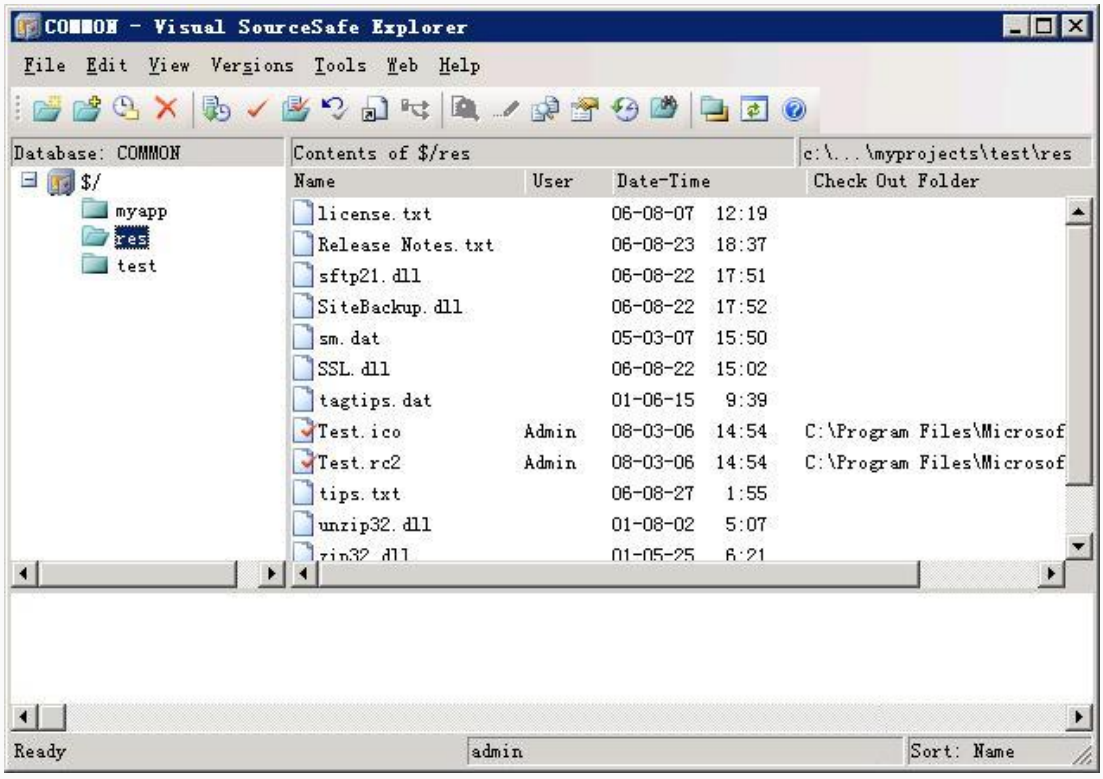


(Screen shot: Check In)

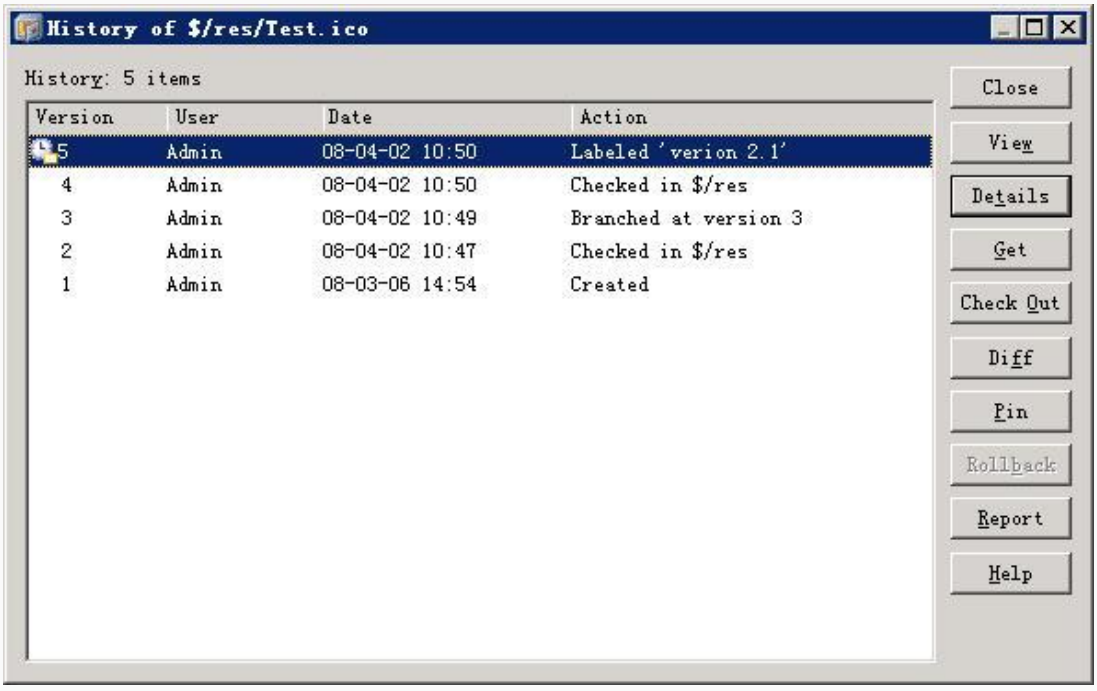
Some Screenshots

The following screenshots shows the typical interface of SourceSafe.

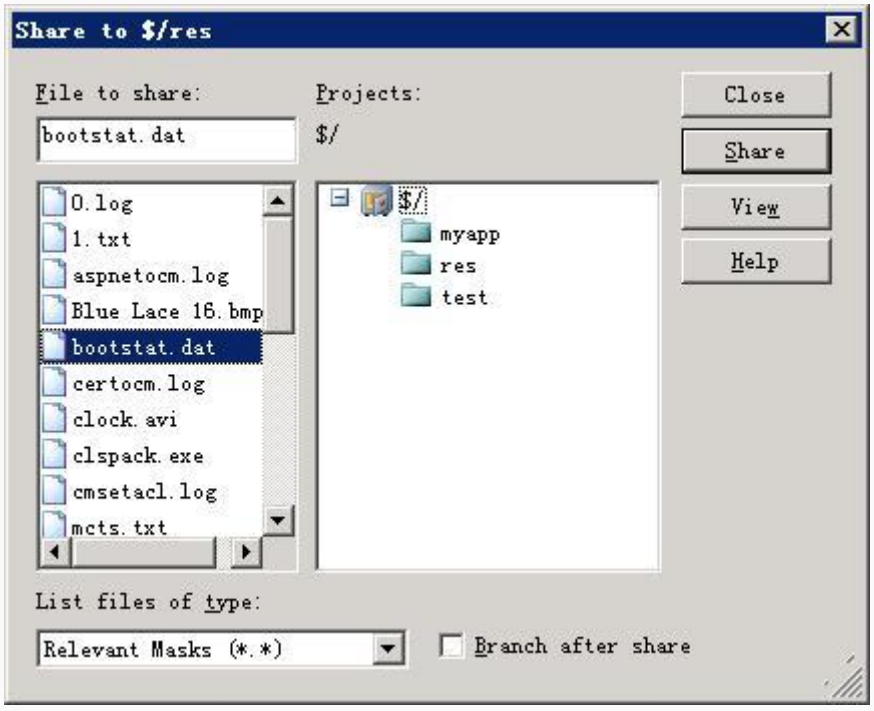
(Log on to VSS DB)



(Client Window)



(File History)



(Share)

Working folder

Working folder introduction

A working folder is the specified corresponding folders on a user's local computer used to store files when working with VSS projects. By default, the files are retrieved to the corresponding working folder when a user does **Get** or **Check Out**. A user can make changes to files in the working folder and then check the modified files back into the VSS database for version tracking.

A working folder is required for many version control commands. For example, if a working folder is not set, when you do **Get Latest Version**, VSS will prompt you to set a working folder for the current VSS project or else the **Get** operation cannot continue.

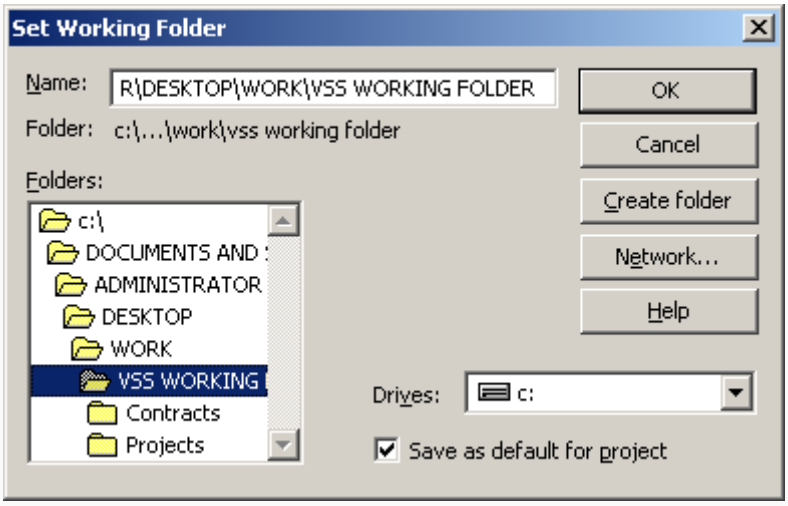
View File is the only command working on files that does not require a working folder is.

A working folder is based on per project, per user and per machine. That means one user can have different working folders for different VSS projects on one machine, different users can have different working folders for the same VSS project on the same machine and one user can have different working folders for a VSS project on different machines.

Setting working folder

To set a working folder for a VSS project, you can select a VSS project first, then click the**Set Working Folder** icon in the toolbar or click **Set Working Folder** on the **File** menu.

The following figure shows the **Set Working Folder** dialog box of VSS 6. This dialog box was designed a long time ago and we can see the old-style presentation of the UI.



(Set working folder dialog box in VSS 6)

This design was kept for more than 10 years and when Microsoft released Visual SourceSafe 2005, the SourceSafe team had a new implementation of the Set Working Folder dialog box. A system standard Browse For Folder dialog box is used.

(Set working folder dialog box in VSS 2005)

However, there is a serious flaw in the new Set Working Folder interface. I will explain it in the coming Working Folder Is Inheritable and Clearing Working Folder sections.

Working folder is inheritable

Working folder is inheritable, which means that if a working folder is set for a parent project, the sub projects automatically inherit the relative working folder from its nearest parent. For example, we have a VSS project of $/Work/WebService, with a working folder of C:\Kevin\Work\WebService. For $/Work/WebService/Res, if its working folder is not set explicitly, it has a working folder of C:\Kevin\Work\WebService\Res automatically. In most cases, it is recommended that you only need to set the working folder for a parent project and let all the sub projects inherit working folders from the parent. But if you like, you can always set working folder for any project explicitly.

If a working folder is set for a project, the project does not inherit working folders from its nearest parent anymore.

Clear working folder

Sometime, you need to clear the working folder settings of a project to make it inherit working folder from its parent.

To clear the working folder, you just set the working folder to blank.

But in the SourceSafe 2005 set working folder interface, there is NO way to set the folder to blank. If we ever set a working folder for a project in VSS 2005, there is no way to clear it so it is impossible for the project to inherit working folders from its nearest parent. This caused a huge problem for our team. Luckily, the Microsoft Visual SourceSafe had an undocumented “backdoor” to handle this problem. To bring up the VSS 6 style Set Working Folder dialog box, you need to press the Shift key and then click the toolbar icon or the menu item.

Add operation of SourceSafe

Add introduction

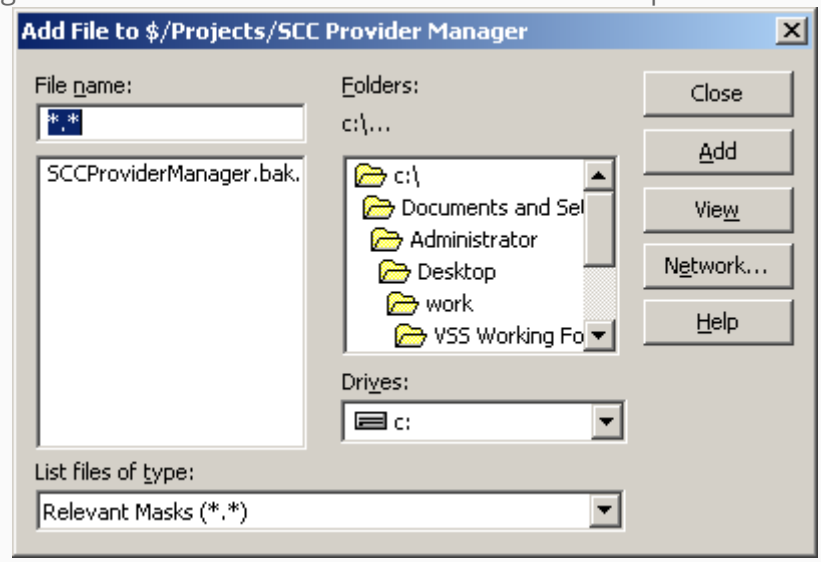
A file must be added in a VSS database first before you can do check in, check out and other operations. To add one or more files, you can click **Add File** on the **File** menu or use drag and drop from **Windows Explorer** to **SourceSafe Explorer**.

You can also add a whole folder into SourceSafe recursively. If there are some files or sub projects in VSS already, VSS keeps the existing files and sub projects and only adds the ones not in VSS.

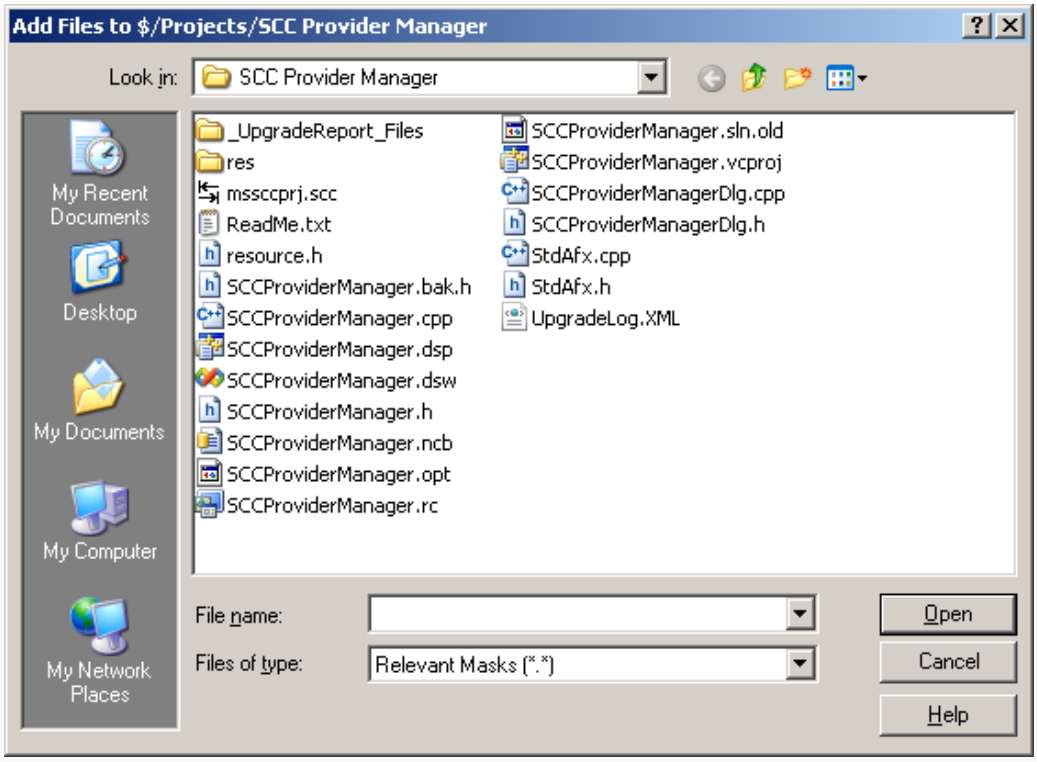
Differences between SourceSafe 6 and 2005

1. Different interface

   The dialog boxes of Add File in SourceSafe 6 and 2005 are quite different:



(Add File dialog box in VSS 6)



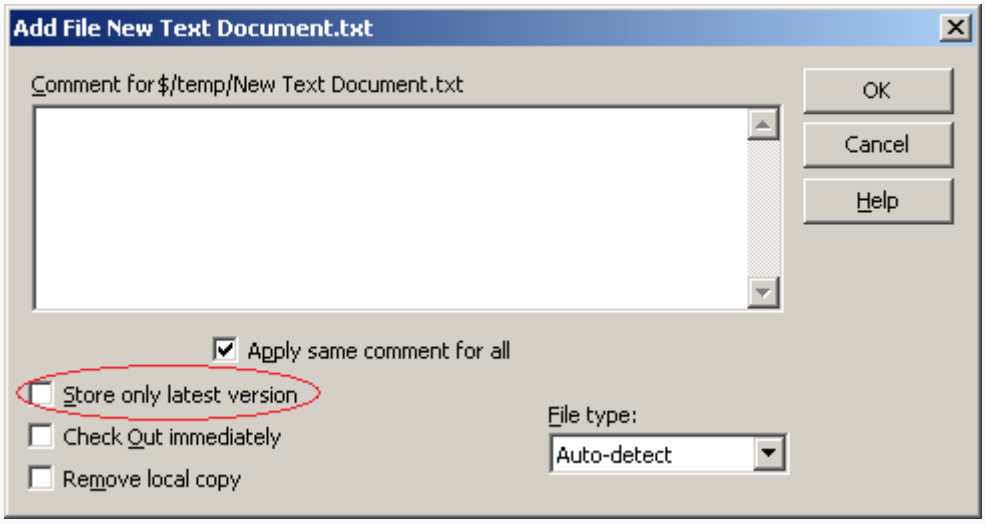
(Add File dialog box in VSS 2005)

2. Folder cannot be added in the VSS 2005 **Add File** dialog box. You can use Drag and Drop to add folders to VSS.
3. The files already under version control are not filtered in the VSS 2005 **Add File**dialog box. In the VSS 6 **Add File** dialog box, only the files not in the VSS database are shown. But in VSS 2005, all the files are shown. This is very inconvenient if you have tens of or even hundreds of files. For example, if you have 100 files in the VSS database already and newly added 2 files. To add the 2 new files into VSS, in VSS 6, you will see only 2 files in the Add File dialog box. But in VSS 2005, you will see 102 files and you must find the 2 new files.

The Microsoft Visual SourceSafe team is aware of this issue and they have an undocumented secret feature. By pressing the Shift button and then launch Add File dialog box, the classical VSS 6 style Add File dialog box will be brought up.

Store only latest version

There is a Store Only Latest Version option in the Add File option dialog box:



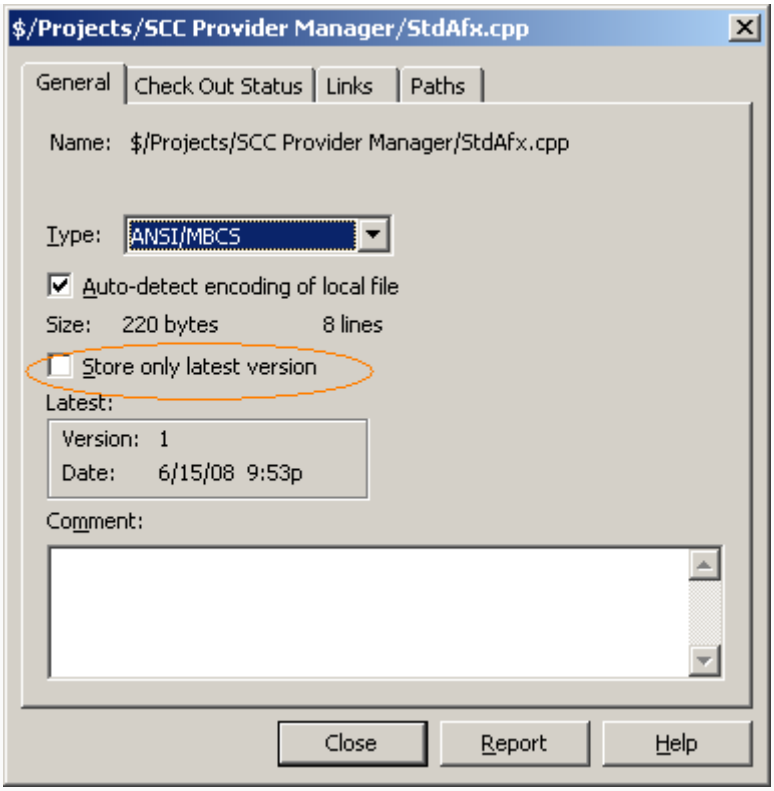
(Store only latest version option)

If this option is selected, only the file content of the latest version is stored. Even though the history information is stored as usual, when you want to retrieve the file content of the previous version, a "File [%filename] does not retain old versions of itself" error will be shown.

Since only the file content of the latest version is stored when this option is selected, the VSS database is smaller and the operations can be faster.

This option is useful when we use VSS to store files but do not care about the previous versions. For example, every time when we build a new release for our test team, we add the executable file to VSS. For this executable file, it may be desirable that we select the Store only latest version option.

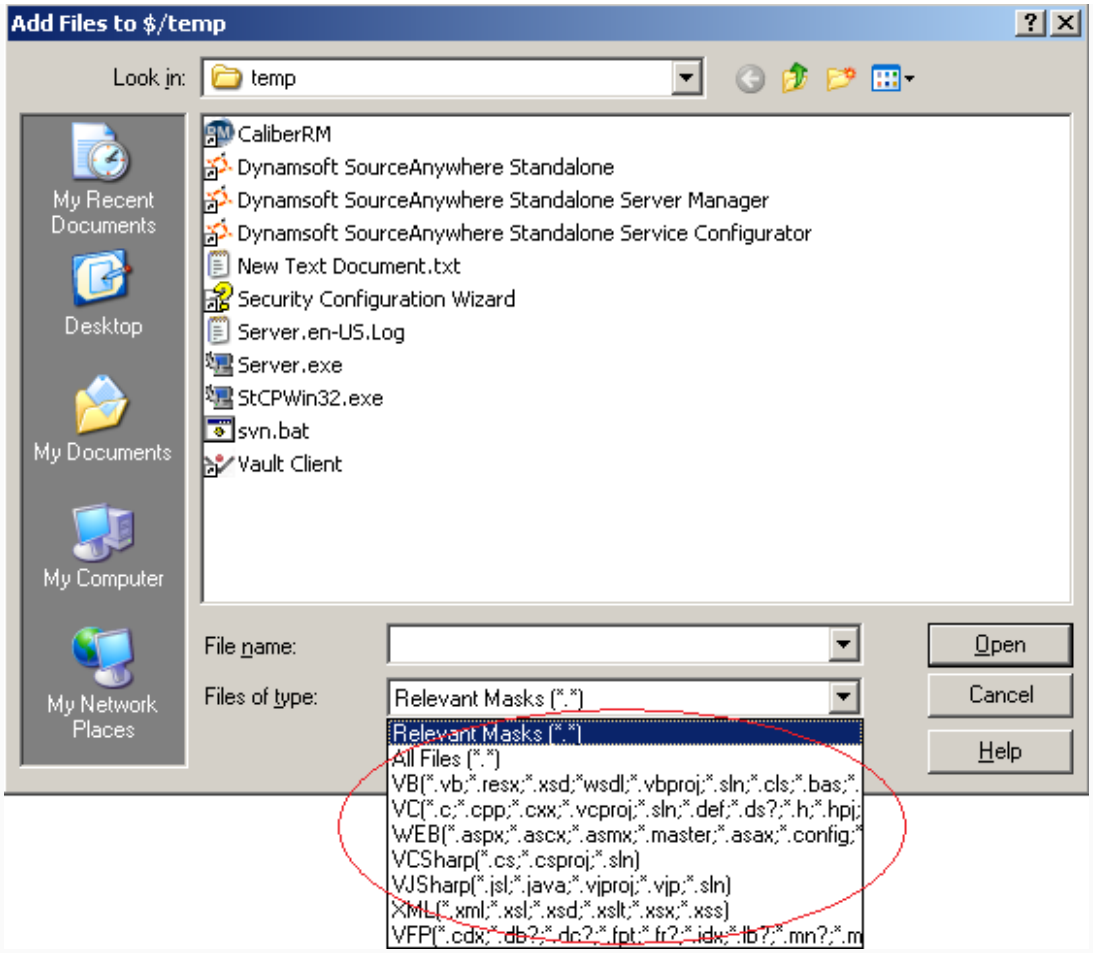
You can use the following dialog box to change the option:



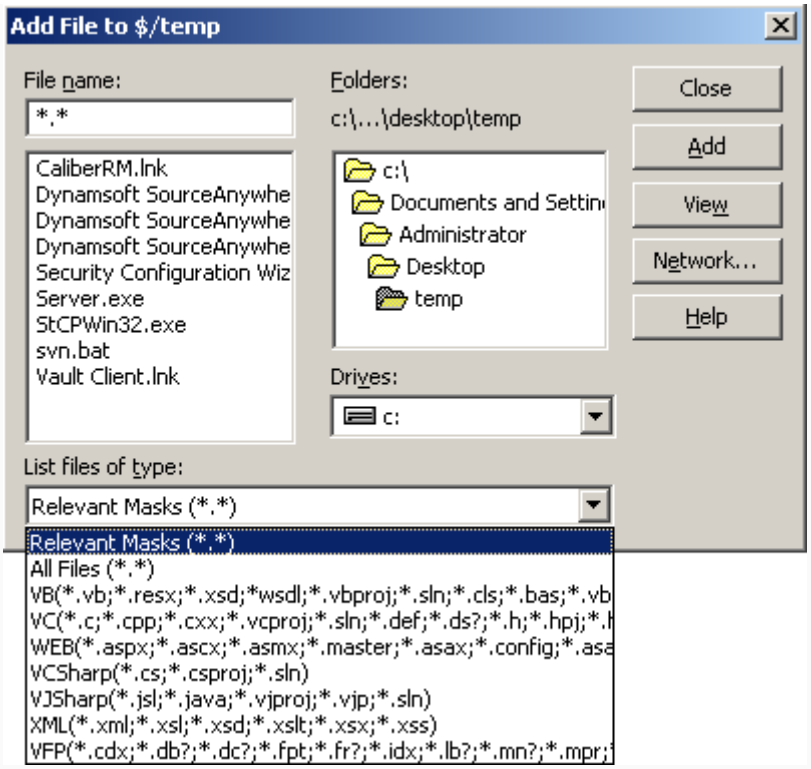
(File property – Store only latest version)

File type filter

In the **Add File** dialog box, you can use **File Type** combo box to filter the files, which makes it easier to find the files you are looking for:

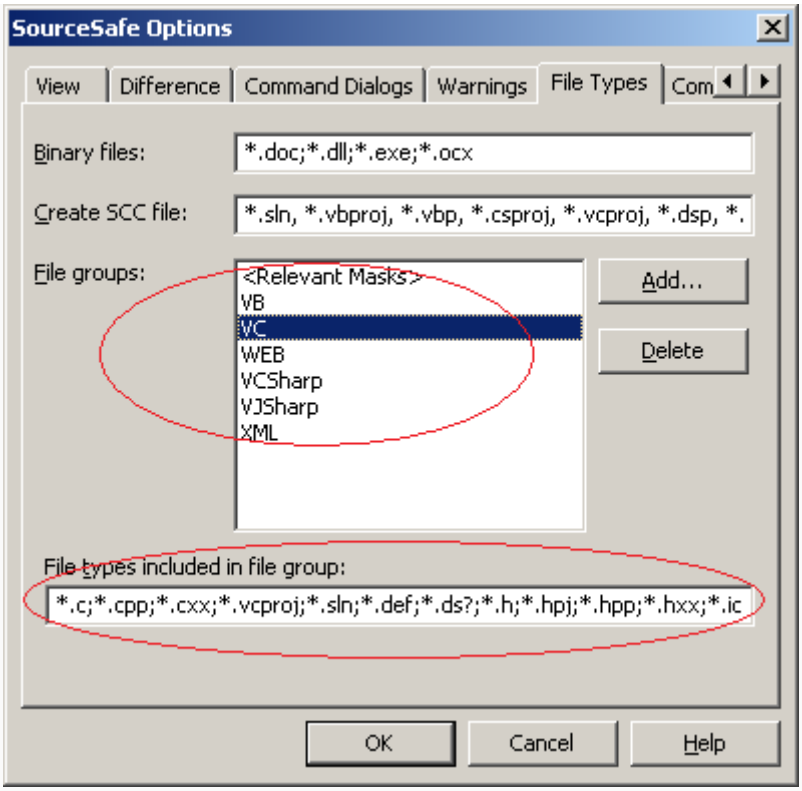


(File type filter in VSS 2005)



(File type filter in VSS 6)

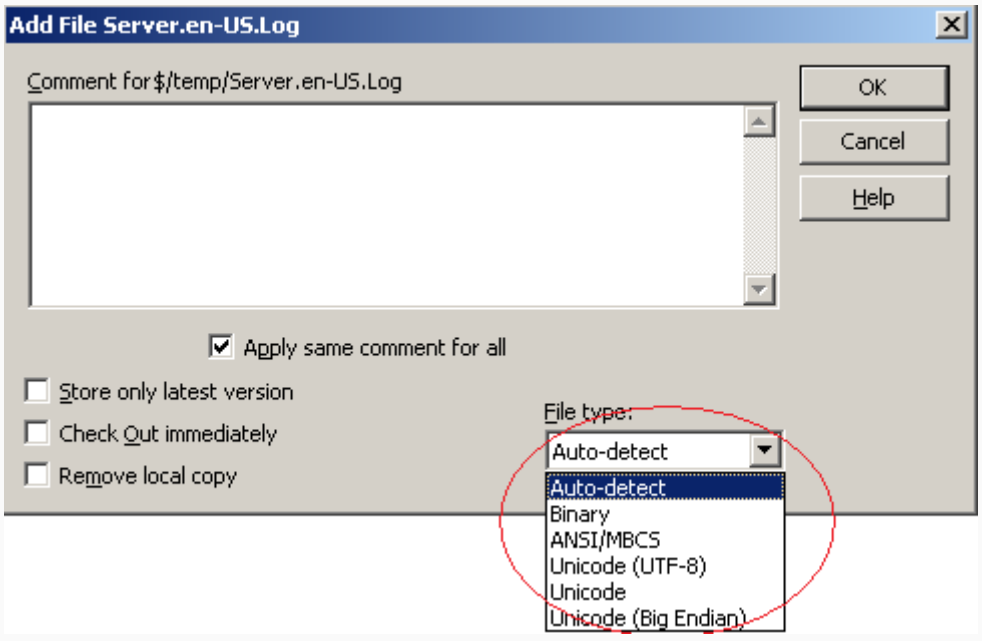
You can manage the **file type group** through SourceSafe Administration tool and SourceSafe Explorer. The settings specified in the SourceSafe Administration tool affect all users. The settings in the SourceSafe Explorer affect only the current user. To set **file type group**, select Tools -> Options menu and go to File Types tab:



(File type group)

Binary or not

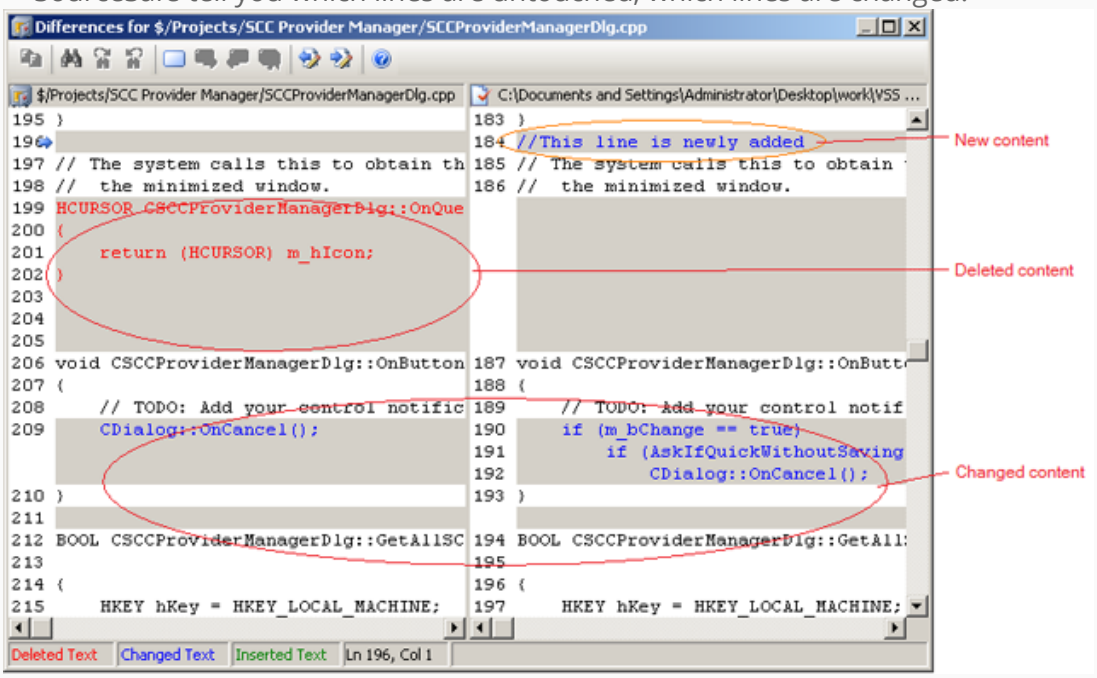
In the Add File dialog box, you can specify that the file is binary:



(File type)

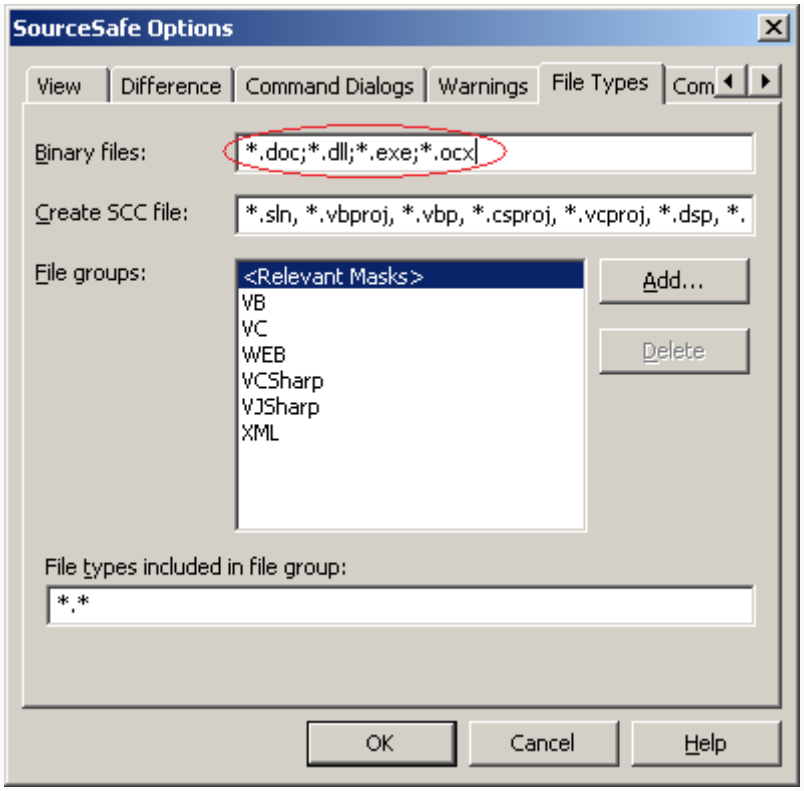
There are two kinds of files: binary file or not-binary file (also called mergeable file). There are two important differences between binary files and mergeable files:

1. Binary files cannot be checked out by multiple users. Even if the Allow multiple checkouts option is enabled, VSS always places exclusive lock on binary files. For mergeable files, if the Allow multiple checkouts option is enabled, multiple users can check out the files. Usually, all the program files are mergeable.
2. Binary files cannot be visually diffed. For binary files, when you do a diff, SourceSafe only tells you if the files are the same or not. For mergeable files, SourceSafe tell you which lines are untouched, which lines are changed:



You can also choose **Auto-Detect** to let SourceSafe decide if a file is binary or not. When you choose **Auto-Detect**, VSS does the following to determine if a file is binary:

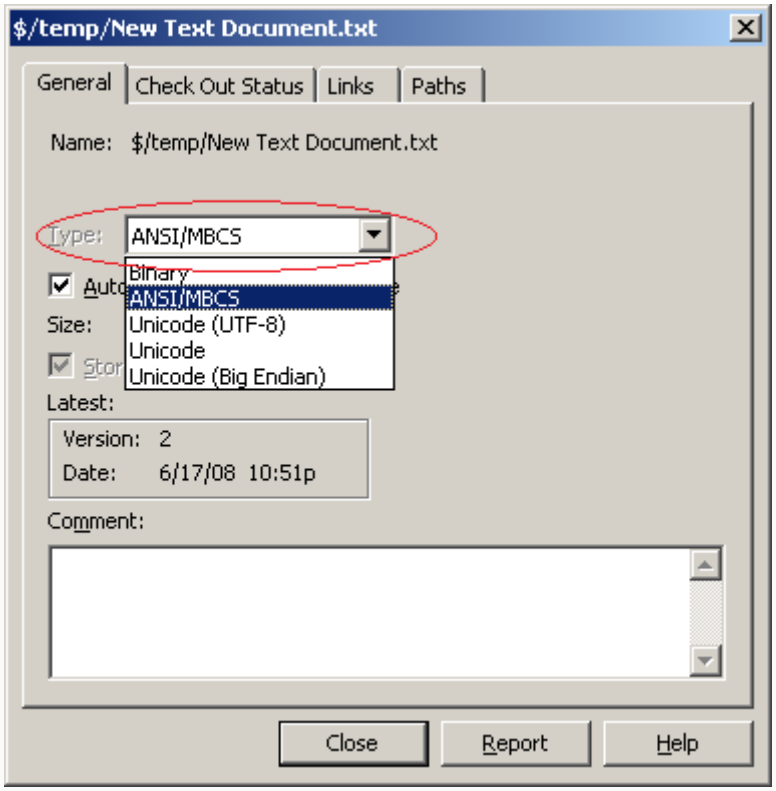
1. If the file type is in the **binary file** extension list, the file is a binary file. Or else, go to step 2. The **binary file** extension list can be specified in the following interface:



(Binary file type)

2. If the file contains a byte with a value of 0, the file is a binary one. Or else, go to step 3
3. The file is mergeable.

   You can also change the file type after the file was added into VSS database:

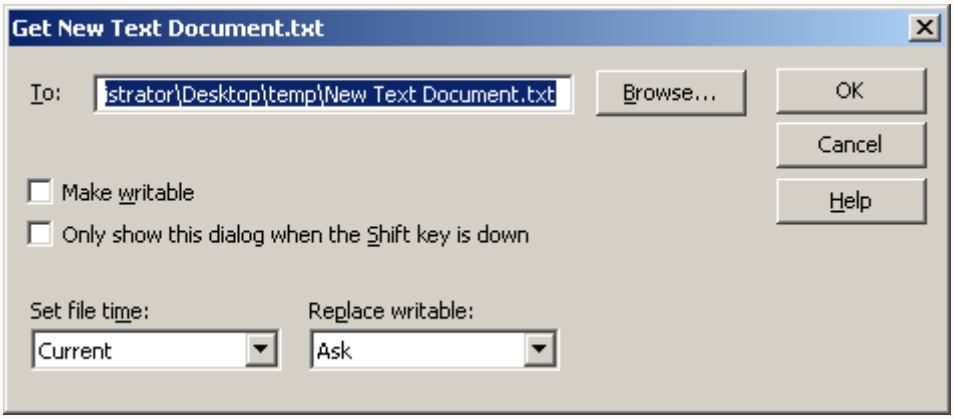


(File type in file property dialog box)

Get Latest

Get Latest Version Basics

Get Latest, along with Check Out and Check In, is one of the most common operations in many version control systems.
The screen shot of Get Latest Version:



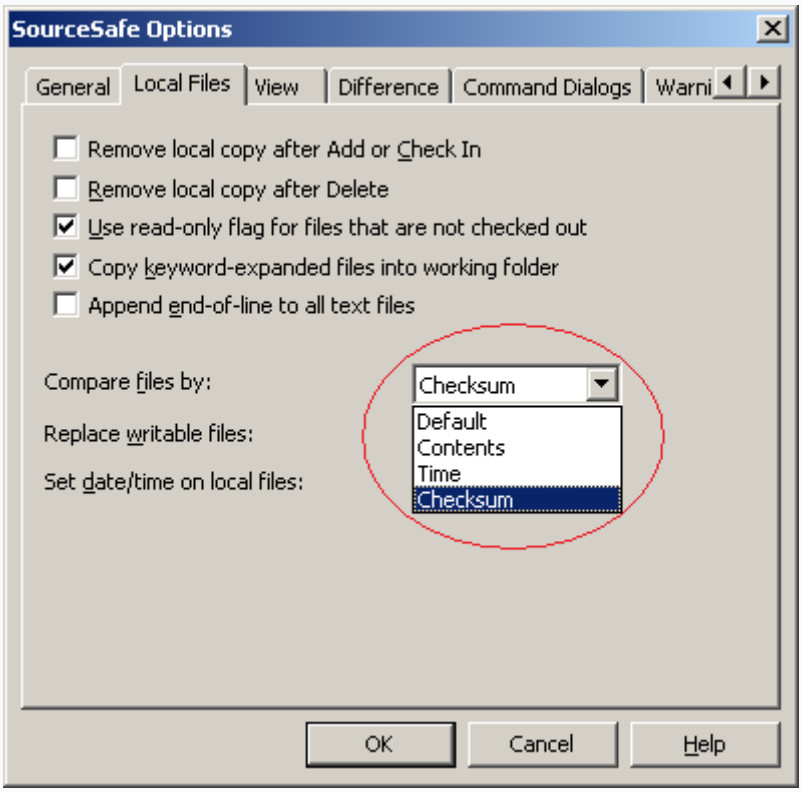
(Get Latest Version screen shot)

The **Get Latest Version** command retrieves the latest or the pinned version of the files to our local disk. For pinned files, the pinned version is retrieved, instead of the latest version.
A working folder is required for the Get Latest Version command.

The **Get Latest Version** command is the most commonly used command in SourceSafe. Usually when we come to work in the morning, want to compile a solution, or simply want to view the changes by our team members, we do **Get Latest Version**.
To get the previous versions by date, version or label, we need to use **Show History**command and then work on the previous versions.
The **Get Latest Version** can automatically detect which files are changed and only get the changed files. So even if we have a huge project, there is no problem, just go ahead and carry out the Get Latest Version function. There will be no big performance penalty. There is an option in VSS to let us specify how VSS determines if our local copy is the latest one. I will talk about the option in the following section.

How does VSS determine if our local copy is the latest version?

The interface for specifying how VSS compares files:



(Compare files by)

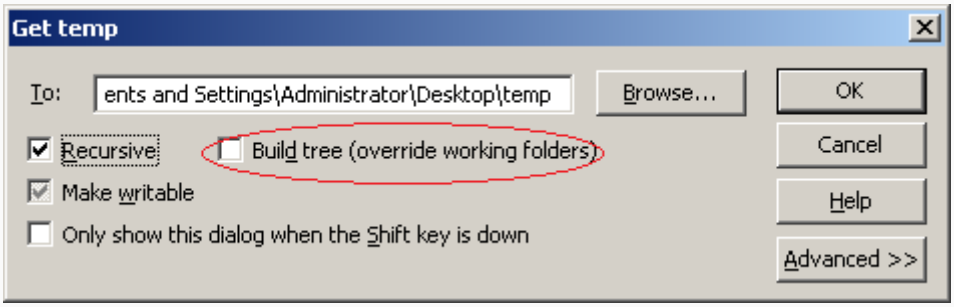
The default option is checksum, which is enough for most cases.

Compare by	Details
Default	By checksum.
Contents	VSS compares our local file against the database version byte by byte. If we choose this option, the performance of get, check in and check out will be much slower.
Time	This is the trickiest option. Even the explanation in the SourceSafe official help document is not completely correct. There are lots of details about this option. If you have any question, you can contact me.
Checksum	This is the default option, which is enough for most cases. According to Microsoft, comparison by checksum is the fastest way.VSS uses 4-

byte/32-bit CRC checksum. Since CRC is not strong enough (much weaker than MD4 or MD5), there is possibility of collision (different file contents have the same checksum).

Build tree

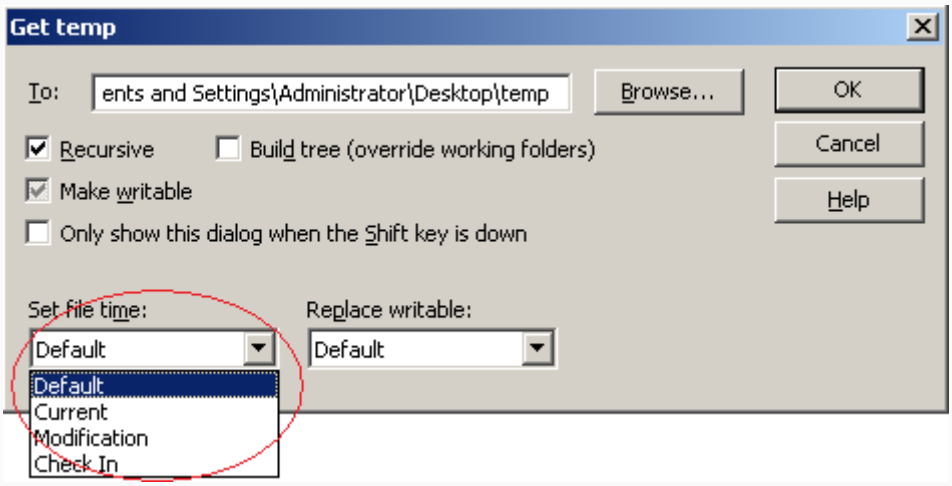
When we try to get the latest of a project, if we select the Recursive option, a Build tree option is shown:



(Build tree option in Get Latest dialog box)

What does **Build tree** mean? When the **Build tree** option is not selected, if a sub project has a working folder explicitly set, the files under that project will be retrieved to the working folder. When the **Build tree** option is selected, VSS ignores/overrides the working folders setting of the sub projects and retrieves the files to the relative sub folders according to the VSS database project structure. For example, we have $/working/projects with the working folder set to C:\VSS\working\projects, and $/working/projects/SCCManager with the working folder set to D:\SCCManager. When we do Get Latest recursively on $/working/projects, all the files directly under $/working/projects are retrieved to C:\VSS\working\projects. That is for sure. If the **Build tree** option is selected, all the files under $/working/projects/SCCManager will be retrieved to C:\VSS\working\projects\SCCManager (the working folder setting of $/working/projects/SCCManager is overridden). If the **Build tree** option is not selected, all the files under $/working/projects/SCCManager will be retrieved to D:\SCCManager (the working folder setting of $/working/projects/SCCManager is used).

Set file time



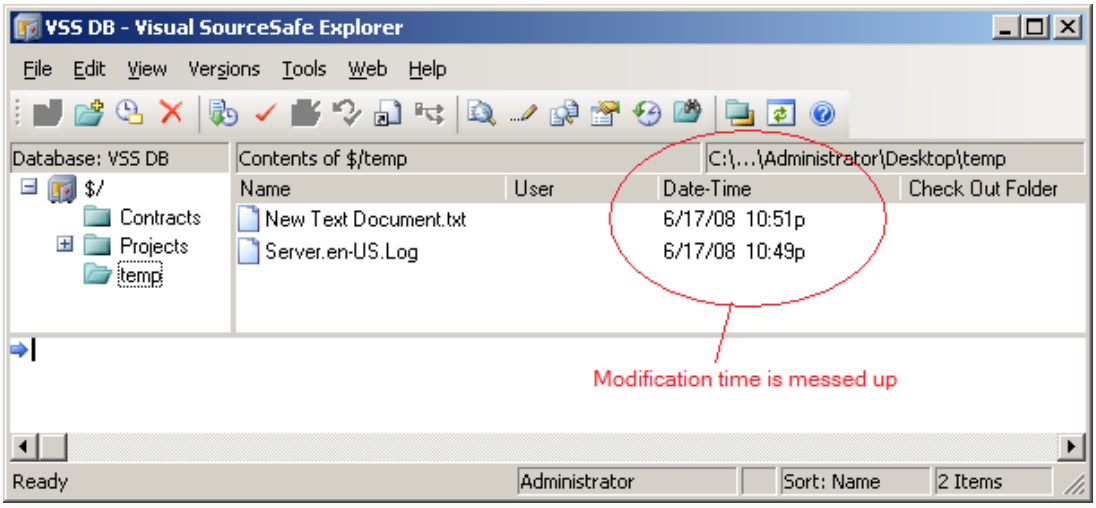
(Set file time)

When we do Get or Check out, VSS allows us to choose how VSS sets the file time.

Before we talk about the options, I need to point out that the time setting mechanism in VSS has issues if we have a distributed team. VSS was designed for local teams more than 10 years ago. Time zone difference is not incorporated into the design.

Let's see an example. We have 2 developers using VSS. One is in North America east coast using east time and another is in west coast using pacific time. If the developer checked in a file from the east coast office at 10:00AM ET, and then if another developer checks out the file from the west coast office using check in time for the set **file time** option, the time of the file will be set to 10:00 AM. That is 10:00AM pacific time, not east time.
So why does this can cause problem? The answer is that if our developers are all in the same time zone, there is no problem at all. But if not, there are problems:

- The modification time in the file list of the main interface will be totally messed up:



(Modification time is messed up)

Depending on the time zone of the machine that the file is checked in through, the modification time listed here represents the time in different time zones.

- Our compiler may not be able to compile our solution correctly.
  Most compilers use time to determine if a file is changed and only compile the changed files. If we choose **Modification** or **Check In** for the **Set File Time** option, we have to rebuild the whole project/solution all the time. If we choose Current for the **Set File Time** option, our compiler can work properly, since the files are set to the local time when we retrieve the file.

Set file time	Details
Default	It is Current.
Current	The local time when we do the command.
Modification	The modification of the file when the file was checked in.
Check in	The date and time when the file was checked in.

