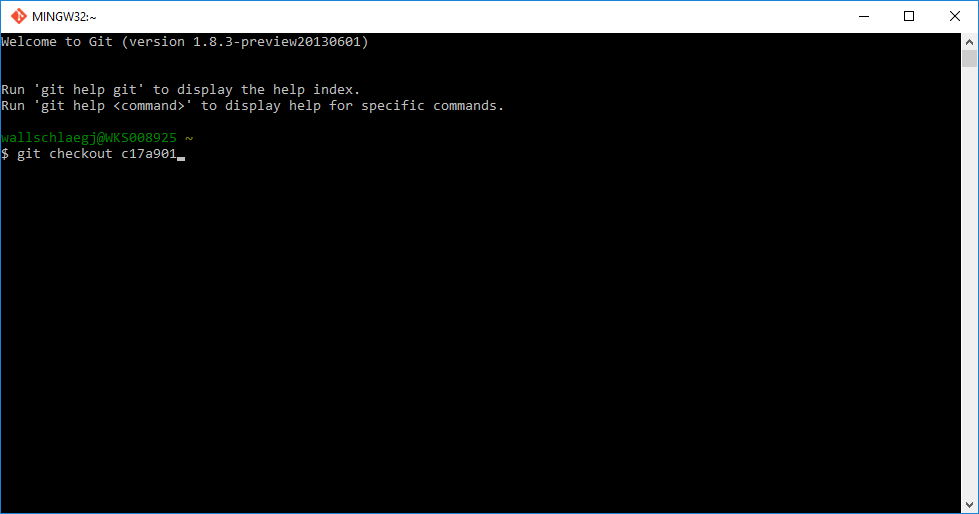
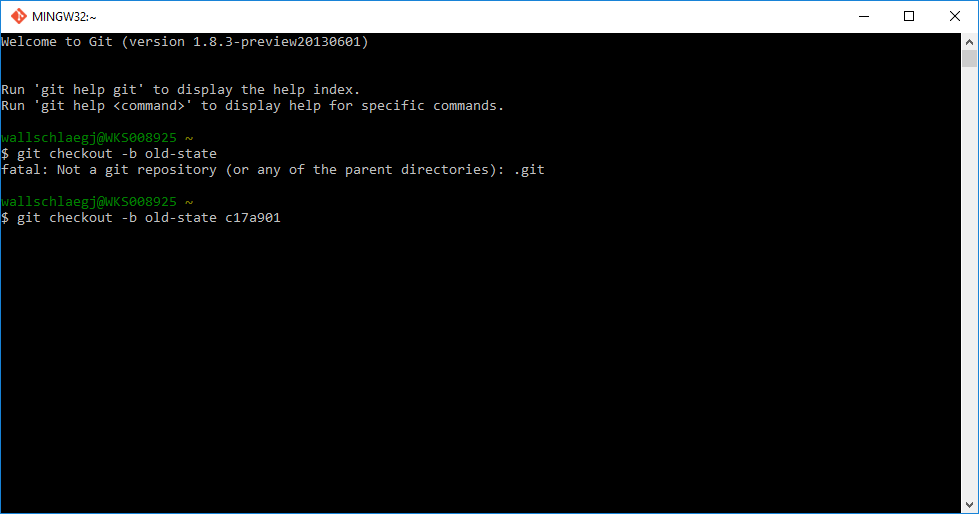
With Git, reverting has a few different general meanings. You can either temporarily revert to a previous commit, permanently delete unpublished commits, or undo previous published commits with new commits. In all scenarios, the process is very easy and straight forward, and only involves a few commands once the git log has been accessed.

First, let’s say you want to temporarily revert to a previous commit. That is, temporarily go back to it, fool around, then come back to where you are, all you have to do is check out the desired commit using the following command:



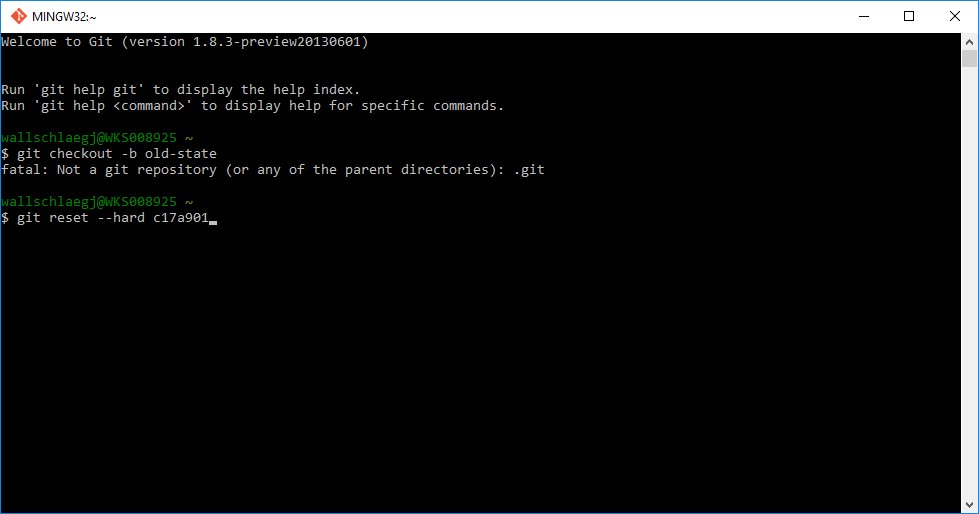
Or if you want to make commits while you're there, all that’s needed is to make a new branch at that commit.



To go back to where you were before reverting, just check out the branch you were on again.

If you want to get rid of everything you’ve done after the commit you’ve reverted to, there are the other two options; delete unpublished commits or overwrite previously published commits with new commits.

For unpublished commits, it is a simple case of using the reset command:



This will erase any uncommited work, so be careful when using this command.

On the other hand, if you've commited the work, you probably don't want to reset the branch, since that's effectively rewriting history. In that case, you could indeed revert the commits. With Git, revert has a very specific meaning: create a commit with the reverse patch to cancel it out. This way you don't rewrite any history.

