IBM

How to use Subversion with Eclipse

Integrating is easy

Level: Intermediate

Chris Herborth (chrish@pobox.com), Freelance Writer, Author

11 Jul 2006

From the beginning, Eclipse included tight integration with the Concurrent Versions System (CVS) in order to provide access to change-management capabilities. Now, many projects -- notably those run by the Apache Software Foundation -- are using a different change-management system: Subversion. Find out how to use Eclipse for projects that use a Subversion repository.

More dW content related to: **eclipse svn team**

A stock Eclipse installation has integrated support for CVS, a popular open source change-management system. The abilities of CVS, and its limitations, are well known, but many groups have been investigating other version-control systems to provide better scaling, better support for merging changes and branching versions, and better support for binary file formats.

Subversion (SVN) is a popular replacement for CVS, offering improved performance (courtesy of intelligent local caching and a database back end), easy and fast branching, and an answer to every one of the shortcomings that people often run into while using CVS.

Read on to see how to add Subversion support to Eclipse and how to perform basic version-control activities from the IDE.

Before you start

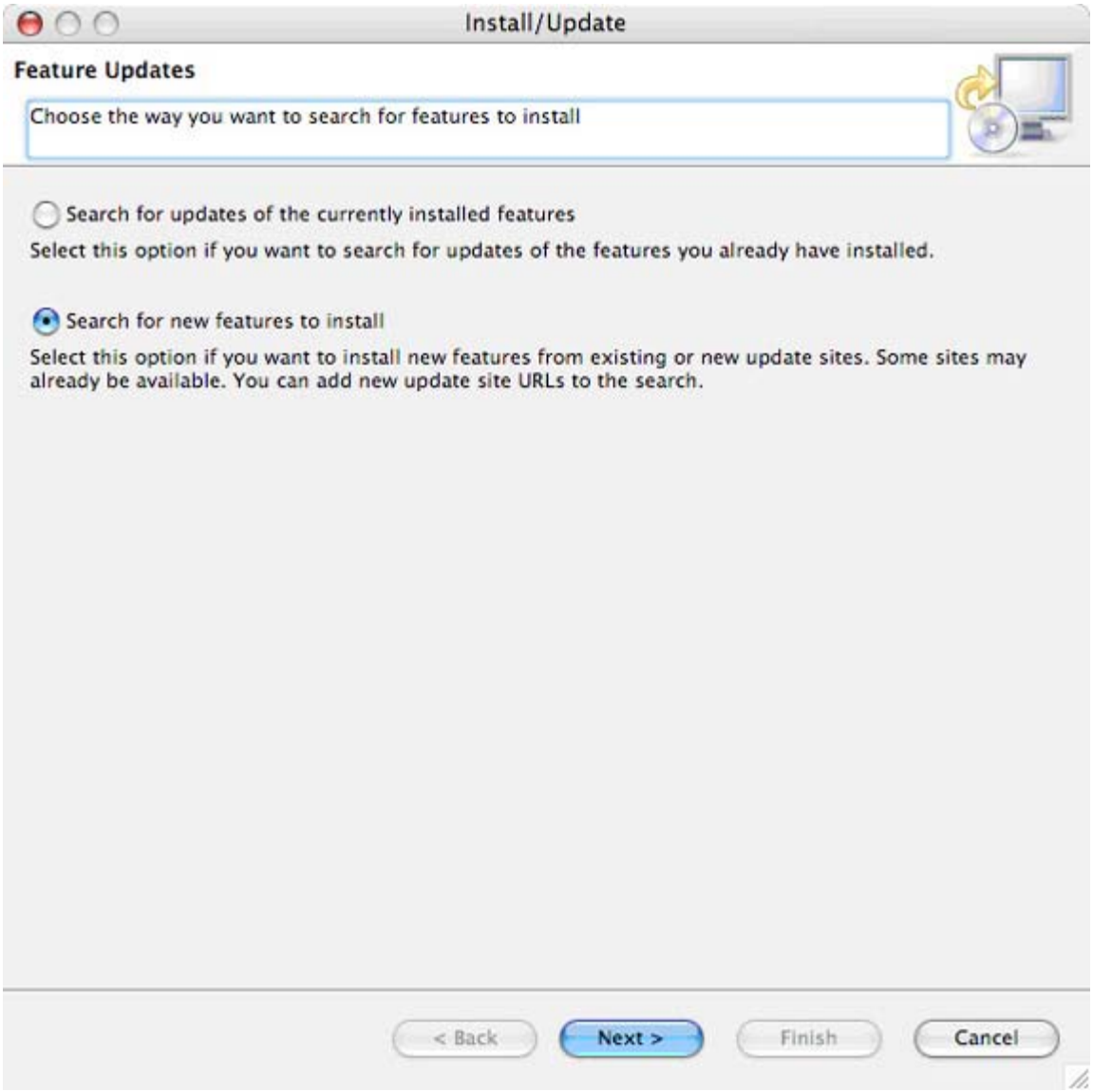
You're going to need to download and install Eclipse (see Resources) to follow along. Downloading the Eclipse SDK package for your platform will give you the base Eclipse IDE (referred to as the Eclipse Platform), as well as the Java™ Development Kit. If you plan on working with C/C++ (as I tend to), visit the C Development Tooling (CDT) Web site and install the CDT using the update manager (using the update manager is described in the next section).

You'll also need access to a Subversion repository. If you need to set one up, you can find excellent documentation at the Subversion Web site (see Resources). For demonstration purposes, I'll show you how to check out the Subclipse project and work with projects in a repository on my LAN.

