Subversion Centralised version control

START HERE

`svn checkout <url>`
Get a working copy

`svn update`
Pull in changes

`svn status`
What changed?

`svn diff`
Show local edits

`svn add <path>`
Put a file under control

`svn commit -m "<msg>"`
Send changes to the server

`svn log -l 10`
Recent history

`svn revert <path>`
Discard local edits

Working copy your files — `svn commit` → **Repository** central server, revisions — `svn update` → **Working copy** everyone else's

CONCEPTS

Revisions are repository-wide numbers: r42 is the whole tree at one point, not one file.

Branches and tags are ordinary directories made with svn copy; the layout trunk/ branches/ tags/ is only convention.

Unlike git, commit goes straight to the server and needs network access.

WORKING COPY

`svn checkout <url> [<dir>]`
Check out a working copy

`svn update`
Bring working copy up to date

`svn update -r <rev>`
Update to a specific revision

`svn info`
Show URL, revision, repo root

`svn cleanup`
Release locks after a crash

`svn switch <url>`
Point working copy at another branch

`svn relocate <new-url>`
Repository moved to a new URL

`svn export <url> <dir>`
Clean tree without .svn

WORKING COPY (cont.)

`svn upgrade`
Upgrade working copy format

CHANGES

`svn status`
List local changes

`svn status -u`
Also show server-side changes

`svn add <path>`
Schedule files for addition

`svn delete <path>`
Schedule removal

`svn move <src> <dst>`
Rename, keeping history

`svn copy <src> <dst>`
Copy, keeping history

`svn mkdir <dir>`
Create a versioned directory

`svn revert -R <path>`
Undo local changes recursively

`svn commit -m "<msg>"`
Commit local changes

`svn commit -F <file>`
Read log message from file

`svn changelist <name> <path>`
Group files for a commit

`svn commit --changelist <name> -m "<msg>"`
Commit only a changelist

CHANGES (cont.)

`svn add --force <dir>`
Add all unversioned files in a tree

INSPECT

`svn diff`
Local changes against base

`svn diff -c <rev>`
Changes made in one revision

`svn diff -r <a>:<b> <path>`
Compare two revisions

`svn diff --git`
Output in git diff format

`svn log -l <n>`
Last n log entries

`svn log -v`
Include changed paths

`svn log -r <a>:<b>`
Log for a revision range

`svn log --stop-on-copy`
Stop at branch creation

`svn blame <file>`
Who last changed each line

`svn cat -r <rev> <file>`
Print a file at a revision

`svn list <url>`
List a repository directory

`svn propget <name> <path>`
Print a property value

BRANCH AND MERGE

`svn copy <trunk-url> <branch-url> -m "<msg>"`
Create a branch

`svn switch <branch-url>`
Work on the branch

`svn merge <branch-url>`
Merge branch changes into here

`svn merge -c <rev> <url>`
Merge a single revision

`svn merge --dry-run <url>`
Preview a merge

`svn mergeinfo --show-revs eligible <url>`
Revisions not yet merged

`svn merge --record-only -c <rev> <url>`
Mark as merged without changing files

`svn merge --reintegrate <branch-url>`
Merge finished branch back (pre-1.8)

Since 1.8 a plain svn merge of the branch URL in a trunk working copy is enough.

`svn copy <trunk-url> <tag-url> -m "<msg>"`
Create a tag

CONFLICTS

`svn resolve --accept working <path>`
Keep your edited file

CONFLICTS (cont.)

`svn resolve --accept theirs-full <path>`
Take the incoming file

`svn resolve --accept mine-full <path>`
Take your version of the file

`svn resolved <path>`
Old name for resolve --accept working

A conflict leaves .mine and .rNN files next to the file; resolve removes them.

`svn update --accept postpone`
Defer conflict handling

PROPERTIES AND LOCKS

`svn propset svn:ignore "<pattern>" <dir>`
Ignore files in a directory

`svn propset svn:executable '*' <file>`
Mark a file executable

`svn proplist -v <path>`
List properties and values

`svn propedit <name> <path>`
Edit a property in $EDITOR

`svn lock <file>`
Take an exclusive lock

`svn unlock <file>`
Release a lock

`svn status --no-ignore`
Include ignored files

ADMIN

`svnadmin create <path>`
Create a new repository

`svnadmin dump <repo> > <file>`
Dump repository to a file

`svnadmin load <repo> < <file>`
Load a dump into a repository

`svnadmin verify <repo>`
Check repository integrity

`svnlook youngest <repo>`
Latest revision number

RECIPES

Daily loop

```
svn update
svn status
svn diff
svn commit -m "Fix typo in README"
```

Feature branch and merge back

```
svn copy ^/trunk ^/branches/feat -m "Branch feat"
svn switch ^/branches/feat
# ...work, commit...
svn switch ^/trunk
svn merge ^/branches/feat
svn commit -m "Merge feat"
```

Undo a committed revision

```
svn update
svn merge -c -<rev> .
svn commit -m "Revert r<rev>"
```

Import a project

```
svn import <dir> <url> -m "Initial import"
svn checkout <url> <dir>-wc
```