
Perforce 2007.3 Using IDE Plug-ins

March 2008

This manual copyright 2001-2008 Perforce Software.

All rights reserved.

Perforce software and documentation is available from <http://www.perforce.com>. You may download and use Perforce programs, but you may not sell or redistribute them. You may download, print, copy, edit, and redistribute the documentation, but you may not sell it, or sell any documentation derived from it. You may not modify or attempt to reverse engineer the programs.

Perforce programs and documents are available from our Web site as is. No warranty or support is provided. Warranties and support, along with higher capacity servers, are sold by Perforce Software.

Perforce Software assumes no responsibility or liability for any errors or inaccuracies that may appear in this book.

By downloading and using our programs and documents you agree to these terms.

Perforce and Inter-File Branching are trademarks of Perforce Software. Perforce software includes software developed by the University of California, Berkeley and its contributors.

Visual Studio, Visual .NET, Visual C++ and Visual Basic copyright Microsoft Corporation.

WASTE Text Engine © 1993-2002 Marco Piovanelli.

All other brands or product names are trademarks or registered trademarks of their respective companies or organizations.

Table of Contents

Preface	About This Manual	7
	Please Give Us Feedback	7
Chapter 1	Basic Concepts	9
	About IDEs and plug-ins.....	9
	About Perforce	9
	Tracking changes	10
	Handling conflicts	10
	File types.....	11
	Configuring IDEs with plug-ins.....	12
	Configuring Perforce preferences for Visual Studio IDEs.....	12
	Defining the client workspace and view	15
	Add a project to source control	16
	Adding files to the depot	16
	Basic SCM tasks	17
	When to go outside the IDE	17
Chapter 2	Microsoft Visual C++	19
	Basic SCM tasks	19
	Adding a project to the depot.....	19
	Retrieving files from the depot.....	21
	Checking files out.....	22
	Checking files in	22
	Resolving file conflicts.....	23
	Diffing files	24
	Reverting files	24
	Working with Visual C++ files.....	24
	Which files do I put in the depot?.....	25
	Location of project files.....	25

Chapter 3	Microsoft Visual Basic.....	27
	Configuring Visual Basic with Perforce.....	27
	Basic SCM tasks.....	27
	Adding a project to the depot.....	28
	Checking files out.....	29
	Retrieving files from the depot.....	29
	Checking files in	30
	Resolving file conflicts	30
	Diffing files	31
	Reverting files	32
	Working with Visual Basic files	32
	Which files do I put in the depot?	32
	Location of project files.....	32
	Recommended Perforce file types.....	33
	Checking out the project file	33
Chapter 4	Microsoft Visual Studio .NET.....	35
	Configuring Visual Studio .NET with Perforce.....	35
	Basic SCM tasks.....	36
	Adding a project to the depot.....	36
	Checking files out.....	38
	Retrieving files from the depot.....	39
	Checking files in	40
	Resolving file conflicts	40
	Diffing files	41
	Reverting files	42
	Which files do I put in the depot?	42
	Miscellaneous tasks.....	42
Chapter 5	Microsoft Visual Studio 2005.....	43
	Basic SCM tasks.....	43
	Adding a project to the depot.....	44
	Checking files out.....	45
	Retrieving files from the depot.....	46
	Checking files in	47
	Resolving file conflicts	47

Diffing files.....	48
Reverting files	49
Which files do I put in the depot?.....	49
Miscellaneous tasks.....	49
 Index	 51

About This Manual

This guide describes the configuration, and operation of Perforce plug-ins for a variety of integrated development environments (IDEs). For installation instructions, refer to the release notes for the SCC plug-in.

This manual assumes some familiarity with Perforce. For basic information on Perforce, consult *Introducing Perforce*.

All our documentation is available from our web site at <http://www.perforce.com>.

Please Give Us Feedback

We are interested in receiving opinions on it from our users. In particular, we'd like to hear from users who have never used Perforce before. Does this guide teach the topic well? Please let us know what you think; we can be reached at manual@perforce.com.

About IDEs and plug-ins

Perforce plug-ins enable you to perform basic Perforce software configuration management (SCM) tasks from within integrated development environments (IDEs). The Perforce SCC Plug-in is installed when you install Perforce on Windows, and is based on the Microsoft Source Code Control (SCC) API and has been tested with Microsoft Visual C++, Visual Basic, and Visual Studio .NET. For details about supported versions of each IDE, refer to the release notes for the plug-ins.

Perforce also offers plug-ins for other development tools. See our Web site for details.

Note that the terminology for SCM tasks varies depending on which development environment you are using. The following table compares some common terms.

IDE with SCM plug-in	Other commonly used terms	Corresponding Perforce command
Add to source control/Perforce	Add	p4 add
Check out	Edit	p4 edit
Check in	Submit	p4 submit
Show differences	Diff	p4 diff
Get latest version	Sync, Refresh	p4 sync
Show history	Filelog	p4 filelog
Undo checkout	Revert	p4 revert
Remove from source control	Delete	p4 delete

About Perforce

Perforce is an SCM system that is based on a client/server architecture. The main repository (the *depot*) resides on a central server while the files you work on reside in a workspace on your local machine. You can place some or all of the files in your workspace under source control. When you perform SCM tasks, the files remain in your workspace, and Perforce reads or writes them as required. For example, when you submit a change, Perforce reads the edited files in your workspace and updates the information in the database accordingly. When you issue a **Get latest version** or **refresh** command, Perforce transfers files from the depot to your workspace.

The most current revision of the file in the depot is called the *head revision*. Perforce allows you to check out the head revision or any previous revision of a file. To enforce the IDEs' check-in and check-out procedure, Perforce controls the read-write permissions of files. When files are checked out for edit, their permissions are set to read-write. When files are not checked out, Perforce sets them to read-only.

Perforce submits changed files in groups called *changelists*. Perforce keeps track of a project's revision history as a sequence of changelists. This approach allows you to reconstruct a project in your workspace as it appeared at any point in its history. Perforce makes changes *atomically*, meaning that when you submit changes to a group of files, either all of the changed files are accepted simultaneously or none of them are. If conflicts result from multiple users working on the same files, these conflicts must be resolved before Perforce will accept the changes.

Perforce offers a command-line client (P4), a native Windows GUI (P4Win), a cross-platform GUI (P4V) and a Web-based interface (P4Web). For detailed information about Perforce, refer to the user documentation available on the web at:

<http://www.perforce.com/perforce/technical.html>

Most IDEs are project-based: they manage a group of files according to the project to which the files belong. However, Perforce manages files with no provisions for specific IDEs' project structure. You must determine which files you need to place into your Perforce depot, depending on the conventions of your IDE and your group development practices.

Tracking changes

When files are added to source control, deleted from source control, or checked out for edit, Perforce adds them to a changelist. The changelist contains the file names, revision numbers, and operations to be performed. Any edits you make to checked out files are kept in your local client workspace until you send the changelist to the depot with a check in or submit command.

The Perforce server tracks changelists by numbering them sequentially. Changelist numbers are displayed in a results window after a change has been submitted.

Handling conflicts

The ability to detect and resolve conflicts is important in team development, when multiple developers are working on the same files. For example, suppose two programmers copy the same file from the depot into their workspaces and each programmer edits the file differently. When the first programmer submits his version of the file to the depot, the file becomes the head revision. When the second programmer tries to submit her changes to the depot, Perforce determines that her changes are based on a previous revision and does not allow the file to be checked in. If the second file were accepted without question, the first programmer's changes would be overwritten.

When Perforce detects a conflict, it requires you to choose the changes to be checked in. When resolving file conflicts, you can use a merge utility to display the differences between two text files, to help you determine how to resolve the conflict. For more details, see the *Perforce Command Reference*.

File types

Perforce automatically detects whether files placed under source control are text or binary files and stores them accordingly on the server machine. By default, text files are stored in reverse delta format, and binary files are stored in their entirety. The following table lists recommended Perforce file types and attributes for common file types. For details about Perforce file types, refer to *Introducing Perforce*.

