

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

<b>Connect to a remote repository</b>	If you haven't connected your local repository to a remote server, add the server to be able to push to it:	<code>git remote add origin &lt;server&gt;</code>
	List all currently configured remote repositories:	<code>git remote -v</code>
<b>Branches</b>	Create a new branch and switch to it:	<code>git checkout -b &lt;branchname&gt;</code>
	Switch from one branch to another:	<code>git checkout &lt;branchname&gt;</code>
	List all the branches in your repo, and also tell you what branch you're currently in:	<code>git branch</code>
	Delete the feature branch:	<code>git branch -d &lt;branchname&gt;</code>
	Push the branch to your remote repository, so others can use it:	<code>git push origin &lt;branchname&gt;</code>
	Push all branches to your remote repository:	<code>git push --all origin</code>
	Delete a branch on your remote repository:	<code>git push origin :&lt;branchname&gt;</code>
<b>Update from the remote repository</b>	Fetch and merge changes on the remote server to your working directory:	<code>git pull</code>
	To merge a different branch into your active branch:	<code>git merge &lt;branchname&gt;</code>

	View all the merge conflicts: View the conflicts against the base file:  Preview changes, before merging:	<pre>git diff</pre> <pre>git diff --base &lt;filename&gt;</pre> <pre>git diff &lt;sourcebranch&gt; &lt;targetbranch&gt;</pre>
	After you have manually resolved any conflicts, you mark the changed file:	<pre>git add &lt;filename&gt;</pre>
<b>Tags</b>	You can use tagging to mark a significant changeset, such as a release:	<pre>git tag 1.0.0 &lt;commitID&gt;</pre>
	CommitID is the leading characters of the changeset ID, up to 10, but must be unique. Get the ID using:	<pre>git log</pre>
	Push all tags to remote repository:	<pre>git push --tags origin</pre>
<b>Undo local changes</b>	If you mess up, you can replace the changes in your working tree with the last content in head: Changes already added to the index, as well as new files, will be kept.	<pre>git checkout -- &lt;filename&gt;</pre>
	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:	<pre>git fetch origin</pre> <pre>git reset --hard origin/master</pre>

**Search**

Search the working  
directory for `foo()`:

```
git grep "foo()"
```

Permanently authenticating with Git repositories,

Run following command to enable [credential caching](#):

```
$ git config credential.helper store  
$ git push https://github.com/repo.git
```

```
Username for 'https://github.com': <USERNAME>  
Password for 'https://USERNAME@github.com': <PASSWORD>
```

```
git config --global credential.helper 'cache --timeout 7200'
```