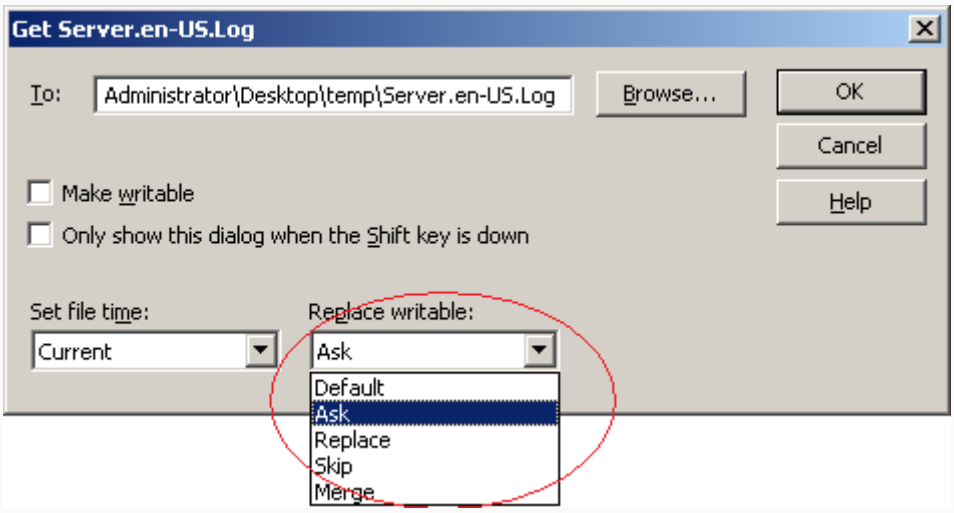
Replace writable

There is a very important concept in SourceSafe: VSS treats read-only and writable files very differently and overwrites read-only files without any warning or prompt. By default, after we add a file into VSS database, the file is set to read-only. And only after we check that file out, the file is set to writable. Under any other circumstances,

the file is read-only. When we do Get or Check out, if the local file is read-only, no matter if the file content is the same as the one in VSS database, VSS overwrites the file.

Best practice: never change the read-only property of a file manually if that file is under VSS control.

VSS lets us specify how it replaces the local writable copy:



(Replace writable)

Replace writable	Details
Default	This is actually Ask.
Ask	VSS shows a dialog box to ask us to choose replace, skip or merge if the content of the local file is different from the one in VSS database.
Replace	Overwrite the local copy. Use this option very carefully, since VSS overwrites our local changes.
Skip	Leave the local copy. The content in the VSS database will not be retrieved to our local disk.
Merge	If we have checked the file out and made some changes, VSS uses its merge tool to merge the changes for us.

Lock-Modify-Unlock or Copy-Modify-Merge

VSS supports two modes of work, Lock-Modify-Unlock and Copy-Modify-Merge. Lock-Modify-Unlock is the default style.

The details of Lock-Modify-Unlock mode:

Under this mode, only one developer can check out a file and work on it. Other developers need to wait for the file to be checked in.

1. Developer A checks out a file, exclusively. An exclusive lock is placed on the file by VSS.
2. Developer A works on the file.
3. If Developer B wants to edit the same file, according to the behavior of SourceSafe, he/she needs to check the file out first. Since the file is exclusive checked out (locked) by Developer A, it cannot be checked out by Developer B.
4. After Developer A is done with the file, he/she checks the file back in. The file is unlocked.
5. Other developers can check out the file and work on it.

The details of Copy-Modify-Merge

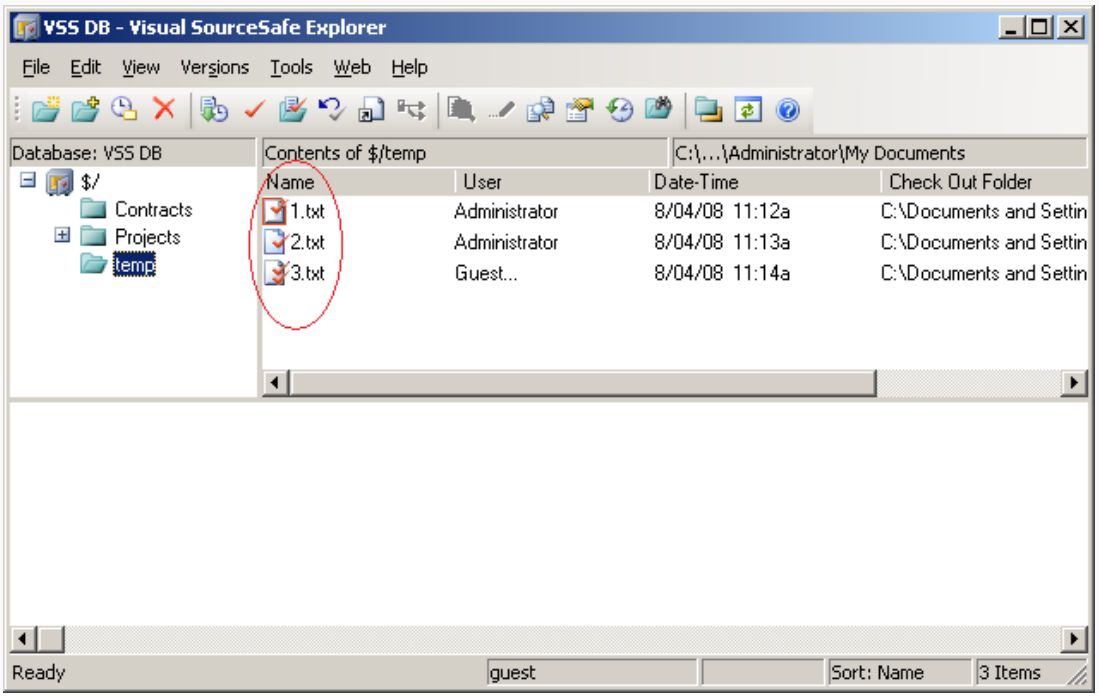
Under this mode, multiple developers can check out a text file not-exclusively and work on it at the same time, which is also called parallel development.

1. Developer A checks out a file, not-exclusively.
2. Developer A works on the file.
3. Developer B wants to edit the same file, according to the behavior of SourceSafe, he/she needs to check the file out first. Since the file is NOT exclusively checked out (locked) by Developer A, the file can also be checked out by Developer B not-exclusively.
4. If Developer C wants to edit the same file, he/she can also check out the file not-exclusively.

5. When someone checks in the file, VSS checks if the current version number is the same as the version number when the file was checked out by the current user.
   - If the current version number is the same as the version number when the file was checked out by the user, a regular check in is performed by VSS.
   - If the current version number in the database is bigger than the version number when the file was checked out by the user (which means that someone else has checked in the file after the current user checked out the file), VSS will try to merge the file content first and check in the merged file content into the database.

Icons for different checkout modes

SourceSafe uses different icon for different check out modes as shown in the following screen shot:

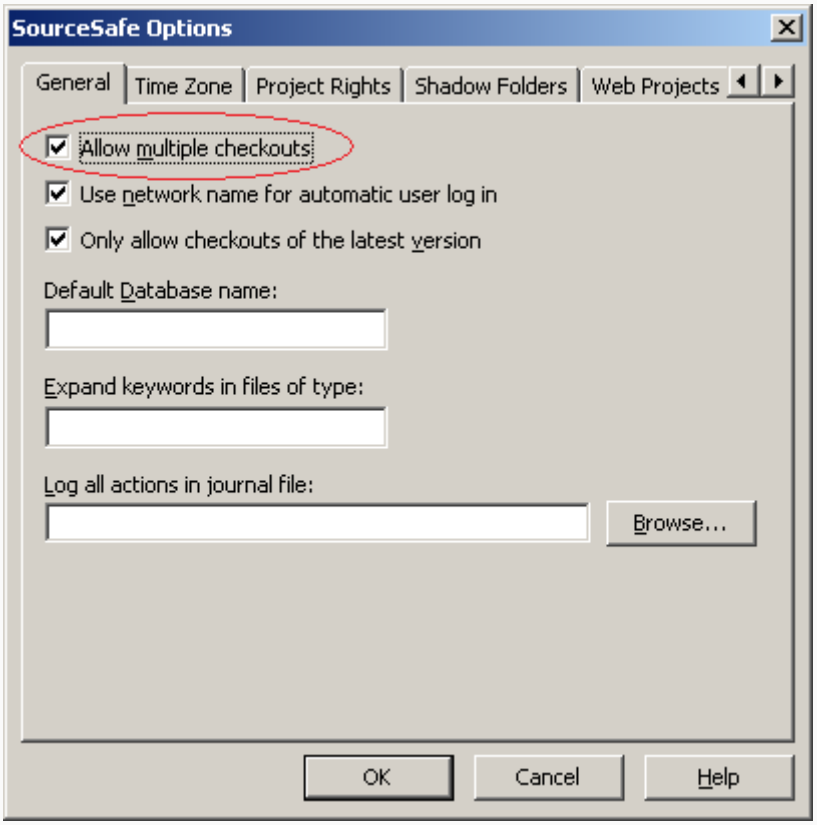


1. Exclusive check out: . In this icon, there is a red box surrounding the icon;
2. Not-exclusive check out by single user: . In this icon, there is no red box surrounding the icon;
3. Not-exclusive check out by multiple-user: . In this icon, there is no red box surrounding the icon and there are two ticks in the icon.

How to choose the Lock-Modify-Unlock or Copy-Modify-Merge mode

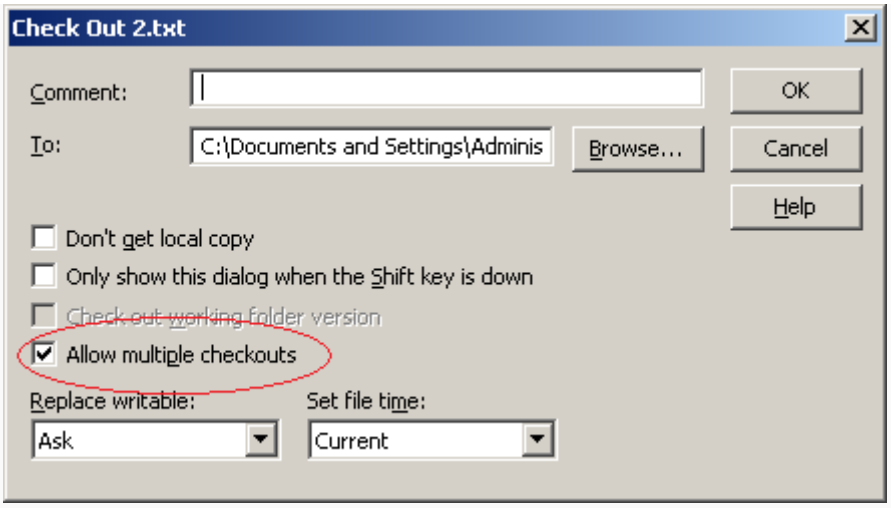
1. Multiple check out needs to be enabled at the database side. To enable multiple check out:
   - Start SourceSafe Administrator;
   - Choose menu Tools -> Options, and click the **General** tab;
   - Check the **Allow multiple checkouts** checkbox.

   The interface is shown as follows:



(Enabling multiple checkouts)

2. After the **Allow multiple checkouts** option is enabled on the server side, we can choose that if we want to check out a file exclusively on the client side. Please note that binary file can only be checked out exclusively no matter if the **Allow multiple checkouts** option is enabled or not.

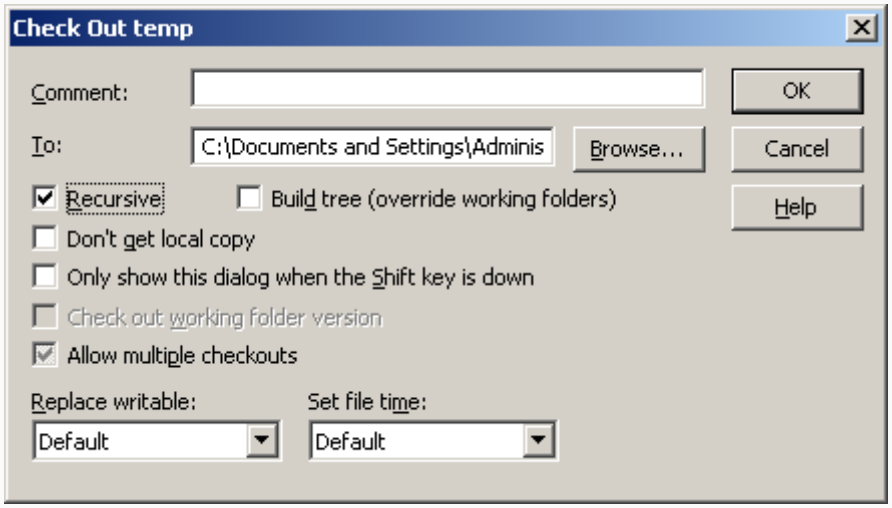


(Allow multiple checkouts at the client side)

Check Out

Check out basics

Check Out, along with Get Latest and Check In, is one of the most common operations in many version control systems.
The screen shot of Check Out in VSS 2005:



(Screen shot of Check Out project)

If we are checking out files (not a project), the **Recursive and Build Tree** options are not available.
In VSS 2005, we can check out a previous version in the **history** dialog box. In this article, we only talk about checking out the latest version.
To make changes to a file we must first check it out of the VSS database. When we **Check Out** an item, VSS retrieves the latest copy of the file to our working folder and makes it writable.
There are exclusive check out and non-exclusive check out. If a file is checked out exclusively, it cannot be checked out by other users.

Please always check out a file first before making any changes for the following reasons:

1. If a file is not checked out, it cannot be checked in;

2. If a file is not checked out, our local copy may not be the latest copy. Not working on a latest copy will definitely cause extra work, overwriting other's change, confusion and many other problems;
3. If we change a file without checking it out first, other team members can check out the file and make changes to it, which can cause huge problem if parallel changing to a file is not desired.

Recursive, Build tree, Replace writable and Set file time

Please see the Get Latest article for details.

Allow multiple checkouts

There are two types of check out in VSS: exclusive check out or non-exclusive check out. For binary files, VSS automatically makes every checkout exclusive, since binary files cannot be merged. For mergeable files, if multiple checkouts is allowed by the administrator, we can choose to check out a file exclusively or not. If a file is checked out, but not exclusively, other team members can check out the same file and make changes to it.

Please see Lock-Modify-Unlock or Copy-Modify-Merge for details.

File Diff

Why File Diff

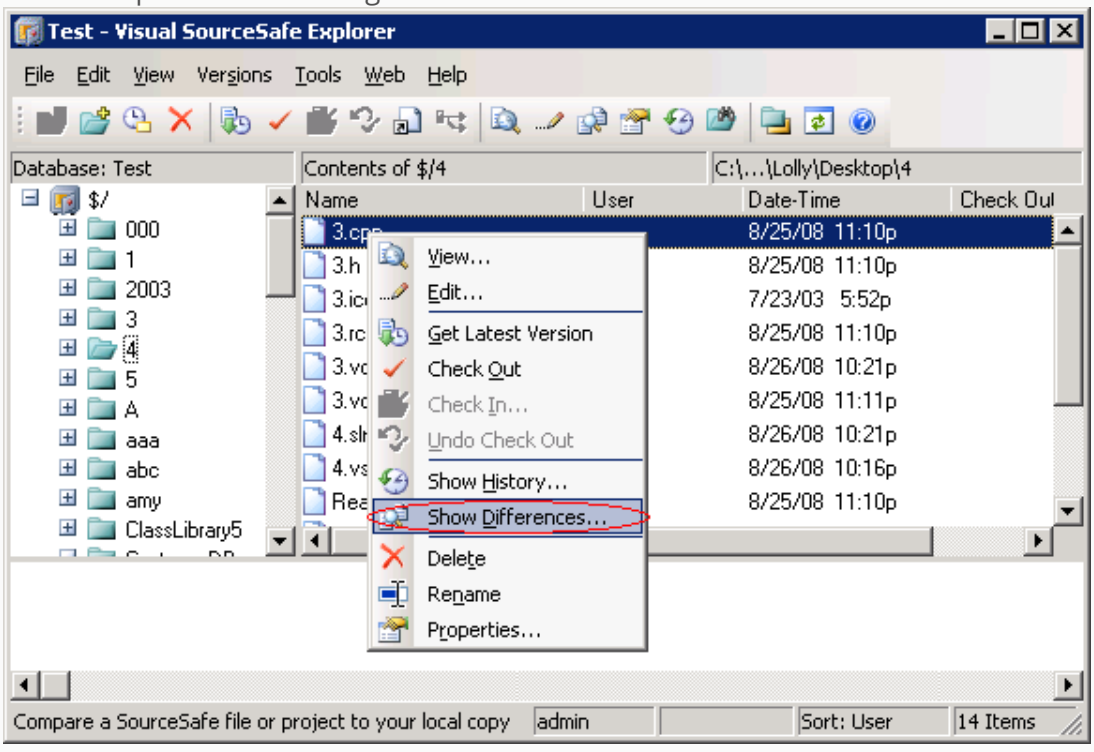
File Diff is one of the important features that we use with a version control system. File Diff compares two files and shows the differences. With these differences, we can better track the modification of our code or documents.

What File Diff can do

File Diff can compare:

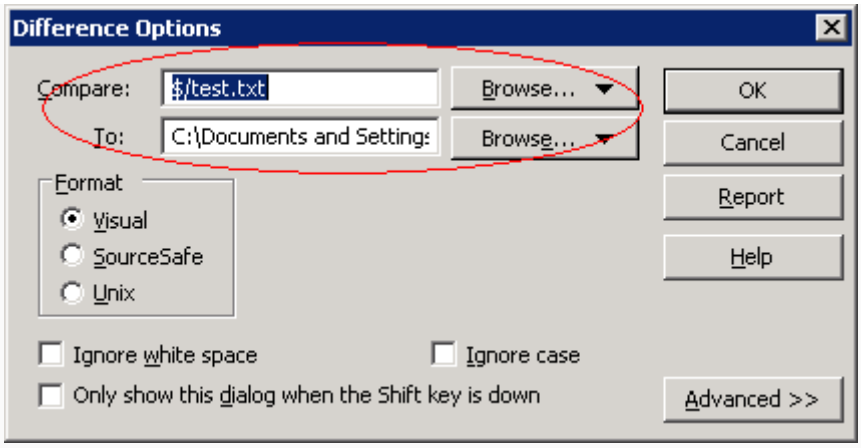
- A local file and a source-controlled file.
- Two local files.
- Two source-controlled files.
- Two versions of a source-controlled file.

For the first three situations, we can launch Difference Options by right clicking on the file list pane and selecting **Show Differences...**



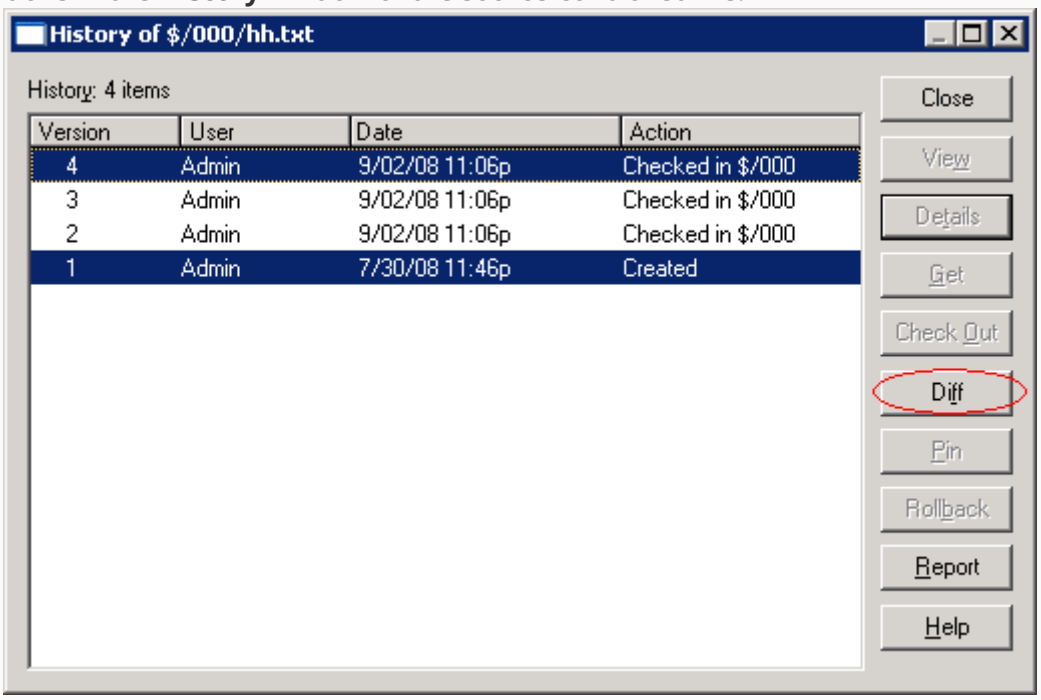
(Launch Difference Options)

In Difference Options, we can specify the two files, local or source-controlled.



(Specify the two files for comparison)

To compare two versions of a source-controlled file, we can launch Difference Options in the **History** window of the source-controlled file.



(Launch Difference Options in History)

For binary files, File Diff can indicate if they are identical but cannot show the differences.

How File Diff works

File Diff compares the two files line by line.

- If a line exists in the first file only, SourceSafe indicates this line as deleted.
- If a line exists in the second file only, SourceSafe indicates this line as added.
- If a line in both two files has different contents, SourceSafe indicates this line as changed.

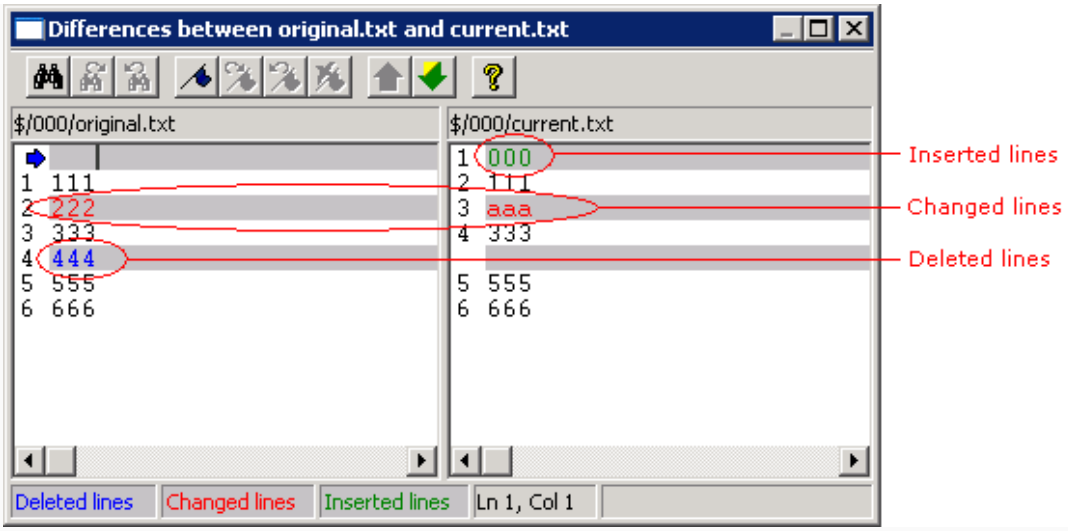
Example of File Diff

For example, we have an original file with the content:
111
222
333
444
555
666

After we make the following changes:
Added '000' before the first line
Changed 2nd line from '222' to 'aaa'
Deleted the 4th line

We get the current file with the content:
000
111
aaa
333
555
666

For the diff result:


Differences between original.txt and current.txt
$/000/original.txt
$/000/current.txt
1 111
2 222
3 333
4 444
5 555
6 666
1 000
2 111
3 aaa
4 333
5 555
6 666
Inserted lines
Changed lines
Deleted lines
Deleted lines
Changed lines
Inserted lines
Ln 1, Col 1

File Merge

To better understand how files are merged, you first need know something about how diff works. If you have not read the article: Using Visual SourceSafe – File Diff, please click here.

File Merge Introduction

File merge is the foundation of parallel development. Different team members can work on the same file at the same time and merge their changes back when they are done.

SourceSafe supports two work modes, the lock-modify-unlock mode and the copy-modify-merge mode. Lock-modify-unlock is the default mode and in this mode, only one person can work on a file at one time and file merge is avoided in most cases.

How merge is performed

Before discussing how merge is performed in version control, we first need to understand the concept of 3 parties in merge and 3 types of change.

3 parties in merge:

To perform a merge, 3 parties are needed: base version, source version and target version.

3 types of change:

1. Add. We added new lines into the source code;
2. Delete. We deleted lines from the source code;
3. Change. We changed the content of a line.

Merge steps:

1. SourceSafe compares the source version against the base version to have a list of changes made in the source version;
2. SourceSafe compares the target version against the base version to have a list of changes made in the target version;
3. SourceSafe tries to apply all the changes in the source version to the target version

- If there is no conflict, all the changes in the source version are applied to the target version.
- If there are any conflicts, visual merge is required. SourceSafe shows an interface to let us choose if we want the changes in the source version or the target version. We can also edit the content in the visual merge interface.

How Conflict is Detected

If the source version and the target version change the same line of the code, there is a conflict.

Example of merge

For example, the content of the base version is:

1

2

3

4

In the source version, the content is:

0

1

22

3

4

In the target version, the content is:

1

222

3

4

5

So in your current version, you have made the following changes:

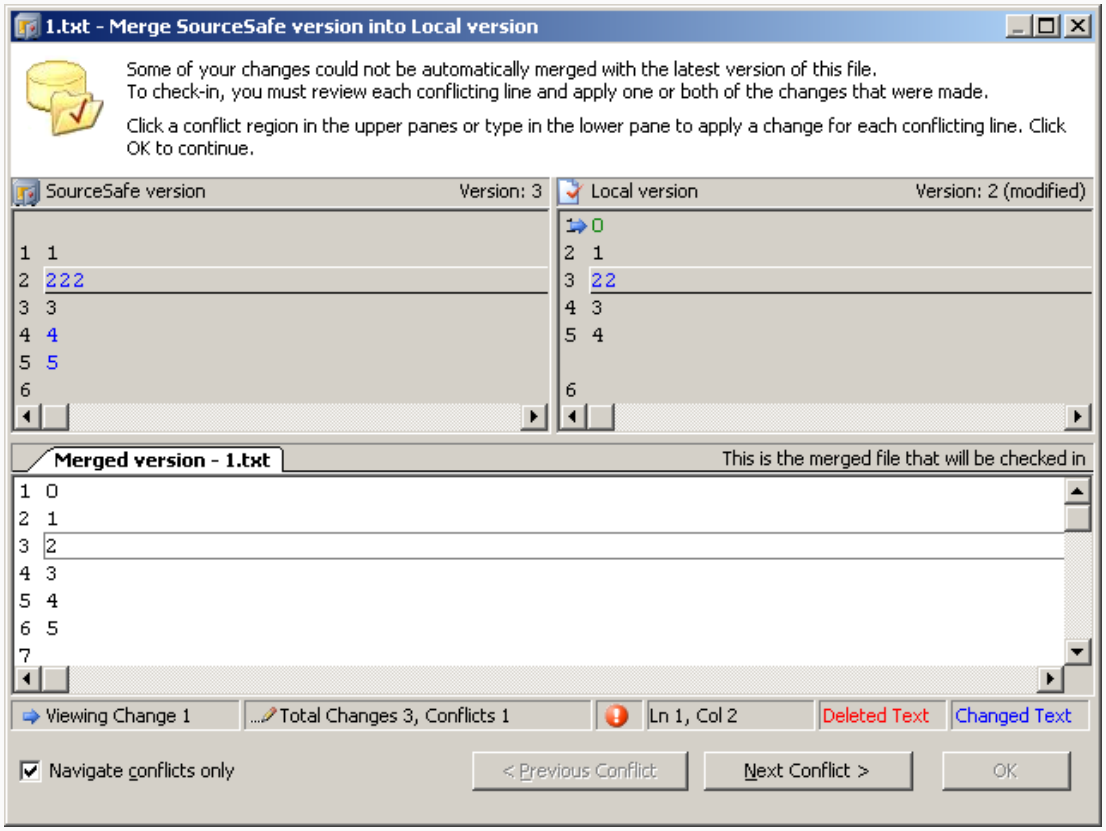
Added '0' before the first line

Changed 2nd line from '2' to '22'

In the another version, the following changes have been made:
Changed 2nd line from '2' to '222'
Add '5' after 4th line.

For the merge result, '0' is added before the first line, '5' is added after the 4th line. But since both of the versions have changed the second line, there is a conflict. When there is a conflict, it is not possible for SourceSafe to merge the files automatically and a visual merge is initiated to let us decide which changes will be kept.

The following is the screen shot for the above scenario:



(Screen shot: Visual merge when there are conflict(s))

As we can see that the '0' and '5' are added to the merge result. Since there is a conflict in the second line, the content from the base version, which is '2', is the default content in the merge result. We can choose '222' or '22' for the conflicting line by clicking the mouse on the content we want.

Four scenarios that merge may be performed

In SourceSafe, there are four circumstances when merge may be performed:

1. Merge branch

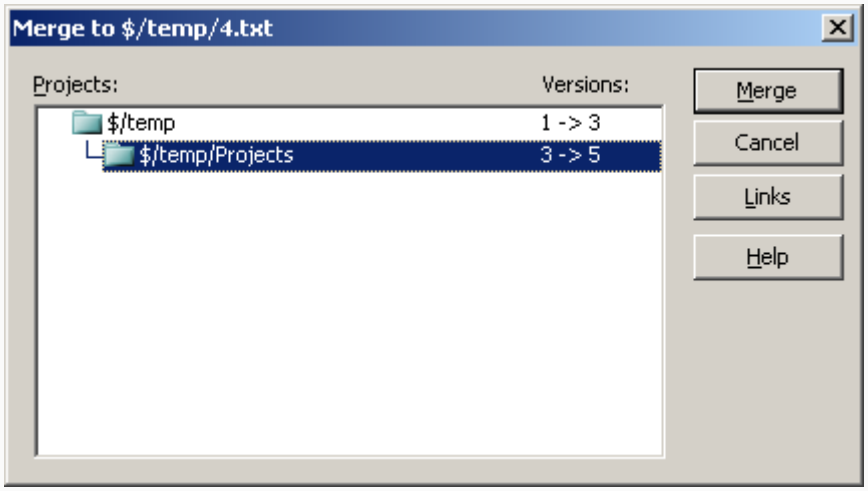
   In SourceSafe, you can branch a file to a different location and change the file content independently.

   Later on, you can merge the file content in the different locations back when needed.Under this situation:

- The base version is the version that the file is branched;
- The source version is the version that you want to merge the changes from;
- The target version is the version you want to merge the changes into.

To merge a branch:

- Select a file as the merge target (the merge result will be in this file)
- Choose menu **Versions** -> **Merge Branches** to bring up the Merge Branches dialog box, as shown:



(Screen shot: Merge branches)

- Choose a branch (this is the merge source) in the branch list;
- Click Merge.

2. Check in when multiple-checkout is enabled.

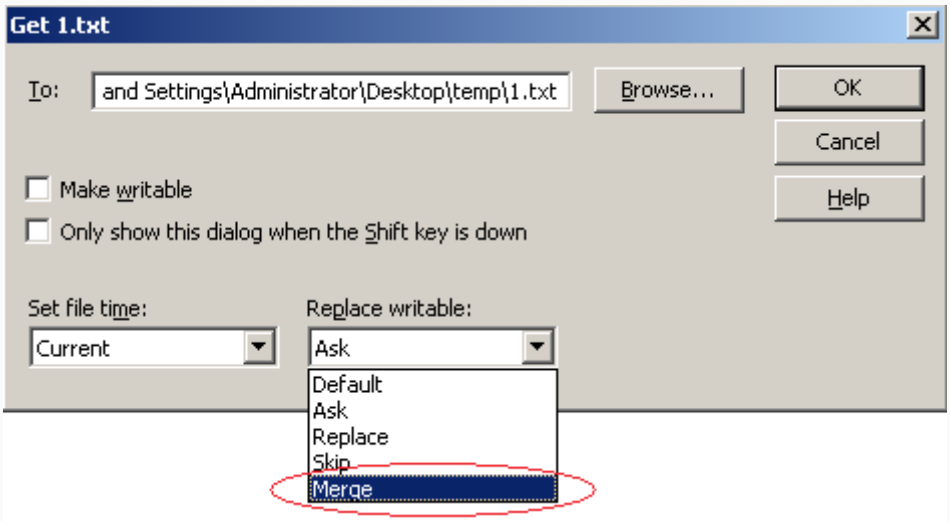
   When multiple-checkout is enabled, after we check out a file non-exclusively, someone else can continue to check out and check in the same file.When we check in the file, SourceSafe will check the version number of the file in the VSS database. If the version number in the database is the same as the one we

checked out (no one has checked in the file after we checked the file out), a normal check in operation is performed. If the version number in the database is bigger than the one we checked out the file (which means someone else checked in the file after we checked out the file), merge is performed before the normal check in operation.Under this situation:

- The base version is the version that we checked the file out;
- The source version is our local version;
- The target version is the current version in the VSS database.

3. Get a checked out file when multiple checkouts is enabled

   After we check out a file, when we get the latest version of the file, if someone checked in the file after we checked out the file and if we choose Merge for the Replace writable option in the check out dialog box, a merge operation will be performed.



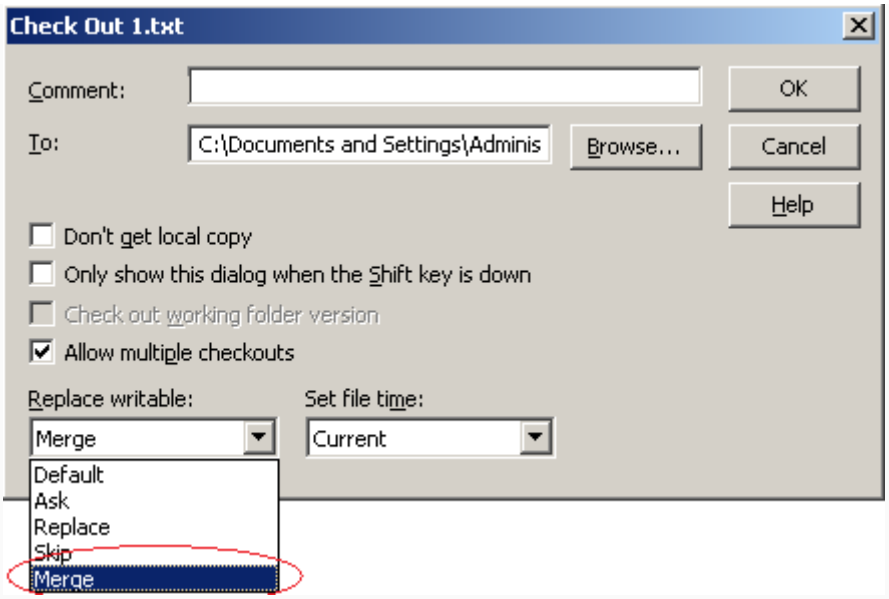
(Screen shot: Merge option in Get Latest Version)

Under this situation:

- The base version is the version that we checked the file out;
- The source version is the latest version in the VSS database;
- The target version is our local copy.

4. Check out a file

   After we get a file to local disk, if we change the file property to writable, when we check out the file, if we choose Merge for the Replace writable option in the Check out dialog box, a merge operation will be performed.



(Screen shot: Merge option in Check out)

Under this situation:

- The base version is the version that we did Get Latest Version;
- The source version is the latest version in the VSS database;
- The target version is our local copy.

Project Diff

Why Project Diff

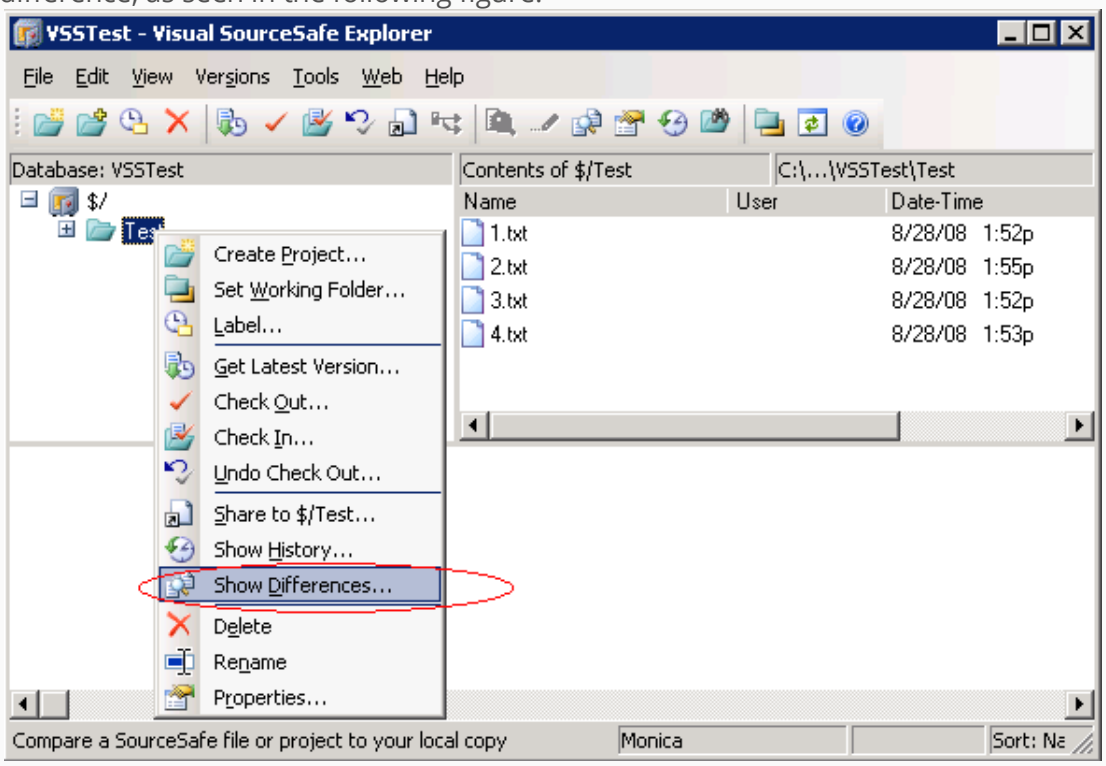
Project Difference compares two versions of a project and displays the comparison results. We can use it to see the differences between our local Windows folder and VSS project, two local Windows folders or two VSS projects.

This is a very useful feature that gives us the big picture of differences between two project trees. In the **Project Diff**, we can also launch **File Diff** by double clicking a file.

How to Launch Project Diff

To show project difference, we can select a project in Visual SourceSafe Explorer, and then click menu **Tools** -> **Show Differences**.

We can also right-click a project, and select **Show Differences** to show project difference, as seen in the following figure:



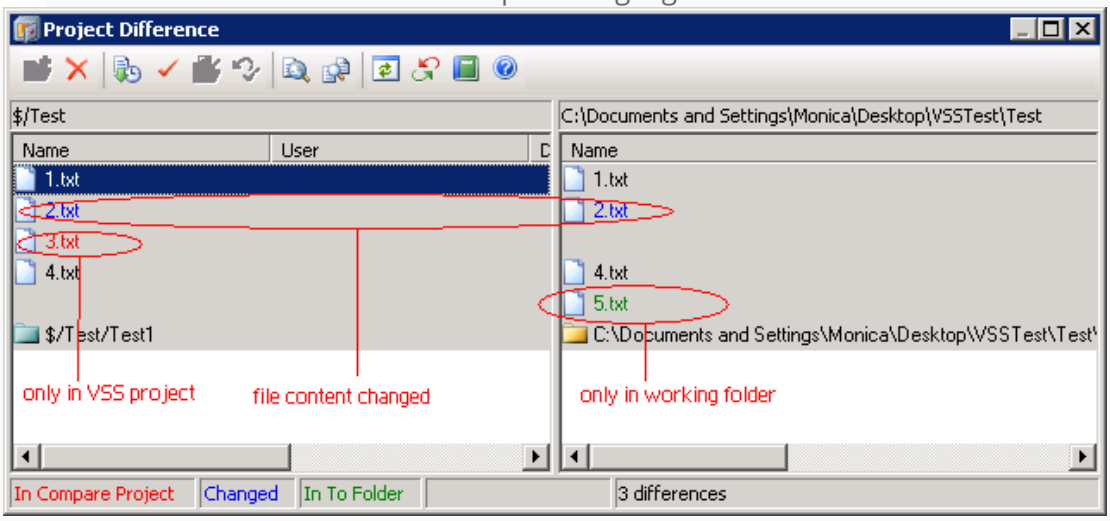
(Show Differences)

Project Diff Results

Results of the Project Diff are displayed as side-by-side panes listing files and subprojects alphabetically. By default, the left pane represents the project version in the VSS repository, while the right pane represents the working folder version.

There are four types of files that can be listed in the comparison results:

- Files that are only in the compare location highlighted with red.
- Files that are only in the to location highlighted with green.
- Files that are different in both places highlighted with blue.
- Files that are the same in both places highlighted with black.

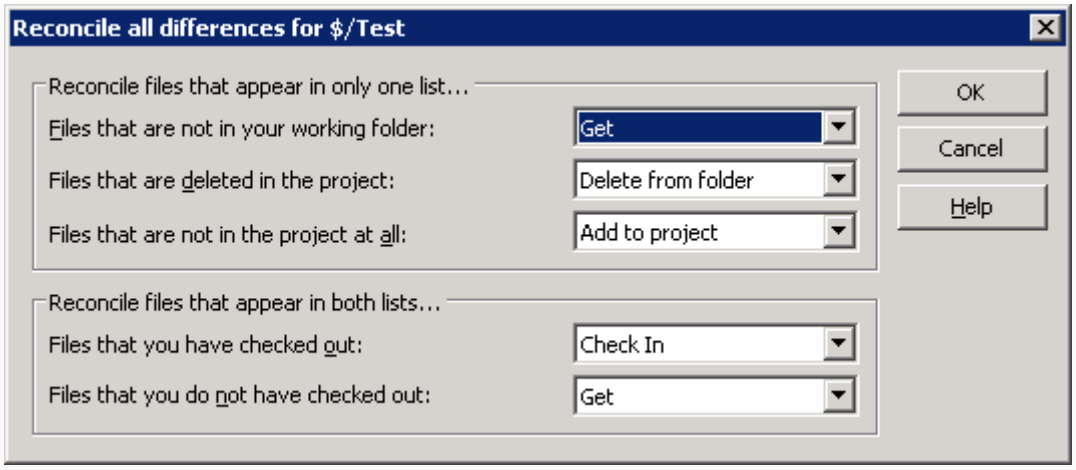


(Project Difference)

Basic Operations

In the Project Difference results window, we can do basic version control operations, like Add Files, Check In Files/Project, Check Out Files/Project, File/Project Difference, Get Latest Version and View, directly in this interface.

Reconcile All



(Reconcile All)

There is also a powerful feature called **Reconcile All**, which can synchronize our whole local folder and VSS database by just making several simple choices. **Reconcile All** is only enabled when we compare the differences between our local working folder and VSS project. It allows us to reconcile file differences between projects. There are 5 possibilities when reconciling differences:

- Files that are not in our working folder, but are in the VSS project.
- Files that are deleted in the VSS project, but are in our working folder.
- Files that are not in the VSS project at all, but are in our working folder.
- Files that are checked out to us, and differ between our working folder and the VSS project.
- Files that are not checked out to us, but differ between our working folder and the VSS project.

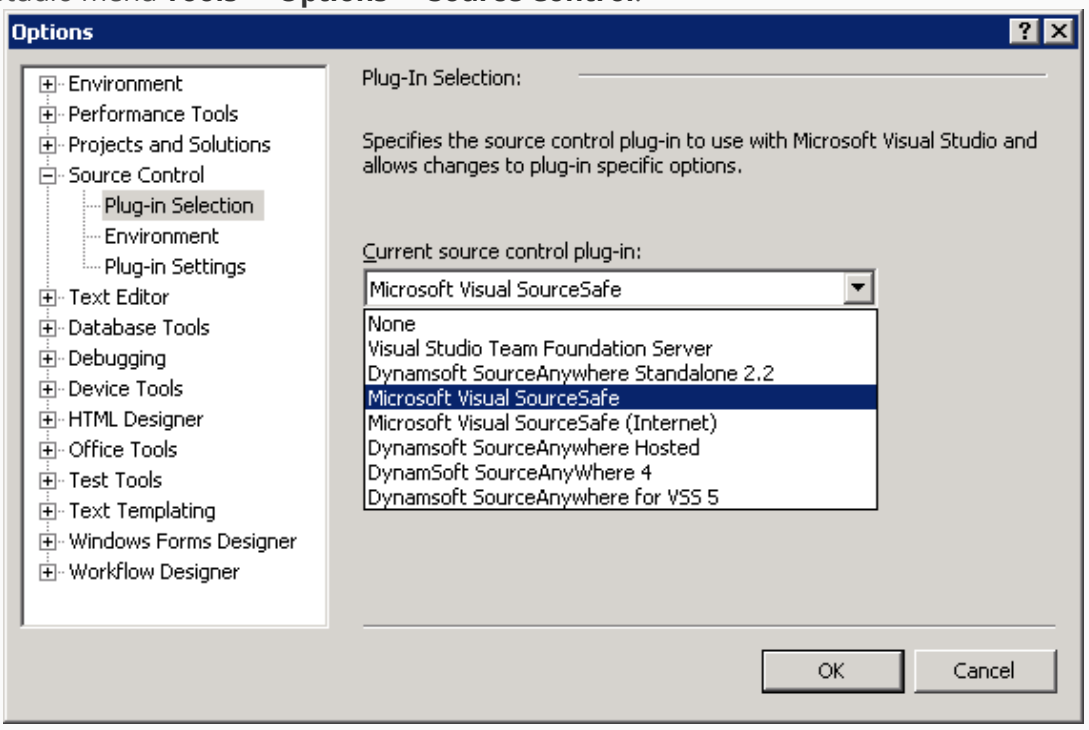
By using **Reconcile All**, we can get, add or check in/undo check out files to synchronize our working folder with the VSS project.

VSS in Visual Studio 2005 & 2008

Visual SourceSafe can be integrated into Visual Studio to source control solution files, project files and application configuration files. Using SourceSafe in Visual Studio allows us to source control the code directly in the Visual Studio IDE. When we try to modify a file source controlled by SourceSafe in Visual Studio, it will check out the file automatically (If you use the default setting **Check out automatically** for **Checked-in item behavior On Edit** in menu **Tools** -> **Options** -> **Source Control** -> **Environment**). We can then check in the modifications into the VSS Database.

Choosing SourceSafe as the SCC Provider in Visual Studio

To use SourceSafe to source control your files in Visual Studio, you need to select SourceSafe as your source control provider first. You can do that through the Visual Studio menu **Tools** -> **Options** -> **Source Control**.