File Type	Perforce file type	Description
.asp	text	Active server page file
.avi	binary+F	Video for Windows file
.bmp	binary	Windows bitmap file
.btr	binary	Btrieve database file
.cnf	text	Conference link file
.css	text	Cascading style sheet file
.doc	binary	Microsoft Word document
.dot	binary	Microsoft Word template
.exp	binary+w	Export file (Microsoft Visual C++)
.gif	binary+F	GIF graphic file
.htm	text	HTML file
.html	text	HTML file
.ico	binary	Icon file
.inc	text	Active Server include file
.ini	text+w	Initial application settings file
.jpg	binary	JPEG graphic file
.js	text	JavaScript language source code file
.lib	binary+w	Library file (several programming languages)
.log	text+w	Log file
.mpg	binary+F	MPEG video file
.pdf	binary	Adobe PDF file
.pdm	text+w	Sybase Power Designer file

File Type	Perforce file type	Description
.ppt	binary	Microsoft Powerpoint file
.xls	binary+w	Microsoft Excel file
.zip	binary+F	ZIP compressed archive file

Note | The Perforce `apple` file type stores Macintosh files in the depot as a single file containing both the data and resource forks. Pre-2000.1 versions of Perforce stored Macintosh file forks as separate files in the depot. If your depots contain Macintosh files stored in this manner, upgrade the existing files to use the `apple` format. If you do not convert existing files after you upgrade, Perforce continues to store them using its two-file approach.

Configuring IDEs with plug-ins

Regardless of the IDE you use, you must define a client view, add files to the Perforce depot, and configure your source control settings and preferences. The following sections provide details about these tasks.

Configuring Perforce preferences for Visual Studio IDEs

To configure Perforce preferences, display the settings control panel by choosing the corresponding menu entry:

- Visual Studio .NET: **Tools > Options > Source Control > SCC Provider** and click **Advanced...**
- Visual Basic 6.0: **Tools > Perforce > Options** and click **Advanced...**
- Visual C++ 6.0: **Tools > Options**, scroll right and click the **Source Control** tab and click **Advanced...**

The following sections describe the Perforce settings you can configure on each tab.

General tab

Option	Description
Date format	Specifies whether dates are displayed using the operating system format or the Perforce format.
Log options:	
Log All	Logs all commands sent to the Perforce server. To log only user-initiated commands, disable this option.
Enable logging to file	Specifies the name, location and maximum size of the log file.
Enable diff2 on file-to-file drag and drop	Enables you to diff files in the depot using drag and drop. To prevent inadvertently launching a diff (if you never use drag-and-drop diffing), disable this feature.
Warn before reverting files	Displays a dialog enabling you to change your mind before reverting files. When you revert an open file, any changes you made are discarded.

Connection tab

Option	Description
When binding a project to source control	<p>Specifies how you want to configure the Perforce server and workspace associated with the solution.</p> <ul style="list-style-type: none">• Show the Perforce connection dialog: enables you to specify the settings manually.• Bind to the workspace that matches your Perforce environment settings: uses the settings in effect when you add the project (“global” settings). <p>If you are adding a solution that contains a large number of projects, choose Bind to the workspace that matches your Perforce environment settings and configure the <code>P4PORT</code> and <code>P4CLIENT</code> settings to specify the server and workspace. If you choose Show the Perforce connection dialog, you must manually enter settings for each project in the solution.</p>
When connecting to a server	
Show Remember Password dialog	<p>For 2003.2 server and below: if enabled, the plug-in displays a “Remember Password” dialog when prompting you for your password. If disabled, the plug-in prompts you every time the Perforce server requires your password.</p>
Show Login Expiration dialog	<p>For 2004.2 server and above: Enables/disables display of login-related dialogs. Perforce servers that are running in a high security mode can set a session time limit. After the session expires, you are prompted to log in when you attempt another Perforce operation.</p>

Option	Description
Unicode servers	
Set encoding for all connections	Specifies encoding used on client machine when connecting to a Perforce Server that is running in Unicode mode. DO NOT SET unless you know for sure that your server is running in Unicode mode.

Diff tab

Option	Description
Default diff application	Enables you to select the application that is used to diff your files. Optionally enables you to configure different applications for specific file types.

Merge tab

Option	Description
Default merge application	Enables you to select the application that is used to perform three-way merges when you resolve files.

Version tab

This tab displays version information about the Perforce SCC Plug-in.

Font tab

This tab enables you to configure the font used by Perforce merge and Time-lapse View utilities.

Defining the client workspace and view

To set up the Perforce SCM environment, you define a *client workspace* (a directory on the client machine that contains the project files) and a *client view* (a mapping of the depot to your client computer). The files in your IDE projects must reside under the client workspace root (the highest-level directory of your client workspace). Your files can be put under source control only if they are located in the client workspace. You can specify a directory that already contains files, or an empty directory for a project you intend to create. You can also populate your client workspace with files from the depot which are currently under version control.

To create a client workspace and view, you must use the Perforce Command-Line Interface (P4) program. Note that P4SCC does not observe settings in Perforce config files. For details about client workspaces and views, refer to *Introducing Perforce*.

Add a project to source control

To add a project to source control, you specify the Perforce settings required to connect to the server where you want to store the project. You must specify the following settings:

- Server host and port

This setting specifies the name of the host on which the server is running and the port on which the server is listening, using the following format:

IDE and Perforce host location	Required Settings	Examples
Same machine	Only port number is required	1666 1818
Different machines	Host name (or IP address) and port are required	myhost:1818 mydomain.com:2550 10.0.0.5:1666

- User name: your Perforce user name.
- Password: (optional) your Perforce user password.
- Client workspace: the name of the client specification describing the workspace where you work on local copies of project files.

Adding files to the depot

After you specify the settings for the server where you want to store project files, you can add the files. The exact commands required to add files to the depot depend on the IDE in which you are working. Typically you perform two steps:

1. Add the files to a Perforce changelist by issuing a command such as **Add to Source Control**.
2. Submit the changelist by issuing a command such as **Check In** or **Submit**.

Files are added when the **Check In** or **Submit** is completed.

Basic SCM tasks

The plug-ins enable you to perform the following basic SCM tasks from within your IDE:

Task	Description	Perforce activity
Add files to the depot	After you add files to your project, you add them to the depot.	Files are opened for add in a pending changelist.
Retrieve files	Get a copy of a file from the depot.	Files are synced to your computer.
Check files out	Get the latest version from the depot for editing.	Files are opened for edit in a pending changelist.
Check files in	Put your edited files in the depot as the most recent revisions.	A pending changelist is submitted.
Diff files	Compare a file with a previous revision to see what's changed.	The Perforce diff utility is launched to display differences between two versions of a file.
Revert files	Discard changes you've made to your local copy of a depot file.	The head revision is synced to your client workspace and removed from its pending changelist.
Delete files	Remove a file from a project and from the depot.	Files are opened for delete in a pending changelist.

When to go outside the IDE

To perform administrative tasks, you need to use the p4 command line client. For details about performing these tasks, see the *Perforce Administrator's Guide*.

Tasks that require you to work outside the IDE include:

- *Modifying a client view or user password:* you can create a client specification or user through the **Connection** dialog when adding a project to source control, but to modify it subsequently, you must use another Perforce client program (the p4 command line, P4Win, or P4V).
- *Creating branches and labels:* these and other administrative tasks must be done using the p4 command. Refer to *Introducing Perforce* for basics.

If you do perform Perforce SCM tasks outside the IDE, be sure to refresh the IDE display afterwards, to ensure that file status is updated and displayed correctly.

This chapter describes how to perform Perforce source code control tasks in the Visual C++ 6.0 environment. It assumes the Perforce SCC Plug-in is installed and that your client machine can communicate with the Perforce server.

