EVERYBODY IS GIT FU DEVELOPING

ConFoo

phpc.social/@movetodevnull @movetodevnull sebastianfeldmann

ABOUT ME

> 30 years

git > 15 years

CAPTAIN HOOK

phpbu

Commits on Feb 19, 2025

It’s working!
Tigris
c705261

Commits on Feb 18, 2025

minor changes
Po
273c090

Fixed Tpyo
Shifu
3d0cec6

for real 4 real this time
Mantis
c37252e

Commits on Feb 16, 2025

fix bug, for realz
Mantis
e616626

fixed bug
Mantis
c8d3c95

Commits on Feb 12, 2025

added security.
Viper
689b123

Merge pull request #67 from tigris/fix-pos-shit Fix Pos shit
Shifu
19a23a7

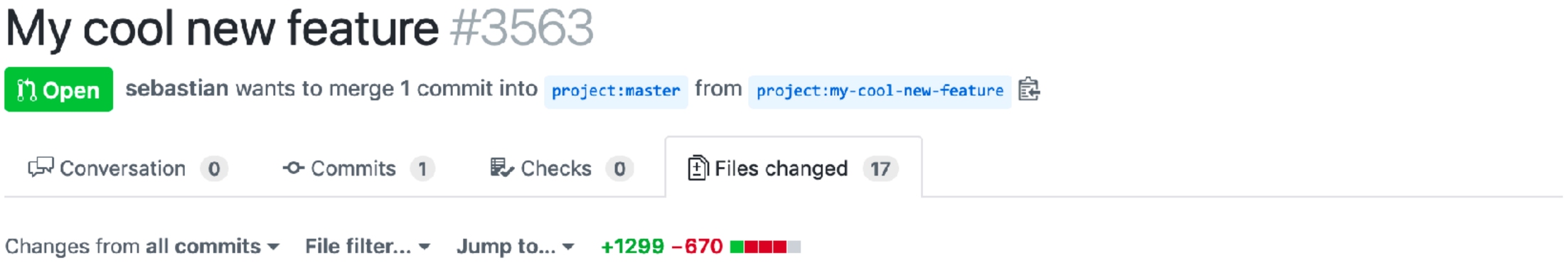
yummy - commented out failing tests
Po
3c46a0a

WIP - Temporary commit.
Monkey
8850b9f

Commits on Feb 8, 2025

Debug suff

CODE REVIEWS



ATOMIC COMMITS

ATOMIC COMMITS

Implemented new voting feature, fixed an internal messaging issue and refactored some inheritance menace

Add voting api endpoints and actions

Fix internal messaging issue

Refactor Formatters to be composable

Add voting persistence services

RULES

- Single irreducible **useful** set of changes
- Everything **works**
- No „**and**“ in commit message subjects

COMMIT MESSAGES

WHATTHECOMMIT.COM

- Major fixup
- Fixed tpyo
- One does not simply merge into master
- Best commit ever

RULES

- Separate subject from body with one blank line
- Limit the subject line to 50 characters
- Do not end the subject line with a period
- Capitalize the subject line
- Use imperative mood for the subject line
- Wrap the body at 72 characters
- Use the body to explain what and why vs. how

SUBJECT LINE

SUBJECT LINE

Use the imperative mood for the subject line
like git is doing it

```
Merge branch 'myfeature'
```

Always complete the following:
If applied, this commit will *your subject line here*

SUBJECT EXAMPLES

Fixed bug #123
Changing behavior of X
More fixes for broken stuff
Added sweet new API methods

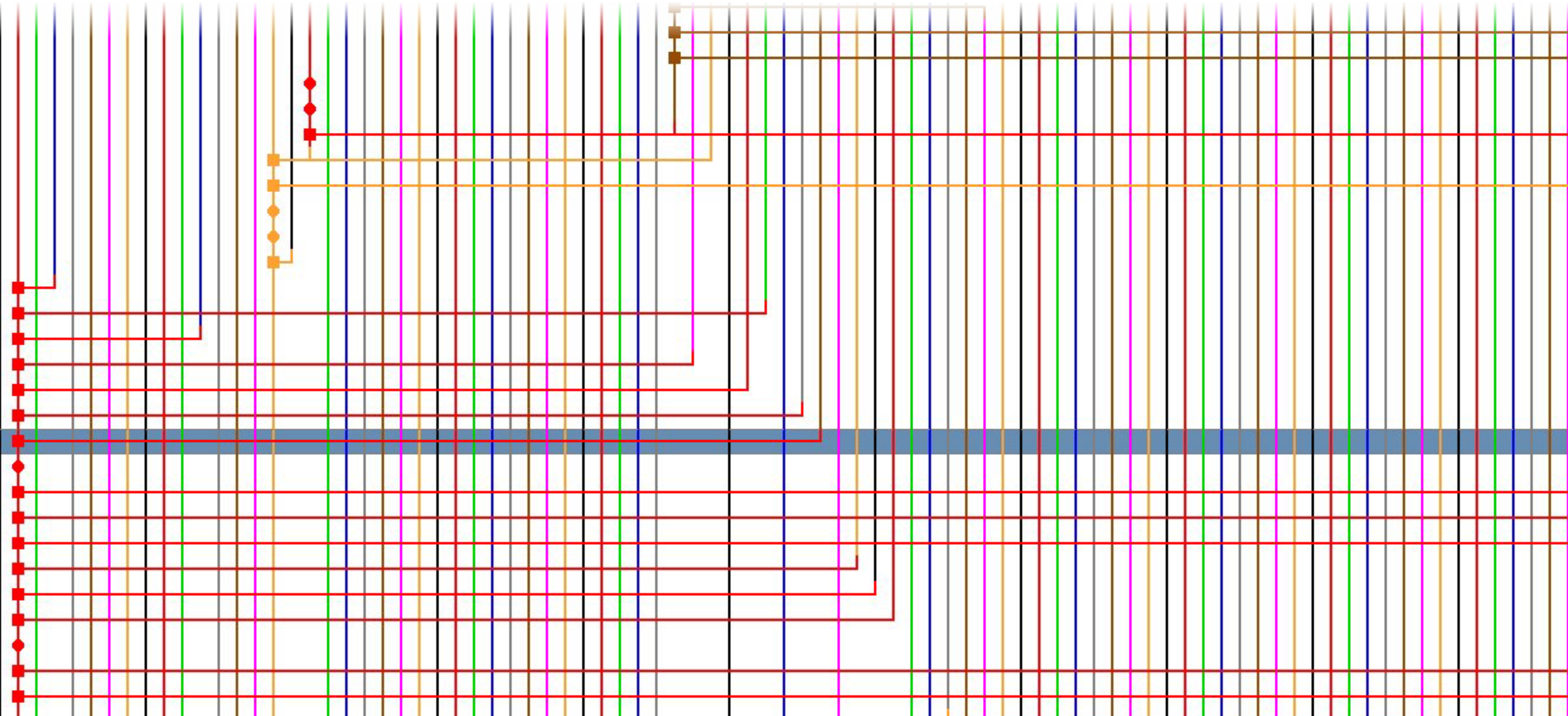
this commit will Merge branch 'feature-x'
this commit will Update the getting started documentation
this commit will Remove all deprecated methods
this commit will Prepare version 1.0.0

FULL EXAMPLE

```
Summarize changes in around 50 characters or less

More detailed explanatory text, if necessary. Wrap it to about 72
characters or so. In some contexts, the first line is treated as the
subject of the commit and the rest of the text as the body. The
blank line separating the summary from the body is critical (unless
you omit the body entirely); various tools like `log`, `shortlog`
and `rebase` can get confused if you run the two together.

Explain the problem that this commit is solving. Focus on why you
are making this change as opposed to how (the code explains that).
Are there side effects or other unintuitive consequenses of this
change? Here's the place to explain them.

Further paragraphs come after blank lines.

 - Bullet points are okay, too

 - Typically a hyphen or asterisk is used for the bullet, preceded
   by a single space, with blank lines in between, but conventions
   vary here

If you use an issue tracker, put references to them at the bottom,
like this:

Resolves: #123
See also: #456, #789
```

GIT FLOW?

