Version 2
July 2007

Daily use of SubclipseSVN

A user guide to the commonly used functions of the Subclipse client for source code management on the CropForge collaborative software development site.

This work was funded by the Generation Challenge Programme
http://www.generationcp.org

CONTENTS

1. Introduction

Subversion (or **SVN** in reference to its command line name) is an open source application used for revision control. It is a modern replacement for the Concurrent Versioning System (**CVS**) which enables developers to work concurrently on their source code projects. The source code is usually in a directory tree that contains the files that make up the software project.

The main reasons for using a source code version control system like SVN are:

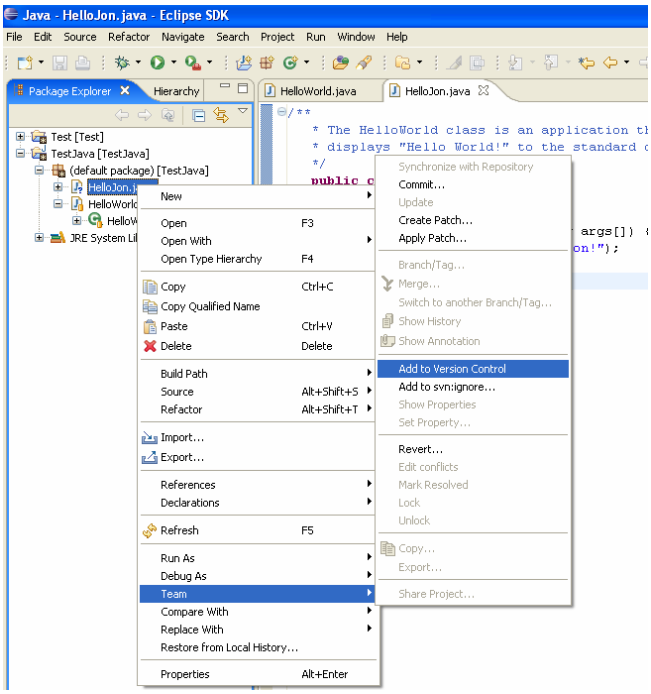
- The SVN repository serves as backup
- The SVN repository maintains a complete version history of every file
- The SVN repository facilitates multiple developers working concurrently on the same project

This manual describes the basic interaction with a CropForge SVN repository that is needed on a daily basis (e.g. modify, commit, add, update, delete, rename, revert). It also addresses some more advanced topics in source code management (e.g. tagging, branching, merging) and release management. It focuses mainly on the use of Subclipse – a SVN plugin for Eclipse, which provides a user interface to Subversion from within the Eclipse IDE.

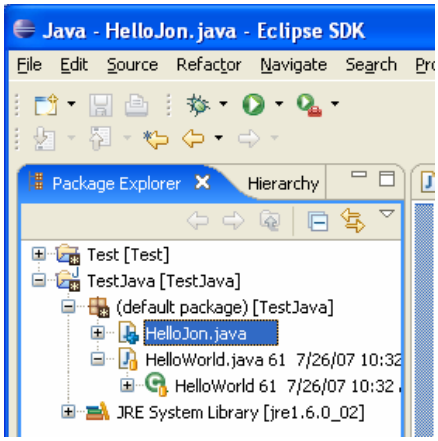
It is assumed that the Subclipse is already installed as a plugin of Eclipse in the user's client machine and he/she has an existing account on CropForge. The user must also be a developer in a project, and have either checked out an existing module from that project, or uploaded a new module to that project. These issues are covered in separate manuals CodeOnCropForge.pdf and BeginningSubclipseSVN.pdf.

2. Add File, Folder or Folder Tree to the SVN Repository

1. In the **Package Explorer,** create the file(s)/folder(s) that you would like to have added to Subversion
2. Select the file(s)/folder(s) and right-click the selected file(s)/folder(s): **Team > Add to Version Control**

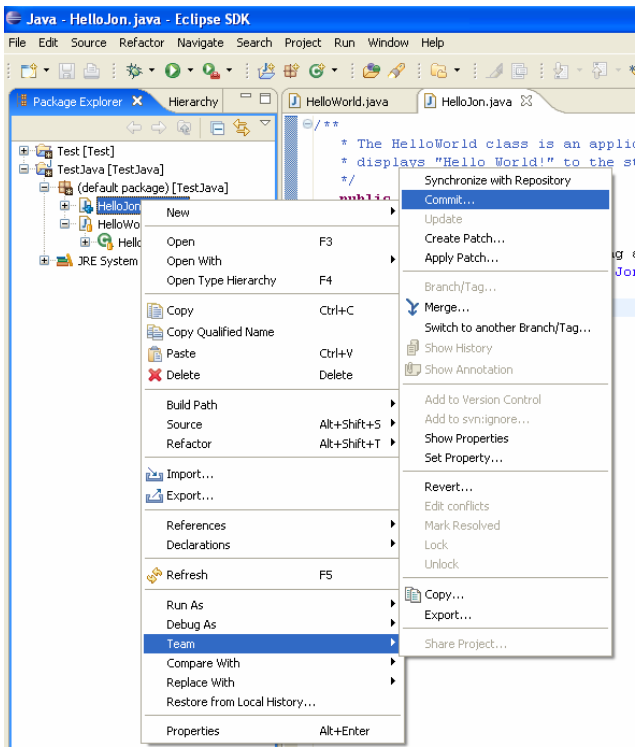


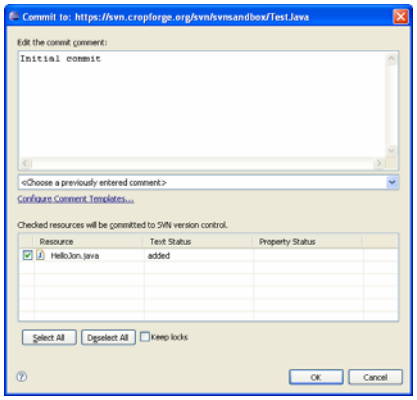
3. Your **Package Explorer** should replace the icon overlay of the file(s)/folder(s) with a **+** instead of a **?**