When you configure Visual C++ with the Perforce SCC Plug-in, the settings are specific to the project you are working on, so no overall Perforce configuration is required. The Perforce SCC Plug-in retains project settings from session to session and it is not necessary to re-enter them. For details about configuring settings, see “Configuring IDEs with plug-ins” on page 12.

Visual C++ has a **Source Code Control** toolbar: to enable it, right-click the toolbar and check **Source Code Control** on the pop-up menu. To specify source control options, choose Tools> Options... and

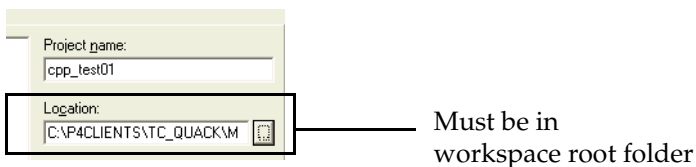
Basic SCM tasks

This section tells you how to perform the following tasks:

- “Adding a project to the depot” on page 19
- “Checking files out” on page 22
- “Retrieving files from the depot” on page 21
- “Checking files in” on page 22
- “Resolving file conflicts” on page 23
- “Diffing files” on page 24
- “Reverting files” on page 24

Adding a project to the depot

When you create a project you intend to manage with Perforce, specify a location that resides within the root folder of the Perforce workspace you intend to use as illustrated in the following figure.



After you save your project, perform the following steps:

1. Right-click the project and choose **Add to Source Control...** The **Open Connection** dialog is displayed.
2. On the **Open Connection** dialog, specify Perforce settings as follows:
 - *Server*: the name of the host computer running the Perforce server in which you want to store the project.
 - *Port*: the port number on which the specified Perforce Server accepts commands.
 - *User*: the Perforce user name you want associated with the SCM operations you perform on the project. To choose from a list of users defined for the specified server, click **Browse**.
 - *Password*: the user's password, if any.
 - *Workspace*: the workspace you want associated with the SCM operations you perform on the project. You must save the project in a folder under the workspace root. To choose from a list of workspaces defined for the specified server, click **Browse**.
3. Click **OK** to save settings and dismiss the dialog. The **Add to Source Control** dialog is displayed, listing the files to be added.
4. Click **OK** to dismiss the dialog.
5. Right-click the project and choose **Check In...** The **Check in Files** dialog is displayed.
6. Click to dismiss the dialog. The **Submit Changelist** dialog is displayed.
7. Check the files you want to add, enter a description and click **Submit**. The files are checked into the depot.

When you open a project that you have placed under Perforce control, Visual C++ attempts to connect to the Perforce server that you configured. If Visual C++ can not connect to Perforce, it asks you whether it should try to connect in future sessions. Choose **Yes**. If you choose **No**, Visual C++ assumes the project is no longer under version control, requiring you to add it again. If you try to add the project again, Perforce returns an error indicating that the project is already under source control, and you must retrieve the project from the depot.

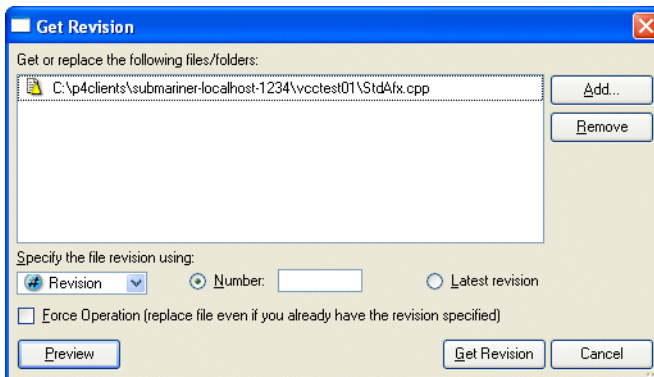
Retrieving files from the depot

To get the most recent file revisions from the depot:

1. In the File View pane, select the files you want to retrieve.
2. Choose **Project >Source Control >Get Latest Version...** The **Get Latest Version** dialog is displayed.
3. Check the files you want to retrieve and click **OK**.

To retrieve revisions prior to the head (latest) revision, choose **Project>Source Control>Show History** or perform the following steps:

1. In the File View pane, select the files you want to retrieve.
2. Choose **Project >Source Control > Get Latest Version...** The **Get Latest Version** dialog is displayed.
3. Click **Advanced...** Dismiss the “Advanced sync dialog...” prompt by clicking **OK**, then click **OK** on the **Get Latest Version** dialog. The **Get Revision** dialog is displayed as shown in the following figure.



4. Specify the revision you want to retrieve, using a Perforce changelist number, label, version number, or date. For details about specifying revisions, refer to the *Perforce Command Reference*.
 - To preview the results of the operation without actually retrieving files, click **Preview**.
 - To retrieve a file from the depot if you’ve deleted your local copy manually, check **Force Sync**. By default, Perforce assumes you still have your copy and doesn’t retrieve it again.

5. Click **Get Revision**.

Note | If you retrieve a file that you already have checked out, for example, to obtain changes checked in by another user, Perforce will require you to resolve the file, to enable you to merge your changes with changes made by other users and to ensure that you don't inadvertently overwrite other changes.

Checking files out

Before you check a file out, be sure you have retrieved the latest revision of the file. (If you check out a file for which the depot contains later revisions, you will be required to resolve it when you check it in.)

To check out a file for edit, right-click the file and choose **Check out filename**.

Checking files in

After editing the open files, you can save them to your local directory and check the files into the depot. To save the files on your local directory, choose **File>Save**. Note that saving does not update the depot.

To make your changes available to others working on the project, you must check your edited files into the depot. If the changes are accepted, your files become the most recent (or *head*) revisions in the depot. If any of the changes conflict with changes previously submitted by another user, the check-in fails and the differences must be resolved using Perforce. To minimize the complexity of merges, check in small sets of changes frequently.

To check in a file:

1. Right-click the file and choose **Check in...**
The **Check in file(s)** dialog is displayed.
2. Click **OK**. The **Submit Changelist** form is displayed.
3. Enter description and click **Submit**.

To view all your pending changelists, click the **Manage Changelists** button on the **Submit Changelist** form. P4SCC displays a dialog listing your pending changelists. In this dialog, you can move files among changelists by dragging and dropping them.

Resolving file conflicts

If you encounter conflicts when checking in files, (for example, from multiple users working on the same files,) you must resolve the conflicts before you can submit the files. For example, if you and another user have changed the same file and the other user checked in changes before you, when you attempt to check in your changes, Perforce notifies you that resolution is required by displaying the following message in the **Submit Changelist** dialog:



Out of date: Get latest revision to set up a resolve (this will not overwrite your copy of the file).

To resolve file conflicts:

1. Right-click the file that has conflicts and choose **Get Latest Revision**.

The file is now scheduled for resolve, indicated by the message “Resolve this file before submitting.” (Your workspace version is not overwritten.)

2. Right-click the file and choose **Resolve Files...** The resolve dialog is displayed, describing the extent of the file differences and listing the following options:
 - *Accept Merge*: merge both files and accept the result, including both the changes you made and the changes made by other users.
 - *Accept Yours*: check your changes in, overwriting the most recently checked-in version.
 - *Accept Theirs*: discard your changes and preserve the other user’s changes.
 - *Run Merge Tool*: launch the merge utility so you can view the differences between your version and the most recently checked-in version. Using the merge utility, you can select individual passages or enter new changes to create the final version that you check in.
 - Other choices: you can view either file (**Open File...**), diff your file with the conflicting file, display the file’s revision history either textually or graphically.

After you choose how the files are resolved, the **Submit Changelist** dialog is redisplayed.

3. Enter a description of your changes and click **Submit**. Your changes are checked in.

Diffing files

To diff a file you are editing with the head revision in the depot, right-click the file and choose **Project > Source Control > Show Differences**.