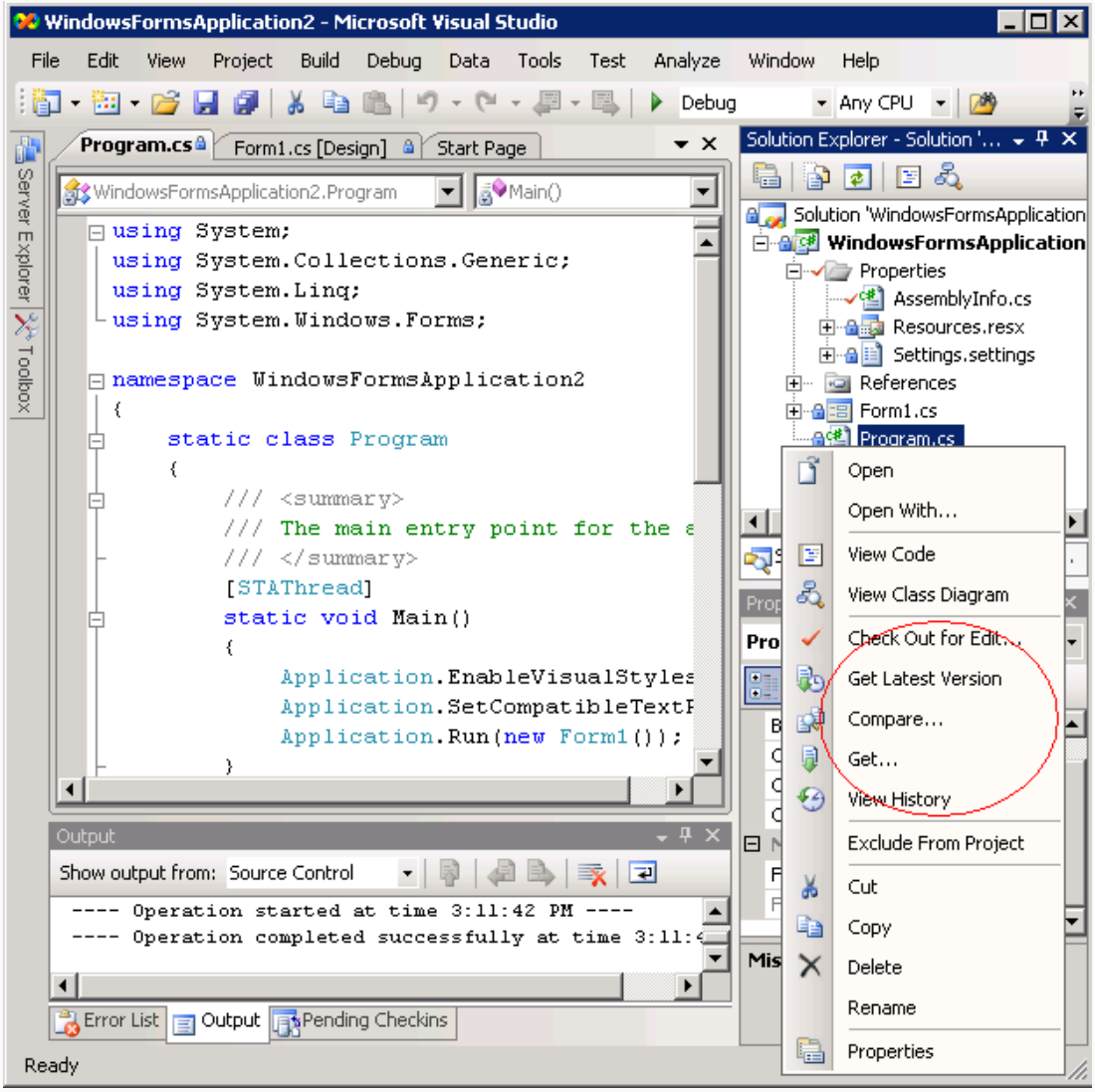
(Choose Source Control Provider)

Adding Solution into Source Control of SourceSafe

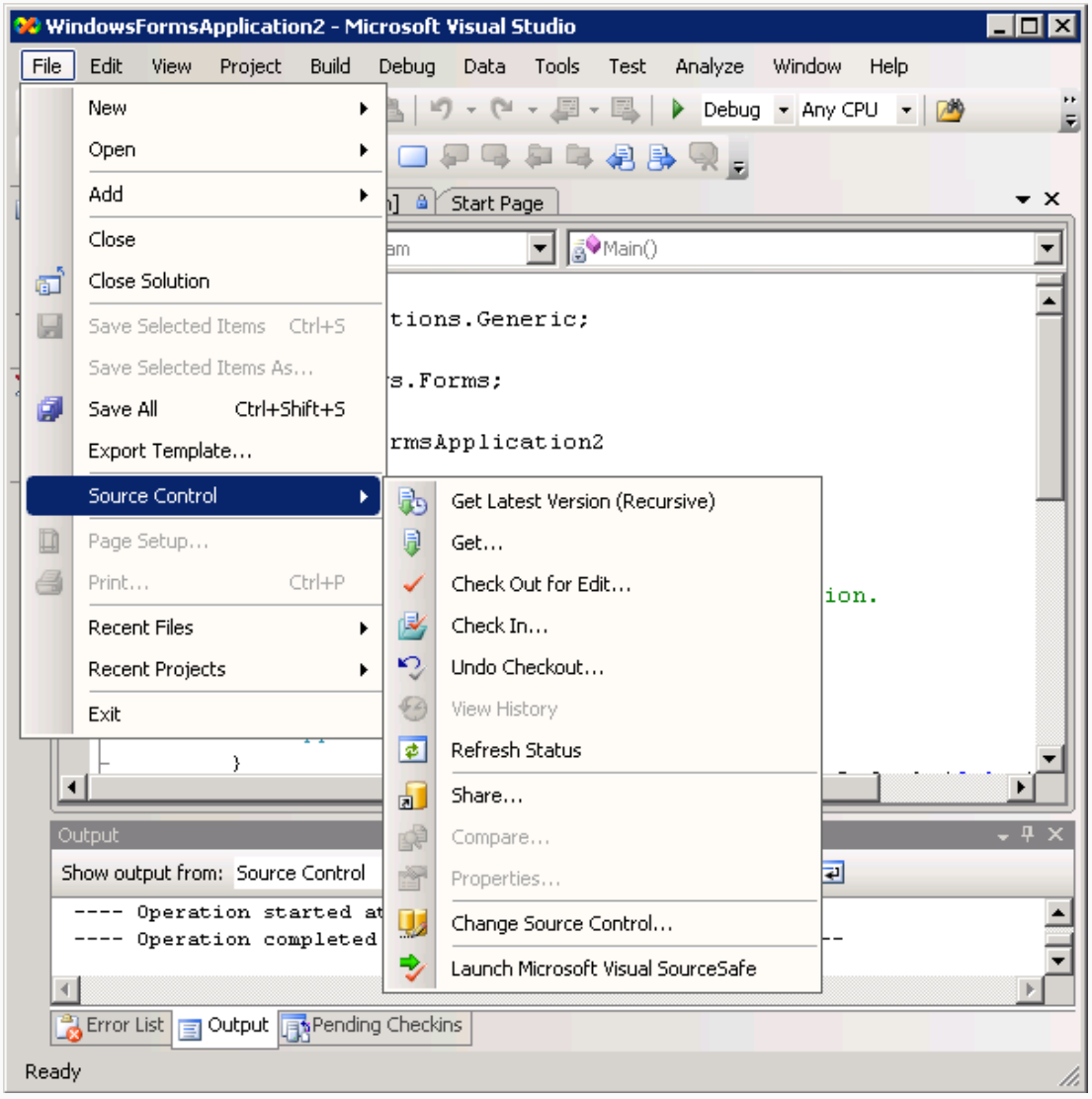
The files must be added into Source Control of SourceSafe before you can perform SourceSafe operations on them. To add a solution or project into VSS DB, right-click a solution/project file in Solution Explorer and click **Add Solution to Source Control**. In the following dialog boxes, enter your credentials and select a location for your project.

Performing SourceSafe Operations in Visual Studio

Now if you right-click an item in Solution Explorer or click menu **File** -> **Source Control**, you will see some additional SourceSafe functions, such as Get, Check Out/In, Undo Check Out, Compare etc.



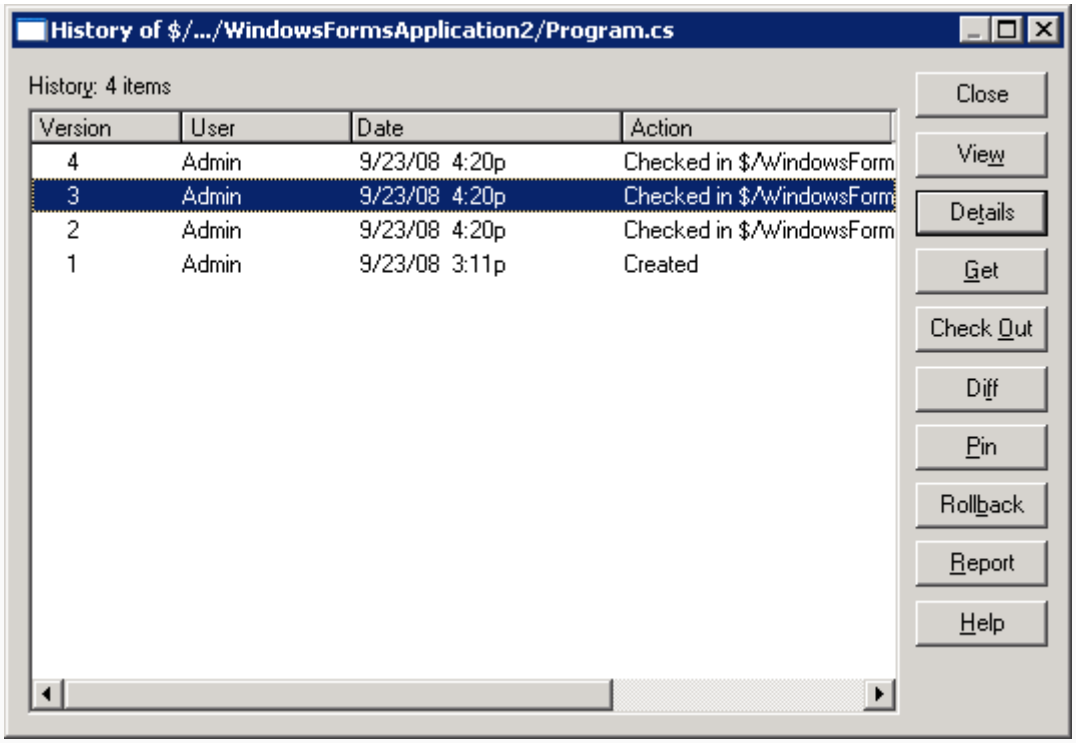
(Access SourceSafe functions through right mouse button)



(Access SourceSafe functions through File menu)

View History

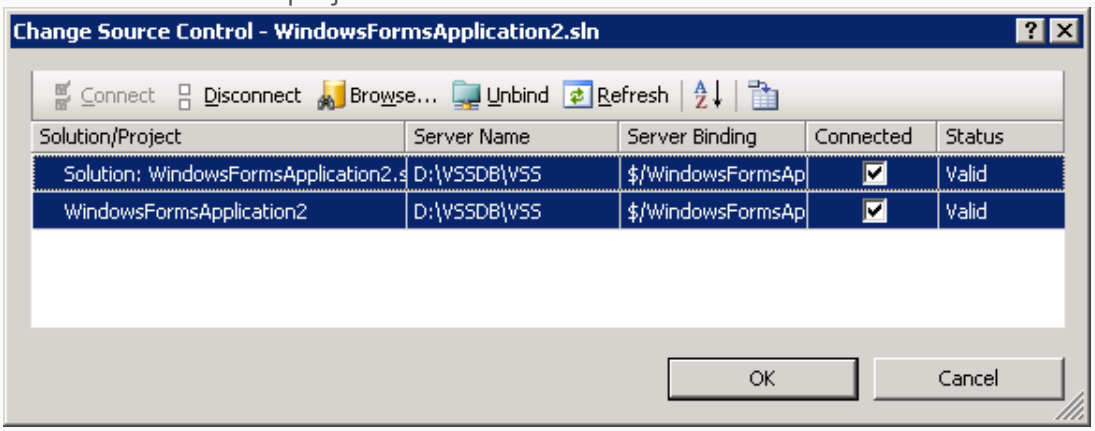
To view the history information of an item, you can right-click the item and select **View History**. You can perform operations like Check Out, Get, Pin in the History Explorer.



(History Explorer)

Changing Source Control Binding

If the project structure in SourceSafe is changed, or the project is moved to a new VSS DB, you will need to change the source control binding of the solution. Go to menu **File** ->**Source Control** -> **Change Source Control**, perform **Unbind** to removes the solution/project from source control and then perform **Bind** to associate the solution/project with the recent server folder or database.



(Change source control binding)

Pending Checkins Window

Visual Studio provides a Pending Checkins Window which allows you to see all of the checked out files in the current solution. You can select items to check-in in the Pending Checkins window. To open it, click menu **View** -> **Pending Checkins**.

Viewing Source Control Message

In cases when source control operations failed for some reason, you may be confused at the vague error message prompted by Visual Studio. In this situation, you may take a look at the Output Window of Visual Studio. If it is not already open, you can click menu **View** -> **Output** to open the window, choose to view the output message of **Source Control** and you can find more detailed information there.

Share

In version control, **Share** enables files to be shared among multiple projects. It creates share links among these projects, so that the item can be viewed in all projects. If an item is modified in one project, the changes will be reflected in other projects simultaneously.

To use the **Share** command, we must have the Check Out/Check In right in the project we are sharing from, and the Add/Rename/Delete right in the project to which we are sharing.

Why Share

Consider a scenario in which your team develops several projects at the same time, and they all use one header file named common.h. You can share common.h in all projects. That way, any change made to common.h in one project is immediately propagated to all other projects. The file in all projects will always keep the same content.

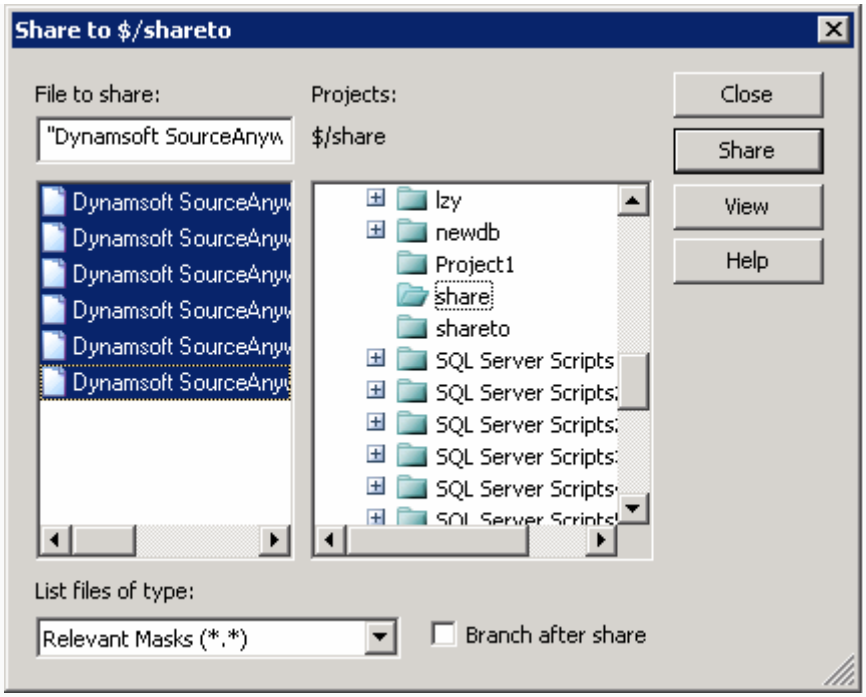
If we do not use the **Share** command, we may need to copy the shared files to other projects. This will cause a problem: We need to synchronize the file manually every time we make changes to the file.

Now we can understand **Share** better. Share command enables files to be viewed/ modified in multiple projects at the same time.

How to Share file(s)

To share an item, we can use **Share** command on the **Versions** menu.

1. In Visual SourceSafe Explorer, select a project to share the file(s) to.
2. On the **Versions** menu, click **Share to**.
3. In the **Share to** dialog, choose the project from which you want to share the file(s) in the **Projects** box, and then select files in the **File to share** box. You can check the **Branch after share** option if you need to branch the file(s).

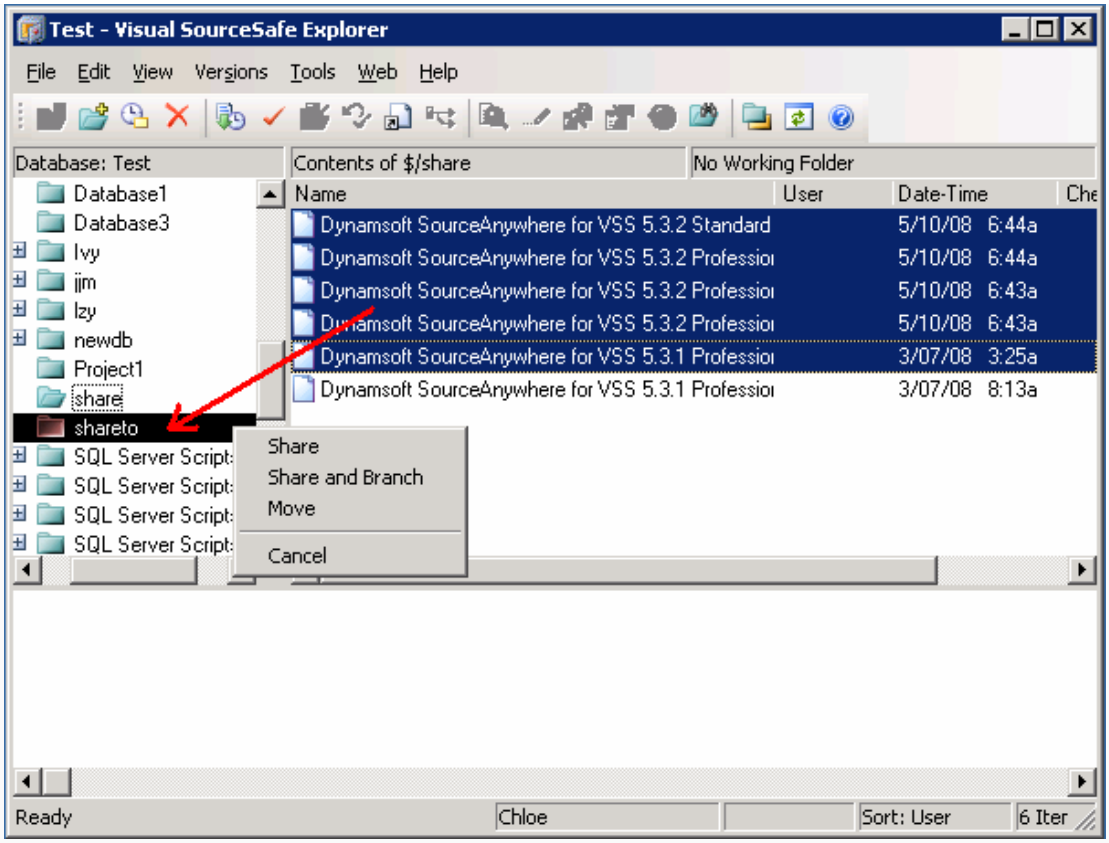


(Share using Versions menu)

4. Click **Share** to share the selected file.

However, there is an easier way to perform the **Share** command. We can Share the file(s) using drag-and-drop:

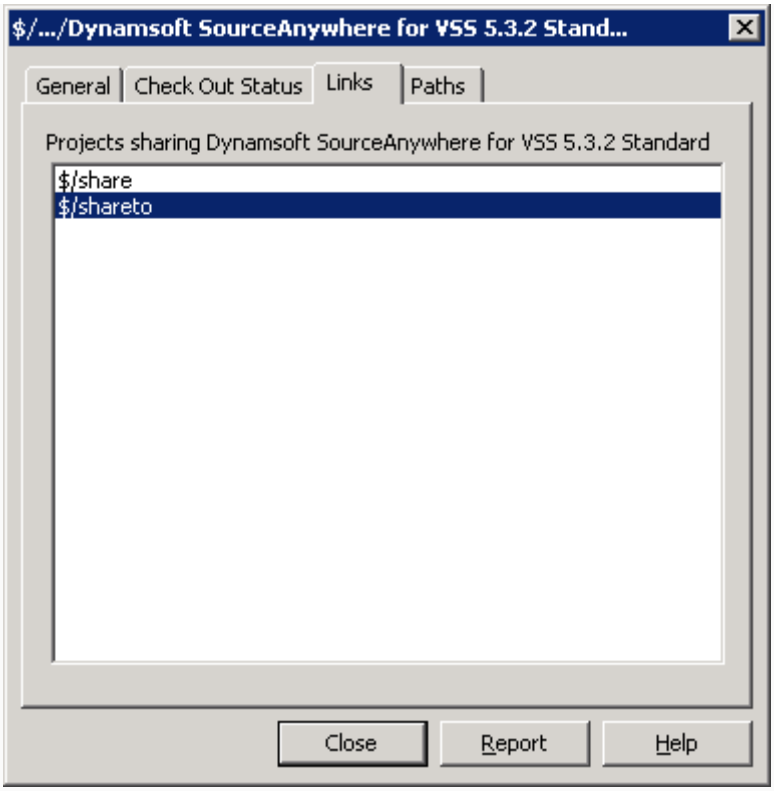
1. In Visual SourceSafe Explorer, right-click an item.
2. Drag the item over the receiving project and drop the item.
3. In the resulting menu, click **Share**.



(Share using drag and drop)

How to Find the Share Links

After we shared the file(s), we may want to know which projects the file(s) are shared in. In Visual SourceSafe explorer, we can right-click on a shared file and click **Properties** ->**Links** tab, thus all share links for the selected file will be displayed. A share link is a Visual SourceSafe path to a project that shares the file with other projects.



(Share links)

How Project Share is performed in VSS

According to the Visual SourceSafe help file, VSS supports file and recursive project sharing. This makes Share in VSS very convenient and efficient.

However, please be advised that VSS does not truly share projects. Instead of sharing project structure, VSS shares project through sharing all of its subsidiary files (recursively). So, if we change the project structure by adding or deleting a file, the change will not be reflected in other shared projects.

Share needs to be combined with **Branch** to play a greater role. We will go into details of**Branch** in my blog later. Please continue to pay attention.

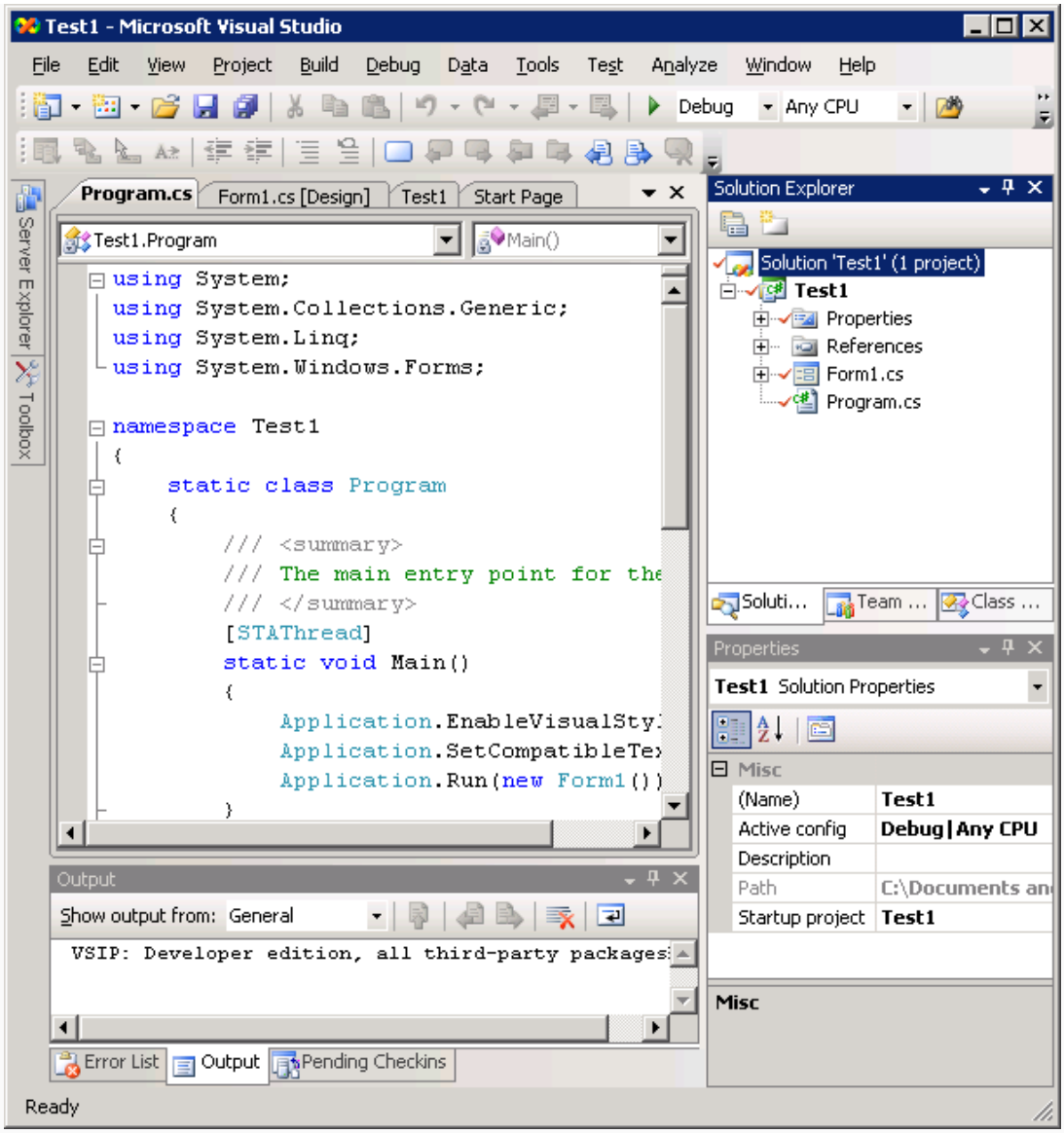
Switching Visual Studio projects from SourceSafe to other SCC providers

If you previously used SourceSafe to source control your Visual Studio projects and then find a better source control tool, you may need to switch the projects under SourceSafe to the new SCC (source code control) provider.
In the following steps, I use SourceAnywhere Standalone, the SQL-based SourceSafe replacement/alternative, as the target source control provider.

For Visual Studio 2005/2008

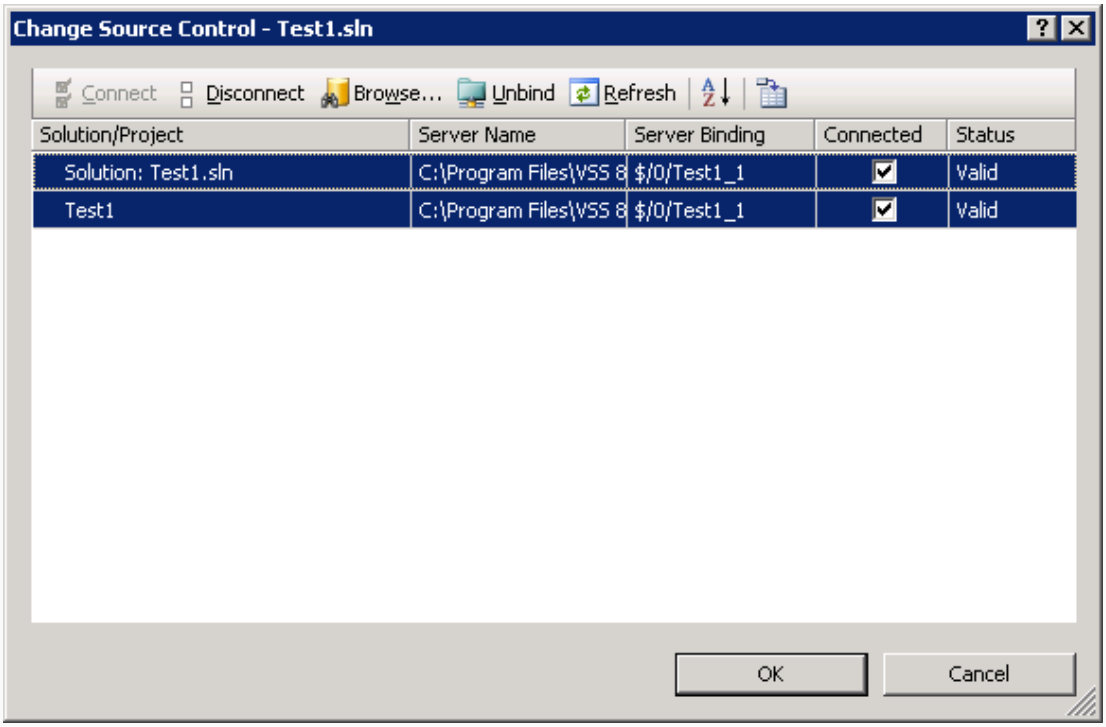
1. Open the solution in Visual Studio.

2. Make the solution and project files writable so that the unbind info can be written. You can check out the solution recursively. Or at a minimum you should check out the solution and project files. Or you can modify the file attribute to be writable manually.



(Check out the solution)

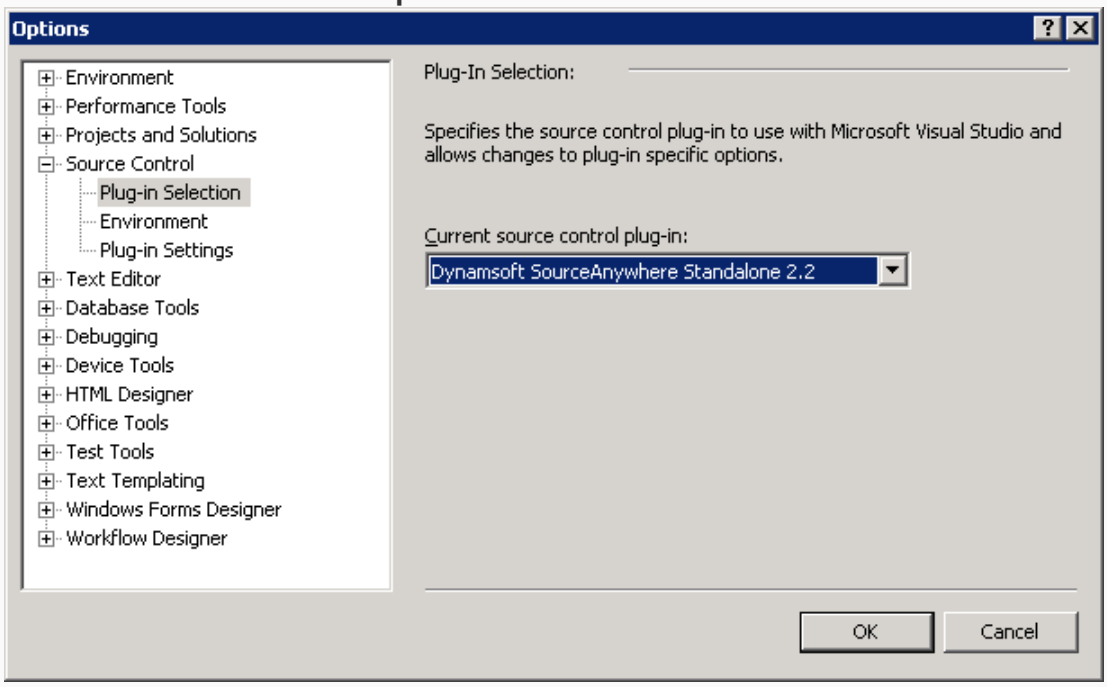
3. Unbind the solution and projects from SourceSafe through Visual Studio menu **File** ->**Source Control** -> **Change Source Control**.



(Unbind VS2008 solution/project)

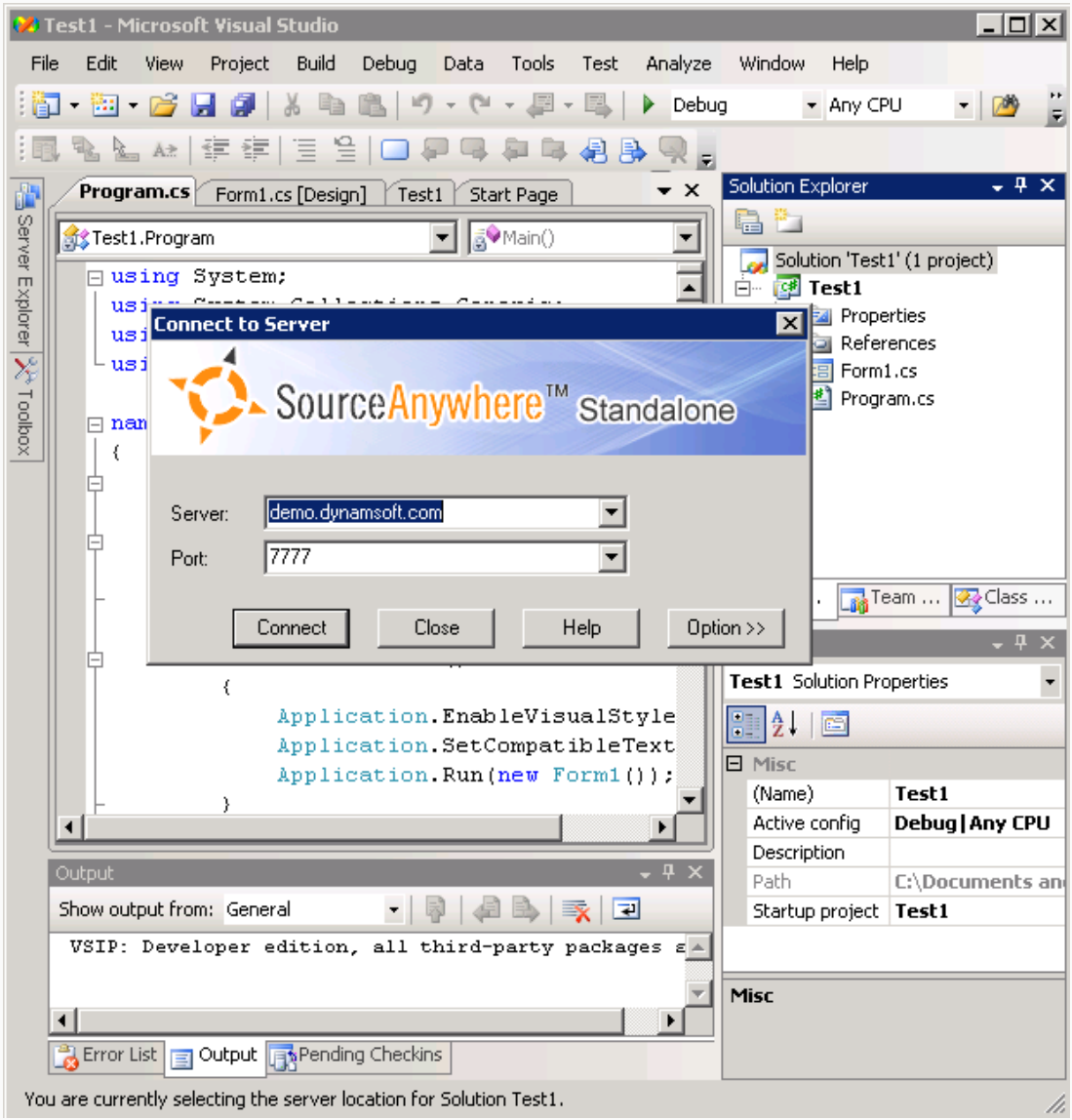
4. Close Visual Studio and then re-open it.

5. Select SourceAnywhere Standalone as the current source control plug-in through Visual Studio menu **Tools** -> **Options** -> **Source Control**.



(Select SourceAnywhere Standalone as the current source control plug-in)

6. Add the solution into the source control of SourceAnywhere Standalone. You can right-click the solution file in the Solution Explorer and click **Add Solution to Source Control**. Or you can find the option through Visual Studio menu **File** -> **Source Control**.



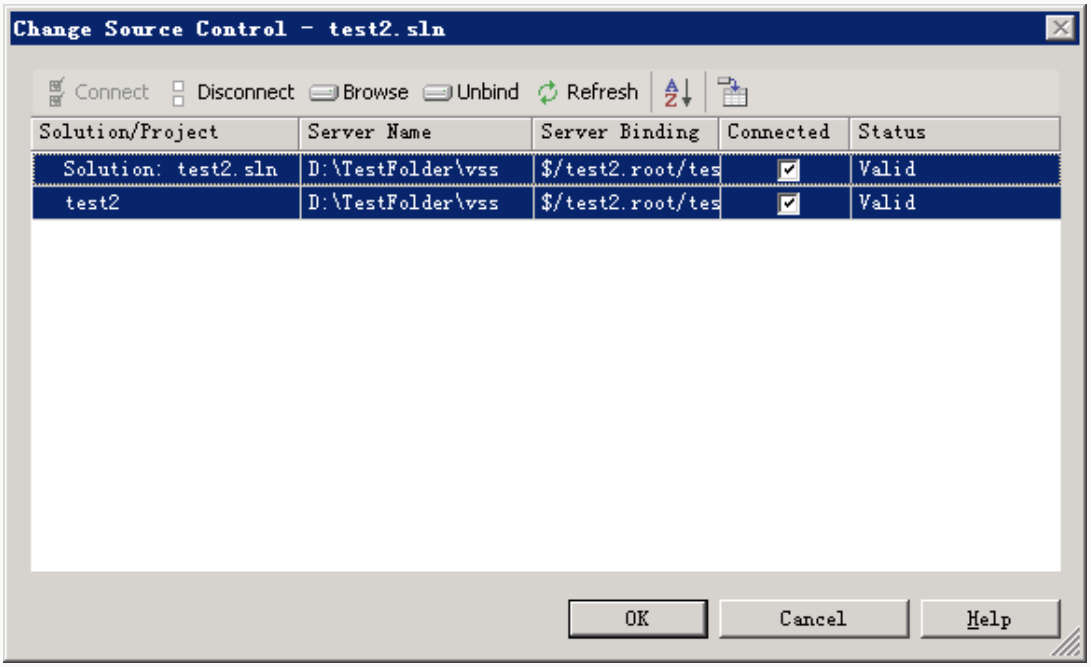
(Add solution into source control of SourceAnywhere Standalone)

For Visual Studio .NET 2003/Visual Studio 6.0

1. Open the solution in Visual Studio.

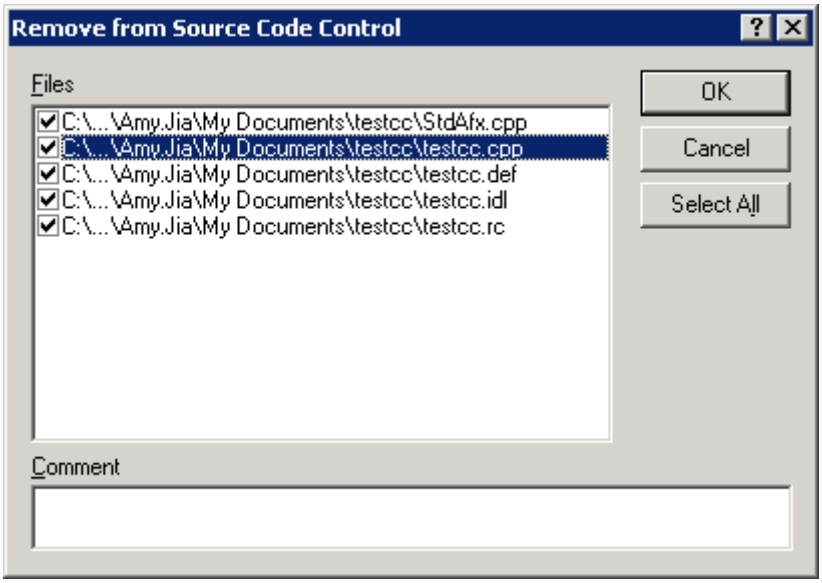
2. Unbind the solution and projects with SourceSafe.

- For Visual Studio .NET 2003, go to menu **File** -> **Source Control** -> **Change Source Control**.



(Unbind VS 2003 solution/project)

- For Visual Studio 6.0, click menu **Project** -> **Source Control** -> **Remove from Source Control**.

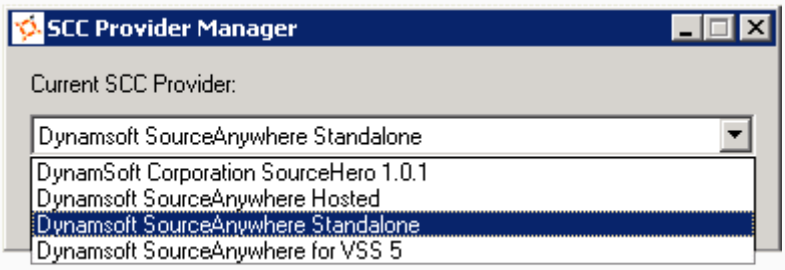


(Remove VS 6.0 files from source control)

3. Close Visual Studio.

4. Choose SourceAnywhere Standalone as the current SCC provider.

To do that, you can use the SCC Provider Manager utility which can be launched from the SourceAnywhere Standalone Client program group.



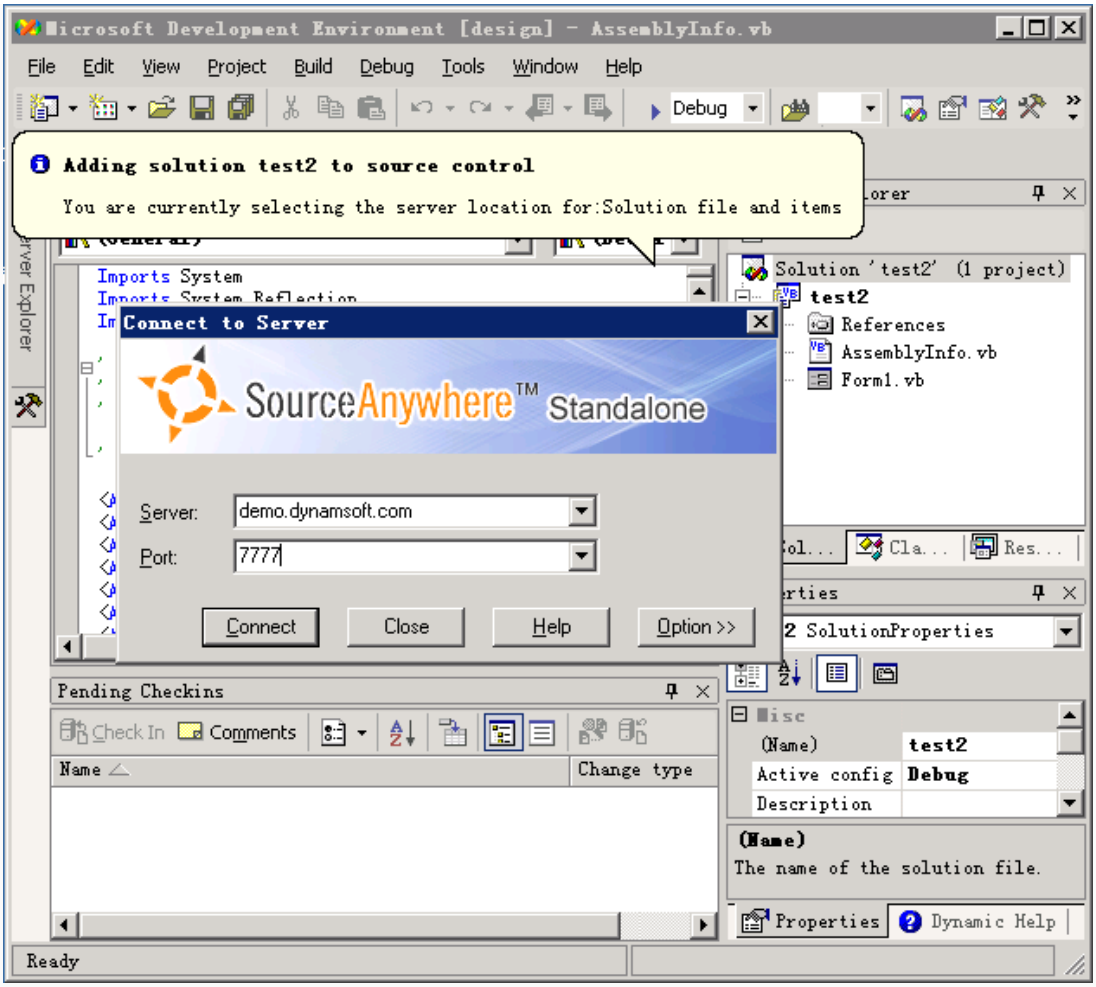
(Choose SCC Provider using SCC Provider Manager)

Or you can do it through registry: http://support.microsoft.com/kb/319318

5. Open the solution in Visual Studio.

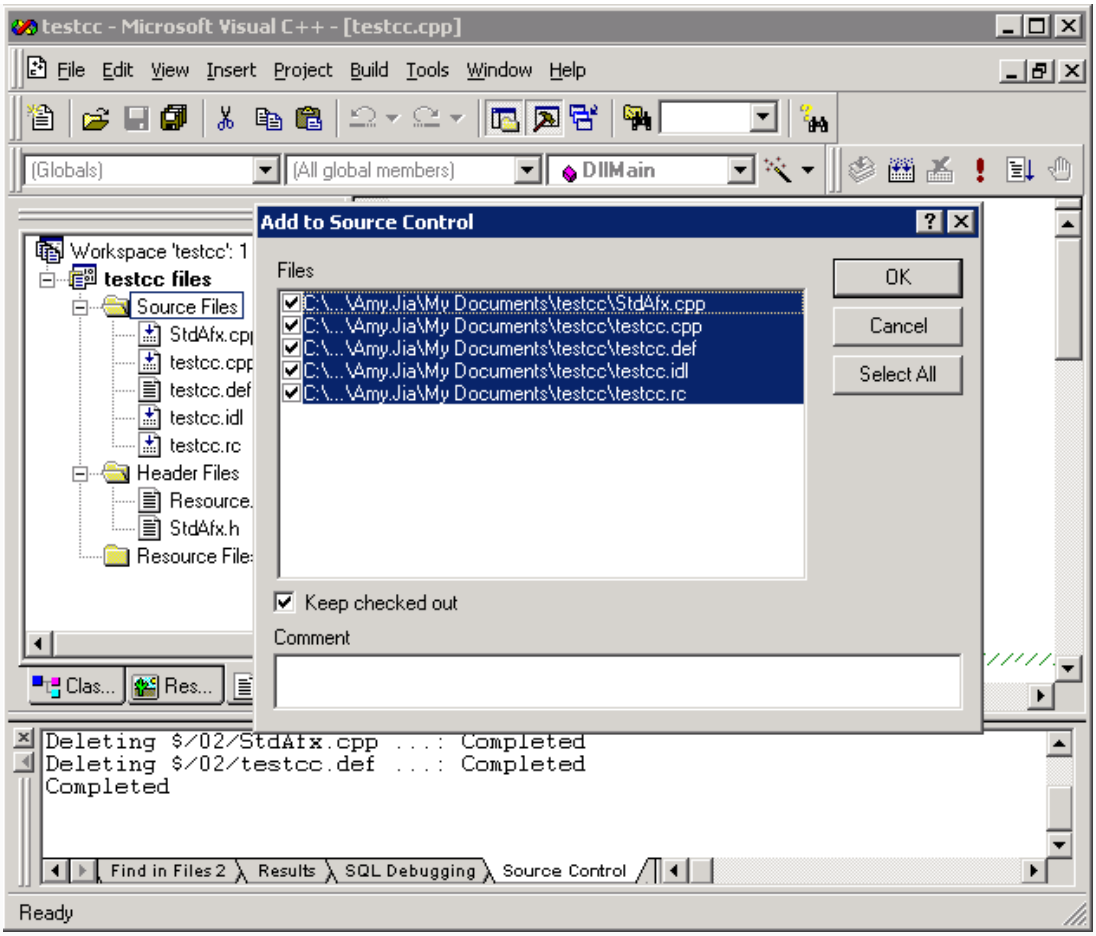
6. Add the solution into the source control of SourceAnywhere Standalone.

- For Visual Studio .NET 2003, right-click the solution file in the Solution Explorer and click **Add Solution to Source Control** or click menu **File** -> **Source Control** -> **Add Solution to Source Control**.



(Add VS 2003 solution into source control of SourceAnywhere Standalone)

- For Visual Studio 6.0, right-click the files and click **Add to Source Control** or click menu **Project** -> **Source Control** -> **Add to Source Control**.



(Add VS 6.0 files into source control of SourceAnywhere Standalone)

Now the solution is in the source control of SourceAnywhere Standalone. And next time you open the solution, Visual Studio will prompt the Login dialog box of SourceAnywhere Standalone to connect to the server.

Branch

As we talked about in our last post Share, share is only a small function in version control. It should be combined with Branch to play a greater role. This article will talk about the following topics: Introduction to Branch, Why Branch, How to Branch and How to Track Different Branches.

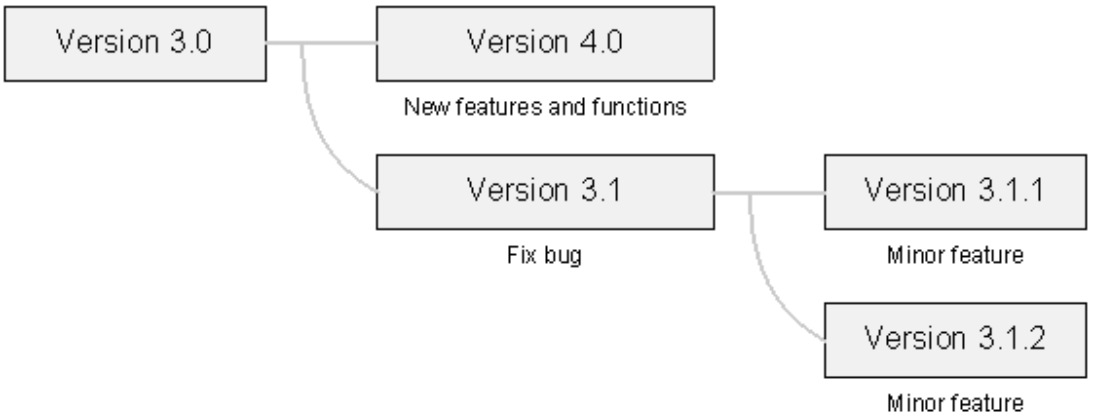
Introduction to Branch

After a file is shared in several projects, modifications to the file in one project are automatically propagated to all sharing projects. This feature has great advantages in some cases. However, sometimes we may want the file and its counterparts to be independent. That is when **Branch** comes to handy.
VSS allows us to branch a shared file with the **Branch** command. After branching, the file and its counterparts will be independent, and changes made to the file will not be reflected elsewhere. We can use **Merge Branches** to merge the branches back.

Why Branch

The Visual SourceSafe **Branch** command breaks the shared link between copies of the file, making the file versions independent and able to be part of separate projects. Here are two common scenarios in which we need **Branch**:

- Suppose that after we released a new Version 3.0, the R&D team starts working toward the next major release, the 4.0 version with new features and functions. We can implement it by branching all files into a new project $/Project4. If our users find bugs or need minor features in Version 3.0, we may need to work out a maintenance version, say, Version 3.1. In this case, we can create a new project named $/PATCH to represent the 3.1 version, and branch all the files from Version 3.0. Then some specific developers can fix bugs in the PATCH project while the other team members stay on the 4.0 version development.



(Branch scenario 1)

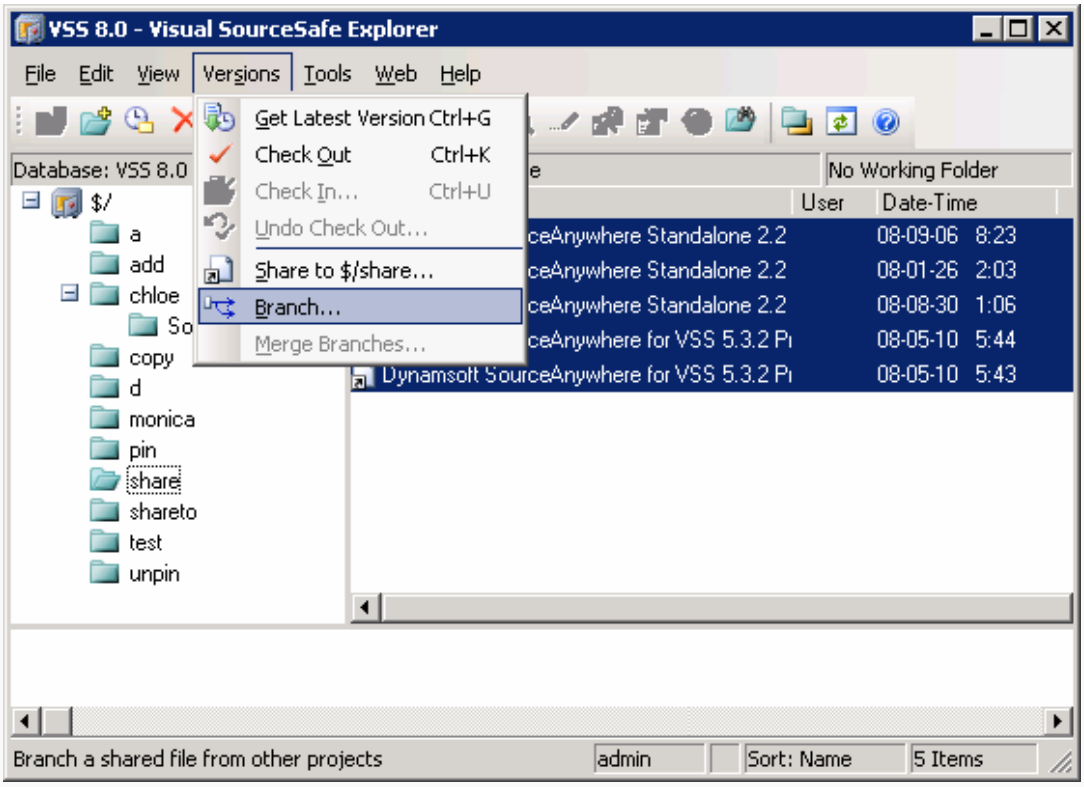
- Suppose that we are working on two projects which are virtually identical for two clients. We need to tailor program behavior in one or two files. First, we share the project $/CLIENT1 to a new project $/CLIENT2 recursively. Next, we select the few files we want to tailor in $/CLIENT2 and branch them. Now, when we modify these specific files in $/CLIENT2, the changes do not propagate to $/CLIENT1. All the other files, however, are still shared between the two projects. So, if we modify any of the shared files, the changes will be propagated.

