TortoiseCVS User's Guide

Version 1.9.14

Ben Campbell
Martin Crawford
Hartmut Honisch
Francis Irving
Torsten Martinsen
Ian Dees

TortoiseCVS User's Guide: Version 1.9.14

by Ben Campbell, Martin Crawford, Hartmut Honisch, Francis Irving, Torsten Martinsen, and Ian Dees

Copyright © 2001 - 2006 TortoiseCVS

Table of Contents

List of Figures

List of Tables

Chapter 1. Getting Started

What is CVS?

CVS, or the Concurrent Versioning System, is a version control system. Version control systems are generally used as part of the software development cycle to track and co-ordinate source code changes among a team of developers.

For example, bugs sometimes creep in when software is modified, and you might not detect the bug until a long time after you make the modification. With CVS, you can easily retrieve old versions to see exactly which change caused the bug. This can sometimes be a big help.

You could of course save every version of every file you have ever created. This would however waste an enormous amount of disk space. CVS stores all the versions of a file in a single file in a clever way that only stores the differences between versions.

CVS also helps you if you are part of a group of people working on the same project. It is all too easy to overwrite each others' changes unless you are extremely careful. Some editors, like GNU Emacs, try to make sure that the same file is never modified by two people at the same time. Unfortunately, if someone is using another editor, that safeguard will not work. CVS solves this problem by insulating the different developers from each other. Every developer works in his own sandbox, and CVS merges the work when each developer is done.

What is TortoiseCVS?

TortoiseCVS is a front-end client to make using CVS easier and more intuitive. It allows developers to work with files controlled by CVS directly from Windows Explorer™.

One of the major drawbacks of CVS is the command-line interface that is provided. Many developers today are becoming more accustomed to the graphical integrated development environments (IDEs). TortoiseCVS aims to provide that "point-and-click" environment in a clever and intuitive way.

Note that TortoiseCVS is a CVS *client*, not a *server*. This document assumes that you either know how to set up your own server, or that you are using a server that somebody else set up. If you want to learn about setting up a CVS server, see CVS Book [58].

Where to Begin?

1. The best way to learn how to use TortoiseCVS is to play with it. Start by installing TortoiseCVS.
2. If you are new to CVS start following along with Basic Usage of TortoiseCVS.
3. Once you've learned the ropes check out the advanced features in Advanced Usage of TortoiseCVS.
4. For pure reference, the chapters Command Reference for TortoiseCVS and Dialog Reference for TortoiseCVS will help you get the most out of TortoiseCVS.
5. And finally check out Articles, Tips and Tricks for a complete and enjoyable version control experience.

Errors and Omissions in this Manual

This manual is not perfect, although we try to make it better all the time. If you find something which you believe is wrong, or if you think something is missing from the manual, please let us know [https://sourceforge.net/tracker/?func=add&group_id=48103&atid=451972]. (Please set the **Category** to `Documentation`; you do not need to include your OS version and CVSROOT).

Chapter 2. Basic Usage of TortoiseCVS

Sandboxes

CVS has a unique method of working from most other version control systems in that developers can edit the same files concurrently. First you *Checkout* a version of the source code from the repository into a local copy on your computer. This local copy is called a *sandbox*.

You then simply edit the files that you want to change. You can *Add* new files or remove files you no longer require. When you're done you *Commit* the changes to the repository.

If someone else has changed the same file while you were working on it, then the commit will fail. You must then *Update* all your source code files from the repository. This will automatically merge the other developers changes into your copy of the file.

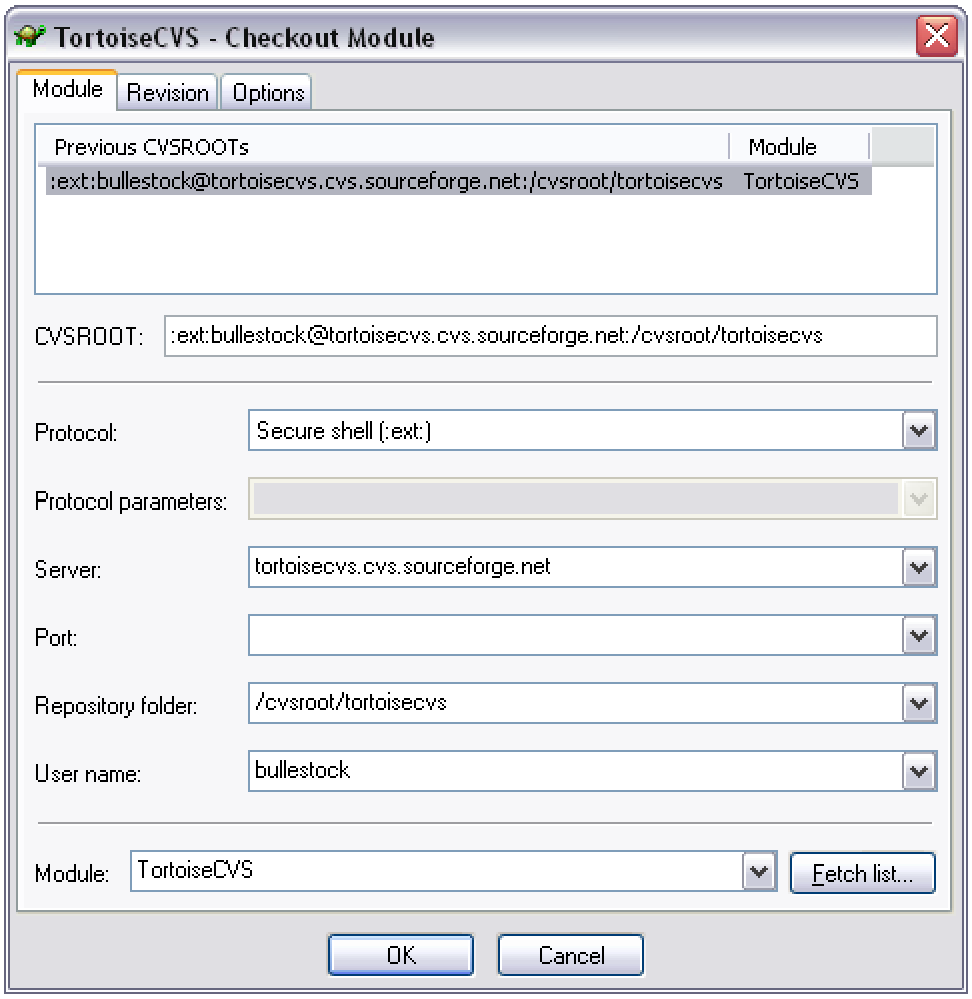
Sometimes CVS cannot do this automatically, for example if you both changed the same line of code. This is called a *Conflict*. Conflicts happen much less often than you might expect. CVS puts both versions of the conflicting code in the file, with markings separating them. Then you manually edit the file to resolve the conflict before you can commit the changes.

This method of working has lots of advantages. Each developer lives in a sandbox. Changes that another developer makes are isolated from you until you want to check in your changes. It stops bottlenecks where people cannot do things because someone else has the file checked out. Any developer can work on files without direct access to the server, they only need to connect to update or commit.

Checking out a Module

To obtain a module from the CVS server for the first time is known as a checkout. Checking a module out from the repository creates a local sandbox of the module.

Figure 2.1. Checkout Dialog



Previous CVSROOTs	Module
:ext:bullestock@tortoisecvs.cvs.sourceforge.net:/cvsroot/tortoisecvs	TortoiseCVS

CVSROOT: :ext:bullestock@tortoisecvs.cvs.sourceforge.net:/cvsroot/tortoisecvs

Protocol: Secure shell (:ext:)

Protocol parameters:

Server: tortoisecvs.cvs.sourceforge.net

Port:

Repository folder: /cvsroot/tortoisecvs

User name: bullestock

Module: TortoiseCVS

To perform a checkout, Right-click on the folder where you would like the module placed, and pick **CVS Checkout...** from the pop-up menu. The Checkout Dialog will appear with the following fields:

- **Protocol** The protocol to use when communicating to the remote CVS repository.
- **Protocol parameters** This field is only enabled for certain protocols, currently `:sserver:` and `:sspi:`. With certain combinations of different CVS client and server versions, or with restrictive firewalls, it may be necessary to enter additional information here in order to be able to connect to the CVS server. Your CVS server administrator will know whether this is necessary.

  Note that for the `:ext:` protocol it is not possible to specify additional parameters here; this is a technical limitation due to the way that `:ext:` is designed. Instead, those parameters can be set in the Preferences dialog, on the Tools tab.
- **Server** The name of the server hosting the remote CVS repository.
- **Port** The port for the remote CVS repository. Not usually required.
- **Repository Directory** The location of the CVS repository on the remote server.
- **User name** The username of your account for the CVS repository.
- **CVSROOT** The full connection string, comprised of the above fields. Often you will be given the connection string in this format, in which can just paste it in here.

- **Module** The name of the module you want to checkout. The module name is case sensitive.

You will need to know this information in advance. Most projects should have some documentation (often online) on how to connect and checkout their modules. Additionally, your software lead, or project manager should be able to provide this information to you.

An exception to this is the module name - depending on how your CVS server is setup, you might find the module name in the dropdown after clicking **Fetch list...** (for details, see How 'Fetch list...' finds the list of modules).

Note: It is much easier to choose your project checkout settings now than it is to move your project to a different server or protocol later. For example, if your server offers two different CVS protocols, make sure you pick the one that best suits your needs.

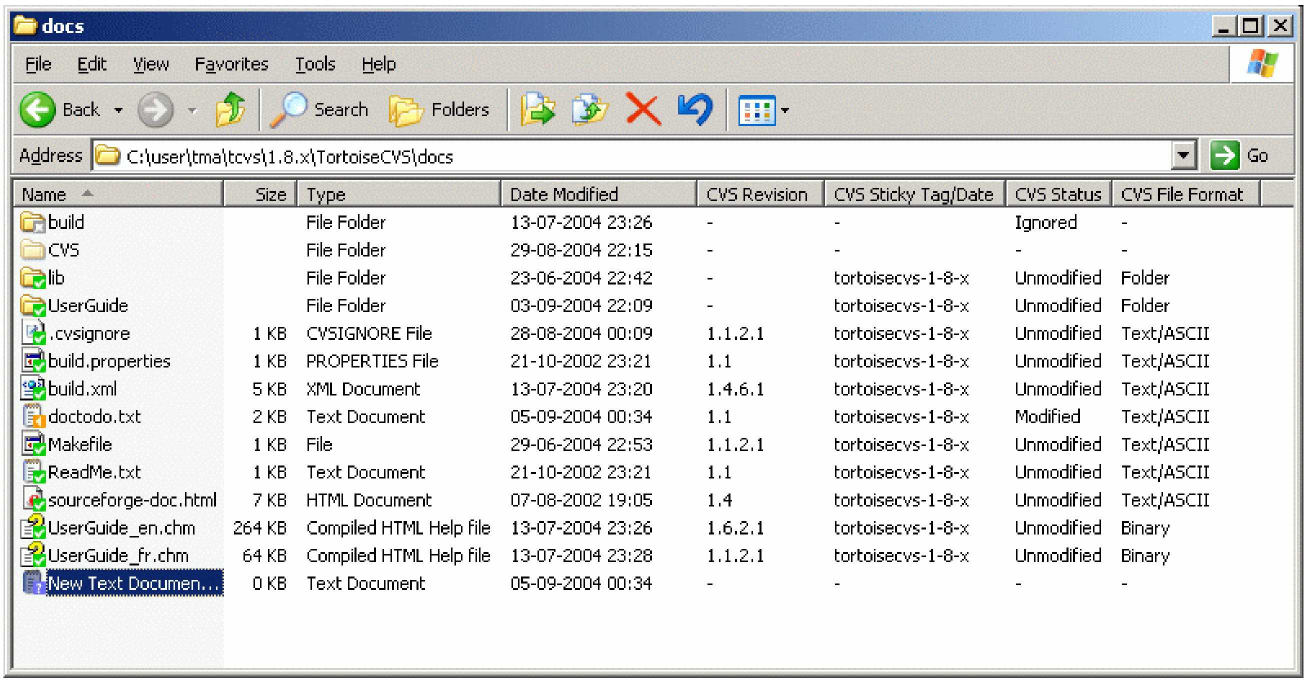
A folder named after the module will be created within the folder you checkout to, so you can keep all your checked out modules in the same folder, even if they are for different projects.

For more information see Obtaining a Working Copy: CVS Checkout....

Windows Explorer and TortoiseCVS

Having checked out a module, we can now *explore* how TortoiseCVS works with Windows™. You will notice that your files appear in Windows Explorer with different icon overlays. Additionally, if you are using either Windows 2000™ or Windows XP™ you can also see new CVS columns in the *Detail View* of Windows Explorer™ (see Showing More Information: CVS Explorer Columns).

Figure 2.2. TortoiseCVS and Windows Explorer



C:\user\tma\tcvs\1.8.x\TortoiseCVS\docs

Name	Size	Type	Date Modified	CVS Revision	CVS Sticky Tag/Date	CVS Status	CVS File Format
build		File Folder	13-07-2004 23:26	-	-	Ignored	-
CVS		File Folder	29-08-2004 22:15	-	-	-	-
lib		File Folder	23-06-2004 22:42	-	tortoisecvs-1-8-x	Unmodified	Folder
UserGuide		File Folder	03-09-2004 22:09	-	tortoisecvs-1-8-x	Unmodified	Folder
.cvsignore	1 KB	CVSIGNORE File	28-08-2004 00:09	1.1.2.1	tortoisecvs-1-8-x	Unmodified	Text/ASCII
build.properties	1 KB	PROPERTIES File	21-10-2002 23:21	1.1	tortoisecvs-1-8-x	Unmodified	Text/ASCII
build.xml	5 KB	XML Document	13-07-2004 23:20	1.4.6.1	tortoisecvs-1-8-x	Unmodified	Text/ASCII
doctodo.txt	2 KB	Text Document	05-09-2004 00:34	1.1	tortoisecvs-1-8-x	Modified	Text/ASCII
Makefile	1 KB	File	29-06-2004 22:53	1.1.2.1	tortoisecvs-1-8-x	Unmodified	Text/ASCII
ReadMe.txt	1 KB	Text Document	21-10-2002 23:21	1.1	tortoisecvs-1-8-x	Unmodified	Text/ASCII
sourceforge-doc.html	7 KB	HTML Document	07-08-2002 19:05	1.4	tortoisecvs-1-8-x	Unmodified	Text/ASCII
UserGuide_en.chm	264 KB	Compiled HTML Help file	13-07-2004 23:26	1.6.2.1	tortoisecvs-1-8-x	Unmodified	Binary
UserGuide_fr.chm	64 KB	Compiled HTML Help file	13-07-2004 23:28	1.1.2.1	tortoisecvs-1-8-x	Unmodified	Binary
New Text Documen...	0 KB	Text Document	05-09-2004 00:34	-	-	-	-

The icon overlays indicate a file or folder's status in CVS. The following figure indicates the corresponding status for each icon[1]:

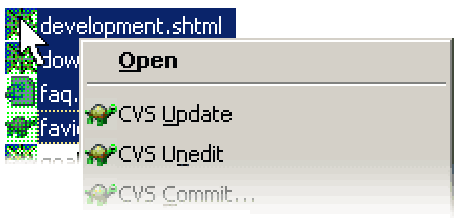
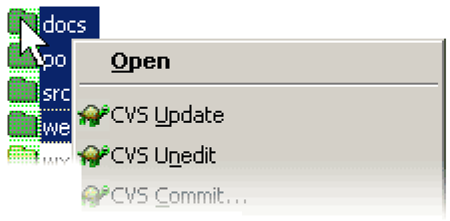
[1]The above table shows the default icon overlays; TortoiseCVS may be configured to use alternative icon sets — see Overlay Icons.

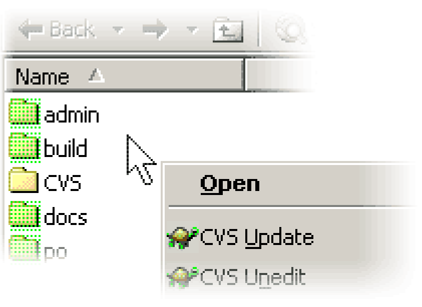
Table 2.1. Icon Overlays

✓	*Unmodified*	The file or folder is up-to-date with the CVS repository version.
←	*Modified*	The file or folder has been modified from the current CVS repository version.
+	*Added*	The file or folder has been added to CVS, but not yet committed.
!	*Conflict*	The file or folder has a conflict with the current CVS repository version.
?	*Not In CVS*	The file or folder is not in the CVS repository.
🔒	*CVS Watch*	The file or folder is controlled under CVS Watch.
✗	*Ignored*	The file or folder is being ignored by CVS.

You interact with TortoiseCVS by right-clicking within Windows Explorer and choosing CVS operations from the context menu. Which files and folders the operation is performed on depends on what you have selected and where you have clicked:

Table 2.2. Right-Click Context

 development.shtml dow faq. favi Open CVS Update CVS Unedit CVS Commit...	One or more selected files: The operation is performed on those files.
 docs po src we Open CVS Update CVS Unedit CVS Commit...	One or more selected folders: The operation is performed on these folders and the files and folders contained within.

 Back Name admin build CVS docs po Open CVS Update CVS Unedit	Anywhere else within the Explorer pane, the operation is performed on all files and folders within the view.

Total Commander and TortoiseCVS

The popular Total Commander™ (previously known as Windows Commander) file manager also works with TortoiseCVS. We recommend that you go into the options menu and configure the right mouse button for Windows standard operation.

By default, Total Commander does not display overlay icons and custom columns. You can, however, install the third-party plugin ShellDetails™ from http://lefteous.totalcmd.net/tc/shelldetails_eng.htm to get this feature.

You need to tell ShellDetails which folders it should be active for: Open the file: `c:\WINDOWS\ShellDetails.ini` and insert the following two lines, where dir_1 is the location of your sandbox:

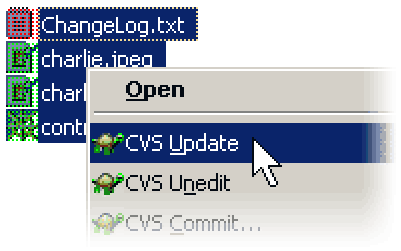
```
[Directories]
Dir_1=C:\sandbox
```

Finally, inside Total Commander go to **Configuration** → **Options** → **Display** and select the check box **Show overlay icons, e.g. for links**.

Updating your Sandbox

Occasionally you may want changes done by others to get incorporated in your local working copy. The process of getting changes from the server to your local copy is known as Updating. Updating may be done on single files, a set of selected files, or recursively on entire folder hierarchies. To update, highlight the files and/or folders you want, right-click and select **CVS Update**. The Progress Dialog will pop up displaying the progress of the update as it runs.

Figure 2.3. Updating Files and Folders



Changes done by others will be merged into your files, keeping any changes you may have done to the same files. The repository is *not* affected by performing an update.

If you receive reports of conflicts during the update, please read Resolving Conflicts.

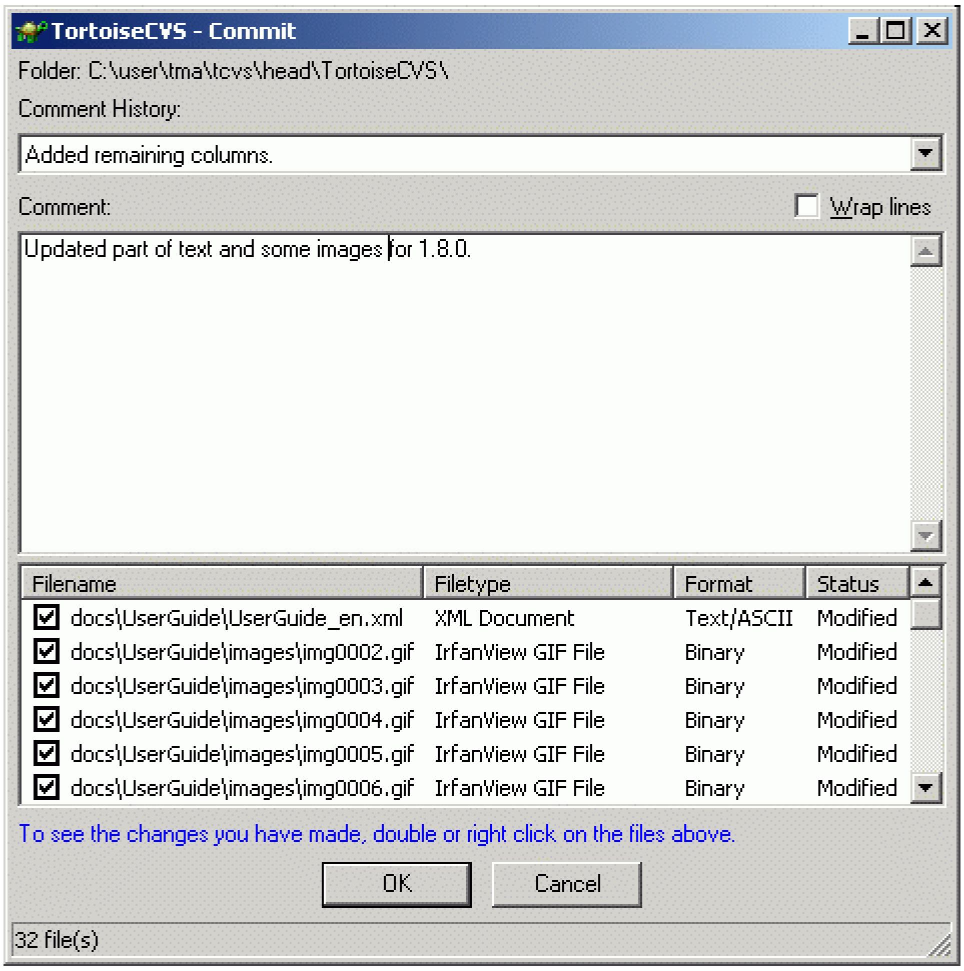
For more information see Getting Other People's Changes: CVS Update.

Committing your Changes to the Repository

Making local modifications available in the repository is known as *committing* the changes. Before committing, you should do an update to make sure there are no conflicts (see Updating your Sandbox).

To commit your changes start by selecting the file(s) or folder(s) that you want to commit. Right-click on the selection, and choose the **CVS Commit...** menu item.

Figure 2.4. Commit Dialog



Folder: C:\user\tma\tcvs\head\TortoiseCVS\

Added remaining columns.

Updated part of text and some images for 1.8.0.

	Filename	Filetype	Format	Status
☑	docs\UserGuide\UserGuide_en.xml	XML Document	Text/ASCII	Modified
☑	docs\UserGuide\images\img0002.gif	IrfanView GIF File	Binary	Modified
☑	docs\UserGuide\images\img0003.gif	IrfanView GIF File	Binary	Modified
☑	docs\UserGuide\images\img0004.gif	IrfanView GIF File	Binary	Modified
☑	docs\UserGuide\images\img0005.gif	IrfanView GIF File	Binary	Modified
☑	docs\UserGuide\images\img0006.gif	IrfanView GIF File	Binary	Modified

To see the changes you have made, double or right click on the files above.