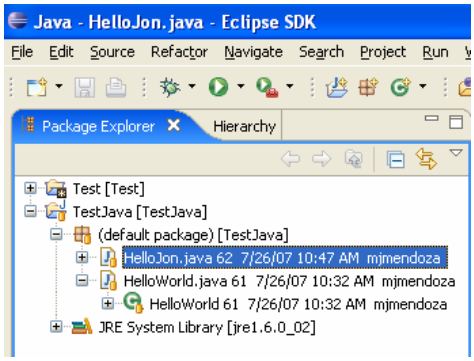
4. Now to finalize the Add, you have to do a Commit by right-clicking the project: **Team > Commit...**

5. Enter your **commit message** into the **Edit the commit comment** text box and click **Ok**





The **+** should be replaced with a silo-looking icon and you should also see revision information for your file(s). **(Subclipse does not show revision information for folders)**

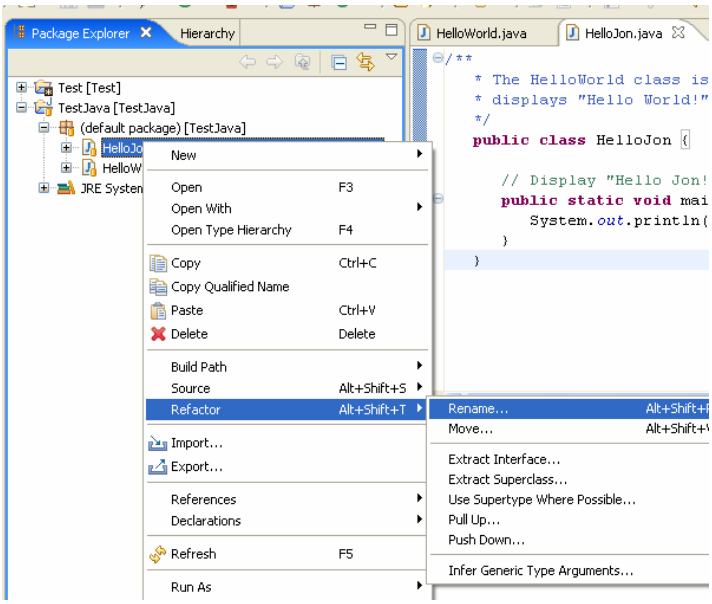


3. Modify Contents in the SVN Repository

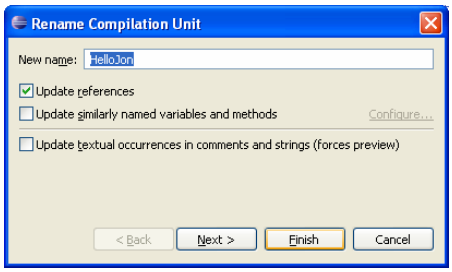
1. Make the required modifications in your working copy
2. Right click on the file that was modified: **Team > Commit...**
3. Enter a log message
4. Click **Ok**

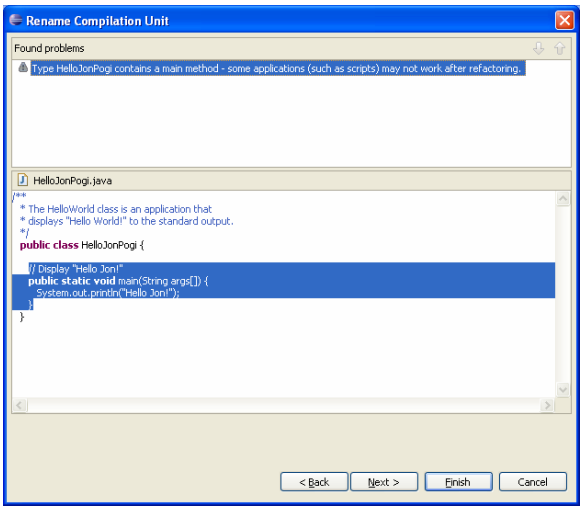
4. Rename a File or Folder

1. Right click on the file/folder that you want to rename: **Refactor > Rename**



2. A Window will appear where you can modify the file/folder name.
3. Click **Next** and resolve needed refactoring changes, if any





4. Click **Finish**
5. Navigate to the parent folder and right-click: **Team > Commit...**
6. Enter a log message, then click **Ok**

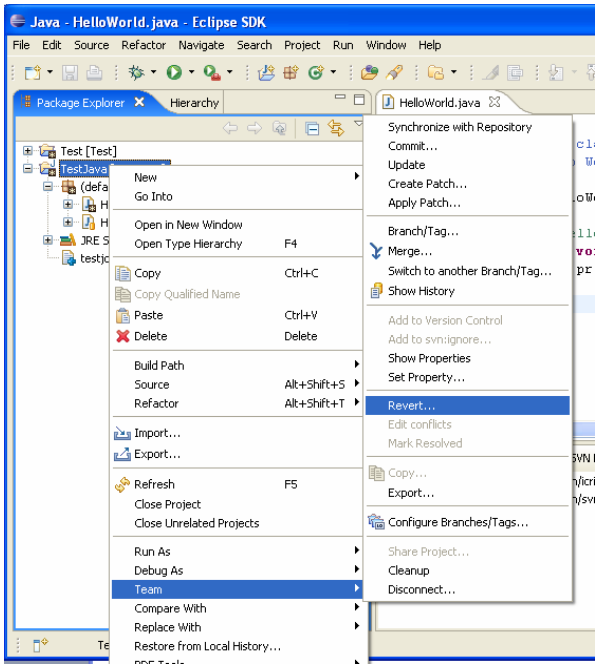
5. Delete a File or Folder

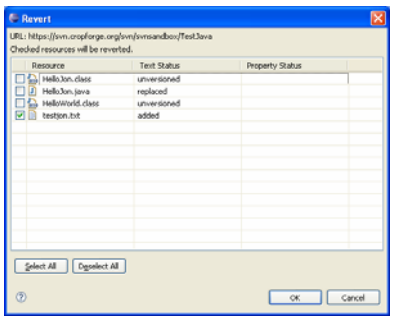
1. Right click on the file/folder that you want to delete
2. Navigate to the parent folder and right-click: **Team > Commit...**
3. Enter a log message
4. Click **Ok**

6. Revert Changes

If you wish to remove all changes made to the local copy that have been added but not committed, you can use the revert operation. This will undo all changes made to a file since the last commit.