Adding Subclipse to Eclipse

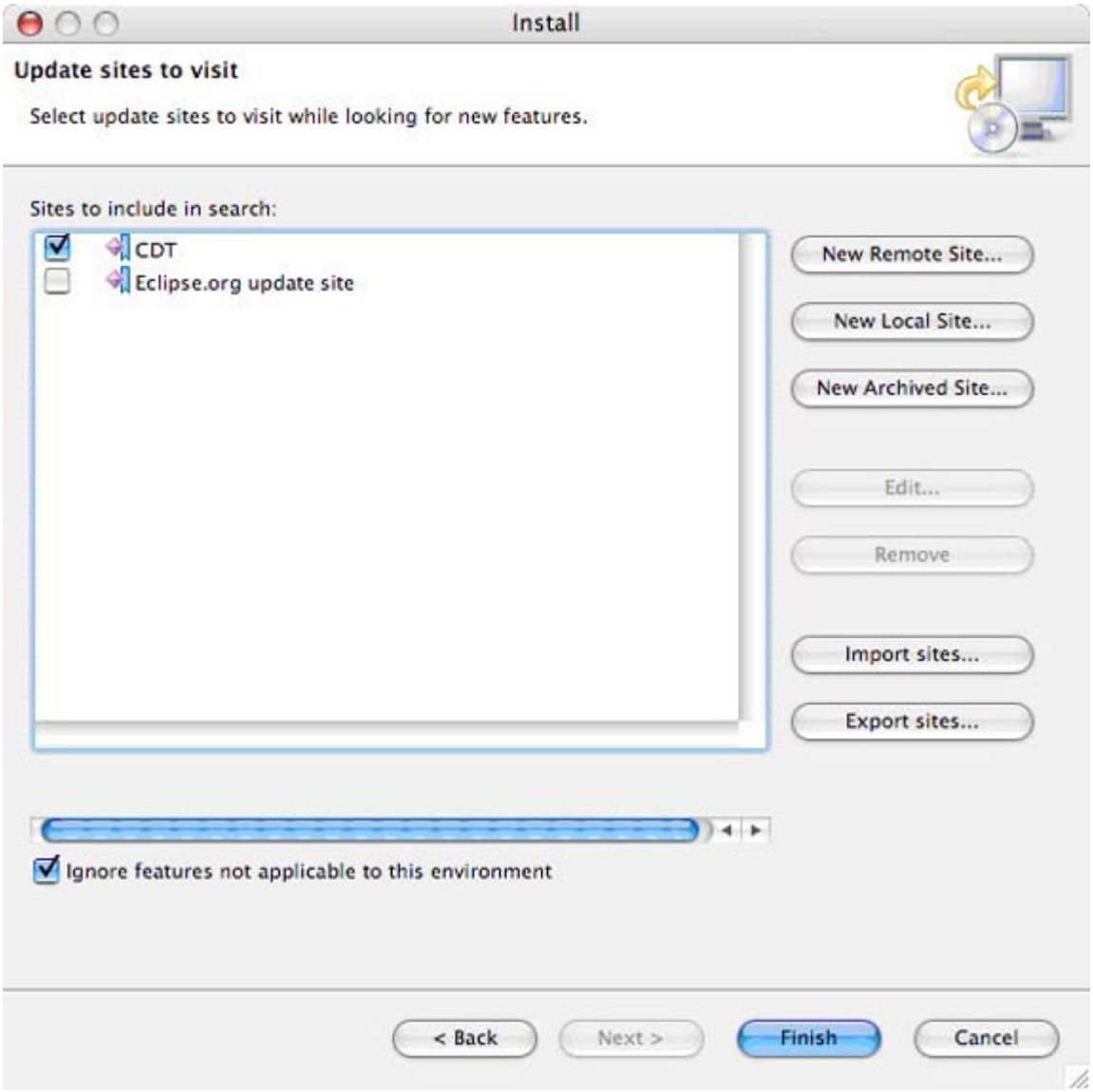
Subclipse is a project to add Subversion support to the Eclipse IDE. We'll use Eclipse's update manager to add Subclipse to our Eclipse IDE. From the Help menu in Eclipse, choose **Software Updates > Find and Install** to open the update manager.

Figure 1. The Eclipse update manager



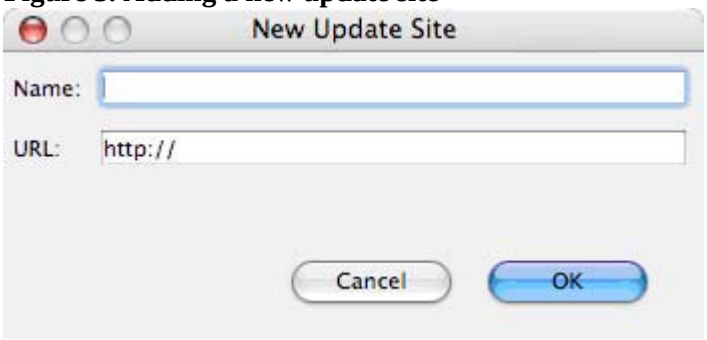
In addition to using this to look for software updates, we can use the update manager to find and install new features, such as Subclipse. Be sure that **Search for new features to install** is selected, then click **Next** to continue. Eclipse displays the next update manager panel.

Figure 2. Update manager sites



Since we're after a specific feature, un-check the existing sites, then click **New Remote Site** to display the New Update Site dialog (see Figure 3). We'll use this to add the Subclipse update site to the list.

Figure 3. Adding a new update site

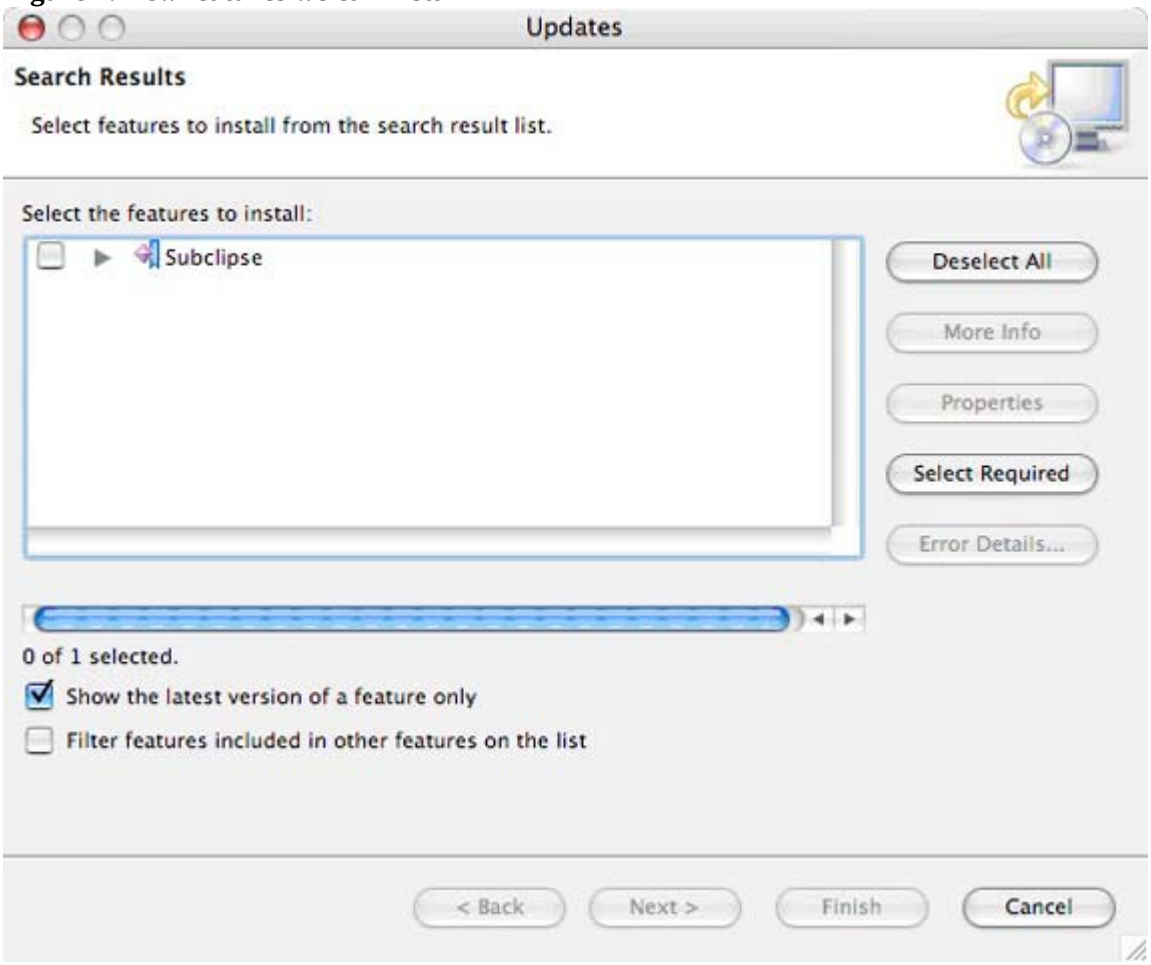


Enter whatever you want for the Name (`Subclipse` is a good choice) and enter the following for the URL: `http://subclipse.tigris.org/update_1.0.x` (the current Subclipse update site). Click **OK** to add the Subclipse update site to the list in the update manager.