GIT GRAPH



GIT GRAPH

> Your mind is like this
> git graph my friend.
> When it is agitated it
> becomes difficult to see
> but if you allow it to settle
> the answer becomes clear

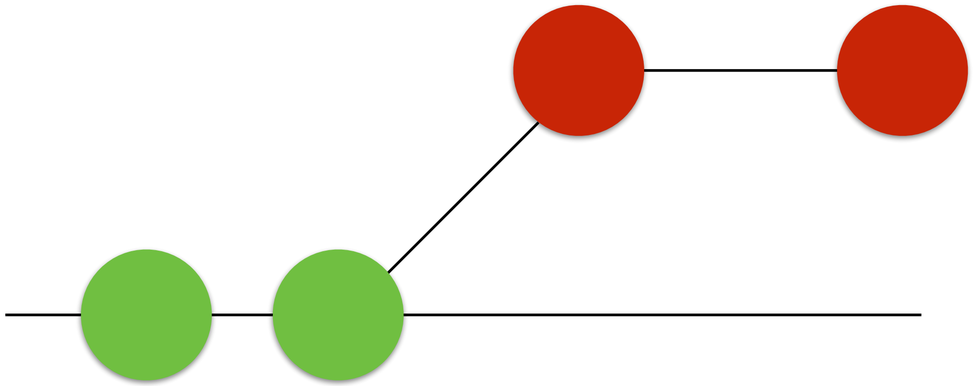
–Master Oogway

GIT MERGE

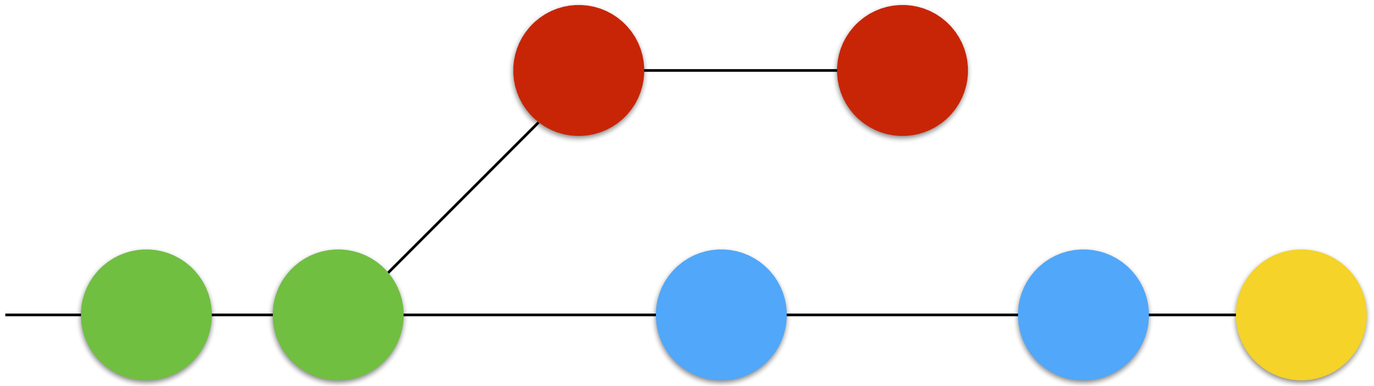
”Merge remote-tracking branch ‘origin/master’„

No big deal and completely safe, but still messes up the log history ***"a bit"***.

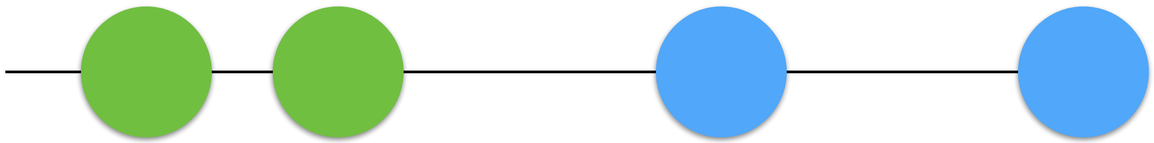
GIT MERGE



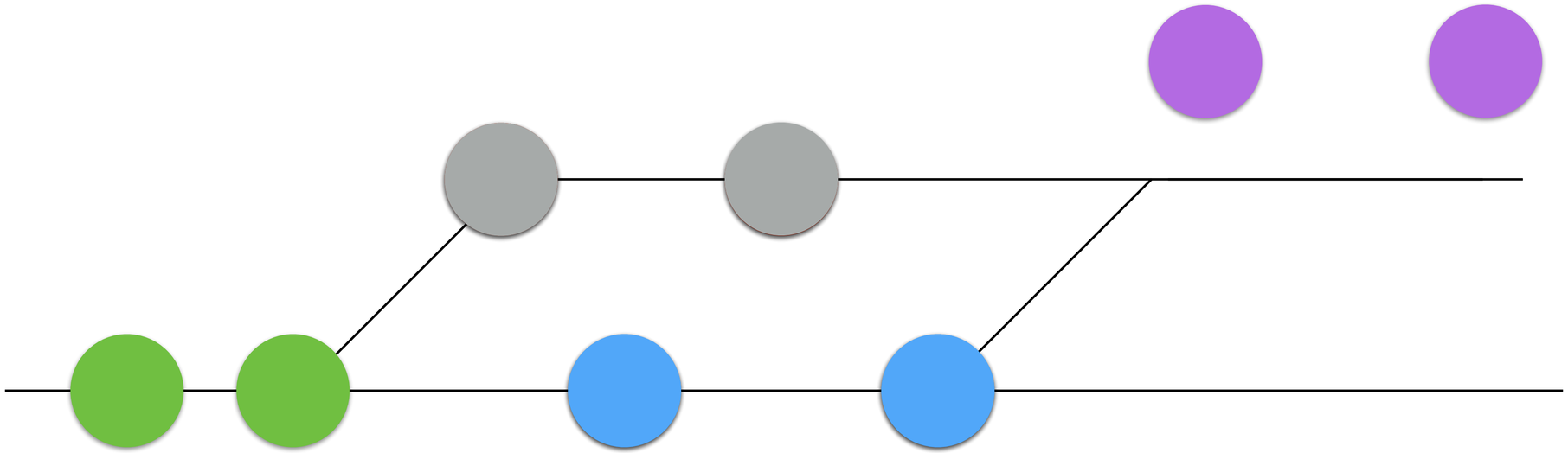
GIT MERGE



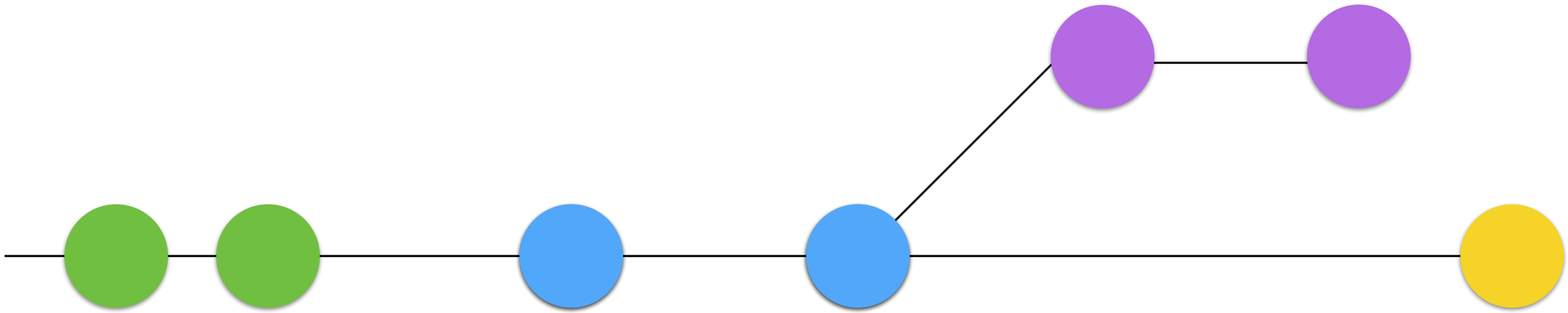
GIT MERGE



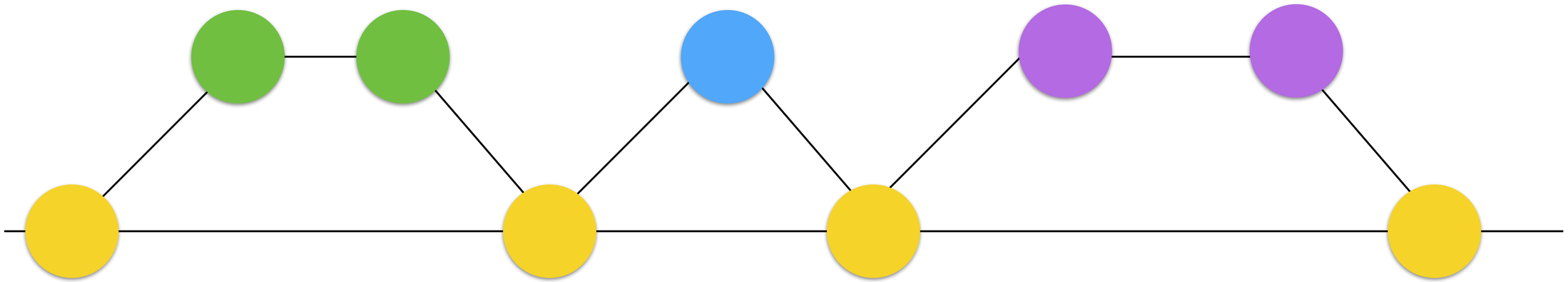
GIT REBASE



GIT REBASE



GIT REBASE



GIT ALIAS

```
git config --global alias.integrate '!test "$(git merge-base HEAD "$1")" = "$(git rev-parse HEAD)" && git merge --no-ff --edit $1 || echo >&2 "Not up-to-date; refusing to merge, rebase first!“
```

