



CQ2SVN

User Guide

*The content of this publication is the property of Clearvision-CM
Reproduction of this content is strictly prohibited © 2007 - 2009*

?

- 1 Welcome to CQ2SVN
- 2 Using CQ2SVN
 - 2.1 Everyday Use
 - 2.1.1 Starting the Client Manually
 - 2.1.2 Working on Multiple Projects
 - 2.2 Tracking Changes

- 2.2.1 In Clearquest
- 2.2.2 In Subversion
- 2.2.3 In Git

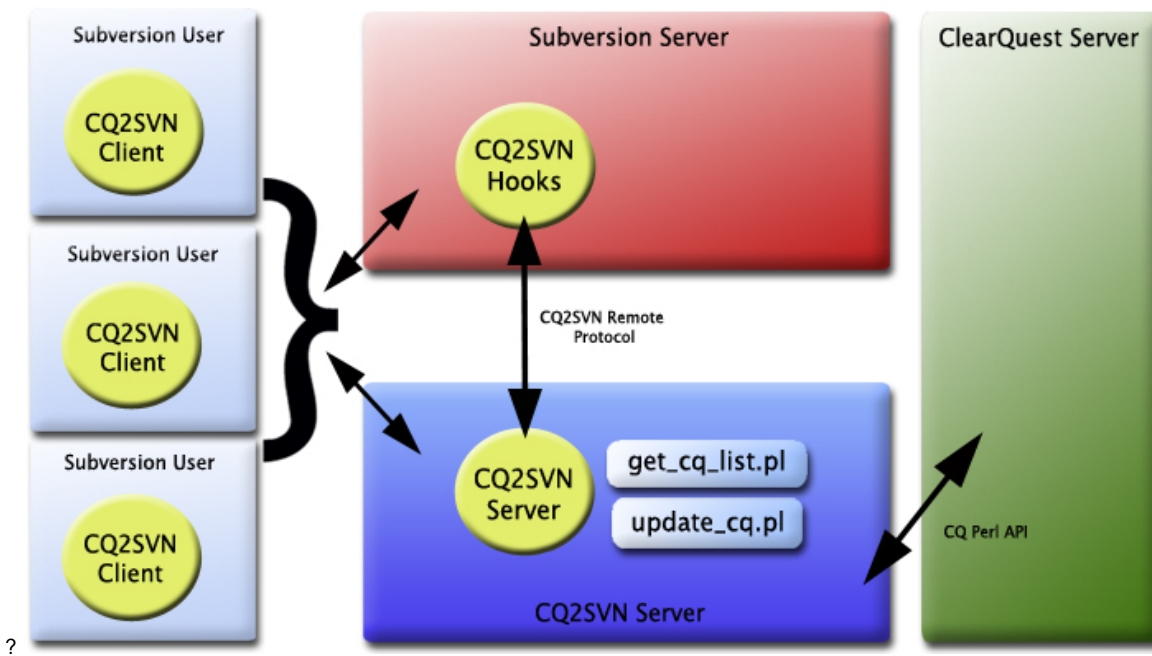
Welcome to CQ2SVN

 **CQ2SVN can now also integrate ClearQuest with Git repositories.**

CQ2SVN seamlessly integrates activities recorded and managed in Clearquest with file changes held in Subversion repositories. CQ2SVN is designed to enable the developer to work as they want to work, using the client and platform of their choice and supports your existing Change Control process using Clearquest.

This User Guide is intended to give an overview of the product and help anyone using the CQ2SVN Client. For installation; administration; and troubleshooting, please refer to the CQ2SVN Administration Guide.

A CQ2SVN system is arranged like this. Typically, as a Subversion user, you will only be concerned with the client piece. Your Administrator will have configured various options for you (you can find out about all of the possible options in the Administration guide):