32 file(s)

You will then be presented with the Commit Dialog where you can enter a brief summary of what was changed. You can also exclude changed files from the commit by unchecking their checkboxes. Once you are satisfied with what you are committing, click the **OK** on the dialog to go ahead with the commit operation.

You can easily forget to enter a comment before selecting **OK**. To prevent this, TortoiseCVS has an option (set in the Preferences dialog, on the Policy tab) that, when enabled, asks for confirmation before committing without a comment.

Please note that committing changes will not automatically add new files that you have created to the repository. See Adding Files and Directories to the Repository on how to add files.

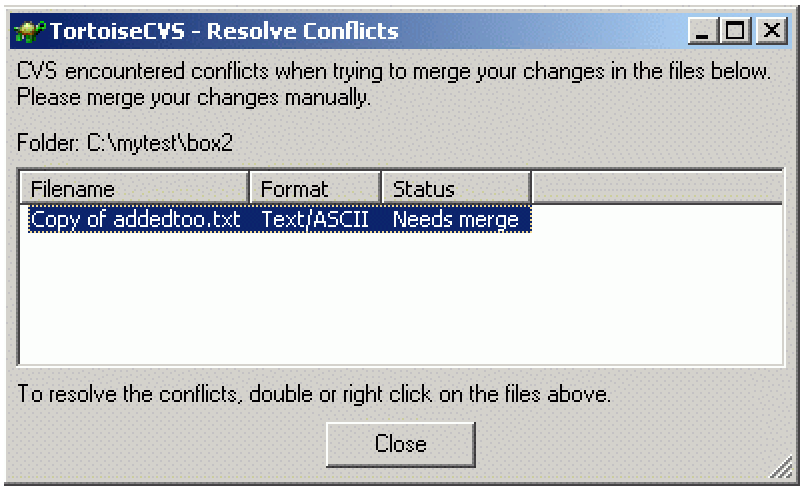
For more information see Making Your Changes Available to Others: CVS Commit....

Resolving Conflicts

Sometimes, the CVS server will report a conflict when you update your files from the repository. A conflict occurs when two or more developers have changed the same lines of a file. As CVS knows nothing of your project, it leaves resolving the conflicts to the developers.

Each conflicting file is marked with a "C" in the progress dialog (see Progress Dialog). When the update is complete, the Resolve Conflicts dialog is shown:

Figure 2.5. Resolve Conflicts Dialog



CVS encountered conflicts when trying to merge your changes in the files below.
Please merge your changes manually.

Folder: C:\mytest\box2

Filename	Format	Status
Copy of addedtoo.txt	Text/ASCII	Needs merge

To resolve the conflicts, double or right click on the files above.

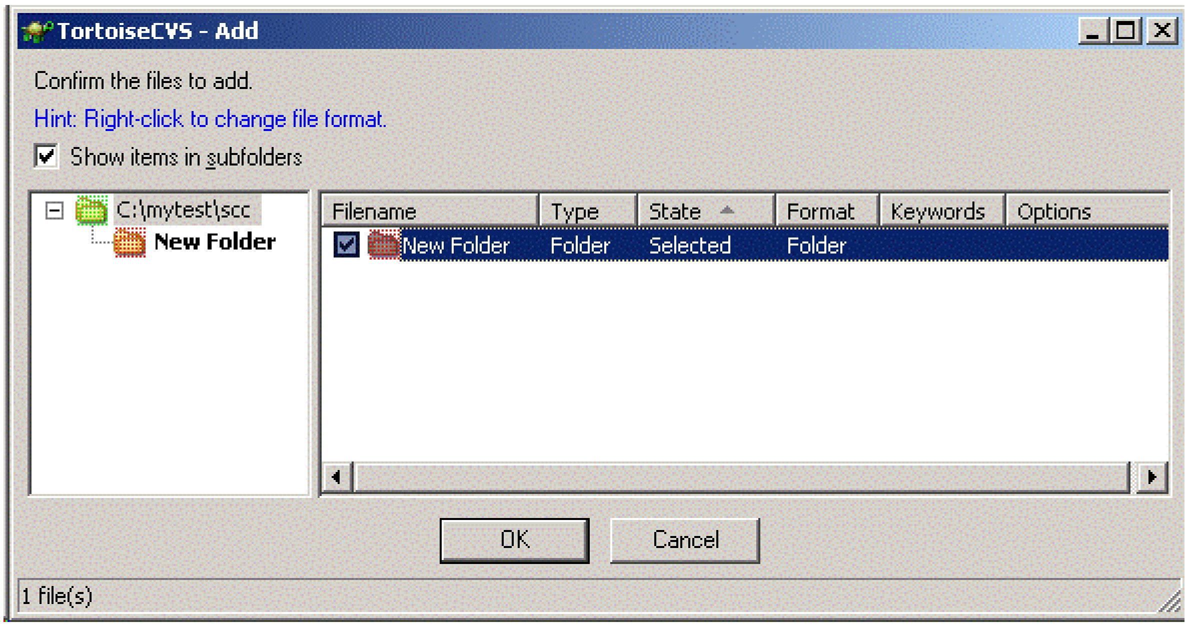
Right-click on the conflicting file(s) and select **Merge Conflicts...**. TortoiseCVS will now invoke the merge application that you have selected in Preferences/Tools/Merge application. Resolve each conflict, save the resulting file, and exit the merge application. TortoiseCVS will ask you to confirm that you want to save the result of the merge.

You can also bring up this dialog after closing the progress dialog by right-clicking in the sandbox and selecting the **CVS** → **Resolve Conflicts...** menu item.

Adding Files and Directories to the Repository

If you have made new files or directories you will notice that they appear with the *Not In CVS* status icon overlay (see Windows Explorer and TortoiseCVS). To put new files or folders under CVS control select the item(s) that you want to add, right-click and choose **CVS Add** from context menu to schedule the addition. The Add Dialog is displayed so you can verify the file you are adding.

Figure 2.6. Add Dialog



TortoiseCVS - Add

Confirm the files to add.

Hint: Right-click to change file format.

☑ Show items in subfolders

C:\mytest\scc
New Folder

Filename	Type	State	Format	Keywords	Options
☑ New Folder	Folder	Selected	Folder		

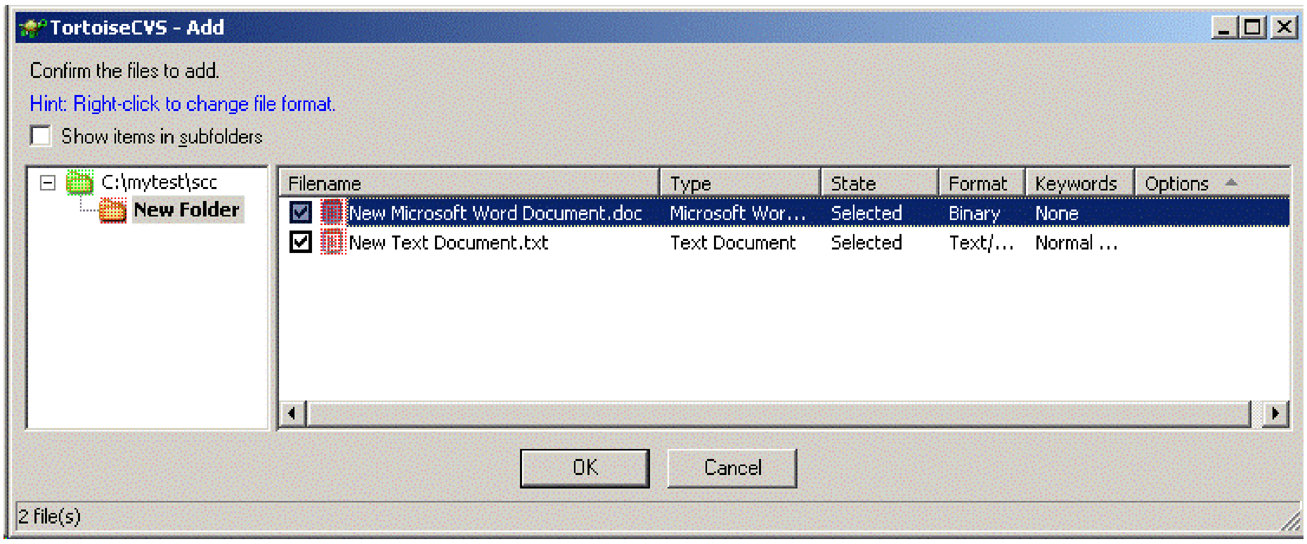
1 file(s)

You do not need to worry about whether a file is ASCII/Text, Unicode/Text, or Binary as TortoiseCVS automatically detects this. For more on how TortoiseCVS handles this see Binary and Unicode Detection.

After an add operation has been performed, the file or files icons appear as "changed". This is because additions are treated as local changes and are not applied to the repository until you *Commit* them.

Additionally, you can add a number of files and folders to CVS using the **CVS Add Contents...** command. This operation recursively descends down the folder structure, and displays all unadded files in the Add Dialog. Here you can check and uncheck the files and folders you wish to add to CVS.

Figure 2.7. Add Contents Dialog



TortoiseCVS - Add

Confirm the files to add.

Hint: Right-click to change file format.

☐ Show items in subfolders

C:\mytest\scc
New Folder

Filename	Type	State	Format	Keywords	Options
☑ New Microsoft Word Document.doc	Microsoft Wor...	Selected	Binary	None	
☑ New Text Document.txt	Text Document	Selected	Text/...	Normal ...	

2 file(s)

For more information see Adding New Files: CVS Add and CVS Add Contents....

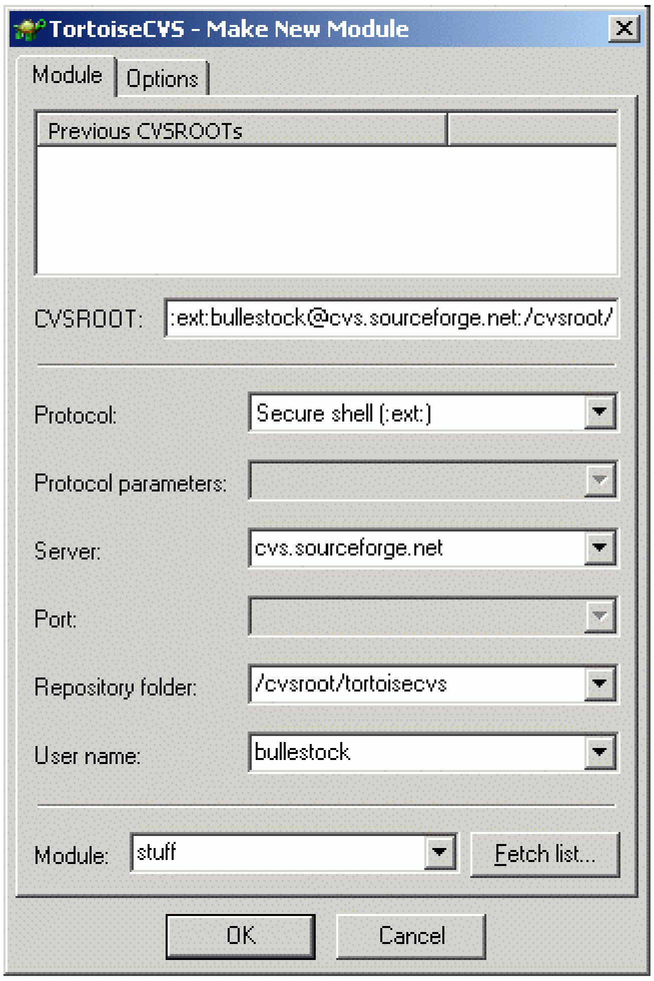
Chapter 3. Advanced Usage of TortoiseCVS

Creating a New Repository or Module

Typically a System Administrator will setup the CVS repository on a remote server. TortoiseCVS, however, can configure repositories and modules either locally or remotely (if you have the correct permissions).

To create a new repository and/or module prepare the folders and files locally as a new sandbox. Right-click on the top level folder for your new module or repository and choose the **CVS** → **Make New Module...** menu item. The Make New Module Dialog will be displayed. Enter the details for the new or existing CVS repository as you would for a *Checkout*. Click **OK** to create the new module.

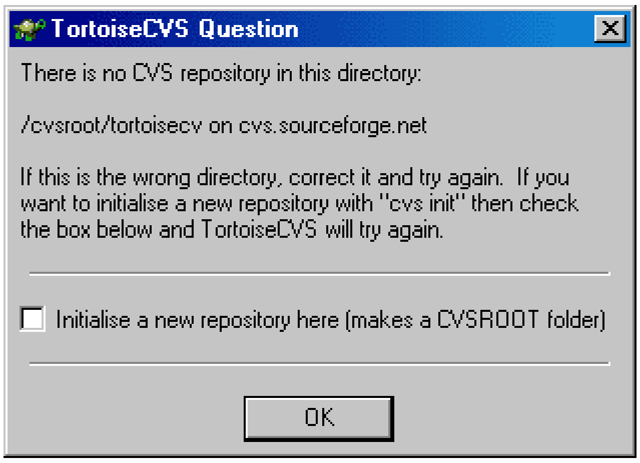
Figure 3.1. Make New Module Dialog



CVSROOT: :ext:bullestock@cvs.sourceforge.net:/cvsroot/
Protocol: Secure shell (:ext:)
Protocol parameters:
Server: cvs.sourceforge.net
Port:
Repository folder: /cvsroot/tortoisecvs
User name: bullestock
Module: stuff

If you are using the `:local:` protocol, and the CVS repository does not exist, you will be presented with the following question:

Figure 3.2. Make New Repository Question



There is no CVS repository in this directory:

/cvsroot/tortoisecv on cvs.sourceforge.net

If this is the wrong directory, correct it and try again. If you want to initialise a new repository with "cvs init" then check the box below and TortoiseCVS will try again.

☐ Initialise a new repository here (makes a CVSROOT folder)

To create the new repository enable the checkbox **Initialise a new repository here** and click **OK**. If you have the right permissions on the remote server TortoiseCVS will setup the new repository.

Once TortoiseCVS has either created a new or connected an existing CVS repository the new module will be created. By default TortoiseCVS uses the name of the selected folder for the new module's name, but you can change this by modifying the **Module** textbox.

Watch, Edit and Unedit

CVS can also follow the semantics of *Lock/Unlock* that most version control systems use, although CVS calls it *Edit/Unedit*. This feature, however, is not enabled in CVS by default so before you can begin you need to enable *Watch*.

You can enable Watch using TortoiseCVS when you *Make a New Module* by enabling the checkbox **Check files read-only** (on the **Options** tab). If you wish to enable watch on an existing module, you will need run the following command (using the Command Prompt) on the top level of the module:

```
cvs watch on
```

Once Watch is enabled, when performing *Checkout* or *Update* the working files will be created as read-only. To change one or more files, select the directory, file or files that you want to modify, right-click on the selection, and choose the **CVS Edit** menu item. This command works slightly different depending on whether the file is text or binary. For binary files (such as `.doc`), TortoiseCVS performs an *Update* on the file and then checks if another user is already editing the file. If this is the case, you will not be allowed to edit it. For text files (such as `.txt` or `.cpp`), you may work on the file even if another user is already working on it; in this case, TortoiseCVS will notify you of that fact.

When you are done working on the file or files, *Commit* them as you would normally. If, however, you decide that you do not want to make changes select the file or files: Right-click on this selection and choose the **CVS → Unedit** menu item. TortoiseCVS will ask you if you want to revert the file to the original version (thus losing your changes).

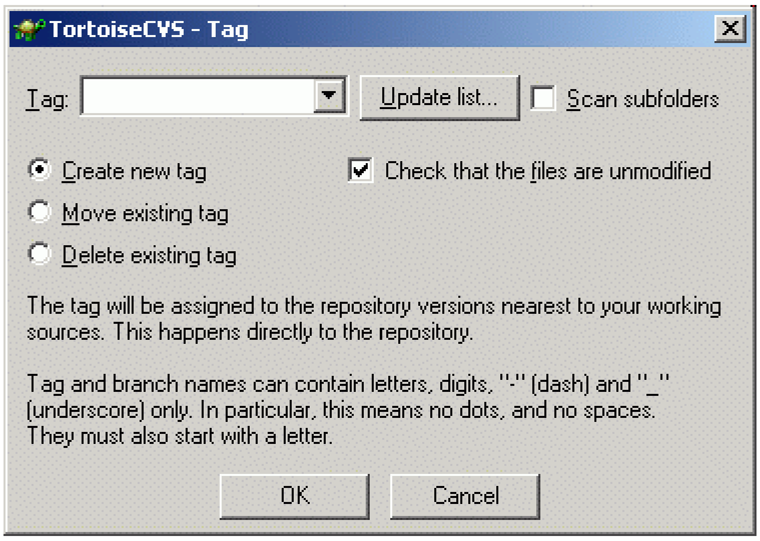
If you would like to see which files are currently being edited and by whom, right-click anywhere in Windows Explorer and choose the **CVS → Show edited files** menu item.

Tagging and Labeling

At a given stage of development, giving one or more files a common label that refers to their revisions, is known as tagging those files. Tagging is typically used on entire modules, so that the current state of the module can be reconstructed in the future. This kind of tagging should always be done on project deliverables, and before starting major changes.

To tag one or more files or directories with a label, select the directory, file or files that you want to tag. Right-click the mouse button on the selection, and choose the **CVS** → **Tag...** menu item.

Figure 3.3. Tag Dialog



[✓ Create new tag] Move existing tag Delete existing tag

[✓ Check that the files are unmodified]

The tag will be assigned to the repository versions nearest to your working sources. This happens directly to the repository.

Tag and branch names can contain letters, digits, "-" (dash) and "_" (underscore) only. In particular, this means no dots, and no spaces. They must also start with a letter.

You will then be presented with the Tag Dialog. Here you can enter a label in the **Tag** field. CVS is quite restrictive when it comes to what characters a tag may contain. A tag must start with a letter, and may contain letters, digits, "-" (dash) and "_" (underscore) only. In particular, this means no dots, and no spaces. If you want to include version numbers in a tag, replace the dots with dashes. Two tag names are reserved, as they have special meaning in CVS: "HEAD" refers to the most recent version available in the repository, while "BASE" is the revision you last checked out into the local directory.

Once the label has been entered, press the **OK** button. Unlike most CVS operations, the tag is immediately applied to the repository, and no commit is required.

For more information see Making a Snapshot: CVS Tag....

Reverting to an Older Version of a File

Developers occasionally need to undo changes that have already been checked in. Going back to a specific file revision in TortoiseCVS is easy:

1. Right-click on the file and select **CVS** → **History** to bring up the History Dialog.
2. Right-click on the revision you need and choose the "Save this revision as..." option.
3. When the Save As dialog appears, do not click the "Save" button; instead, double-click on the file's name in the directory listing.
4. Answer "Yes" when TortoiseCVS prompts you to overwrite the file.

Renaming or Moving a File

Sometimes a file may have been added to CVS with the wrong name or in the wrong location. Recent revisions of CVSNT [http://www.cvsnt.org/wiki] support renaming. If your CVS server does not, you will have to copy the file to the new name and perform a CVS Add and a CVS Remove followed by a CVS Commit.

If your CVS server does support renaming, renaming a file is simple:

1. Right-click on the file and select **CVS** → **Rename** to bring up the Rename Dialog.
2. Enter the name you want the file to have (if you want to move the file, enter the desired relative path to the file).
3. Click **OK**.

Branching And Merging

One of the features of version control systems, is the ability to isolate changes onto a separate line of development. This line is known as a *Branch*.

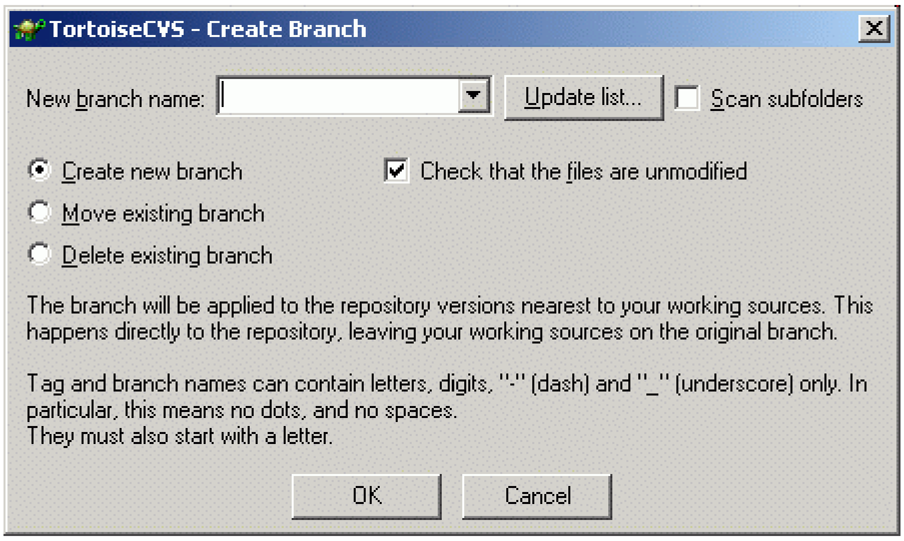
Branching is useful to control changes during the lifecycle of a software project. For example, suppose you have made the first release of your software project version 1.0. You now begin adding new features in preparation for your next major release 2.0. Between your initial release and the new version it is discovered that there are bugs in the software. The current revision of the source code is in a state of flux and is not expected to be stable for at least another month and therefore there is no way to make a bugfix release based on the newest sources.

Instead of attempting to make the fix to the current version, or the *HEAD* branch, you would create a branch on the revision trees for all the files that made up release 1.0. You can then make modifications to the branch without disturbing the head branch. When the modifications are finished you can either incorporate them on the head branch or leave them on the release 1.0 branch.

Creating a Branch

To create a branch, select the directory or files that you want to branch. Right-click the on the selection, and choose **CVS** → **Branch...** from the context-menu.

Figure 3.4. Branch Dialog



TortoiseCVS - Create Branch

New branch name: [ ] Scan subfolders

(●) Create new branch
( ) Move existing branch
( ) Delete existing branch

[✓] Check that the files are unmodified

The branch will be applied to the repository versions nearest to your working sources. This happens directly to the repository, leaving your working sources on the original branch.

Tag and branch names can contain letters, digits, "-" (dash) and "_" (underscore) only. In particular, this means no dots, and no spaces.
They must also start with a letter.