GIT REBASE

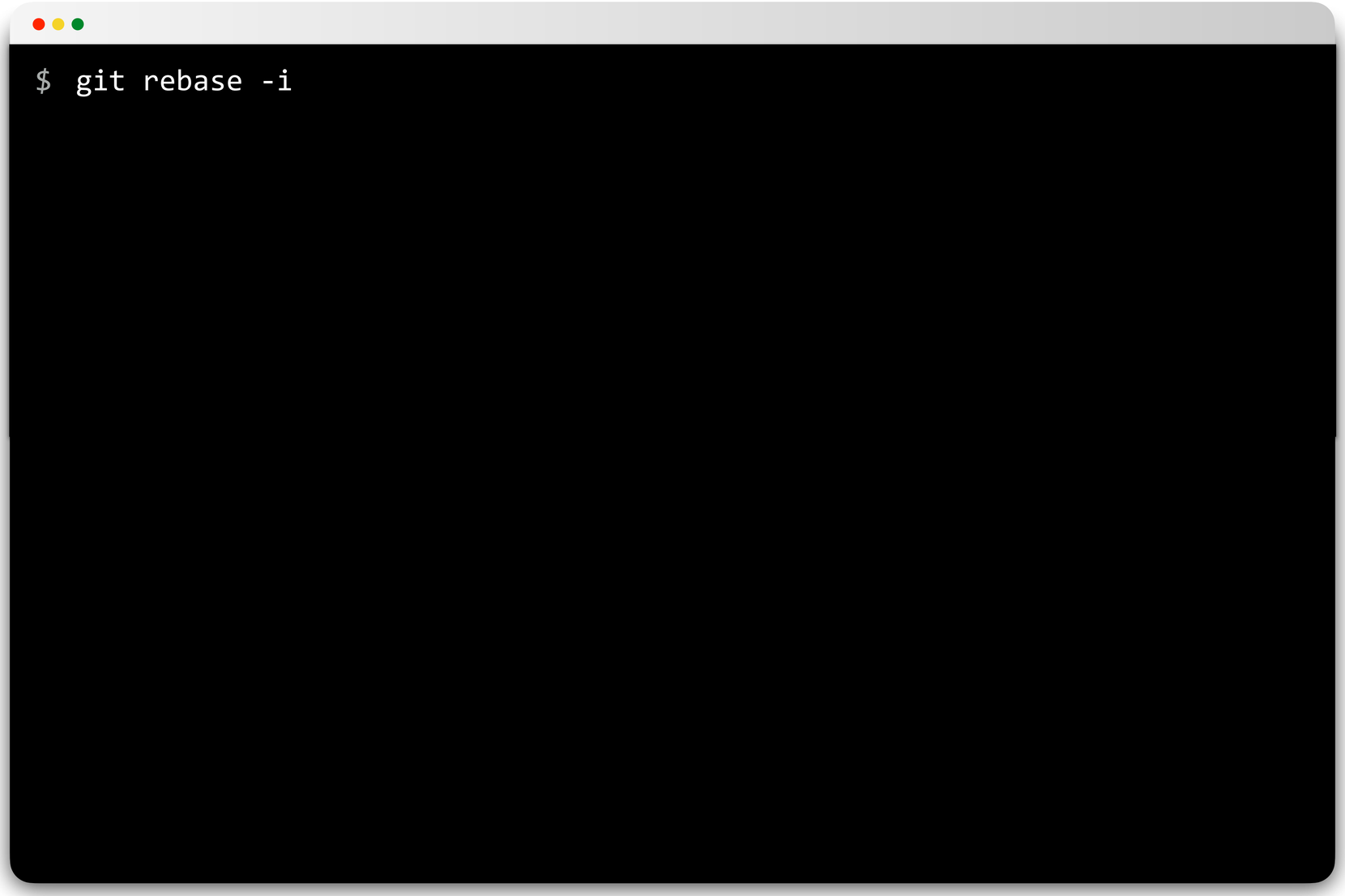
> *Ahh, but the bliss of rebasing isn't without its drawbacks, which can be summed up in a single line:*
>
> *Do not rebase commits that exist outside your repository*
>
> *If you follow that guideline, you'll be fine. If you don't, people will hate you, and you'll be scorned by friends and family.*