How to Branch

How to Branch Shared Files

Visual SourceSafe allows us to branch a shared file using the **Branch** command:

1. Select the shared file(s) to branch in Visual SourceSafe Explorer.
2. On the **Versions** menu, click **Branch**.
3. On the **Branch** dialog box, type a comment in the **Comment** box if necessary. Then we can click **OK** to finish the operation.

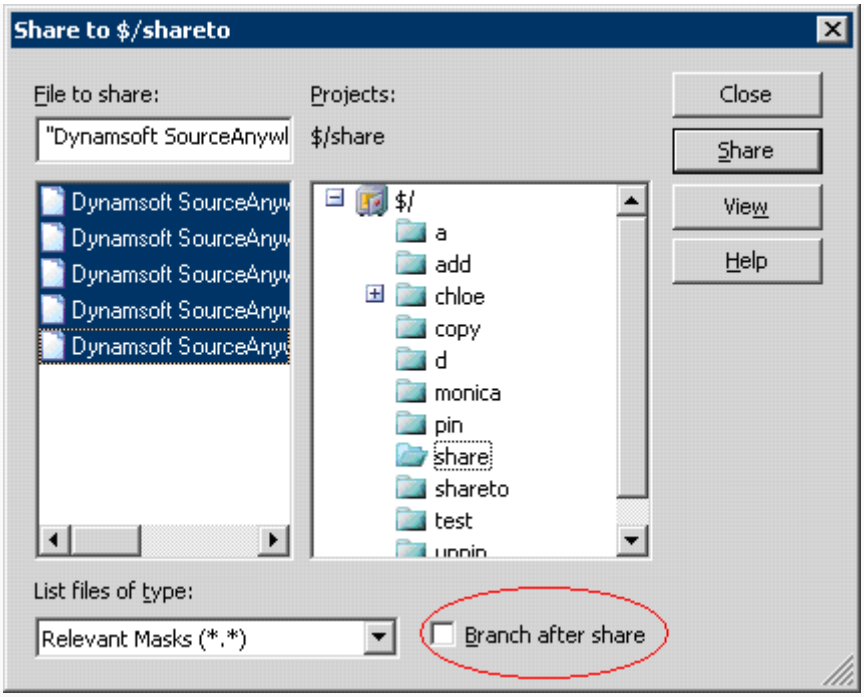


(Branch shared files using versions menu)

How to Share and Branch

The Share command supports branching of a file immediately after sharing it. To share and branch, we can use **Share** on the **Versions** menu:

1. In VSS Explorer, select the project to which the file(s) is branched.
2. On the **Versions** menu, click **Share to**.
3. In the **Share to** dialog box, use the **File to share** box to select the name of the file to share with the selected project.
4. Click **Share** with the **Branch after share** option checked.



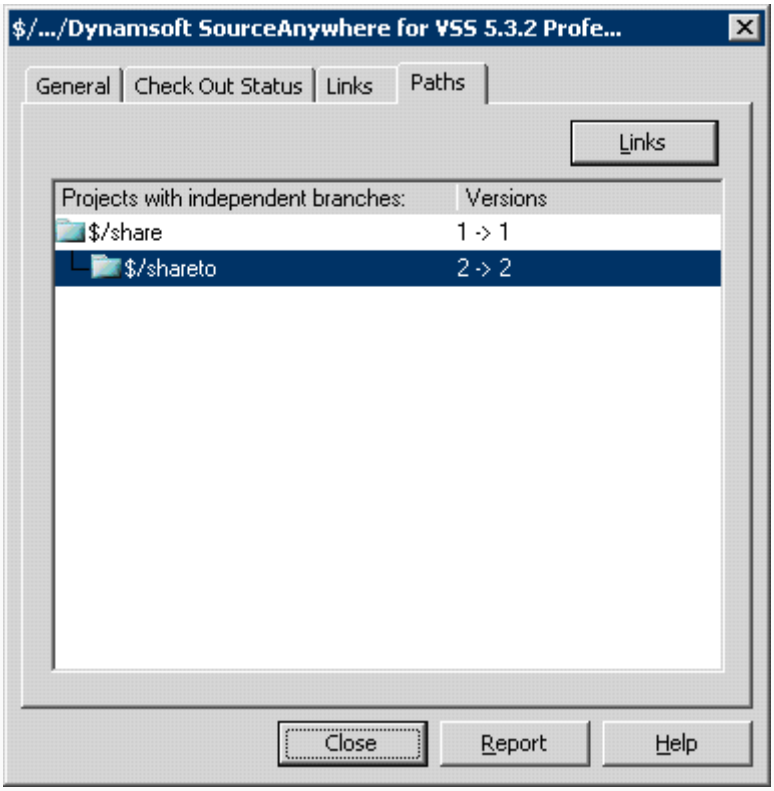
(Branch after share)

OR, we can use drag and drop:

1. In Visual SourceSafe Explorer, right-click a file.
2. Drag the item over the receiving project and drop the item.
3. In the resulting menu, click **Share and Branch**.

How to Track Different Branches

When we branch a file after sharing, the **Links** tab for file properties no longer shows a file relationship. However, we can track the different branches by right-clicking on the branched file, and then clicking **Properties** -> **Paths** tab.



(Branch links)

After the file branches are modified in different projects separately, we can bring them back together using the Merge Branches command. To learn more details, please refer to another article File Merge.

Cloak

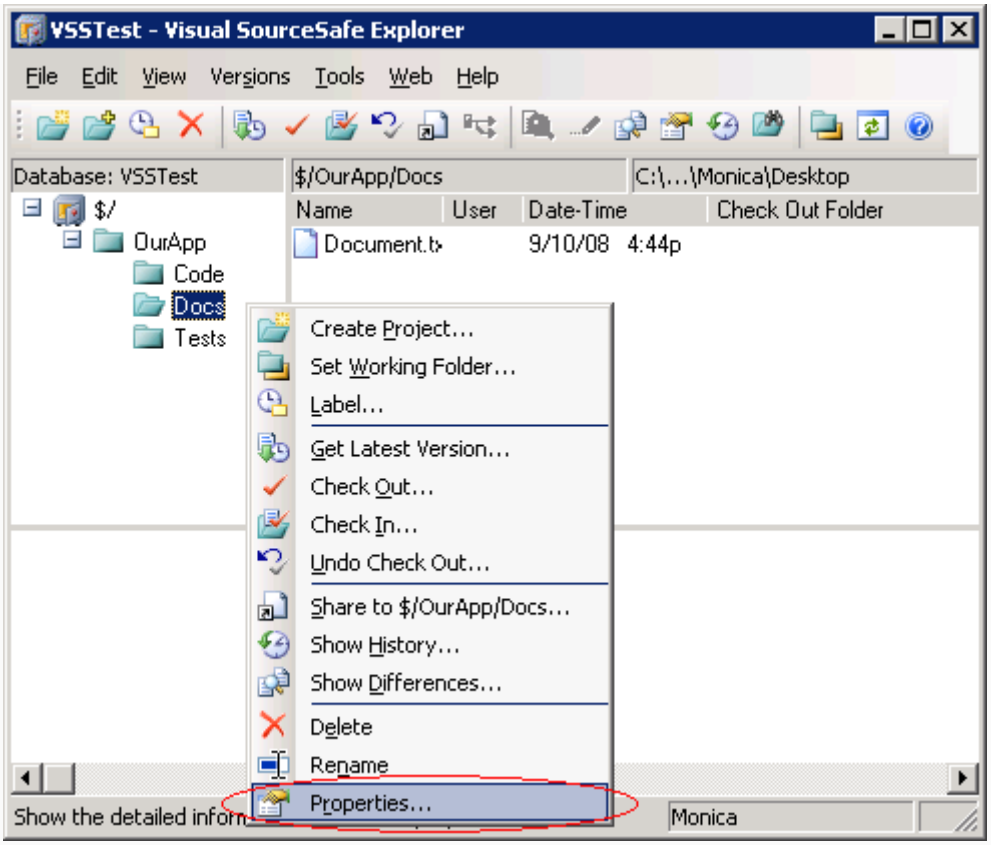
Cloak is a useful little function in version control. It allows us to specify the projects to be ignored during recursive operations such as **Get**, **Check In**, **Check Out**, and **Undo Check Out**.

Why Cloak

Some of the subprojects may never or rarely be used. However, when we do some recursive operations like **Get**, all the sub files and folders are operated.
For example, the help document needs to be updated whenever a new release comes out. It takes a long time to get the whole project of the help document, especially the image files. To improve efficiency, we can put all the image files in a subproject and cloak it. This way, the subproject cloaked will not be downloaded to local when getting the whole project.

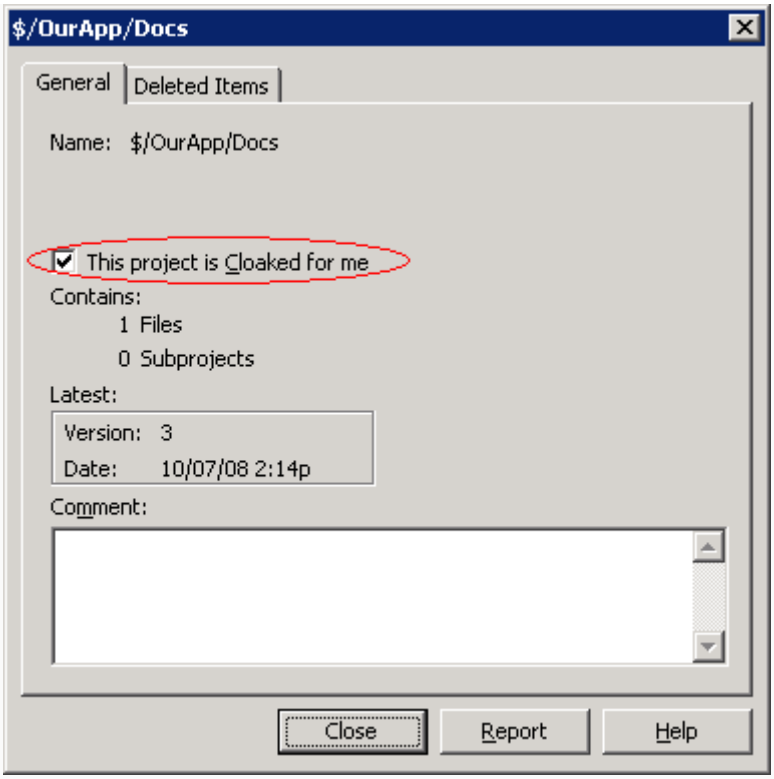
To Cloak a Visual SourceSafe Project

To cloak a project, we need to launch the Project Properties dialog first: right click the target project and select **Properties**.



(Cloak – Launch the project Properties dialog)

In the **General** tab of the project **Properties** dialog, we check the **This project is Cloaked for me** checkbox to cloak a project, as seen in the following figure:

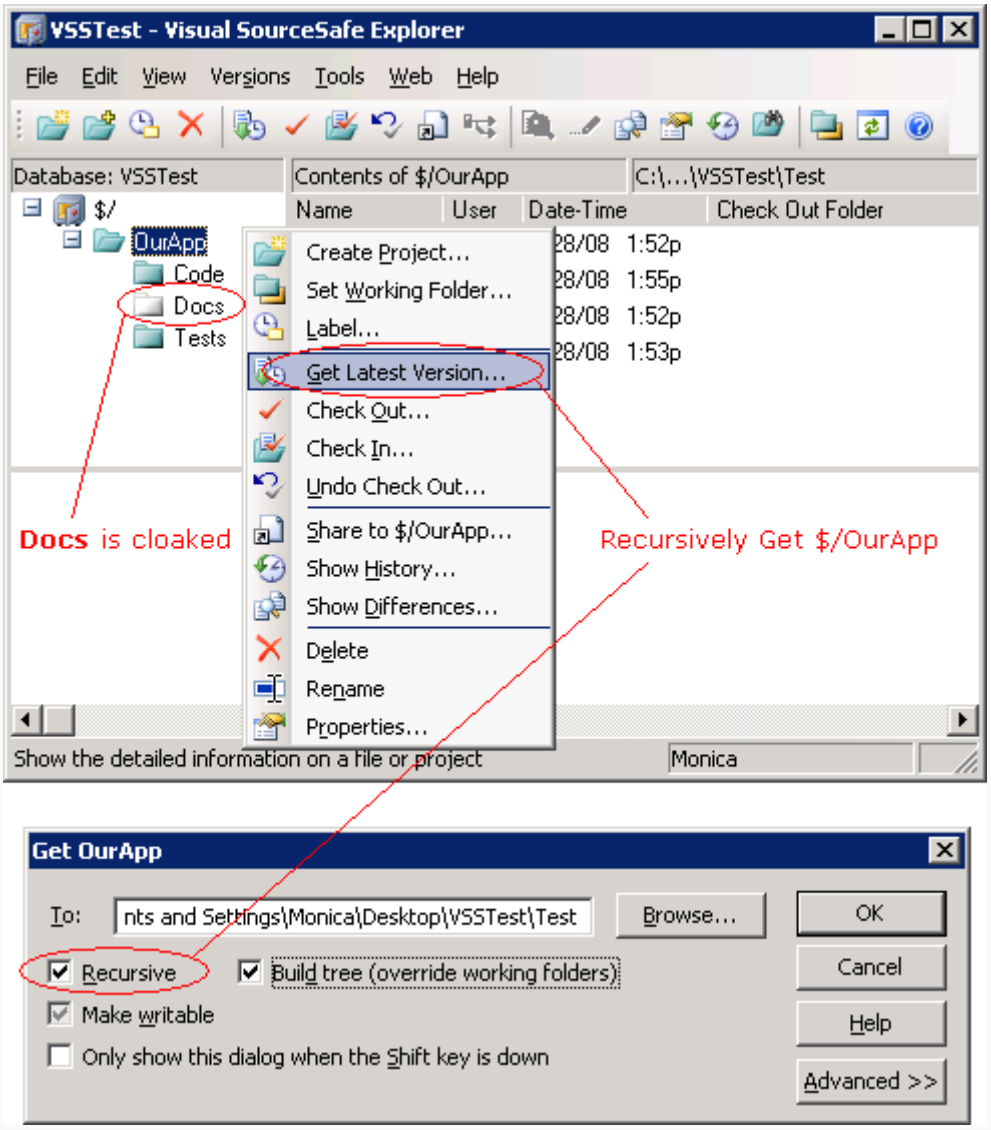


(Cloak a Visual SourceSafe project)

When a project is cloaked, the icon of the project is turned to grey. To de-cloak a project, just simply uncheck the **This project is Cloaked for me** checkbox in the project **Properties**dialog.

How Cloak Works

If a project is cloaked, **Get**, **Check In**, **Check Out**, **Undo Check Out**, and **Project Difference**operations do not apply to the project when we do these operations on the cloaked project's parent project. But when we do these operations on the cloaked project itself, these commands still work as normal.
For example: Suppose that we work on a project called $/OurApp. In this project there are subprojects called Code and Tests that we need and another subproject called Docs, which we almost never use. Now, we cloak the Docs project and recursively Get $/OurApp, the Docs project (and its subprojects) will be ignored.



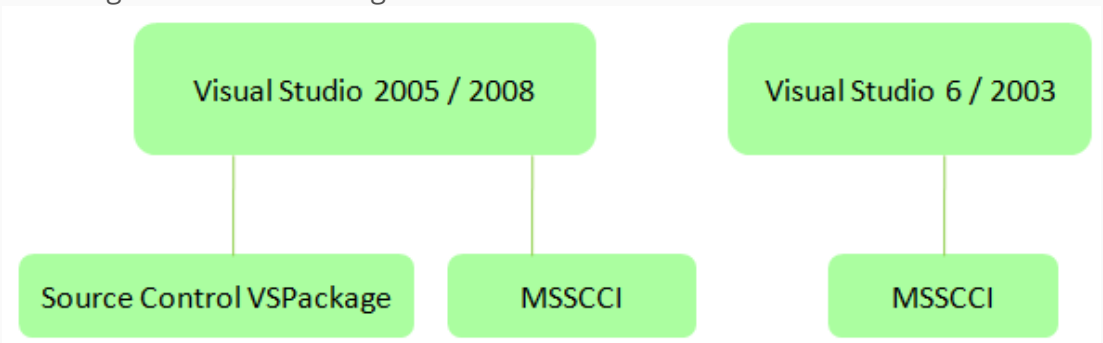
However, it does not mean that we cannot get a cloaked project. For example, if we want to get all the sub files and folders in the **$/OurApp/Docs** project which is cloaked, we can make the focus on the project (like clicking on the project) and then perform **Get**operation.

Once a project is cloaked, VSS will record the name and location of the project. If you delete or purge the project and create a new project with the same name in the same location, VSS will cloak the new project automatically. If a project is renamed or moved, the project will lose the cloak information. So besides unchecking the **This project is Cloaked for me** checkbox in the project **Properties** dialog, we can rename or move the project to another place to de-cloak a project. ^_^

Introduction to Microsoft Source Code Control Interface (MSSCCI)

MSSCCI stands for Microsoft Source Code Control Interface. The latest version of MSSCCI is 1.3.

In Visual Studio 6 and 2003/net, MSSCCI is the only option to integrate source control features into Visual Studio. In Visual Studio 2005 and 2008, we can use VSPackage or MSSCCI to integrate source control features into Visual Studio.



(Visual Studio and MSSCCI)

VSPackage is a new way to extend Visual Studio and is much more complex than MSSCCI. The newer version of SourceAnywhere will use VSPackage to provide better integration with Visual Studio.

MSSCCI has been around for more than a decade and is supported by many development products, like PowerBuilder, IBM Rational product line and Visual Studio.

MSSCCI was available only to VSIP (Visual Studio Industry Partner) members. To download the specification, you needed to sign a NDA (non-disclosure agreement) with Microsoft. Now, the specification is freely available at http://msdn.microsoft.com/en-us/library/bb166170(VS.80).aspx

Microsoft is opening up gradually and slowly.

Microsoft Source Code Control Interface (MSSCCI) Registry Entries

MSSCCI Registry Structure

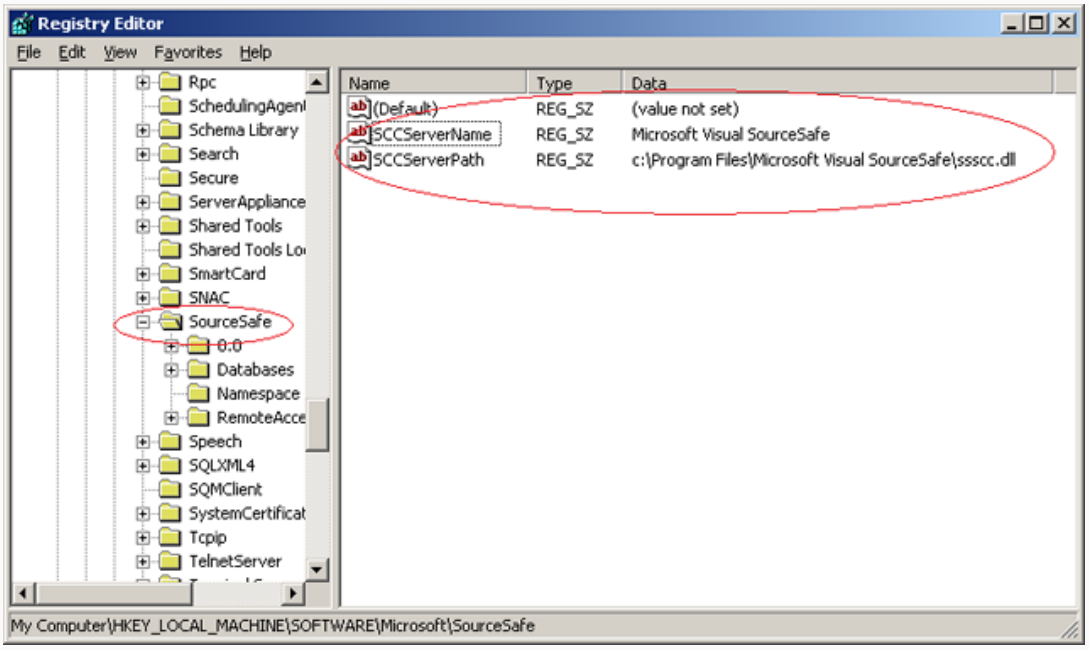
MSSCCI uses registry to organize multiple SCC providers.

There are 3 elements in the registry:

1. The provider specific registry entries

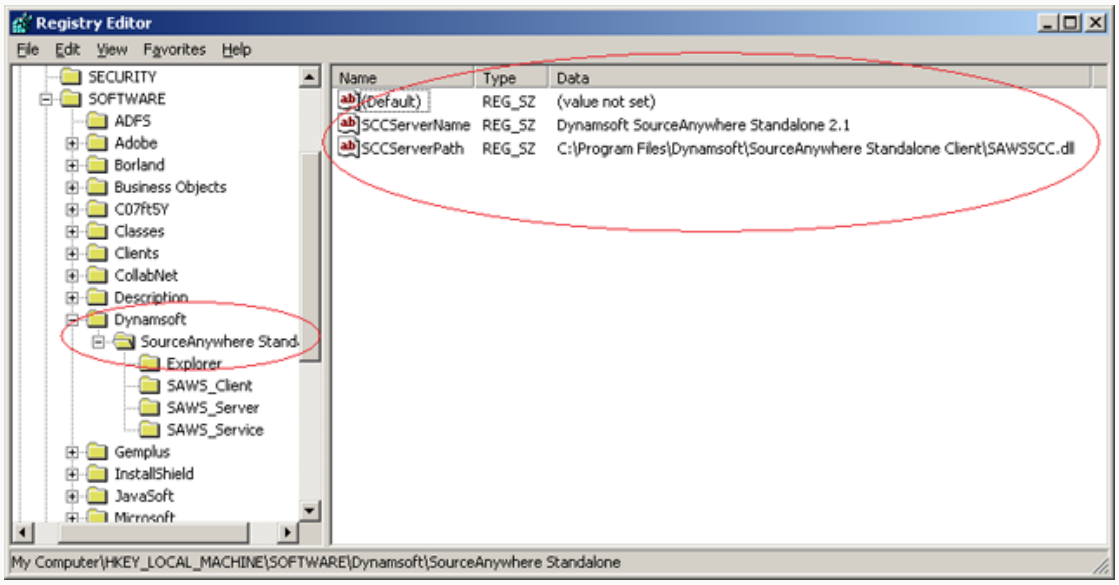
For every SCC provider, there is a registry entry: **HKEY_LOCAL_MACHINE\SOFTWARE\[company name]\[product name]**. For Microsoft SourceSafe, that is**HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\SourceSafe**. Under that entry, there are 4 sub entries: **SCCServerName**, **SCCServerPath**,**HideInVisualStudio** and **DisableSccManager**. **SCCServerName** is the name of the product.**SCCServerPath** is the full path to the SCC dll. **HideInVisualStudio** and **DisableSccManager**are not so commonly used.

The following is the screen shot of the SourceSafe/VSS MSSCCI registry:



(SourceSafe/VSS MSSCCI registry)

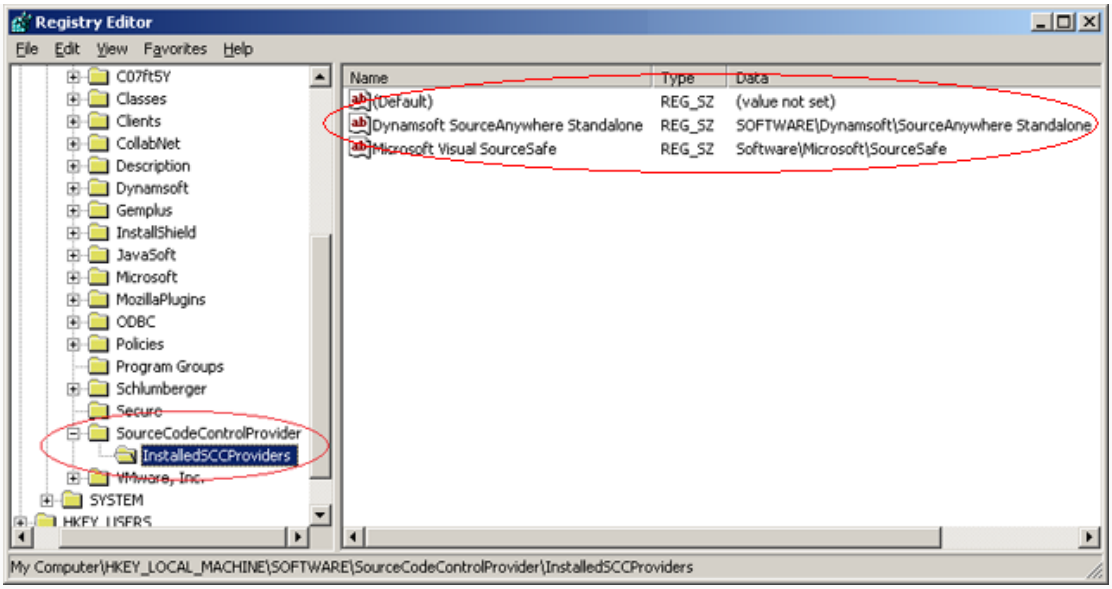
Other MSSCCI provider has similar values. The following is the screen shot of the SourceAnywhere MSSCCI registry:



(SourceAnywhere MSSCCI registry)

2. The central place to list all the providers

All of the SCC providers are listed under the**HKEY_LOCAL_MACHINE\SOFTWARE\SourceCodeControlProvider\InstalledSCCProviders**entry. Under the **InstalledSCCProviders** entry, there are list of **[display name] = [the path to the provider specific registry entry]**. The following is the screen shot:

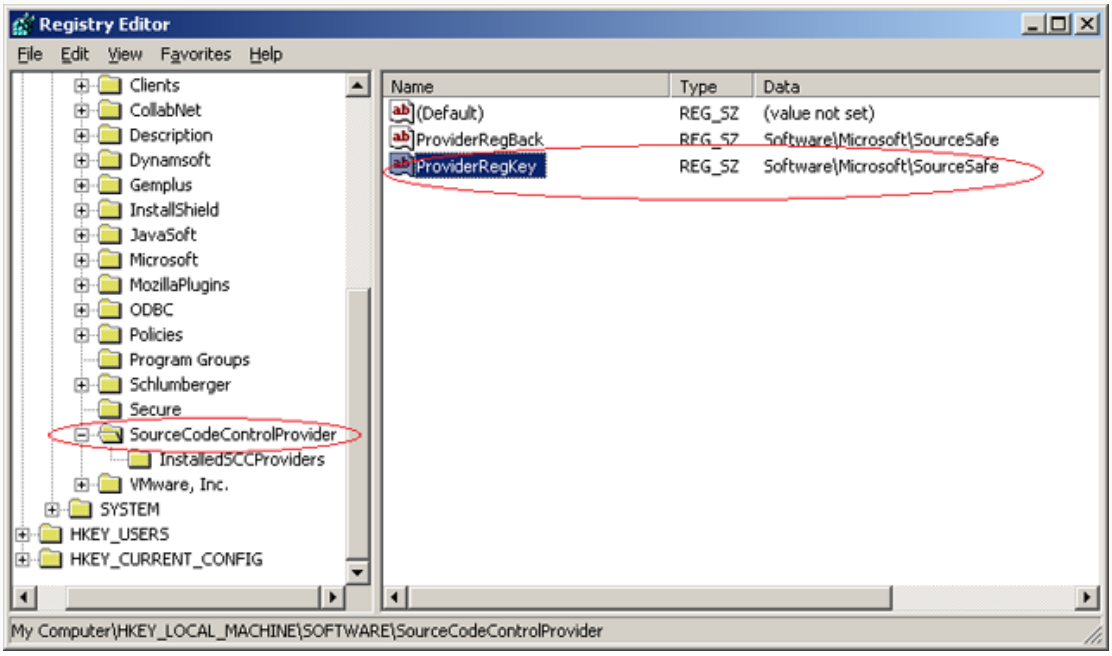


(The installed SCC providers in the system)

3. The entry to specify the current provider

Finally, there must be an entry to specify which provider is the current provider in the system. The pattern is**HKEY_LOCAL_MACHINE\SOFTWARE\SourceCodeControlProvider\ProviderRegKey = [the path to the provider specific registry entry]**.

The following is the screen shot of my machine that specifies that SourceSafe is the current provider:



How to Edit/View the Registry

We can use regedit.exe to edit or view the registry.

Improvement in Visual Studio 2005 and 2008

From the above, we know that there is only one**HKEY_LOCAL_MACHINE\SOFTWARE\SourceCodeControlProvider\ProviderRegKey** entry, which means that we can only have one default SCC provider in the system. If we have two projects that use two different MSSCCI providers, we have to switch the current provider when we open a different project.
Since Visual Studio 2005, Microsoft made improvement by writing the SCC provider info in the solution file. This way, the SCC provider information is solution specific. Different solutions can use different source control providers without switching provider info in the registry.
For how to choose the SCC provider in Visual Studio 2005 and 2008, please refer to the article Switching Visual Studio projects from SourceSafe to other SCC providers. The url is:
http://www.kevingao.net/sourcesafe/switching-visual-studio-projects-from-sourcesafe-to-other-scc-providers.html

Label

Label Introduction

Visual SourceSafe allows you to define a label for a file or project version. A label is a short and concise description given for purposes of identification, for example, “3.0Beta” or “Release”. **Label** is one way in Visual SourceSafe that allows us to keep track of the file and project versions. By using **Label**, we can smoothly manage version releases/builds and easily find the previous versions, especially the project versions. You must have the Add/Rename/Delete permission to use the **Label** command.

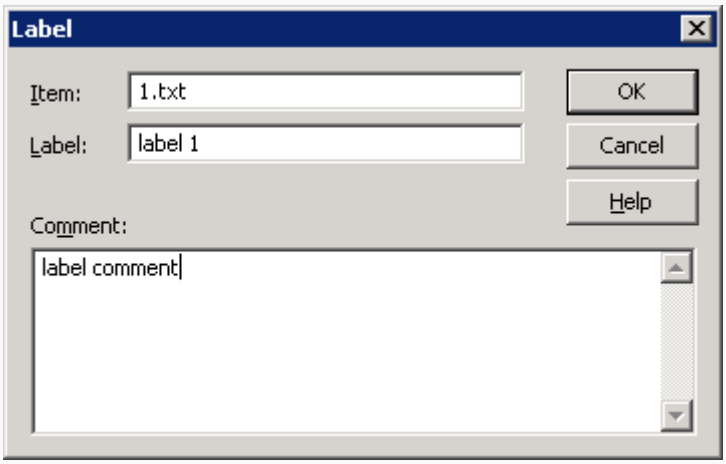
Label is a very useful and important feature in any version control system. If we just released version 3.1.2 yesterday, we may be able to find the release version of 3.1.2. But how about 1 month or even 1 year later, can we still find that particular version? Without a label, it is not likely.

How to Label a File/Project

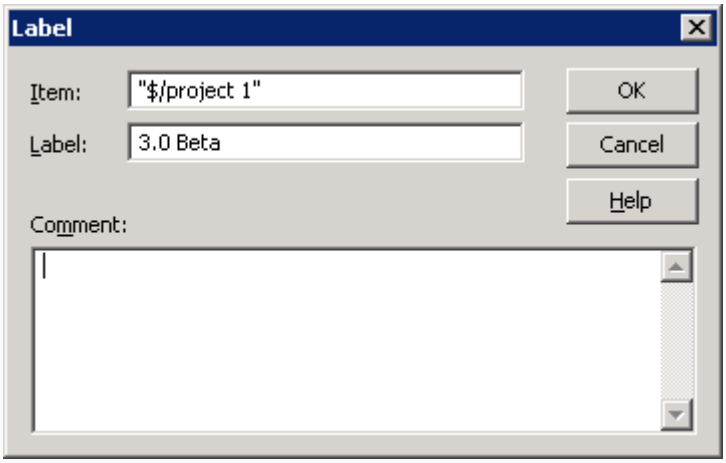
We can label the current version and a historical version of a file or project.

To label the current version of a file or project:

1. Select the file or project to label in Visual SourceSafe Explorer.
2. Click **Label** on the menu **File**.
3. Input a label text in the **Label** edit box in the **Label** dialog box, as seen in the following screen shot:



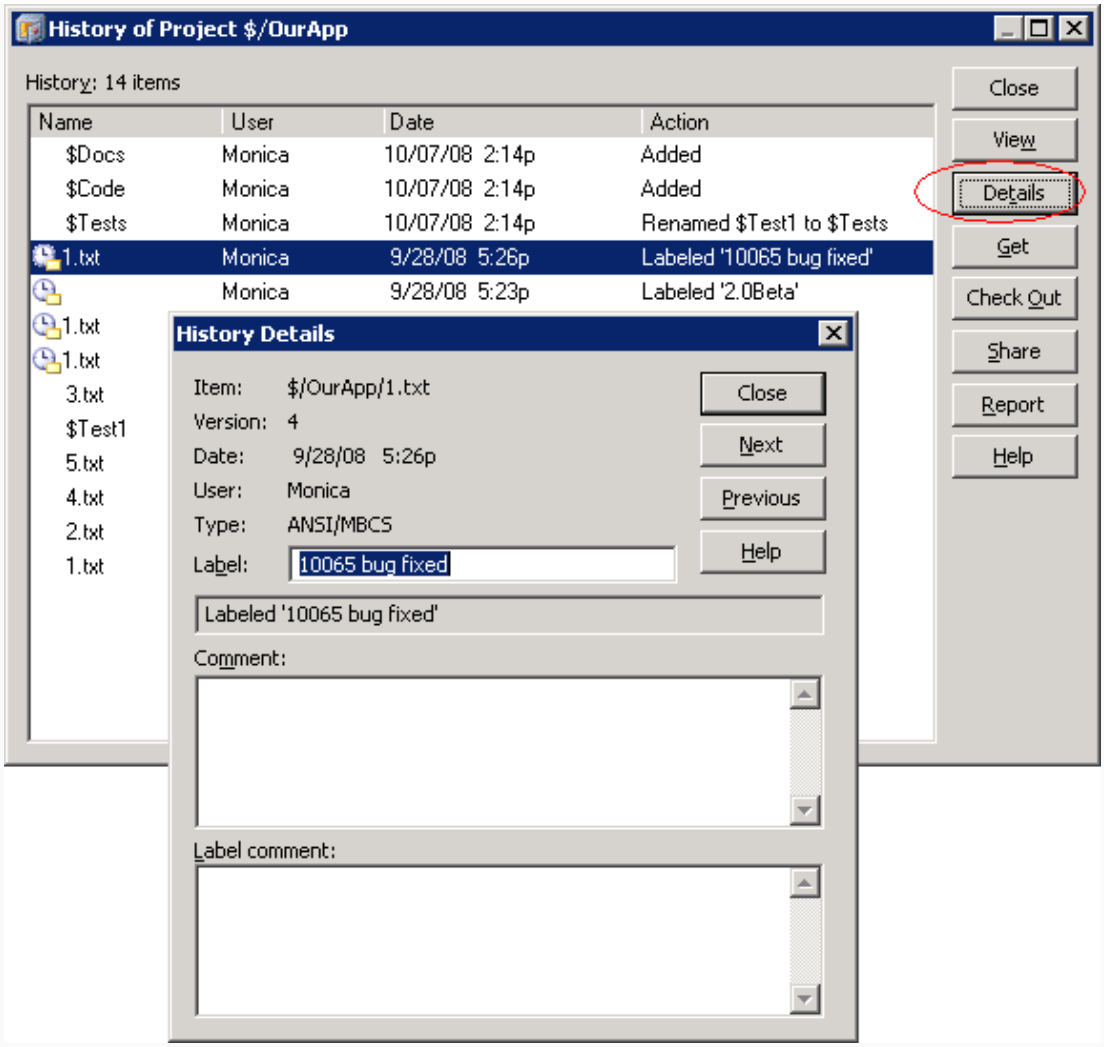
(Label File)



(Label Project)

To label an historical version of a file or project:

1. Select the file or project to label in Visual SourceSafe Explorer.
2. Click **Show History** on the **Tools** menu.
3. Click **OK** in the **History Options** dialog box.
4. Select the version of file or project to be labeled.
5. Click Details in the **History of File/Project** dialog box.
6. Input a label text in the **Label** edit box in the **History Details** dialog box, as seen in the following screen shot:

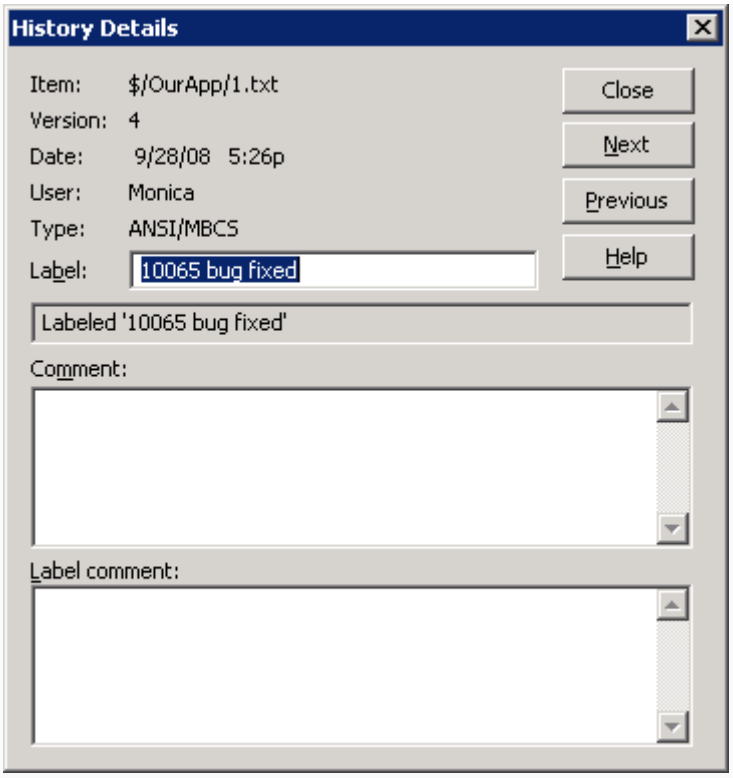


(Label History Version of a Project)

Please note that when we label a project, all the files and subprojects in that project inherit the label.

How to Modify a Label

We can modify the label of a file or project in the **History Details** dialog box, as seen in the following screen shot:



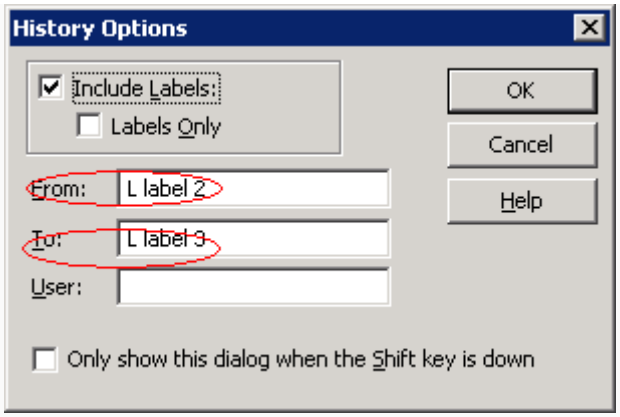
(History Details)

If we modify the label of a project, the label of the files and subprojects in that project will be changed accordingly.

How to Get Files/Projects by Label

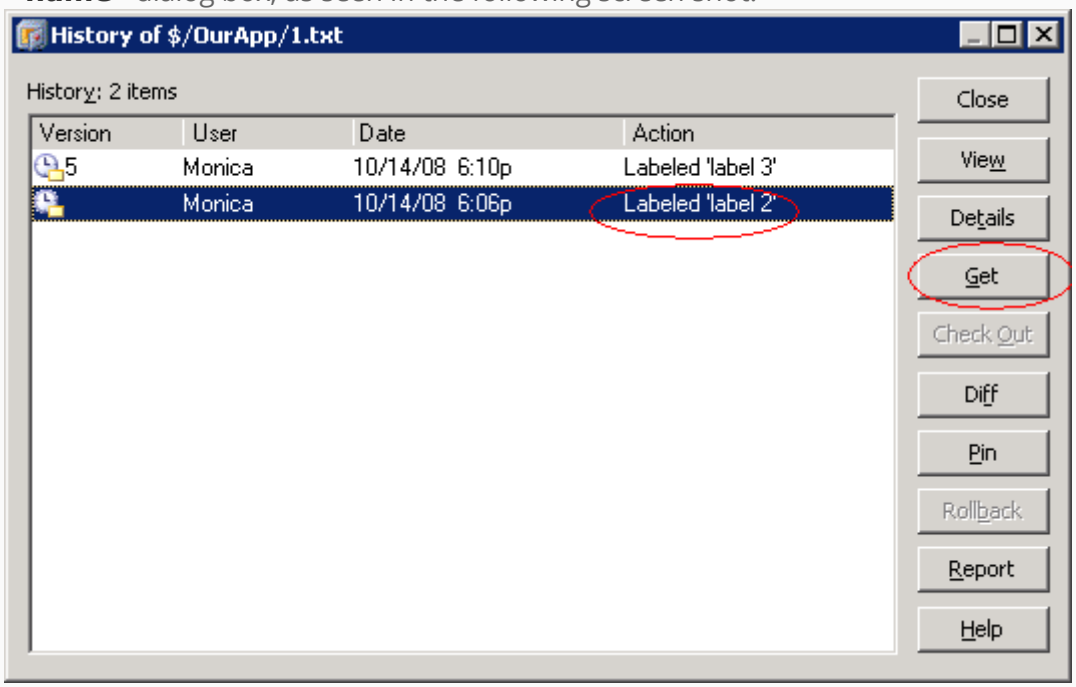
We can get a file with specific label. To do so, we can follow steps below:

1. Select the file to get in the Visual SourceSafe Explorer.
2. Click **Show History** on the menu **Tools**.
3. Set options as seen in the following screenshot to show file versions only between label 2 and label 3 in the **History Options** dialog box.



(History Options)

4. Click **Get** to retrieve the selected version of the file in the **History of <file name>** dialog box, as seen in the following screen shot:



(History of <file name>)

Scenarios When Label May Be Performed

- Assign a label to a new version of the file or project to keep track of the file and project versions.
- Edit a label in the **History Details** dialog box to assign a new label to an existing version.
- Get by label on the labeled files or projects.

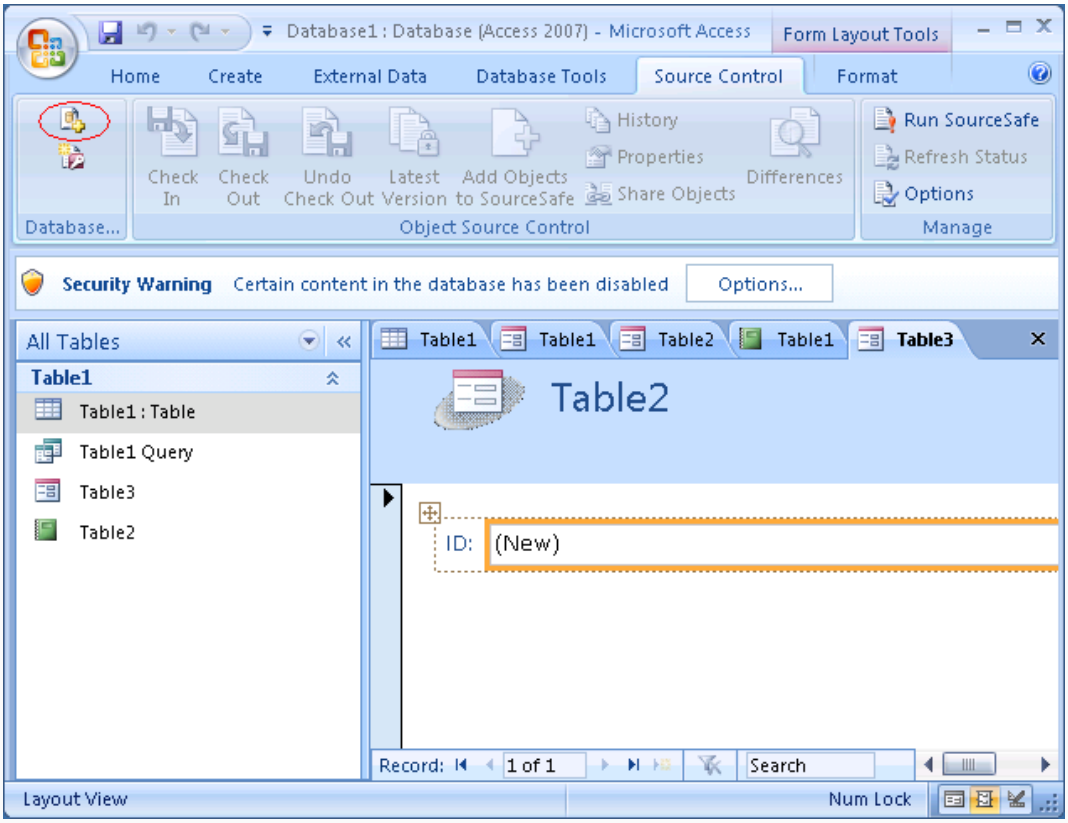
Integrating VSS with Access 2007

With Access Source Code Control add-in, Visual SourceSafe can be integrated into MS Access to source control Access queries, forms, reports, macros, modules and data. SourceSafe stores each Access objects as a text file. When you add an Access object to SourceSafe, Access exports the object to a text file in the working folder. Then the Access Source Code Control add-in adds the file to SourceSafe Database. When you check out/get an Access object from SourceSafe, the Access Source Code Control add-in copies the corresponding text file from SourceSafe to the working folder. Then Access imports the text file into MS Access database and turns the file into the appropriate Access object.

To integrate SourceSafe with Access 2007, please follow the steps below:

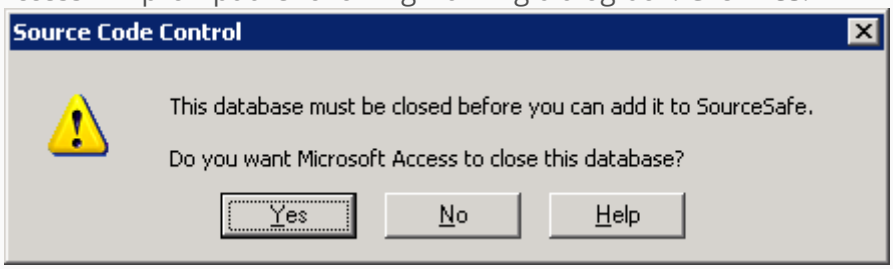
1. Install Access 2007 Developer Extensions, which includes source code control component. You can download it from: http://www.microsoft.com/downloads/details.aspx?familyid=d96a8358-ece4-4bee-a844-f81856dceb67&displaylang=en
2. Choose SourceSafe as the current SCC (Source Code Control) provider. For information on how to do it, see this: http://www.kevingao.net/sourcesafe/microsoft-source-code-control-interface-msscci-registry-entries.html
3. Open the database in Access 2007.

4. Add the database into source control of SourceSafe. Click menu **Source Control** and click **Add Database to SourceSafe** button.