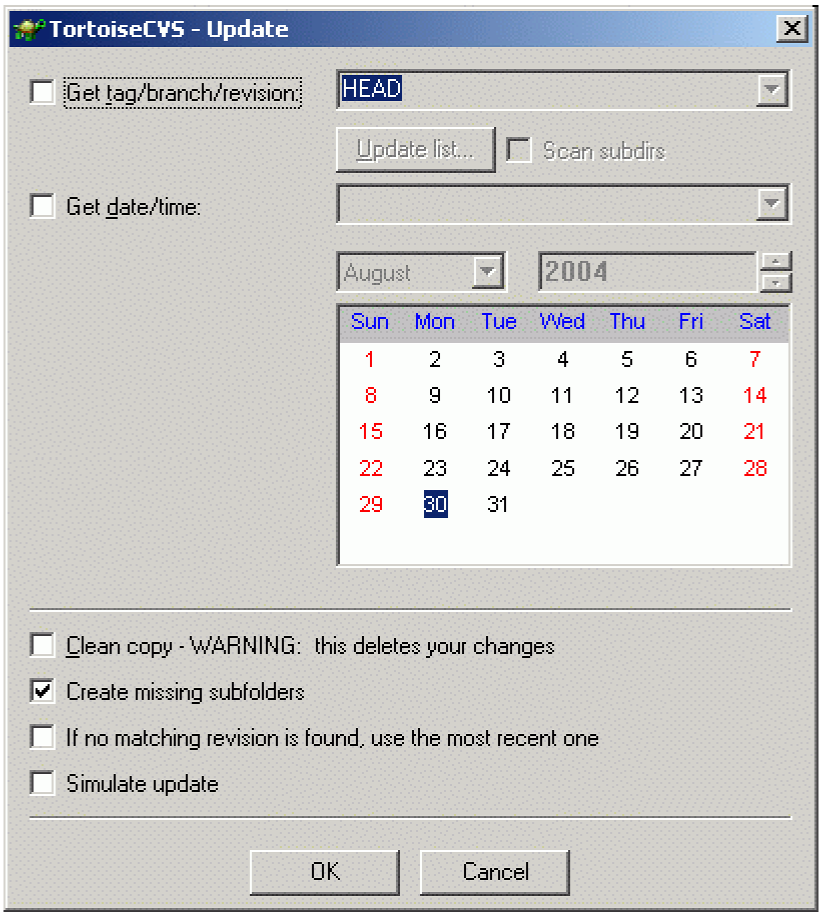
You will then be presented with the Branch Dialog. Here you can enter a label in the **Branch** field. The same name restrictions that apply to *Tags*, apply to branches as well. Once you have entered the desired branch name click the **OK**. Unlike most CVS operations, the branch is immediately applied to the repository, and no commit is required. The branch, however, is only applied on the repository. To start working on the newly created branch select the branch to work on.

Selecting a Branch to Work On

To start working on a branch instead of the default development line, you have to bind your local copy to the branch. This is needed to make sure that actions such as updates, commits etc. work on the branch rather than on the main line of development.

To move your local copy to another branch, select the top level folder of the project. You can also select the exact folders and files that are part of the desired branch if you know this information. Right-click the on the selection, and choose **CVS** → **Update Special...**.

Figure 3.5. Update Special Dialog


TortoiseCVS - Update

☐ Get tag/branch/revision: HEAD

Update list... ☐ Scan subdirs

☐ Get date/time:

August 2004

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

☐ Clean copy - WARNING: this deletes your changes

☑ Create missing subfolders

☐ If no matching revision is found, use the most recent one

☐ Simulate update

OK Cancel

You will then be presented with the Update Special Dialog. Here you can enter the branch name in the **Get Tag/Branch/Revision** field you wish to select. Click the **OK** button, and TortoiseCVS will now do the necessary updates to move your working copy to the desired branch. The updating may also include adding or removing files depending on the state of the branch.

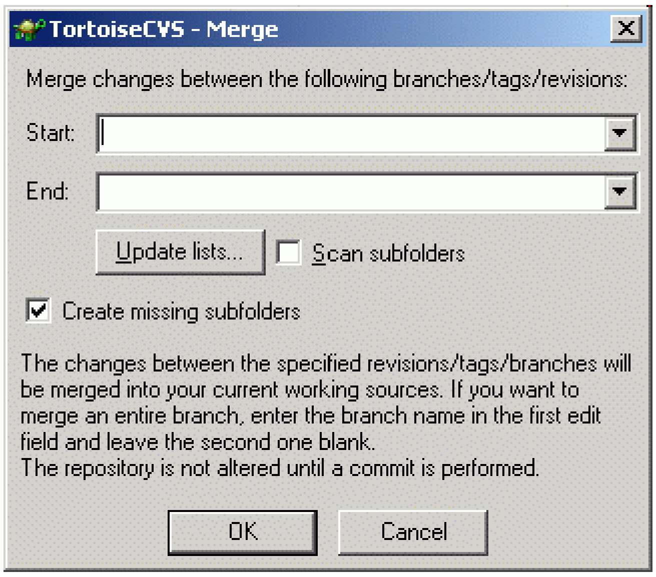
TortoiseCVS puts what is known as sticky tags on the files that are affected by the branch. To remove the sticky tags you must go back to the head branch.

Merging from a Branch

When you are satisfied with the changes you have done on a branch, you may want those changes to be available on the head branch of development. Incorporating changes from one branch to another, is known as merging.

To merge from a branch, move your local copy to the branch you want to merge the changes into. See Selecting a Branch to Work On or Going Back to the Head Branch. Select the top level folder of the project. You can also select the exact folders and files that are part of the desired branch if you know this information. Right-click the on the selection, and choose **CVS** → **Merge...**.

Figure 3.6. Merge Dialog


TortoiseCVS - Merge

Merge changes between the following branches/tags/revisions:

Start:

End:

Update lists...

[ ] Scan subfolders

[✓] Create missing subfolders

The changes between the specified revisions/tags/branches will be merged into your current working sources. If you want to merge an entire branch, enter the branch name in the first edit field and leave the second one blank.
The repository is not altered until a commit is performed.

OK Cancel

You will then be presented with the Merge Dialog. Here you can enter the branch name in the **Branch to merge from** field you wish to merge. Click the **OK** button, and TortoiseCVS will now merged the branch specified into your local copy. Your changes will not be made on the server repository, until you commit your changes.

The merge given above will try to merge changes from the start of the branch. If you do the operation a second time (to merge changes done to the branch after the last merge), merging from the start of the branch is not what you want, and it will most likely get you into trouble. To get around this problem, you should give the branch a new tag after every merge, and use the new tag when naming the branch for subsequent merges.

Note: The above paragraph only applies for standard Unix CVS (aka cvshome.org CVS). If your server runs CVSNT, you can take advantage of a special feature known as *merge points*. This means that CVS keeps track of your last merge, so that you can effortlessly merge from the same branch repeatedly. For more information on this feature, see the CVSNT Wiki page [http://www.cvsnt.org/wiki/MergePoint] and the CVSNT Manual [58].

Going Back to the Head Branch

If you want to stop working on a branch and move your local copy back to the main line of development, you have to make TortoiseCVS remove all sticky tags.

To remove the sticky tags, and thus update your local copy to the main development line, select the top level folder of the project. You can also select the exact folders and files that are part of the desired branch if you know this information. Right-click the on the selection, and choose **CVS** → **Update Special...**.

You will then be presented with the Update Special Dialog. Select the **Return to main (HEAD) branch** check-box and click the **OK** button. TortoiseCVS will now do the necessary updates to move your working copy back to the head branch.

Binary and Unicode Detection

TortoiseCVS tries to automatically detect the type of file you are adding to CVS. It can detect whether the file is *Text/ASCII*, *Text/Unicode*, or *Binary*. TortoiseCVS first checks the file extension to determine whether the file is Binary or Text. For example `.doc` and `.exe`, are always assumed to be Binary while other extensions, such as `.cpp` and `.txt`, are always assumed to be text. TortoiseCVS does this using a built-in list of Binary and Text extensions which includes most the common file extensions. To customize this list, you can copy the file `TortoiseCVS.FileTypes` from the TortoiseCVS installation directory to your HOME directory and edit it for your needs.

By default TortoiseCVS will also examine the first 4000 bytes of a file to determine the type. This is used in cases where the file extension is not known. These first few bytes are also used to determine whether the file is Text/ASCII or Text/Unicode.

TortoiseCVS also provides a sophisticated plugin mechanisim that you can implement a DLL to perform the file type detection. The details of this plugin interface is also available in the `FileTypes.config`.

In all cases, the Add Dialog presents the files you are about to add their file types, so that you have a chance to correct the choices made.

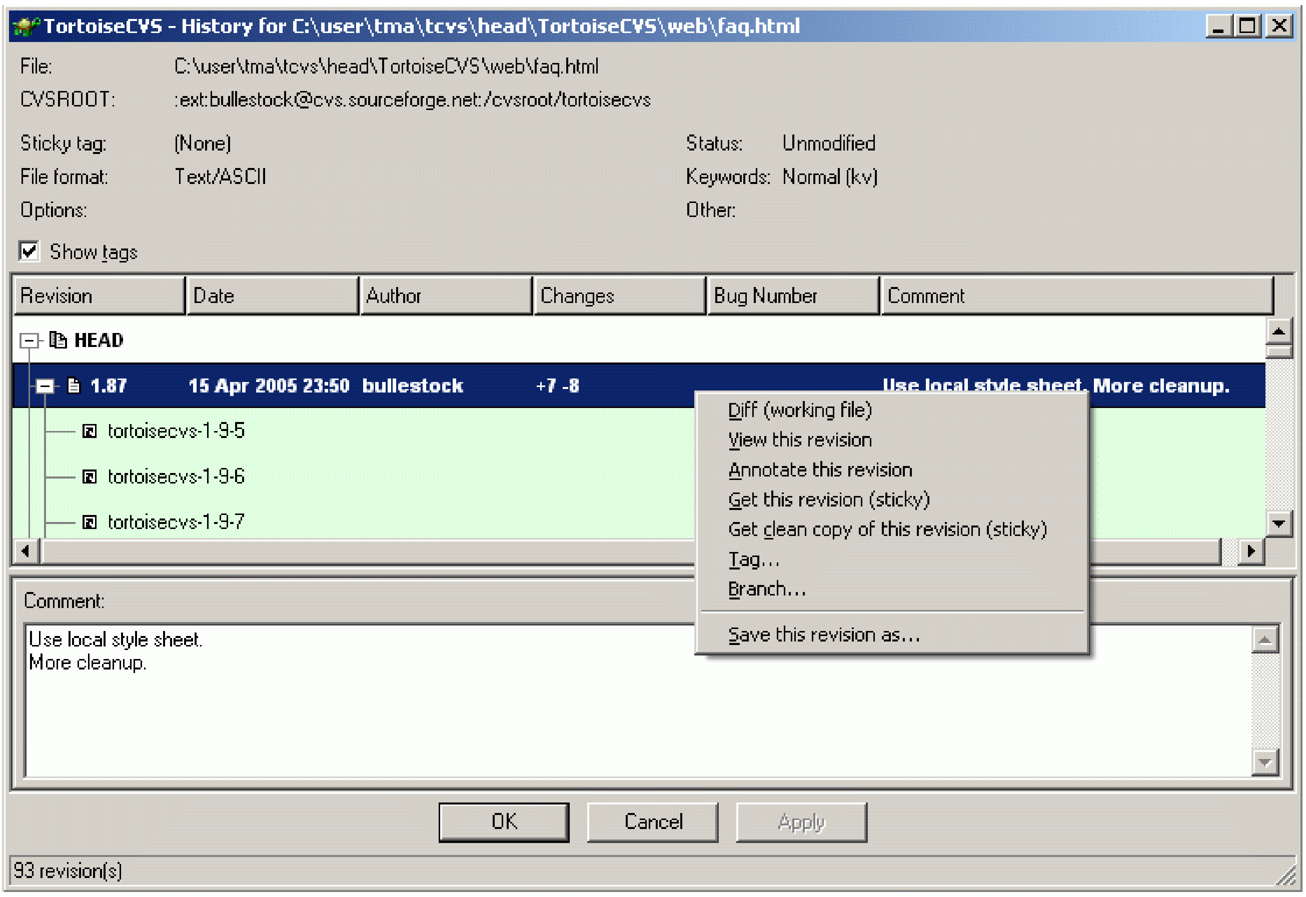
File Revision History

TortoiseCVS provides two methods of reviewing and working with the history of a file. The first is via the History Dialog.

History Dialog

You can view a file's history by selecting it and right-clicking **CVS** → **History**.

Figure 3.7. History Dialog


TortoiseCVS - History for C:\user\tma\tcvs\head\TortoiseCVS\web\faq.html

File:	C:\user\tma\tcvs\head\TortoiseCVS\web\faq.html		
CVSROOT:	:ext:bullestock@cvs.sourceforge.net:/cvsroot/tortoisecvs		
Sticky tag:	(None)	Status:	Unmodified
File format:	Text/ASCII	Keywords:	Normal (kv)
Options:		Other:	

☑ Show tags

Revision	Date	Author	Changes	Bug Number	Comment
HEAD					
1.87	15 Apr 2005 23:50	bullestock	+7 -8		Use local style sheet. More cleanup.
tortoisecvs-1-9-5					
tortoisecvs-1-9-6					
tortoisecvs-1-9-7					

Comment:

Use local style sheet.
More cleanup.

The History Dialog will show a log of the revisions, tags, and branches for the selected file. It will also offer you the opportunity, by right-clicking on a specific revision, tag, or branch, to *diff*, *save* the revision locally, or *update* with sticky tags (an advanced CVS concept; usually, the *save* option is preferable to a sticky *update*). Unless the file is an executable, you can also select *view* to view that revision using the default application for the file type in question. You can also edit any current or previous commit message by clicking on a revision and typing into the text area at the bottom of the window.

When you click on a revision, detailed information on that revision is shown at the top of the dialog. This includes the file format (text, binary, etc.) and the keyword expansion mode. With older CVS servers, this information will be the same for every revision of a file, but newer servers allow it to be set specifically for each revision.

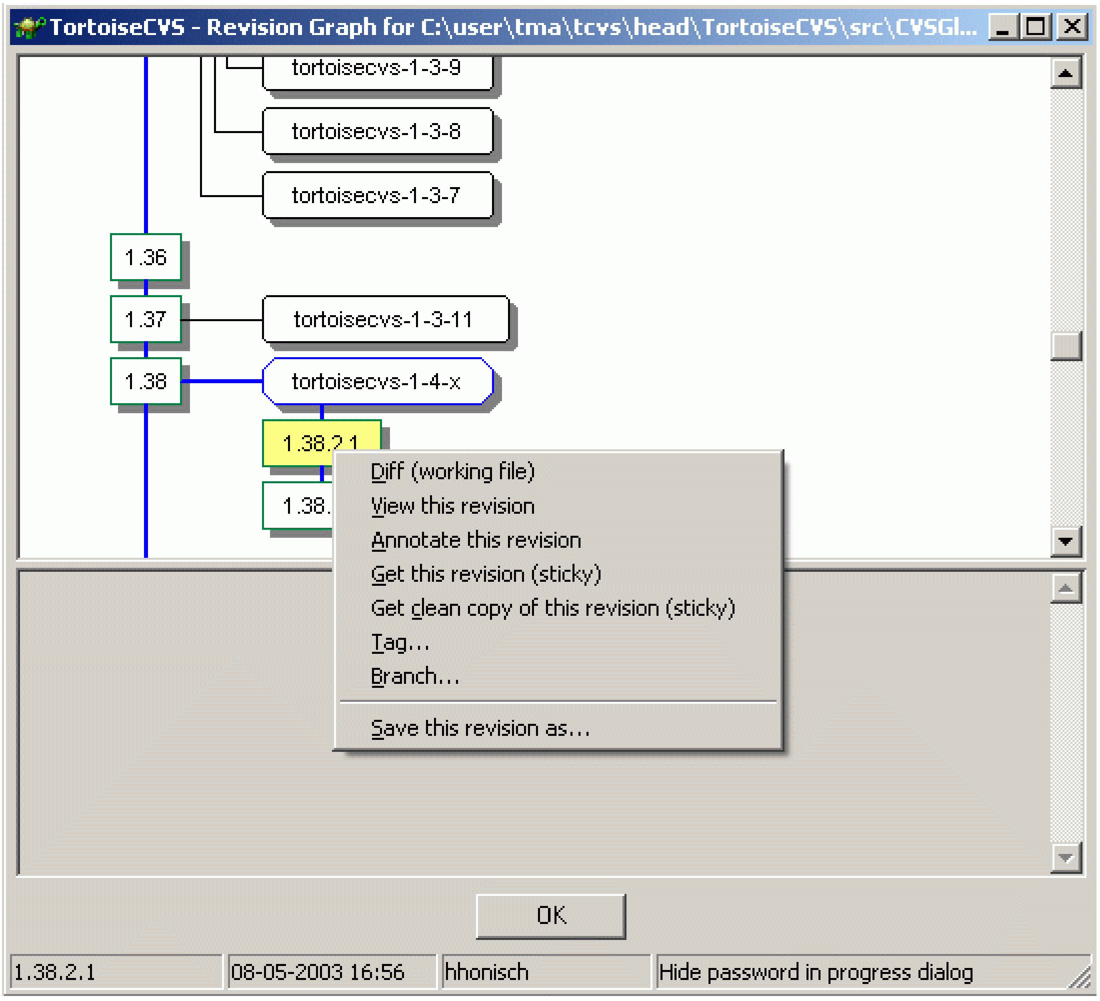
The current *filename* is also shown; if the file has been renamed at some point, you can see what its name has been for each revision.

To simplify the view, you can deselect the **Show Tags** checkbox. When you do this, only branches are shown in the dialog.

Revision Graph Dialog

TortoiseCVS also provides a Revision Graph Dialog to graphically show you the history of a file. You can view a file's revision graph by selecting it and right-clicking **CVS** → **Revision Graph**.

Figure 3.8. Revision Graph Dialog



Like the History Dialog, the Revision Graph Dialog will allow you to see the details for and perform CVS operations on any revision, tag, or branch.

Revisions where a file has been removed are indicated by a red outline.

If your CVS server runs CVSNT, you will also be able to see which merge operations have been performed on the file: A purple arrow is drawn from the source revision to the destination revision (see Mergepoints [16]).

Web Log

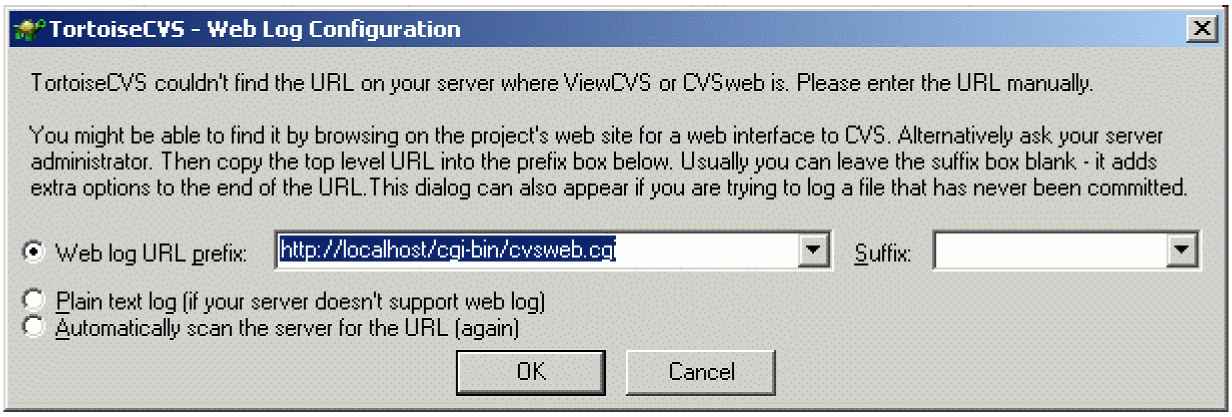
Web Log offers much of the same functionality as the History and Revision Graph dialogs. One important difference is that Web Log is not part of TortoiseCVS itself, but depends on special software that may be installed on the CVS server you use. Two popular packages for providing Web Log functionality are ViewCVS and CVSweb — ask your CVS administrator if one of these is available.

To view the Web Log, you open a special URL using a web browser. TortoiseCVS knows the most common URLs in use for this and can in most cases detect the correct one automatically[1]. If your configuration is nonstandard, you may have to enter the URL manually the first time - next time TortoiseCVS will remember it.

You can view Web Log for a file or folder by selecting it and right-clicking **CVS** → **Web Log**. If TortoiseCVS fails to detect the URL, the dialog below will appear.

[1] For details about the automatic URL detection, see How Web Log Autodetects the Server URL.

Figure 3.9. Web Log Configuration Dialog


TortoiseCVS - Web Log Configuration

TortoiseCVS couldn't find the URL on your server where ViewCVS or CVSweb is. Please enter the URL manually.

You might be able to find it by browsing on the project's web site for a web interface to CVS. Alternatively ask your server administrator. Then copy the top level URL into the prefix box below. Usually you can leave the suffix box blank - it adds extra options to the end of the URL.This dialog can also appear if you are trying to log a file that has never been committed.

[✓ Web log URL prefix:] http://localhost/cgi-bin/cvsweb.cgi   Suffix:

Plain text log (if your server doesn't support web log)

Automatically scan the server for the URL (again)

Enter the correct URL and click OK. If the URL of the Web Log server has changed, you may have to tell TortoiseCVS to perform the automatic scan again.

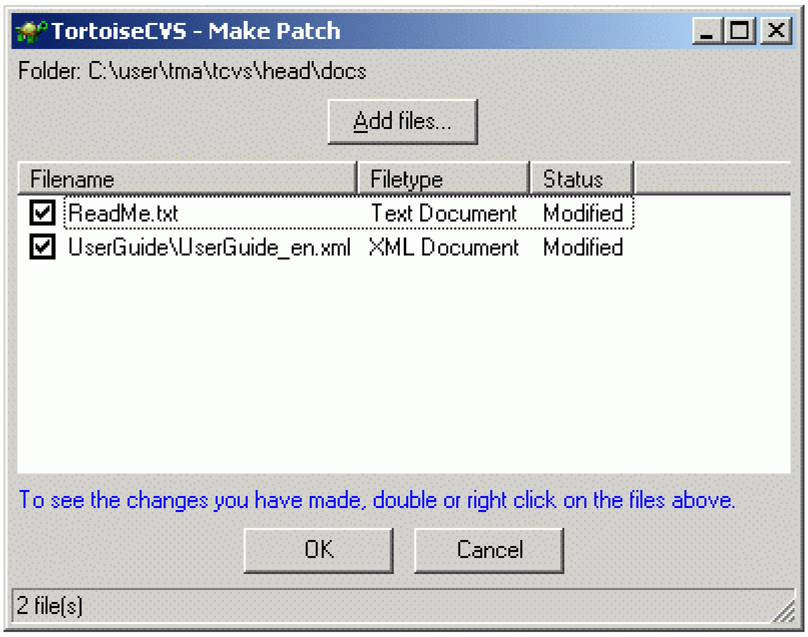
You can also force this dialog to appear by holding down the **Ctrl** key while clicking on **Web Log**. This is useful if your Web Log server has been moved to another machine.