To diff two revisions of a file, perform the following steps:

1. Select the file and choose **Project > Source Control > Show History...** The **History** dialog is displayed.
2. Click and drag one of the revisions you want to diff, and drop it on the other revision.

The diff utility is launched, displaying file differences.

Reverting files

To discard any changes you have made to a file after checking it out, right-click the file and choose **Undo Check-out**. The head revision from the depot is copied to your client workspace, overwriting any changes you have made. Using Perforce, you can verify that the file is removed from the pending default changelist.

Working with Visual C++ files

Visual C++ displays a file status icon next to each file in the project window. In addition, the Source Control menu is context-sensitive: if you highlight a file, the menu selections are enabled or disabled depending on file status. You can use the icon or **Source Control** menu appearance to identify file status.

File status	Icon appearance	Source Control menu appearance
Files are not under Perforce source code control	Original VC++ icon	Add to Source Control option is enabled
Files have been added to a changelist but not submitted to Perforce.	Icon is gray with red check mark	Check in and Undo Check out options are enabled
Files have been successfully submitted to Perforce.	Icon is gray	Check out option is enabled

Which files do I put in the depot?

Put the following files in your Perforce depot:

- .dsp files
- .rc files
- .cpp files
- .h files

Do not put IDE-generated files (such as .ncb, .clw and .opt files) in your Perforce depot. If you put the .ncb file under Perforce control, it is marked read-only when it's not checked out. If the .ncb file is read-only, class view information in VC++ is disabled.

If you intend to put the workspace file (.dsw) under Perforce control and multiple developers work on the same project, use relative paths to specify file locations to ensure that the entries in the workspace file work correctly on different client computers.

Location of project files

The FileView window displays source file names and their location on your local directory, but is only an approximate representation. Although the display simplifies project development in Visual C++, it differs from the actual file structure as follows:

- FileView groups files by type, such as Source files and Header files, whereas the local directory does not.
- FileView does not display the exact file names for the workspace file and the project file.

For some source control tasks, you need to specify the names and locations of these files. To view this information, right-click on the file and choose **Properties**. The **Source File Properties** dialog is displayed. The **General** properties option displays the complete file name and path.

This chapter describes how to perform Perforce source code control tasks in the Visual Basic 6.0 environment. It assumes the Perforce SCC Plug-in is installed and that your client machine can communicate with the Perforce server. For details about configuring settings, see “Configuring IDEs with plug-ins” on page 12.

The **Perforce** submenu lists the SCM operations you can perform. Note that the **Tools>Perforce>Create Project from Perforce...** option is not supported.

If the **Add-Ins > Add-In Manager...** menu item does not contain **Source Code Control** in the list box, you do not have the VB Source Code Add-In. For help getting it from Microsoft, contact support@perforce.com.

Configuring Visual Basic with Perforce

Before you can put Visual Projects under Perforce control, verify that source code control is enabled, as follows:

1. In Visual Basic, choose **Add-Ins>Add-In Manager...** The **Add-In Manager** dialog is displayed.
2. If **Source Code Control** is not listed, close Visual Basic, edit your `vbaddin.ini` file (located by default in the `%windir%` folder), add the following entry under `[Add-Ins32]`

```
vb SCC=3
```
3. Save the file.

Basic SCM tasks

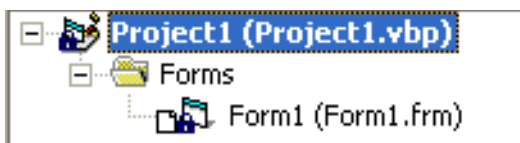
This section tells you how to perform the following tasks.

- “Adding a project to the depot” on page 28
- “Checking files out” on page 29
- “Retrieving files from the depot” on page 29
- “Checking files in” on page 30
- “Resolving file conflicts” on page 30
- “Diffing files” on page 31
- “Reverting files” on page 32

Adding a project to the depot

1. Choose **Tools>Perforce>Add Project to Perforce**.
2. Click **Yes**. If you enabled the **Show the Perforce Connections** option on the **Preferences > Connection** tab, the **Open Connection** dialog is displayed.
3. On the **Open Connection** dialog, specify Perforce settings as follows:
 - *Server*: the name of the host computer running the Perforce server in which you want to store the project.
 - *Port*: the port number on which the specified Perforce Server accepts commands.
 - *User*: the Perforce user name you want associated with the SCM operations you perform on the project. To choose from a list of users defined for the specified server, click **Browse**.
 - *Password*: the user's password, if any.
 - *Workspace*: the Perforce workspace you want associated with the SCM operations you perform on the project. To choose from a list of workspaces defined for the specified server, click **Browse**.
4. Click **OK** to save settings and dismiss the dialog. The Add Files to Perforce dialog is displayed, listing the files to be added.
5. Click **OK** to dismiss the dialog.
6. Right-click the project and choose **Check In....** The Check in Files to Perforce dialog is displayed, listing files to be checked in.
7. Click **OK**. The **Submit Changelist** dialog is displayed.
8. Check the files you want to add, enter a comment and choose **Submit**. The files are added to the depot.

Checked-in files are displayed with a lock icon, as shown in the following figure.



Checking files out

To check out a file for edit, right-click the project file in the Project window and choose **Check Out** from the pop-up menu.

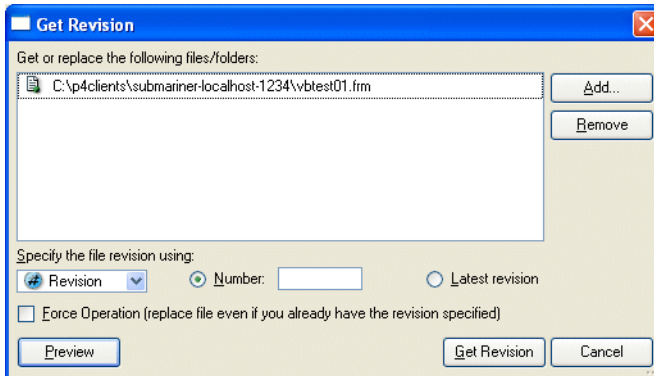
Checked out files are displayed with a red check mark, indicating that the files are writable.

Retrieving files from the depot

To get the most recent revision of a file from the depot, right-click the file and choose **Get Latest Version**.

To retrieve revisions prior to the head (latest) revision, you can sync from the **Revision History** dialog or perform the following steps:

1. Choose **Tools > Perforce > Get Latest Version...** The **Get Latest Version** dialog is displayed.
2. Check the files you want to retrieve and click **Advanced...** Dismiss the “Advanced sync dialog...” prompt by clicking **OK**, then click **OK** on the **Get Latest Version** dialog. The **Sync** dialog is displayed as shown in the following figure.



3. Specify the revision you want to retrieve, using a Perforce changelist number, label, version number, or date. For details about specifying revisions, refer to the *Perforce Command Reference*.
 - To preview the results of the operation with actually retrieving files, click **Preview**.
 - To retrieve a file from the depot if you’ve deleted your local copy manually, check **Force Sync**. By default, Perforce assumes you still have your copy and doesn’t retrieve it again.

4. Click **Sync..**

Note | If you retrieve a file that you already have checked out, for example, to obtain changes checked in by another user, Perforce will require you to resolve the file, to enable you to merge your changes with changes made by other users and to ensure that you don't inadvertently overwrite other changes.

Note | If you retrieve a file that you already have checked out, for example, to obtain changes checked in by another user, Perforce will require you to resolve the file, to enable you to merge your changes with changes made by other users and to ensure that you don't inadvertently overwrite other changes.

Checking files in

After editing checked-out files, you must check them into the depot. Your edited files become the head revision in the depot. (If any of your changes conflict with changes previously submitted by another user, the check-in fails and the differences must be resolved using Perforce.)

To check in files, perform the following steps:

1. Right-click and choose **Check in...** from the pop-up menu.

The **Check In Files to Perforce** dialog is displayed, listing the selected files.