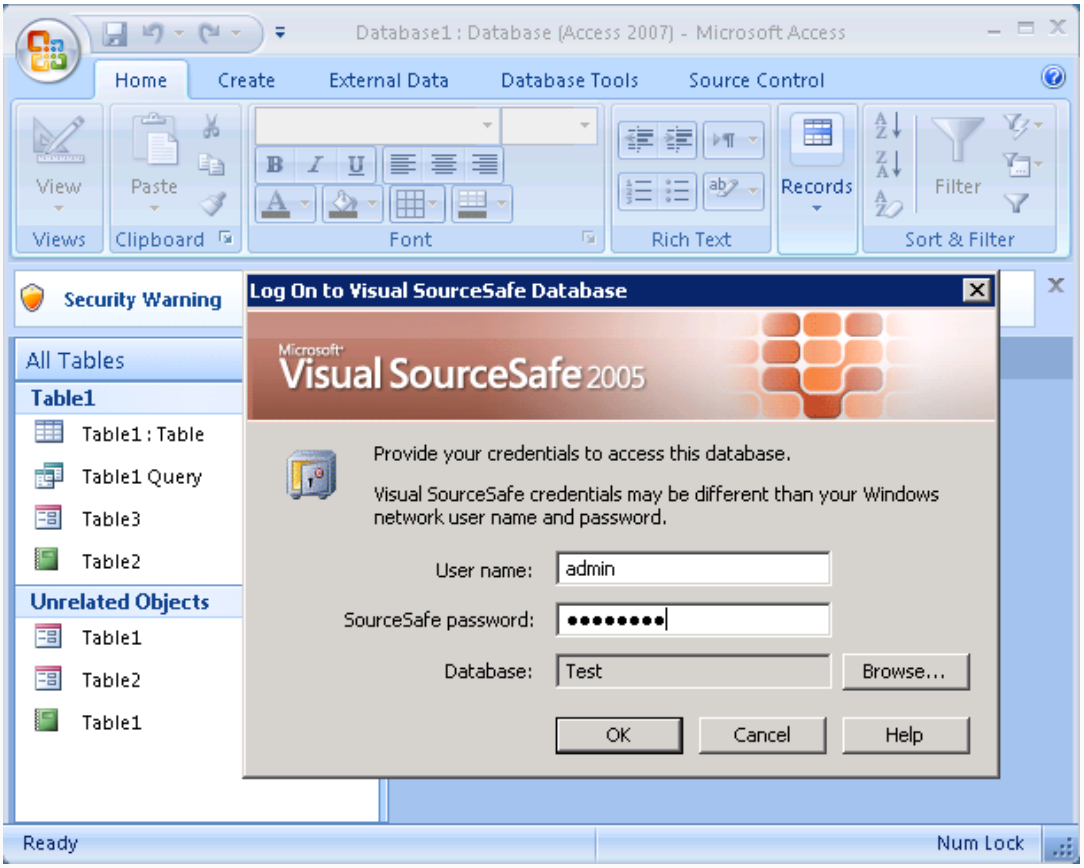


(Add Access database to SourceSafe)

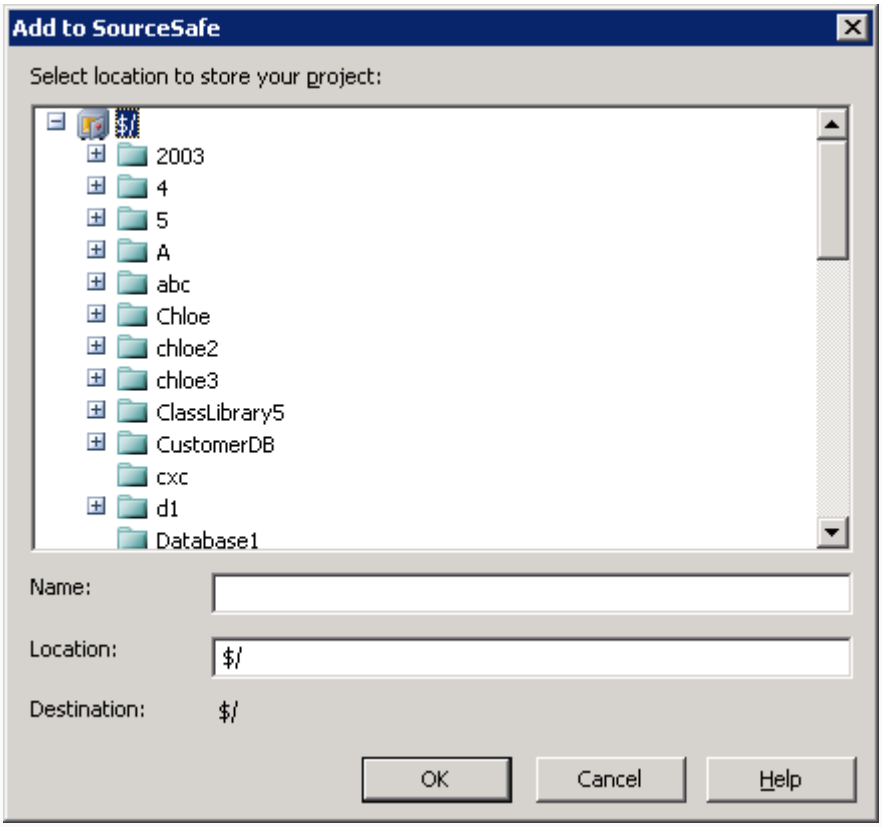
5. Then Access will prompt the following warning dialog box. Click **Yes**.



6. Log into a SourceSafe database and select the location to store the project.

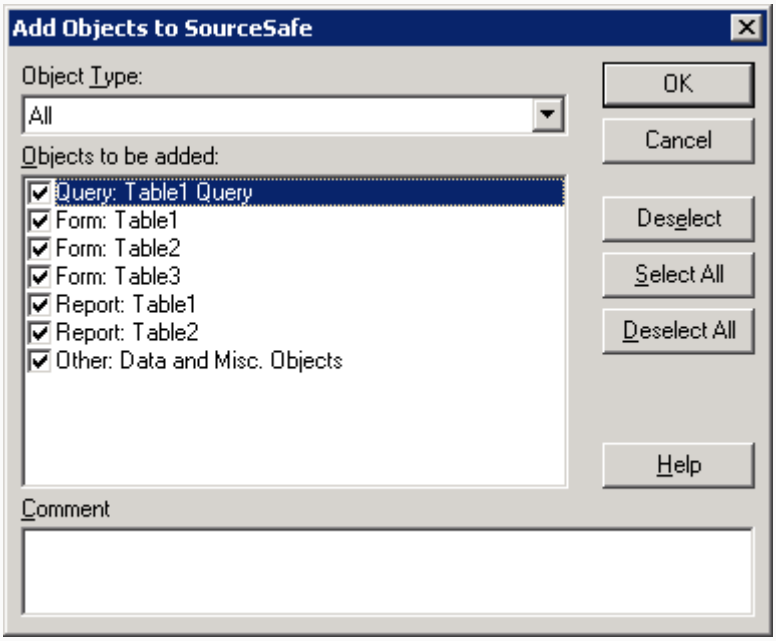


(Log in SourceSafe)



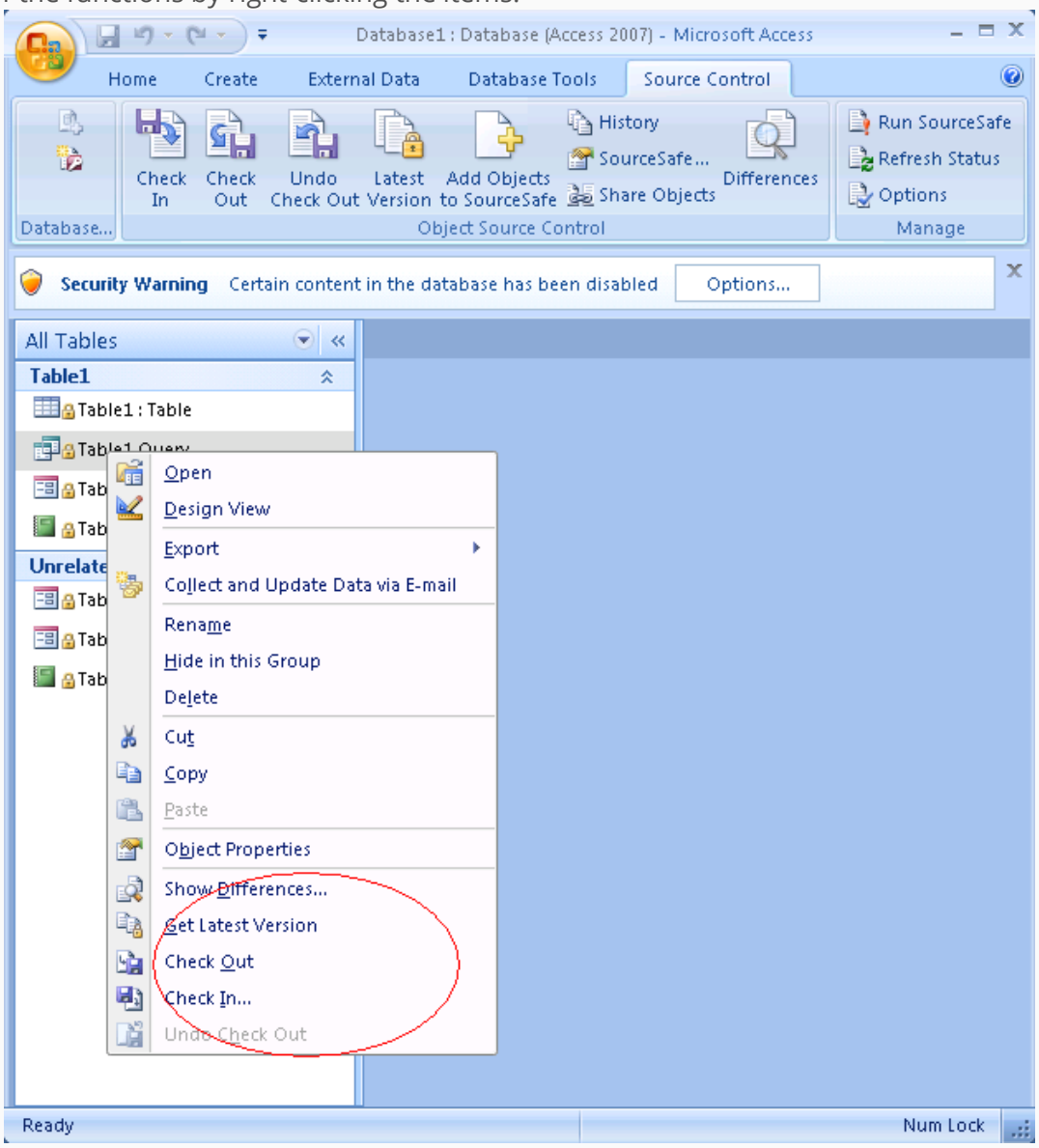
(Add the project to SourceSafe)

7. Add the Access objects into SourceSafe.



(Add Access objects into SourceSafe)

8. Now the objects are in the source control of SourceSafe. You can find the SourceSafe functions by clicking the **Source Control** tab. You can also access some of the functions by right-clicking the items.



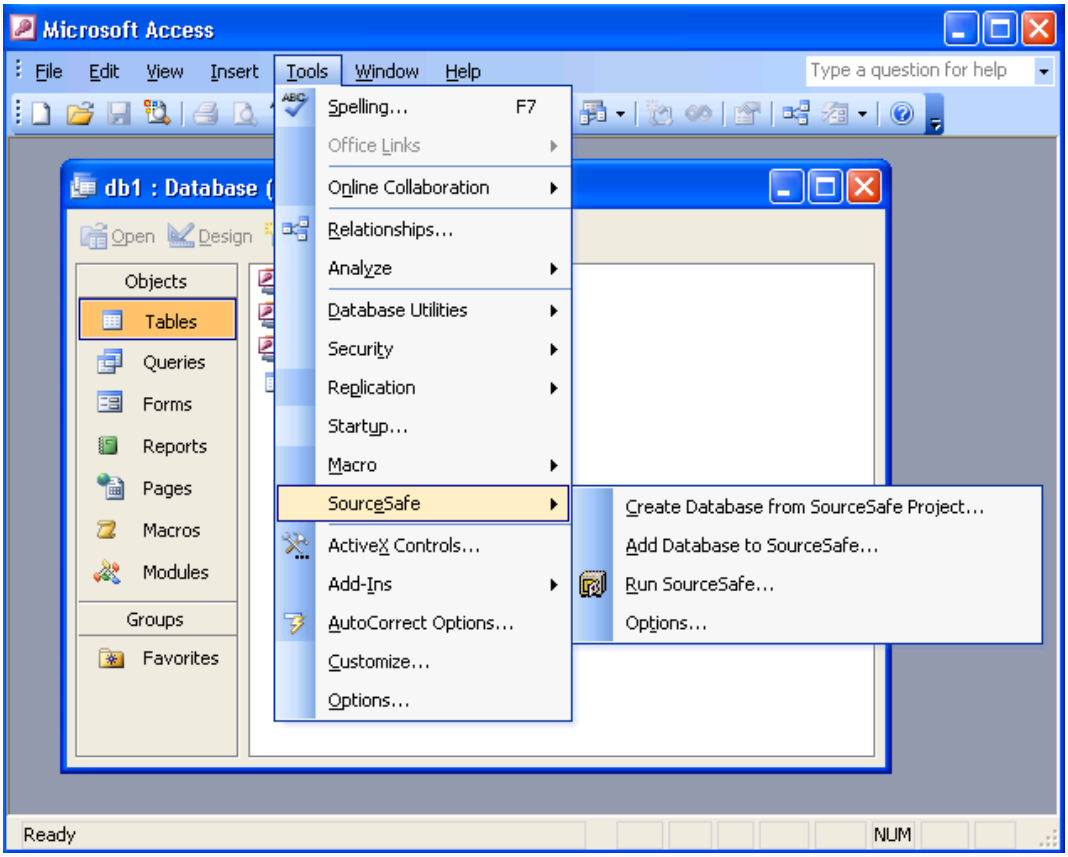
(SourceSafe functions)

Integrating VSS with Access 2003

In previous article, I wrote about how to integrate SourceSafe with Access 2007. This time, I will write about integrating SourceSafe with Access 2003. There is common information in the previous article. Please take a look at that article first at:http://www.kevingao.net/sourcesafe/integrating-sourcesafe-vss-with-access-2007.html

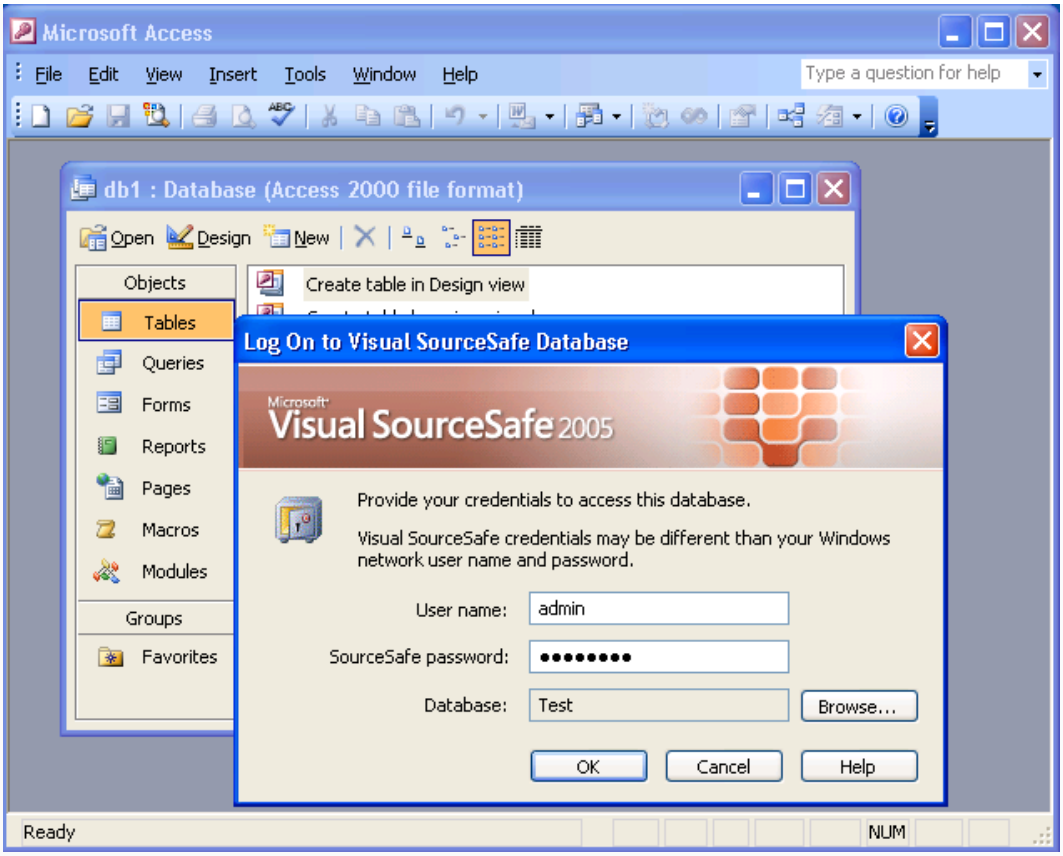
The steps of integrating VSS and Access 2003 are similar:

1. Install Access 2003 source code control add-in. It can be downloaded from: http://www.microsoft.com/downloads/details.aspx?familyid=2ea45ff4-a916-48c5-8f84-44b91fa774bc&displaylang=en
2. Choose SourceSafe as the current SCC (Source Code Control) provider. For information on how to do it, see this: http://www.kevingao.net/sourcesafe/microsoft-source-code-control-interface-msscci-registry-entries.html
3. Open the database in Access 2003 and you will find the **SourceSafe** command under the **Tools** menu.
4. Add the database into source control of SourceSafe by clicking menu **Tools** ->**SourceSafe** -> **Add Database to SourceSafe**.

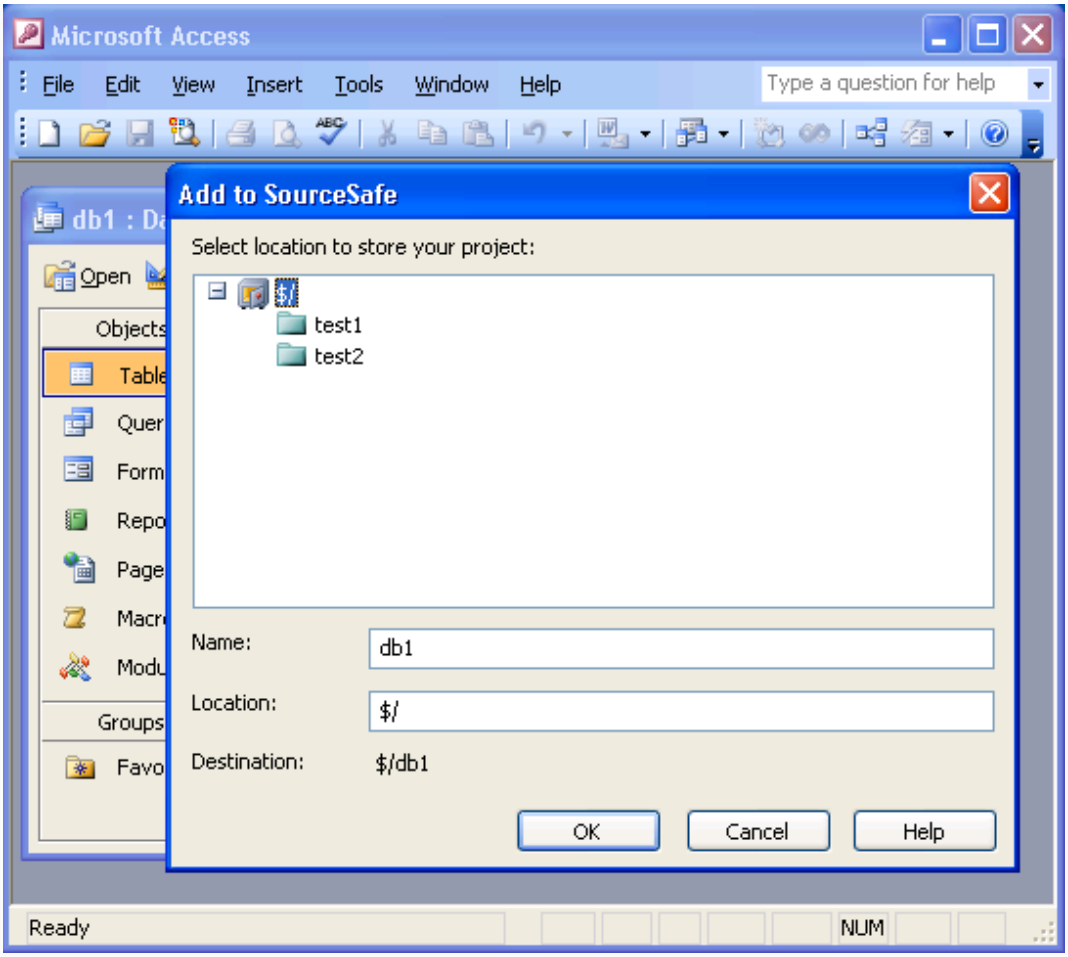


(Add Access database to SourceSafe)

5. Log into a SourceSafe database and select the location to store the project.

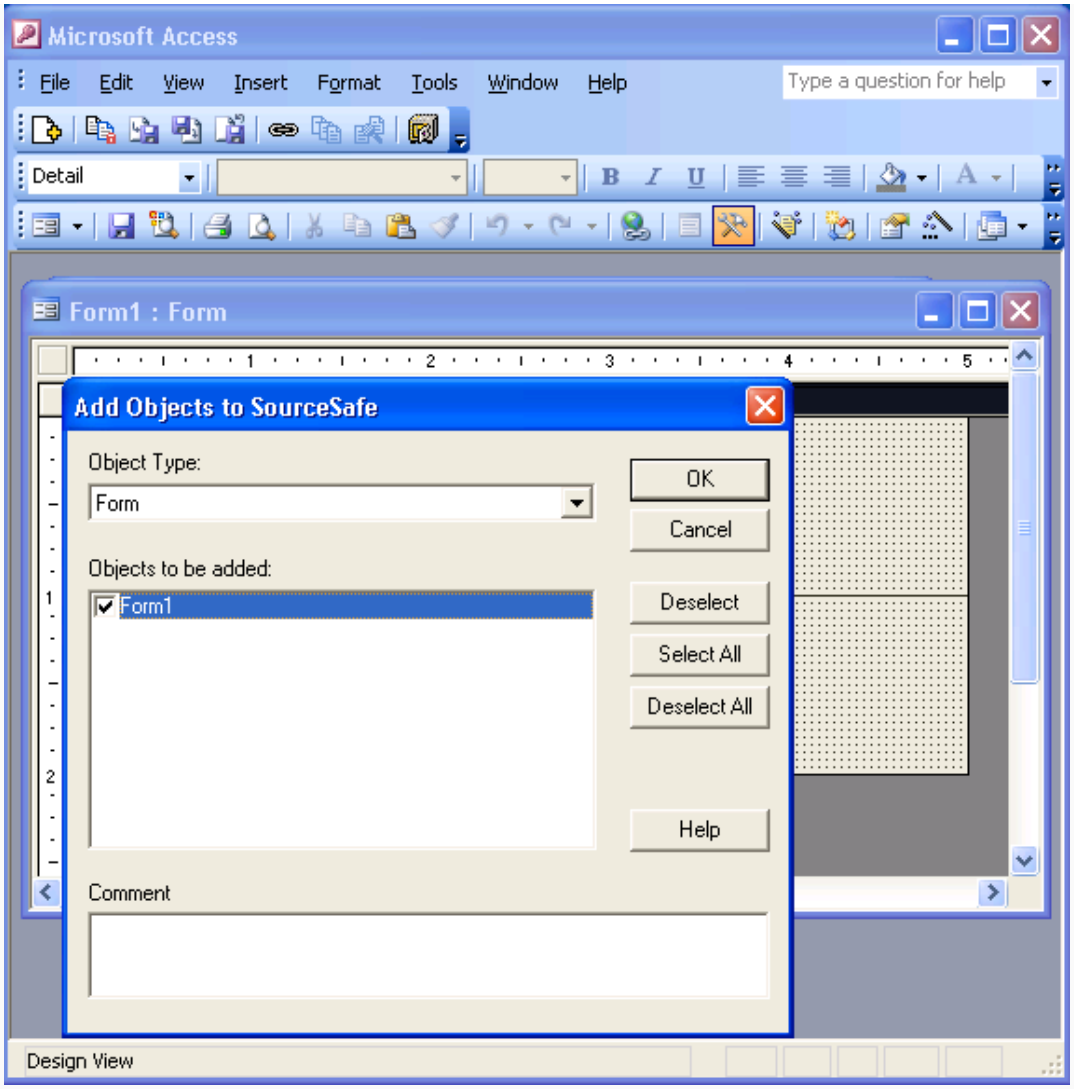


(Log in SourceSafe)



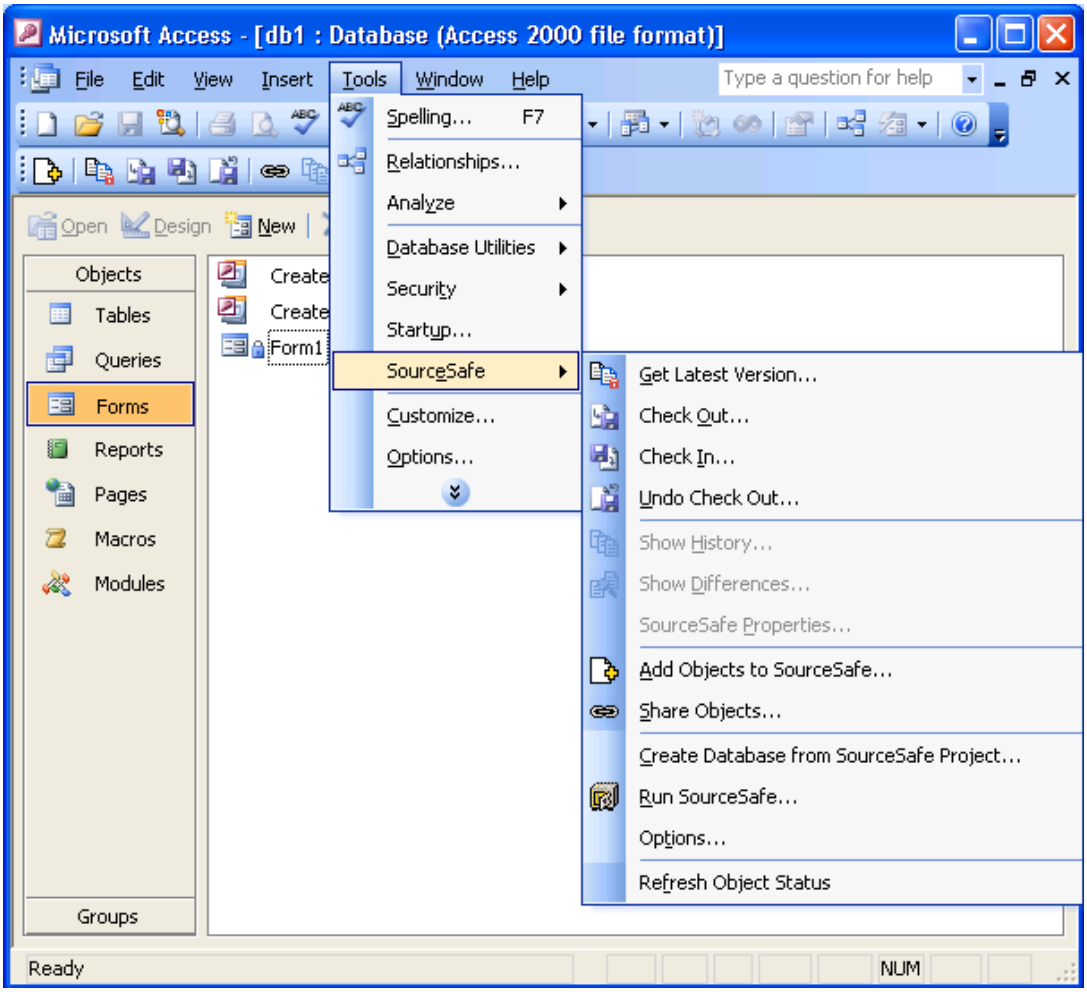
(Add the project to SourceSafe)

6. Add the Access objects into SourceSafe.

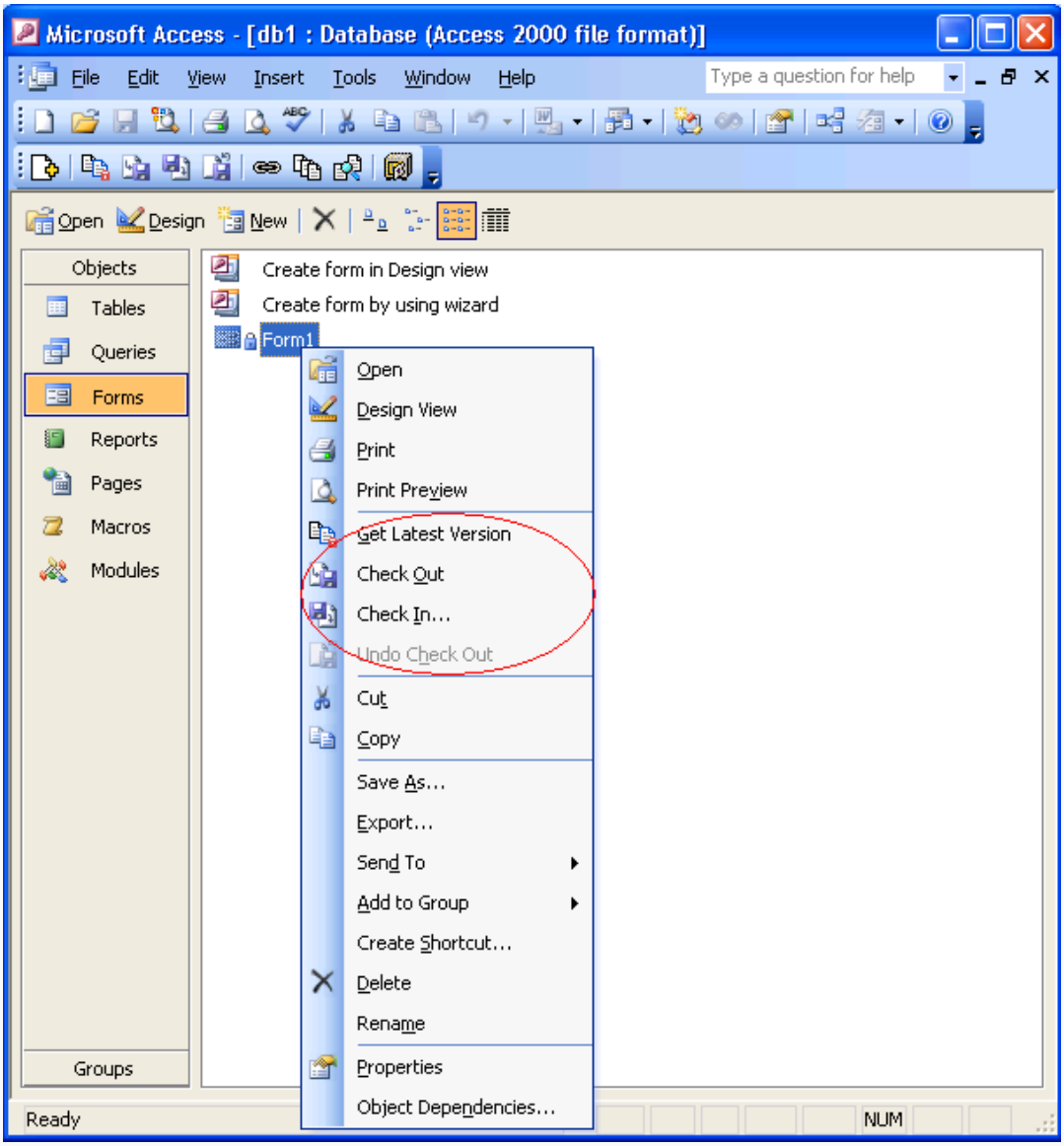


(Add Access objects into SourceSafe)

7. Now the objects are in the source control of SourceSafe. You can find the SourceSafe functions by clicking menu **Tools** -> **SourceSafe**.



You can also access some of the functions by right-clicking the items.


Microsoft Access - [db1 : Database (Access 2000 file format)]
File Edit View Insert Tools Window Help
Type a question for help
Open Design New
Objects
Tables
Queries
Forms
Reports
Pages
Macros
Modules
Groups
Create form in Design view
Create form by using wizard
Form1
Open
Design View
Print
Print Preview
Get Latest Version
Check Out
Check In...
Undo Check Out
Cut
Copy
Save As...
Export...
Send To
Add to Group
Create Shortcut...
Delete
Rename
Properties
Object Dependencies...
Ready
NUM

Integrating VSS with SQL Server 2008

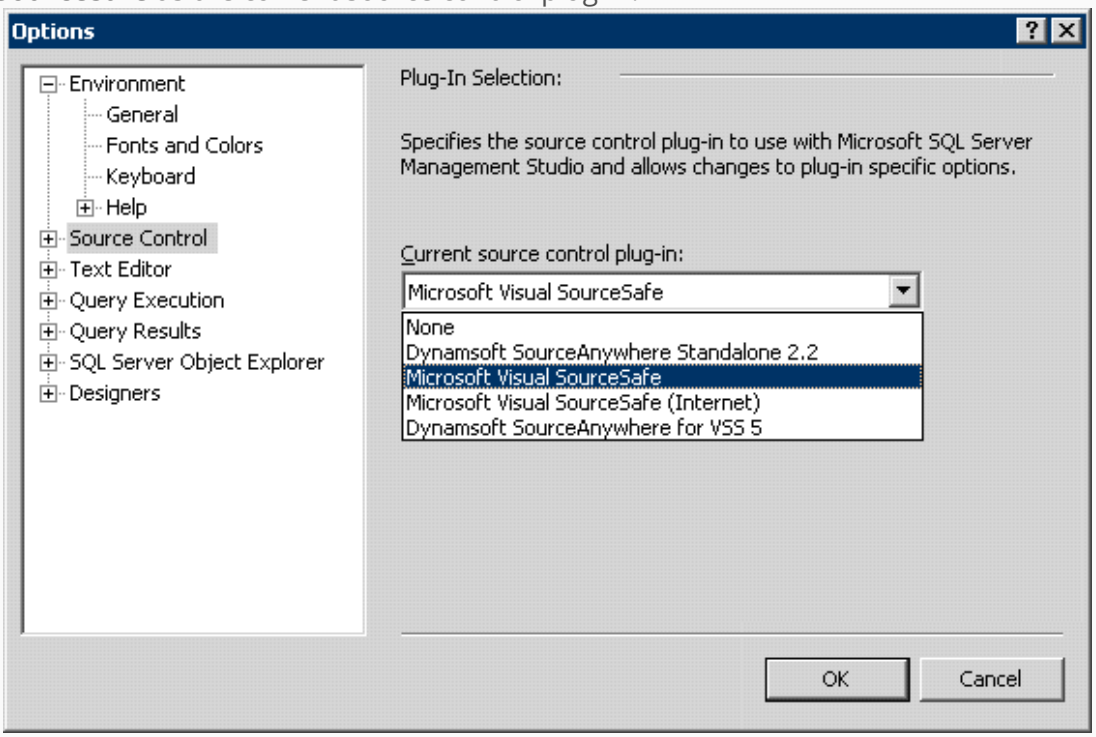
Visual SourceSafe can be integrated with SQL Server Management Studio 2008 to facilitate team activities and enable parallel development. We can version control Table, Stored Procedure, Trigger, Rule, etc. in SQL Server 2008.

We cannot directly version control the SQL objects, such as stored procedures and tables, in SQL Server Management Studio. However, SQL Server Management Studio (SSMS) does provide a mechanism to create scripts for most object types. We could create scripts for the database and all its objects via Management Studio, save them to a solution and then add the solution to source control.

In this article, we will see how to add solution/project into source control in SSMS. Here are the steps we can follow:

1. We need to install Visual SourceSafe on the machine hosting SQL Server SSMS.

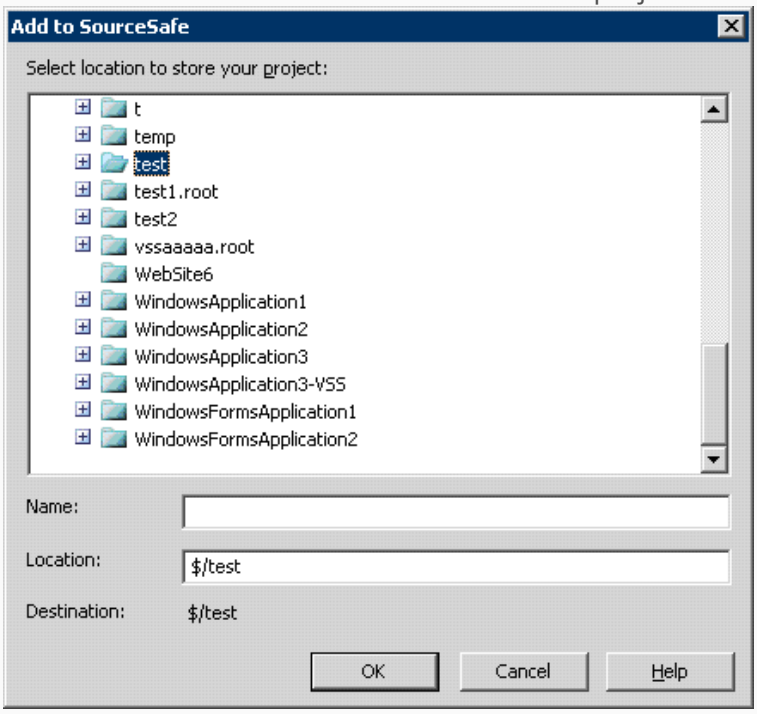
2. Open SQL Server Management Studio 2008, and click menu **Tools** -> **Options** -> **Source Control**. In the Source Control page, we can select **Microsoft Visual SourceSafe** as the current source control plug-in.



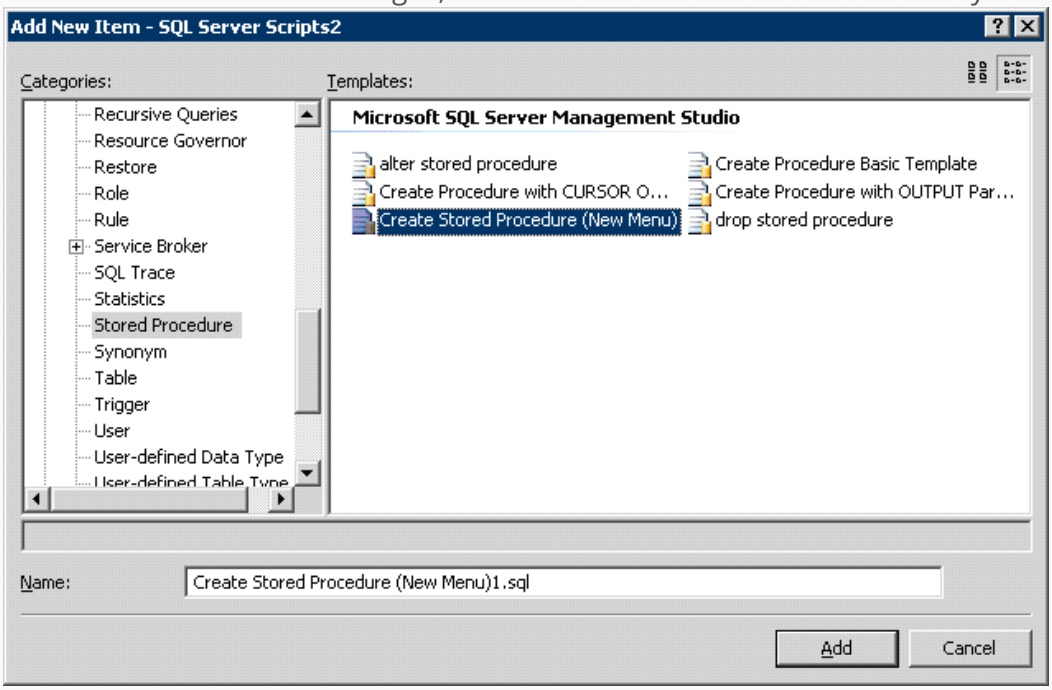
3. Open an existing project/solution or create a new one through the SQL Server Management Studio menu **File**. We can open the solution explorer by click menu **View** ->**Solution Explorer**.

4. Next, we need to add the solution/project into source control by right-click the solution/project file and clicking **Add Solution to Source Control** menu. The following dialog box will prompt out:

5. In the **Log On to SourceSafe Database** dialog box, please enter the credentials and click**OK**. Then we need to choose a location in the VSS project tree:



6. Now we can add new item/ existing item/ new connection/ new query by right-clicking the project file and click **Add**. These items will be in the **Pending Checkins** window. After checking in, the items will be under source control by VSS.



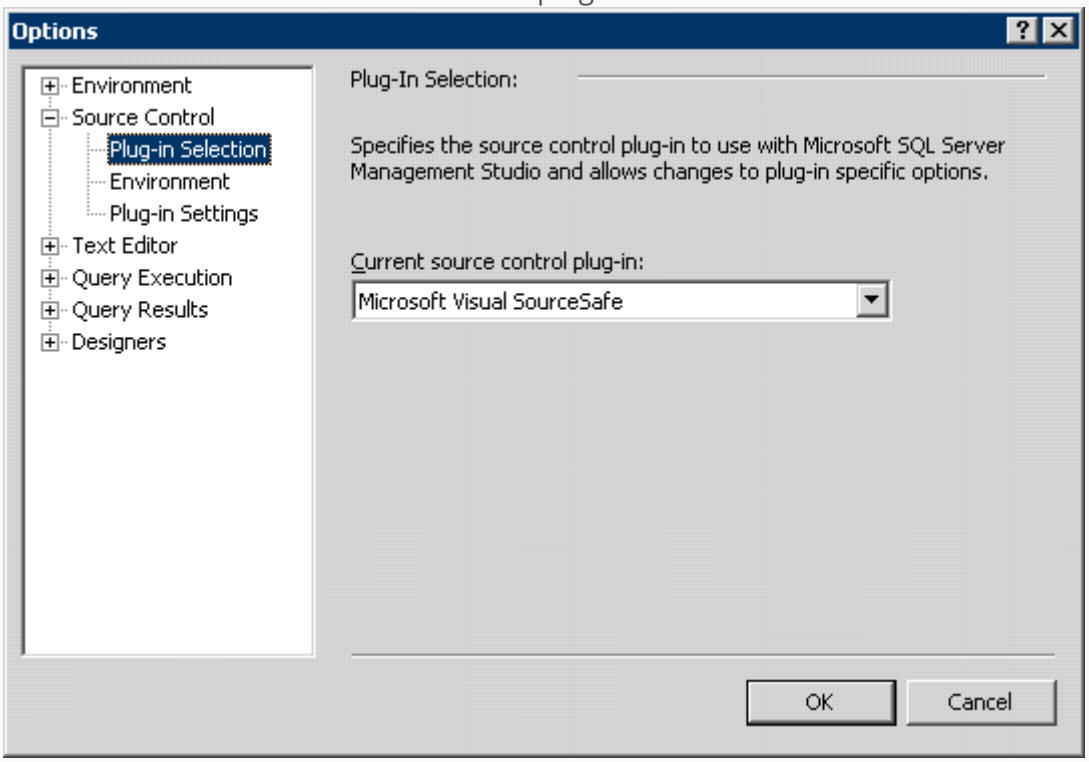
In the following articles, I will talk about VSS integration with SQL Server 2000 and 2005.

Integrating VSS with SQL Server 2005

The Visual SourceSafe integration with SQL Server Management Studio (SSMS) 2005 is virtually identical to SQL Server Management Studio (SSMS) 2008.
Similar to SSMS 2008, we cannot version control SQL objects directly in SSMS 2005. However, we can create scripts for the database and all its objects, save them to a solution and then add the solution to source control. For more details, please refer to another article: How to add SQL Server 2005/2008 Stored Procedures to VSS.

Here are the steps we can follow:
1. Install Visual SourceSafe on the machine hosting SQL Server 2005.

2. Open SQL Server Management Studio 2005, and click menu **Tools** -> **Options** -> **source control**. In the Source Control page, we can select **Microsoft Visual SourceSafe** as the current source control plug-in.



(SQL Server 2005 Options)

3. Open an existing project/solution or create a new one through the SQL Server Management Studio menu **File**. We can open the solution explorer by click menu **View** ->**Solution Explorer**.

4. Next, we need to add the solution/project into source control by right-click the solution/project file and clicking **Add Solution to Source Control** menu.
5. We can add existing item/ new connection/ new query by right-clicking the project file and click **Add**. These items will be in the **Pending Checkins** window. After checking in, the items will be under source control by VSS.

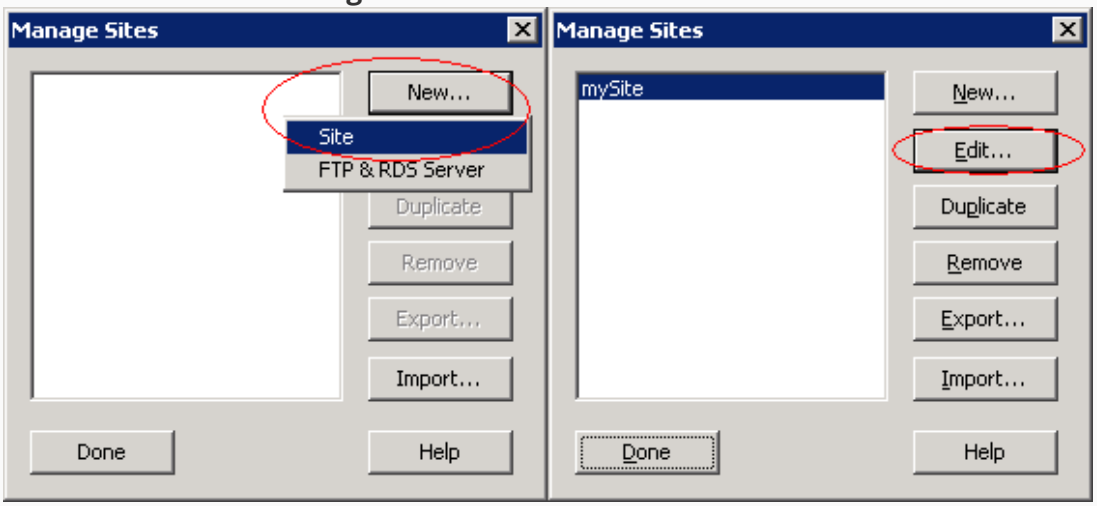
For the Visual SourceSafe integration with SQL Server Management Studio 2008, please refer to my other article: Integrating SourceSafe / VSS with SQL Server 2008.

Integrating VSS with Dreamweaver

Visual SourceSafe can be integrated into Dreamweaver to source control projects and files.

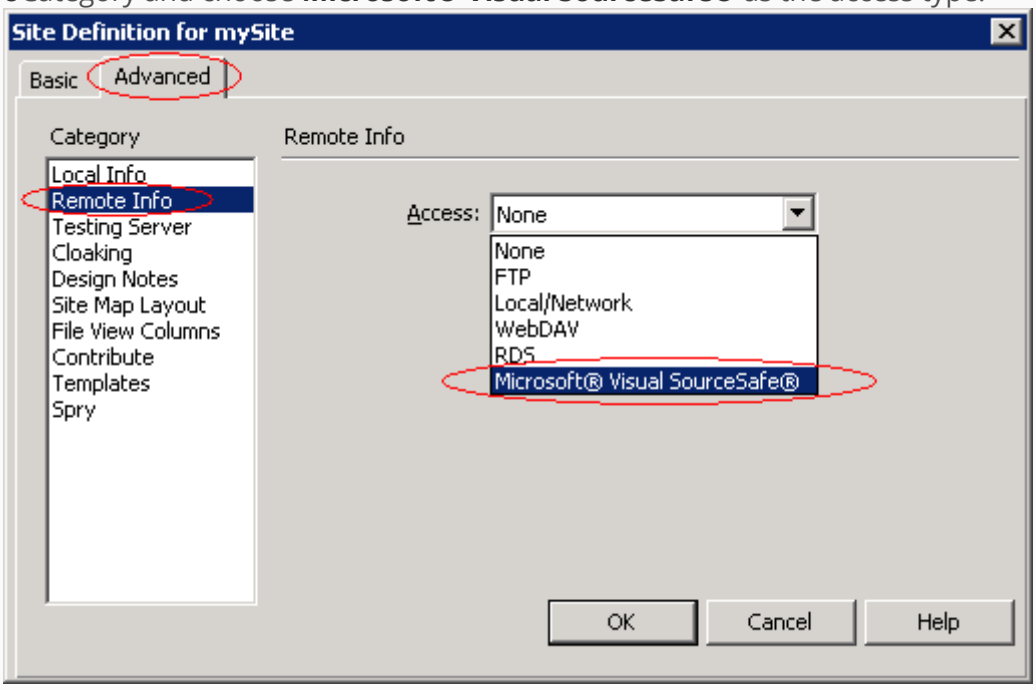
1. Open Dreamweaver.

2. Click **Site** menu -> **Manage Sites** to create a **New** site or to **Edit** a site.



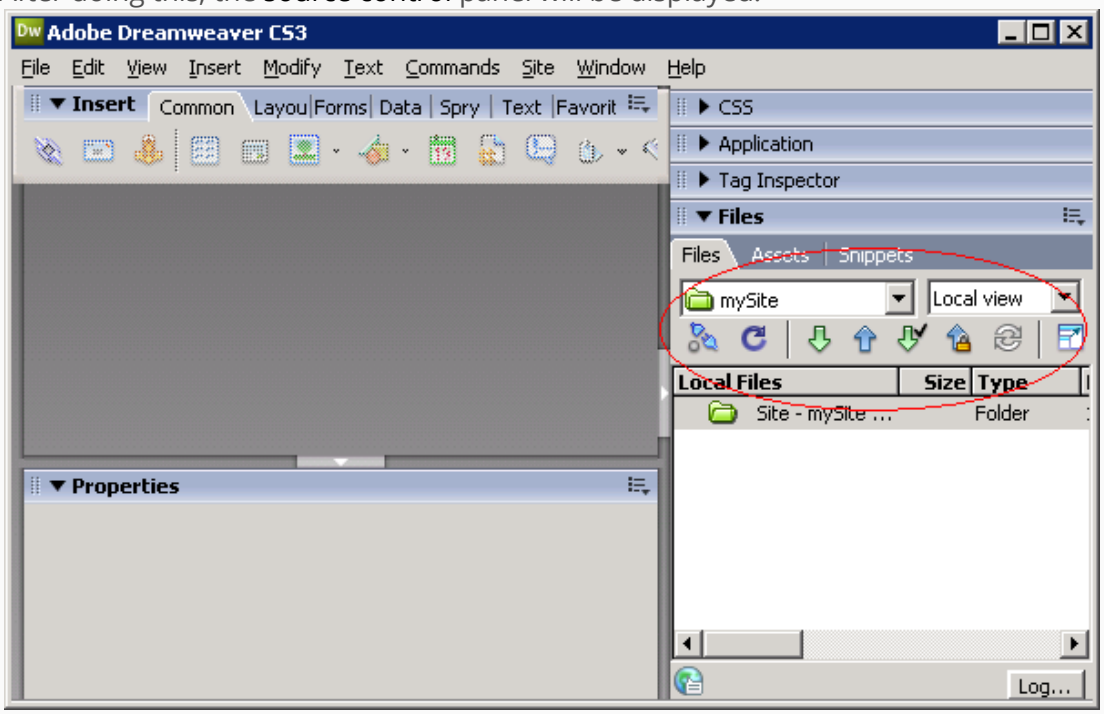
(New or Edit a site)

3. In the **Site Definition** dialog box, go to the **Advanced** tag, select the **Remote Info**Category and choose **Microsoft® Visual SourceSafe®** as the access type.