Making a Patch File

Often you might have read access to a CVS repository or module, but not write access. Open source projects typically work this way to maintain quality control for all contributions to a project. The easiest and most reliable way to contribute changes to these projects is by creating a *Patch File* to submit to them.

In TortoiseCVS this is done by selecting the folders with changes that you wish to include in the patch file. Right-click on your selections and choose **CVS** → **Make Patch...** from the context menu. TortoiseCVS determines which files have been modified and lists them in the Make Patch dialog:

Figure 3.10. Make Patch Dialog


TortoiseCVS - Make Patch

Folder: C:\user\tma\tcvs\head\docs

	Filename	Filetype	Status
✓	ReadMe.txt	Text Document	Modified
✓	UserGuide\UserGuide_en.xml	XML Document	Modified

To see the changes you have made, double or right click on the files above.

2 file(s)

If you want to exclude some files from your patch, simply deselect them in the dialog.

Sometimes your contribution involves adding new files which are not yet in CVS. TortoiseCVS cannot guess that you would like these to be included in the patch, but in the Make Patch dialog you can click the **Add files...** button to bring up the familiar Add dialog. Here you can select files to be included in the patch.

When you are satisfied with your selection, click **OK**. You will then be prompted with a Save dialog to indicate where to save your patch file. Finally, TortoiseCVS will open the new patch file in Notepad, where you can validate your changes.

Chapter 4. Customizing TortoiseCVS

Overlay Icons

Selecting a Different Set of Overlay Icons

TortoiseCVS comes with several sets of overlay icons. You might prefer one of the alternative icon sets to the default one; open the Preferences Dialog - Iconset Chooser Iconset Chooser dialog to select another icon set. Regrettably, Explorer caches the overlay icons, which means that you have to restart Windows to see the effect of the change. The dialog shows a preview of the icons to help you choose the correct one.

Changing how the Overlay Icons Work

By default, the overlay icon for a folder will indicate "modified" if any of the files in the folder are modified. It is possible to configure TortoiseCVS to indicate "modified" if any of the files in the folder *or in any subfolders* are modified.

As updating this status can easily be resource-intensive, TortoiseCVS offers the possibility of either updating the status each time you open the folder, or only on demand. In the latter case, the icons will show the cached status, and you will have to invoke **CVS** → **Refresh Folder Status** to update the status.

Go to Preferences Dialog - Advanced Tab and change the **Folder icon overlays** setting as you desire.

Chapter 5. Command Reference for TortoiseCVS

This chapter is written to illustrate most main features and functionality of TortoiseCVS. It assumes that a) you already have an understanding of CVS and b) you have access to an already installed a CVS server with created a CVS repository. If you are new to CVS then please refer to the section "TortoiseCVS for Beginners."

Installing TortoiseCVS

1. Download the latest version of TortoiseCVS from the web site (http://tortoisecvs.org/) and save to a temporary location on your hard drive.
2. Run the installer. If you are running Windows NT or 2000 you must have local admin privileges to install TortoiseCVS correctly.
3. Choose the location for TortoiseCVS to be installed. The default is `C:\Program Files\TortoiseCVS`.
4. Choose the installation type, either **Full**, **Compact**, or **Custom installation**. If you desire internationalisation support, you will need either the **Full** or **Custom installation** options.
5. Once the installer has completed you must reboot the computer for TortoiseCVS to finish its installation[1].

Obtaining a Working Copy: CVS Checkout...

Creates a local sandbox from a CVS repository.

1. Create a folder for the sandbox in Windows Explorer.
2. Right-click on the folder.
3. Select **CVS Checkout...** from the context-menu.
4. The Checkout Dialog will appear. From here you can either enter the repository's CVSROOT into the **CVSROOT** field, or construct a CVSROOT using the **Protocol**, **Server**, **Port**, **Repository name**, and **User name** fields.
5. If you know the module name you can enter it in the **Module** field. Conversely, you can checkout an entire repository by entering a period (**.**) into the **Module** field. If you do not know the module name you can click on **Fetch list** to populate the drop-down with available modules. Note: Some CVS servers do not support this feature; if you cannot populate this list you should contact the CVS server administrator.

   If you are checking out a single file you can also specify a bug number that this work relates to.

[1] Strictly speaking, it is only necessary to restart Windows Explorer. If you are feeling adventurous, you can try killing `explorer.exe` from Task Manager. In some cases Windows™ will automatically start a new Explorer instance, in other cases you will have to do it yourself.

6. Click **OK** to being checking out the sandbox.

7. Depending on the protocol and if this is the first time checking out from this CVS repository, you will be prompted to enter a password. Enter the password in the dialog and click **OK**.

8. The Progress Dialog will appear, which provides a detail status of what files are being checked out by TortoiseCVS. By default this dialog will close after the operation is complete, unless it encounters a problem.

Getting Other People's Changes: CVS Update

Synchronizes a local sandbox with a CVS repository. See also Update Special.

1. To update a folder either right-click on it or within it on the Explorer view (but not on any specific files) or to update one or more files select the files to update and right-click on them.

2. Select **CVS Update** from the context-menu.

3. The Progress Dialog will appear to provide a detail status of the files that are being updated by TortoiseCVS. By default this dialog will close after the operation is complete, unless it encounters a problem.

CVS Update Special...

Synchronizes a local sandbox with a CVS repository. See also Update.

1. To update a folder either right-click on it or within it on the Explorer view (but not on any specific files) or to update one or more files select the files to update and right-click on them.

2. Select **CVS** → **CVS Update Special...** from the context-menu.

3. The Update Special Dialog will appear. Enter into the textbox labeled **Get tag/revision/branch:** either the Revision number (i.e. `1.3`), Tag (i.e. `release_1_1_0`), or Branch (i.e. `vendorx_branch`) to change your sandbox copy to.

4. The Progress Dialog will appear to provide a detail status of the files that are being updated by TortoiseCVS. By default this dialog will close after the operation is complete, unless it encounters a problem.

Making Your Changes Available to Others: CVS Commit...

Update the CVS repository with your local changes.

1. To commit changes either right-click on the desired folder and/or files.

2. Select **CVS Commit...** from the context-menu.

3. The Commit Dialog will appear, which will allow you to enter a comment for the commit action. Additionally, you can choose which changes to commit by checking the individual files on the dialog list.

4. Click **OK** (or press **Ctrl-Enter**) once you have entered the appropriate comments and selected the changes to commit.

5. The Progress Dialog will appear to provide a detail status of the committing process performed by TortoiseCVS. By default this dialog will close after the operation is complete, unless it encounters a problem.

Adding New Files: CVS Add and CVS Add Contents...

Add local files and/or folders to the CVS repository.

1. To add files or folders to the CVS select and right-click the files or folders you wish to add. These files must have the CVS Status of Not In CVS.

2. If you have selected a folder you will be provided the option to **Add Recursively** in the context-menu. This option will add all files in the selected folder and its sub-folders. Otherwise, to only add those files and folders selected select **Add** from the context-menu.

3. The Progress Dialog will show a detail status of the adding process performed by TortoiseCVS.

   TortoiseCVS adds the files and shows a Tortoise Tip that reminds you that you must commit the added files to finish the process.

Discarding Obsolete Files: CVS Remove

Remove a file from the current branch. The file will still be available when retrieving an older revision or tag.

1. To remove one or more files from the CVS repository select the desired files and/or folders to remove and right-click on them.

2. Select **CVS** → **Remove** from the context-menu. TortoiseCVS performs the removal and shows a Tortoise Tip that reminds you that you must commit the removal to finish the process.

3. Finally, invoke the **CVS Commit** command on the folder (or a parent folder).

4. When another user now performs an Update, CVS will remove the file(s) from the sandbox.

Renaming or Moving Files: CVS Rename

Rename a file, and/or move it to a new location. The file will still be available under the old name and location when retrieving an older revision or tag.

1. To rename one or more files select the desired files and/or folders to renamee and right-click on them.

2. Select **CVS** → **Rename** from the context-menu.

3. Enter the new name and/or location (as a relative path) of the files (if you have selected more than one item, you must enter a path).

4. Click **OK**. TortoiseCVS performs the rename and shows a Tortoise Tip that reminds you that you must commit the rename to finish the process.

5. Finally, invoke the **CVS Commit** command on the folder (or a parent folder).

6. When another user now performs an Update, CVS will rename and/or move the file(s) in the sandbox.

Finding Out What Has Changed: CVS Diff...

Compare a local file to a version in the CVS repository.

1. To compare a file to a different version from the CVS repository, select the desired file and right-click on it.

2. Select **CVS** → **Diff** from the context-menu.

3. If this is the first time you have used TortoiseCVS's Diff feature, you will be asked to select an external diff application. WinMerge [http://winmerge.sf.net] and ExamDiff [http://www.prestosoft.com/ps.asp?page=edp_examdiff] are both available free of charge and are well-suited for text-based source code. If you are developing in the LabVIEW programming language, you can use LVDiff [http://meta-diff.sf.net/lvdiff.html] to compare your block diagrams visually. CSDiff [http://www.componentsoftware.com/products/csdiff/download.htm] is designed to compare different versions of Word documents. To use a combination of diff tools based on the file type you are comparing, try meta-diff [http://meta-diff.sf.net].

4. TortoiseCVS will launch the diff tool you have chosen and compare your local copy of the selected file with the version from the repository. For more sophisticated diff options (such as comparing to an older repository copy or comparing two older versions to each other), see File Revision History.

Making a Snapshot: CVS Tag...

Make a snapshot of a project at a given point in time for later retrieval.

1. To create a tag, select the desired files and/or folders to tag and right-click on them. Note: A tag is normally applied to a folder, usually the top folder of the sandbox.

2. Select **CVS** → **Tag** from the context-menu.

3. Enter the desired tag name. Note that CVS places some constraints on the tag name; these constraints are shown at the bottom of the dialog. To see which tags are already used in this folder, click the **Update list...** button.

4. By default, the **Check that the files are unmodified** checkbox is enabled. This prevents you from applying the tag if you have uncommitted changes. Under special circumstances, you might want to disable this.

5. If the tag already exists, CVS will not move it unless you select **Move existing tag**.

6. To remove a tag which you applied by mistake, or no longer want, select **Delete existing tag**.

7. Finally, click **OK**.

Lines of Development: CVS Branch...

Create a branch (aka fork) for a separate line of development.

1. To create a branch, select the desired files and/or folders you want to branch and right-click on them. Note: A branch is normally created on an entire folder, usually the top folder of the sandbox.
2. Select **CVS** → **Branch** from the context-menu.
3. Enter the desired branch name. Note that CVS places some constraints on the branch name; these constraints are shown at the bottom of the dialog. To see which branches are already used in this folder, click the **Update list...** button.
4. By default, the **Check that the files are unmodified** checkbox is enabled. This prevents you from creating the branch if you have uncommitted changes. Under special circumstances, you might want to disable this.
5. If the branch already exists, CVS will not move it unless you select **Move existing branch**.
6. To remove a branch which you created by mistake, or no longer want, select **Delete existing branch**. You should be very careful when doing this.
7. Finally, click **OK**.
8. Note that your sandbox is not automatically changed so that you are working on the newly created branch. To do this, see Selecting a Branch to Work On.

CVS Merge...

Update one branch with changes from another branch.

See the section called “Merging from a Branch”.

CVS Make New Module

Create a new module (project) in the CVS repository.

See the section called “Creating a New Repository or Module”.

Watching And Locking

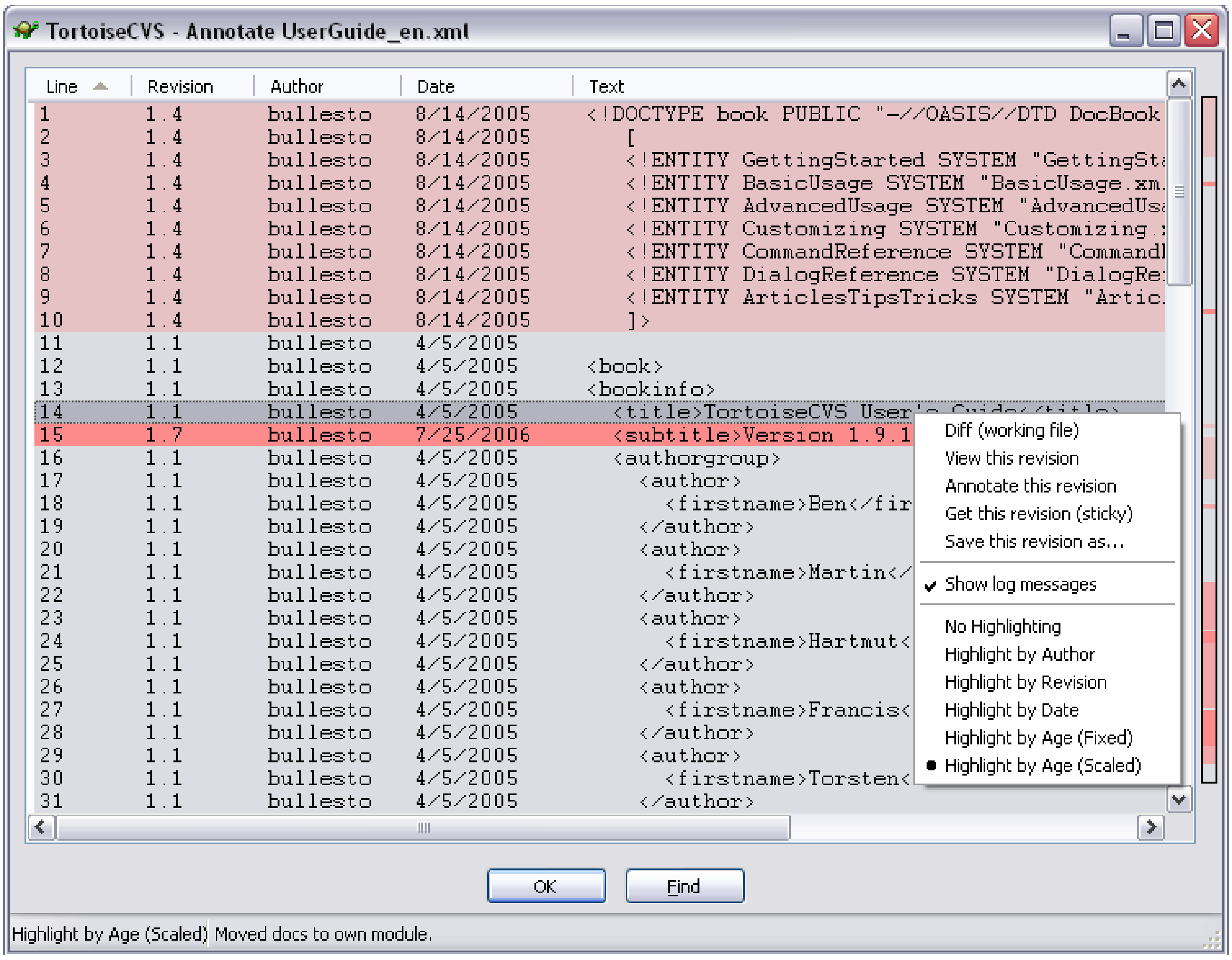
Track who is editing which files.

See the section called “Watch, Edit and Unedit”.

Finding Out Who to Blame: CVS Annotate

When something suddenly stops working, it can be valuable to find out exactly who is to blame for the change. This is what **CVS Annotate** is for.

Figure 5.1. Annotate Dialog



Line	Revision	Author	Date	Text
1	1.4	bullesto	8/14/2005	<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook
2	1.4	bullesto	8/14/2005	[
3	1.4	bullesto	8/14/2005	<!ENTITY GettingStarted SYSTEM "GettingSta
4	1.4	bullesto	8/14/2005	<!ENTITY BasicUsage SYSTEM "BasicUsage.xm.
5	1.4	bullesto	8/14/2005	<!ENTITY AdvancedUsage SYSTEM "AdvancedUs
6	1.4	bullesto	8/14/2005	<!ENTITY Customizing SYSTEM "Customizing.:
7	1.4	bullesto	8/14/2005	<!ENTITY CommandReference SYSTEM "Command]
8	1.4	bullesto	8/14/2005	<!ENTITY DialogReference SYSTEM "DialogRe:
9	1.4	bullesto	8/14/2005	<!ENTITY ArticlesTipsTricks SYSTEM "Artic.
10	1.4	bullesto	8/14/2005	]>
11	1.1	bullesto	4/5/2005	
12	1.1	bullesto	4/5/2005	<book>
13	1.1	bullesto	4/5/2005	<bookinfo>
14	1.1	bullesto	4/5/2005	<title>TortoiseCVS User'
15	1.7	bullesto	7/25/2006	<subtitle>Version 1.9.1
16	1.1	bullesto	4/5/2005	<authorgroup>
17	1.1	bullesto	4/5/2005	<author>
18	1.1	bullesto	4/5/2005	<firstname>Ben</fir
19	1.1	bullesto	4/5/2005	</author>
20	1.1	bullesto	4/5/2005	<author>
21	1.1	bullesto	4/5/2005	<firstname>Martin</
22	1.1	bullesto	4/5/2005	</author>
23	1.1	bullesto	4/5/2005	<author>
24	1.1	bullesto	4/5/2005	<firstname>Hartmut<
25	1.1	bullesto	4/5/2005	</author>
26	1.1	bullesto	4/5/2005	<author>
27	1.1	bullesto	4/5/2005	<firstname>Francis<
28	1.1	bullesto	4/5/2005	</author>
29	1.1	bullesto	4/5/2005	<author>
30	1.1	bullesto	4/5/2005	<firstname>Torsten<
31	1.1	bullesto	4/5/2005	</author>

```
Highlight by Age (Scaled) Moved docs to own module.
```

For each line of the file, the dialog shows

1. From which revision that line originates (e.g. lines marked as `1.1` have not been changed since the first revision of the file).
2. The author of the line, i.e. the person who made the last change. (TortoiseCVS tries to display the first 30 characters of the author name, but many CVS servers are only capable of displaying the first 8 characters, as can also be seen in the screenshot above).
3. The time and date when the line was last changed.
4. The current content of the line.

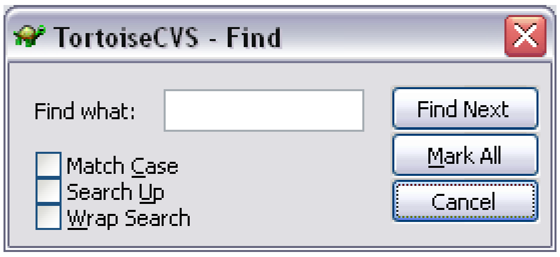
If you right-click a line in the dialog, you have the option of launching various commands, including viewing an annotation for a particular revision.

The **Show log messages** menu item toggles display of log messages; when this is enabled, the log message for the selected line is shown in the status bar at the bottom of the dialog.

Finally, you can change the appearance of the dialog by choosing between various highlighting options. Note that when highlighting is active, the bar to the right of the scroll bar gives an overview of the entire file.

The **Find** button allows you to search for a piece of text using the dialog shown in Find Dialog.

Figure 5.2. Find Dialog



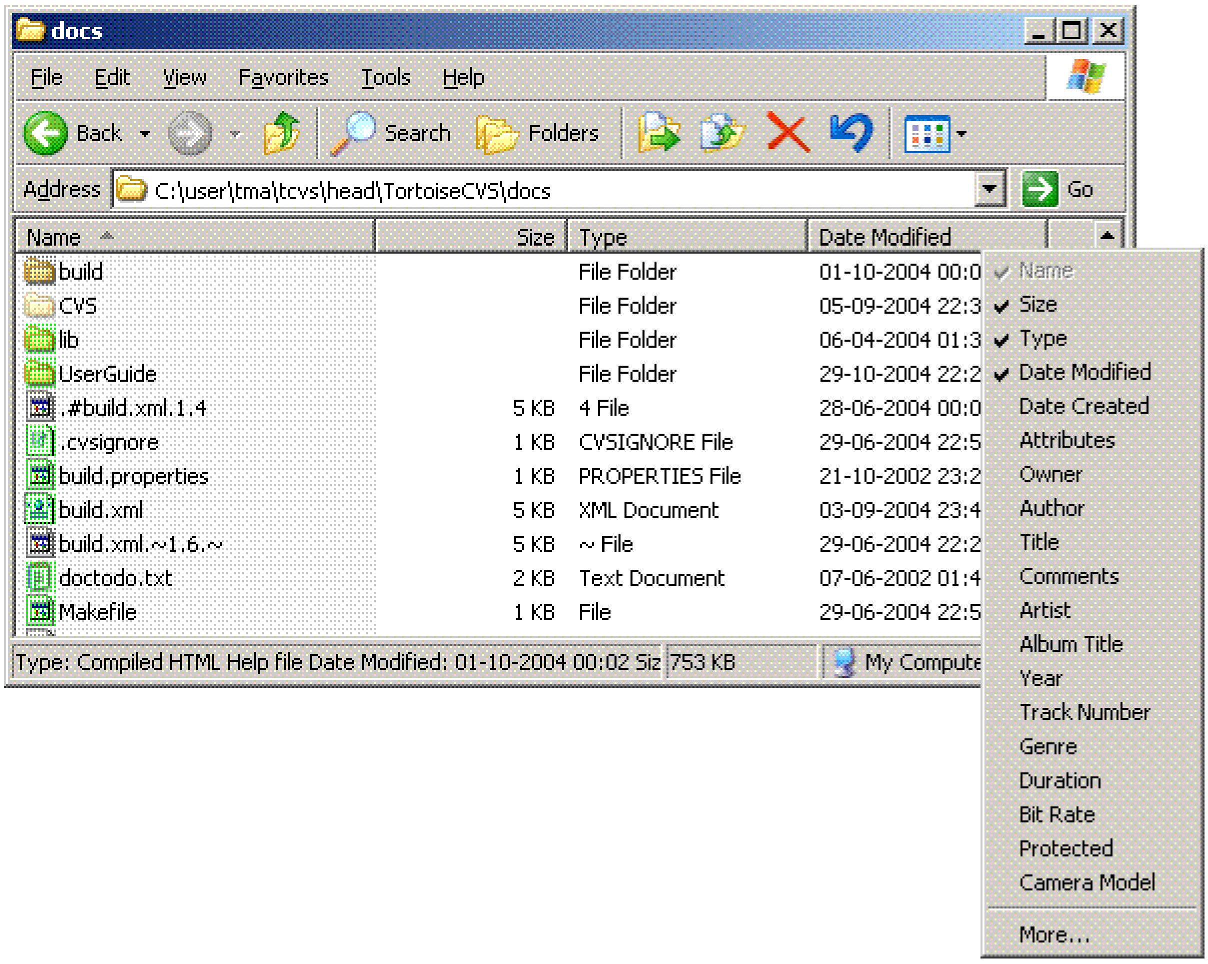
Showing More Information: CVS Explorer Columns

In Details view, TortoiseCVS can show CVS-specific columns in the Explorer view (only on Windows 2000 and later). The available columns are

- CVS Revision
- CVS Sticky Tag
- CVS Status
- CVS File Format

To select which columns to display, right-click on the header as shown in the figure below. Select **More...**, and you will see a dialog listing all the available columns on your system. Scroll down to find the "CVS" columns and select the ones you want.