2. Check the files you want to checked in and click **OK**.

The **Submit Changelist** form is displayed.

3. Enter your comments and click **OK**.

Your files are checked in. Checked-in files are displayed with lock icons.

To view all your pending changelists, click the **Manage Changelists** button on the **Submit Changelist** form. P4SCC displays a dialog listing your pending changelists. In this dialog, you can move files among changelists by dragging and dropping them.

Resolving file conflicts

If you encounter conflicts when checking in files, (for example, from multiple users working on the same files,) you must resolve the conflicts before you can submit the files. For example, if you and another user have changed the same file and the other user checked in changes before you, when you attempt to check in your changes, Perforce

notifies you that resolution is required by displaying the following message in the **Pending Changelist** dialog:

 **Out of date: Get latest revision to set up a resolve (this will not overwrite your copy of the file).**

To resolve file conflicts:

1. In the **Submit Changelist** dialog, Right-click the file that has conflicts and choose **Get Latest Revision**.

The file is now scheduled for resolve, indicated by the message “Resolve this file before submitting.”

2. Right-click the file and choose **Resolve Files...** The resolve dialog is displayed, describing the extent of the file differences and listing the following options:
 - *Accept Merge*: merge both files and accept the result, including both the changes you made and the changes made by other users.
 - *Accept Yours*: check your changes in, overwriting the most recently checked-in version.
 - *Accept Theirs*: discard your changes and preserve the other user’s changes.
 - *Run Merge Tool*: launch the merge utility so you can view the differences between your version and the most recently checked-in version. Using the merge utility, you can select individual passages or enter new changes to create the final version that you check in.
 - Other choices: you can view either file (**Open File...**), diff your file with the conflicting file, display the file’s revision history either textually or graphically.

After you choose how the files are resolved, the **Submit Changelist** dialog is redisplayed.

3. Enter a description of your changes and click **Submit**. Your changes are checked in.

Diffing files

To diff a file you are editing with the head revision in the depot, right-click the file and choose **Tools > Perforce > Show Differences...**

To diff two revisions of a file, perform the following steps:

1. Select the file and choose **Tools > Perforce > Show History...** The **History** dialog is displayed.
2. Click and drag one of the revisions you want to diff, and drop it on the other revision.

The diff utility is launched, displaying file differences.

Reverting files

To discard changes you've made to a checked-out file and reload the head revision from the depot, right-click the file and choose **Undo Check Out**.

Working with Visual Basic files

Which files do I put in the depot?

In general, add the project file (such as .vbproj files) but not the workspace file (.vbproj files), especially if multiple developers are working on the same project. For example, if a developer has a workspace file checked out for edit and another developer adds a file to the project, the change is not reflected in the workspace file. Exclude the following Visual Basic file types from Perforce control; Visual Basic caches and recreates them as required.

- .dca: active designer cache
- .oca: control typelib cache
- .vbg: group file

You can exclude them using Perforce protections or client views to prevent them from being inadvertently added to the depot.

Location of project files

The Project window displays source file names and their location on your local directory, but is only an approximate representation. Although the display simplifies project development in Visual Basic, it differs from the actual file structure in the following ways:

- The Project window groups source files by file type, such as Forms and Designers, but the local directory does not.
- The Project window does not display the workspace file (.vbproj file) and some other files.

For some source control tasks, you need to identify the actual names and locations of these files. For detailed information about Visual Basic project files, see:

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vbcon98/html/vbconformprojectfileformats.asp>

Recommended Performer file types

Following are recommended Performer file types for Visual Basic files. For details about Performer file types and attributes, refer to *Introducing Performer*.

File type	Perforce file type	Description
.bas	text	Basic file
.cls	text	class file
.ctl	text	control file
.ctx	binary	control binary file
.dsr	text+w	designer file
.dsx	binary	active designer binary file
.frm	text	form file
.frx	binary	form binary file
.vbg	text+w	group project
.vbp	text	project
.vbw	text+w	workspace

Checking out the project file

In Visual Basic 6.0 it is necessary to check out the project file (.vbp file) for edit to save *any* changes, including changes that affect only .frm files. If you do not want to submit an unchanged project file, right- click the project and choose **Undo checkout**. If multiple developers are working on the same project, keep the project file writable by setting its Performer file type to +w when you add it to the depot.

This chapter describes how to perform Perforce source code control tasks in the Visual Studio .NET environment. It assumes the Perforce SCC Plug-in is installed and that your client machine can communicate with the Perforce server. For details about configuring settings, see “Configuring IDEs with plug-ins” on page 12.

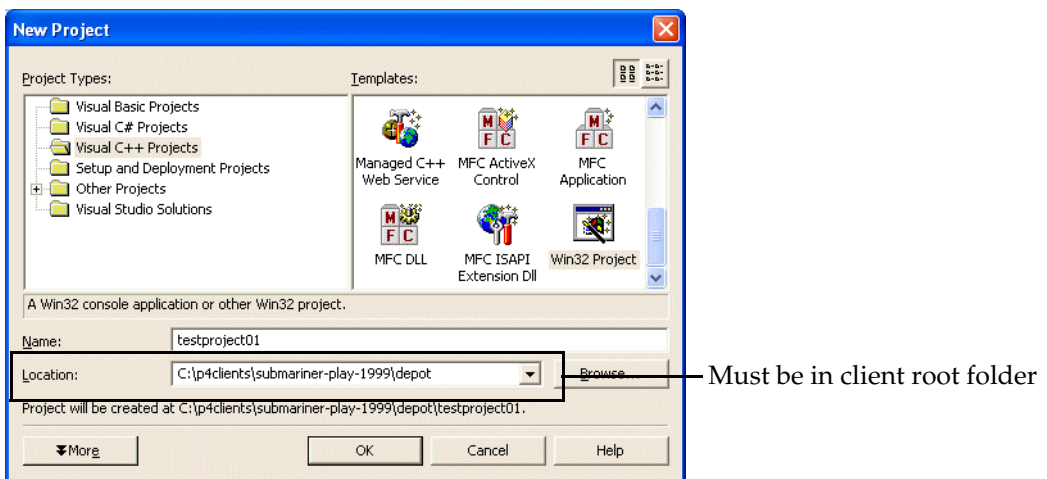
This screen illustrations in this chapter shows Visual C++, but the same dialogs are displayed regardless of which Visual Studio IDE you use.

Note The version control dialogs displayed for check in, check out, and undo checkout include comment fields, but these comment fields are not stored by the Perforce SCC Plug-in. To avoid displaying the check in dialog, choose **Check-in Now**.

Configuring Visual Studio .NET with Perforce

When you configure VS .NET with the Perforce SCC Plug-in, the settings are specific to the project you are working on, so no overall Perforce configuration is required for Visual Studio. The Perforce SCC Plug-in retains project settings from session to session and it is not necessary to re-enter them.

When you create a project you intend to manage with Perforce, specify a location that resides within the root folder of the workspace you intend to use as illustrated in the following figure.



Basic SCM tasks

This section tells you how to perform the following tasks.

- “Adding a project to the depot” on page 36.
- “Checking files out” on page 38
- “Retrieving files from the depot” on page 39
- “Checking files in” on page 40
- “Resolving file conflicts” on page 40
- “Diffing files” on page 41
- “Reverting files” on page 42
- “Miscellaneous tasks” on page 42

Also note that the **Exclude from source control** option is intended for files that, by nature, do not belong under source code control (such as build outputs).

