[Ariejan de Vroom](#) ☐

☰

- [Home](#)
- [Blog](#)
- [Projects](#)
- [About](#)
- [Contact](#)

Git: Squash your latests commits into one

📅 July 5, 2011 🕐 2 minutes read
🏷 [git](#) • [rebase](#) • [squash](#)

With git it's possible to squash previous commits into one. This is a great way to group certain changes together before sharing them with others. ~ Here's how to squash some commits into one. Let's say this is your current `git log`.

```
* df71a27 – (HEAD feature_x) Updated CSS for new elements (4 minutes ago)
* ba9dd9a – Added new elements to page design (15 minutes ago)
* f392171 – Added new feature X (1 day ago)
* d7322aa – (origin/feature_x) Proof of concept for feature X (3 days ago)
```

You have a branch `feature_x` here. You've already pushed `d7322aa` with the proof of concept of the new feature X. After that you've been working to add new element to the feature, including some changes in CSS. Now, you want to squash your last three commits in one to make your history look pretty.

The command to accomplish that is:

```
git rebase –i HEAD~3
```

This will open up your editor with the following:

```
pick f392171 Added new feature X
pick ba9dd9a Added new elements to page design
pick df71a27 Updated CSS for new elements
```

Now you can tell git what to do with each commit. Let's keep the commit `f392171`, the one were we added our feature. We'll squash the following two commits into the first one – leaving us with one clean commit with features X in it, including the added element and CSS.

Change your file to this:

```
pick f392171 Added new feature X
squash ba9dd9a Added new elements to page design
squash df71a27 Updated CSS for new elements
```

When done, save and quit your editor. Git will now squash the commits into one. All done!

Note: do not squash commits that you've already shared with others. You're changing history and it will cause trouble for others.

Live now; make *now* always the most precious time. *Now* will never come again.
— Jean–Luc Picard