1. Right-click the project: **Team > Revert**
2. From the **Revert** dialog, uncheck any files that you do not want to revert

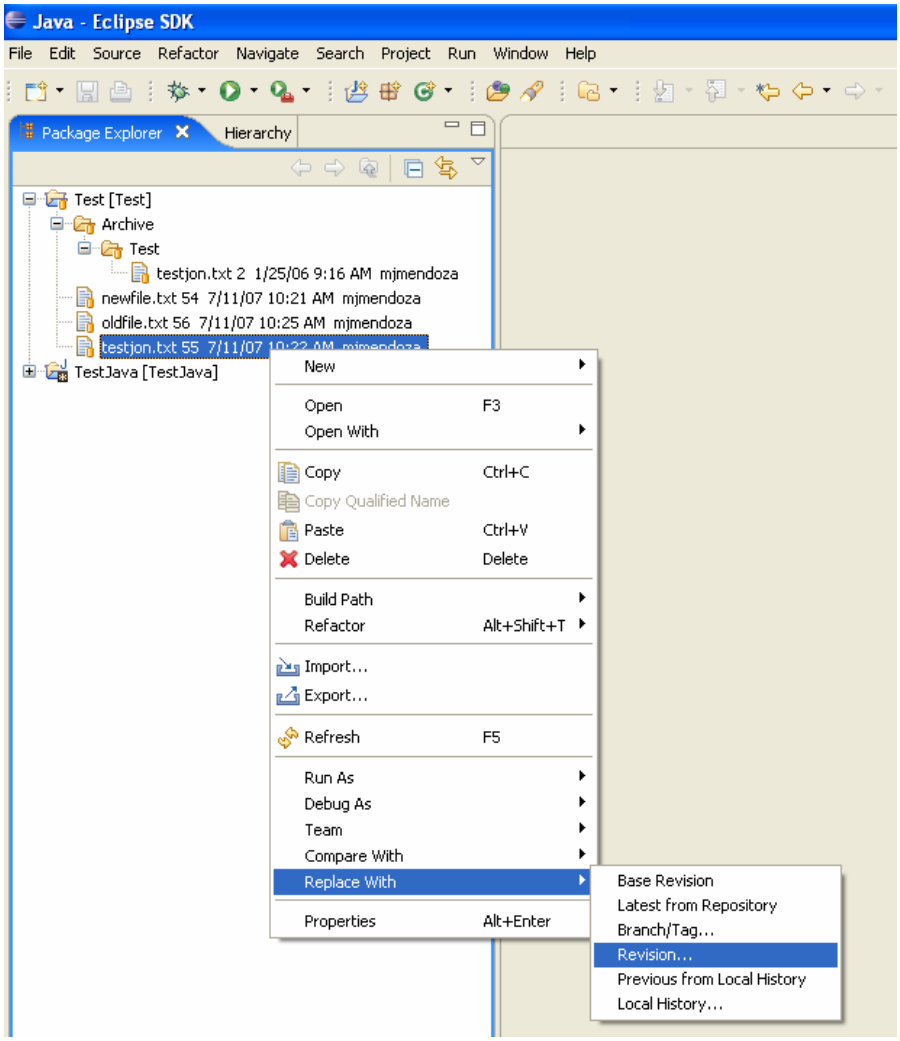




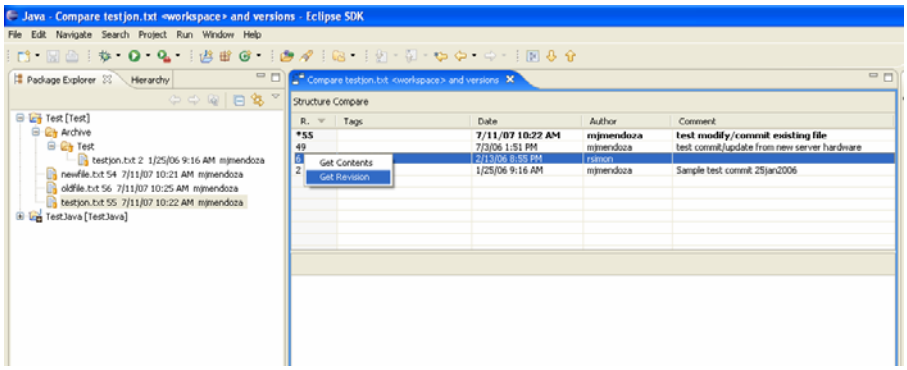
7. Get the Contents of an Earlier Revision

If you want to view the contents as in an earlier repository revision, you can do that by using the 'Update to revision' option.