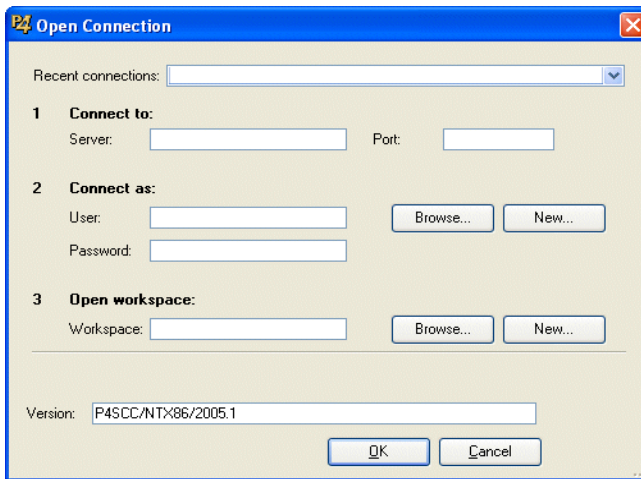
Adding a project to the depot

Make sure your Visual Studio .NET projects reside within your Perforce client workspace.

If you are adding a solution that contains a large number of projects, you can avoid specifying settings manually for every project by choosing the **Options > Connection** tab setting **Bind to the workspace that matches your Perforce environment settings** and setting `P4CLIENT` and `P4PORT` to the desired server and workspace.

To add a Visual Studio .NET project to your depot, perform the following steps:

1. In the Solution Explorer, right-click the project and choose **Source Control>Add Solution to Source Control**. If you enabled the **Show the Perforce Connections** option on the **Preferences > Connection** tab, the **Open Connection** dialog is displayed, as shown in the following figure.



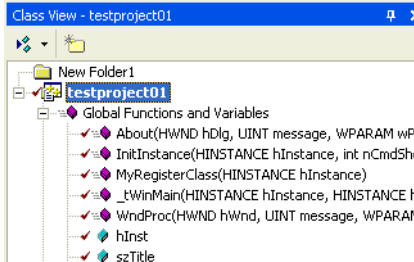
2. On the **Open Connection** dialog, enter the required settings as follows:
 - *Server*: the name of the host computer running the Perforce server in which you want to store the project.
 - *Port*: the port number on which the specified Perforce Server accepts commands.
 - *User*: the Perforce user name you want associated with the SCM operations you perform on the project. To choose from a list of users defined for the specified server, click **Browse**.
 - *Password*: the user's password, if any.
 - *Workspace*: the Perforce workspace associated with the SCM operations you perform on the project. You must save the project in a folder under the workspace root. To choose from a list of workspaces defined for the specified server, click **Browse**.
3. Click **OK** to dismiss the dialog.
4. In the Solution Explorer, right-click the solution and choose **Check In**.
5. The **Submit Changelist** dialog, listing files, is displayed.
6. Enter a description in the description field and click **Submit**. Your files are added to the depot. (Using Perforce, you can verify that the default changelist has been submitted and the files are now in the depot.)

Checked-in files are displayed with a lock icon. If you attempt to edit a file that is not checked out, a check-out dialog is displayed.

Checking files out

To check out files:

1. In the Solution Explorer pane, right-click the desired file (or the project, if you want to check out all its files) and choose **Check Out...** The **Check Out** dialog is displayed.
2. Click **OK**. In the Solution Explorer pane, files are displayed with a check mark, as shown in the following figure.

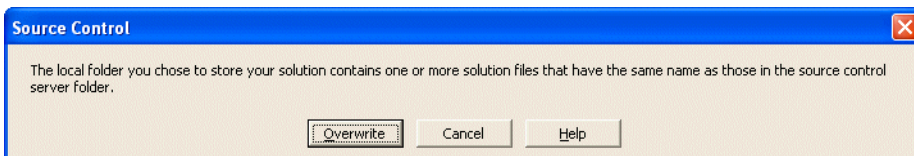


The checked-out files are listed in the **Pending Checkins** pane.

If you branch a project using Perforce, the Visual Studio bindings must be corrected to correspond to the project's new location. To correct bindings for a branched solution, you must perform the following steps the first time you check out the branched project.

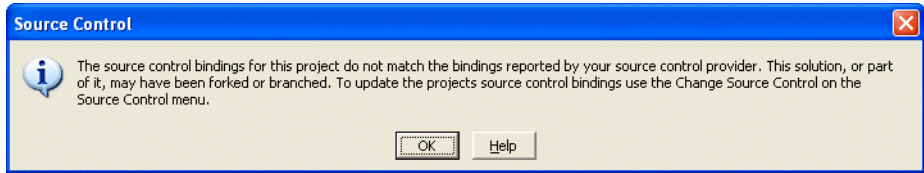
To check out a newly-branched project (first time only):

1. Choose **File> Source Control > Open from Source Control...** The **Open Connection** dialog is displayed.
2. Specify connection settings for the server that contains the project you want to check out and click **OK**. The **Open Project** dialog is displayed.
3. Browse to the branched project you want to open and double-click its **.sln** file.
4. If the newly-branched files are present in your workspace, Visual Studio displays the following dialog:



5. Click **Overwrite**. Visual Studio displays the **Open Solution** dialog.

6. Double-click the `.sln` file. Visual Studio displays the following dialog.



7. Click **OK**.

The project now contains correct binding information.

Retrieving files from the depot

To get the most recent revision of a file from the depot, right-click the file in the Solution Explorer pane and choose **Get Latest Version**.

To retrieve revisions prior to the head (latest) revision, you can sync from the **Revision History** dialog or perform the following steps:

1. In the Solution Explorer pane, click the file.
2. Choose **File > Source Control > History**. The **History** dialog is displayed, listing the revisions in the depot.
3. Right-click the desired revision and choose **Get Revision**.

Note | If you retrieve a file that you already have checked out, for example, to obtain changes checked in by another user, Perforce requires you to resolve the file, to ensure that you don't inadvertently overwrite other users' changes.

To sync all files for a project that resides in the depot:

1. Choose **File > Source Control > Open from Source Control...** The **Open Connection** dialog is displayed.
2. Enter the settings that specify the server where the project is stored and the client workspace and user with which you want to check the project files out, and click **OK**. The **Perforce Open Project** dialog is displayed.
3. Browse to the `.sln` file for the project you want to sync and click **OK**. The project files are synced to your workspace.

Checking files in

After editing checked-out files, you must check them into the depot. Your edited files become the head revision in the depot. (If your changes conflict with changes previously submitted by another user, the check-in fails and the differences must be resolved using Perforce.).

To check in files, perform the following steps:

1. In the Solution Explorer pane, right-click the file and choose **Check In Now**. The **Submit Changelist** dialog is displayed.
2. Enter a description of your changes and click **Submit**.

In the Solution Explorer pane, the check marks are replaced by locks, indicating that the submitted files were checked in.

Note To produce meaningful change history, check in related files together. For example, if you changed two different files to fix a bug, check them both in at the same time. The files are submitted in the same changelist.

To view all your pending changelists, click the **Manage Changelists** button on the **Submit Changelist** form. P4SCC displays a dialog listing your pending changelists. In this dialog, you can move files among changelists by dragging and dropping them.

Resolving file conflicts

If you encounter conflicts when checking in files, (for example, from multiple users working on the same files,) you must resolve the conflicts before you can submit the files. For example, if you and another user have changed the same file and the other user checked in changes before you, when you attempt to check in your changes, Perforce notifies you that resolution is required by displaying the following message in the **Pending Changelist** dialog:

 Out of date: Get latest revision to set up a resolve (this will not overwrite your copy of the file).

To resolve file conflicts:

1. Right-click the file that has conflicts and choose **Get Latest Revision**.
The file is now scheduled for resolve, indicated by the message "Resolve this file before submitting."
2. Right-click the file and choose **Resolve Files...** The resolve dialog is displayed, describing the extent of the file differences and listing the following options:
 - *Accept Merge*: merge both files and accept the result, including both the changes you made and the changes made by other users.
 - *Accept Yours*: check your changes in, overwriting the most recently checked-in version.
 - *Accept Theirs*: discard your changes and preserve the other user's changes.
 - *Run Merge Tool*: launch the merge utility so you can view the differences between your version and the most recently checked-in version. Using the merge utility, you can select individual passages or enter new changes to create the final version that you check in.
 - Other choices: you can view either file (**Open File...**), diff your file with the conflicting file, display the file's revision history either textually or graphically.After you choose how the files are resolved, the **Submit Changelist** dialog is redisplayed.
3. Enter a description of your changes and click **Submit**. Your changes are checked in.