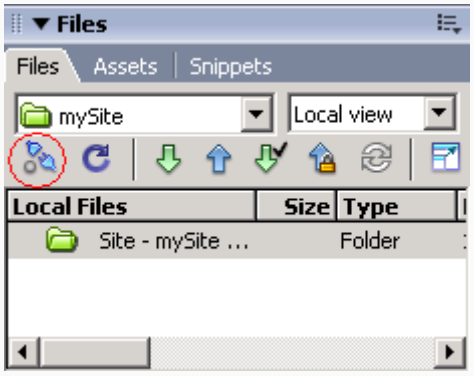
(Remote Info of site)

After doing this, the source control panel will be displayed.



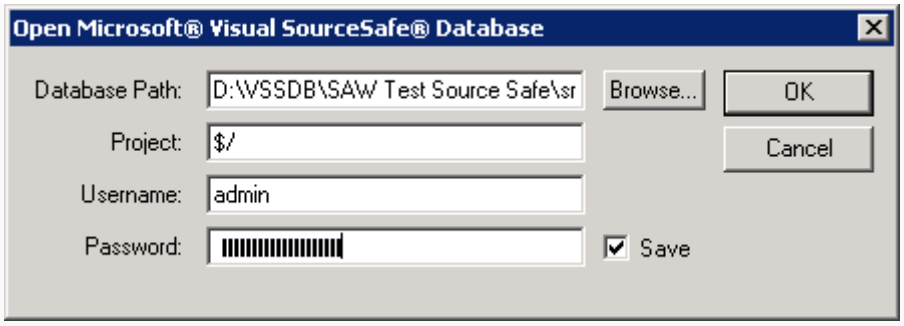
(Source control panel in Dreamweaver)

4. Click the **Connect** button to connect to remote host.



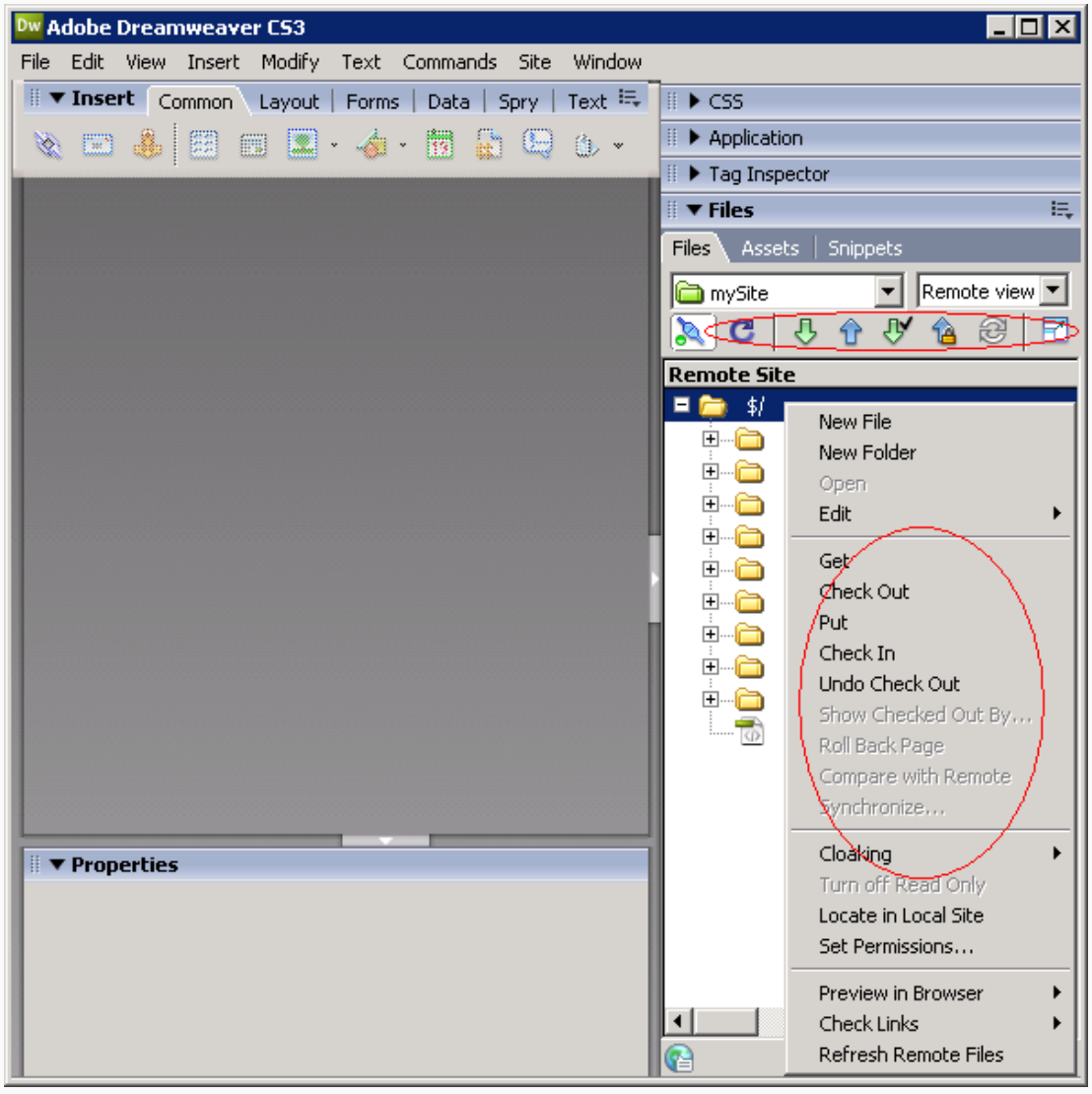
(Connect to remote host)

Input the information of **Database Path**, **Project**, **Username** and **Password** in the**Microsoft® Visual SourceSafe® Database** dialog box to open SourceSafe database.



(Open SourceSafe database)

5. Now the objects are in the source control of SourceSafe. You can find the SourceSafe functions by clicking the icons or right-clicking the items.



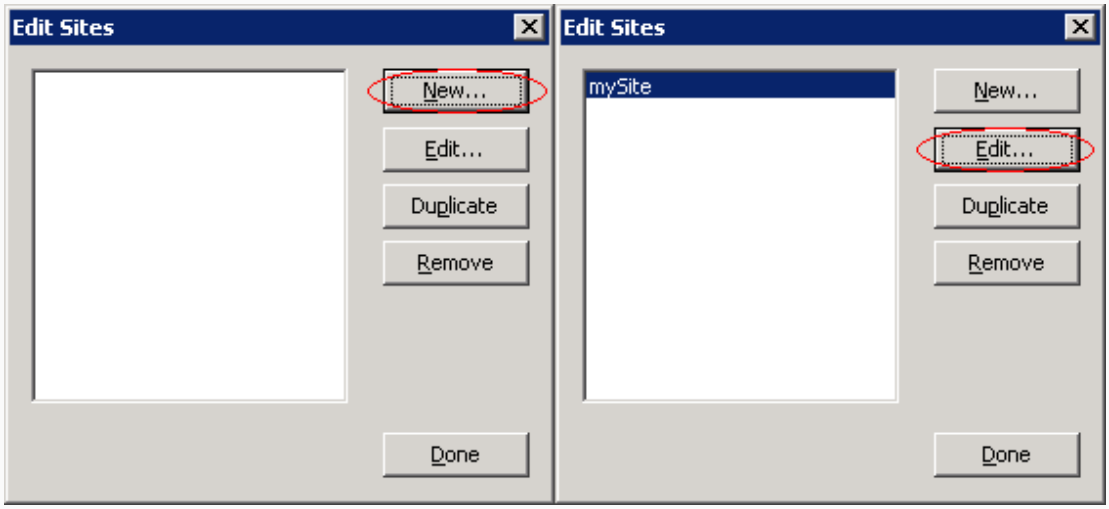
(Use SourceSafe in Dreamweaver)

Integrating VSS with Flash

Visual SourceSafe can be integrated into Flash (**Professional edition** only) to source control projects. To use source control functions, we should define a site for the project.

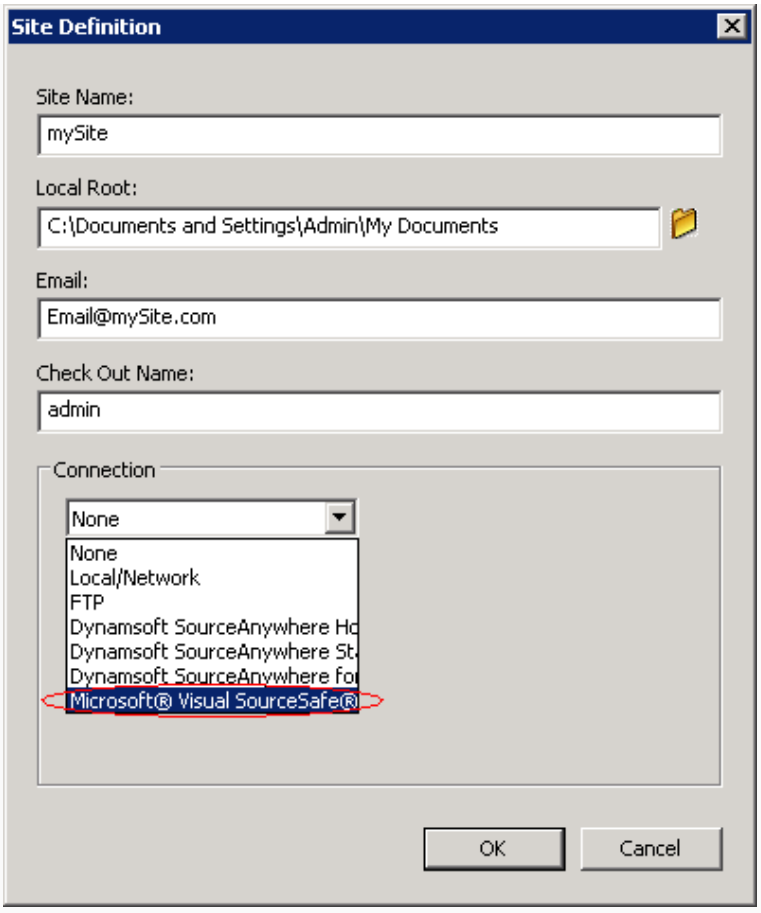
1. Open Flash.

2. Click menu **Site** -> **Edit Sites** to **New** or **Edit** a site.



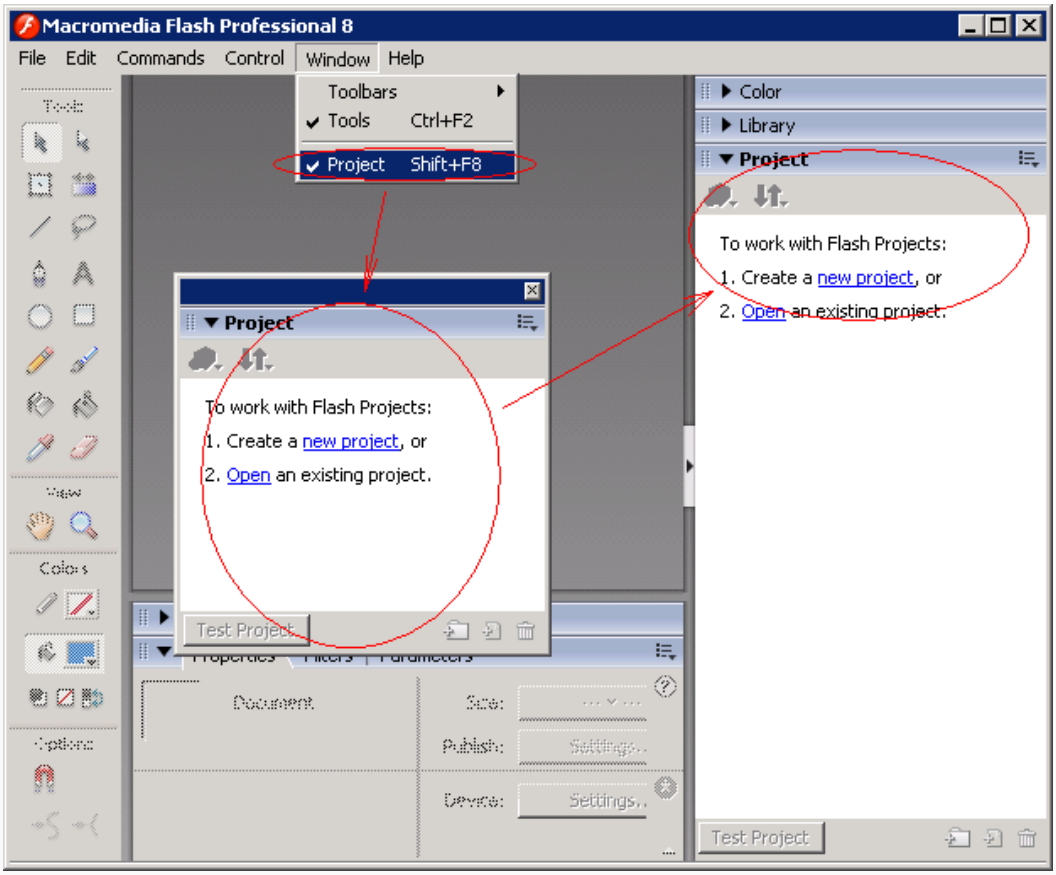
(New or Edit a site)

3. In the **Site Definition** dialog box, choose **Microsoft® Visual SourceSafe®** as the connection type.

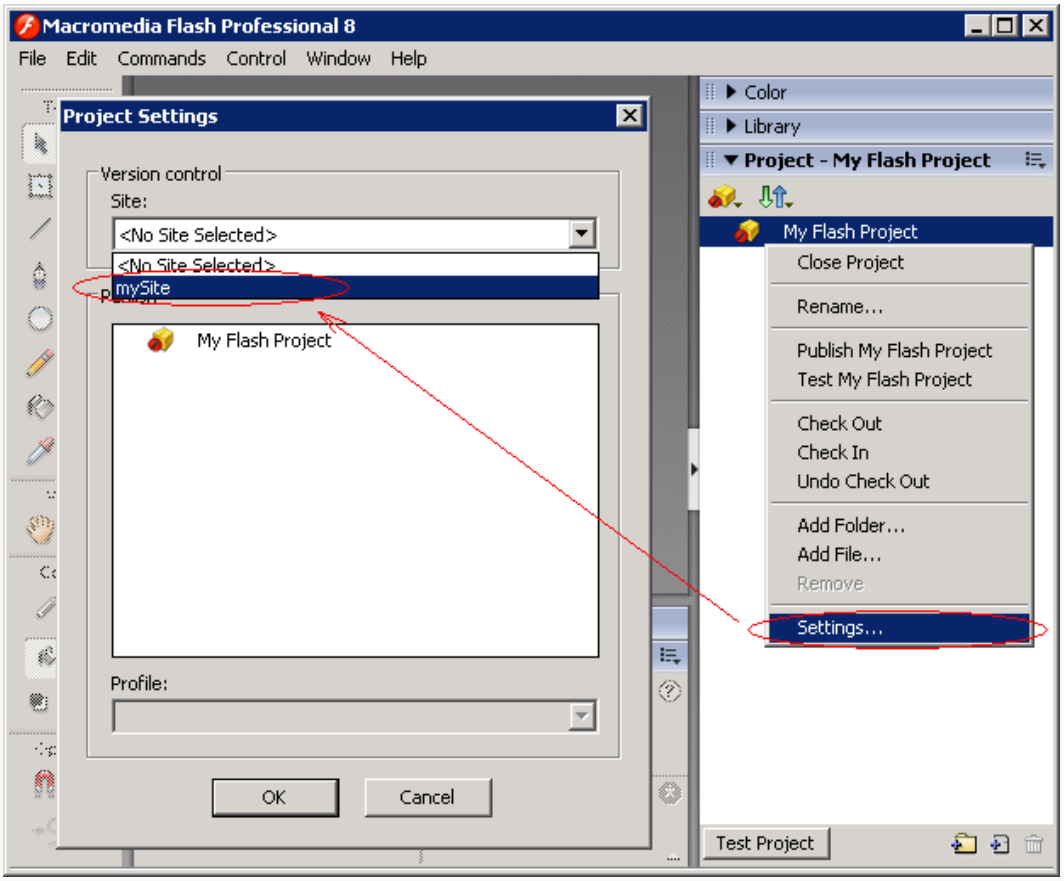


(Site Definition)

4. Check **Project** under menu **Window** to display the dialog box of the **Project** panel. Also, the dialog box of the **Project** panel can be dragged to the right menu.

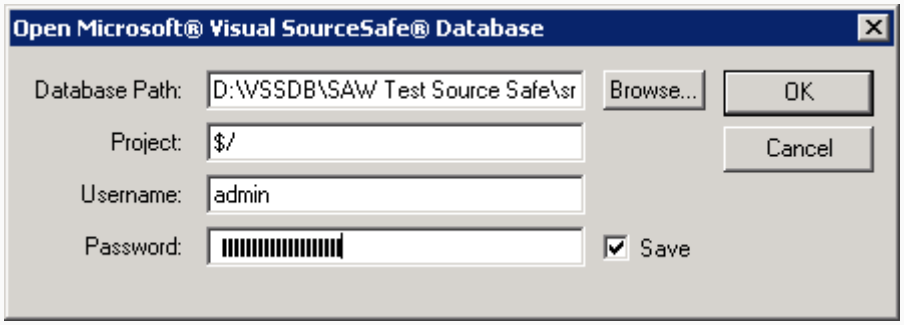


5. After creating a new project or opening an existing project, right click the project to show the **Project Settings** dialog box, and then select a site from the **Site** menu in the **version control** section and click **OK**.



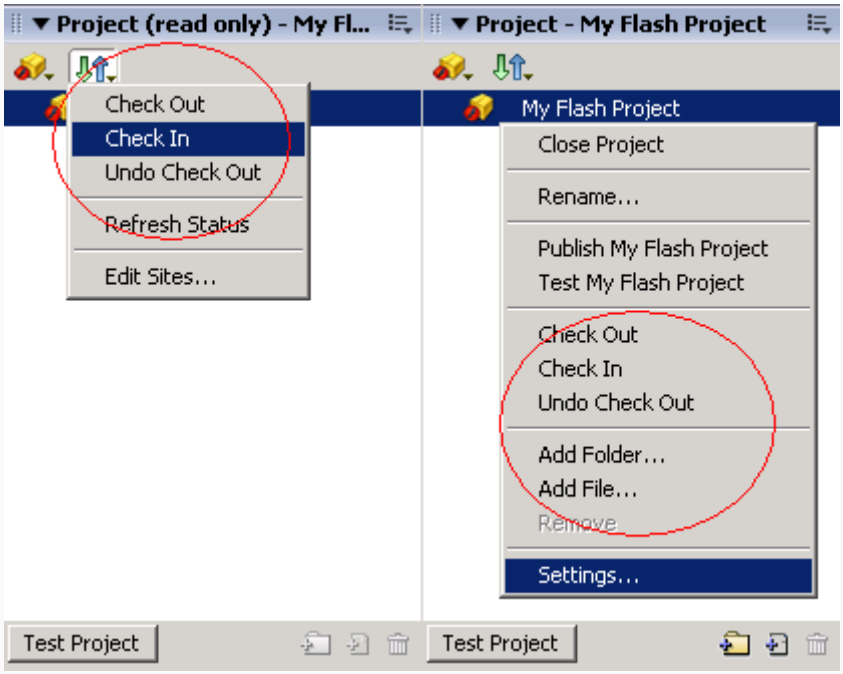
(Connect to remote host)

Input the information of **Database Path**, **Project**, **Username** and **Password** in the**Microsoft® Visual SourceSafe® Database** dialog box to open SourceSafe database.



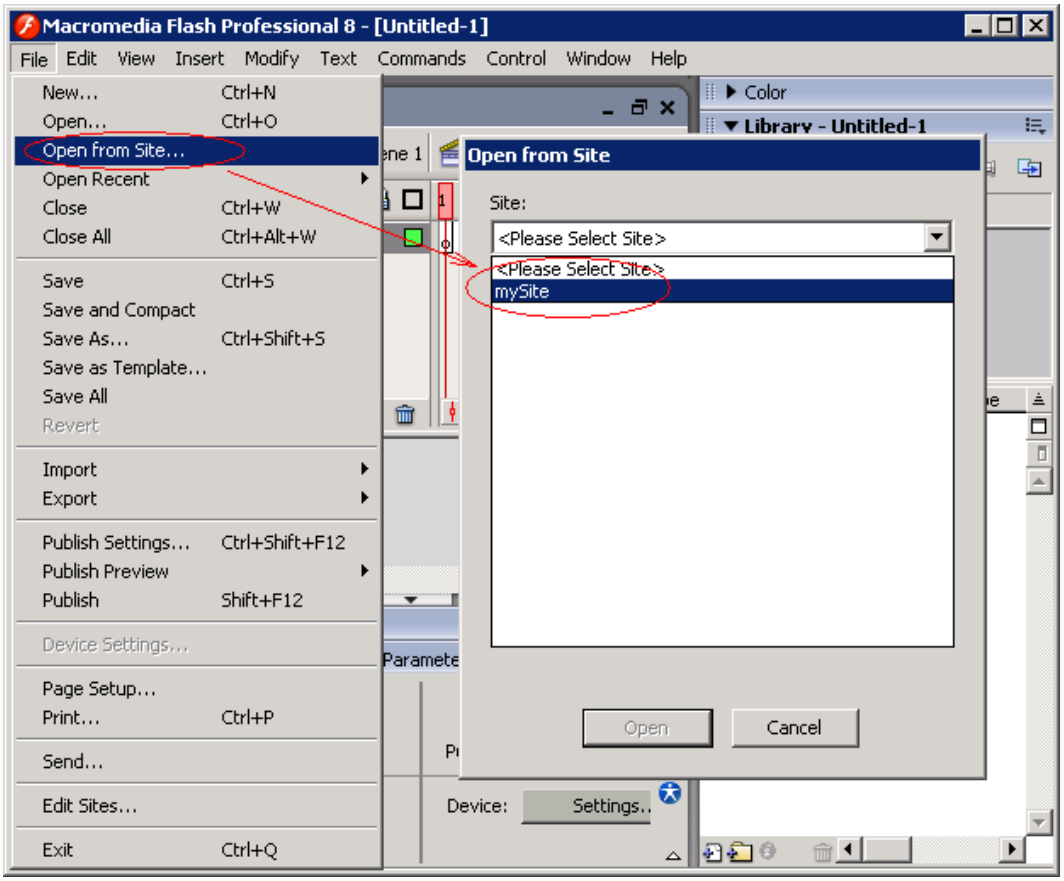
(Open SourceSafe database)

6. Now the project is in the source control of SourceSafe. You can find the SourceSafe functions by clicking the icon or right-clicking the project.



(Use SourceSafe in Flash)

7. When a project has version control applied, we can also open it by using **Open from Site**. Click menu **File** -> **Open from Site**. In the **Open from Site** dialog box, select the site from the Site menu and then select the file in the site.

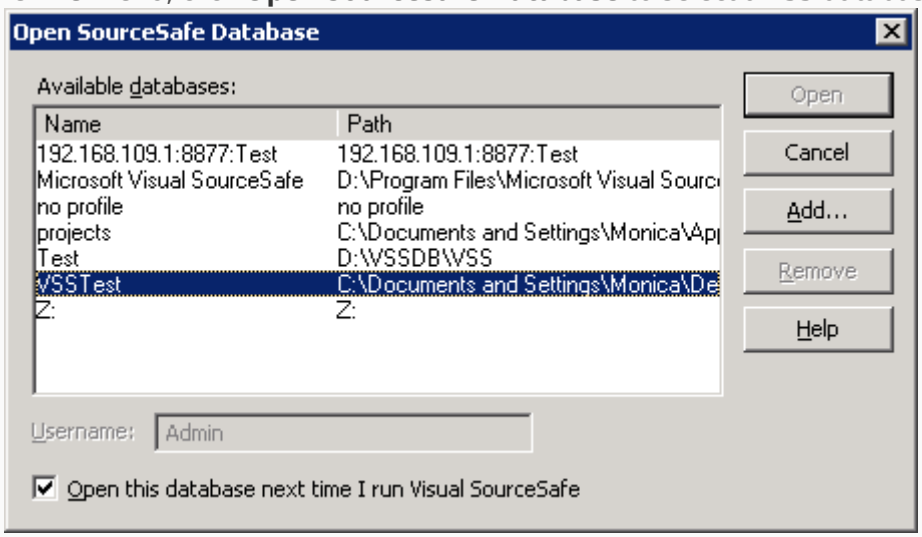

Macromedia Flash Professional 8 - [Untitled-1]
File Edit View Insert Modify Text Commands Control Window Help
New... Ctrl+N
Open... Ctrl+O
Open from Site...
Open Recent
Close Ctrl+W
Close All Ctrl+Alt+W
Save Ctrl+S
Save and Compact
Save As... Ctrl+Shift+S
Save as Template...
Save All
Revert
Import
Export
Publish Settings... Ctrl+Shift+F12
Publish Preview
Publish Shift+F12
Device Settings...
Page Setup...
Print... Ctrl+P
Send...
Edit Sites...
Exit Ctrl+Q
Color
Library - Untitled-1
Open from Site
Site:
<Please Select Site>
<Please Select Site>
mySite
Open
Cancel
Device:
Settings...

How to manage users

Visual SourceSafe (VSS) users are based on VSS databases. When a VSS database is created, there will be two default users added to the database: Admin and Guest. As the database administrator, we need to add user accounts for all of the users who will work on the database.

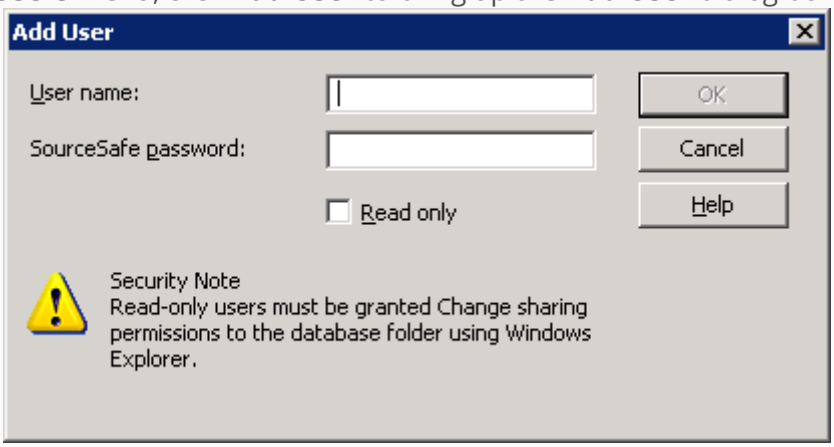
To add a user in a VSS database, please follow steps below:

1. Log in **Visual SourceSafe Administrator** as Admin.
2. On the **File** menu, click **Open SourceSafe Database** to select a VSS database.



(Open SourceSafe Database)

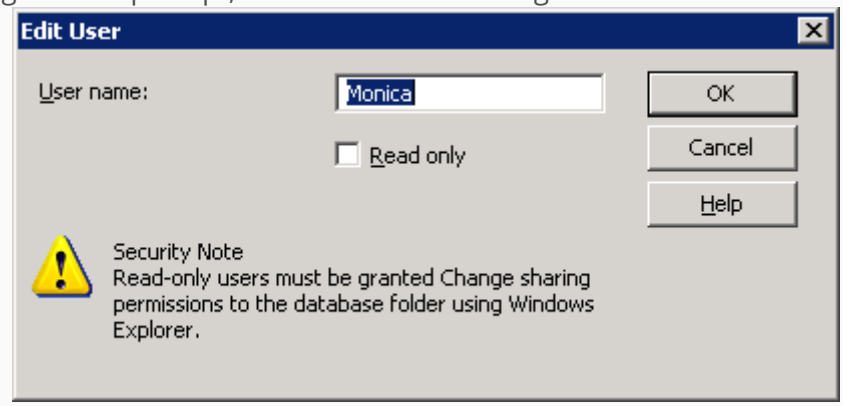
3. On the **Users** menu, click **Add User** to bring up the **Add User** dialog box.



(Add User)

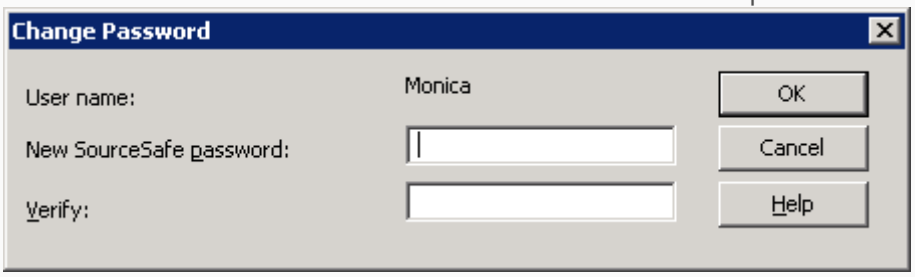
4. Input the user name and password. Check the **Read only** option to assign the new user read-only permissions for the current database, or leave it unchecked to assign the new user read-write permissions.
5. Click **OK** to finish adding a new user.

To edit a user in a VSS database, we can click **Edit User** on the **Users** menu in Visual SourceSafe Administrator, or we can just double-click the selected user, and the **Edit User**dialog box will prompt, as seen in the following screen shot:



(Edit User)

If other users forget their passwords, we as Admin can click **Change Password** on the**Users** menu in Visual SourceSafe Administrator to reset their passwords.



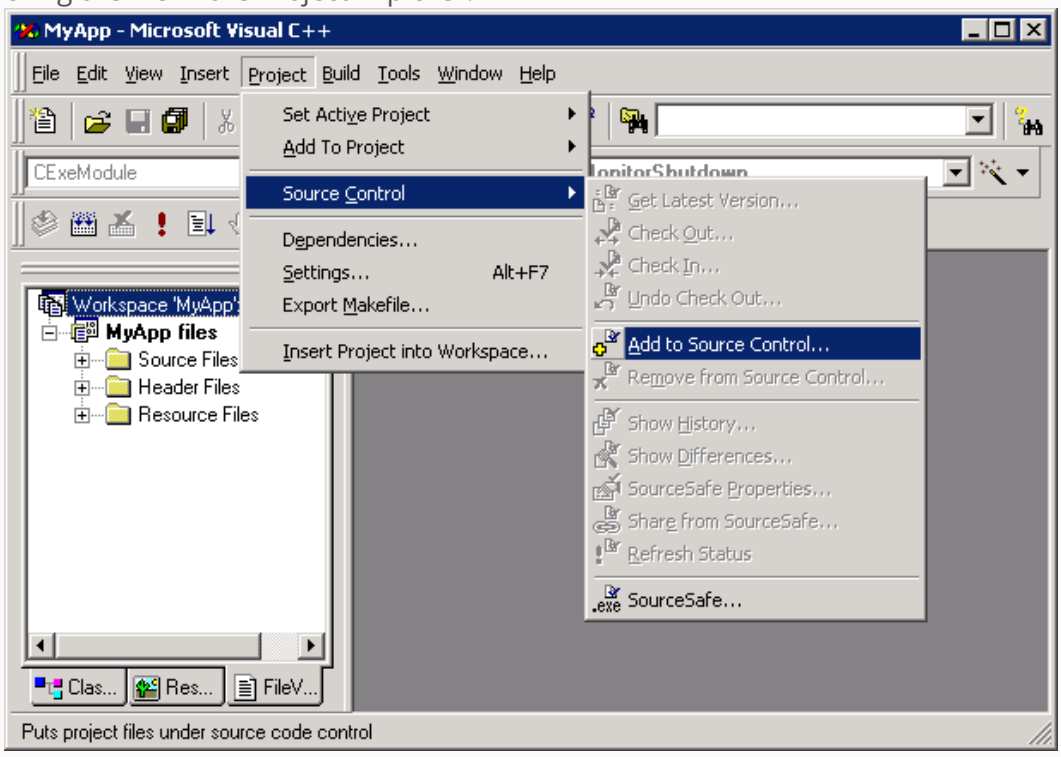
(Change Password)

Integrating VSS with Visual C++ 6.0

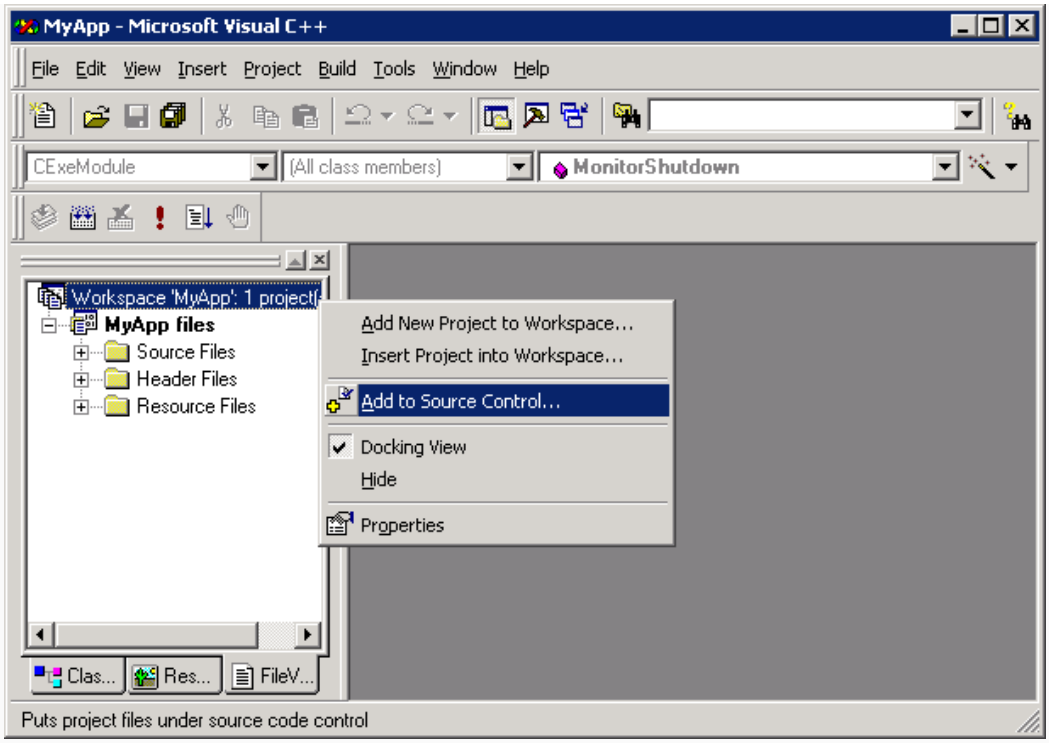
SourceSafe can be integrated into Visual C++ 6.0 to source control VC projects and files.

To integrate SourceSafe with VC 6.0, we can do as follows:

1. Choose SourceSafe as the current source control provider.
For information on how to do it, refer to: http://www.kevingao.net/sourcesafe/microsoft-source-code-control-interface-msscci-registry-entries.html
2. Add the VC project into source control of SourceSafe by clicking menu **Project** -> **Source Control** -> **Add to Source Control**. We can also add the project by right-clicking the file in the Project Explorer.

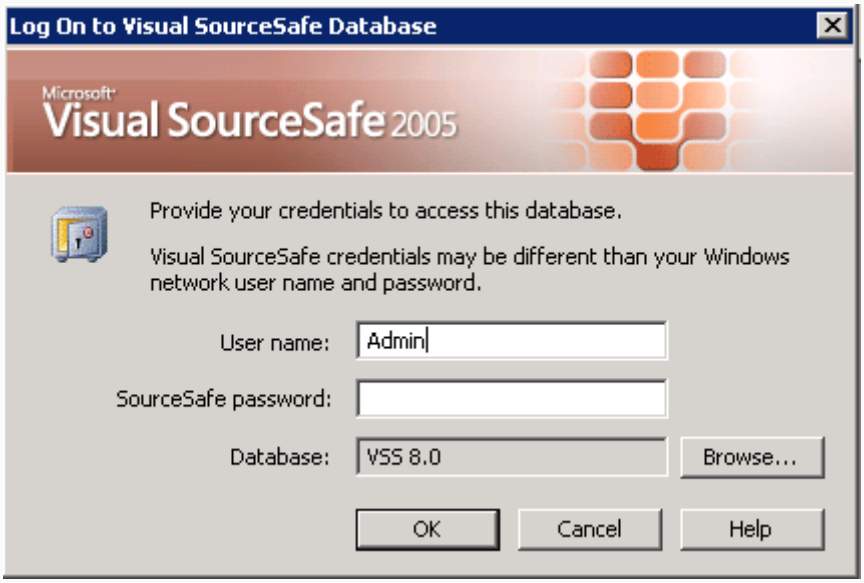


(Add VC project to SourceSafe from menu)



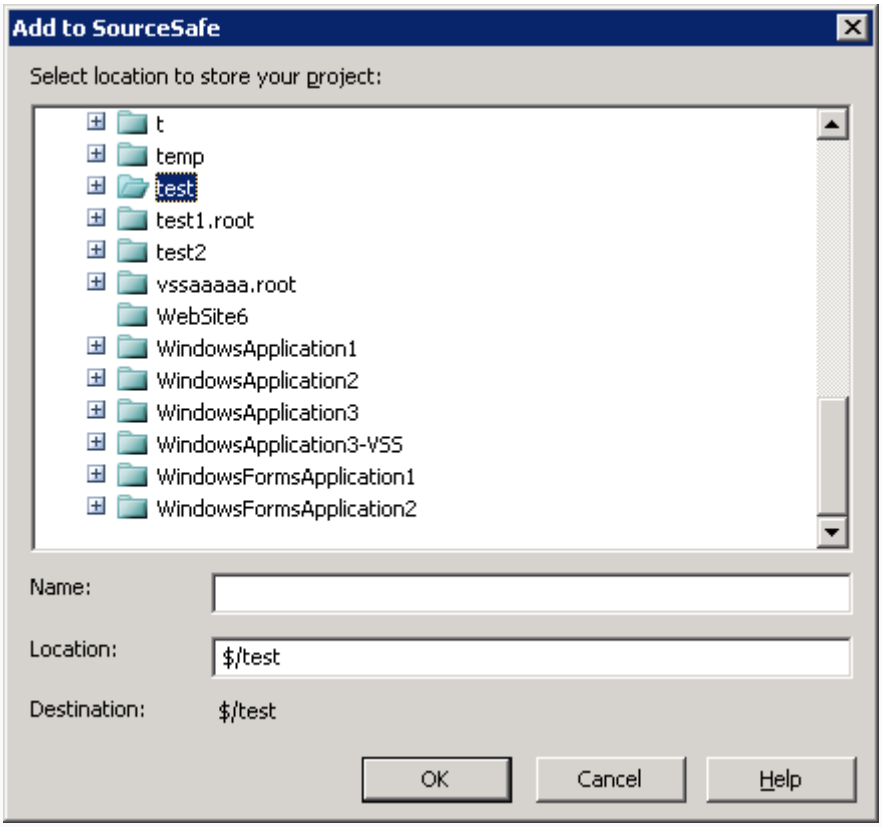
(Add VC project to SourceSafe)

3. In the following **Log On to SourceSafe Database** dialog box, enter the credentials to log on a VSS DB.



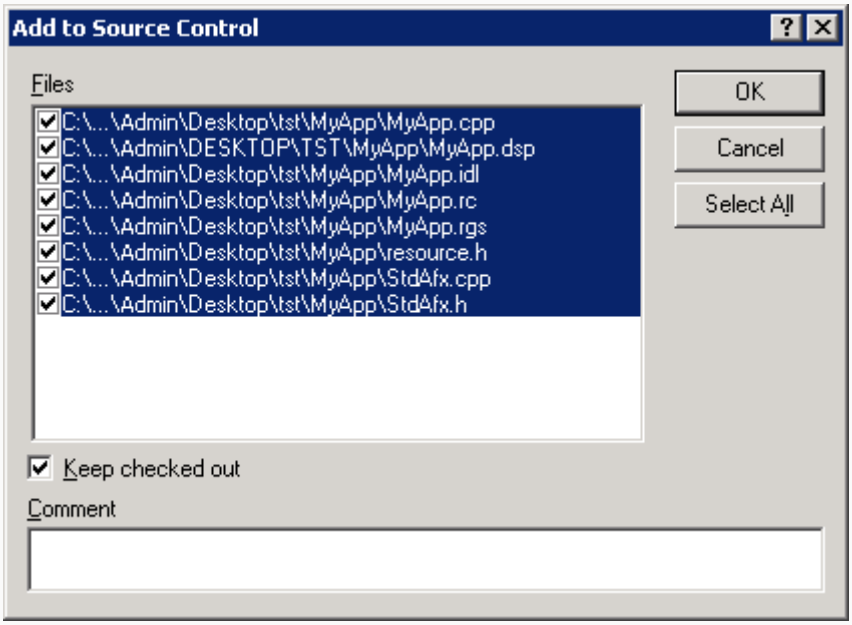
(Log on to VSS Database)

4. Choose a location in the VSS project tree to store the VC project.



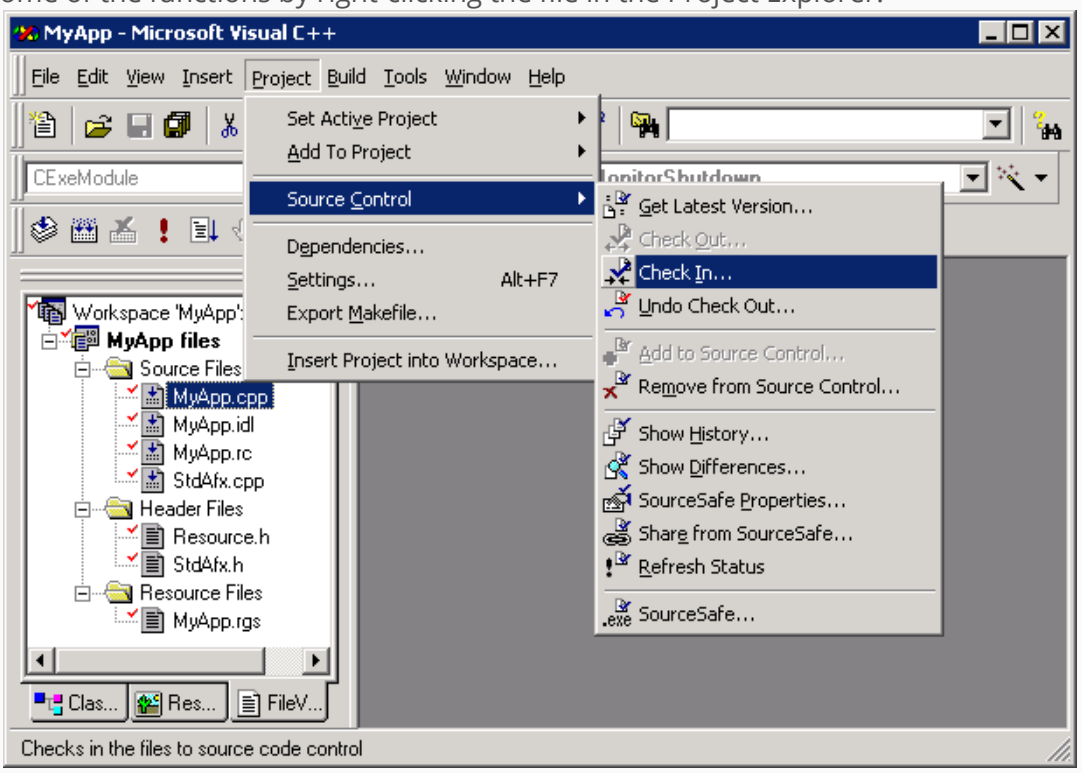
(Choose location to place the VC project)

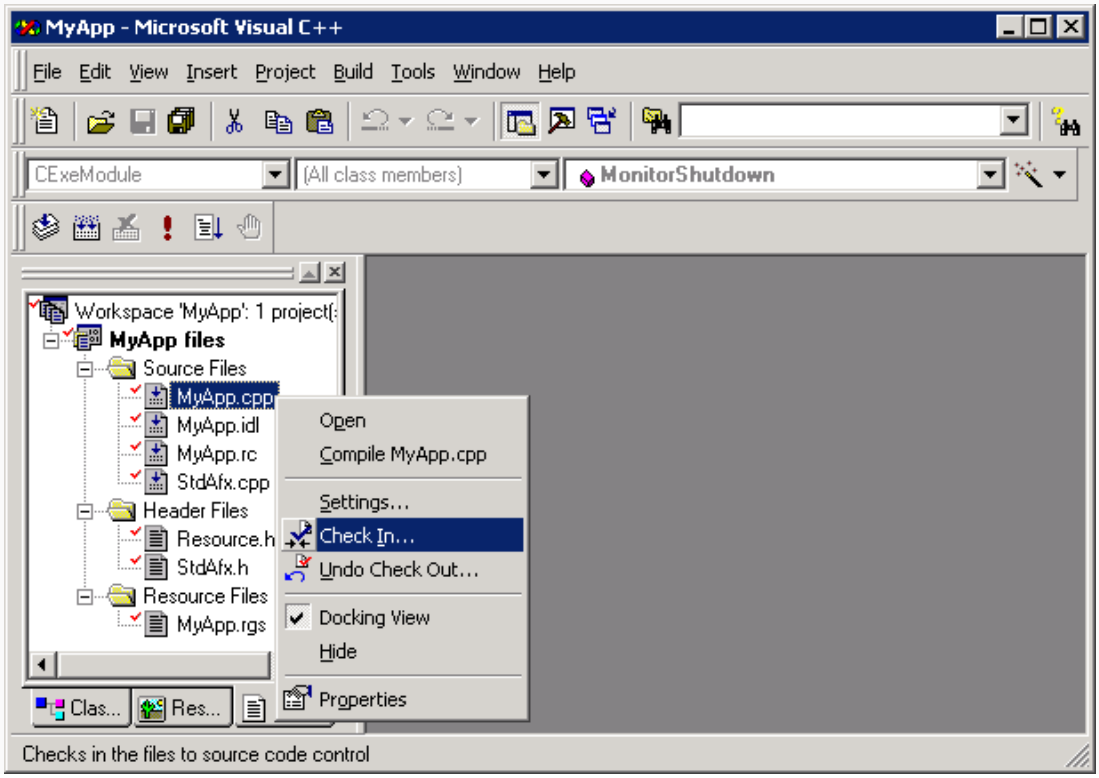
5. Select the files we want to add into SourceSafe for source control and click **OK**.



(Add VC files to SourceSafe)

6. Now, all of the files are under source control of SourceSafe. We can find the SourceSafe functions through menu **Project** -> **Source Control**. We can also access some of the functions by right-clicking the file in the Project Explorer.



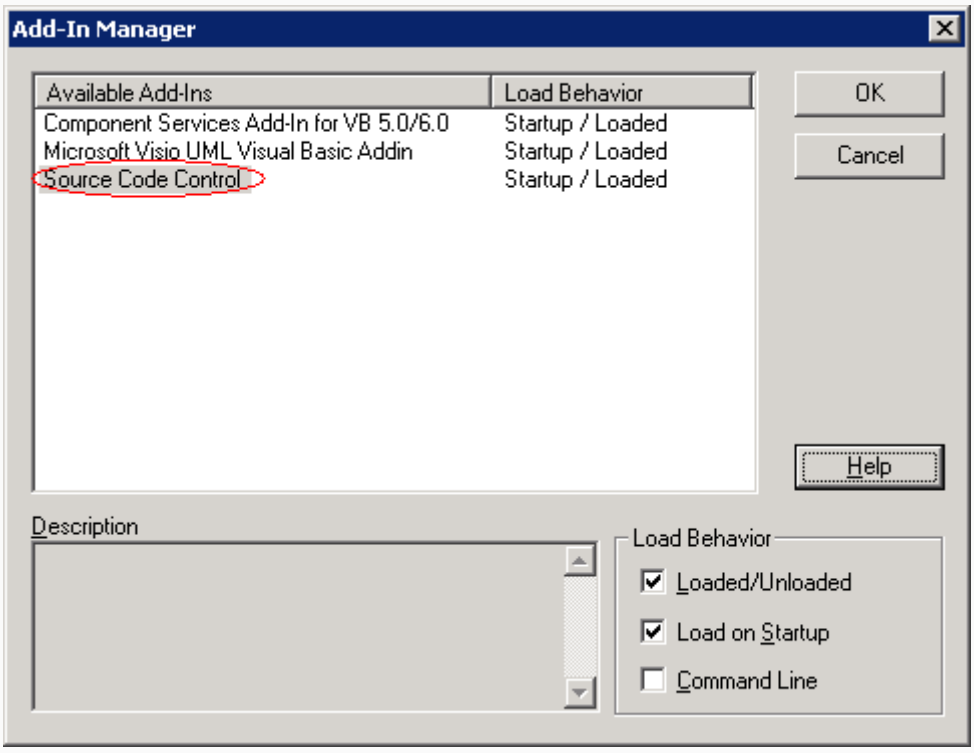


Integrating VSS with Visual Basic 6.0

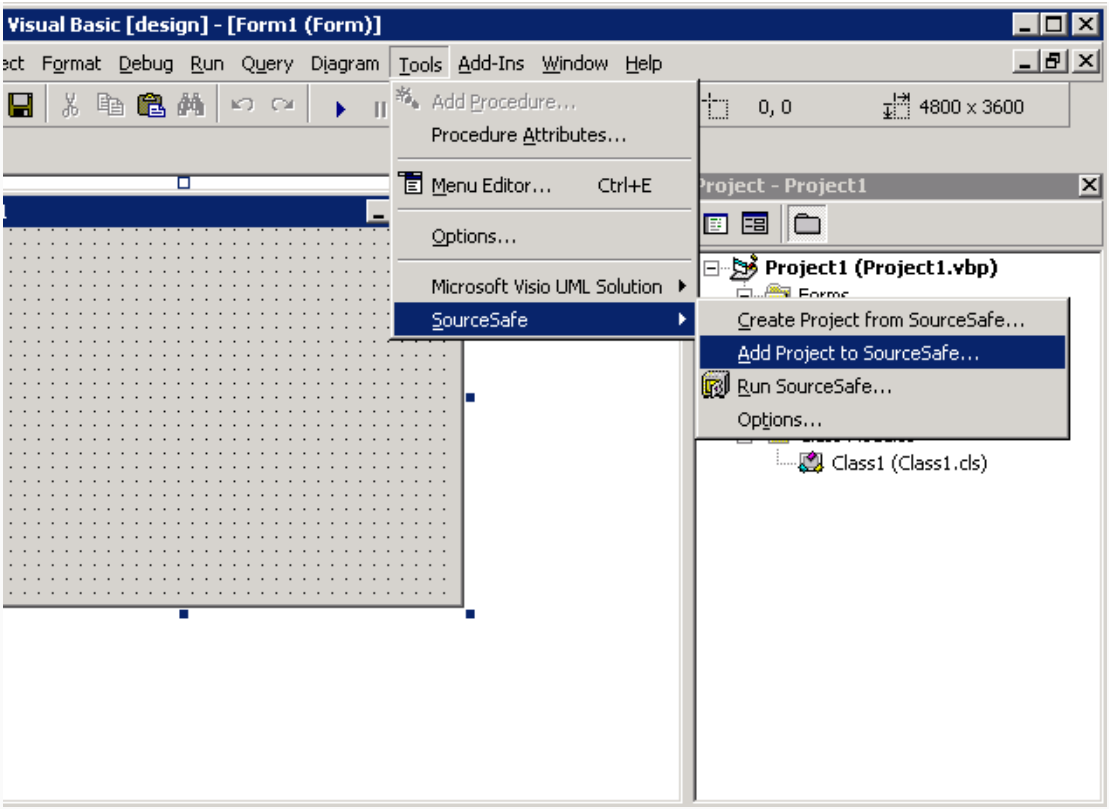
SourceSafe can be integrated into Visual Basic 6.0 to source control the VB forms, modules, class modules, etc.