1. Right-click project: **Replace With > Revision...**

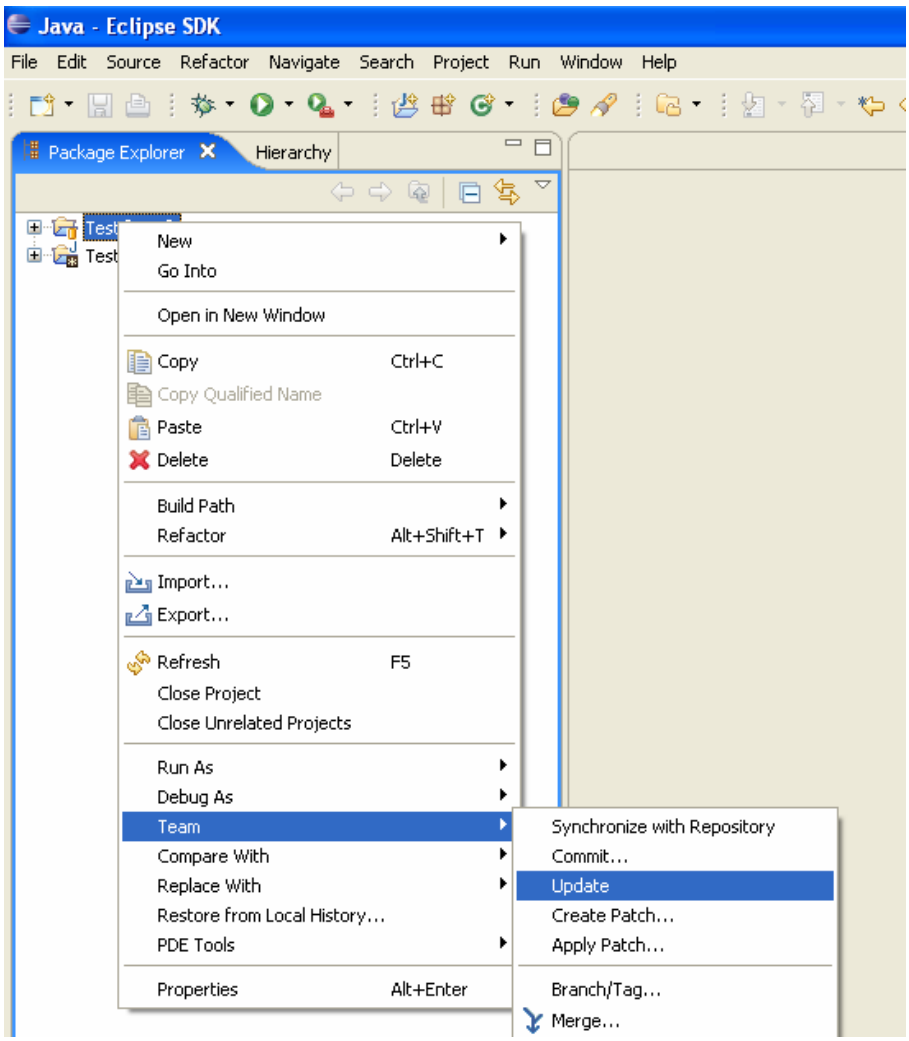


2. The **Compare** editor should display with all available revisions of this file
3. Right-click the revision you would like to retrieve: **Get Revision**



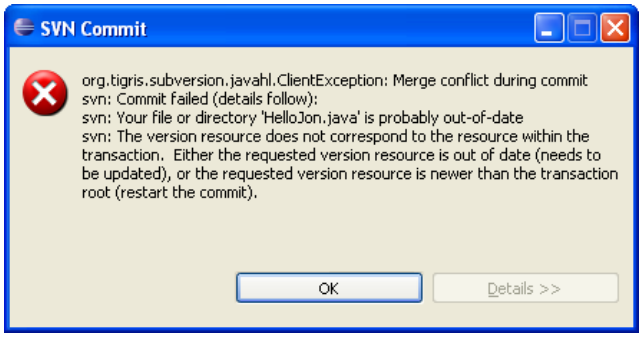
8. Update the Working Copy

The update command merges changes in repository with local copy. To update the working copy with the latest repository revision, right-click project/file/folder: **Team > Update**

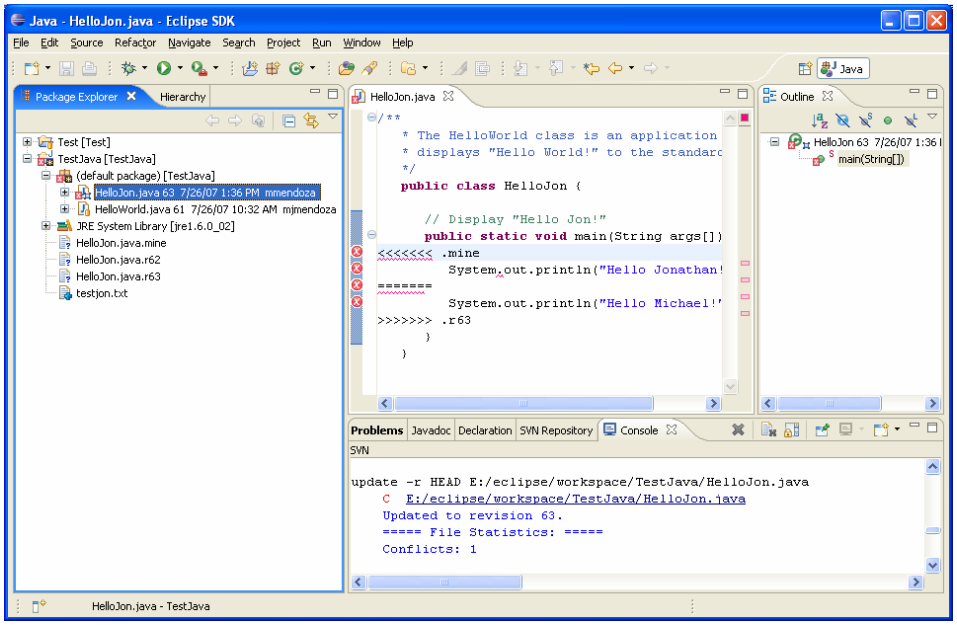


9. Conflict Resolution

During updates/merges, conflict may arise that prevent Subclipse from merging.

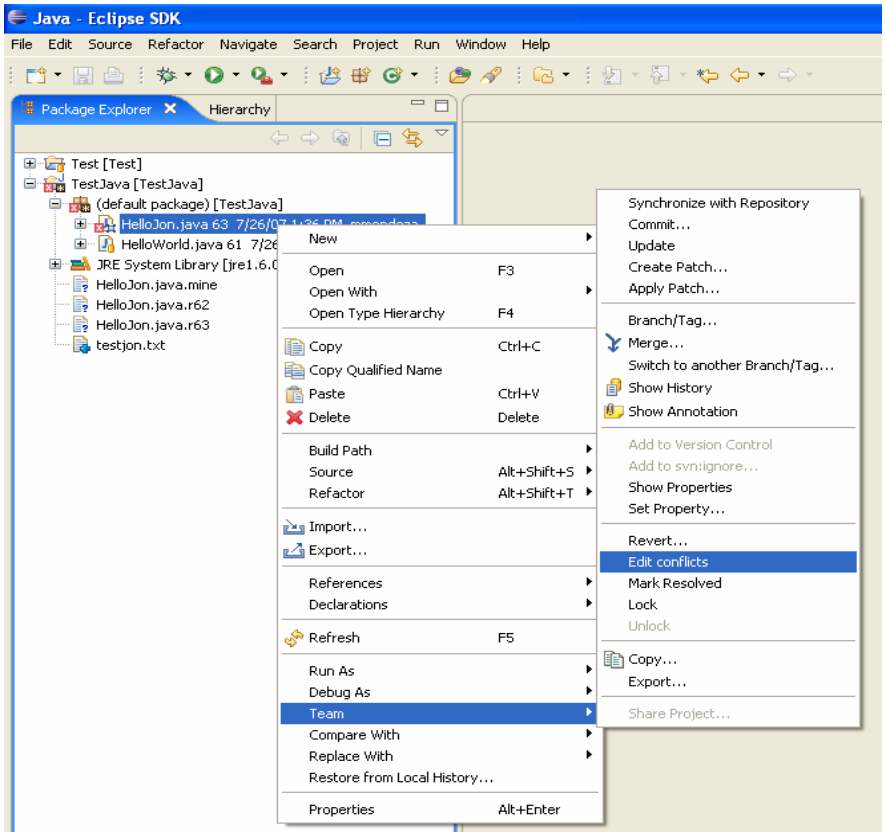


This occurs when two or more developers change the same few lines of a file. When such conflict arises, a message from the console window will appear:

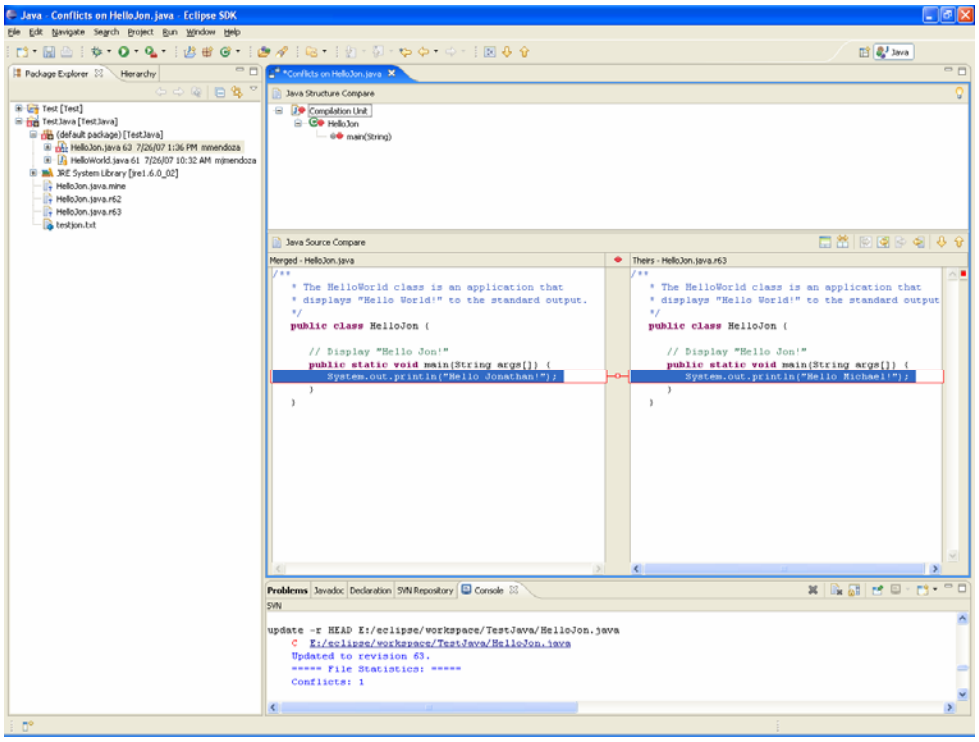


You should solve the conflict by:

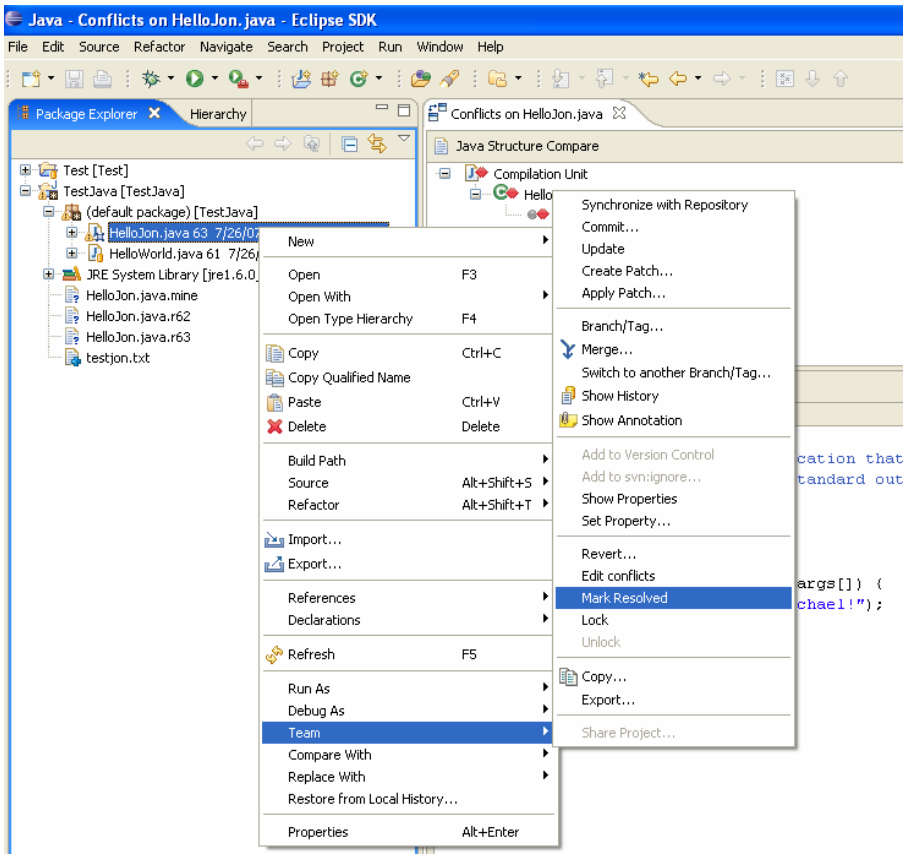
1. Right-click the conflicted file: **Team > Edit conflicts**



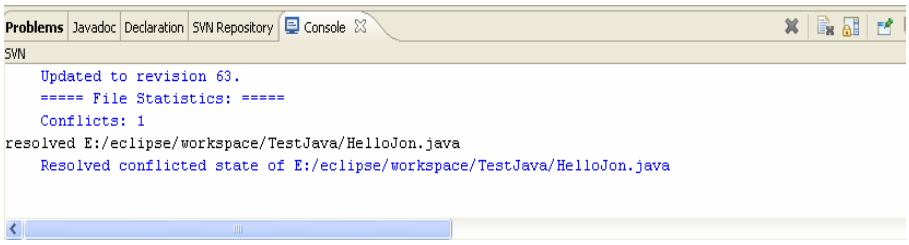
2. Fix the conflict
   The **Conflicts** editor displays two conflicting versions of the files. One has changes made by you and the other is the version of the file currently residing in the repository. Clicking on a conflict allows you to modify it.