Diffing files

To diff a file you are editing with the head revision in the depot, right-click the file and choose **Compare Versions...**

To diff two revisions of a file, perform the following steps:

1. In the Solution Explorer pane, right-click the file and choose **File > History**. The **History** dialog is displayed, listing the revisions in the depot.
2. To diff two revisions drag one revision to the other.

The diff utility is launched, displaying file differences.

Reverting files

To discard changes you've made to a checked-out file and reload the head revision from the depot:

1. In the Solution Explorer pane, right-click the file and choose **Undo Check Out...** The **Undo Check Out** dialog is displayed, listing the files to be reverted.
2. Check the files you want to revert and click **Undo Checkout**.

In the Solution Explorer pane, the reverted files are displayed with a lock icon, indicating that they are no longer checked out. The head revision is copied from the depot to your workspace, overwriting any changes you made to the workspace file.

You can also revert unchanged files from the **Pending Changelist** dialog.

Which files do I put in the depot?

Visual Studio .NET adds all appropriate files when the project is first added to source control, so, in general, don't change the contents of the changelist that is created when you choose **Add to Source Control**.

Put `.sln` files in the depot. Do not put the following files in the depot. .

- `.idb`
- `.ncb`
- `.obj`
- `.opt`
- `.pch`
- `.plg`

These files are managed by Visual Studio .NET separately for each client computer

Miscellaneous tasks

To view the Perforce commands issued by the plug-in, choose **View>Other Windows>Output** and display the **Source Control** pane.

This chapter describes how to perform Perforce source code control tasks in the Visual Studio 2005 environment. It assumes the Perforce SCC Plug-in is installed and that your client machine can communicate with the Perforce server. For details about configuring settings, see “Configuring IDEs with plug-ins” on page 12.

This screen illustrations in this chapter shows Visual C++, but the same dialogs are displayed regardless of which Visual Studio IDE you use.

Note | The version control dialogs displayed for check in, check out, and undo checkout include comment fields, but these comment fields are not stored by the Perforce SCC Plug-in. To avoid displaying the check in dialog, choose **Check-in Now**.

When you configure Visual Studio 2005 with the Perforce SCC Plug-in, the settings are specific to the project you are working on, so no overall Perforce configuration is required for Visual Studio.

Basic SCM tasks

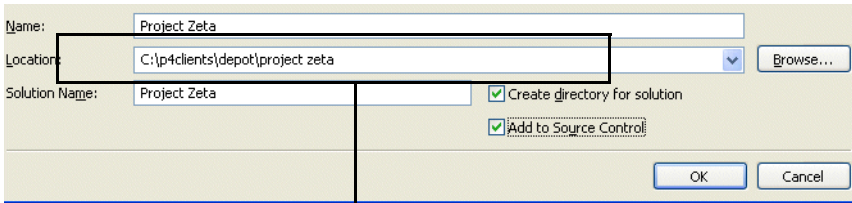
This section tells you how to perform the following tasks.

- “Adding a project to the depot” on page 44.
- “Checking files out” on page 45
- “Retrieving files from the depot” on page 46
- “Checking files in” on page 47
- “Resolving file conflicts” on page 47
- “Diffing files” on page 48
- “Reverting files” on page 49
- “Miscellaneous tasks” on page 49

Also note that the **Exclude from source control** option is intended for files that, by nature, do not belong under source code control (such as build outputs).

Adding a project to the depot

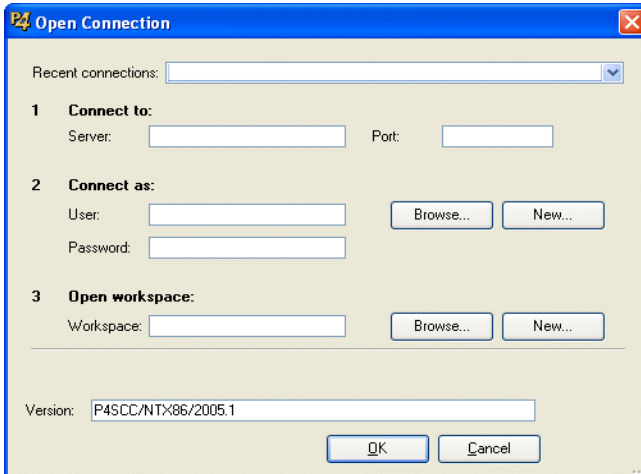
When you save a project that you intend to manage with Perforce, specify a location that resides within the root folder of the workspace you intend to use as illustrated in the following figure.



Must be in client root folder

To add a Visual Studio 2005 project to your depot, perform the following steps:

1. In the Solution Explorer, right-click the project and choose **Source Control>Add Solution to Source Control**. The **Open Connection** dialog is displayed, as shown in the following figure.



2. On the **Open Connection** dialog, enter the required settings as follows:
 - *Server*: the name of the host computer running the Perforce server in which you want to store the project.
 - *Port*: the port number on which the specified Perforce Server accepts commands.

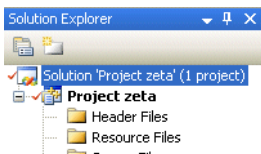
- *User*: the Perforce user name you want associated with the SCM operations you perform on the project. To choose from a list of users defined for the specified server, click **Browse**.
 - *Password*: the user's password, if any.
 - *Workspace*: the Perforce workspace associated with the SCM operations you perform on the project. You must save the project in a folder under the workspace root. To choose from a list of workspaces defined for the specified server, click **Browse...**
3. Click **OK** to dismiss the dialog. P4SCC puts the files into a Perforce changelist, ready to add to the depot, as indicated by a red check mark displayed in the Solution Explorer.
 4. In the Solution Explorer, right-click the solution and choose **Check In Now**. The **Submit Changelist** dialog, listing files, is displayed.
 5. Enter a description in the description field and click **Submit**. Your files are added to the depot. (Using Perforce, you can verify that the default changelist has been submitted and the files are now in the depot.)

Checking files out

Files that are not checked out are displayed with a lock icon in the Solution Explorer.

To check out files:

In the Solution Explorer pane, right-click the desired file (or the project, if you want to check out all its files) and choose **Check Out for Edit Now**. In the Solution Explorer pane, files are displayed with a check mark, as shown in the following figure.



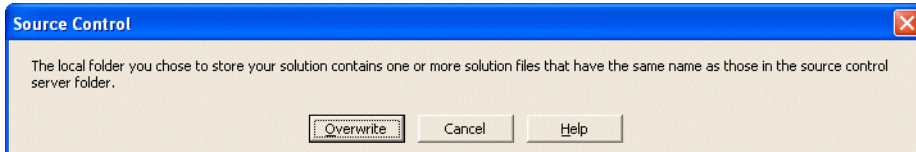
The checked-out files are listed in the **Pending Checkins** pane.

If you branch a project using Perforce, the Visual Studio bindings must be corrected to correspond to the project's new location. To correct bindings for a branched solution, you must perform the following steps the first time you check out the branched project.