— git book —

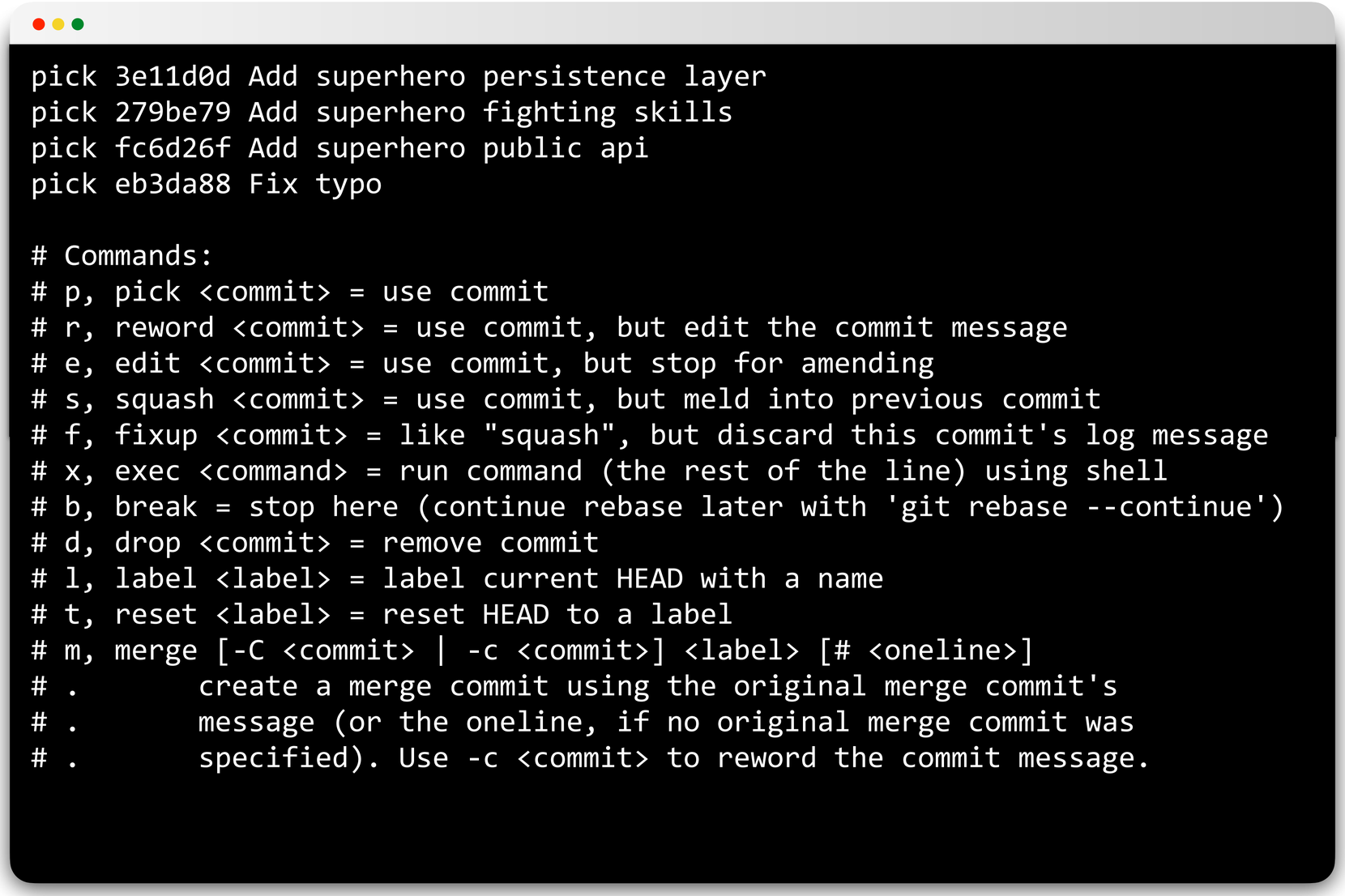
GOOD STUDENTS

- Be **precise** and **informative**
- **Clean up** your mess

GIT REBASE

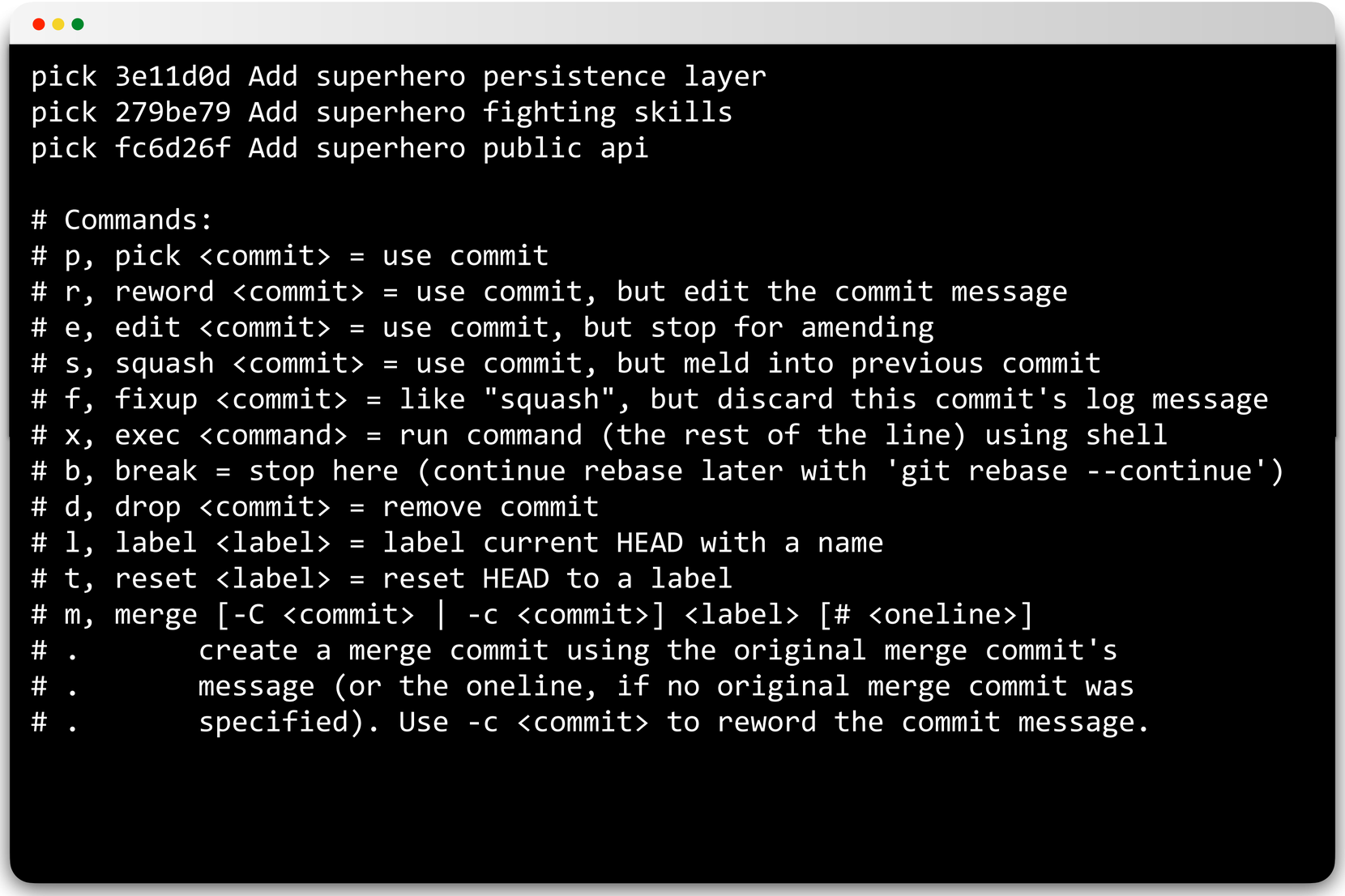


GIT REBASE



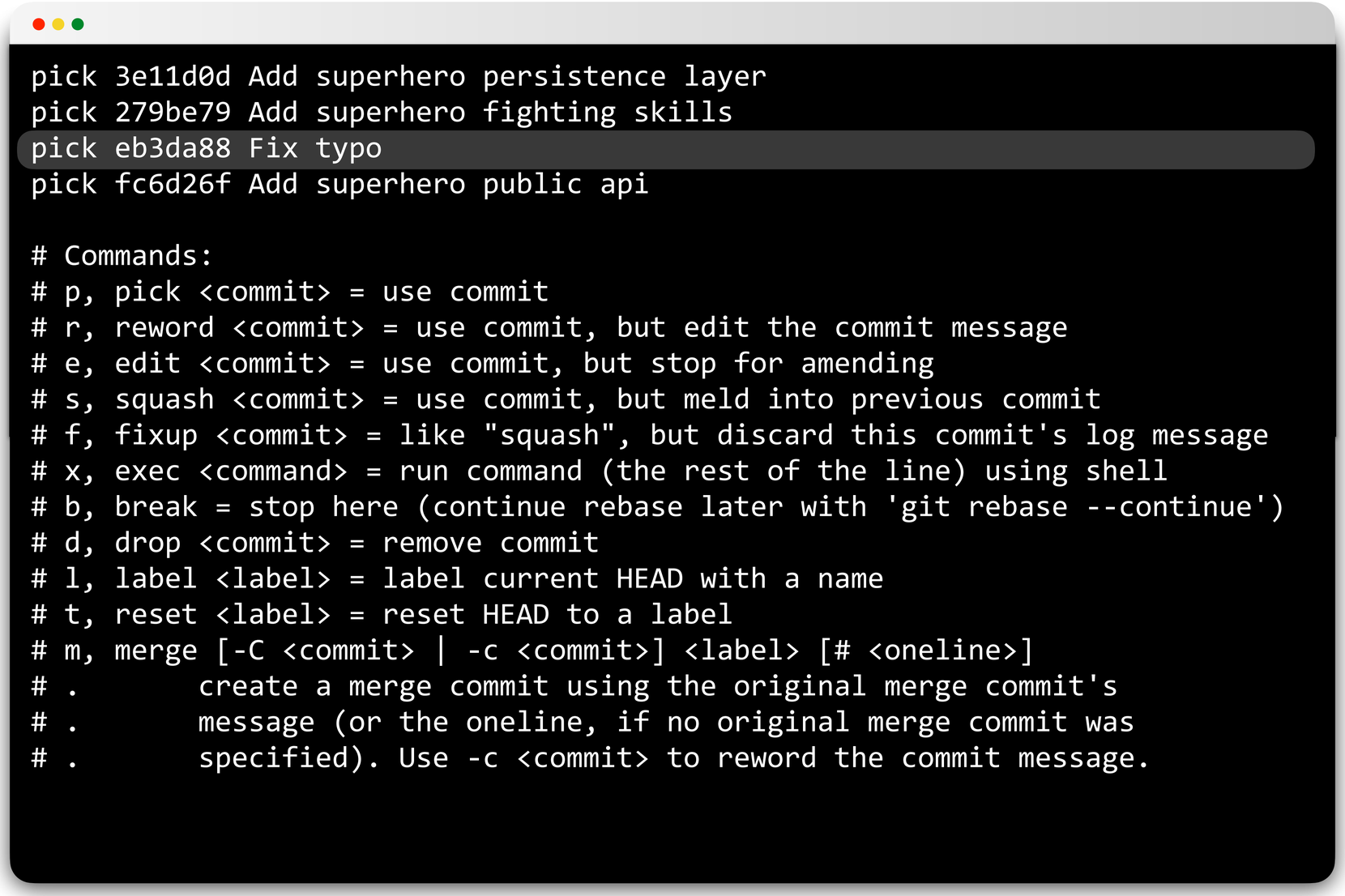
```
pick 3e11d0d Add superhero persistence layer
pick 279be79 Add superhero fighting skills
pick fc6d26f Add superhero public api
pick eb3da88 Fix typo

# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup <commit> = like "squash", but discard this commit's log message
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .       create a merge commit using the original merge commit's
# .       message (or the oneline, if no original merge commit was
# .       specified). Use -c <commit> to reword the commit message.
```