Figure 5.3. Choosing which columns to display



Name	Size	Type	Date Modified
build		File Folder	01-10-2004 00:0
CVS		File Folder	05-09-2004 22:3
lib		File Folder	06-04-2004 01:3
UserGuide		File Folder	29-10-2004 22:2
.#build.xml.1.4	5 KB	4 File	28-06-2004 00:0
.cvsignore	1 KB	CVSIGNORE File	29-06-2004 22:5
build.properties	1 KB	PROPERTIES File	21-10-2002 23:2
build.xml	5 KB	XML Document	03-09-2004 23:4
build.xml.~1.6.~	5 KB	~ File	29-06-2004 22:2
doctodo.txt	2 KB	Text Document	07-06-2002 01:4
Makefile	1 KB	File	29-06-2004 22:5

Type: Compiled HTML Help file Date Modified: 01-10-2004 00:02 Siz 753 KB

Keyboard Shortcuts

TODO: This section will explain how to use TortoiseCVS without the mouse.

How Web Log Autodetects the Server URL

As mentioned in Web Log, TortoiseCVS can in many cases automatically detect the URL used by your ViewCVS or CVSweb server. It does this by using the following algorithm:

1. Start with the name of the CVS server, with a `http://` prefix.

2. Append one item from the list

   - `/cgi-bin`
   - `/viewcvs`
   - `/cvsweb`
   - `/bin`
   - (the empty string)

   and an item from the list

- `/viewcvs.cgi`
- `/viewcvs`
- `/cvsweb.cgi`
- `/cvsweb.pl`
- `/cvsweb`

3. Append the relative path for the file or folder (i.e. the value of 'Repository' shown in the CVS Properties dialog).
4. Check if the computed URL works. If not, add `?cvsroot=` followed by the name of the module and try again.

The above steps are done for each possible permutation of the different prefixes and suffixes. If a URL is found which returns HTTP status 200 (OK), the search stops. If no such URL is found, and an URL is found which returns HTTP status 401 (Denied) or 403 (Forbidden), that URL is used (based on the assumption that the user will be allowed access after authenticating to the web server).

Actually, there is also a little bit of magic done if the server name contains `.sourceforge.net`; SourceForge uses a non-standard mapping from CVSROOT/module to server name.

How 'Fetch list...' finds the list of modules

In the Checkout Dialog, there is a button named **Fetch list...**. Clicking this causes TortoiseCVS to attempt determining which modules exist on the selected CVS server. It does this by using the following algorithm:

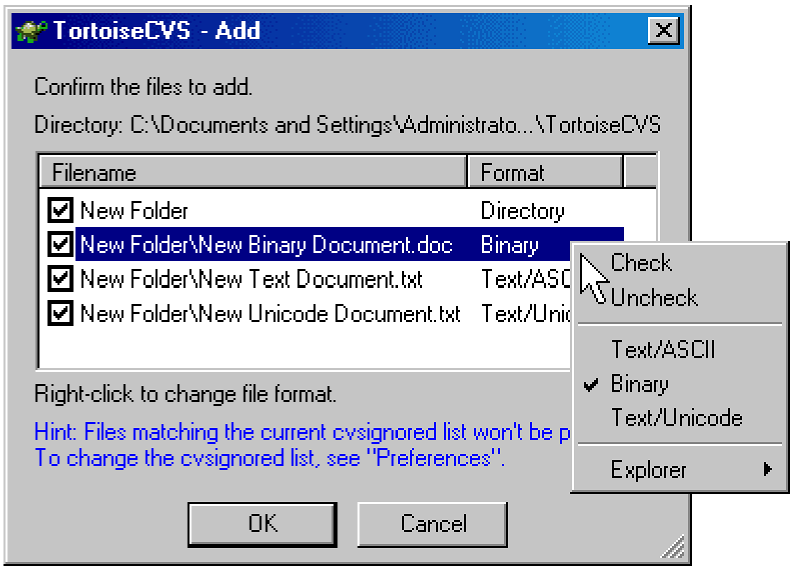
1. Run the command **cvs co -c** to obtain the contents of `CVSROOT/modules`. Note that the CVS administrator must maintain this file manually, so it might be out of date (or empty).
2. Run the command **cvs ls** to obtain all existing modules. This is a reliable way of obtaining the module list, but only works if the server runs CVSNT or a recent version of cvshome.org CVS.
3. Attempt to guess the Web Log URL (see How Web Log Autodetects the Server URL) and determine the available modules by parsing the output from the Web Log server. This of course only works if the CVS server provides Web Log.

Chapter 6. Dialog Reference for TortoiseCVS

This chapter details every dialog within TortoiseCVS.

Add Dialog

Figure 6.1. Add Dialog



TortoiseCVS - Add

Confirm the files to add.

Directory: C:\Documents and Settings\Administrato...\TortoiseCVS

Filename	Format
☑ New Folder	Directory
☑ New Folder\New Binary Document.doc	Binary
☑ New Folder\New Text Document.txt	Text/ASC
☑ New Folder\New Unicode Document.txt	Text/Unic

Right-click to change file format.

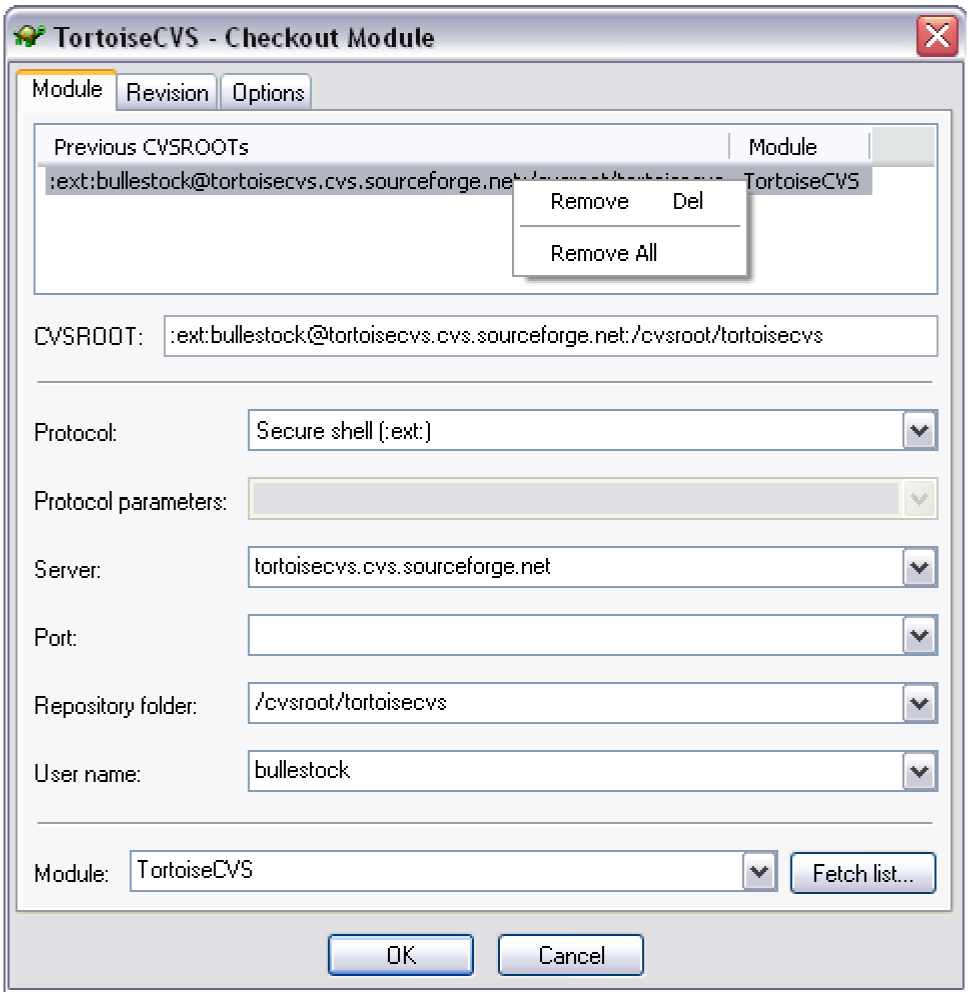
Hint: Files matching the current cvsignored list won't be p
To change the cvsignored list, see "Preferences".

Note

TODO: Write

Checkout Dialog

Figure 6.2. Checkout Dialog - Module Tab



TortoiseCVS - Checkout Module

Module | Revision | Options

Previous CVSROOTs	Module
:ext:bullestock@tortoisecvs.cvs.sourceforge.net:/cvsroot/tortoisecvs	TortoiseCVS

CVSROOT: :ext:bullestock@tortoisecvs.cvs.sourceforge.net:/cvsroot/tortoisecvs

Protocol: Secure shell (:ext:)

Protocol parameters:

Server: tortoisecvs.cvs.sourceforge.net

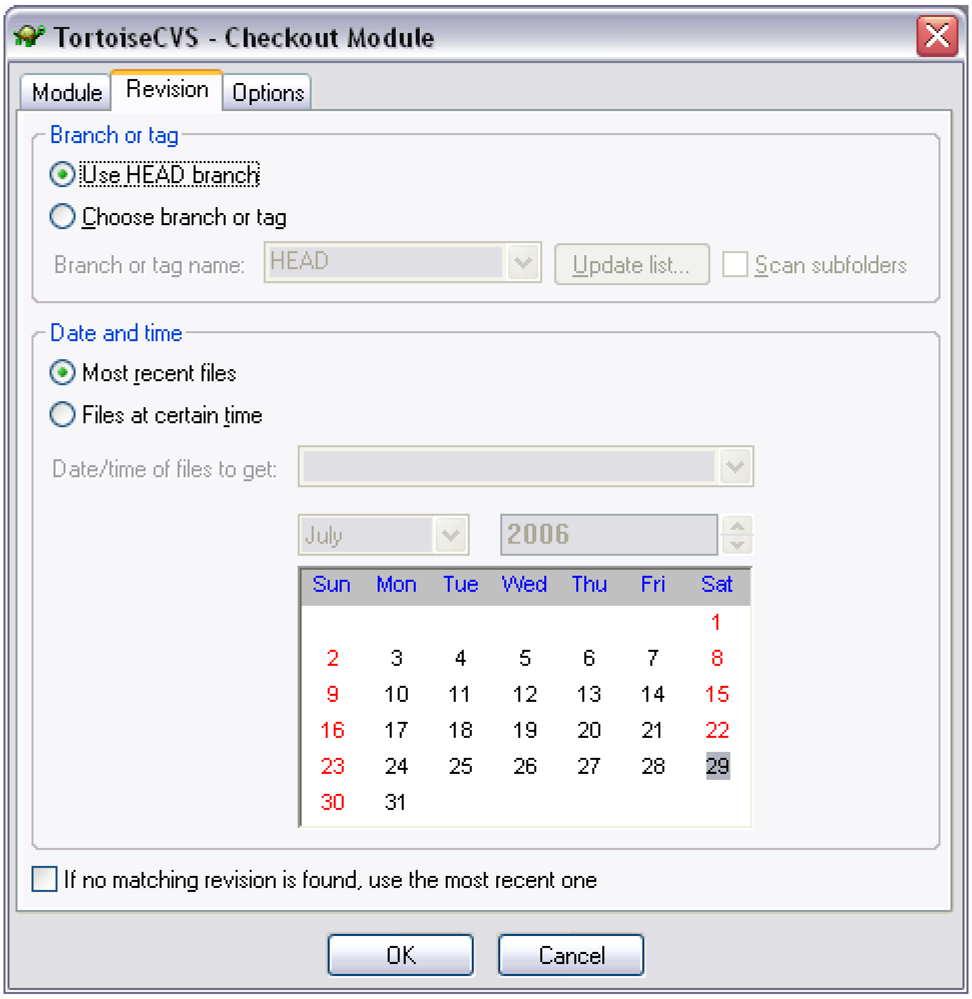
Port:

Repository folder: /cvsroot/tortoisecvs

User name: bullestock

Module: TortoiseCVS

Figure 6.3. Checkout Dialog - Revision Tab



Branch or tag

[✓ Use HEAD branch]

Choose branch or tag

Branch or tag name: HEAD

Scan subfolders

Date and time

[✓ Most recent files]

Files at certain time

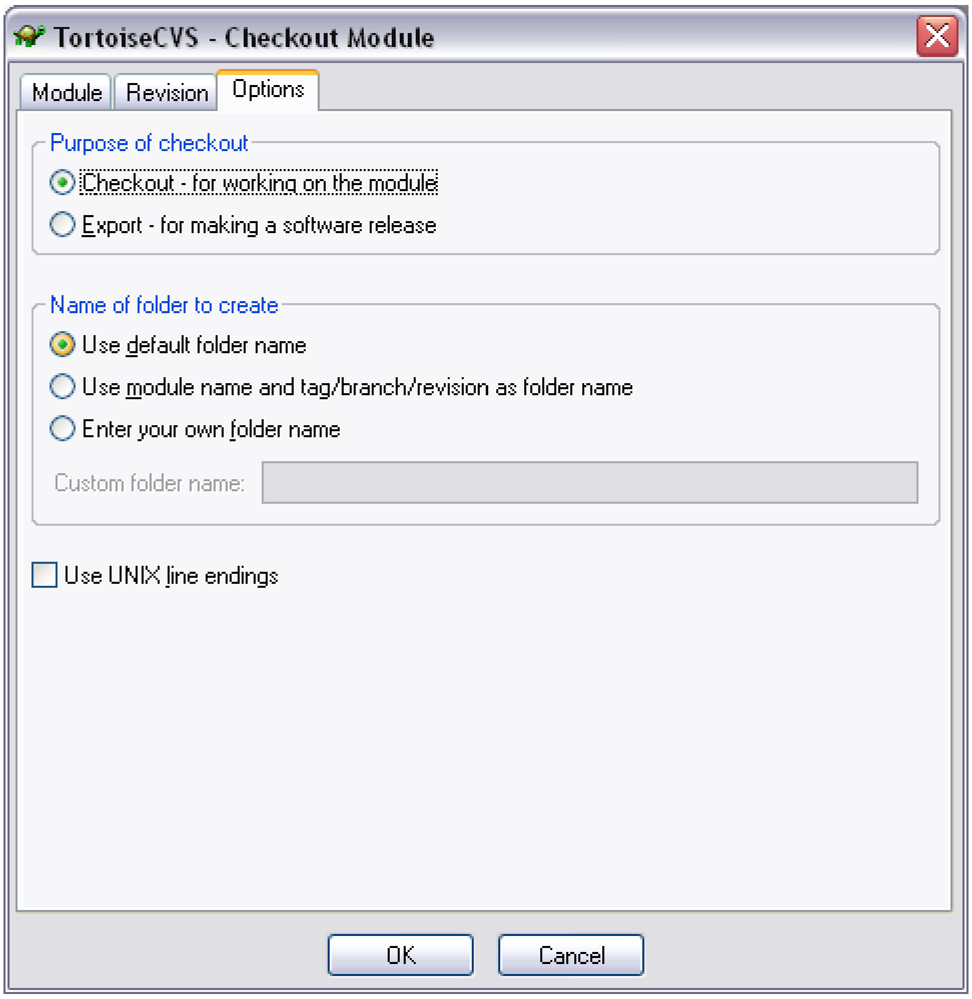
Date/time of files to get:

July 2006

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

If no matching revision is found, use the most recent one

Figure 6.4. Checkout Dialog - Options Tab


TortoiseCVS - Checkout Module

Purpose of checkout

[✓ Checkout - for working on the module]

Export - for making a software release

Name of folder to create

[✓ Use default folder name]

Use module name and tag/branch/revision as folder name

Enter your own folder name

Custom folder name:

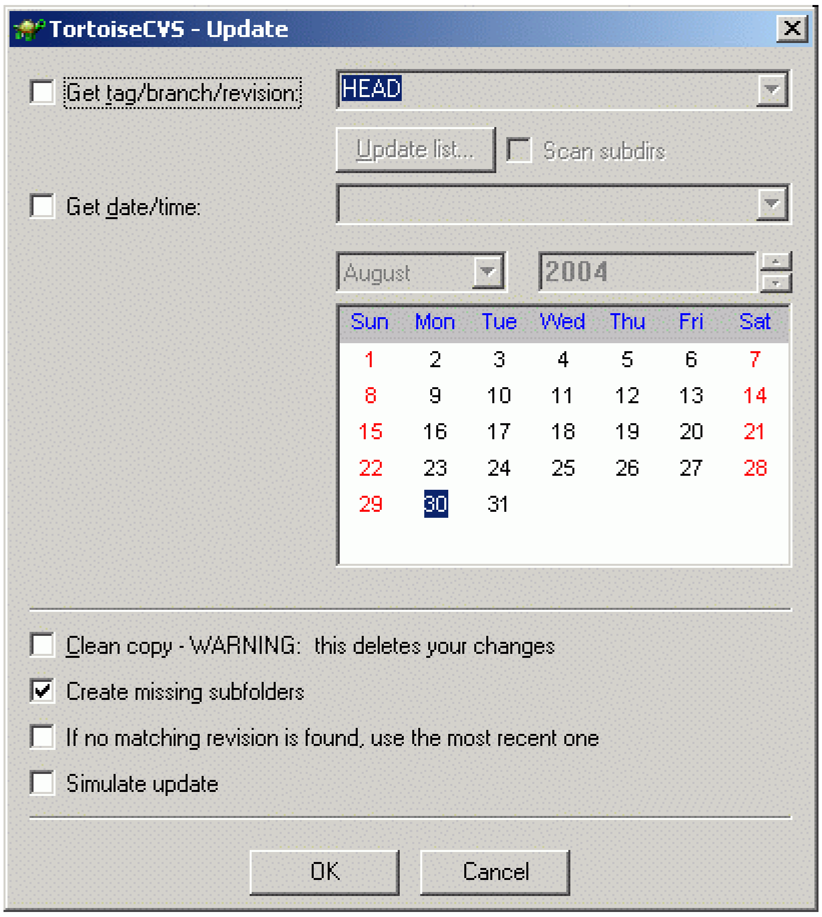
☐ Use UNIX line endings

Note

TODO: Write

Update Special Dialog

Figure 6.5. Update Special Dialog


TortoiseCVS - Update

- [ ] Get tag/branch/revision: HEAD
- Update list... [ ] Scan subdirs
- [ ] Get date/time:

August 2004

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

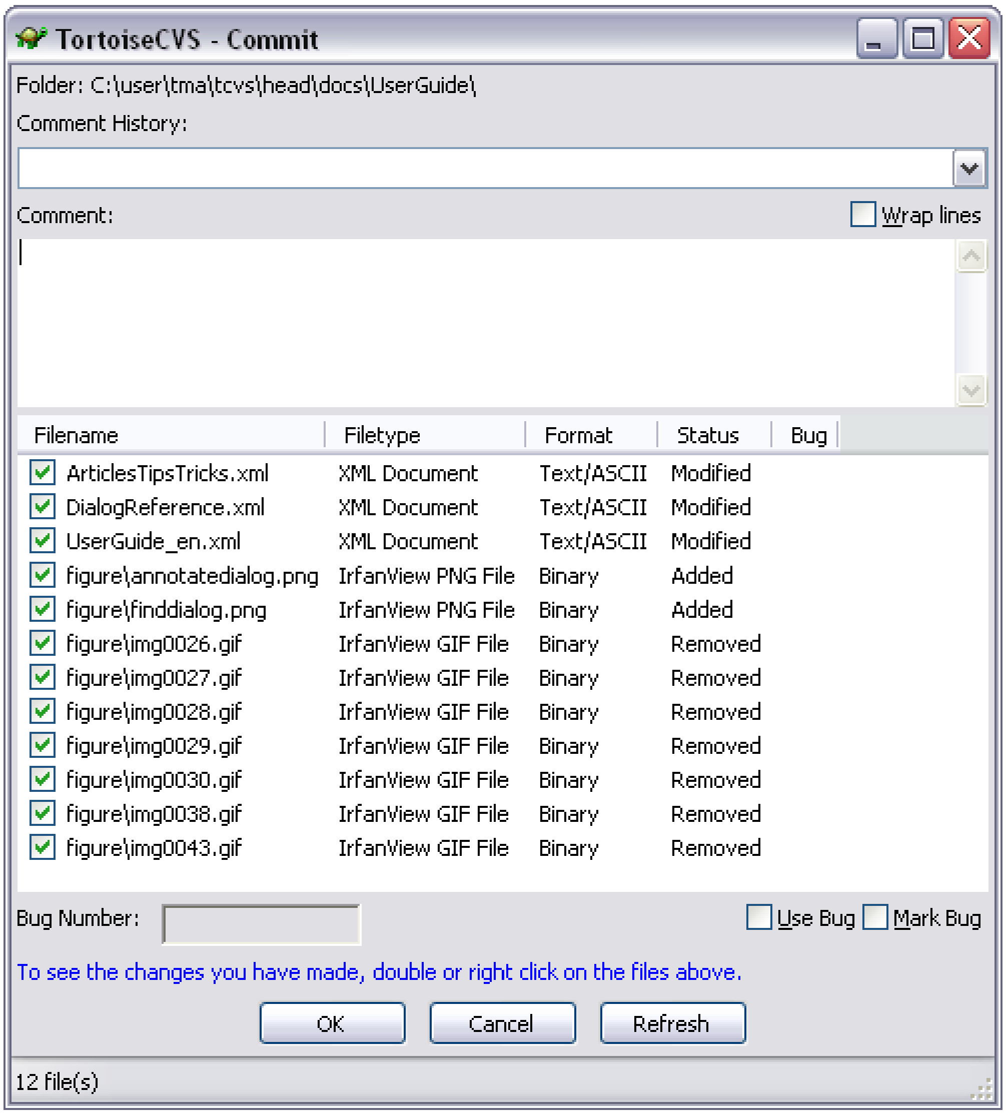
- [ ] Clean copy - WARNING: this deletes your changes
- [✓] Create missing subfolders
- [ ] If no matching revision is found, use the most recent one
- [ ] Simulate update

Note

TODO: Write

Commit Dialog

Figure 6.6. Commit Dialog



Folder: C:\user\tma\tcvs\head\docs\UserGuide\

Comment History:

Comment: [ ] Wrap lines

Filename	Filetype	Format	Status	Bug
✓ ArticlesTipsTricks.xml	XML Document	Text/ASCII	Modified	
✓ DialogReference.xml	XML Document	Text/ASCII	Modified	
✓ UserGuide_en.xml	XML Document	Text/ASCII	Modified	
✓ figure\annotatedialog.png	IrfanView PNG File	Binary	Added	
✓ figure\finddialog.png	IrfanView PNG File	Binary	Added	
✓ figure\img0026.gif	IrfanView GIF File	Binary	Removed	
✓ figure\img0027.gif	IrfanView GIF File	Binary	Removed	
✓ figure\img0028.gif	IrfanView GIF File	Binary	Removed	
✓ figure\img0029.gif	IrfanView GIF File	Binary	Removed	
✓ figure\img0030.gif	IrfanView GIF File	Binary	Removed	
✓ figure\img0038.gif	IrfanView GIF File	Binary	Removed	
✓ figure\img0043.gif	IrfanView GIF File	Binary	Removed	

Bug Number: [ ] Use Bug [ ] Mark Bug

To see the changes you have made, double or right click on the files above.