To integrate SourceSafe with VB 6.0, we can do as follows:

1. Choose SourceSafe as the current source control provider.
For information on how to do it, refer to: http://www.kevingao.net/sourcesafe/microsoft-source-code-control-interface-msscci-registry-entries.html
2. Open VB 6.0 and check if the Source Code Control add-in is loaded through menu **Add-Ins** -> **Add-In Manager**.
If yes, we should be able to find the **SourceSafe** command under **Tools** menu.
If no, please edit the vbaddin.ini file by going to **Start** -> **Run**: vbaddin.ini and adding the line "vbscc=3 " in the file.

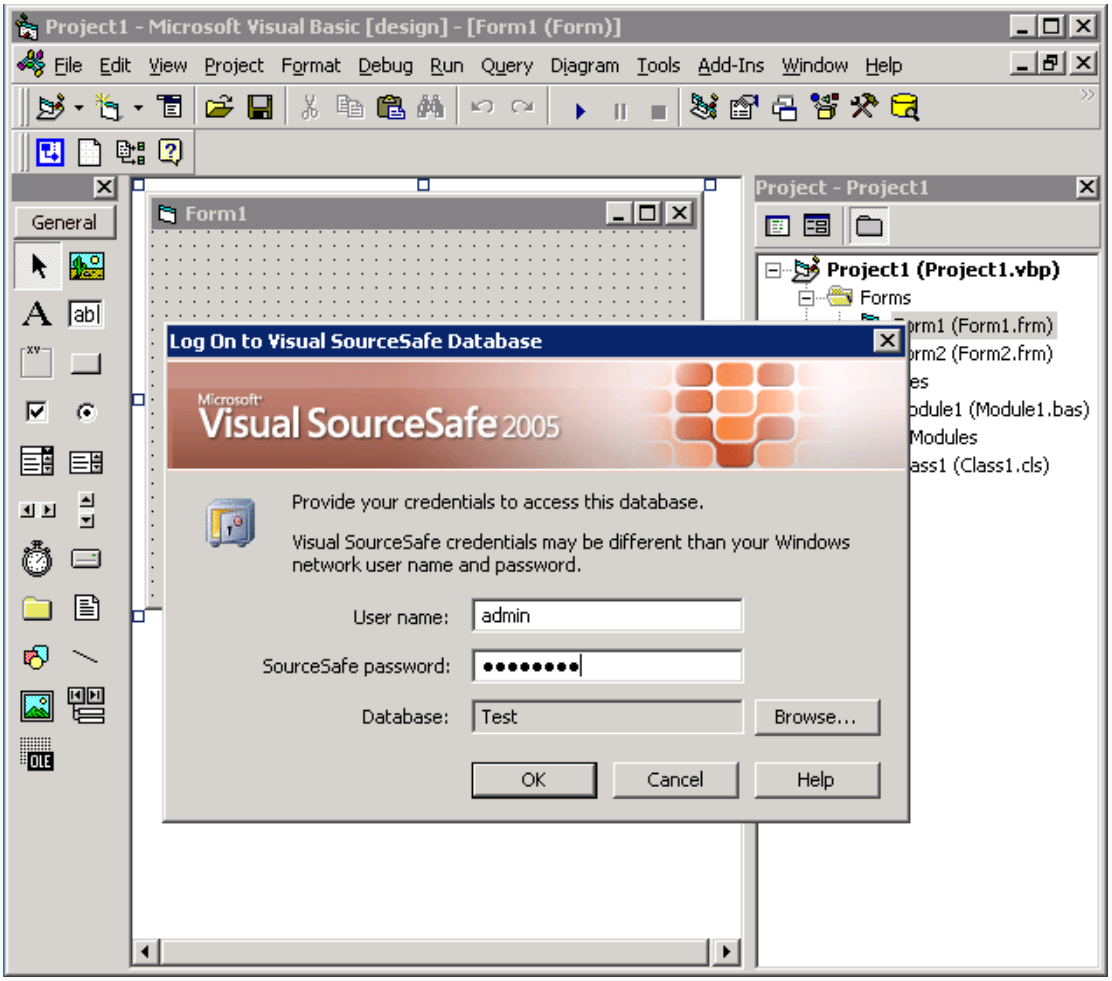


3. Add the VB project into source control of SourceSafe by clicking menu **Tools** ->**SourceSafe** -> **Add Project to SourceSafe**.



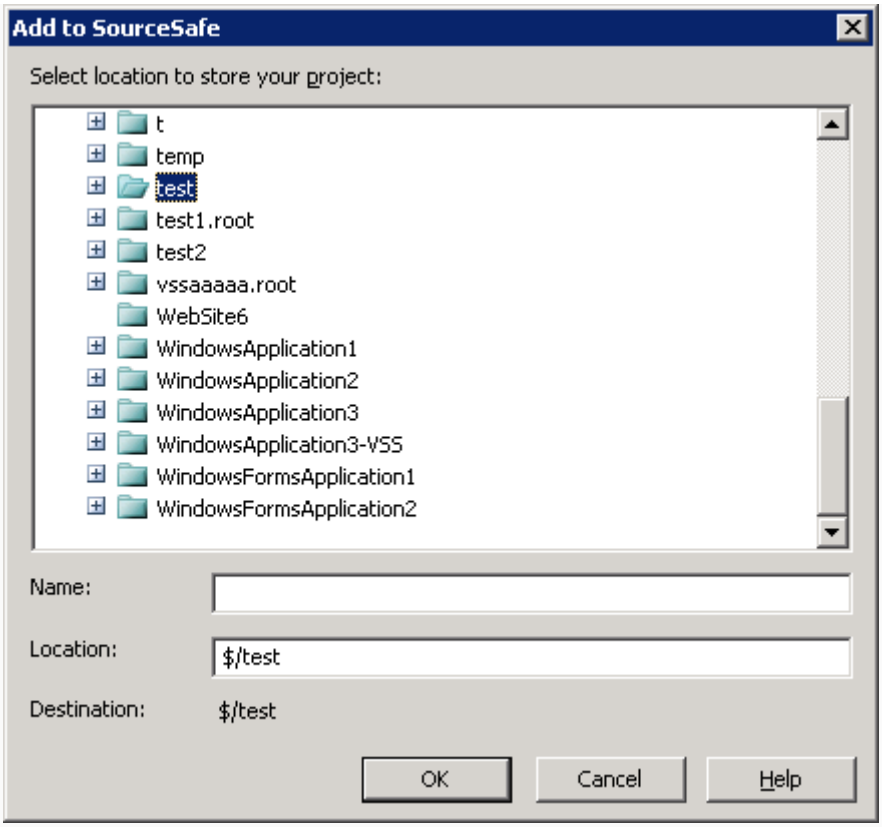
(Add VB Project to SourceSafe)

4. In the following **Log On to SourceSafe Database** dialog box, enter the credentials to log on a VSS DB.



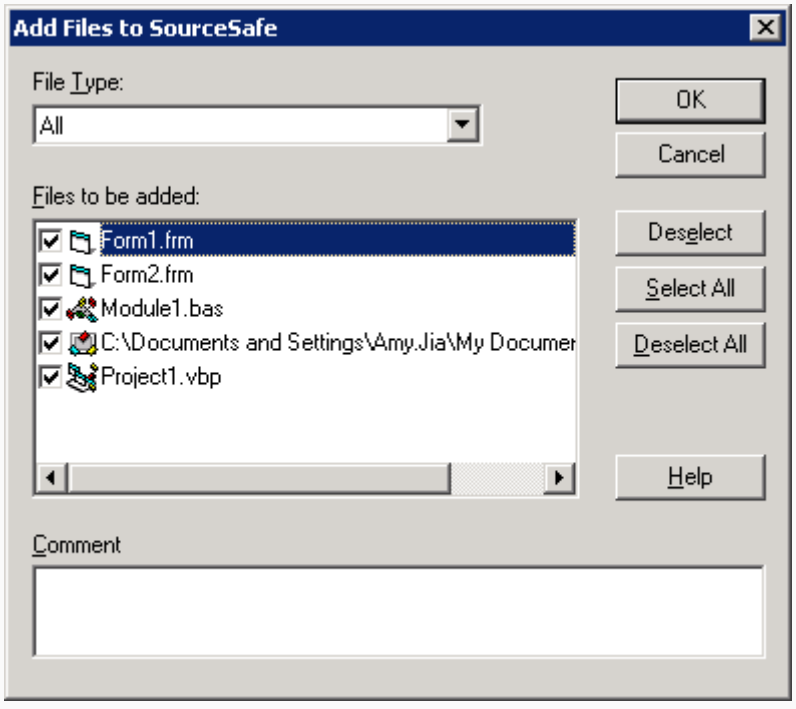
(Log on to VSS Database)

5. Choose a location in the VSS project tree to store the VB project.



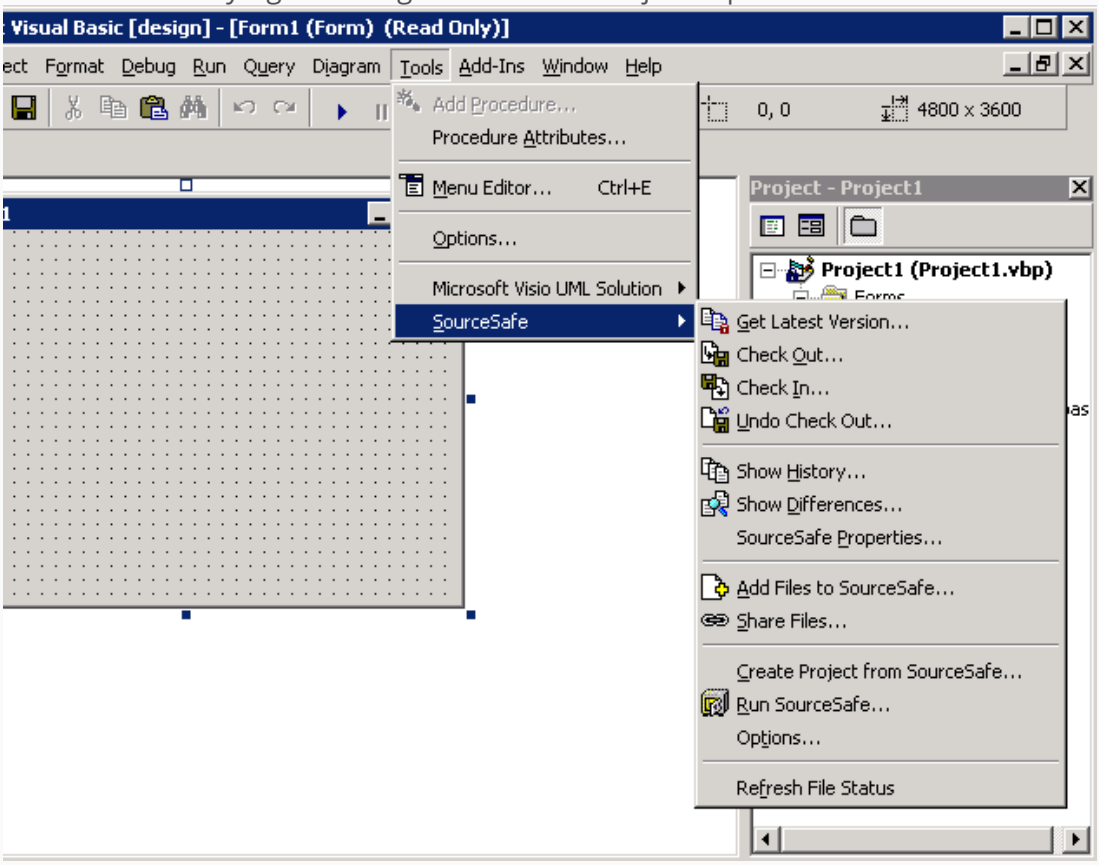
(Choose location to place the VB project)

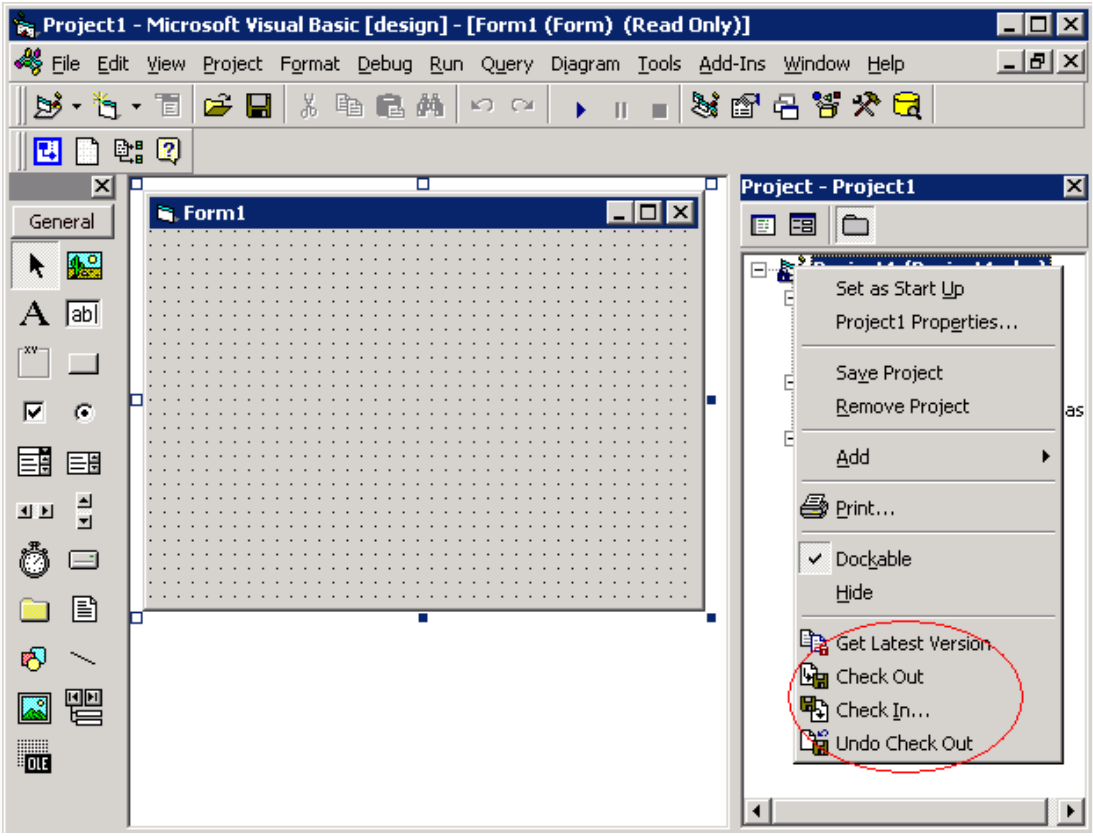
6. Select the files we want to add into SourceSafe for source control and click **OK**.



(Add VB files to SourceSafe)

7. Now, all of the files are under source control of SourceSafe. We can find the SourceSafe functions through menu **Tools** -> **SourceSafe**. We can also access some of the functions by right-clicking the file in the Project Explorer.



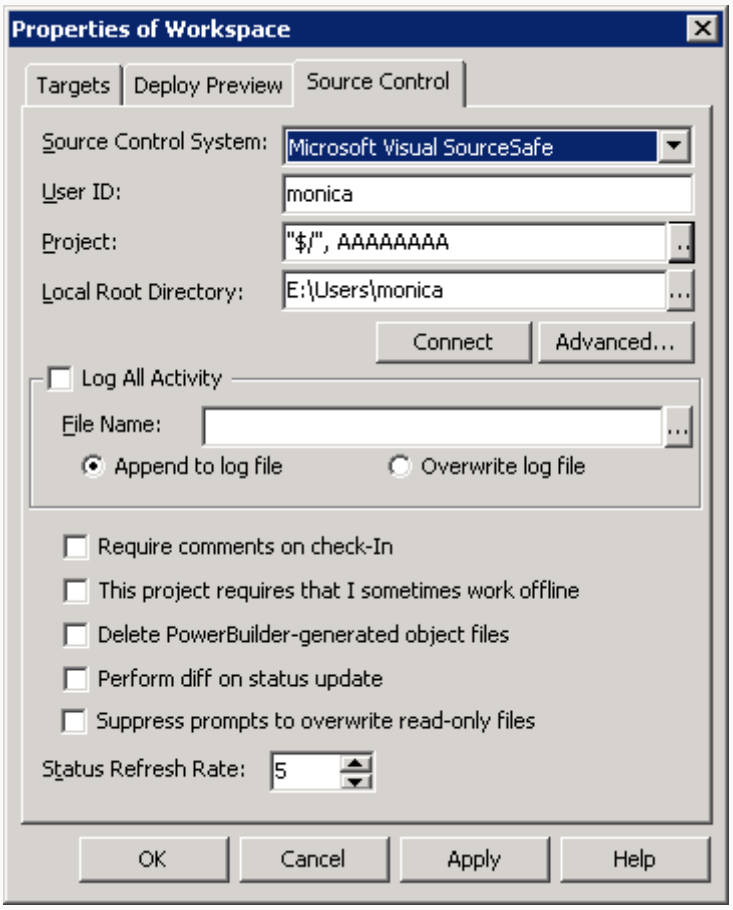

Project1 - Microsoft Visual Basic [design] - [Form1 (Form) (Read Only)]
File Edit View Project Format Debug Run Query Diagram Tools Add-Ins Window Help
General
Form1
Project - Project1
Set as Start Up
Project1 Properties...
Save Project
Remove Project
Add
Print...
Dockable
Hide
Get Latest Version
Check Out
Check In...
Undo Check Out

Integrating VSS with PowerBuilder

PowerBuilder was a popular tool for database front end development. I used PowerBuilder 6 about 10 years ago to develop a MIS (Management Information System) application. The tool was powerful but in the past 5 years, PowerBuilder lost its ground to Java, .NET and other web development languages (like PHP). Recent market surveys show that PowerBuilder is not in the top 5 anymore.

The source code control interface of PowerBuilder is compatible with Microsoft's MSSCCI, so we can use Microsoft Visual SourceSafe (VSS) or other compatible software as the version control tool.
To integrate Visual SourceSafe (VSS) with PowerBuilder, we can follow the steps below:

1. Right-click the selected workspace, and select **Properties**, as seen in the following screen shot:



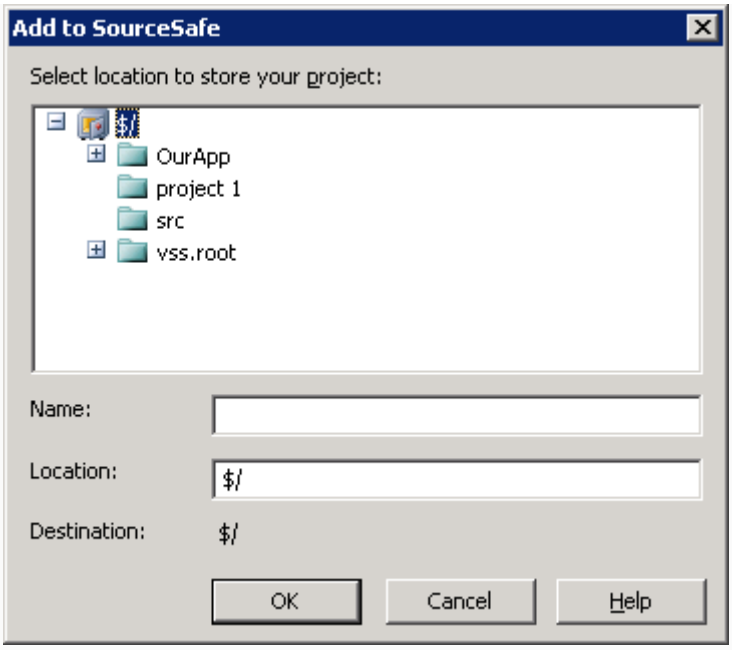
(Properties of Workspace)

2. In Source Control tab, configure source control settings.

- In the **Source Control System** dropdown list, select **Microsoft Visual SourceSafe**.
- In **User ID** edit box, input the name of our VSS account.
- In **Project** edit box, input the VSS project in which we want put the selected local project. We can click the browse button next to choose the VSS project. When we click the browse button, the **Log On to Visual SourceSafe Database** window will prompt, as seen in the following screen shot:

(Log on to Visual SourceSafe Database)

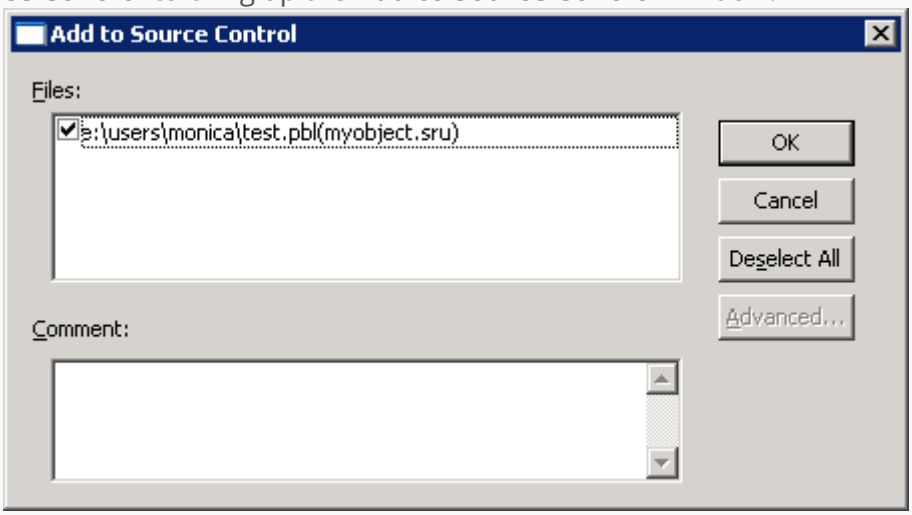
- Log into the VSS database, and select a project to store the selected local project in the following screen shot.



(Add to SourceSafe)

- Click **OK** to finish the workspace binding.

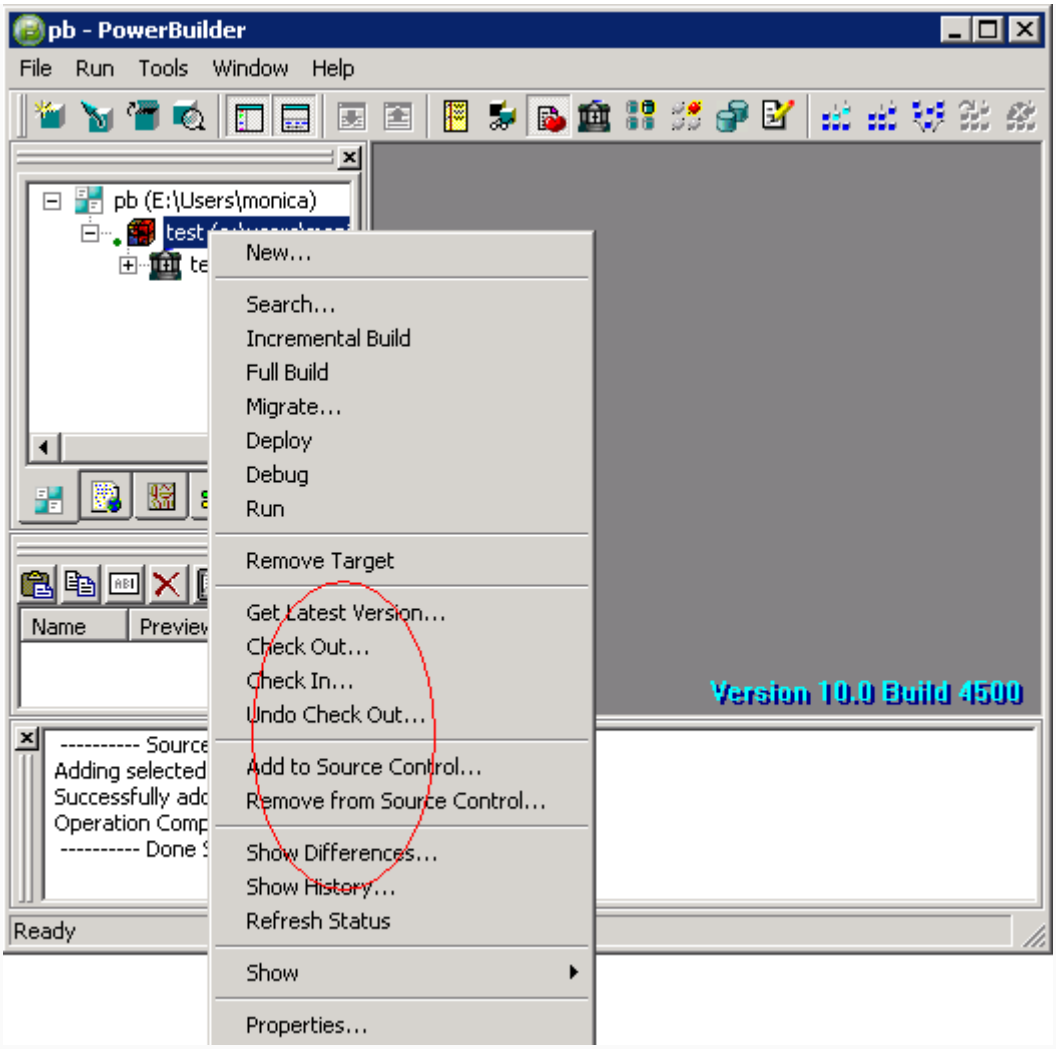
3. In **PowerBuilder System Tree**, right-click the selected workspace, and select **Add to Source Control** to bring up the **Add to Source Control** window.



(Add to Source Control)

4. Click **OK** to add the selected items to the VSS database.

5. Now the target project is source controlled by VSS. We can perform the basic source control operations, such as Get Latest Version, Check Out, Check In, Undo Check Out, Add to Source Control, Show Differences, Show History and so on.



(Basic Source Control Operations)

How to backup & restore VSS DB

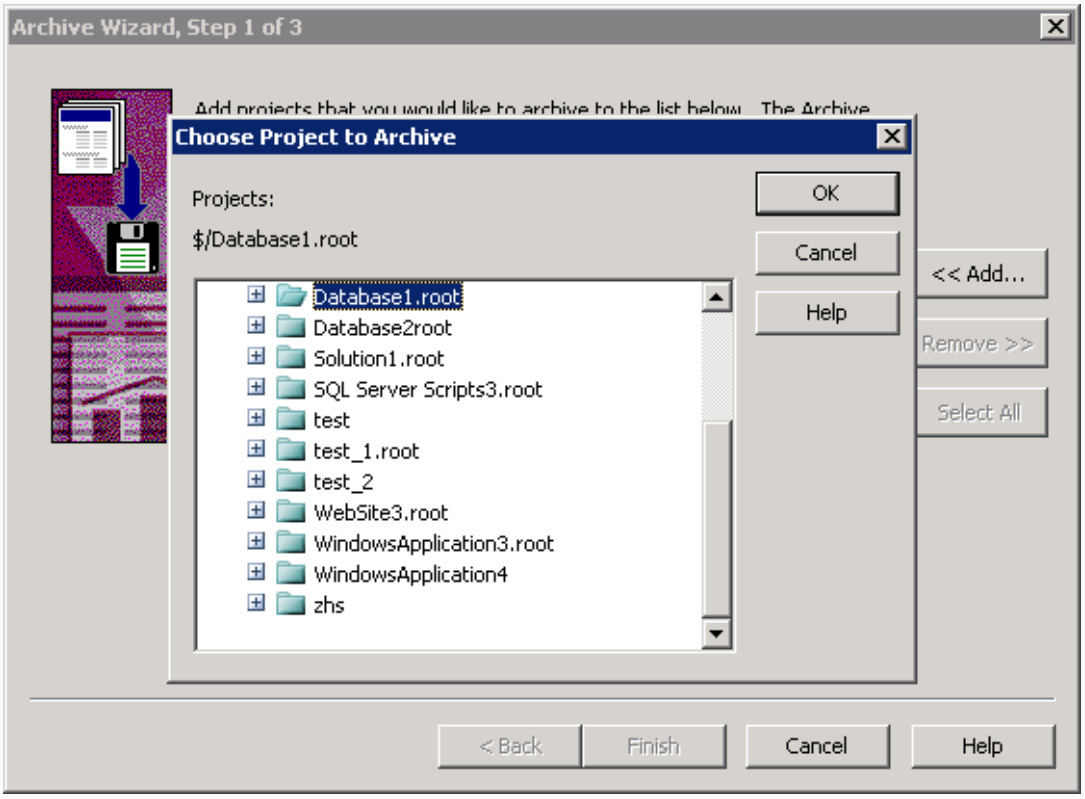
SourceSafe provides an Archive utility, with which we can periodically backup our VSS Database or projects and transport files/projects between SourceSafe databases. SourceSafe also provides a Restore tool which allows us to restore the data from an archive.

- How to archive SourceSafe database
- How to restore SourceSafe database

To archive a SourceSafe database:

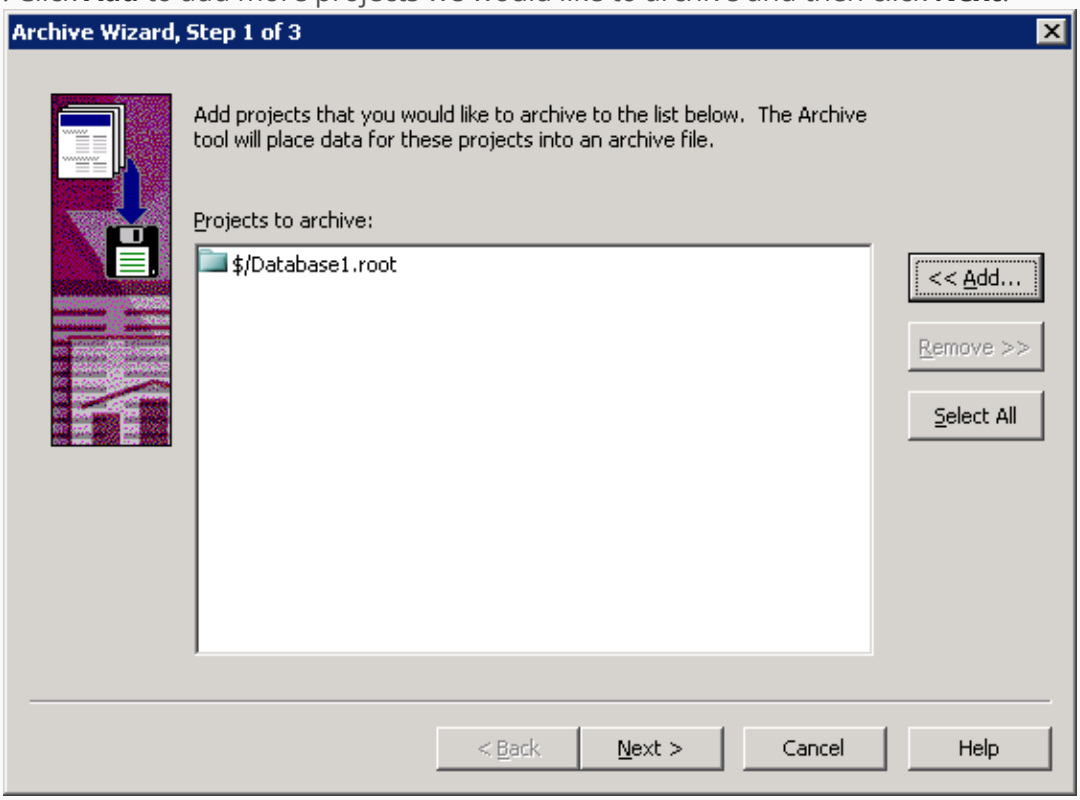
1. First, make sure no one is using the database we are going to archive or the Analyze utility will not run during the process of archive.

2. Open the database in SourceSafe Administrator and start the Archive Wizard through menu **Archive** -> **Archive Projects**.
3. Choose the project to archive from the project list.



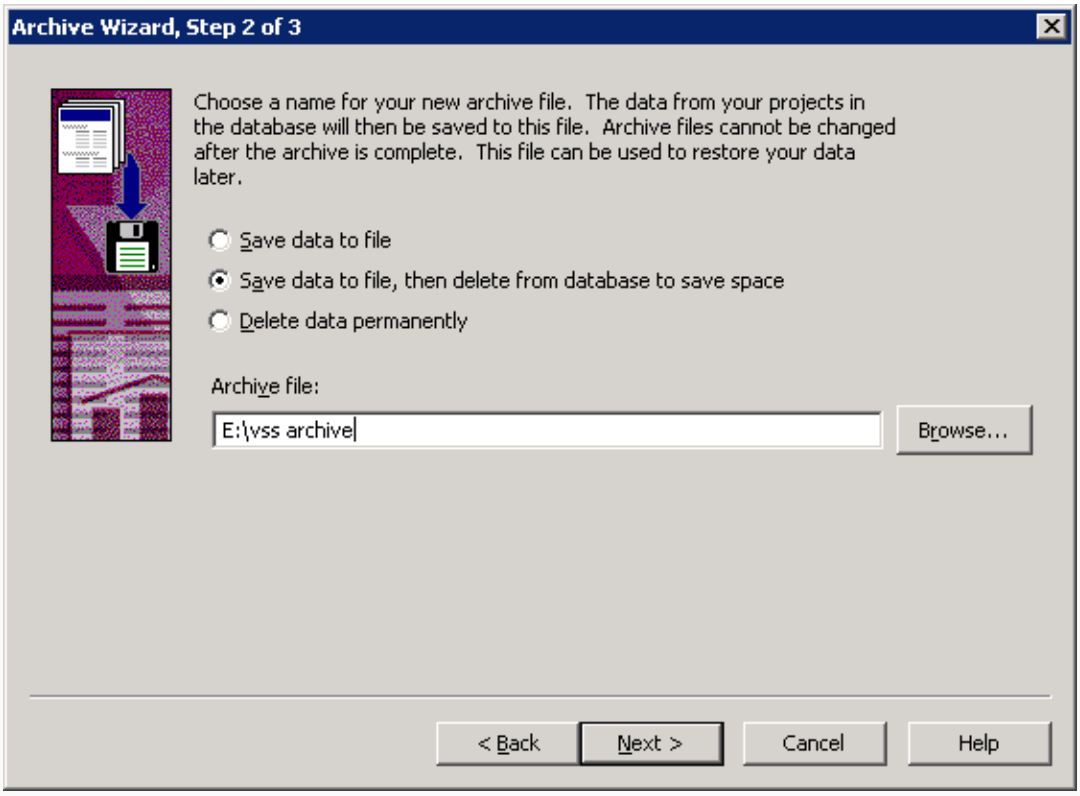
(Choose project to archive)

4. Click **Add** to add more projects we would like to archive and then click **Next**.



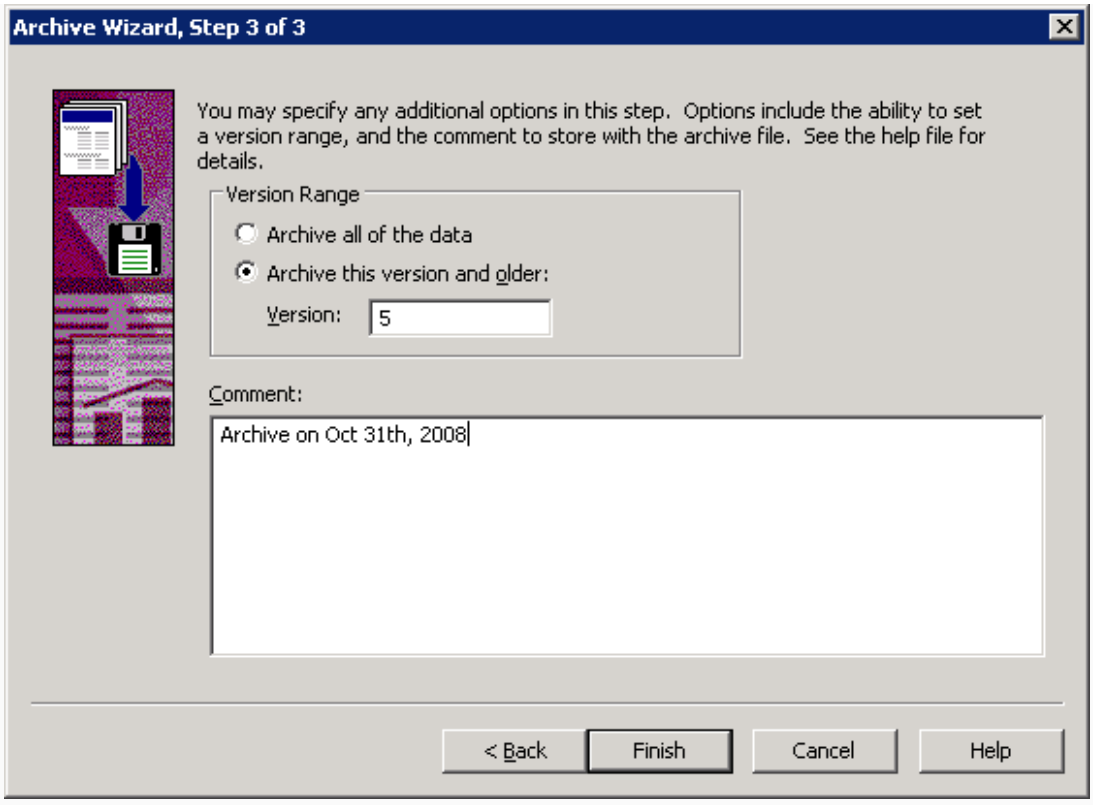
(Add projects you would like to archive)

5. Specify a name for the archive file.



(Specify name for the archive file)

6. Specify the version range of the project to archive. We may choose to archive all of the data or archive the data older than a specific version.

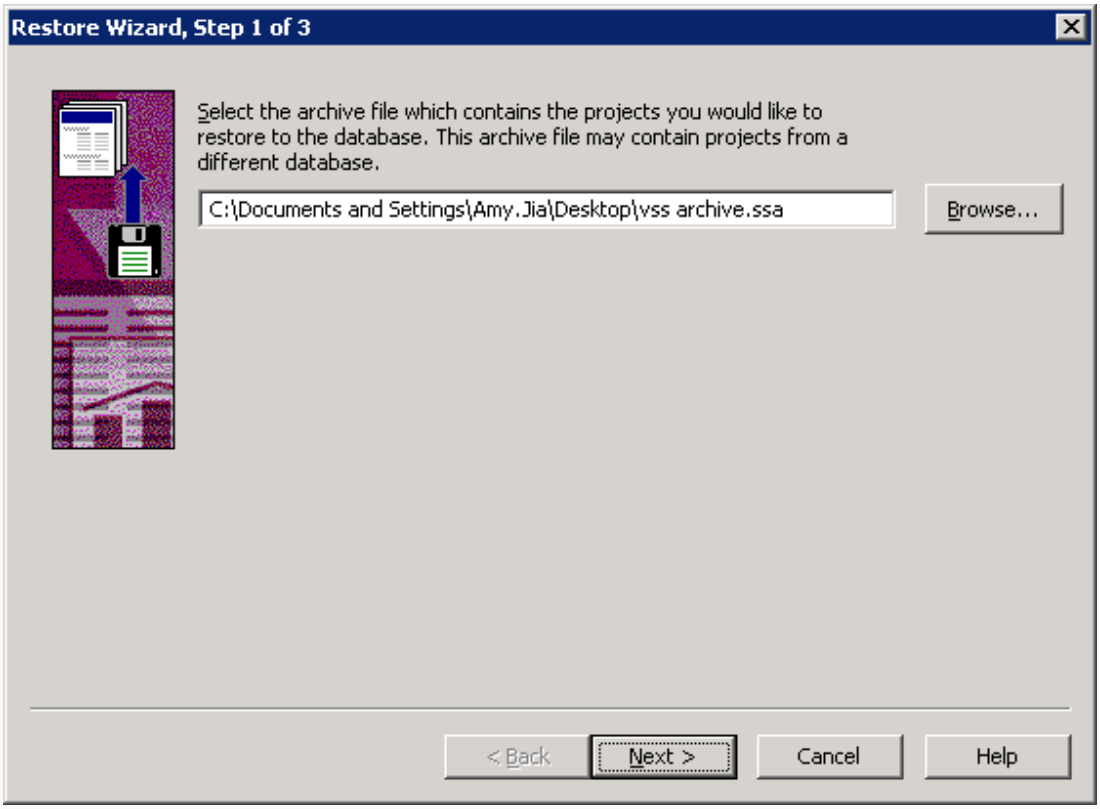


(Specify the version range to archive)

7. Click **Finish** and SourceSafe will archive the projects to an .ssa file.

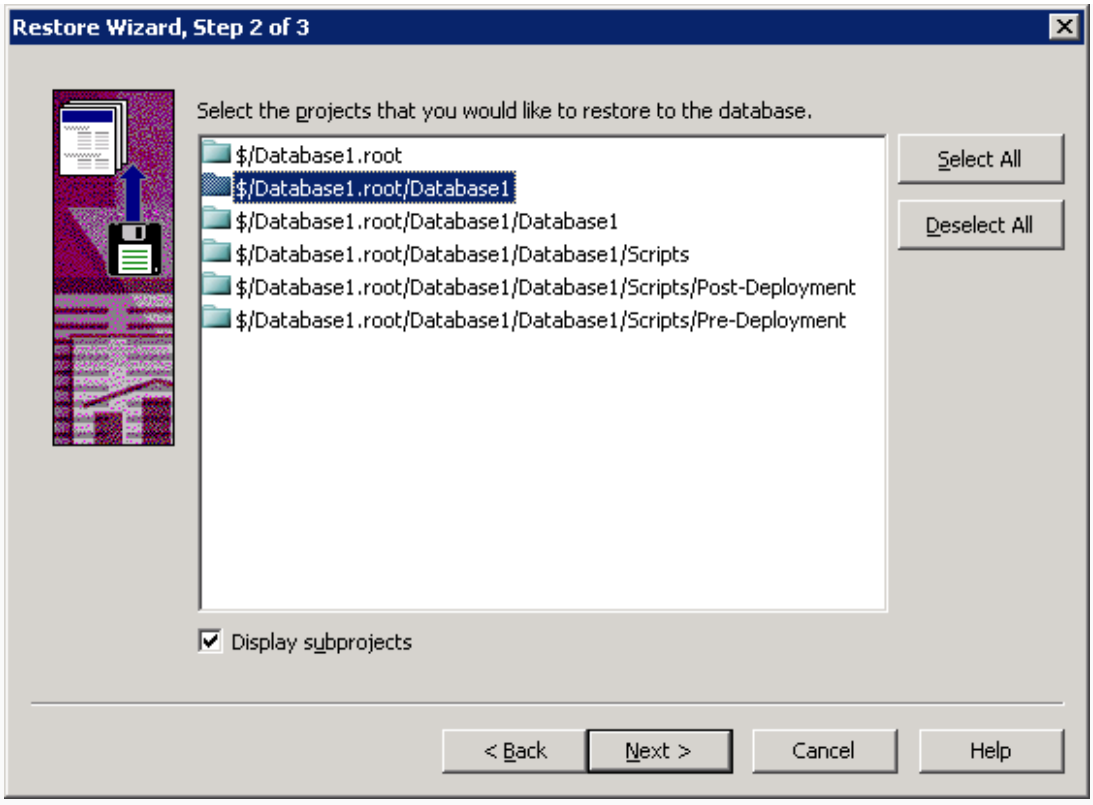
To restore the projects from an archive file:

1. Start the Restore Wizard through SourceSafe Administrator menu **Archive** -> **Restore Projects**.
2. Select the archive file that contains the projects we would like to restore.



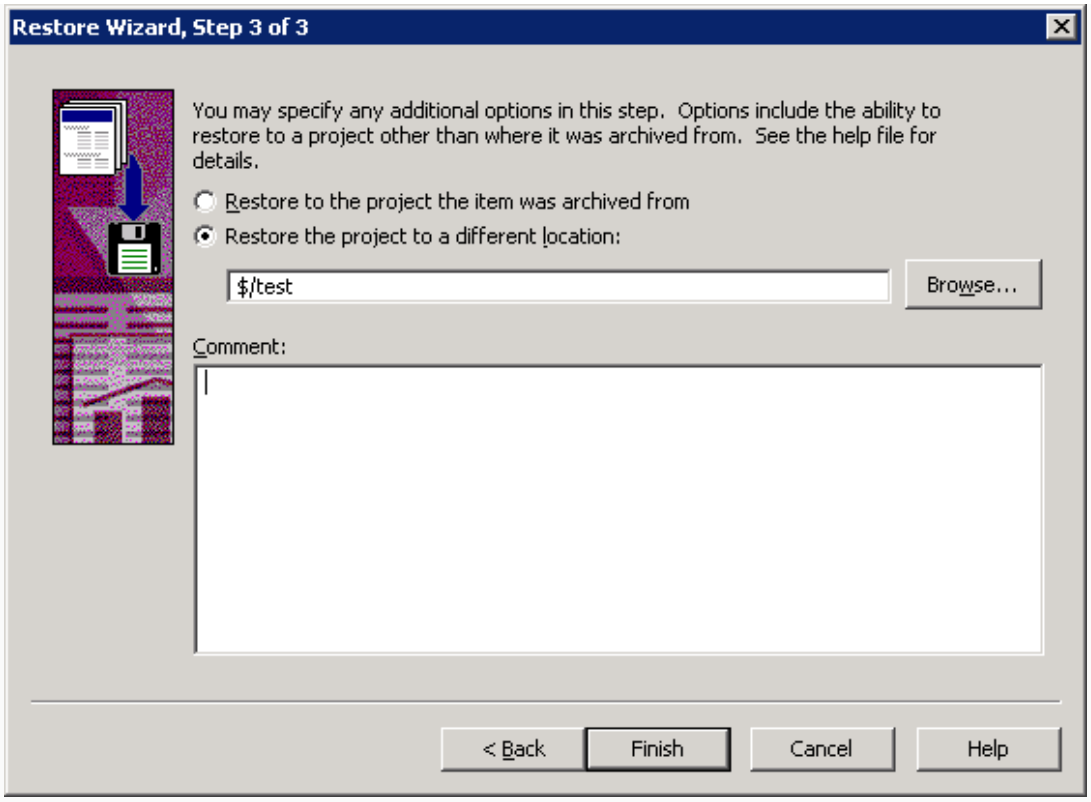
(Select archive file)

3. Select the projects we want to restore to the database. We can check the **Display subproject** option to see all the subprojects.



(Select projects to restore)

4. Specify the destination to restore the project. We may restore the project to where it was archived from or to a different location.



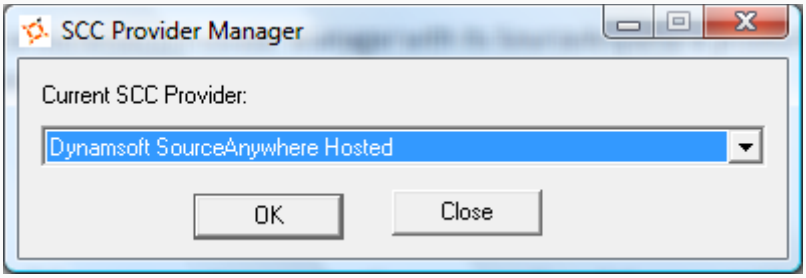
(Select restore destination)

5. After the restore operation is finish, we may log in SourceSafe to check if the projects we want have been properly restored.

Note: If we want to restore a complete backup of a VSS DB, it is recommended that we restore the backup to a new VSS database rather than an existing database.

A Free Tool to Manage the MSSCCI Provider

Dynamsoft provides a tool called **SCC Provider Manager**with its SourceAnywhere source control product family. With this tool, you can choose one of the SCC providers in your system as the default provider. This is a screen shot of the tool:



(SCC Provider Manager)

With the permission of the company, I am posting the source code and the executable file here. The project was developed with Visual C++ 2003. **Download SCC Provider Manager**.

Pin

Pin is a small but sometimes helpful feature in Visual SourceSafe (VSS). VSS defines **Pin** as a marker that designates a specific version of a file. For example, if we want our team members to get a historical version by default, we can pin the file to the specific version.

