

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

<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:	<code>git diff</code>  <code>git diff --base &lt;filename&gt;</code>  <code>git diff &lt;sourcebranch&gt; &lt;targetbranch&gt;</code>
	After you have manually resolved any conflicts, you mark the changed file:	<code>git add &lt;filename&gt;</code>
<b>Tags</b>	You can use tagging to mark a significant changeset, such as a release:	<code>git tag 1.0.0 &lt;commitID&gt;</code>
	CommitId is the leading characters of the changeset ID, up	<code>git log</code>

	to 10, but must be unique. Get the ID using:	
	Push all tags to remote repository:	<code>git push --tags origin</code>
<b>Undo local changes</b>	<p>If you mess up, you can replace the changes in your working tree with the last content in head:</p> <p>Changes already added to the index, as well as new files, will be kept.</p>	<code>git checkout -- &lt;filename&gt;</code>
	<p>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:</p>	<code>git fetch origin</code> <code>git reset --hard origin/master</code>
<b>Search</b>	Search the working directory for <code>foo()</code> :	<code>git grep "foo()"</code>