Click **Finish** in the update manager window to begin searching for new features. In this case, the new feature we're

after is Subclipse. After a few moments, the update manager's search is complete, and it displays the search results.

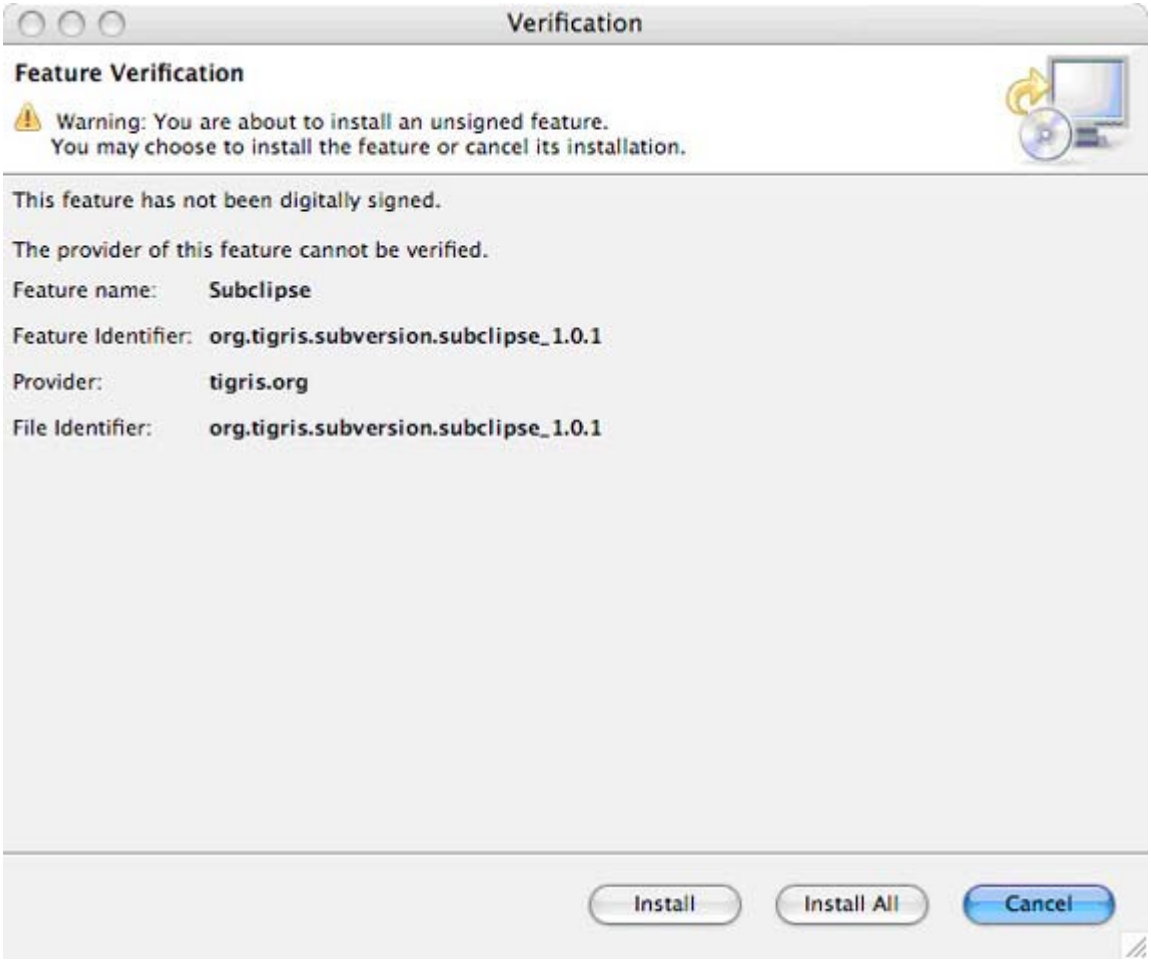
Figure 4. New features we can install



Check Subclipse (you can click the disclosure triangle to see what exactly is included in this feature), then click **Next** to view the feature's license terms. Accept the terms, then click **Next** to review the features you've chosen to install. Click **Finish** to download and install Subclipse.

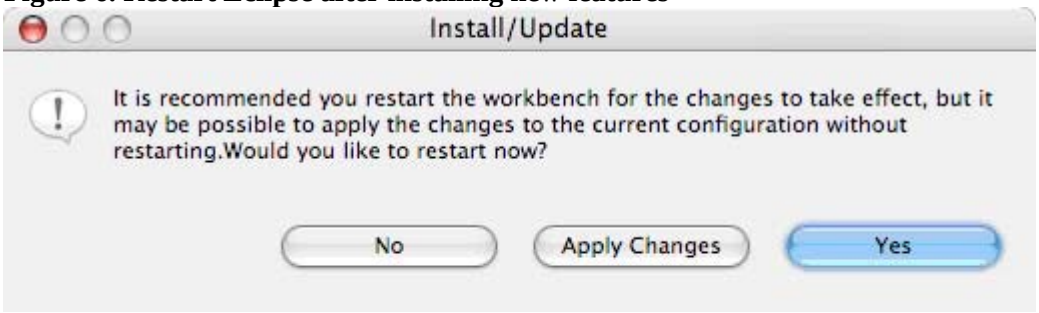
The update manager downloads the Subversion components. Before installing anything, Eclipse will warn you that the features aren't digitally signed (Figure 5). This is your last chance to cancel the installation. Click **Install All** to continue the installation.

Figure 5. Subclipse isn't digitally signed



Once Subversion has been installed, Eclipse warns you that you might need to restart the IDE to activate the new features (see Figure 6). Restart Eclipse, just in case.

Figure 6. Restart Eclipse after installing new features



When Eclipse comes back up, Subclipse is installed and ready to go.

If you're running Eclipse on Mac OS X or Linux®, you may need to install the JavaHL library, which is described in the Troubleshooting section of the Subclipse FAQ (see Resources). Do this before you continue trying to use Subclipse.