3. After choosing the appropriate resolution for all the conflicts, right click on the file conflicted file: **Team > Mark Resolved**



4. After marking the file as resolved, the Console window should look similar to this:

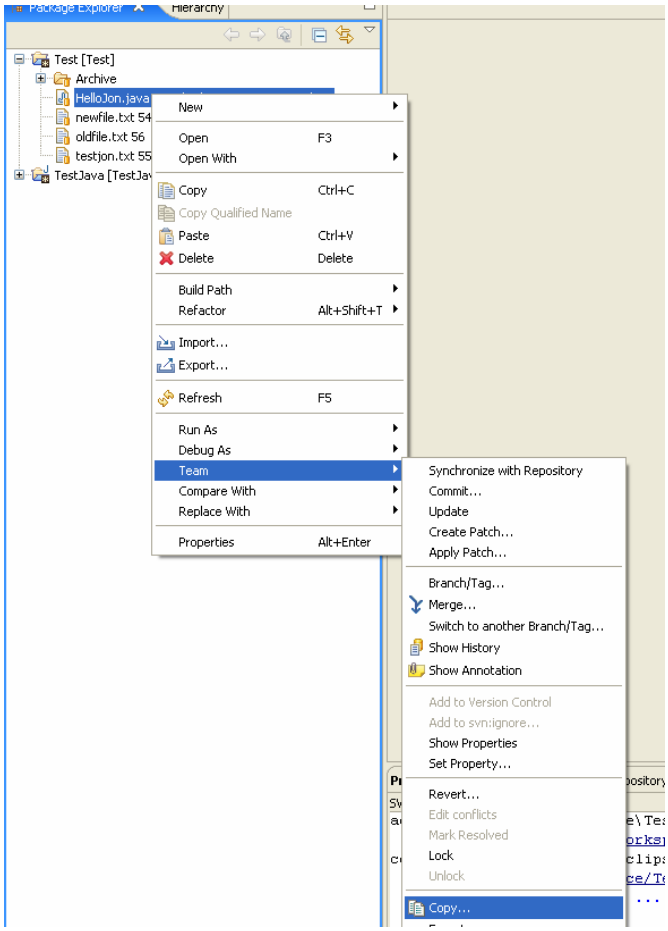


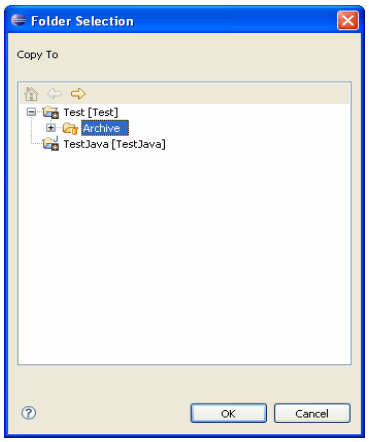
5. If the file is not yet committed, right-click the project: **Team > Commit...**
6. Enter a log message, and click **Ok**

10. Copy/Move a File or Folder

1. Right-click the file/folder you wish to copy/move: **Team > Copy...** or **Team > Move...**

2. From the **Folder Selection** dialog, navigate to the folder where you want the file/folder to be copied/moved to and click **Ok**





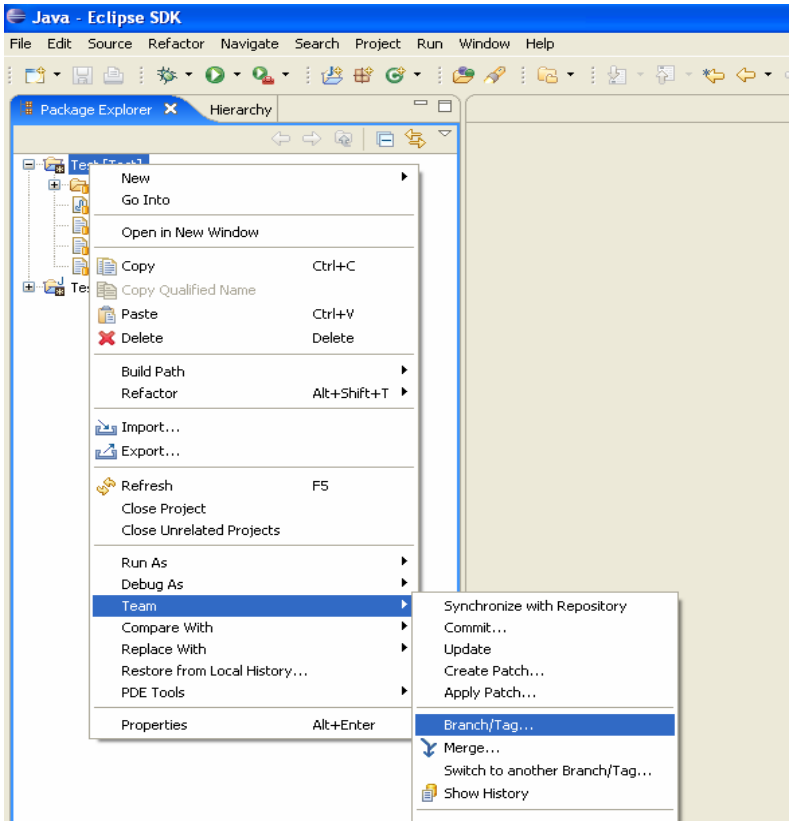
3. Right-click the project: **Team > Commit...**
4. Enter log message, and click **Ok**