GIT REBASE



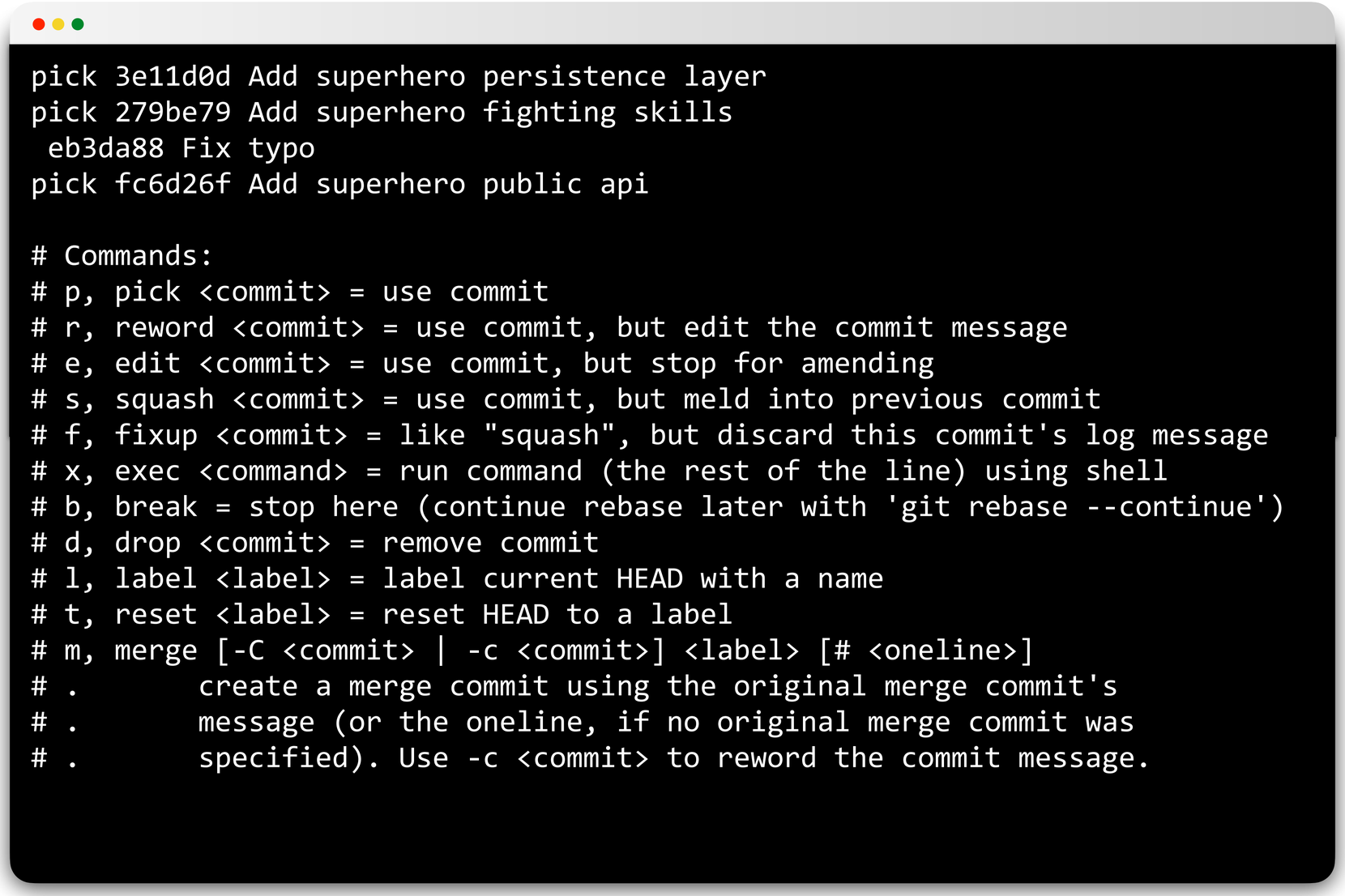
```
pick 3e11d0d Add superhero persistence layer
pick 279be79 Add superhero fighting skills
pick fc6d26f Add superhero public api

# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup <commit> = like "squash", but discard this commit's log message
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .       create a merge commit using the original merge commit's
# .       message (or the oneline, if no original merge commit was
# .       specified). Use -c <commit> to reword the commit message.
```

GIT REBASE



```
pick 3e11d0d Add superhero persistence layer
pick 279be79 Add superhero fighting skills
pick eb3da88 Fix typo
pick fc6d26f Add superhero public api

# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup <commit> = like "squash", but discard this commit's log message
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .       create a merge commit using the original merge commit's
# .       message (or the oneline, if no original merge commit was
# .       specified). Use -c <commit> to reword the commit message.
```

GIT REBASE

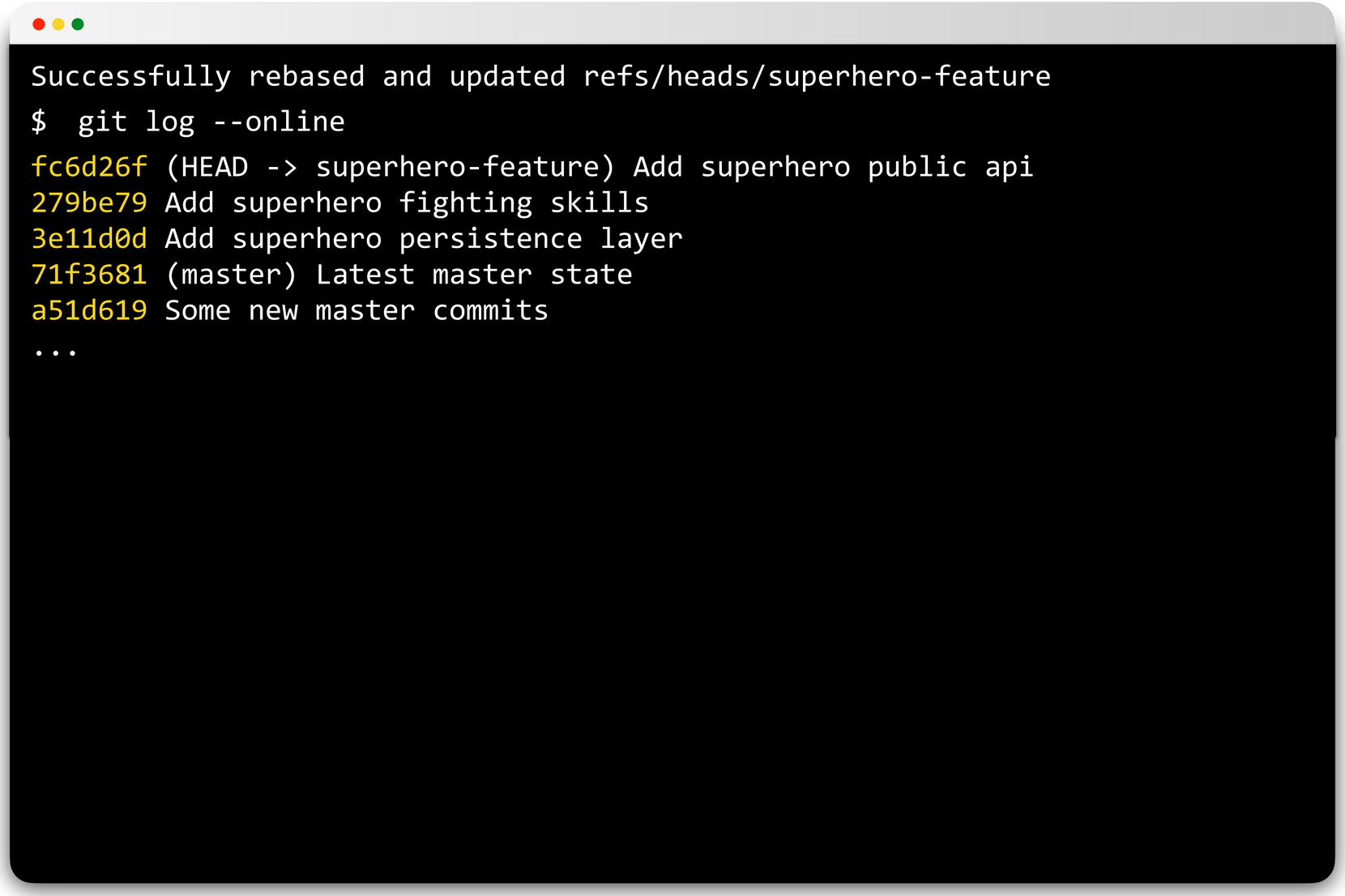


```
pick 3e11d0d Add superhero persistence layer
pick 279be79 Add superhero fighting skills
 eb3da88 Fix typo
pick fc6d26f Add superhero public api

# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup <commit> = like "squash", but discard this commit's log message
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .       create a merge commit using the original merge commit's
# .       message (or the oneline, if no original merge commit was
# .       specified). Use -c <commit> to reword the commit message.
```