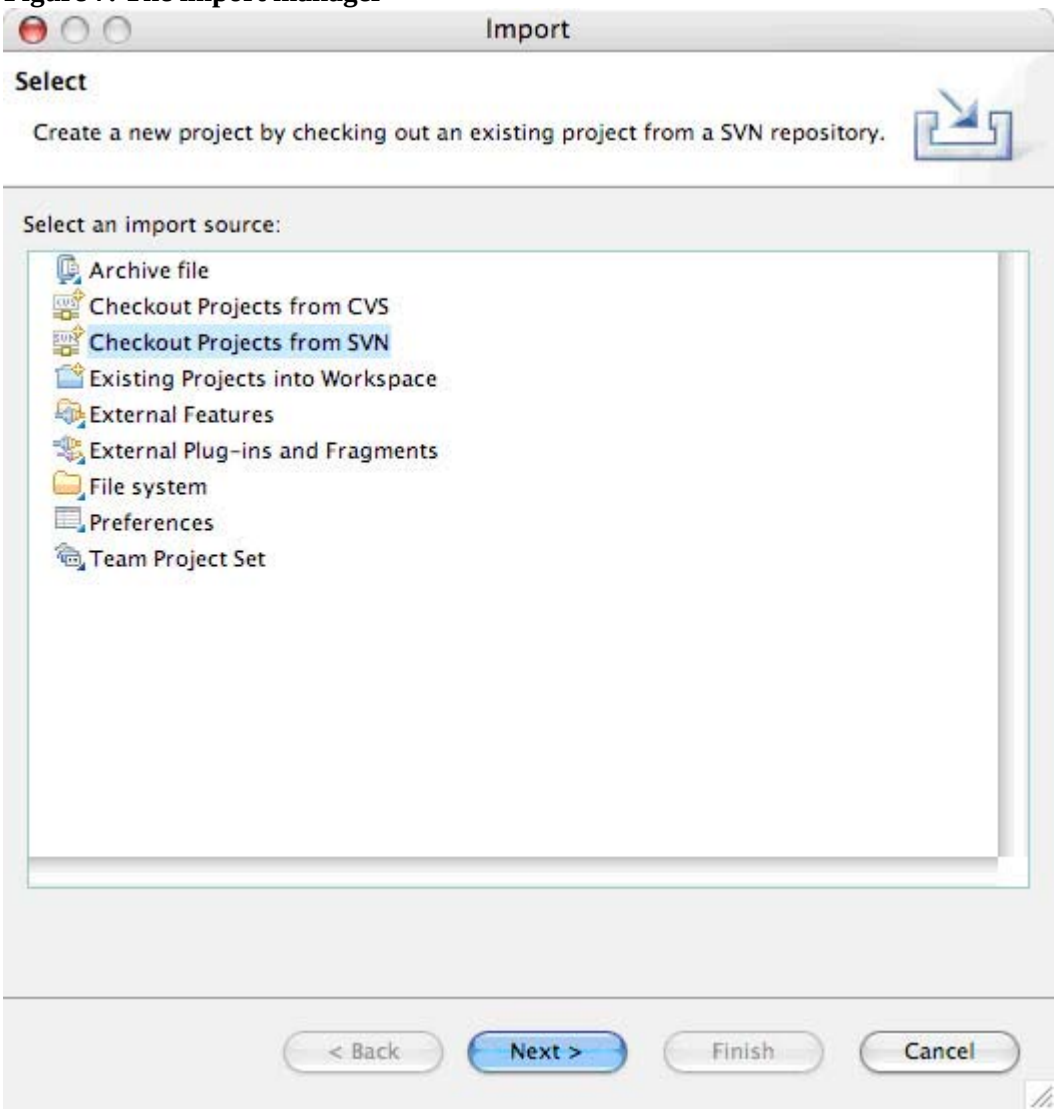
A quick test

It's always nice to test a new feature once you've finished the installation; we'll try checking out a copy of Subclipse from their Subversion repository to make sure it's been properly installed.

From Eclipse's File menu, choose **Import** to display the import manager (see Figure 7). Choose **Checkout Projects**

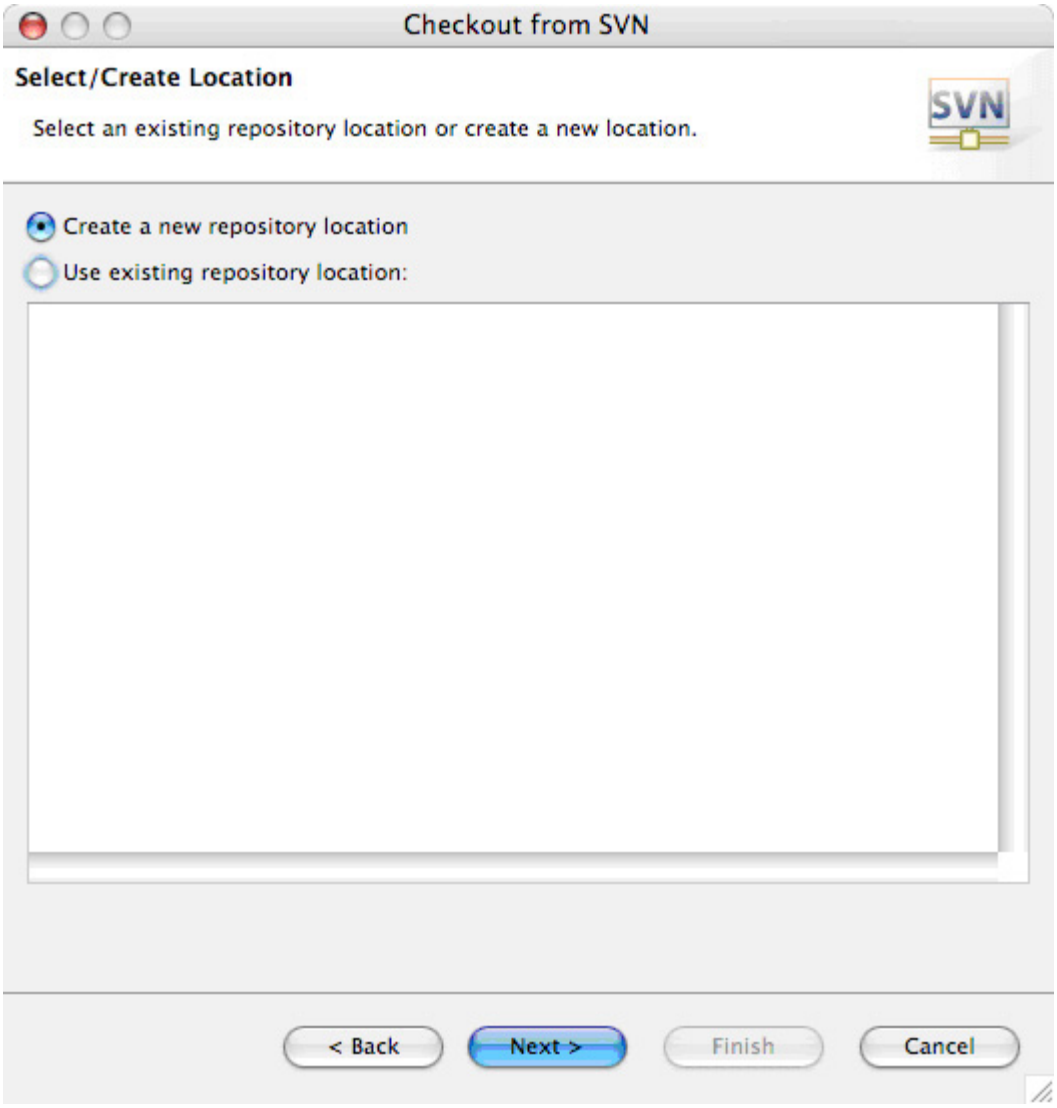
from SVN, then click **Next**.