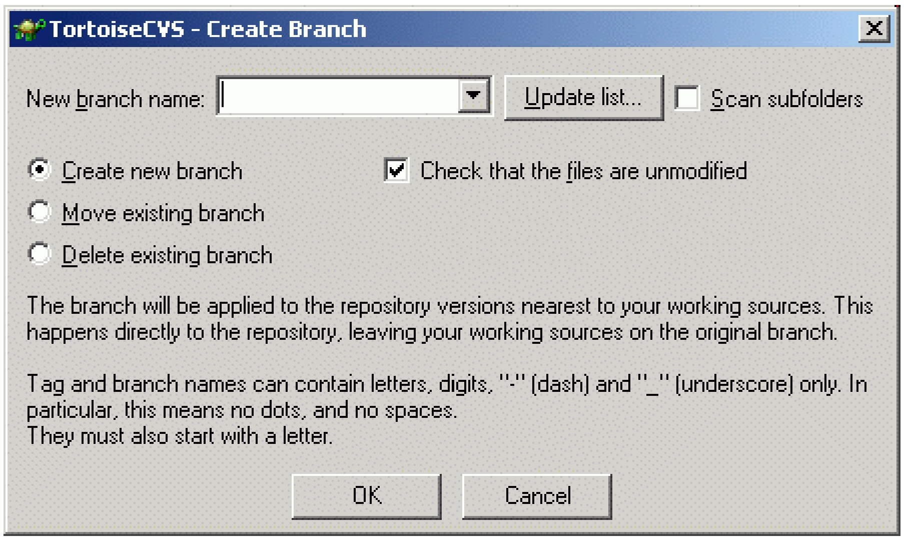
12 file(s)

Note

TODO: Write

Branch Dialog

Figure 6.7. Branch Dialog



The branch will be applied to the repository versions nearest to your working sources. This happens directly to the repository, leaving your working sources on the original branch.

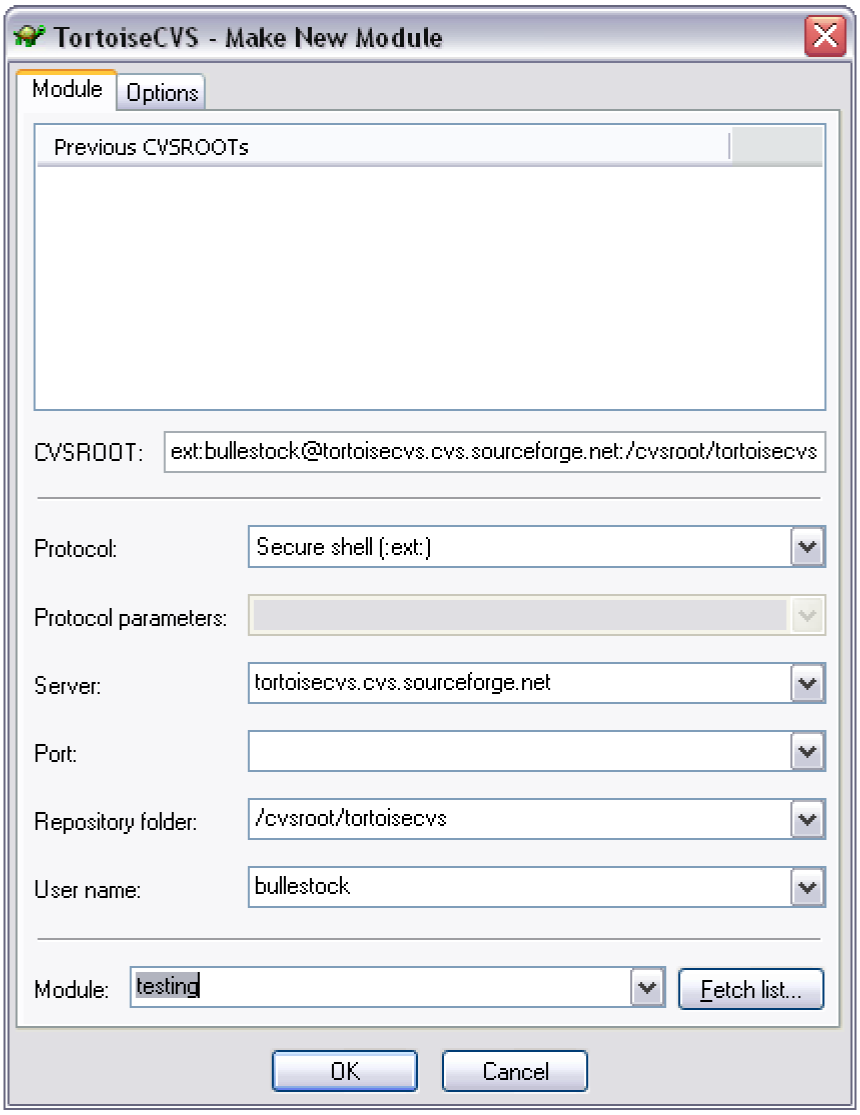
Tag and branch names can contain letters, digits, "-" (dash) and "_" (underscore) only. In particular, this means no dots, and no spaces.
They must also start with a letter.

Note

TODO: Write... Screenshot

Make New Module Dialog

Figure 6.8. Make new module Dialog - Module Tab



CVSROOT: ext:bullestock@tortoisecvs.cvs.sourceforge.net:/cvsroot/tortoisecvs

Protocol: Secure shell (:ext:)

Protocol parameters:

Server: tortoisecvs.cvs.sourceforge.net

Port:

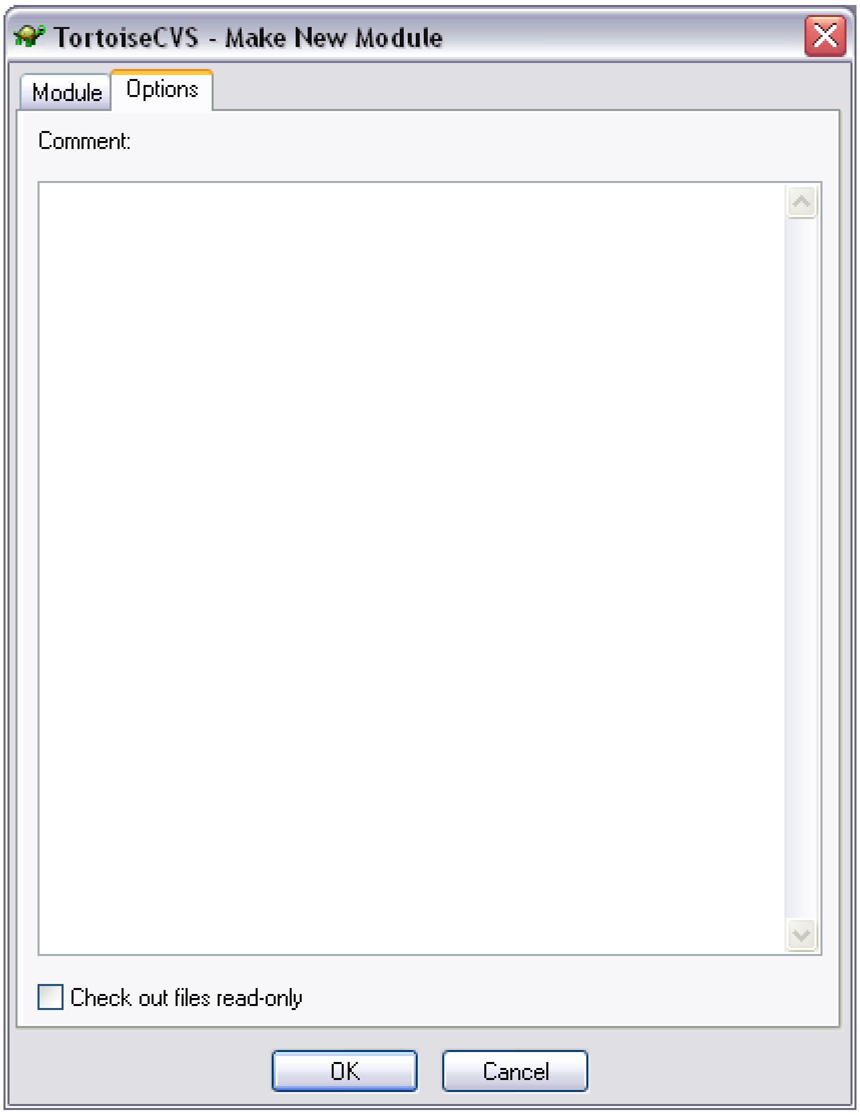
Repository folder: /cvsroot/tortoisecvs

User name: bullestock

Module: testing

The main tab in this dialog is used exactly like that in the Checkout dialog; please see the detailed description on that page.

Figure 6.9. Make new module Dialog - Options Tab

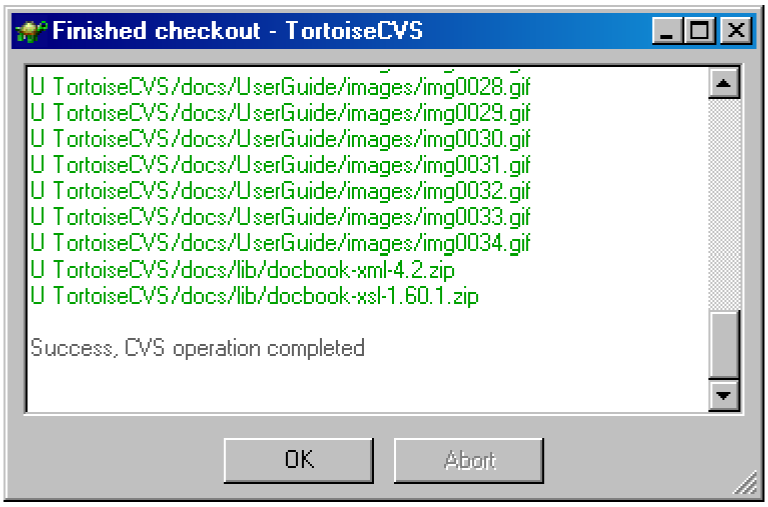


The **Options** tab has two fields:

- The **Comment** field allows you to enter a description of the new module.
- The **Check out files read-only** options causes a *watch* to be set for the module. See the section called "Watch, Edit and Unedit" for more information.

Progress Dialog

Figure 6.10. Progress Dialog



```
U TortoiseCVS/docs/UserGuide/images/img0028.gif
U TortoiseCVS/docs/UserGuide/images/img0029.gif
U TortoiseCVS/docs/UserGuide/images/img0030.gif
U TortoiseCVS/docs/UserGuide/images/img0031.gif
U TortoiseCVS/docs/UserGuide/images/img0032.gif
U TortoiseCVS/docs/UserGuide/images/img0033.gif
U TortoiseCVS/docs/UserGuide/images/img0034.gif
U TortoiseCVS/docs/lib/docbook-xml-4.2.zip
U TortoiseCVS/docs/lib/docbook-xsl-1.60.1.zip

Success, CVS operation completed
```

This dialog shows each CVS command that TortoiseCVS executes, as well as any information, errors, or warnings generated by CVS. The different types of messages are highlighted in different colours (you can customize these colours using the Appearance tab in the Preferences dialog).

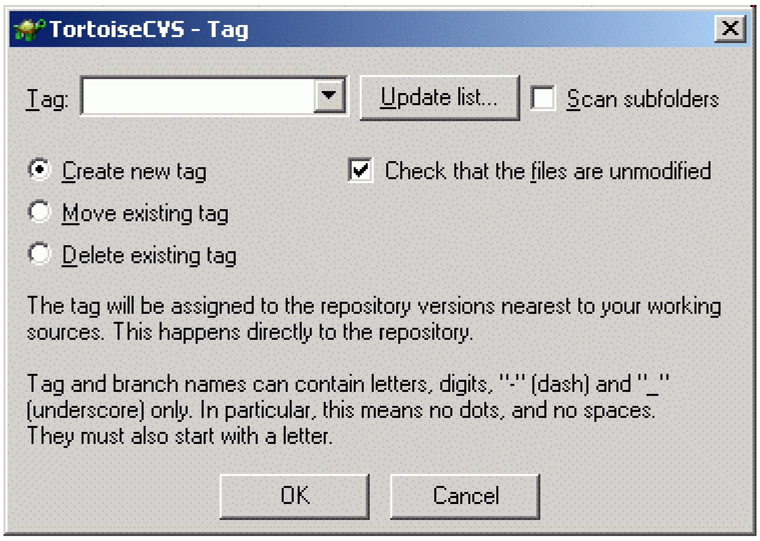
During an Update, the Progress Dialog indicates whether or not each file has changed in CVS or on the local PC, using the following single-letter codes:

C	The file has changed both in CVS and locally. TortoiseCVS attempted to merge both sets of changes into the local copy, but could not (probably because both changes were made to the same part of the file). For more information, see Resolving Conflicts.
M	The file has only changed on the local PC. TortoiseCVS will not change the local copy.
P	The file has changed in CVS, and TortoiseCVS has updated the local copy to match.
U	The file exists in CVS, but not on the local machine. TortoiseCVS will create a copy on the local PC. You will also see this for files which have changed in CVS (as for *P*.
A	The file has been added to CVS, but not yet committed.
R	The file has been removed from CVS, but not yet committed.

On the Main tab in the Preferences dialog, you can customize the closing behaviour of the dialog. This behaviour can also be overridden on a case-to-case basis using the **Close on completion** checkbox.

Tag Dialog

Figure 6.11. Tag Dialog



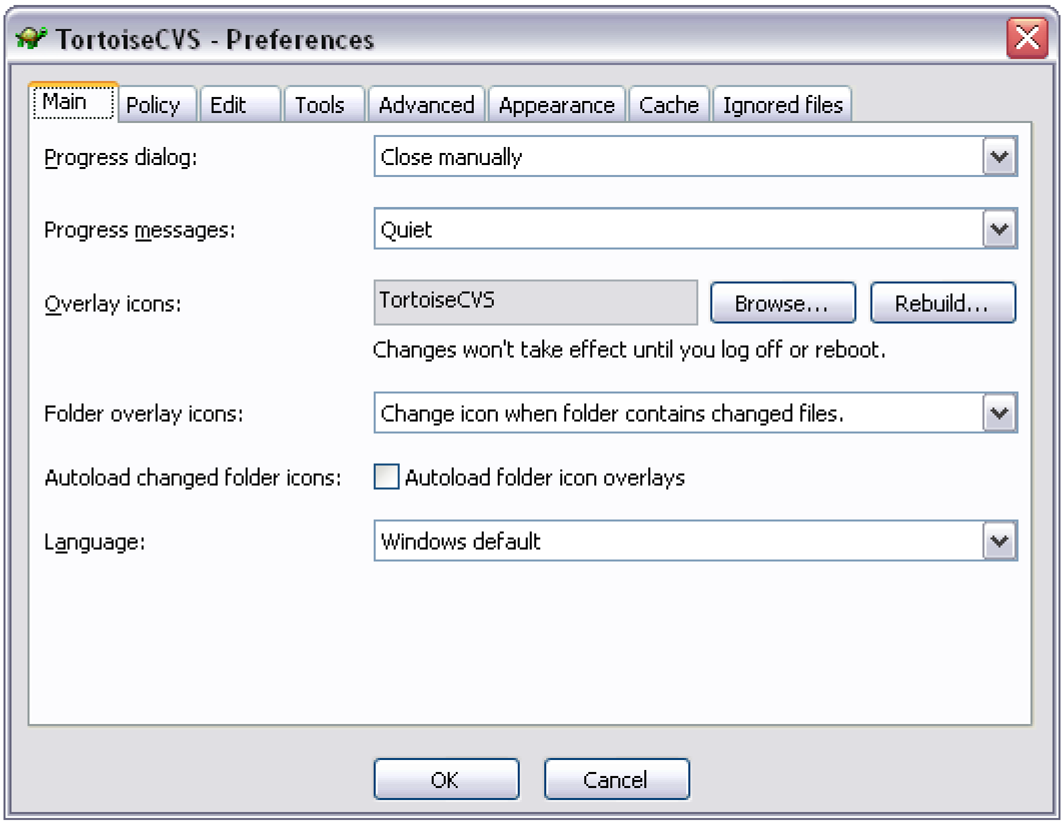
The tag will be assigned to the repository versions nearest to your working sources. This happens directly to the repository.

Tag and branch names can contain letters, digits, "-" (dash) and "_" (underscore) only. In particular, this means no dots, and no spaces. They must also start with a letter.

Preferences Dialog

The preference dialog is divided into tabs. Each tab contains a group of related preferences.

Figure 6.12. Preferences Dialog - Main Tab



Progress dialog: Close manually

Progress messages: Quiet

Overlay icons: TortoiseCVS

Changes won't take effect until you log off or reboot.

Folder overlay icons: Change icon when folder contains changed files.

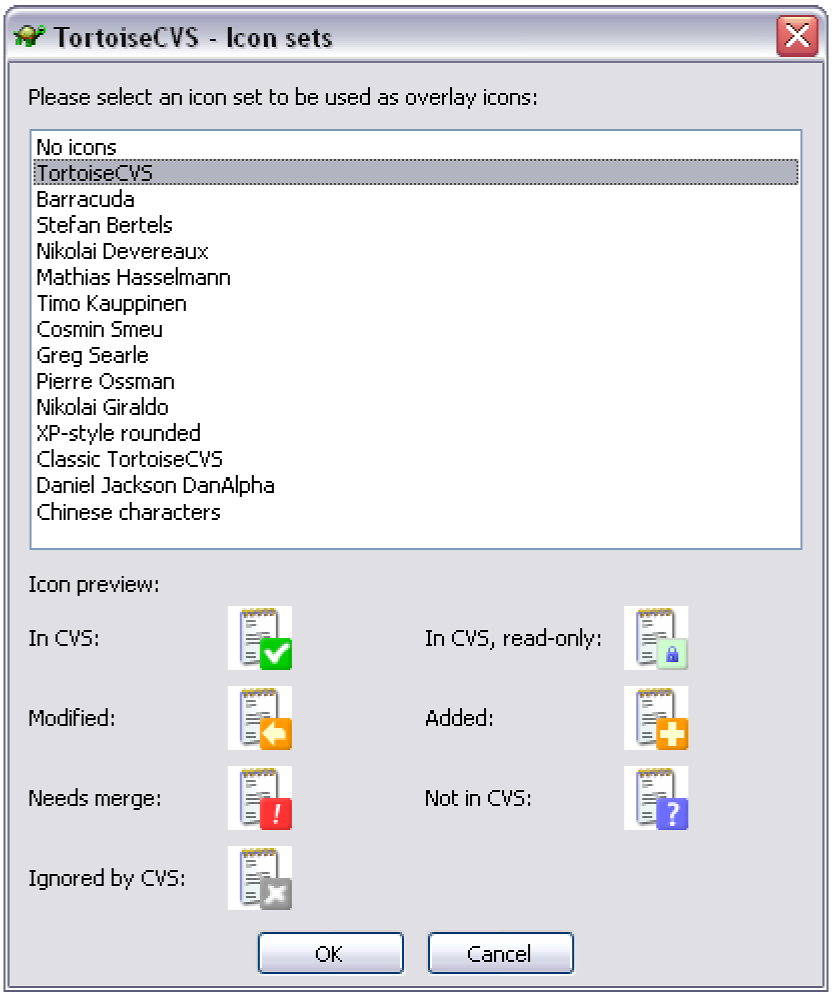
Autoload changed folder icons: ☐ Autoload folder icon overlays

Language: Windows default

The **Main** tab offers access to the most basic settings.

- The **Progress dialog** setting controls if the progress dialog will close itself automatically.
- The **Overlay icons** setting allows you to change the set of overlay icons used. The **Browse...** button causes the icon chooser dialog to be shown:

Figure 6.13. Preferences Dialog - Iconset Chooser


TortoiseCVS - Icon sets

Please select an icon set to be used as overlay icons:

No icons
TortoiseCVS
Barracuda
Stefan Bertels
Nikolai Devereaux
Mathias Hasselmann
Timo Kauppinen
Cosmin Smeu
Greg Searle
Pierre Ossman
Nikolai Giraldo
XP-style rounded
Classic TortoiseCVS
Daniel Jackson DanAlpha
Chinese characters

Icon preview:

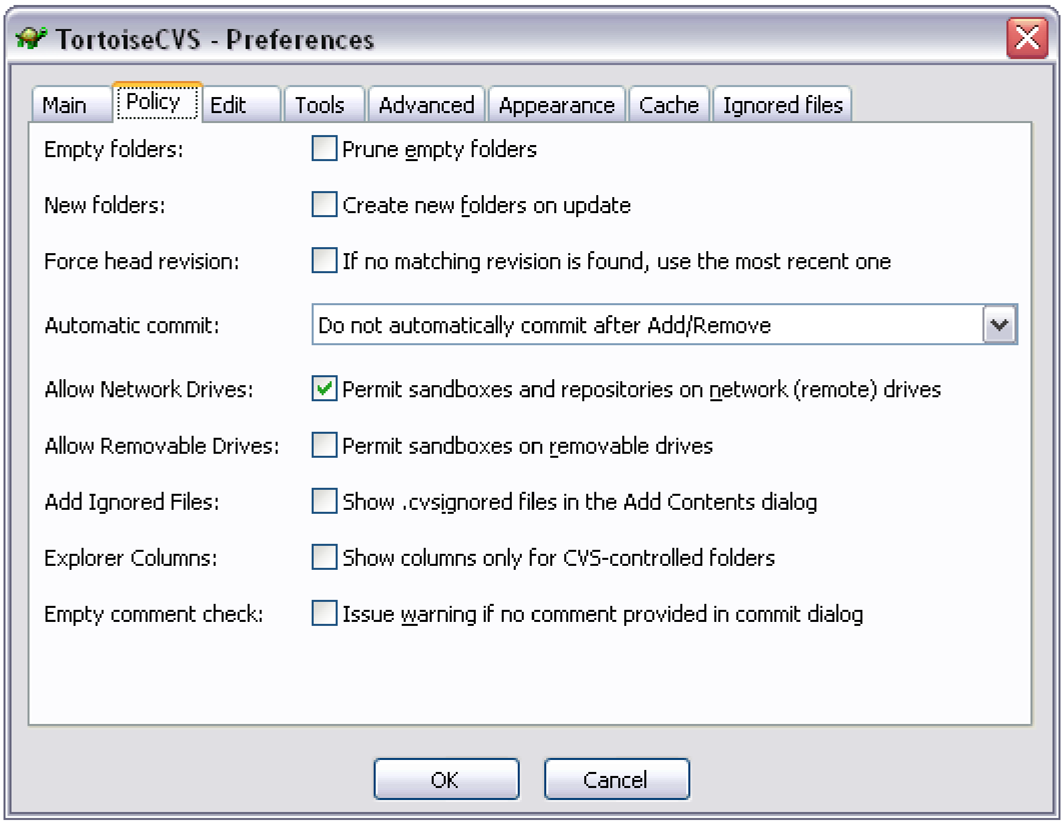
In CVS:
In CVS, read-only:
Modified:
Added:
Needs merge:
Not in CVS:
Ignored by CVS:

The **Rebuild...** button rebuilds Windows™ Explorer's icon cache. This can sometimes be necessary if the overlay icons are not showing correctly.

