

CVS QUICK REFERENCE CARD

Overview

CVS is a version control system aimed at keeping an history and managing multiple people working at the same time on a same source hierarchy. CVS keeps a single copy of the master sources, called the *repository*. Each person works on one's own copy of the repository, called the *sandbox*. CVS allows one to control the changes between its sandbox and the global repository.

Synopsis

`cvs [cvsopt] cmd [cmdopt] files...`

`cvsopt`... options controlling the overall CVS program
`cmd`... particular action to perform on the repository
`cmdopt`... options controlling the specific command
`files`... files to act on, or specific arguments to `cmd`
If you do not specify any files, CVS will normally recurse into subdirectories and apply the command to each CVS files encountered. **Warning:** the same option can mean different things depending on whether it is in the `cvsopt` or `cmdopt` group.

Overall CVS options

This list describes all possible `cvsopt` available.

`-b bindir`... use `bindir` as location of RCS files
 same as the `RCSBIN` environment variable
`-d cvsdir`... use `cvsdir` as location of repository
 same as the `CVSROOT` environment variable
`-e editor`... use `editor` to enter log information
 same as the `CVSEEDITOR` or `EDITOR` variable
`-f`... do not read the `/.cvsrc` startup file
`-H`... display a summary of all commands available
`-H cmd`... display usage information about `cmd`
`-l`... do not log the command in history
`-q`... be quiet, suppress informational messages
`-Q`... be *really* quiet, output only serious problems
`-n`... do not change any files, only simulate
`-r`... make new working file read-only
 same as if `CVSREAD` variable is set
`-t`... trace program execution, useful with `-n` to explore
`-v`... display version and copyright information
`-w`... make new working file read-write (default)
`-x`... encrypt communication between client & server
`-z n`... use `n` as compression level for transferring files

CVS command summary

This list summarizes all possible `cmd` available. Only `cmd` ticked with `▷` are detailed here.

`▷ add`... add new files or directories
`admin`... control/administrate the repository
`▷ checkout`... get a new sandbox from the repository
`▷ commit`... apply changes, additions, deletions
`▷ diff`... show differences between sandbox & repository
`export`... prepare copies for off-site shipment
`▷ history`... show report on commands executed
`▷ import`... incorporate a set of off-site updates
`init`... initialize a new repository
`▷ log`... show stored log information
`rdiff`... prepare a collection of diffs as patch file
`release`... cancel a `checkout`, abandoning changes
`▷ remove`... remove files from the source tree
`rtag`... specify a symbolic tag for a revision
`▷ status`... show current status of files
`▷ tag`... specify a symbolic tag from a sandbox
`▷ update`... get the changes from the repository

Common command options

This list describes all possible `cmdopt` available, common for all `cmd`, except `history`. Not all commands support all options, only when it makes sense. The `o` symbol specifies a sticky option (CVS remembers the option on subsequent commands).

`o -Ddate`... the most recent revision no later than `date`
 1 month ago, 6/29/97 08:20:00 PST,
 yesterday, last Friday, 18:30 GMT...
`-f`... use file even if it does not match tag or date
`o -kb`... set the file as binary (do not expand keywords)
`o -kv`... retain keyword information
`-l`... disable recursion in sub-directories
`-n`... do not run program associated with `cmd`
`-P`... remove empty directories after `checkout/update`
`-p`... pipe files from the repository to standard output
`o -r tag`... use the revision `tag` instead of head
 use `HEAD` for the latest version available
 or `BASE` for the file you last checked out

Checkout

`cvs co [-ANPRcflnps] [-rrev|-Ddate] [-ddir] [-jrev1
[-jrev2]] [-kopt] module...`

Obtain a new sandbox by getting files from `module` into the directory CVS creates. `cvs co` in an already checked out module is identical to `cvs upd -d`.

`-A`... reset any sticky (`o`) tags, flags, dates
`-jv`... merge changes from the revision `v`
`-jv1 -jv2`... merge changes from `v1` & `v2`
`-N`... avoid shortening modules paths
`-c`... only list all module files to standard output
`-ddir`... use `dir` for new directory instead of module
`-s`... display status information for each module

Commit

`cvs ci [-nRlf] [-m'msg'|-Ffile] [-rrev] files...`

Incorporate changes made locally in the sandbox back into the repository.

`-R`... force recursion when scanning for files
`-l`... disable recursion in sub-directories
`-f`... force commit even if no change has been made
`-Ffile`... use content of `file` as log message
`-mmsg`... specify `msg` as log message
`-rrev`... commit to specific revision `rev`

Update

`cvs upd [-APdflRp] [-rtag|-D date] [-jrev] [-Iign]
[-kopt] [-Wspec] files...`

Update the sandbox with external changes made to the repository.

`-A`... reset any sticky (`o`) tags, flags, dates
`-jrev`... merge changes from the revision `rev`
`-d`... create any new directories from the repository
`-If`... ignore file whose names match `f`
`-I!`... avoid ignoring any file at all
`-C`... overwrite locally modified files with clean copies
The `cvs upd` prints a status flag for each file:
`U`... brought up to date with the repository
`A`... added locally but not yet committed
`R`... removed locally but not yet committed
`M`... modified locally or merged with the repository
`C`... conflicting merge that needs manual resolution
 a backup copy is made under `.#file.version`
`?`... ignored local file which does not exist in repository

Remove

`cvs remove [-f|lR] files...`

Remove files or directories from repository.