Figure 7. The import manager



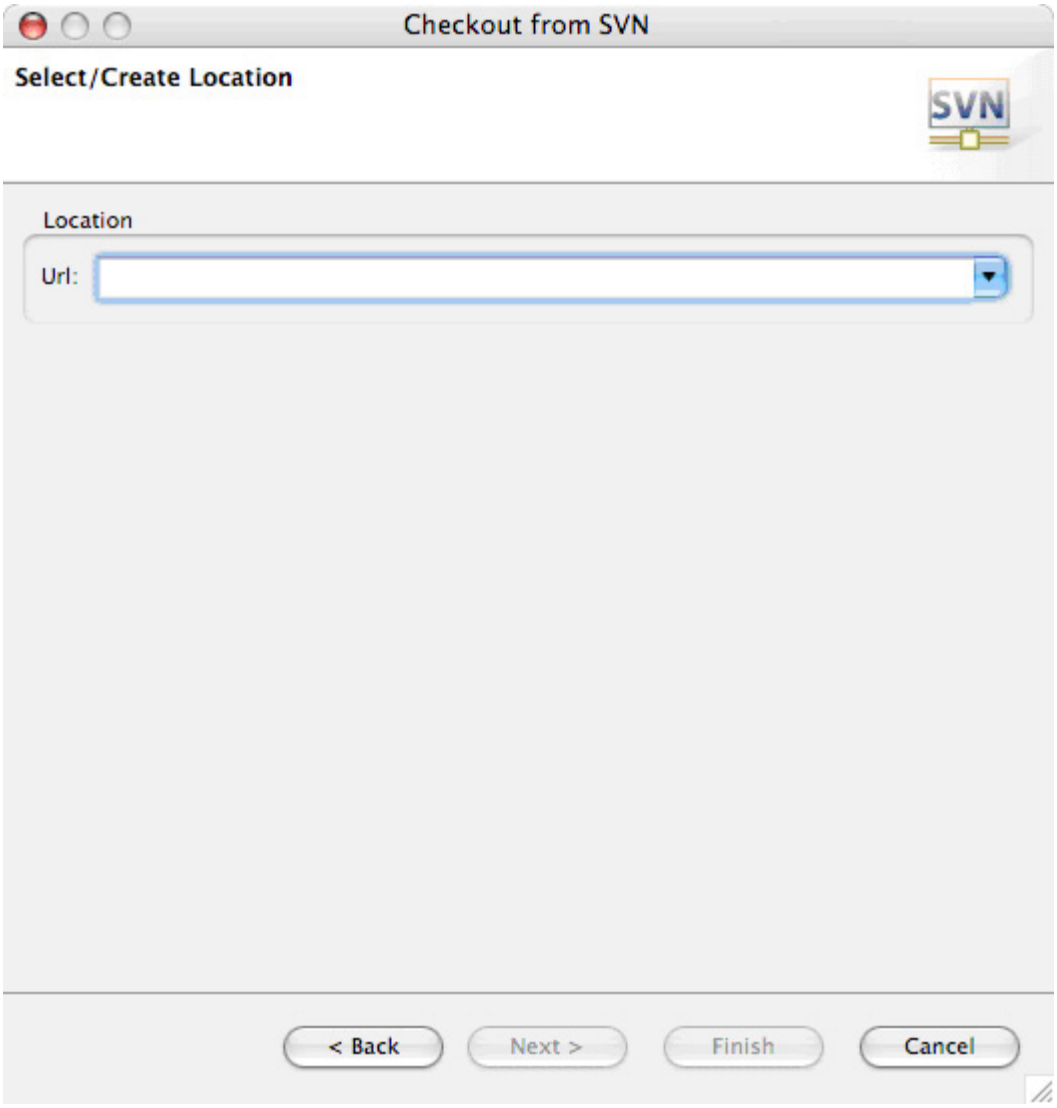
On the Select/Create Location panel (see Figure 8), we need to create a new location (since we don't have any configured yet), so click **Next** to continue. If the **Next** button is disabled, switch to the **Use existing repository location** option, then back to **Create a new repository location** to enable the **Next** button.

Figure 8. Creating a new repository location



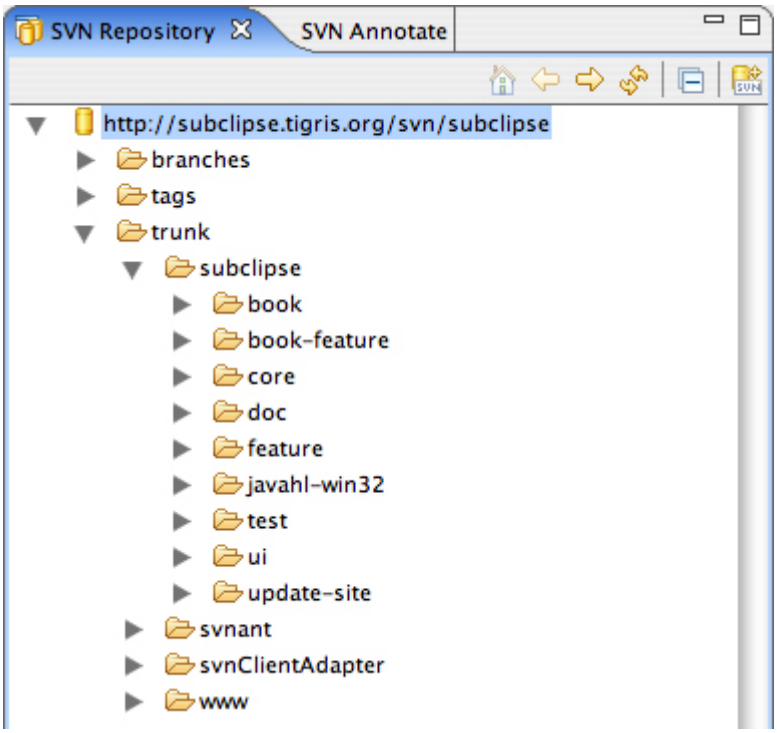
In the next section (see Figure 9), add the repository URL (`http://subclipse.tigris.org/svn/subclipse/`) to the Url field, then click **Next**. After a moment, Eclipse prompts you for user ID and password. If you don't have an account on the Subclipse site, enter `guest` for the user ID and a space for the password, check the **Save Password** box, and click **OK**.

Figure 9. Add the repository URL



Eclipse displays the folders in the Subclipse repository (see Figure 10). Expand the trunk and choose the subclipse folder, then click **Finish** to check out your own copy of the Subclipse project's source code. Since you have no idea what this is, choose **Simple > Project** when the New Project wizard prompts you.

Figure 10. Subclipse repository



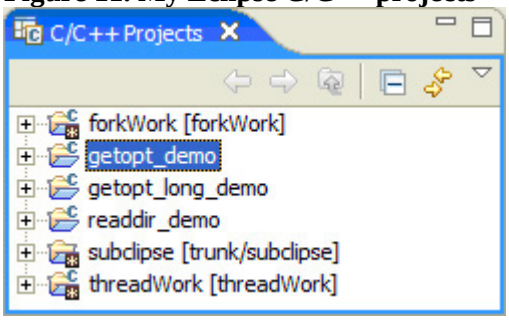
Basic Subversion operations

At this point, we've installed Subclipse successfully, which added support for Subversion servers to our Eclipse setup, and we've tested Subclipse by downloading the current Subclipse source code from the repository. Now we should look at doing something with our own code and our own Subversion repository.