- The **Folder overlay icons** setting controls if and how the overlay icon for a folder reflects the state of the files and folders inside it.
- The **Autoload changed folder icons** setting (only applicable if you have selected something else than "Don't change icon" above) determines if the folder status is updated automatically, or only when you select **Refresh folder status** from the CVS context menu in Explorer.
- Finally, the **Language** setting allows you to change the language used for menus and dialogs.

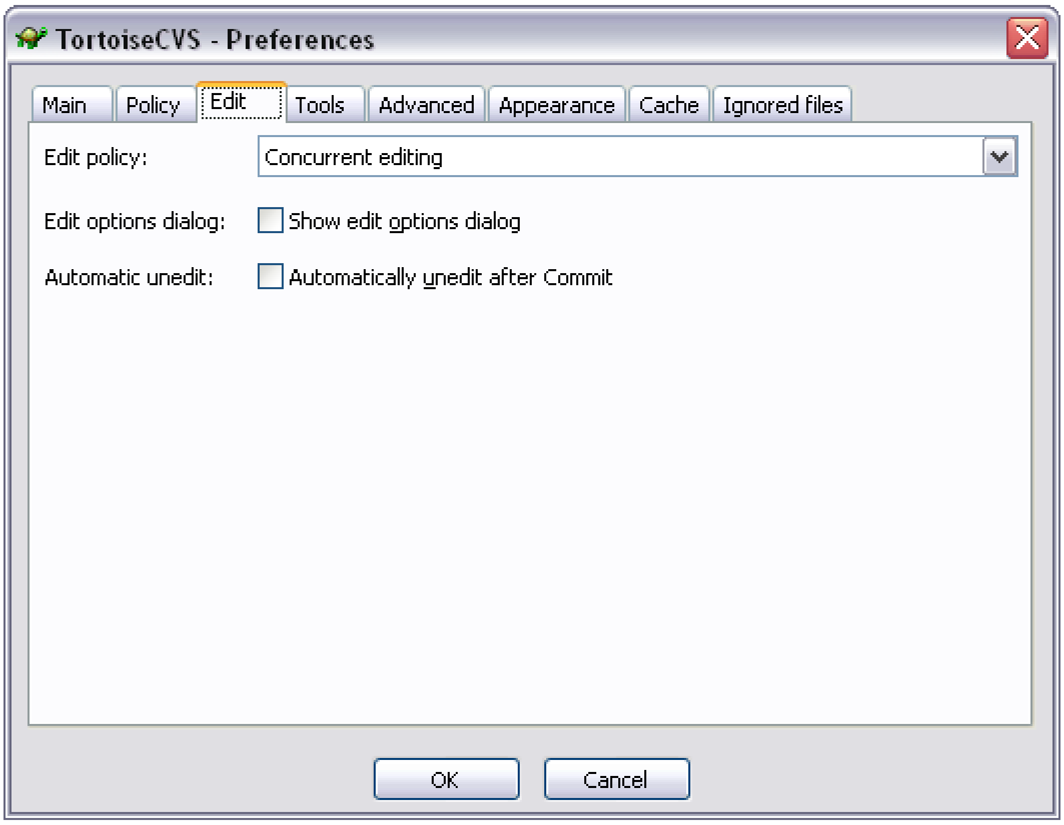
The **Policy** tab...

Figure 6.14. Preferences Dialog - Policy Tab



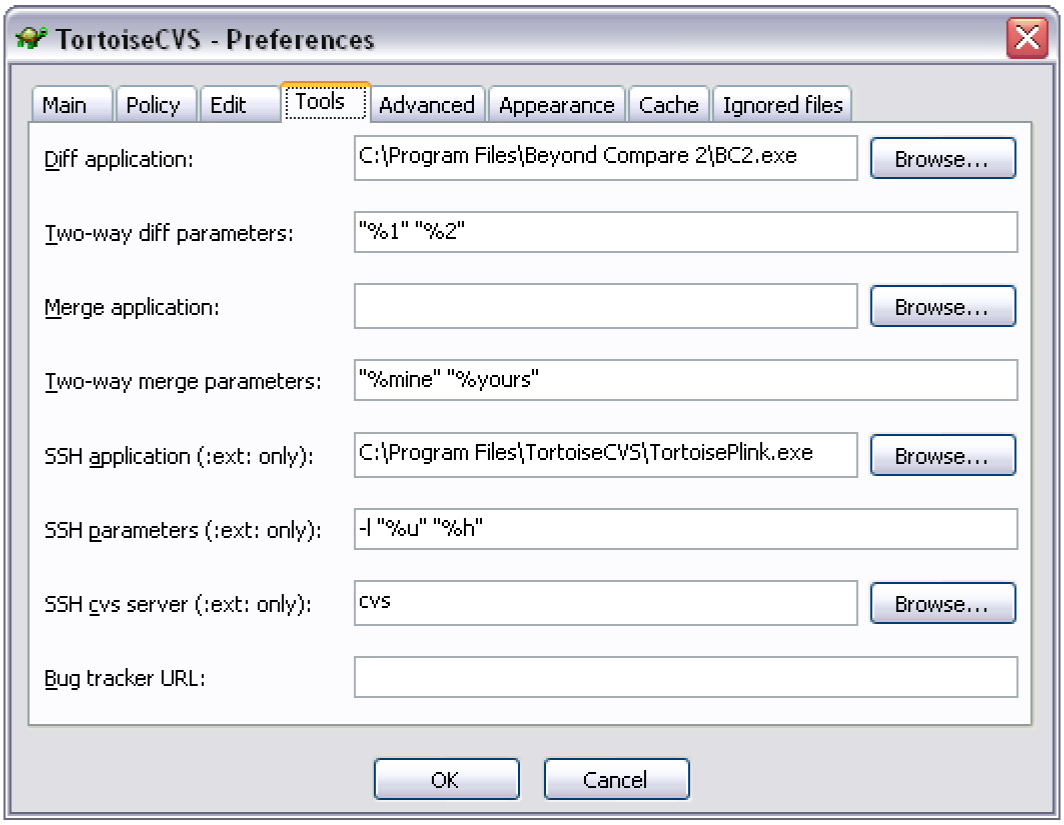
Empty folders:	☐ Prune empty folders
New folders:	☐ Create new folders on update
Force head revision:	☐ If no matching revision is found, use the most recent one
Automatic commit:	Do not automatically commit after Add/Remove
Allow Network Drives:	✓ Permit sandboxes and repositories on network (remote) drives
Allow Removable Drives:	☐ Permit sandboxes on removable drives
Add Ignored Files:	☐ Show .cvsignored files in the Add Contents dialog
Explorer Columns:	☐ Show columns only for CVS-controlled folders
Empty comment check:	☐ Issue warning if no comment provided in commit dialog

Figure 6.15. Preferences Dialog - Edit Tab



Edit policy:	Concurrent editing
Edit options dialog:	☐ Show edit options dialog
Automatic unedit:	☐ Automatically unedit after Commit

Figure 6.16. Preferences Dialog - Tools Tab



Diff application: C:\Program Files\Beyond Compare 2\BC2.exe

Two-way diff parameters: "%1" "%2"

Merge application:

Two-way merge parameters: "%mine" "%yours"

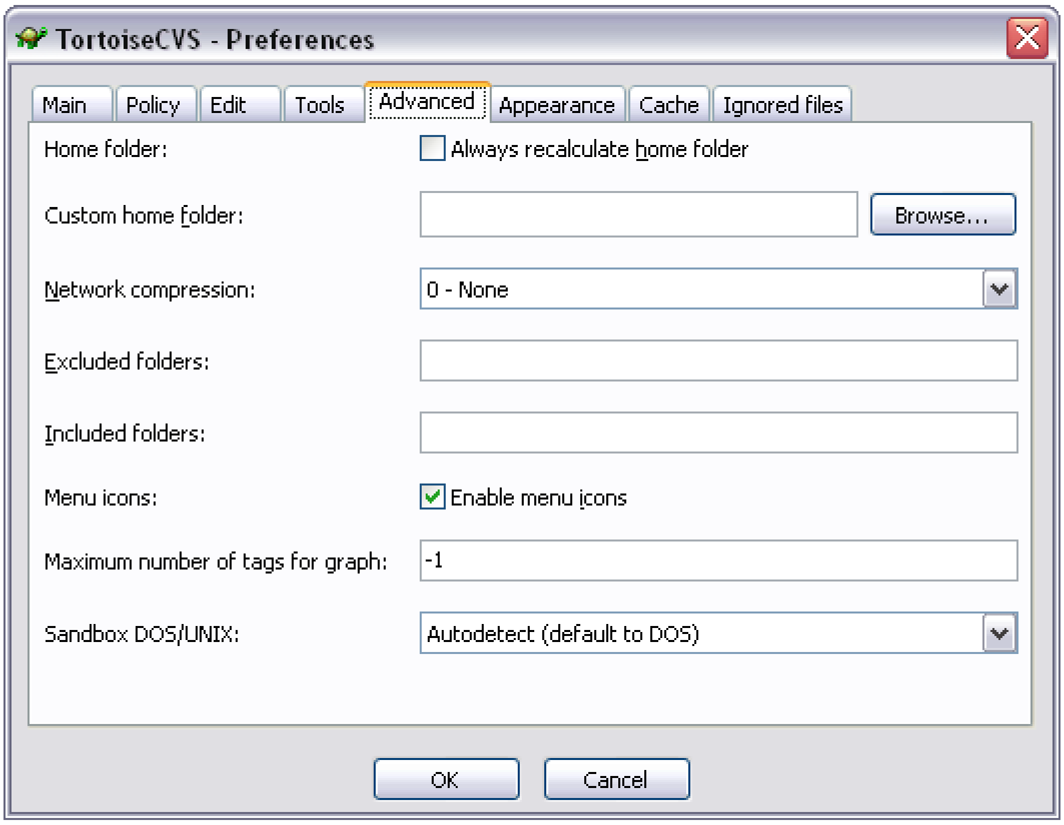
SSH application (:ext: only): C:\Program Files\TortoiseCVS\TortoisePlink.exe

SSH parameters (:ext: only): -l "%u" "%h"

SSH cvs server (:ext: only): cvs

Bug tracker URL:

Figure 6.17. Preferences Dialog - Advanced Tab



Home folder: ☐ Always recalculate home folder

Custom home folder:

Network compression: 0 - None

Excluded folders:

Included folders:

Menu icons: ☑ Enable menu icons

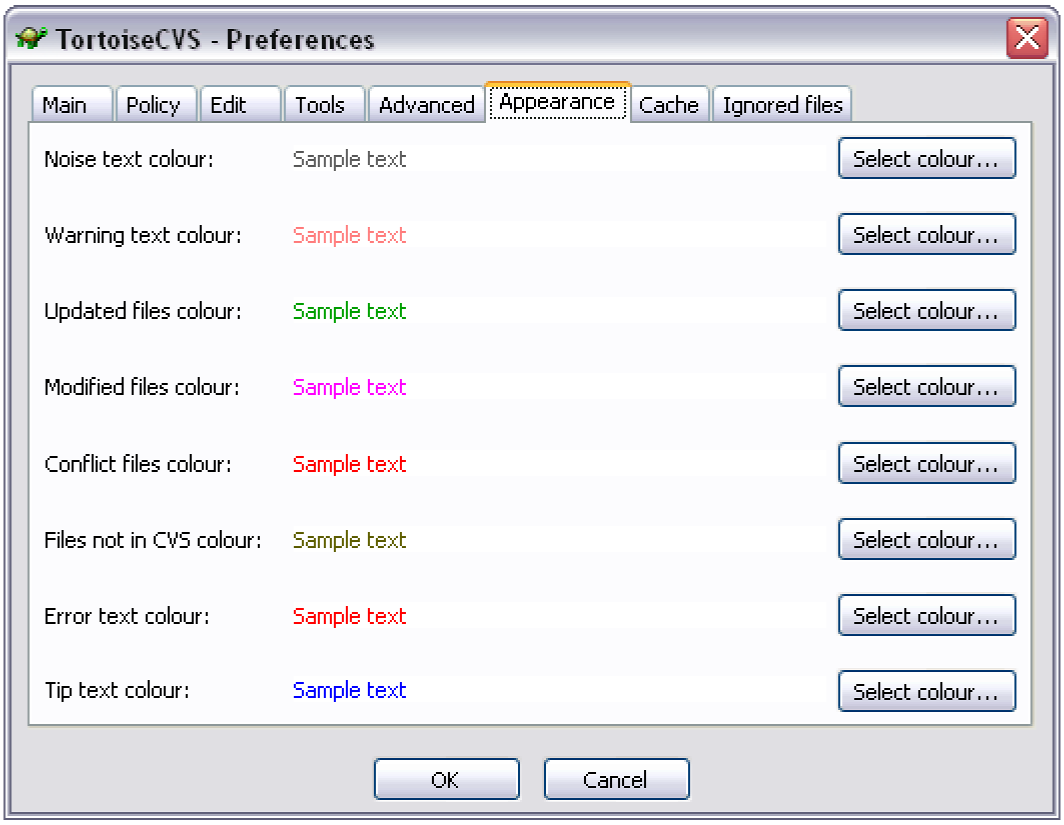
Maximum number of tags for graph: -1

Sandbox DOS/UNIX: Autodetect (default to DOS)

The **Advanced** tab contains settings that the typical user does not need to change.

- The **Sandbox DOS/UNIX** setting controls how line endings are preserved when doing CVS operations in an existing sandbox (as opposed to the **Use UNIX line endings** checkbox in the Checkout and Make new module dialogs, which controls how the sandbox is initially created).
  - **Autodetect (default to DOS)** and **Autodetect (default to UNIX)**: TortoiseCVS checks which line ending format is used and gives the appropriate options to CVS so that the existing line ending format is preserved.
  - **DOS** and **UNIX**: TortoiseCVS ignores the existing line ending format and forces CVS to create the specified line ending format.

Figure 6.18. Preferences Dialog - Appearance Tab



Noise text colour:	Sample text	Select colour...
Warning text colour:	Sample text	Select colour...
Updated files colour:	Sample text	Select colour...
Modified files colour:	Sample text	Select colour...
Conflict files colour:	Sample text	Select colour...
Files not in CVS colour:	Sample text	Select colour...
Error text colour:	Sample text	Select colour...
Tip text colour:	Sample text	Select colour...

Figure 6.19. Preferences Dialog - Cache Tab

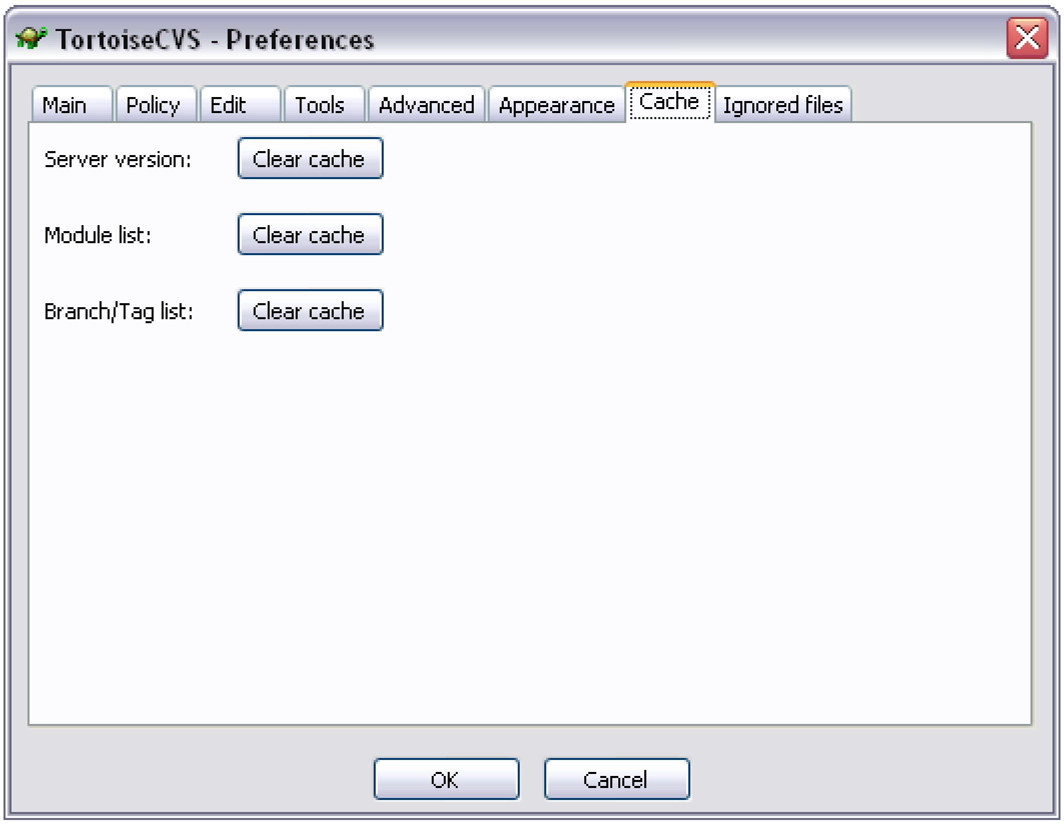
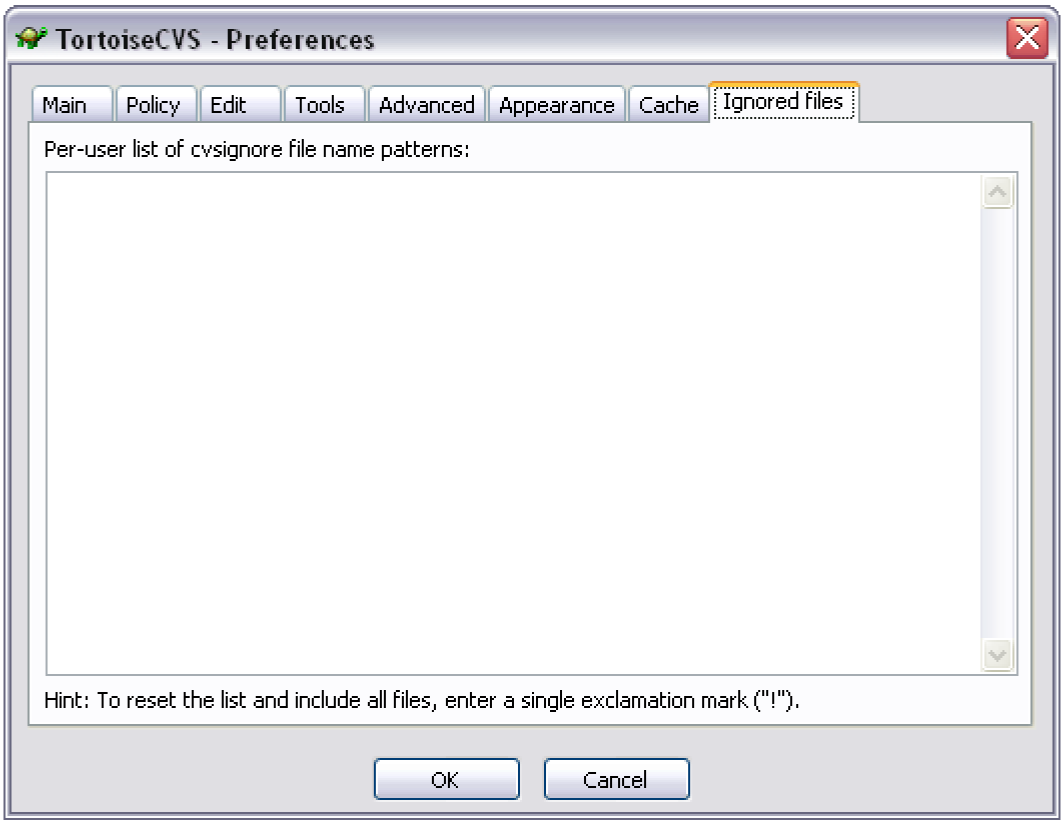


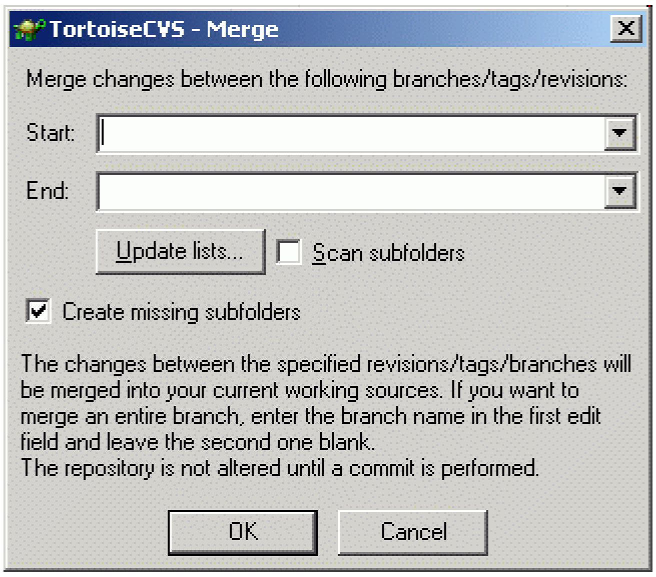
Figure 6.20. Preferences Dialog - Ignored Files Tab


TortoiseCVS - Preferences
Main
Policy
Edit
Tools
Advanced
Appearance
Cache
Ignored files
Per-user list of cvsignore file name patterns:
OK
Cancel

Hint: To reset the list and include all files, enter a single exclamation mark ("!").

