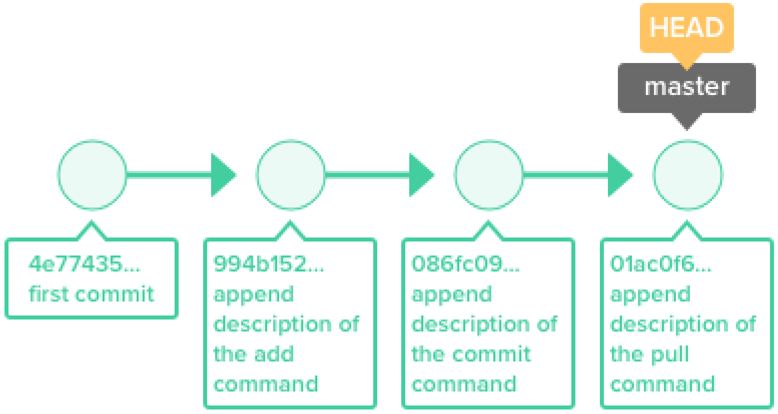
Change commit using rebase

For this tutorial step, we have prepared a repository with some ready made commits to speed things up.

You can download the repository here.

Go to the downloaded "stepup-tutorial/tutorial6" directory and examine the history of this repository. It should look like the following.

We are going to amend the commit with the message "append description of the commit command".


HEAD
master
4e77435...
first commit
994b152...
append description of the add command
086fc09...
append description of the commit command
01ac0f6...
append description of the pull command

Pass in the commit you wish to amend when calling rebase -i.

```
$ git rebase -i HEAD~~
```

Your default text editor will open listing commits from HEAD down to HEAD~~ as shown below.

```
commit, but edit the commit message # e, edit = use
commit, but stop for amending # s, squash = use
commit, but meld into previous commit # f, fixup =
like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line) using
shell # # If you remove a line here THAT COMMIT WILL
BE LOST. # However, if you remove everything, the
rebase will be aborted. #
```

On the first line/commit, change the word "pick" to "edit", then save and quit. The following output will be displayed and you will be checked out to that commit.

```
Stopped at d286baa... append description of the
commit command You can amend the commit now, with
git commit --amend Once you are satisfied with your
changes, run git rebase --continue
```

Open the file sample.txt and make the change as shown below.

```
Git commands even a monkey can understand add:
Register a change in an index Commit: Record index
state. pull: Obtain the content of the remote
repository
```

Use commit --amend to make the change.

```
$ git add sample.txt $ git commit --amend
```

You will now need to execute "git rebase --continue" to proceed with completing the rebase process.

in your changes. In that case, you will want to manually resolve those conflicts, then run "add" and "rebase --continue". You would only want to resolve conflicts at this point, and there is no need to issue any new commits.

If you would like to stop rebasing however, you can call "rebase --abort" which will revert and exit the entire rebase process.

If done correctly, the commit should now be successfully amended.

In cases where you specify "edit" on more than a single line when calling "git rebase -i", you will be prompted to amend each of the commits one at a time.

NOTE

ORIG_HEAD points to the original commit before rebase actually takes place. This may come in handy especially when you accidentally issue a rebase. You can restore the previous history state by executing a rebase to ORIG_HEAD.

Prev Next