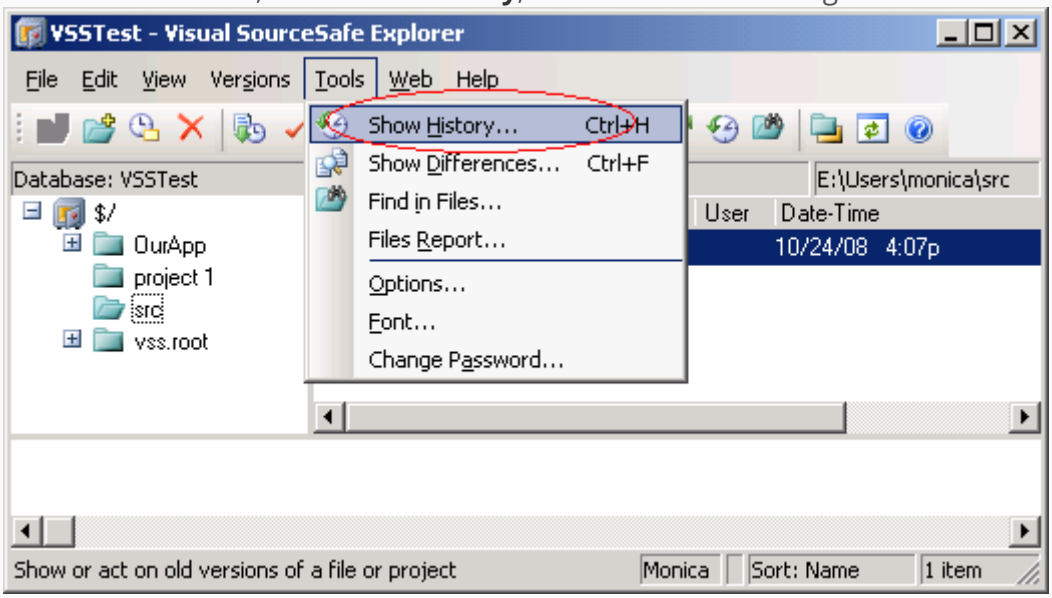
Pin is not a native feature in SourceSafe. I usually do not use **Pin** very often and my personal recommendation is that we should avoid **Pin** if possible.

Pin has the following features:

- When we do **Get Latest**, for a pinned file, VSS retrieves the pinned version, not the latest version.
- We can still get the other versions in the **Show History** dialog box.
- After a file is pinned, the file cannot be modified. We cannot do **Check In** / **Check Out** on a pinned file.

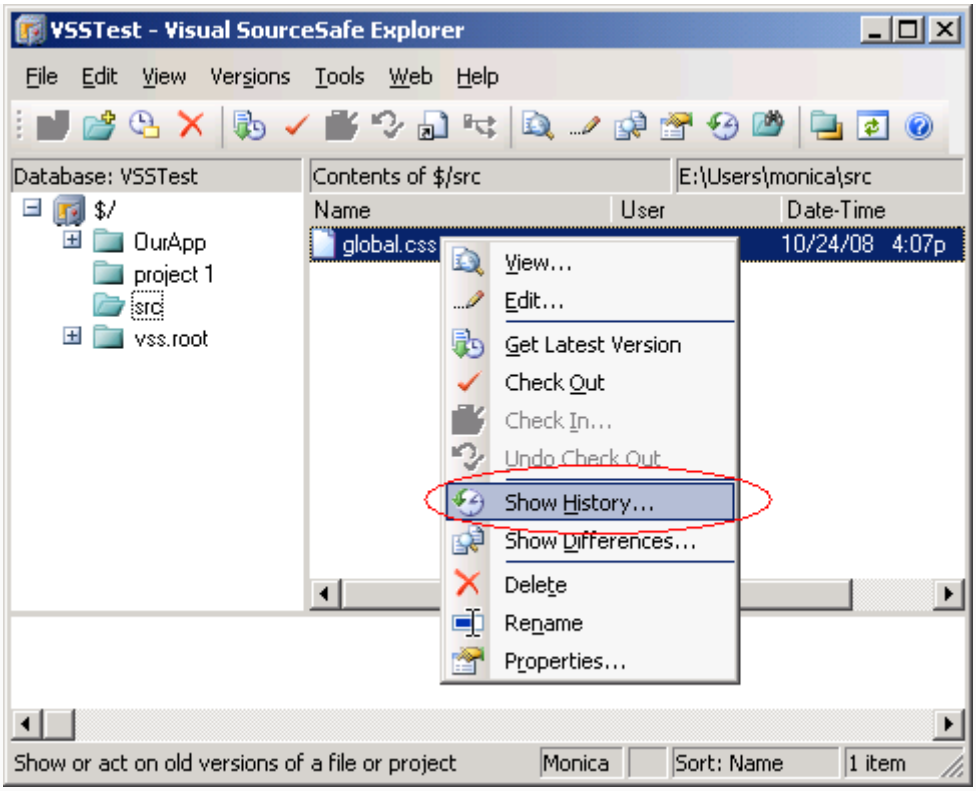
To pin a version of a file, please follow steps below:

1. In **Visual SourceSafe (VSS) Explorer**, select the file that you want to pin.
2. On the menu **Tools**, click **Show History**, as seen in the following screen shot:



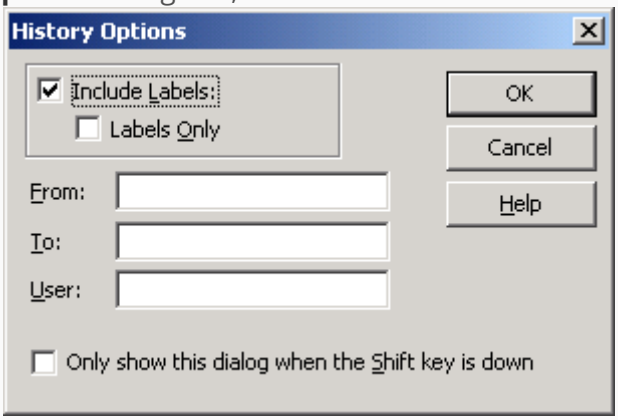
(Show History)

Or right-click the file that you want to pin, and select **Show History**.



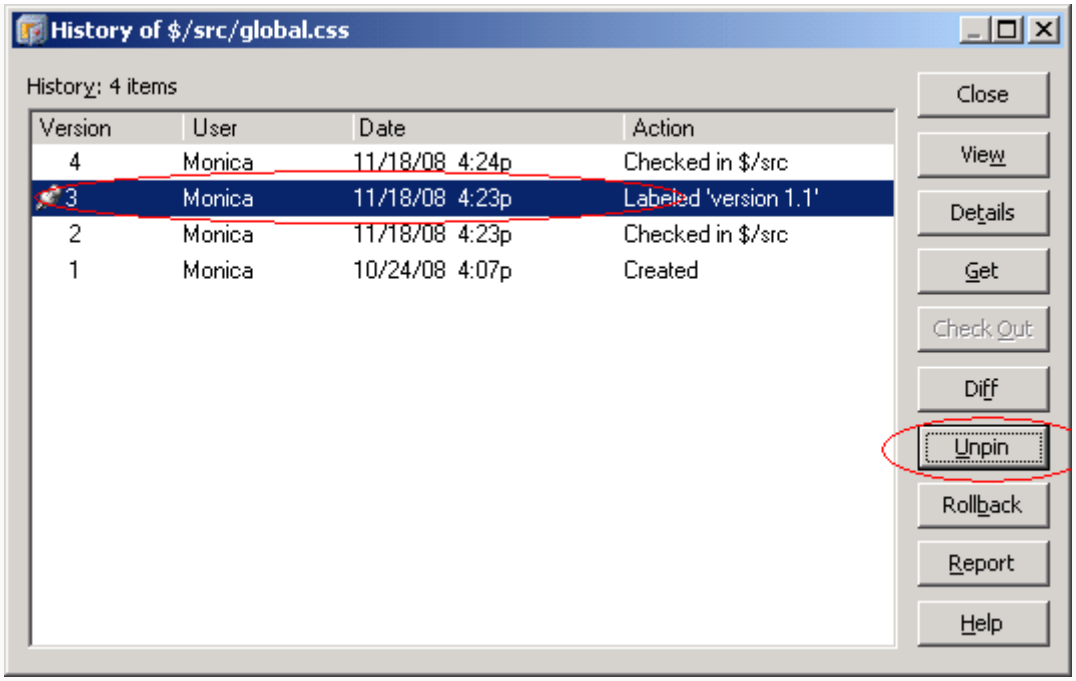
(Show History)

3. In the **History Options** dialog box, click **OK**.



(History Options)

4. In the **History of** dialog box, select the version that you want to pin, and then click **Pin**. A pushpin icon will appear next to the pinned file.


History of $/src/global.css
History: 4 items
Version
User
Date
Action
4
Monica
11/18/08 4:24p
Checked in $/src
3
Monica
11/18/08 4:23p
Labeled 'version 1.1'
2
Monica
11/18/08 4:23p
Checked in $/src
1
Monica
10/24/08 4:07p
Created
Close
View
Details
Get
Check Out
Diff
Unpin
Rollback
Report
Help

(Pin)

Show History

Show History Basics

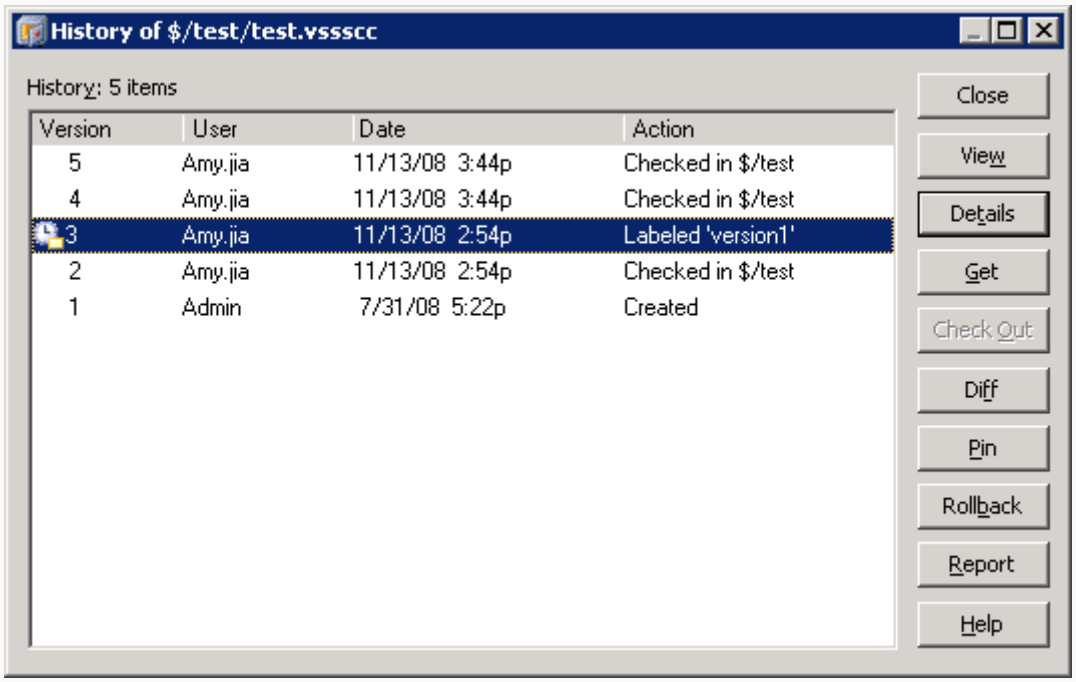
Show History is one of the most important features in SourceSafe. My personal feeling is that being able to go back to the pervious versions is the main purpose that software development teams use version control tools. It gives us peace of mind when we implement new features and fix bugs.
The Show History command in SourceSafe allows us to view the history information of a file/project by listing all the versions of an item from the latest version to the creation of the item. In the History Explorer, we can see the version number of the item, the user who performed the action, the date/time of the event and the action. We can also perform operations like Get, View, Pin on a specific version of the item and rollback a file to an old version.

Many developers only use the History Explorer to view a previous version or do diff and may not know many useful features of the History Explorer. I am listing some of the features in the following section and hope you will find it useful.

How to view the history of an item

To view history of a file/project, we can click **Show History** under the **Tools** menu or from the right-click menu of the item, set history options in the following dialog box and then the history explorer will appear listing all the historical information of the item.

History Explorer



(File History Explorer)

Get an old version of a file/project

Sometimes we may want to retrieve an old version of a file or project. We can do that through history explorer. Select the version of the file/project we would like to retrieve and click **Get**.

Get a version of a file/project by label

Label is a good way to manage version release/builds. For more information, see Label. VSS also provides the feature to get an item by label. If we check **Include Labels** option in the History Options dialog box, we will see all the labels that have been assigned to the item in the **Action** column of history explorer. Simply selecting the labeled version and clicking **Get** will get the labeled version to the local drive.

Diff two versions of a file

In history explorer, we can also compare two versions of a file. To do that, we can select two versions of the file and click **Diff**. For more information, see File Diff.

Pin an old version of a file

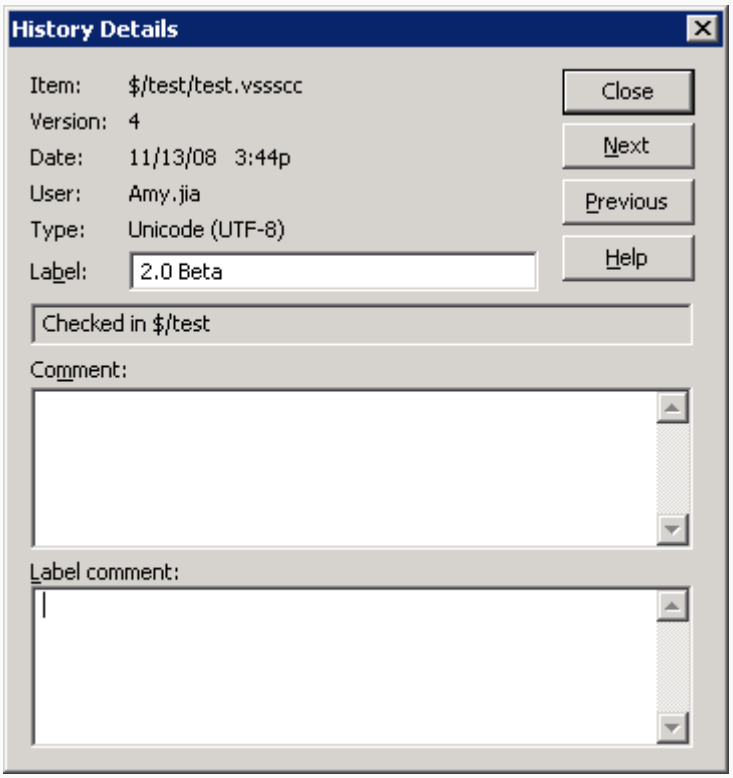
If we want our team members to get a specific historical version of a file by default, we can pin the file to that version by selecting the version in the history explorer, and clicking **Pin**. For more information, see Pin.

Rollback to an old version of a file

We can use the Rollback feature to return a file to an old version and erase all the newer versions. If the file is shared among several projects, Rollback will only affect the current project. It breaks the file in the current project from that in the other projects. To rollback to an old version, we can select the version we want to rollback to and click **Rollback** in the history explorer.

Change label & comment

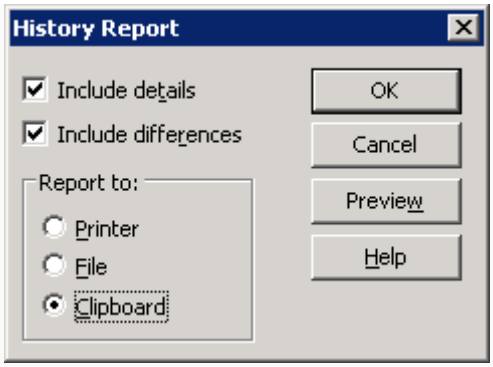
We may want to change the label/comment of an item in some situations. We can do it through history explorer too. Select the item version from the history explorer, click the **Details** button and then we can change label and comment in the **History Details** dialog box.



(History Details)

History report

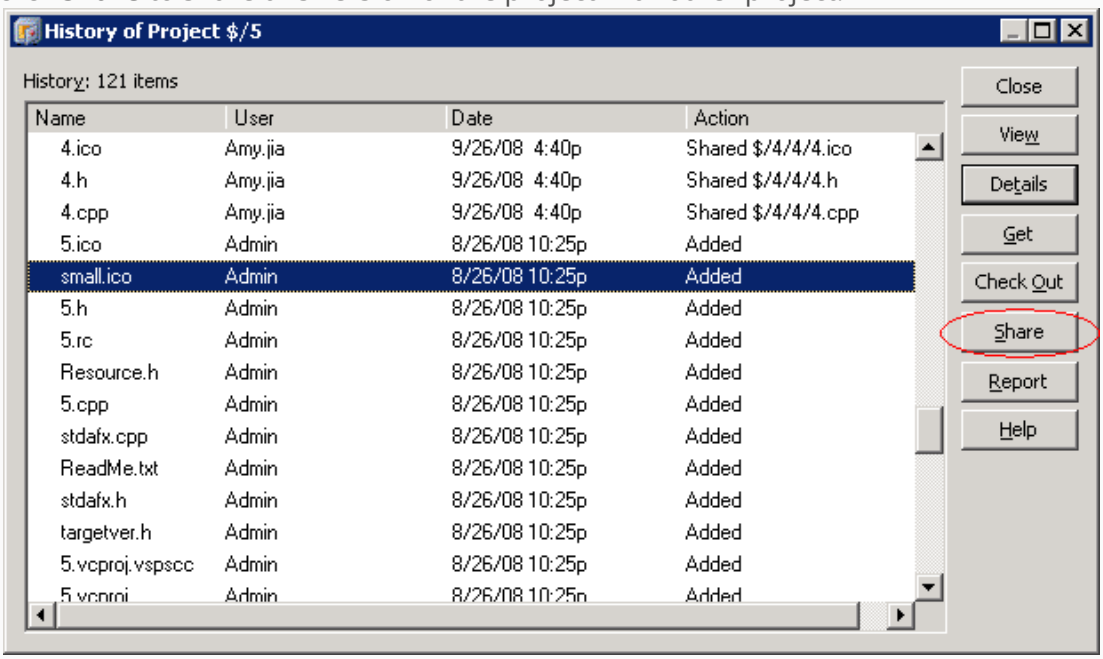
We can report the history information of an item to a printer, file or clipboard by clicking **Report** button in the history explorer. Checking **Include details** can include more detailed information, like comments in the report. Checking **Include differences** can include the differences between versions in the report.



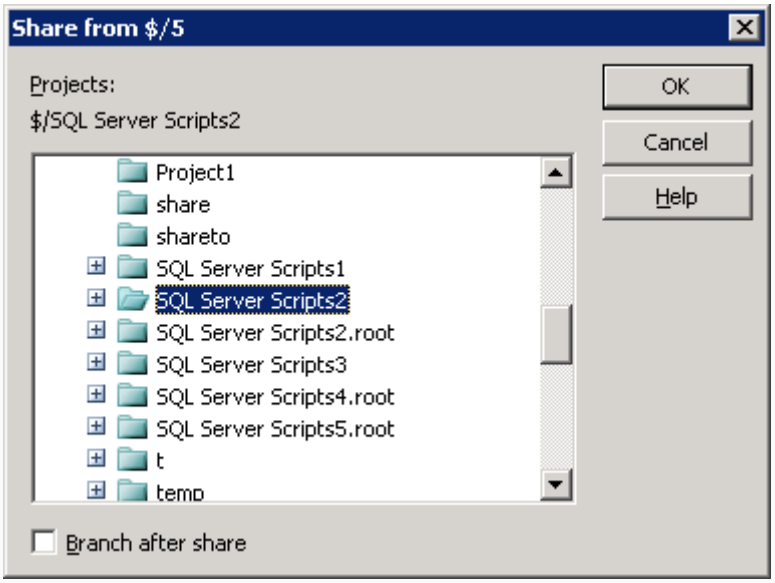
(History Report)

Share an old version of project

In the project history explorer, we can select a version of the project and click **Share** to share this version of the project with other project.



(Project History Explorer)



(History Share)

How to Manage Security

Introduction

SourceSafe provides a tool, **Visual SourceSafe Administrator**, to manage the permission of the VSS users.
However, designed for trusted environment, SourceSafe offers very low security. Regardless of the VSS project level permission, all VSS users must have read & write permission of the whole VSS folder from the file system. This means even for a VSS user who only has read permission of a single file in VSS database, he/she can copy or even delete the whole VSS database from the file system

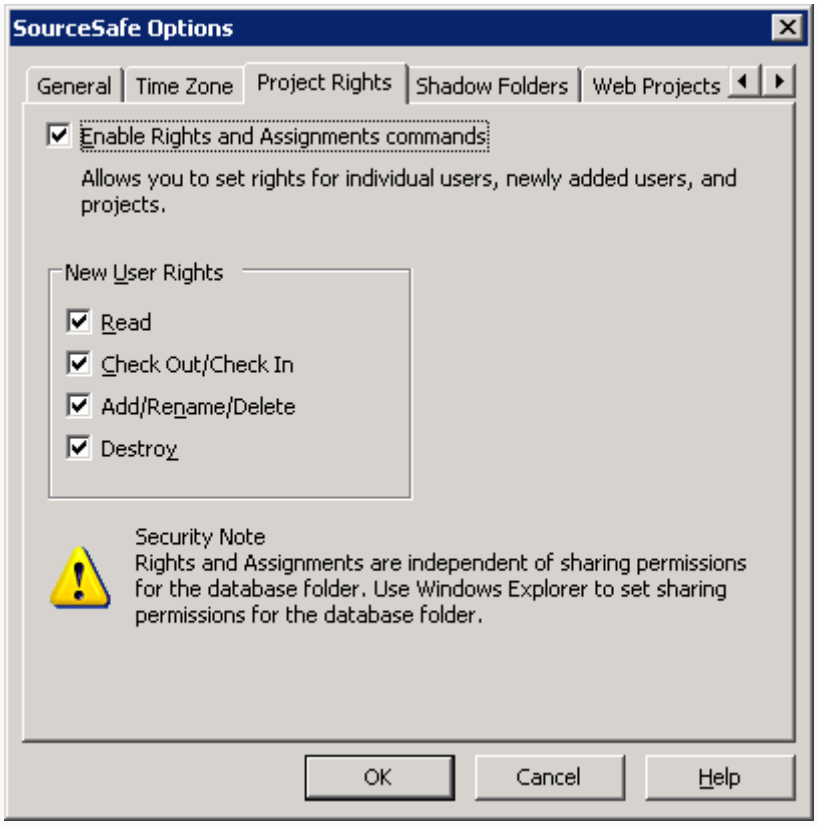
Furthermore, if we have remote SourceSafe users, we need to expose our whole VSS database folder from the file system level, which makes our source code vulnerable to outside hackers.

There is no easy way to solve this security vulnerability since VSS is designed that way. One possible option is to use an add-on tool to convert VSS from a file based system to a client/server architecture based system. A tool I developed, called **SourceAnywhere for VSS**, can do this job. The link for SourceAnywhere for VSS is:
http://www.dynamsoft.com/Products/SAW_Overview.aspx
The project level security mechanism in VSS can only prevent unintended changes. If you are still interested in learning more about how to set the project level securities in VSS, you can read more about it below.

Managing project level security

To manage the project rights for an individual command for each user, we can follow the steps below:

1. Open **Visual SourceSafe Administrator** program.
2. Check the **Enable Rights and Assignments commands** box in the **Visual SourceSafe Administrator** menu **Tools** -> **Options** -> **Project Rights** tab. In the **New User Rights** area of the **Project Rights** tab, we can deselect the project rights that do not apply to any database users.

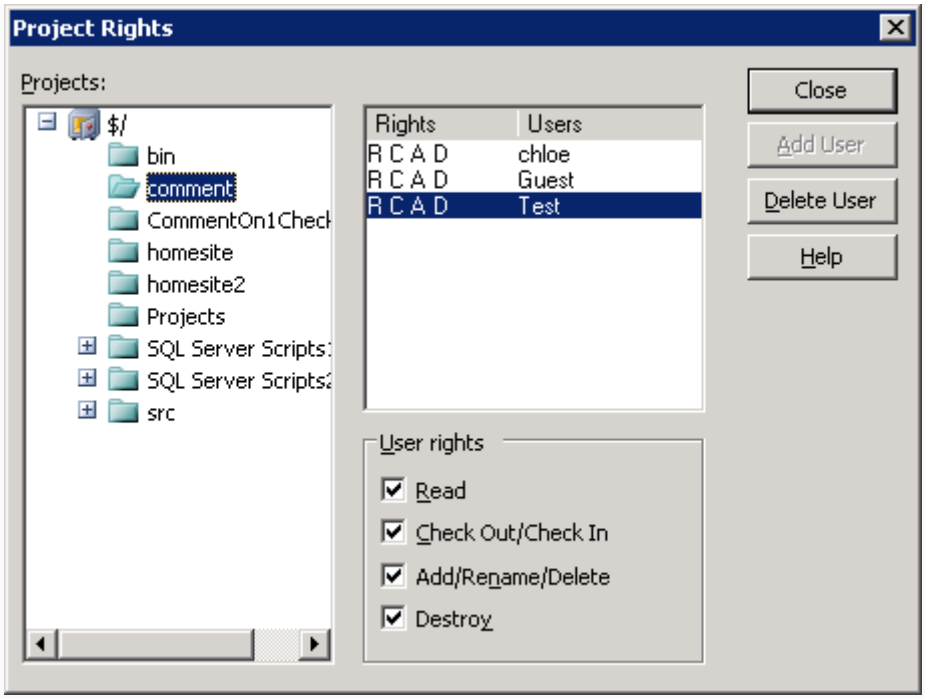


(SourceSafe Options)

3. Now there are 3 rights commands available on the Tools menu: Rights by Project, Rights Assignments for User and Copy User Rights.

To assign project rights from the project list:

1. In Visual SourceSafe Administrator, click **Tools** -> **Rights by Project**.
2. In the **Project Rights** dialog box, select a project and click **Add User** to attach the user for whom to assign project rights.

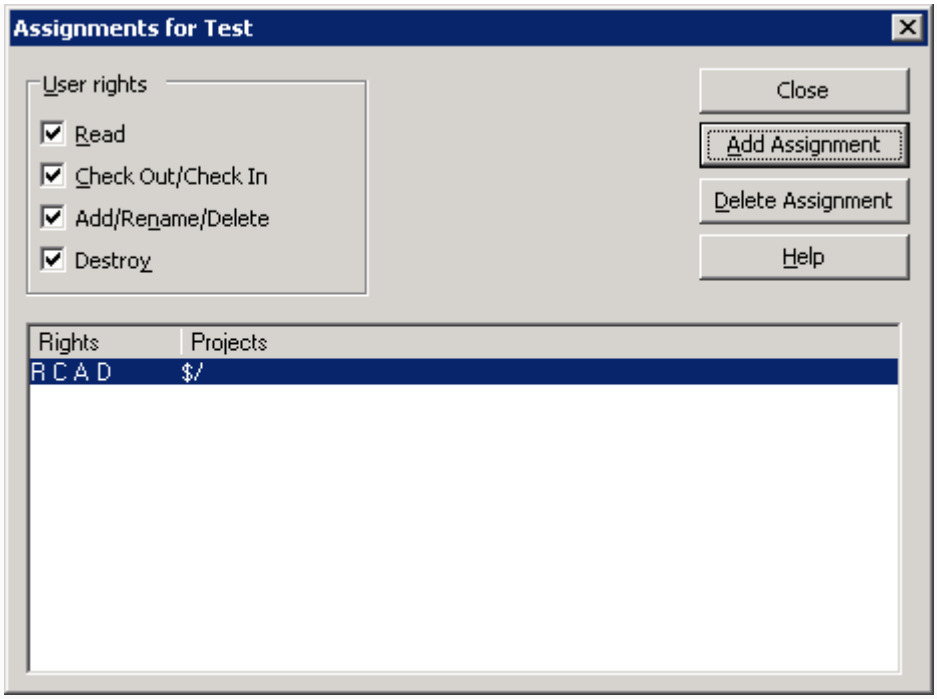


(Project Rights)

3. Select a user in the user list. Under **User rights**, specify the permissions.

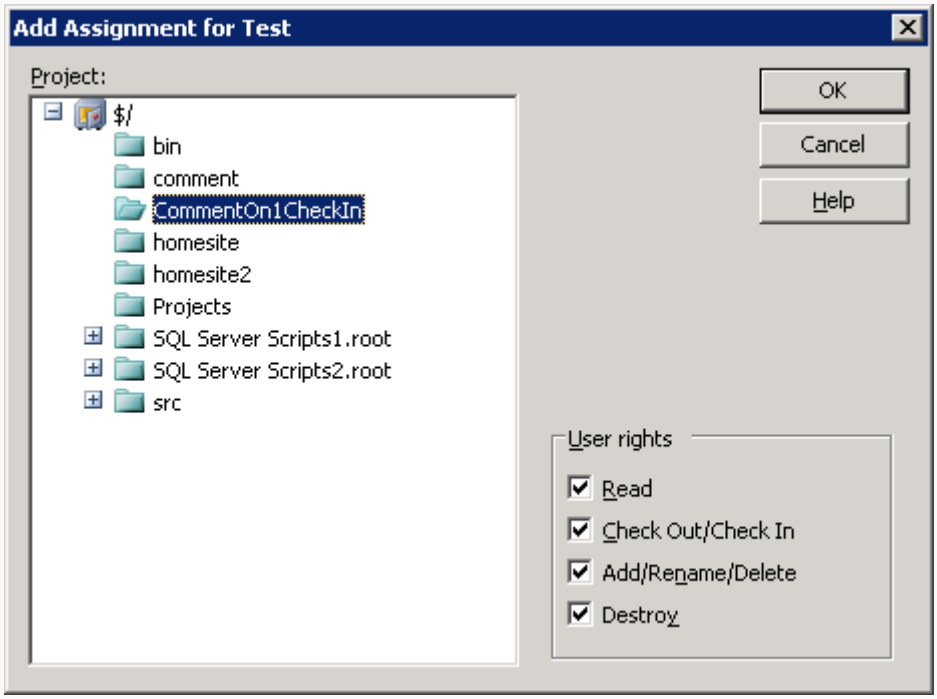
To assign project rights from the user list:

1. In Visual SourceSafe Administrator, select a user in the users list, and click **Tool** ->**Rights Assignments for User**.
2. In the **Assignments for** dialog box, click **Add Assignment**.



(Assignments for)

3. Select a Visual SourceSafe project and then specify permissions for the user on the selected project. Please be advised that a user must have the Destroy project right to deploy a Web site.

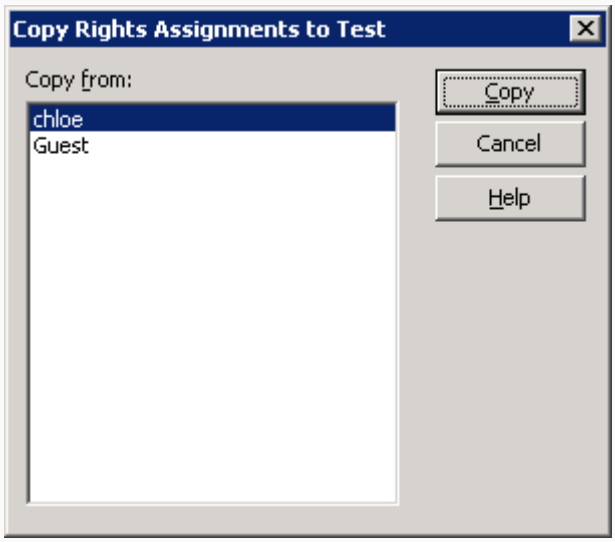


(Add Assignment for)

To copy one's user rights to another user:

1. In Visual SourceSafe Administrator, click the user whose project rights you want to modify in the users list.

2. Click menu **Tools** -> **Copy User Rights**. The **Copy Rights Assignments to** dialog box prompts out.



(Copy Rights Assignment to Test)

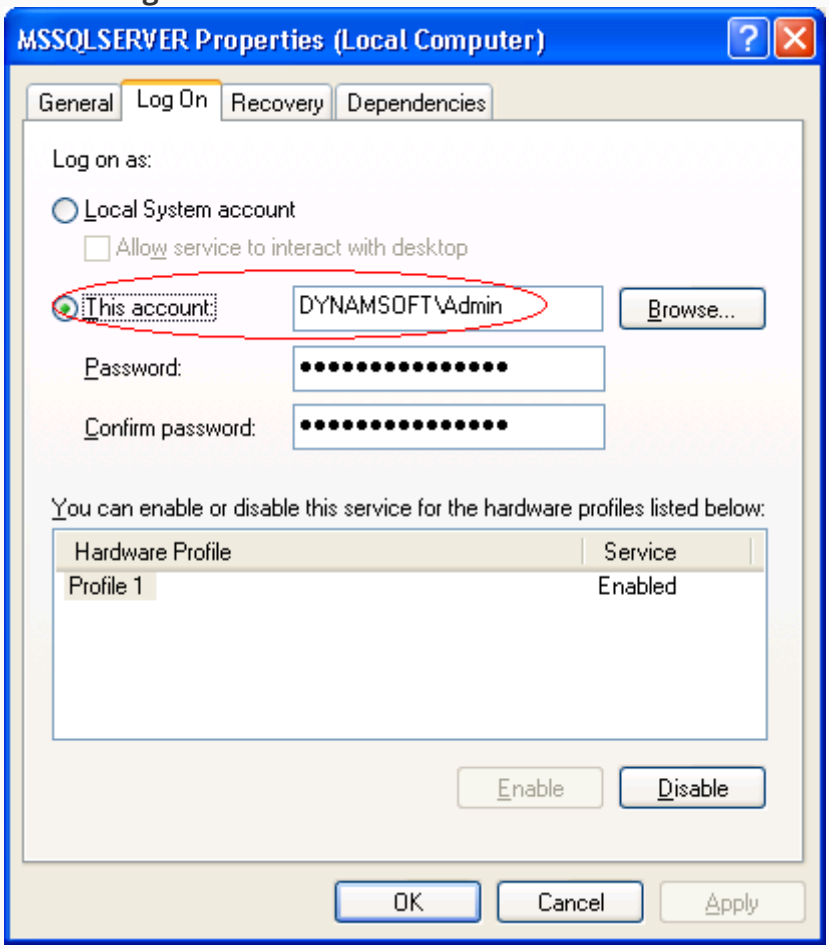
3. Select a user from whom to copy rights, and then click **Copy**.

How to version control SQL Server Stored Procedures with Visual Studio 2003

SQL Server stored procedures can be added to Visual SourceSafe (VSS) for version control by using the source control feature in Visual Studio .NET 2003.

To add SQL Server stored procedures to VSS, please follow the steps below:

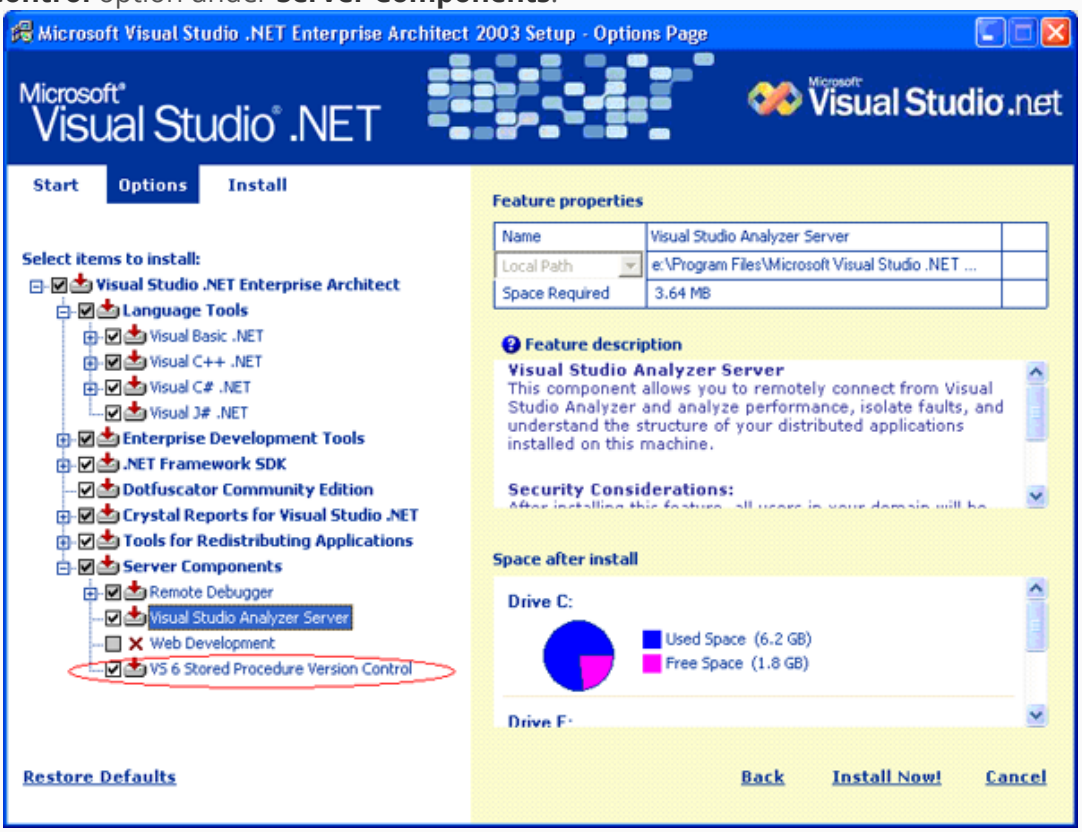
1. Verify that SQL Server is running under a domain account and the current log on user of SQL Server has read & write rights to the VSS Database folder. We can check this by right-clicking **My Computer**, and then clicking **Manage** -> **Services and Applications** ->**Services**. In the service list, right-click MSSQLSEVER and click **Properties** -> **Log On tab**.



(SQL Server Log On user)

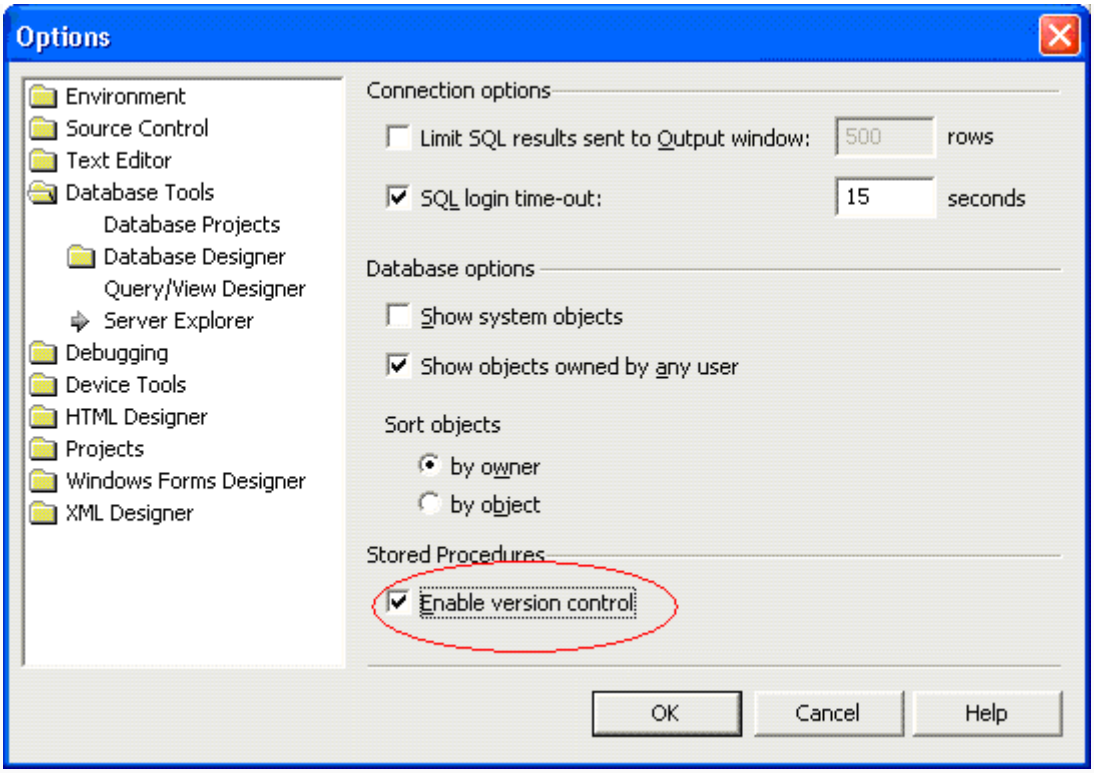
2. Install Visual SourceSafe on the SQL Server machine.

3. Install Visual Studio .NET server components on the computer running SQL Server. During the installation, please check the **VS 6 Stored Procedure Version Control** option under **Server Components**.



(Visual Studio .net 2003 Setup)

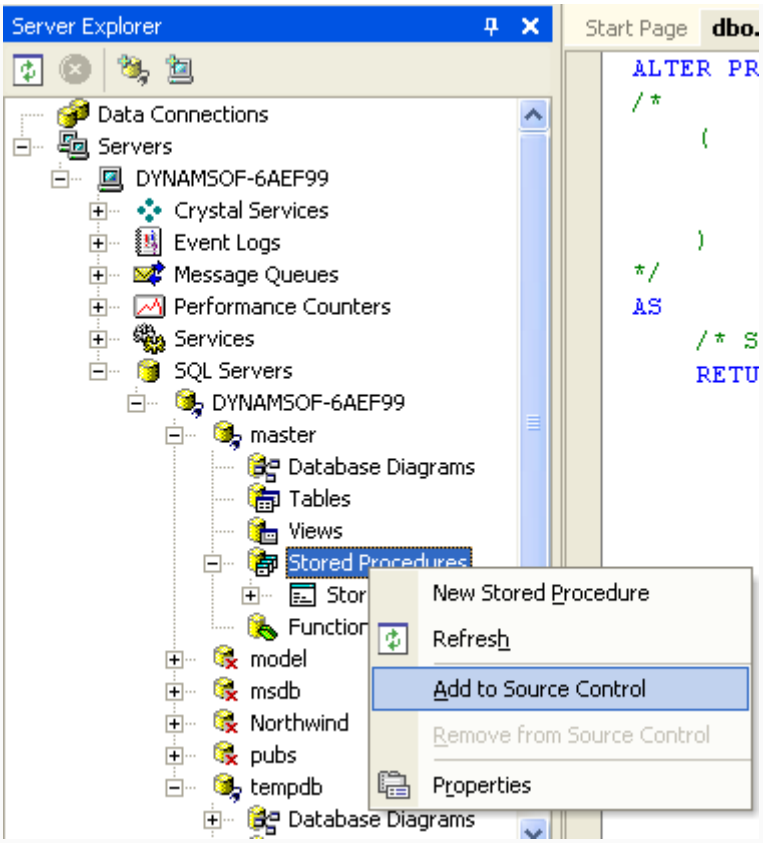
4. Start Visual Studio .NET, click menu **Tools** -> **Options** -> **Database Tools** -> **Server Explorer**, and check **Enable version control** under **Stored Procedures**.



(Enable Version Control)

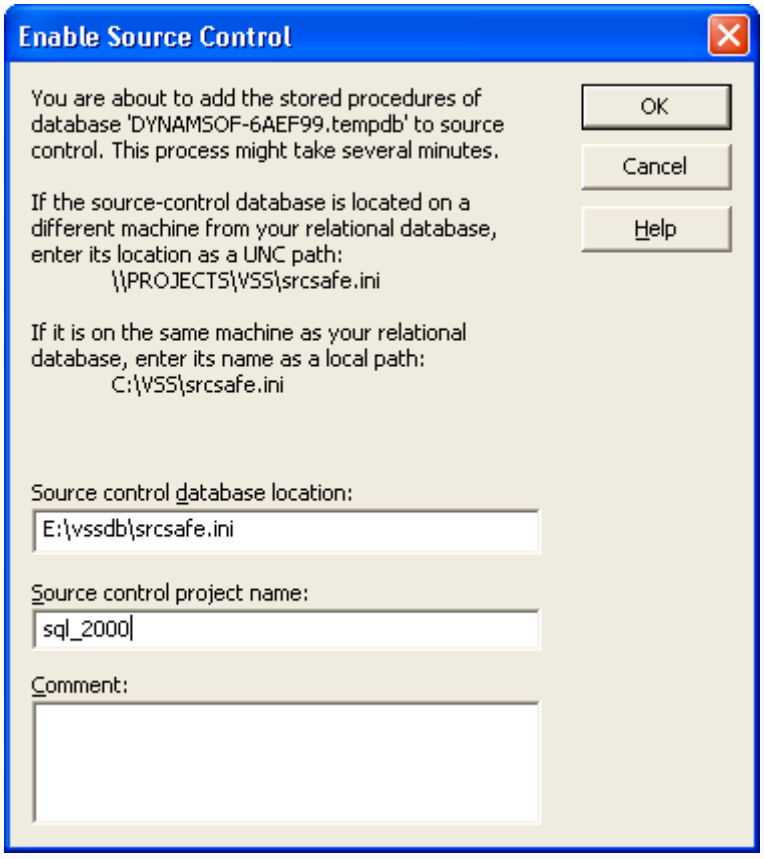
5. Open the **Server Explorer** pane by clicking menu **View** -> **Server Explorer**, and then expand to the **Stored Procedures** folder.

6. Right-click the **Stored Procedures** folder, and click **Add to Source Control**.



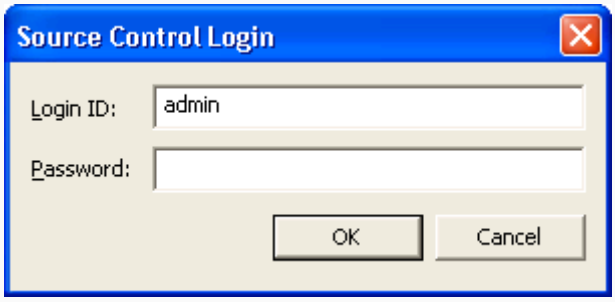
(Add to Source Control)

7. The **Enable Source Control** dialog box prompts out. We need to input the VSS database location and project name here. Please note that we do not need to use the "$/" prefix in the **Source Control Project Name** text box. Visual Studio .NET adds "$/" automatically. For example, if the project name in VSS database is "sql_2000", we should input "sql_2000" rather than "$/sql_2000".



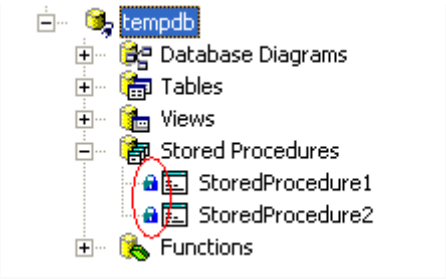
(Enable Source Control)

8. In the **Source Control Login** dialog box, type the VSS **Login ID** and the **Password**, and then click **OK**.



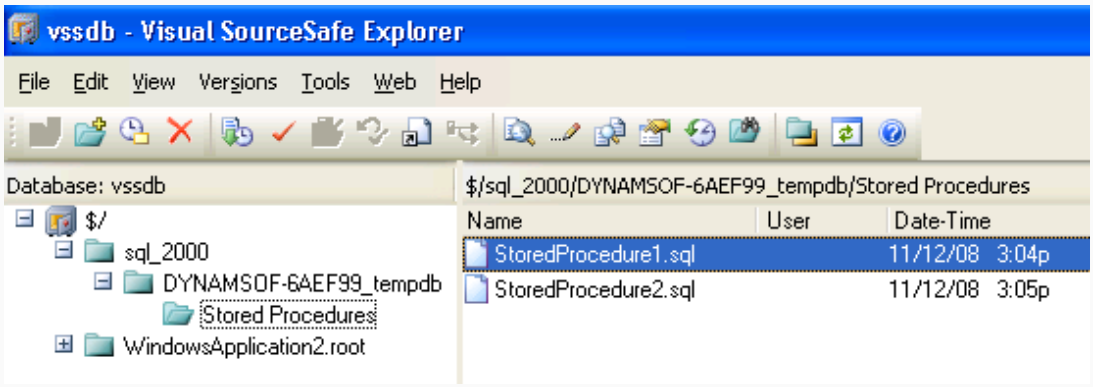
(Source Control Login)

9. We can now add stored procedures into source control of SourceSafe by right-clicking on the stored procedure files and selecting **Add to Source Control**. There will be lock icons on the left side of the stored procedure files denoting the files are in source control, as shown in the following screenshot:



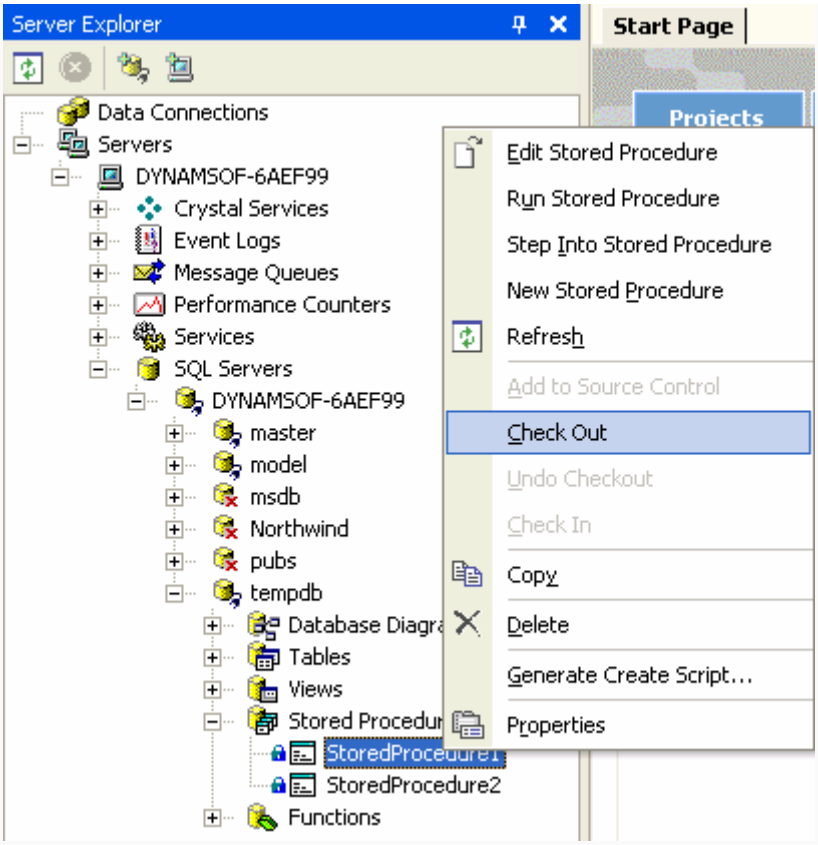
(Lock Icon)

And we will see the stored procedure files in SourceSafe Explorer:



(VSS Explorer)

10. After all the steps above were done, we will be able to find the SourceSafe options by right-clicking the stored procedure files in the **Server Explorer** pane.



(Source Control Operations)

For information on how to source control SQL Server objects in SQL Server Management Studio 2005/2008, please refer to my other articles: Integrating SourceSafe / VSS with SQL Server 2005 and Integrating SourceSafe / VSS with SQL Server 2008.