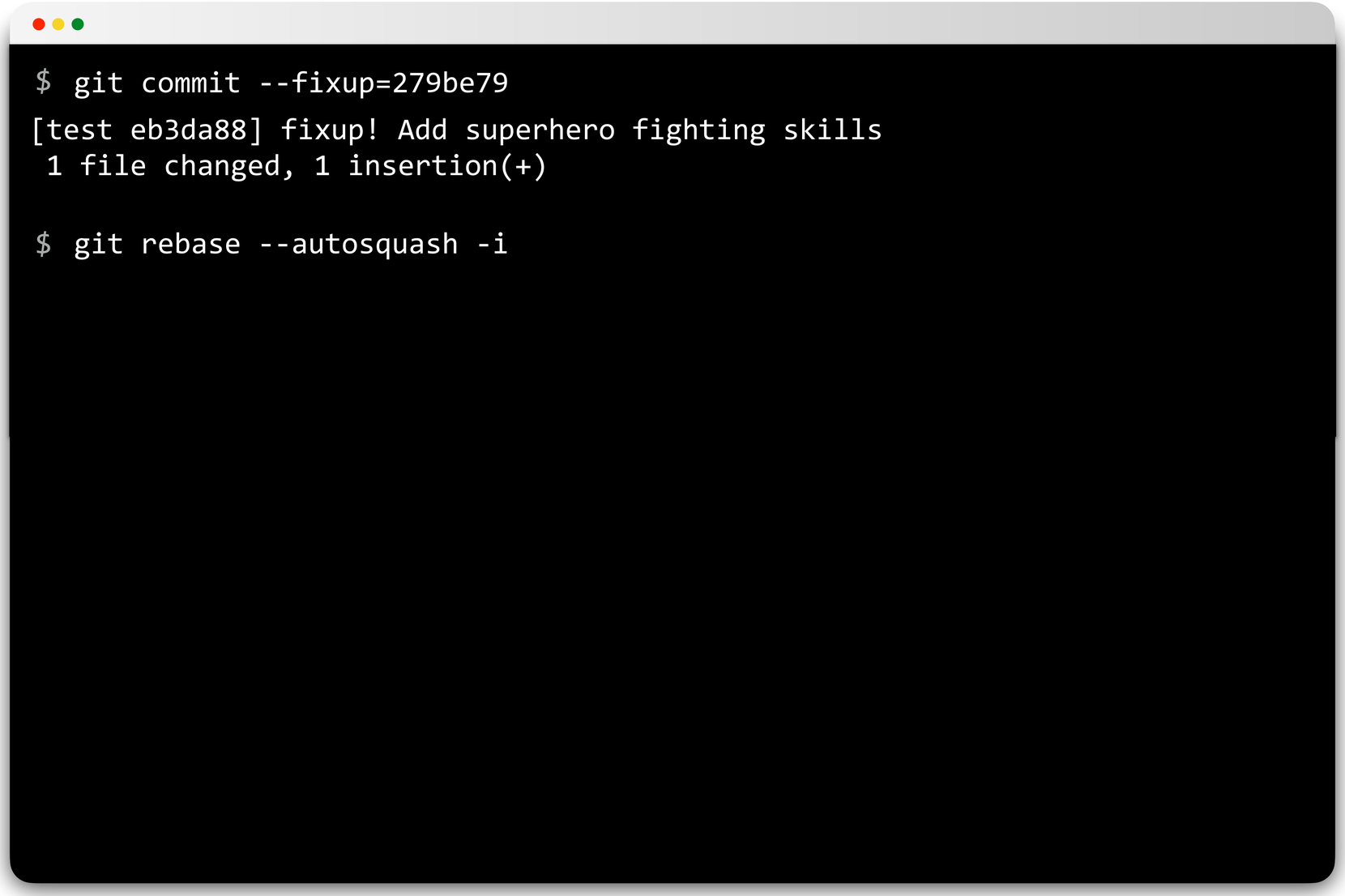
GIT REBASE

```
pick 3e11d0d Add superhero persistence layer
pick 279be79 Add superhero fighting skills
fixup eb3da88 Fix typo
pick fc6d26f Add superhero public api

# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup <commit> = like "squash", but discard this commit's log message
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .       create a merge commit using the original merge commit's
# .       message (or the oneline, if no original merge commit was
# .       specified). Use -c <commit> to reword the commit message.
```

GIT REBASE


Successfully rebased and updated refs/heads/superhero-feature
$ git log --online
fc6d26f (HEAD -> superhero-feature) Add superhero public api
279be79 Add superhero fighting skills
3e11d0d Add superhero persistence layer
71f3681 (master) Latest master state
a51d619 Some new master commits
...

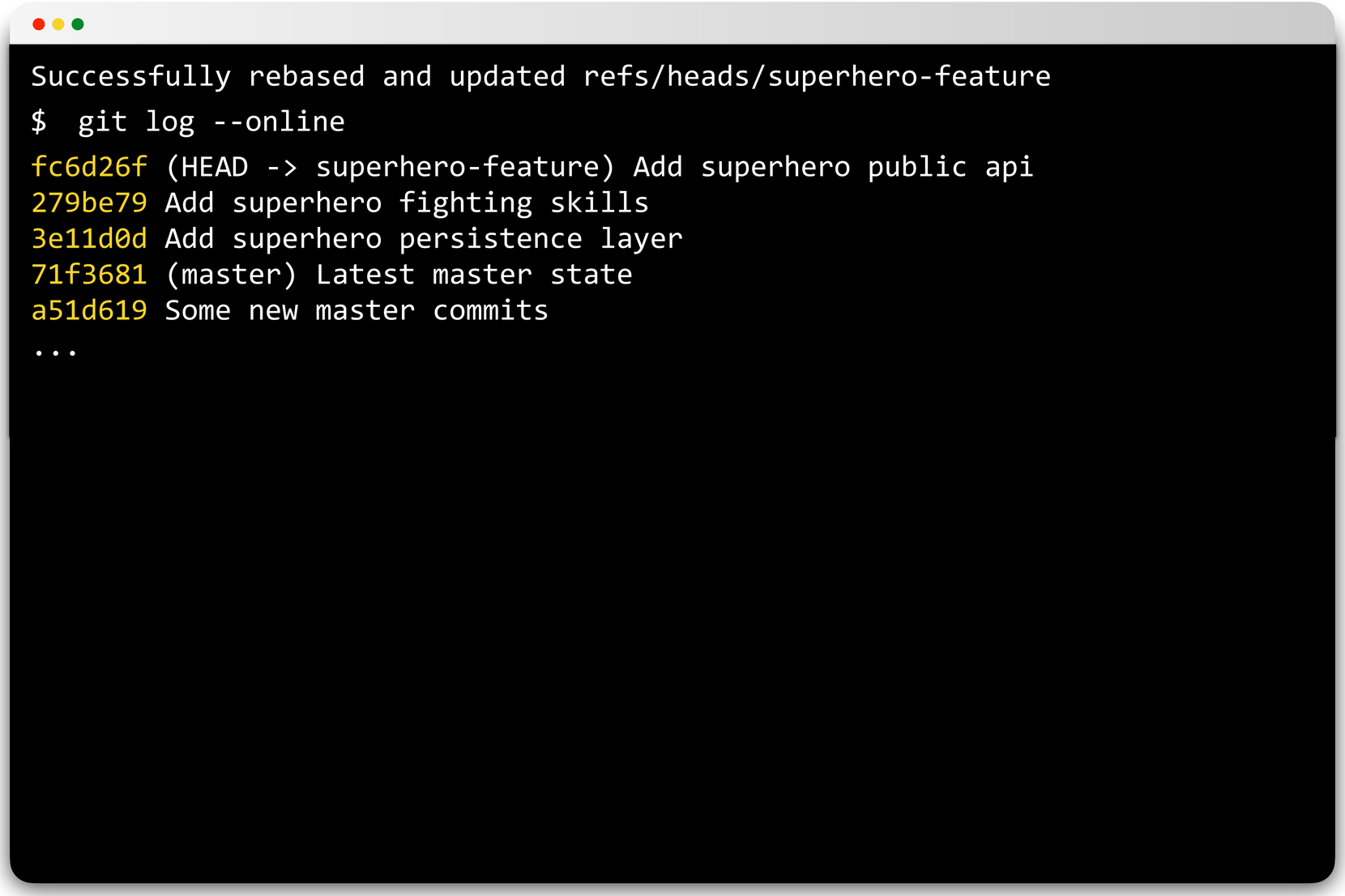
GIT FIXUP


$ git commit --fixup=279be79
[test eb3da88] fixup! Add superhero fighting skills
1 file changed, 1 insertion(+)
$ git rebase --autosquash -i

