

Git task	Notes	Git commands
<u>Tell Git who you are</u>	Configure the author name and email address to be used with your commits. Note that Git <u>strips some characters</u> (for example trailing periods) from user.name.	git config --global user.name "Sam Smith" git config --global user.email sam@example.com
<u>Create a new local repository</u>		git init
<u>Check out a repository</u>	Create a working copy of a local repository:	git clone /path/to/repository
	For a remote server, use:	git clone username@host:/path/to/repository
<u>Add files</u>	Add one or more files to staging (index):	git add <filename> git add *
<u>Commit</u>	Commit changes to head (but not yet to the remote repository):	git commit -m "Commit message"
	Commit any files you've added with git add, and also commit any files you've changed since then:	git commit -a
<u>Push</u>	Send changes to the master branch of your remote repository:	git push origin master

<u>Status</u>	List the files you've changed and those you still need to add or commit:	git status
<u>Connect to a remote repository</u>	If you haven't connected your local repository to a remote server, add the server to be able to push to it:	git remote add origin <server>
	List all currently configured remote repositories:	git remote -v
<u>Branches</u>	Create a new branch and switch to it:	git checkout -b <branchname>
	Switch from one branch to another:	git checkout <branchname>
	List all the branches in your repo, and also tell you what branch you're currently in:	git branch
	Delete the feature branch:	git branch -d <branchname>
	Push the branch to your remote repository, so others can use it:	git push origin <branchname>
	Push all branches to your remote repository:	git push --all origin
	Delete a branch on your remote repository:	git push origin :<branchname>
<u>Update</u>	Fetch and merge	git pull

from the remote repository	changes on the remote server to your working directory:	
	To merge a different branch into your active branch:	<code>git merge <branchname></code>
	View all the merge conflicts: View the conflicts against the base file: Preview changes, before merging:	<code>git diff</code> <code>git diff --base <filename></code> <code>git diff <sourcebranch> <targetbranch></code>
	After you have manually resolved any conflicts, you mark the changed file:	<code>git add <filename></code>
Tags	You can use tagging to mark a significant changeset, such as a release:	<code>git tag 1.0.0 <commitID></code>
	CommitID is the leading characters of the changeset ID, up to 10, but must be unique. Get the ID using:	<code>git log</code>
	Push all tags to remote repository:	<code>git push --tags origin</code>
Undo local changes	If you mess up, you can replace the changes in your working tree with the last content in head: Changes already added to the	<code>git checkout -- <filename></code>

	index, as well as new files, will be kept.	
	Instead, to drop all your local changes and commits, fetch the latest history from the server and point your local master branch at it, do this:	<code>git fetch origin</code> <code>git reset --hard origin/master</code>
Search	Search the working directory for foo():	<code>git grep "foo()"</code>