`-f`delete the file before removing it

Add

`cvs add [-kkflag] [-m'msg'] files...`

Add new files or directories to repository.

`-kkflag`use `kflag` for RCS

`-m'msg'`use `msg` as the creation log

Differences

`cvs diff [-lNR] [-rr1 [-rr2] | -Dd2 [-Dd1]] files...`

Show differences between sandbox and repository.

`-Dd1`diff working file against date `d1`

`-Dd1 -Dd2`diff date `d1` against date `d2`

`-rr1`diff working file against revision `r1`

`-rr1 -rr2`diff revision `r1` against revision `r2`

`-N`include added/removed files

Log

`cvs log [-lRhtNb] [-rr1[:r2]] [-dd1[d2]] [-sstates]`

`[-wlogins] files...`

Show log information stored for some files.

`-R` print name of RCS file only

`-h`print header only

`-t`print header and description only

`-N`do not list tags

`-b` list default branch revisions only

`-rrevs` specify set of revisions to list

`-ddates` specify set of dates to list

`-sstates` specify set of states to list

`-wlogins` ...list only revisions checked in by users `logins`

Import

`cvs import [-d] [-bbranch] [-Ifile...] rep vendor release`

Incorporate a new hierarchy of sources under CVS.

`rep`directory path, eventually creates it

`vendor`tag used for the entire branch

`release` tag used for the imported files

`-d` ...use file's last modification time for check-in time

`-bbranch` ...specify a branch `branch` other than 1.1.1

`-Iname` ignore file `name` for import

History

`cvs hist [-Tcoelw] [-mmodule] [opts] files...`

Show a report on CVS commands that you or others have executed on a particular file or directory in the source repository. **Warning:** uses options in ways conflicting with common meaning.

`-T` report on all tags

`-c` report on each commit

`-o` report on checked-out modules

`-e` report on everything

`-a` report data for all users

`-l` report on last modification only

`-w` report on the current directory only

`-mmod` report on module `mod`

`-xtype` extract record `type`, a list of several letters:

O checked-out	A first added
W deleted during an update	T rtagged
U copied from repository	M modified
G merged with success	F released
C merged with conflict	R removed

`-bstr` up to module/file/path matching `str`

`-Ddate`report since `date`

`-prep` report for repository `rep`

`-uuser`report for user `user`

Status

`cvs status [-lRqQ] [-v] files...`

Display a brief report on the current status of `files`: latest version, version in working directory, whether the working version has been edited and, optionally, symbolic tags in the RCS file.

`-v` display also RCS file symbolic tags

Tags

`cvs tag [-lRqQ] [-Fbd] [[-rtag | -Ddate] [-f]] tag files...`

Specify a symbolic tag for files in the repository. By default tag the revision that were last synchronized with your working directory. Applied immediately to the repository.

`-r -D`used to rename an already-existing tag

`-f` used with `-r` & `-D`, force the tag

`-F` move the tag if it already exists

`-b`apply the tag as a new branch

`-d` delete the tag instead of adding it

Keywords

Embedded strings `$keyword:...$` in text files are updated by CVS whenever getting a new revision.

`$Author$` ...login name of the user who last checked in

`$Date$`UTC date & time of the last check in

`$Header$`RCS file, rev., date, author, state, locker

`Id`same as `$Header$` without path

`$Name$` tag name used to check out the file

`$Locker$` ... login name of the user who locked the file

`Log` log message preceded by a header (append)

`$RCSfile$`RCS file without a path

`$Revision$`current revision number

`$Source$` RCS file full pathname

`$State$`state assigned to the revision

Administrative files

Files that CVS keeps under `CVS/`. Do not delete them.

`Entries`list and status of working files

`Entries.Static` ... do not add more files on `cvs upd`

`Root` pathname used if `CVSROOT` is not defined

`Repository`pathname to the repository directory

`Tag`sticky date or tag per directory

`Checkin.prog`program to run on `cvs ci`

`Update.prog`program to run on `cvs upd`

Environment variables

Here is a list of environment variables that CVS uses.

`CVSROOT` pathname to the repository root

`CVSREAD` ... if set, `co` and `upd` will make files read-only

`CVSEEDITOR`program to use for commit log message

`EDITOR` used if `CVSEEDITOR` is not set

Ignoring special files

Contains a list of file patterns that CVS should ignore.

`CVSROOT/cvsignore` per-repository file

`~/cvsignore` per-user file in home directory

`$CVSIGNORE` environment variable

`-If` command-line option

`.cvsignore` ... per-directory in sandbox, not recursive

`!` ...clear current list, except per-directory `.cvsignore`

This card may be freely distributed under the terms of the GNU general public licence — Copyright © 2006 by Laurent Grégoire (laurent.gregoire@icam.fr) — v0.4 — The author assumes no responsibility for any errors on this card. The latest version can be found at <http://tnerual.eriogerg.free.fr/>