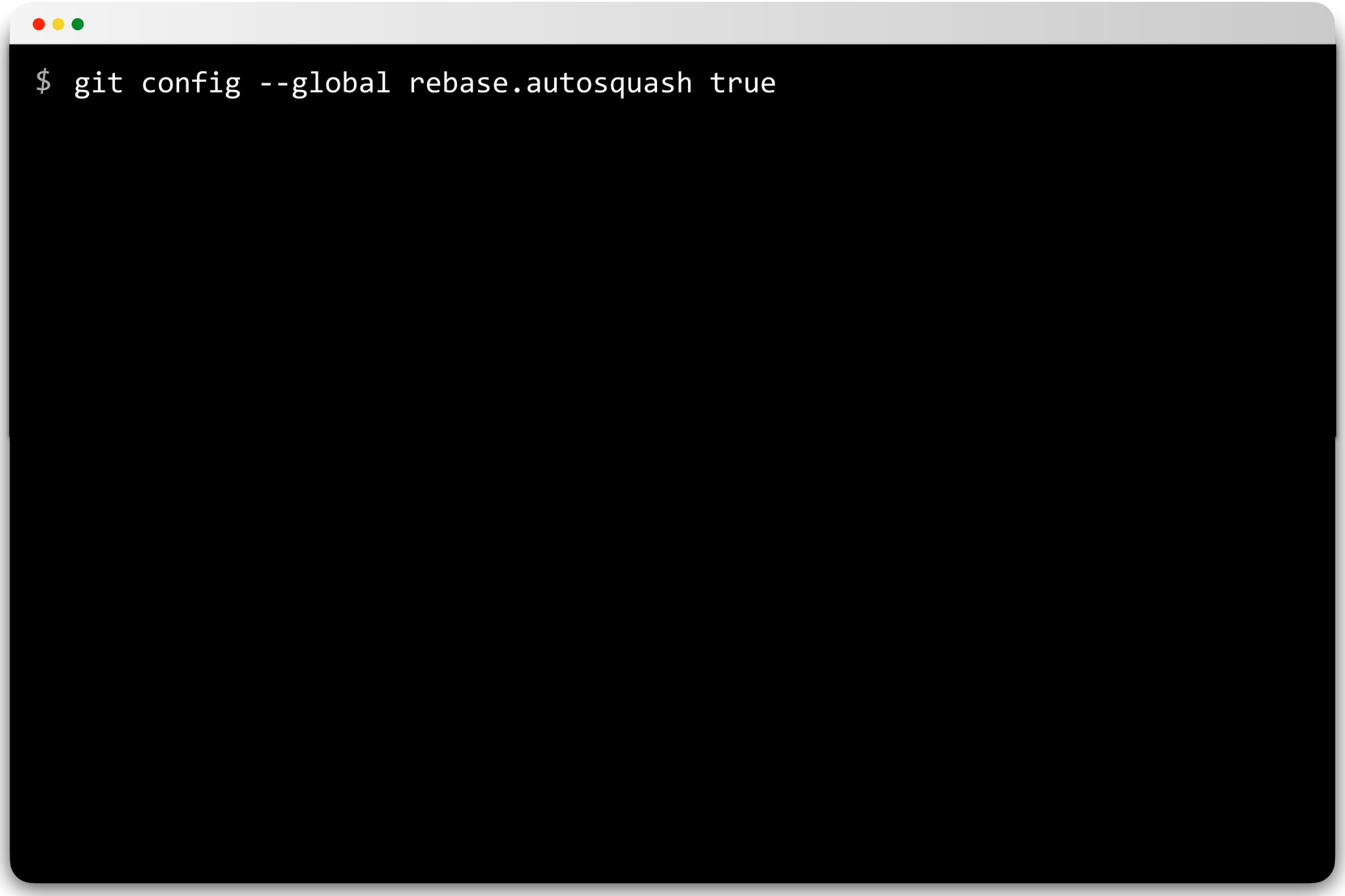
GIT FIXUP

```
pick 3e11d0d Add superhero persistence layer
pick 279be79 Add superhero fighting skills
fixup eb3da88 fixup! Add superhero fighting skills
pick fc6d26f Add superhero public api

# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup <commit> = like "squash", but discard this commit's log message
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .       create a merge commit using the original merge commit's
# .       message (or the oneline, if no original merge commit was
# .       specified). Use -c <commit> to reword the commit message.
```

GIT FIXUP


Successfully rebased and updated refs/heads/superhero-feature
$ git log --online
fc6d26f (HEAD -> superhero-feature) Add superhero public api
279be79 Add superhero fighting skills
3e11d0d Add superhero persistence layer
71f3681 (master) Latest master state
a51d619 Some new master commits
...