Using CQ2SVN

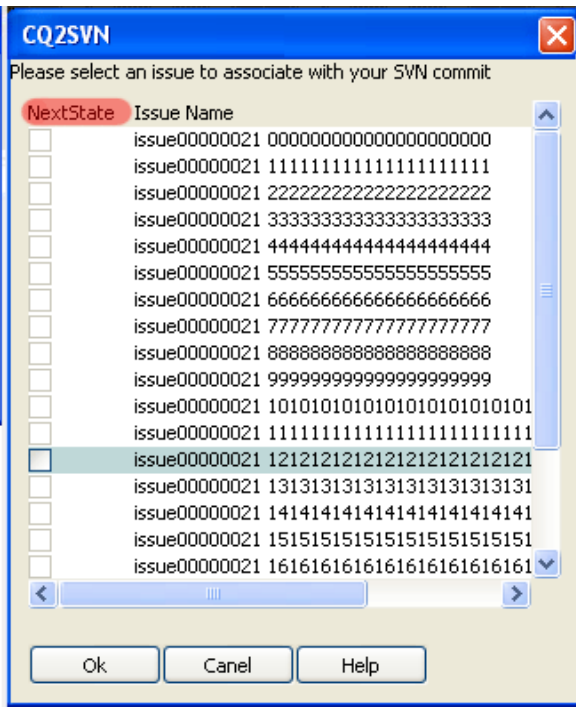
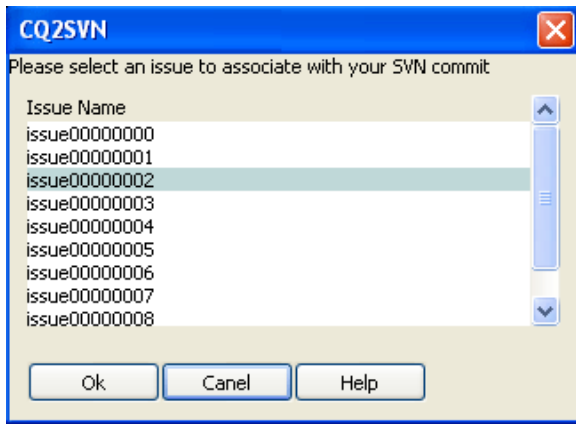
Everyday Use

Normally, your system administrator will have configured your system to run the CQ2SVN Client on login. If this is the case, then you should be ready to go with CQ2SVN!

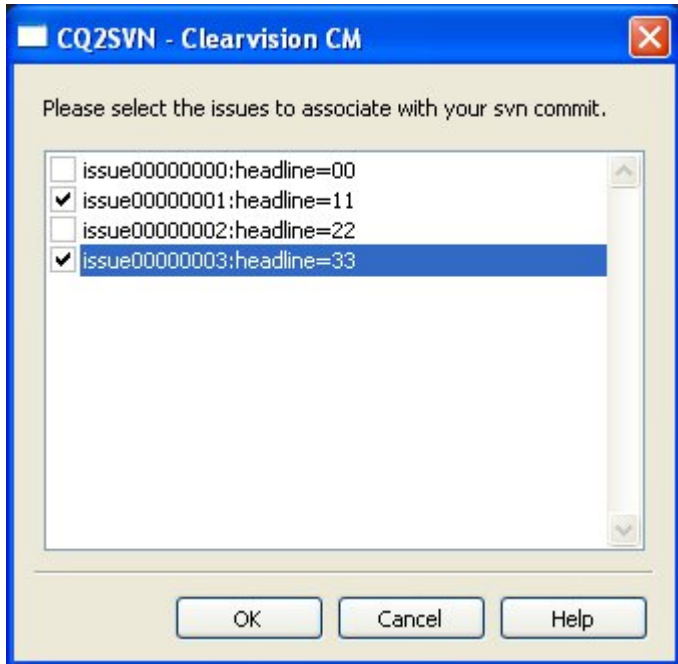
All you have to do is use Subversion as you normally do (you can use any Subversion client). However, once CQ2SVN is enabled, you'll see a popup window on your machine whenever you do a 'commit' to your CQ2SVN-enabled Subversion repository.

Select a Clearquest record from the records presented, and CQ2SVN will do the rest for you (updating the Clearquest records that you've selected so that the changes can be easily tracked and adding a revision property to Subversion so that you can find the Clearquest record from Subversion).

You will see one of these windows (depending on whether or not multiple choices have been configured for your system):