Merge Dialog

Figure 6.21. Merge Dialog



Merge changes between the following branches/tags/revisions:

Start:

End:

[ ] Scan subfolders

[✓] Create missing subfolders

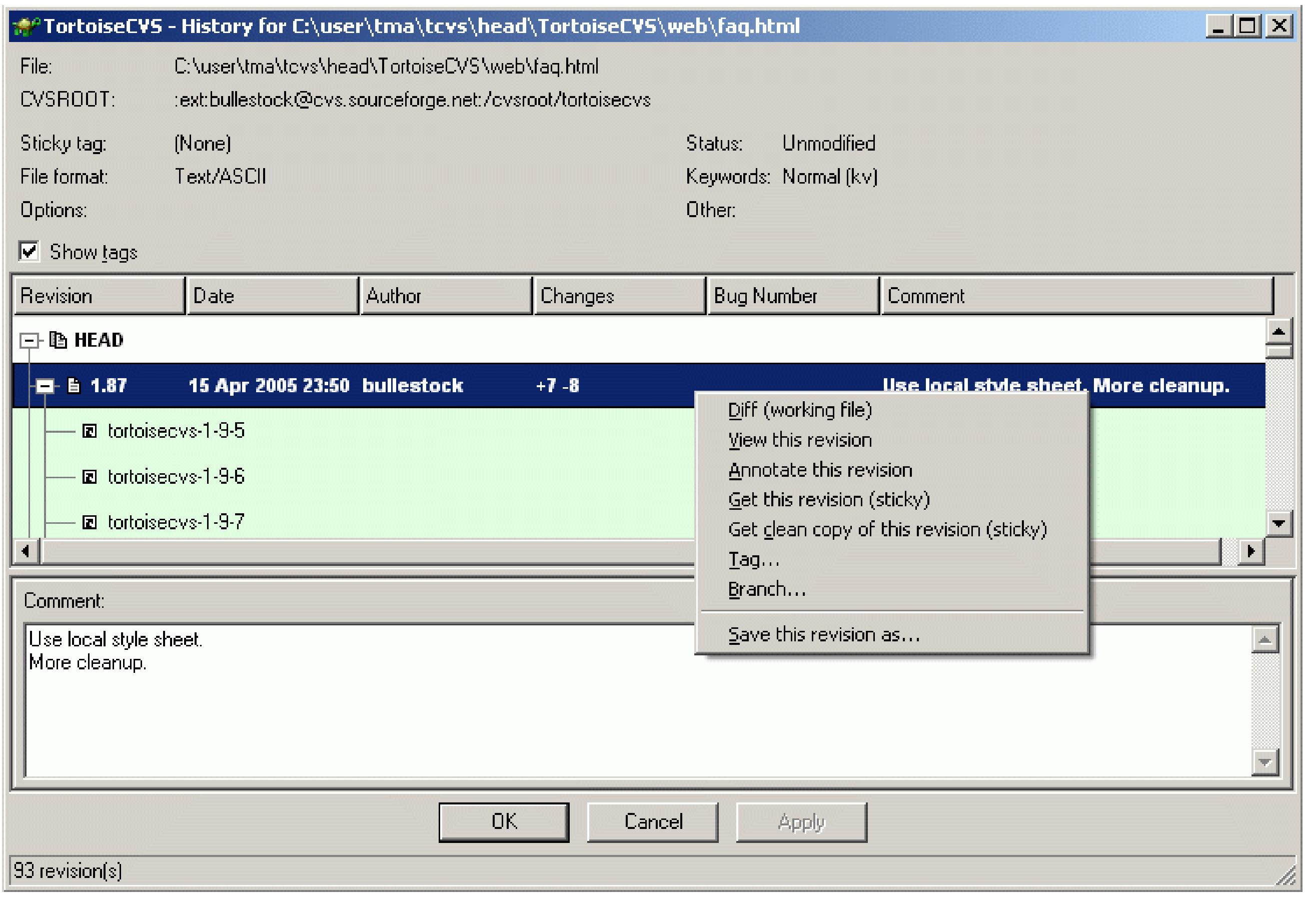
The changes between the specified revisions/tags/branches will be merged into your current working sources. If you want to merge an entire branch, enter the branch name in the first edit field and leave the second one blank.
The repository is not altered until a commit is performed.

Note

TODO: Write... Screenshot

History Dialog

Figure 6.22. History Dialog


TortoiseCVS - History for C:\user\tma\tcvs\head\TortoiseCVS\web\faq.html

File:	C:\user\tma\tcvs\head\TortoiseCVS\web\faq.html		
CVSROOT:	:ext:bullestock@cvs.sourceforge.net:/cvsroot/tortoisecvs		
Sticky tag:	(None)	Status:	Unmodified
File format:	Text/ASCII	Keywords:	Normal (kv)
Options:		Other:	

☑ Show tags

Revision	Date	Author	Changes	Bug Number	Comment
HEAD					
1.87	15 Apr 2005 23:50	bullestock	+7 -8		Use local style sheet. More cleanup.
tortoisecvs-1-9-5					
tortoisecvs-1-9-6					
tortoisecvs-1-9-7					

Comment:

Use local style sheet.
More cleanup.

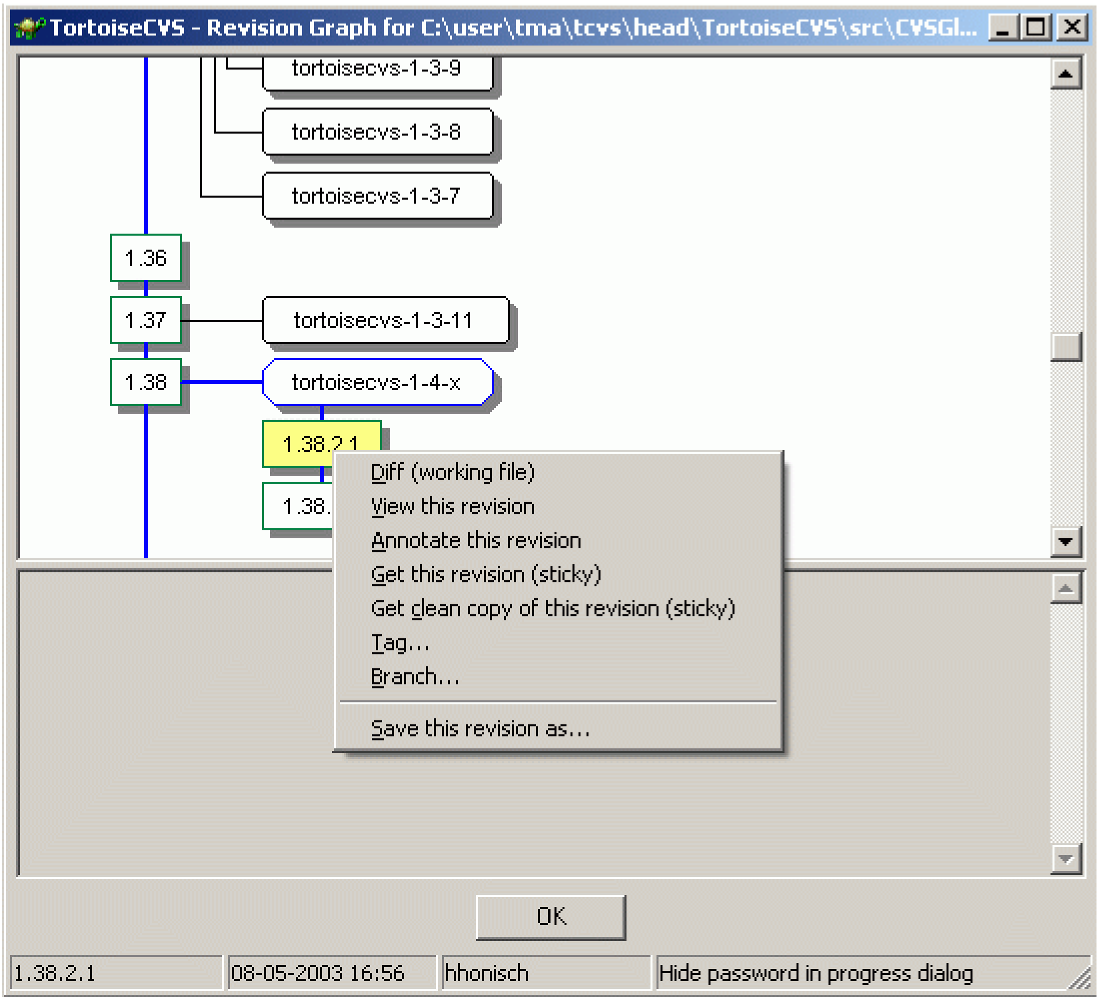
93 revision(s)

Note

TODO: Write

Revision Graph Dialog

Figure 6.23. Revision Graph Dialog

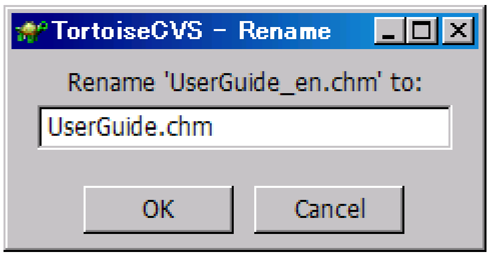


Note

TODO: Write. Unless the file is an executable, you can also select *view* to view that revision using the default application for the file type in question.

Rename Dialog

Figure 6.24. Rename Dialog

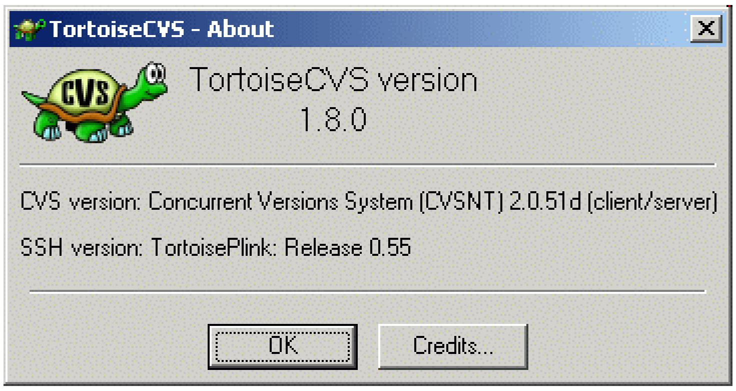


TODO: Write

TODO: Describe and show dialog with and without CVS server version

About Dialog

Figure 6.25. About Dialog



TortoiseCVS version
1.8.0

CVS version: Concurrent Versions System (CVSNT) 2.0.51d (client/server)

SSH version: TortoisePlink: Release 0.55

The About dialog shows the version of TortoiseCVS itself as well as the CVS client it uses, and the SSH application you have configured.

When invoked in a sandbox, the About dialog shows the version of the CVS server as well.

It also has a **Credits...** button, which opens another dialog listing the people who have contributed to TortoiseCVS development. If you hold the mouse over a name, a tooltip will show what that person has contributed.

Chapter 7. Articles, Tips and Tricks

Using TortoiseCVS with SourceForge

TortoiseCVS Client Installation Instructions

This document provides instructions for installing and using the TortoiseCVS CVS client software. These instructions are designed for SourceForge.net project developers who are using 32-bit Microsoft Windows platform. It is for developers who would like to work with the source code for a project which is hosted on SourceForge.net. For background information you may like to read the other available documents about CVS.

TortoiseCVS has advantages over WinCVS for work on SourceForge. It has a clearer interface which is integrated into Windows Explorer. WinCVS has a more complex and mature interface. For more obscure commands it may be easier or necessary to use WinCVS.

A particular benefit is that TortoiseCVS has an SSH client built into it, without special setup. Anyone can easily commit code to SourceForge.net projects using TortoiseCVS.

Installing TortoiseCVS

1. Go to http://tortoisecvs.org/ [http://www.tortoisecvs.org] and download the Download page.
2. To install, run the executable and follow the instructions.
3. Since TortoiseCVS is a shell extension, you need to reboot your machine.

Checking out an existing project

If you are working on an existing project first you now need to check out the code.

1. Make sure you have made an account on SourceForge.net, and that you have been added as a developer to the project you are going to work on. A project administrator needs to do this.
2. In Windows Explorer go to the folder that you would like to check the code out into.
3. Right click and from the context menu choose the command **CVS Checkout...**; it's next to the Tortoise icon.
4. You will need your user name on SourceForge.net, and the project UNIX name. Fill in the dialog as follows:
    - Protocol: Secure Shell (:ext:)
    - Server: cvs.sourceforge.net
    - Directory: /cvsroot/*projectname*
    - Username: *username*
5. Click **Fetch List** to get a choice of modules. You will need to enter your password in the dialog at this point. Then pick the module that you would like.
6. Choose **OK**.TortoiseCVS will now check out the code. You will need to enter your password.

7. You can now alter files and then select **CVS Commit** on the top level folder to make your changes in the repository. For more information read the TortoiseCVS User's Guide [http://www.tortoiscvs.org/docs/UserGuide_en/].

Get an error?

If you get an error saying something like `Could not chdir to home folder /home/users/r/rd/rdonkin: No such file or folder` then you need to log into the server with SSH to force creation of your home folder. Do this by connecting to `username@cvs.sf.net` either with Putty or a command line SSH.

Working on a new project

If you are working on a new project then you first need to create a new module in the repository.

1. Create a folder with the name you would like the module to have.
2. Put the files you would like to add in the folder.
3. Right click on the folder, and choose **Make New Module** from the CVS submenu.
4. Fill in the dialog as described in the previous section. Note that the module name is filled in for you.
5. Choose **OK**. You will need to enter your password twice, and the module will be made for you.
6. Proceed to add files and folders by right clicking on them and choosing **CVS Add** or **CVS Add All Recursively** from the context menu.
7. For more information read the TortoiseCVS Daily User's Guide [http://www.tortoisecvs.org/docs/UserGuide_en/]

Tired of entering your password all the time?

You need to make a public/private key pair. The public key goes on the server, and the private key sits on your hard drive. Now, when you connect the server can verify who you are because only you have your private key.

To further protect your private key it is encrypted on your hard disk with a passphrase. You can however choose an empty passphrase, in which case you never have to enter a password but anyone with access to your computer can access your SSH CVS account. Or you can use an SSH Agent (such as Pageant) which will remember your passphrase for the duration of one session, so you only have to enter it once.

TortoiseCVS uses a version of Plink as its SSH client. This means you can use PuTTYgen to make a public/private key pair, and you can use Pageant to point to where the private key is. First of all download them both from the PuTTY web site [http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html].

For information on how to use them read the PuTTY documentation [http://www.chiark.greenend.org.uk/~sgtatham/putty/docs.html] on PuTTYgen and Pageant. In particular, make sure you read the sections "8.2: Using PuTTYgen, the PuTTY key generator" and "9.3.1 Making Pageant automatically load keys on startup".

After making your public key with PuTTYgen you need to upload it using the SourceForge.net web interface. Log into the SourceForge web site, and go to your Account Preferences page. Down at the bottom is a section "Shell Account Information" with an entry "CVS/SSH Shared Authorized Keys". You need to add your public key there.

How CVS Differs from Microsoft Visual Source Safe™

TortoiseCVS differs from Visual Source Safe™ (VSS) in many ways. The most apparent difference may be that TortoiseCVS does not require users to lock the files they are working on, as VSS does by default. In fact, the CVS documentation even encourages users not to use file locking. In the rare occasion where several people have changed the same file at the same time, CVS will normally be able to merge their changes. If two or more developers have changed the same few lines, TortoiseCVS will report a conflict, insert directives in the file, and leave it to the developers to decide what to do. Such conflicts are very rare, as they normally occur as a result of lacking communication between the developers (eg. two people trying to fix the same problem).

Another important difference is that VSS gives you a server view, while TortoiseCVS shows a client view. In practice this means that, unlike VSS, TortoiseCVS will not tell you about changes in the repository until you do an update, or explicitly query the status of selected files. Changes reported by TortoiseCVS reflect modifications done by you after the last checkout, update or commit.

Creating new icons for TortoiseCVS

TortoiseCVS icons are Windows icon files that are overlaid on top of source code file icons in Explorer to show their status in CVS. The reason this technique works is that most of the area in a TortoiseCVS icon is transparent, allowing the source icon to show through. If you are designing your own TortoiseCVS icon set, you'll need to strike a careful balance between visibility of the original file and easy viewing of CVS states. Some icon sets achieve this balance by adding a distinctive colour or shape to one corner of each icon; others achieve the illusion of shading the entire icon by alternating transparent pixels with coloured ones. You can even combine these two ideas of unique shapes and semitransparent shading: see the "NG" icon set that ships with TortoiseCVS for an example.

1. Create your icon set using a graphics editor. For the best viewing results, you should create 16x16, 32x32, and 48x48 versions of each icon. The icon editors that ship with the Borland and Microsoft compilers can do this task easily, as can most shareware icon editors. If you prefer to stay with free software, you can use The GIMP [http://www.gimp.org] to create a PNG file for each icon size, and then combine them together with png2ico [http://www.winterdrache.de/freeware/png2ico].

2. Make sure you have created one correctly-named icon for each of the seven possible CVS states: `TortoiseNotInCVS.ico`, `TortoiseInCVS.ico`, `TortoiseInCVSReadOnly.ico`, `TortoiseChanged.ico`, `TortoiseConflict.ico`, `TortoiseAdded.ico`, and `TortoiseIgnored.ico`.

3. Create a folder inside the directory where you installed TortoiseCVS, name it something that matches the theme of your icon set, and copy your seven new icon files into the new folder.

4. Open the Registry Editor (Click **Start** → **Run** and type `regedit`). Navigate to `HKEY_LOCAL_MACHINE\SOFTWARE\TortoiseCVS\Icons`. Inside this key, create a new string value. For the name of this new value, type in the name that you would like to call your new icon set. Double-click on the new value to set the actual data, which is the name of the folder you created in step 3.

Creating a new locale (translation) for TortoiseCVS

Overview

Internally, TortoiseCVS uses the GNU gettext library for translating messages. Gettext works like this:

1. All messages are present in English in the source code.
2. A tool (`xgettext`) automatically extracts the messages into a specially formatted text file (a `.po` file), which can then be translated.
3. The `.po` file is finally converted to a `.mo` file, which is used by the TortoiseCVS executable.
4. When texts are updated in the source code, or when new texts are added, another tool (msgmerge) automatically updates the various .po files. The translator will then have to review any *fuzzy* entries (see the Gettext Guide below for an explanation of this).

Gettext Documentation

This simple Gettext Guide [http://www.tortoisecvs.org/gettextguide.shtml] gives an introduction to gettext. For the full story, go to the manual [http://www.gnu.org/software/gettext/manual/].

How to start

1. Check out the po module

In the Checkout dialog, use the CVSROOT

```
:ext:your_sf_login@cvs.sourceforge.net:/cvsroot/tortoisecvs
```

and the module name `po`.

2. Determine the correct language/country code

Go to this page [http://www.gnu.org/software/gettext/manual/html_chapter/gettext_15.html#SEC221] to find the code for your language, and to this page [http://www.gnu.org/software/gettext/manual/html_chapter/gettext_16.html] to find your country code. Put the two together with an underscore between; i.e. for French as spoken in Canada you would choose `fr_CA`.

3. Create a new .po file

Go into the TortoiseCVS folder, and copy the `en_GB.po` file to a new file; in our example this would be `fr_CA.po`.

4. Translate your new file

using a text editor or one of the special tools available such as poEdit [http://www.poedit.org/].

5. **Compile the .po file**

   as described in the above references. For testing, you can just copy the file to the `locale/en_GB` folder under `Program Files/TortoiseCVS` (note that it *must* be named `TortoiseCVS.mo`).

Using PuTTY sessions

If you need different SSH parameters for different hosts, you can do so by using a feature of the PuTTY suite called *sessions*.

1. Download the PuTTY suite from the PuTTY site [http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html].

2. Run `putty.exe` and enter the required parameters, then enter an appropriate session name below the **Saved Sessions** label, and click **Save**.

3. In the TortoiseCVS Checkout Dialog, enter the session name instead of the host name.

Using Pageant to avoid having to enter the SSH password every time

When using SSH, you normally have to enter your password every time you do a CVS operation. To avoid this, while maintaining a reasonable level of security, you can use an *SSH Agent*. The SSH Agent in the PuTTY suite is known as Pageant.

1. Download the PuTTY suite from the PuTTY site [http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html].

2. Run `pageant.exe`. Right-click on its icon in the system tray and select **Add Key**, then browse to your private key file. If the key is password protected, Pageant will ask for the password.

3. To avoid having to do the above every time you boot up your computer, create a shortcut to Pageant in your **Programs** → **Startup** folder, specifying the key file as argument.

How TortoiseCVS manages line endings

TortoiseCVS looks at the line endings of the CVS/Root file. This file is created when the module is first checked out, and is usually not changed afterwards. So when you check the **UNIX line endings** checkbox in the Checkout dialog, CVS creates the Root file with UNIX line endings, and this again causes TortoiseCVS to force CVS to use UNIX line endings on all subsequent CVS operations.

See also Sandbox DOS/UNIX preference setting [46].

Using other CVS clients

You can use other CVS front ends, such as WinCVS, Eclipse, etc. in the same sandbox as TortoiseCVS. Bear in mind that different CVS front ends may have different limitations; refer to the documentation for the particular front end if in doubt.

You can of course also use the command line client directly. We recommend that you do *not* use the CVS client included with Cygwin (see the TortoiseCVS FAQ). The TortoiseCVS installation includes an appropriate version of the CVSNT command line client; you just need to add the TortoiseCVS installation folder to your `PATH`.

Using SSH from the command line needs slightly more: If you want the command line client to use the user-friendly TortoisePlink SSH client, you must set an additional environment variable, `CVS_EXT`. The variable should be set to `tortoiseplink.exe -l "%u" "%h"`.

Chapter 8. Resources and Credits

The book *Open Source Development with CVS* is available in electronic form from the book's web site [http://cvsbook.red-bean.com/].

The reference manual for CVSNT™ is available here [http://www.cvsnt.org/manual/].

Development of TortoiseCVS has been made a lot easier thanks to a number of other projects:

- wxWidgets [http://wxwidgets.org/] is a GUI framework, which is used for all dialogs in TortoiseCVS.
- CVSNT [http://cvsnt.org/wiki] is the client that TortoiseCVS uses to communicate with the CVS server. Commercial support for CVSNT is offered by March Hare [http://www.march-hare.com/].
- Inno Setup [http://www.jrsoftware.org/isinfo.php] is used to build the installer for TortoiseCVS.
- Cygwin [http://www.cygwin.com/] provides a number of UNIX tools that make building TortoiseCVS a lot easier.
- eDE [http://www.e-novative.info/software/ede.php] is a DocBook environment used for building the TortoiseCVS documentation.
- CppUnit [http://cppunit.sourceforge.net/] is a unit test framework which is used for testing some parts of TortoiseCVS.