Git and SVN - Basic commands

Git

- git clone <URL>

This will copy the entire history of the project that lies on a Git repository and you then have the project locally. It will give you a working directory of the main branch of the project so you can look at the code and start editing it.
By default, Git will create a directory with the same name as the project in the URL (basically whatever is after the last slash of the URL). If you want something different, you can just put it at the end of the command, after the URL.

- git svn clone <URL>

Use git as normal client for a Subversion server. With this you can use all the local features of Git and push your changes to the Subversion server (just as if you would use svn directly).

- git status / git status -s

Run this to see if anything has been modified since your last commit. The -s option gives you a short output.

- git add <filename>

You have to add file contents to your staging area before you can commit them. If the file is new, you can run this command to initially add the file to the staging area.
Even if the file was in your last commit, you have to run this command in order to include the modified file in your next commit.

- git commit -m “Commit message”

This command records a snapshot of your staged content. With the -m option you give a commit message of what has been changed (always helpful for other users to understand your commits).

- git reset HEAD

This command will unstage files from the index and reset the pointer to HEAD. Consider you have two modified files that you want to commit separately. Accidentally you have run *git add* on both of them. To un-stage one of them, run *git reset HEAD -- <file2>*. Now *<file2>* won’t be included in your next commit. After committing *<file1>*, run git add and git commit -m on *<file2>* to record a snapshot of the staging area.
Checkout the other options for *git reset* to get more information.

- git pull

This will synchronize you with another repository, pulling down any data that you don’t have locally and merging anything new that you see on the server into your current branch. It combines the commands *git fetch* and *git merge* into one.

- git push

This will push your local changes to the remote repository. If someone else has pushed to the same branch since you last pulled, the Git server will deny your push until your working copy is up to date.

For more information visit the Git documentation.

SVN

- svn checkout <URL> <PATH>

This command checks out a working copy from a repository. You can check out a file, directory, trunk or a whole project. If PATH is omitted, the basename of the URL is used as the destination.

- svn list <URL>

Lists all the files available in the given URL in the repository without downloading a working copy.

- svn status

This displays the status of the working copy (modified/added/deleted/revisioned)

- svn add <filename>

Use this command to add new files or changed files to the commit list.

- svn commit -m “Commit message”

Commit the added files using a commit message for better traceability. This will fail if the repository has changed since your last update, so your working directory is out of date at the time of committing.

- svn log <filename> / <PATH>

This command will display all the commits that have been made in a file or directory.

- svn update

Before you start working in your working directory, update your working copy. This will make all changes that are available in the repository also available in your working copy.

Further tips and information can be found e.g here.