Before I show you how things work with Subversion, I'll tell you a little bit about my repository. It's hosted on a machine called dogma on port 8000, and I've created a new developerworks repository for code associated with my developerWorks articles. I'm going to put my projects directly in the root of the repository. Other repositories often have folders named trunk, tags, and branches off the root, for development versions, tags, and branches, but I don't expect to need to worry about tagging or branching the developerWorks article code.

I've added two projects, forkWork and threadWork, from my first developerWorks article. My Eclipse workspace (see Figure 11) also contains three other projects from developerWorks articles (getopt_demo, getopt_long_demo, and readdir_demo).

Figure 11. My Eclipse C/C++ projects

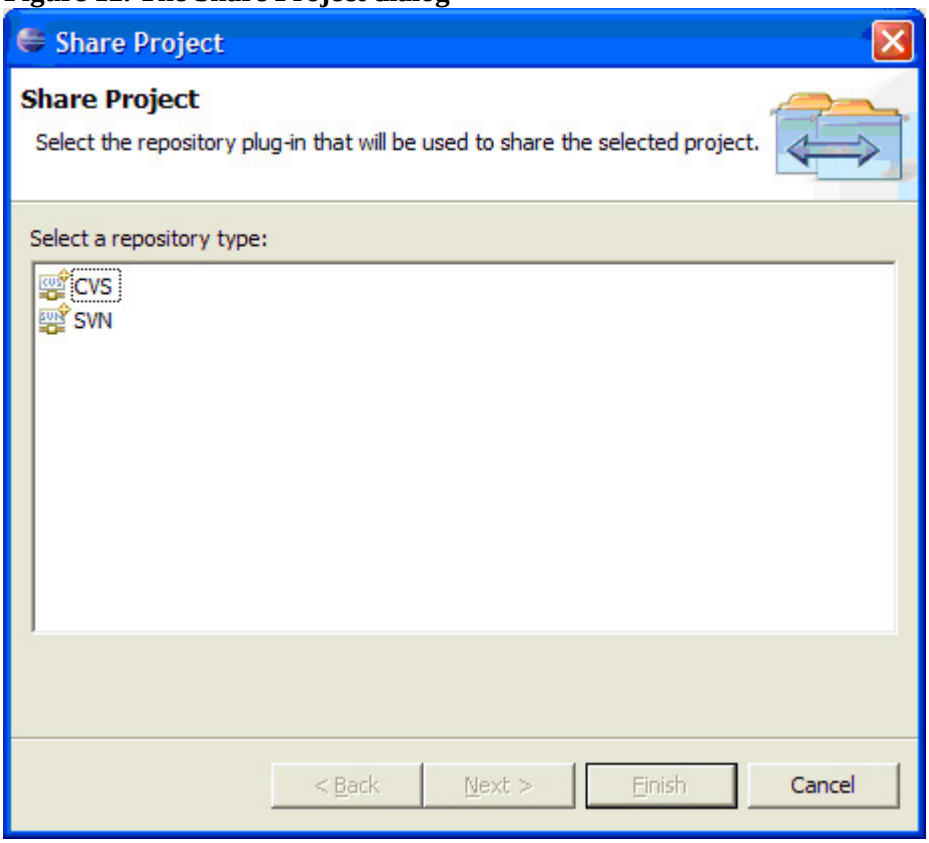


Now we're ready to get to work.

Adding a project to the repository

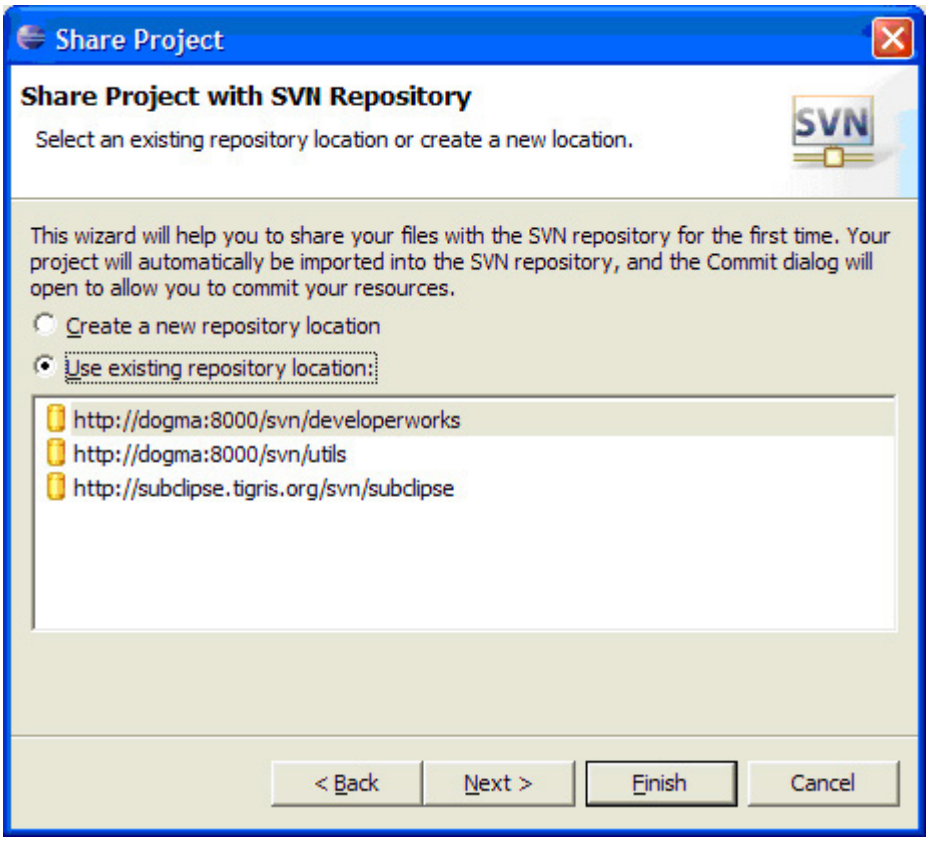
To add a new project to your Subversion repository, right-click the project (in any of Eclipse's project views or the Navigator view) and choose **Team > Share Project** from the menu. Eclipse displays the Share Project dialog.