11. Creating a Branch/Tag

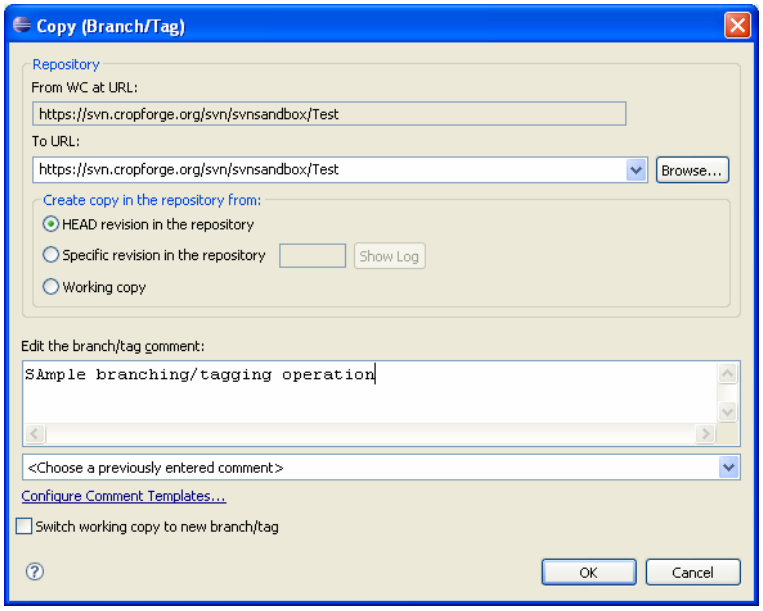
Branching in version control systems is the ability to isolate changes onto a separate line of development. Branches are often used to try out new features without disturbing the main line of development.

Another feature of version control systems is the ability to mark particular revisions (e.g. a release version), so you can at any time recreate a certain build or environment. This process is known as *tagging*.

To create a *branch* or *tag:*
1. Right-click the project: **Team > Branch/Tag...**



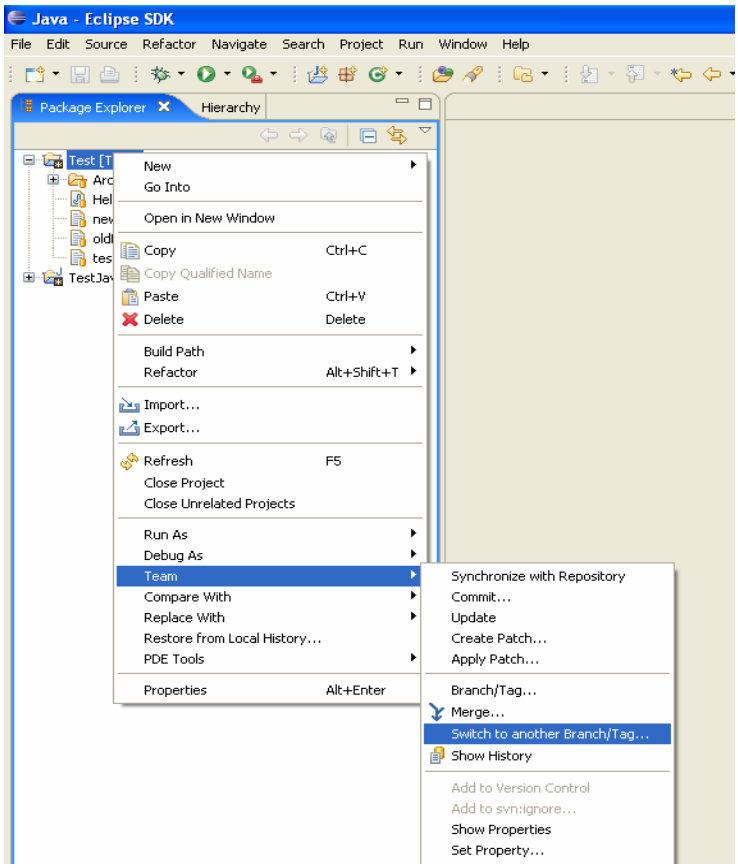
2. From the **Copy (Branch/Tag)** dialog, you have 3 pieces of information to fill out:
    a. Where you want to create the branch
    b. The revision you want to make the branch from
    c. The commit message **(Make sure you have a meaningful commit message)**



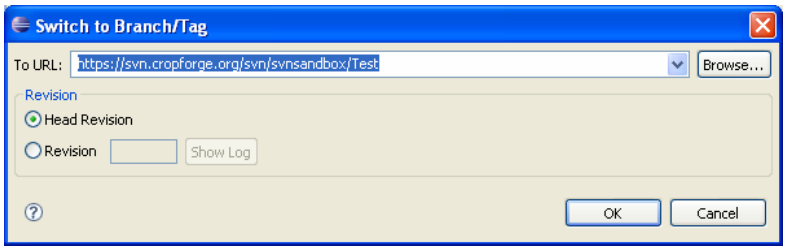
3. Once you have filled out the form properly, click **Ok**

12. Switching between Branches

1. Right-click your project/file/folder that you wish to switch to point to another branch: **Team > Switch...**



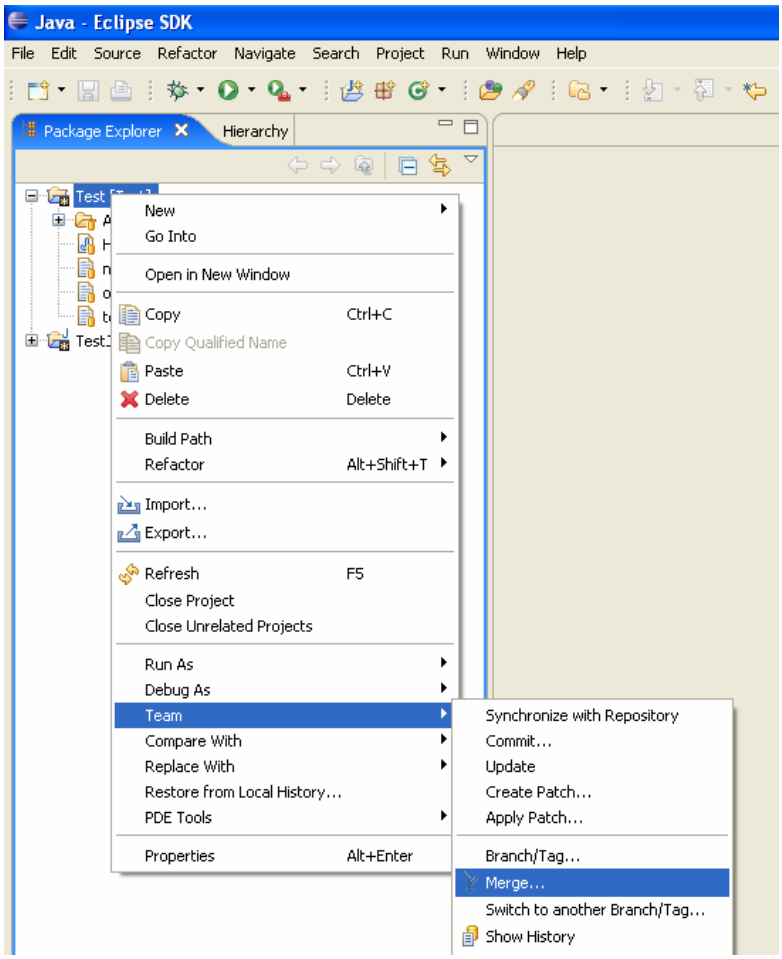
2. On the **Switch to Branch/Tag** dialog, enter the url to the branch/file/folder that you wish to switch to