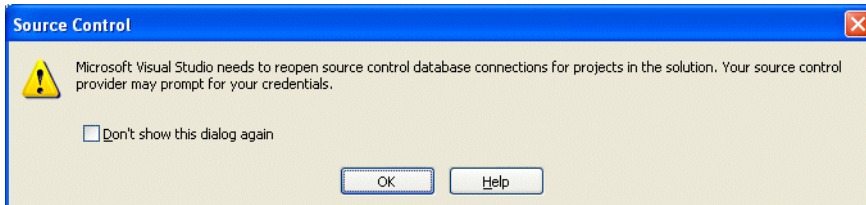
To check out a newly-branched project (first time only):

1. Choose **File> Source Control > Open from Source Control....** The **Open Connection** dialog is displayed.

2. Specify connection settings for the server that contains the project you want to check out and click **OK**. The **Open Project** dialog is displayed.
3. Browse to the branched project you want to open and double-click its `.sln` file.
4. If the newly-branched files are present in your workspace, Visual Studio displays the following dialog:



5. Click **Overwrite**. Visual Studio displays the **Open Solution** dialog.
6. Double-click the `.sln` file. Visual Studio displays the following dialog.



7. Click **OK**.

The project now contains correct binding information.

Retrieving files from the depot

To get the most recent revision of a file from the depot, right-click the file in the Solution Explorer pane and choose **Get Latest Version**.

To retrieve revisions prior to the head (latest) revision, perform the following steps:

1. In the Solution Explorer pane, click the file.
2. Choose **File > Source Control > View History**. The **History** dialog is displayed, listing the revisions in the depot.
3. Right-click the desired revision and choose **Get Revision**.

Note | If you retrieve a file that you already have checked out, for example, to obtain changes checked in by another user, Perforce requires you to resolve the file, to ensure that you don't inadvertently overwrite other users' changes.

To sync all files for a project that resides in the depot:

1. Choose **File > Source Control > Open from Source Control...** The **Open Connection** dialog is displayed.
2. Enter the settings that specify the server where the project is stored and the client workspace and user with which you want to check the project files out, and click **OK**. The **Perforce Open Project** dialog is displayed.
3. Browse to the `.sln` file for the project you want to sync and click **OK**. The project files are synced to your workspace.

Checking files in

After editing checked-out files, you must check them into the depot. Your edited files become the head revision in the depot. (If your changes conflict with changes previously submitted by another user, the check-in fails and the differences must be resolved using Perforce.).

To check in files, perform the following steps:

1. In the Solution Explorer pane, right-click the file and choose **Check In Now**. The **Submit Changelist** dialog is displayed.
2. Enter a description of your changes and click **Submit**.

In the Solution Explorer pane, the check marks are replaced by locks, indicating that the submitted files were checked in.

Note	To produce meaningful change history, check in related files together. For example, if you changed two different files to fix a bug, check them both in at the same time. The files are submitted in the same changelist.
-------------	---

To view all your pending changelists, click the **Manage Changelists** button on the **Submit Changelist** form. P4SCC displays a dialog listing your pending changelists. In this dialog, you can move files among changelists by dragging and dropping them.

Resolving file conflicts

If you encounter conflicts when checking in files, (for example, from multiple users working on the same files,) you must resolve the conflicts before you can submit the files. For example, if you and another user have changed the same file and the other user checked in changes before you, when you attempt to check in your changes, Perforce

notifies you that resolution is required by displaying the following message in the **Pending Changelist** dialog:

 Out of date: Get latest revision to set up a resolve (this will not overwrite your copy of the file).

To resolve file conflicts:

1. Right-click the file that has conflicts and choose **Get Latest Revision**.
The file is now scheduled for resolve, indicated by the message “Resolve this file before submitting.”
2. Right-click the file and choose **Resolve Files...** The **Resolve** dialog is displayed, describing the differences between the file revisions and listing the following options:
 - *Accept Merge*: merge both files and accept the result, including both the changes you made and the changes made by other users.
 - *Accept Yours*: check your changes in, overwriting the most recently checked-in version.
 - *Accept Theirs*: discard your changes and preserve the other user’s changes.
 - *Run Merge Tool*: launch the merge utility so you can view the differences between your version and the most recently checked-in version. Using the merge utility, you can select individual passages or enter new changes to create the final version that you check in.
 - Other choices: you can view either file (**Open File...**), diff your file with the conflicting file, display the file’s revision history either textually or graphically.
3. Click the desired resolve option. (If you choose to merge, the merge utility is displayed. Merge and exit, saving your changes.) The **Submit Changelist** dialog is redisplayed.
4. Enter a description of your changes and click **Submit**. Your changes are checked in.

Diffing files

To diff a file you are editing with the head revision in the depot, right-click the file and choose **Compare...**

To diff two revisions of a file, perform the following steps:

1. In the Solution Explorer pane, right-click the file and choose **File > View History**. The **History** dialog is displayed, listing the revisions in the depot.

2. To diff two revisions, drag one revision to the other. The diff utility is launched, displaying file differences.

Reverting files

To discard changes you've made to a checked-out file and reload the head revision from the depot:

1. In the Solution Explorer pane, right-click the file and choose **Undo Check Out...**. The **Undo Check Out** dialog is displayed, listing the files to be reverted.
2. Check the files you want to revert and click **Undo Checkout**.

In the Solution Explorer pane, the reverted files are displayed with a lock icon, indicating that they are no longer checked out. The head revision is copied from the depot to your workspace, overwriting any changes you made to the workspace file.

You can also revert unchanged files from the **Pending Changelist** dialog.

Which files do I put in the depot?

Visual Studio 2005 adds all appropriate files when the project is first added to source control, so, in general, don't change the contents of the changelist that is created when you choose **Add to Source Control**.

Miscellaneous tasks

To view the Perforce commands issued by the plug-in, choose **View > Output** and display the **Source Control** pane.

Index

A

- .asp files 11
- .avi files 11

B

- .bas files 33
- bindings 38, 45
- .bmp files 11
- branched projects 38, 45
- .btr files 11

C

- changelist
 - defined 10
- client view
 - defined 15
- client workspace
 - defined 15
- .cls files 33
- .clw files 25
- .cnf files 11
- config files 16
- conflicts 10
- .cpp files 25
- .css files 11
- .ctl files 33
- .ctx files 33

D

- .dca files 32
- depot
 - defined 9
- diffing files 11
- .doc files 11
- .dot files 11
- .dsp files 25, 32
- .dsr files 33
- .dsw files 25
- .dsx files 33

E

- .exp files 11

F

- file types 11
- files
 - .asp 11
 - .avi 11
 - .bas 33
 - .bmp 11
 - .btr 11
 - .cls 33
 - .clw 25
 - .cnf 11
 - .cpp 25
 - .css 11
 - .ctl 33
 - .ctx 33
 - .dca 32
 - .doc 11
 - .dot 11
 - .dsp 25, 32
 - .dsr 33
 - .dsw 25
 - .dsx 33
 - .exp 11
 - .frm 33
 - .frx 33
 - generated 25
 - .gif 11
 - .htm 11
 - .html 11
 - .ico 11
 - .inc 11
 - .ini 11
 - .jpg 11
 - .js 11
 - .lib 11
 - .log 11
 - .mpg 11
 - .ncb 25
 - .oca 32

- .pdf 11
- .pdm 11
- .ppt 12
- .rc 25
- .sln 42
- .suo 42
- .vbg 32, 33
- .vbp 33
- .vbw 32, 33
- .xls 12
- .zip 12
- .frm files 33
- .frx files 33
- G**
- generated files 25
- .gif files 11
- global settings 14
- H**
- head revision
 - defined 10
- .htm files 11
- .html files 11
- I**
- .ico files 11
- .inc files 11
- .ini files 11
- J**
- .jpg files 11
- .js files 11
- L**
- .lib files 11
- .log files 11
- M**
- Macintosh files 12
- .mpg files 11
- N**
- .ncb files 25
- O**
- .oca files 32
- P**
- P4CONFIG 16
- .pdf files 11
- .pdm files 11

- Perforce file types 11
- permissions
 - read-write 10, 29
- .ppt files 12
- project file 32, 33
- R**
- .rc files 25
- S**
- .sln files 42
- .suo files 42
- T**
- terminology for SCM tasks 9
- V**
- .vbg files 32, 33
- .vbp files 33
- .vbw files 32, 33
- X**
- .xls files 12
- Z**
- .zip files 12