Figure 12. The Share Project dialog



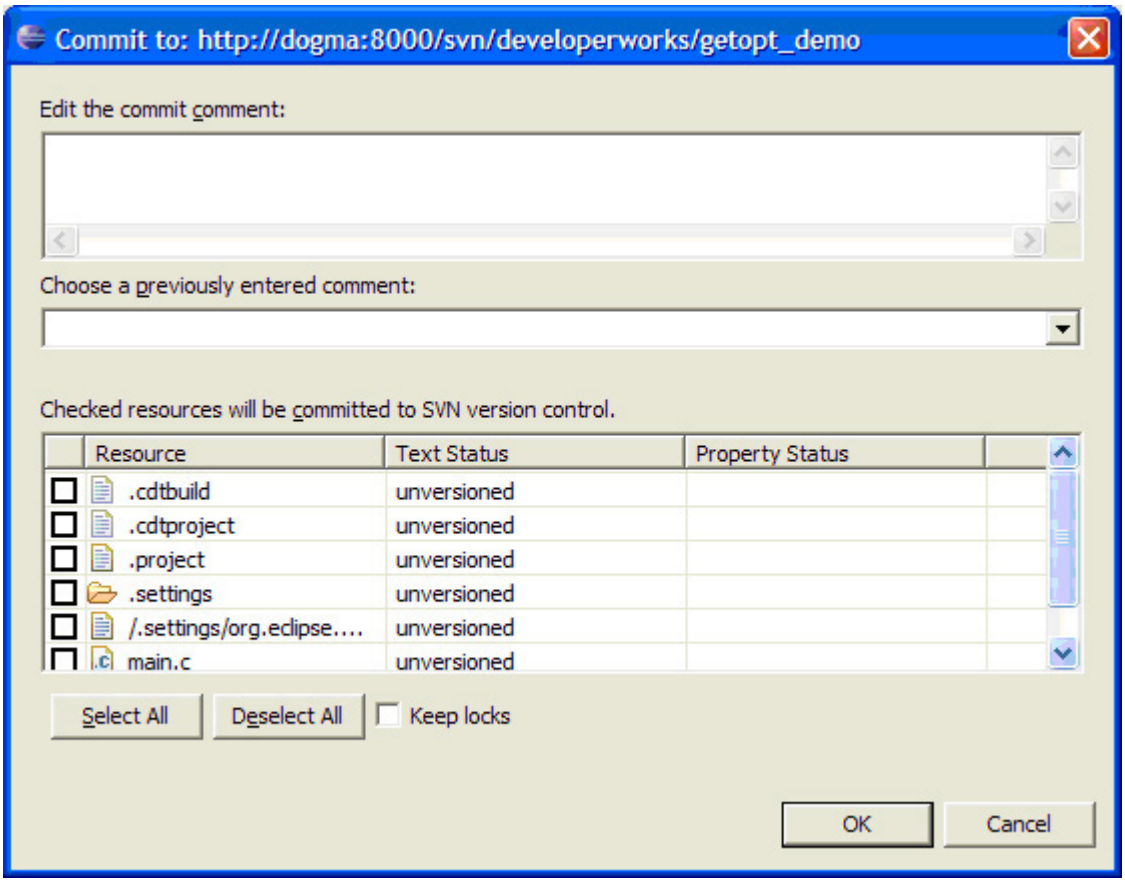
Select SVN from the list of repositories currently supported by your Eclipse, then click **Next**. The next dialog (see Figure 13) lets you choose an existing repository location, or you can create a new one.

Figure 13. Selecting a repository location



If your repository is already listed (as you can see, I've added mine), select it, and click **Finish**. If your repository isn't listed, add it (see A quick test for instructions) and continue. Eclipse creates a new directory in the repository with the same name as your project, and displays a list of all files and folders in the project.

Figure 14. Adding a project's contents



Enter a suitable comment describing this project in the top field, then click **Select All** to check all of the files from the project. Click **OK** to check in your project and transmit its current state to the Subversion repository.

Subversion's commands and output are displayed in the Console view, usually found at the bottom of your Eclipse window, if you want to see exactly what Subclipse did with your project.

Updating a project

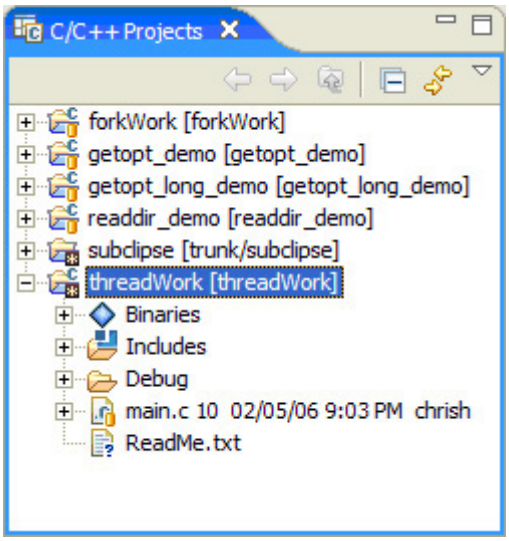
One of the key features of a version-control system is the ability for other developers to continue development and commit their changes whenever they're ready. To download these changes and integrate them with your local copies, you need to update the project.

Right-click on the project you want to update, then choose **Team > Update** from the menu. Eclipse retrieves any changes from the repository and attempts to merge them with your local copy.

Adding a file or directory

If you add a file to your project (see Figure 15), it's not automatically part of version control -- you need to specifically add it to the repository. In the screenshot, you can see that I've added a ReadMe.txt file to the threadWork project.

Figure 15. Adding a new file



Right-click the new file, then choose **Team > Add to Version Control**. That's it! The next time you commit your changes in this project to the repository, the new file will also be checked in.

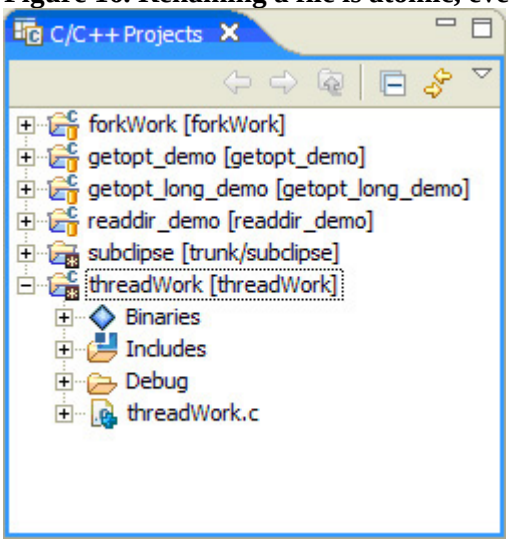
Deleting a file or directory

If you've added a file to the repository that's no longer relevant to your project, you can easily delete it. Right-click the file, then choose **Delete**. No need for the Team menu, Subclipse flags the file for deletion automatically and removes it from your project. The next time you commit your changes to the repository, the file is deleted.

Renaming a file or directory

To rename a file or directory under Subclipse's control, right-click it, then choose **Rename**. Type the item's new name in the entry field and click **Enter**. The file is renamed in the project, and the rename operation (an Add for the new name, and a Delete for the old one) is queued for your next commit. In Figure 16 you can see the threadWork project after I've renamed main.c to threadWork.c, but before I've committed my change. Note the little blue plus sign Subclipse has added to the "new" file to indicate that it's scheduled for addition in the next commit.

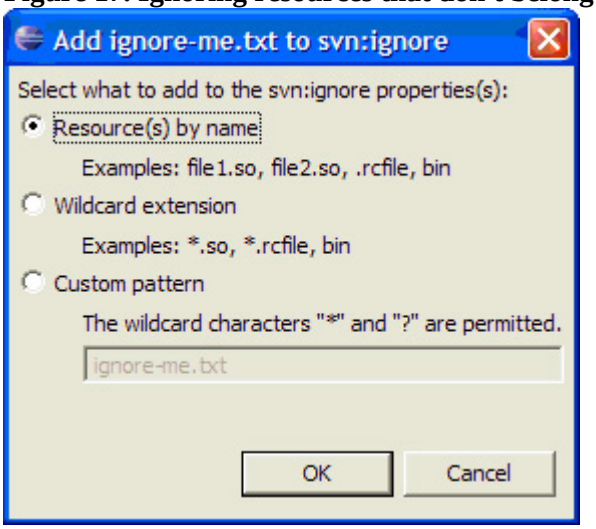
Figure 16. Renaming a file is atomic, even though it's an add and a delete



Ignoring files

If your project generates files, or otherwise includes files that you don't want to check in to the Subversion repository, you can tell Subclipse to ignore them. Right-click the file or directory you want to exclude from version control, then choose **Team > Add to svn:ignore** to display the Add to svn:ignore dialog.