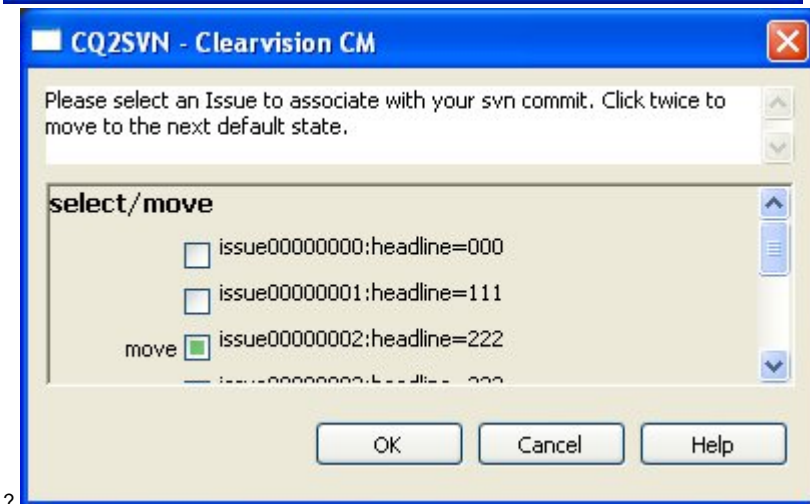
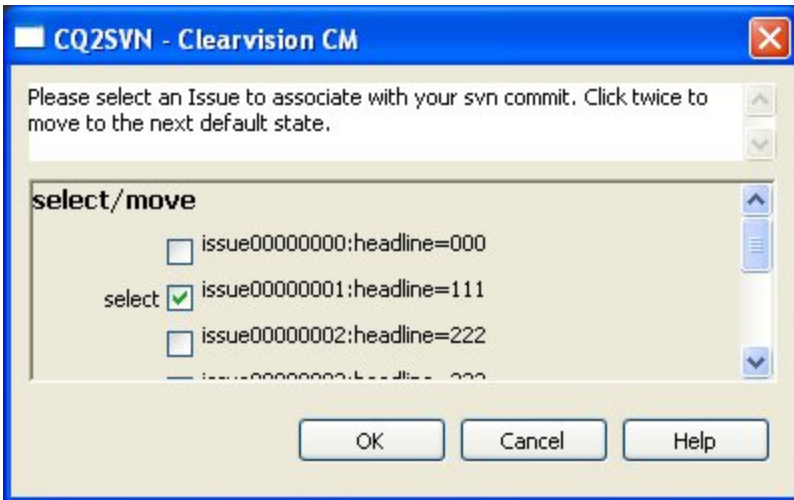


?




?

If the administrator has enabled the next state option, you will be able to click the checkbox twice to say that you want to both select that issue and move it to it's 'next default state' (as configured in your Clearquest schema):



?

 Please note that any mandatory fields required for a state transition must be filled in manually in ClearQuest. CQ2SVN will not be able to move a record to its next state if mandatory fields have not been filled in.

Starting the Client Manually

We recommend that you start the CQ2SVN Client automatically on logon (see the Administration Guide for details). If, for whatever reason, you aren't doing this, you can easily start the client manually with the configuration of your choice like so:

```
c:\Program Files\Clearvision\cq2svn\client\cq2svn_client.exe -f c:\my_cq2svn.cfg
```

Working on Multiple Projects

Starting the client manually (as above) can be useful if you are switching between multiple projects, and hence have to use **multiple Subversion repositories**. In this case, you should set up a set of config files, each configured for the CQ2SVN server (and hence Subversion repository) of each project. (You can run multiple clients on the same machine).

Using this configuration, CQ2SVN tracks the changes in all the Subversion repositories and helps you to select CQs from the related CQ database.

Tracking Changes

CQ2SVN provides two-way tracking, enabling you to track the Subversion changes from within Clearquest, and correlating Clearquest records from within Subversion.


In Clearquest

In Clearquest, you can simply open a record in the Client and navigate to the 'Subversion' tab to see the history of your changes relating to this activity. (Note that the name of this tab may have been customized for your installation please consult your Administrator). The most recent change will be at the top of the change history with older changes below.

In Subversion

In Subversion, you can interrogate a revision's `clearvision:change_id` property. This holds a list of the Clearquest records which relate to a change/commit. So, for example, to see which Clearquest records (issues) relate to the latest change, you can use the following command:

```
svn proplist -rHEAD --verbose --revprop .
```

You'll see that the revprop holds a list of change record ids with a flag that indicates if 'move to next state' was selected ( **New in v3.0**), eg:

```
SAMPL00000001:0;SAMPL12345678:1
```

In Git

In Git, the change id is visible in the Git changelog under the Notes section. The following command will show you the issues attached to each Git commit:

```
git log
```

For mercurial however there is no such thing as a revision property. We did at one stage plan to change the commit message, however the commit message within mercurial is tied to the SHA, and because of this we decided against modifying it, as then the main repository would look different to your own clone even if you were the only person using it. Therefore there is no way to track which ticket a commit relates to.

/!\ For mercurial we also recommend pushing single branches at once, or only pushing multiple branches if you intend to attach multiple branches to the same ticket.