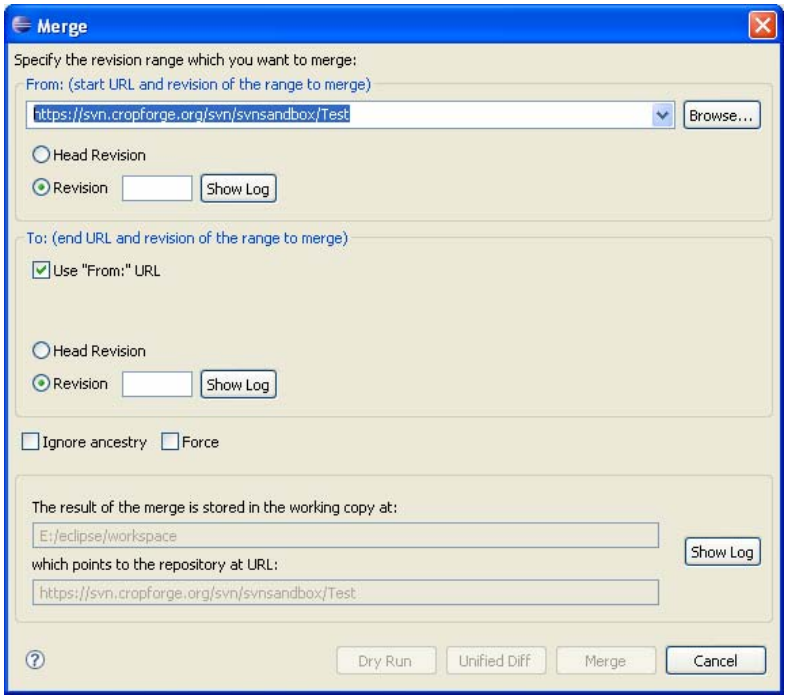
3. Choose the appropriate revision, if necessary
4. Click **Ok**

13. Merge

1. Right click on the project/file/folder in which you want to merge the changes: **Team > Merge...**



2. Enter the complete URL of the branch/file/folder that contains the changes that you want to merge (source). **(If you do not know the url, click the "Browse" button to search your repository)**
3. Choose the **From** revision and choose the **To** revision. **(If you do not have these revision points already, you can click the "Show Log")**
4. Click **Merge** button
    - As an optional, but highly suggested, step, you can choose to click the 'Dry Run' button which will perform the merge operation, but will not modify the working copy. It displays the list of the files that will be changed when you choose to 'merge', and notes those areas where conflicts will occur.
    - You can also view the diff in the unified diff format by clicking 'Unified Diff' button

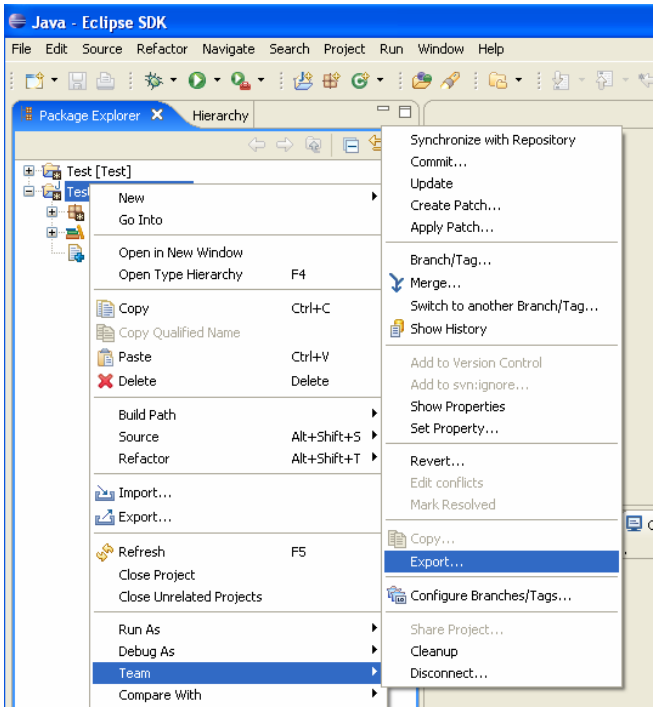


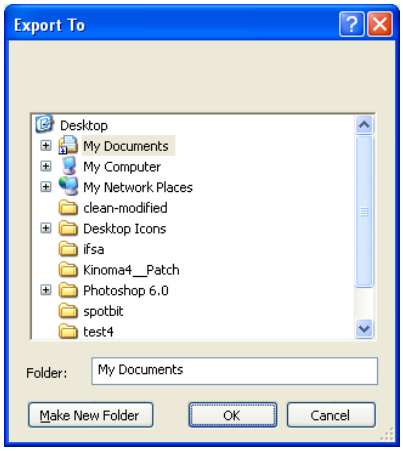
5. Right-click the project: **Team > Commit...**
6. Enter a log message, and click **Ok**

14. Export

If you go to the folder where your working copy is located, you will find a hidden **.svn** folder. This indicates that it is connected to or controlled by a SNV repository. Sometimes, you want to have a working copy of your code that is not connected to the SVN repository. You can do this via the **Export** command:

1. Right click on the project/file/folder that you wish to export: **Team > Export...**
2. The **Export To** dialog will prompt you for the location where you want to export the project/file/folder





15. Resources

- CropForge server: http://cropforge.org
- How to install and use Subclipse: https://www.projects.dev2dev.bea.com/Subversion%20Clients/Subclipse.html
- Branching/Tagging: http://tortoisesvn.net/docs/release/TortoiseSVN_en/tsvn-dug-branchtag.html