Figure 17. Ignoring resources that don't belong in version control



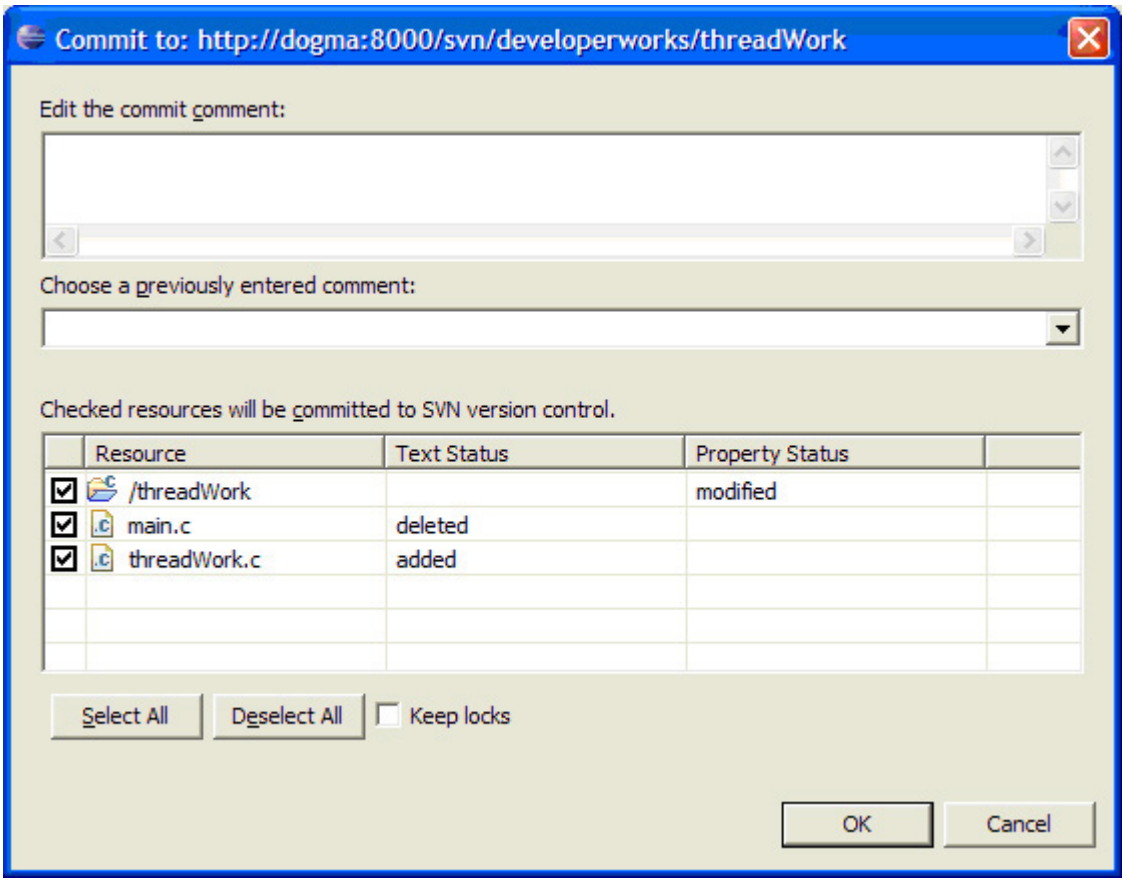
Click **OK** to add this specific file to the svn:ignore property for the project's directory. Choose **Wildcard extension** to ignore all files with the current file's extension, or choose **Custom pattern** to add your own wild card to the ignore list. These changes to the ignore list will be added to the repository the next time you commit your changes.

Committing your changes

Once you're happy with your changes to the project, you've made sure your code compiles, and you've tested your changes, you should commit them to the Subversion repository. This acts as a backup in case your workstation self-destructs, and it lets other developers update their local copies to include your changes.

Be sure to update your project (see "Updating a project") before attempting to commit your changes. Right-click the project and choose **Team > Commit** from the menu. Eclipse displays the Commit dialog (see Figure 18), which summarizes your changes.

Figure 18. Committing your changes to the repository



If you look carefully, you'll see a property change to the project's directory (I've added to the `svn:ignore` property to keep certain files out of the repository) and that main.c was deleted while threadWork.c was added. That pair of changes actually represents one operation (a file rename).

At this point, you can deselect resources if you want to keep them out of the repository. This might be helpful if you're partially finished work in one file, and don't want to check in an incomplete change. Enter a suitable comment in the top text field, then click **OK** to check in your changes to the repository.

Summary

The Subclipse project integrates support for the Subversion version-control system with Eclipse's excellent team project management features, which only support CVS servers out of the box. Using Eclipse's update manager, it's easy to add Subclipse to your Eclipse installation, which lets you use this superior (in my opinion, at least) version-control system directly from Eclipse.

While adding projects to a repository -- and managing your project's resources once it's there -- can be daunting for folks unfamiliar with Subversion, the procedures for common operations are straightforward. This article walked you through the everyday operations to help familiarize you with Subclipse.

Resources

Learn

- Start with "What is Eclipse, and how do I use it?"

- Get started now with Eclipse from developerWorks contains links to the latest version of Eclipse, information on IBM's involvement with Eclipse, and a guide to some of the most interesting Eclipse projects.
- Read "C/C++ development with the Eclipse Platform" to learn how to use the C/C++ Development Toolkit (CDT), which is the best C/C++ toolkit available for Eclipse.
- Visit Subversion for downloads and documentation.
- Check out the Subclipse FAQ.
- Visit the CDT project page at Eclipse.org.
- Learn more about the Eclipse Foundation and its many projects.
- For an excellent introduction to the Eclipse platform, see "Getting started with the Eclipse Platform."
- Expand your Eclipse skills by visiting IBM developerWorks' Eclipse project resources.
- Browse all of the Eclipse content on developerWorks.
- Visit the developerWorks Open source zone for extensive how-to information, tools, and project updates to help you develop with open source technologies and use them with IBM's products.
- Stay current with developerWorks technical events and webcasts.

Get products and technologies

- See the latest Eclipse technology downloads at IBM alphaWorks.
- Innovate your next open source development project with IBM trial software, available for download or on DVD.

Discuss

- The Eclipse newsgroups home has lots of resources for people interested in using and extending Eclipse.
- Get involved in the developerWorks community by participating in developerWorks blogs.

About the author

Chris Herborth is an award-winning senior technical writer with more than 10 years of experience writing about operating systems and programming. When he's not playing with his son, Alex, or hanging out with his wife, Lynette, he spends his time designing, writing, and researching (that is, playing) video games.