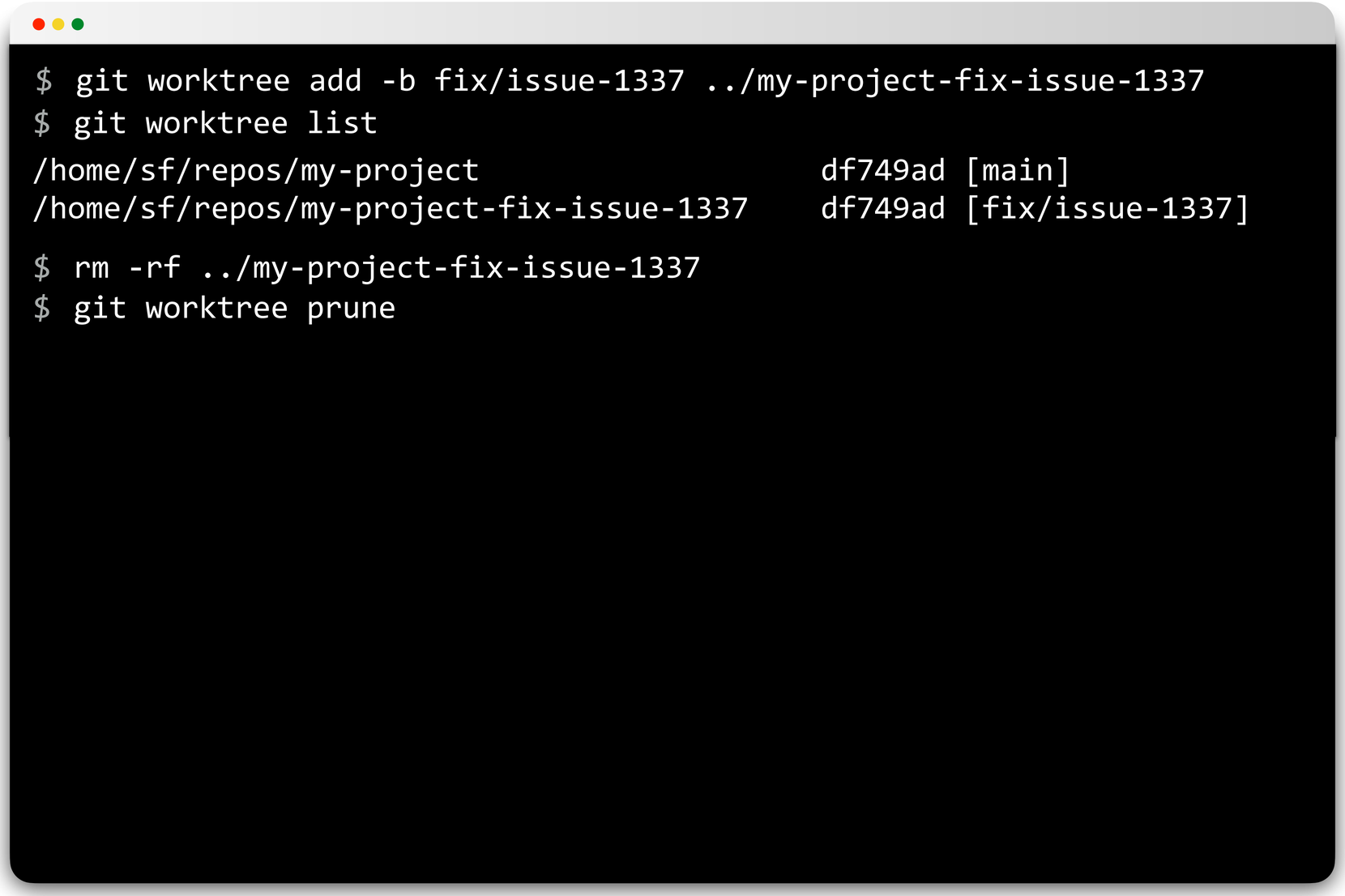
GIT AUTOSQUASH



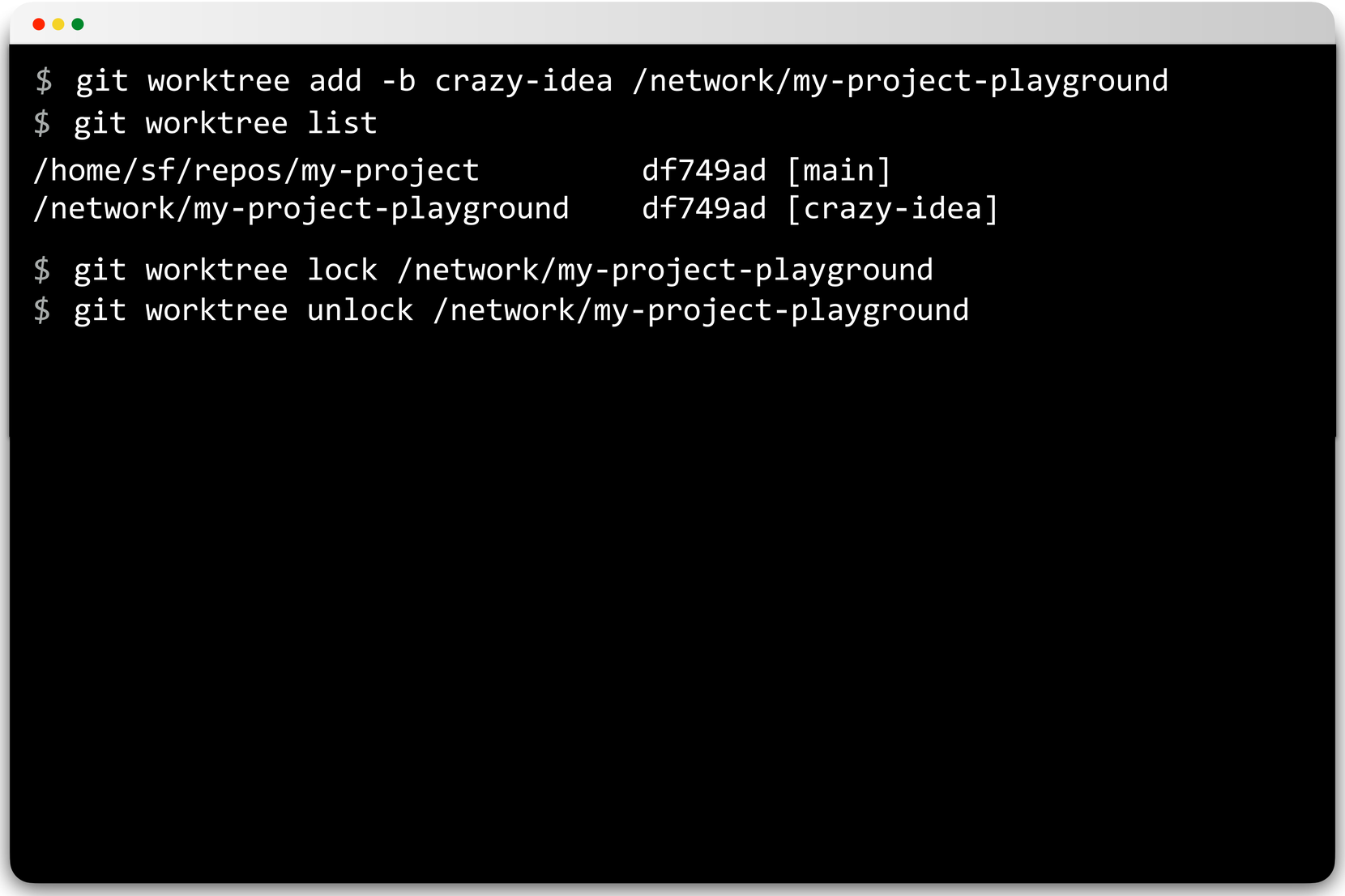
GIT ALIAS

```
git config --global alias.autosquash "-c sequence.editor=: rebase --autosquash --interactive"
```

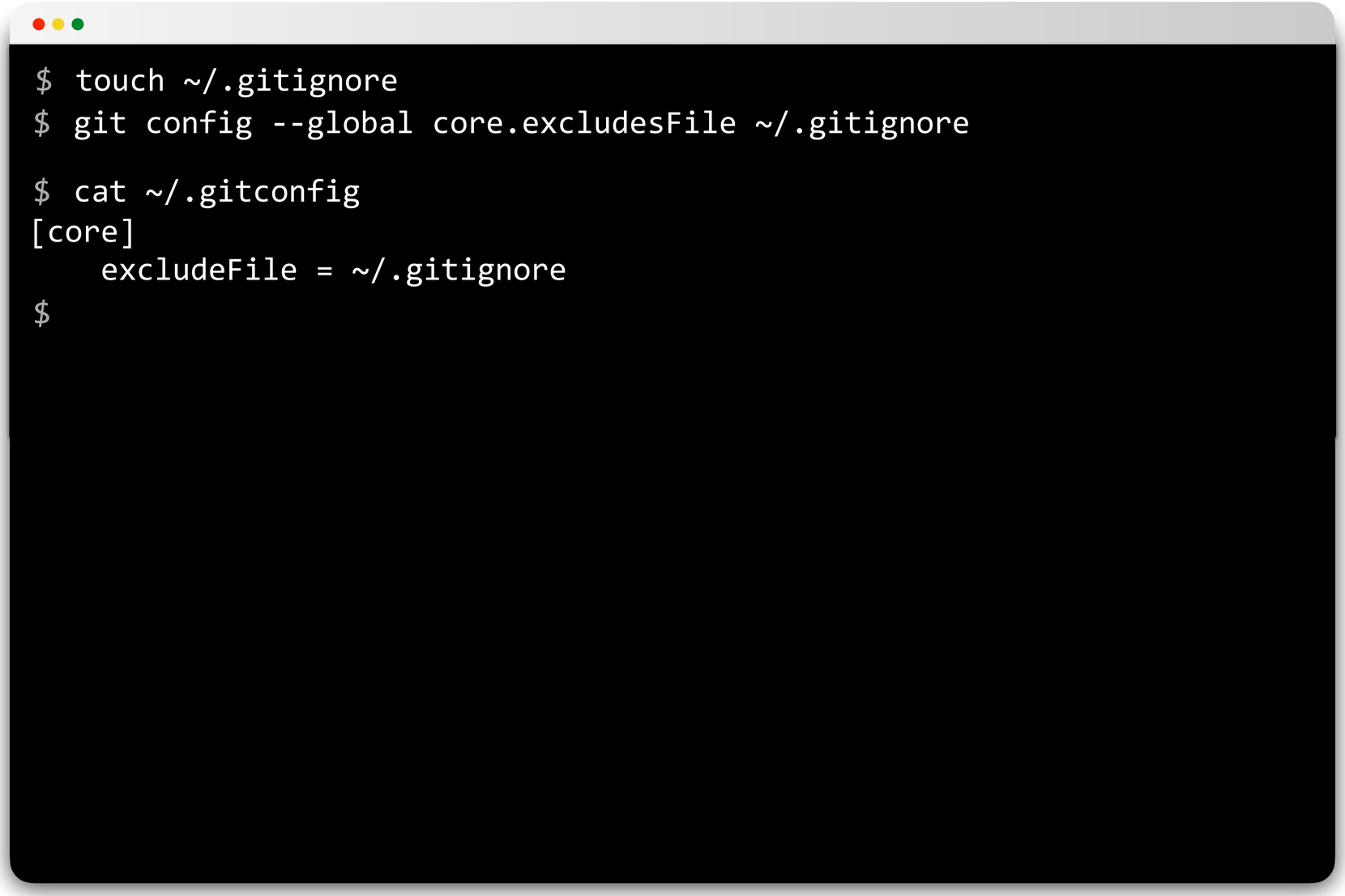
MULTI WORKTREE



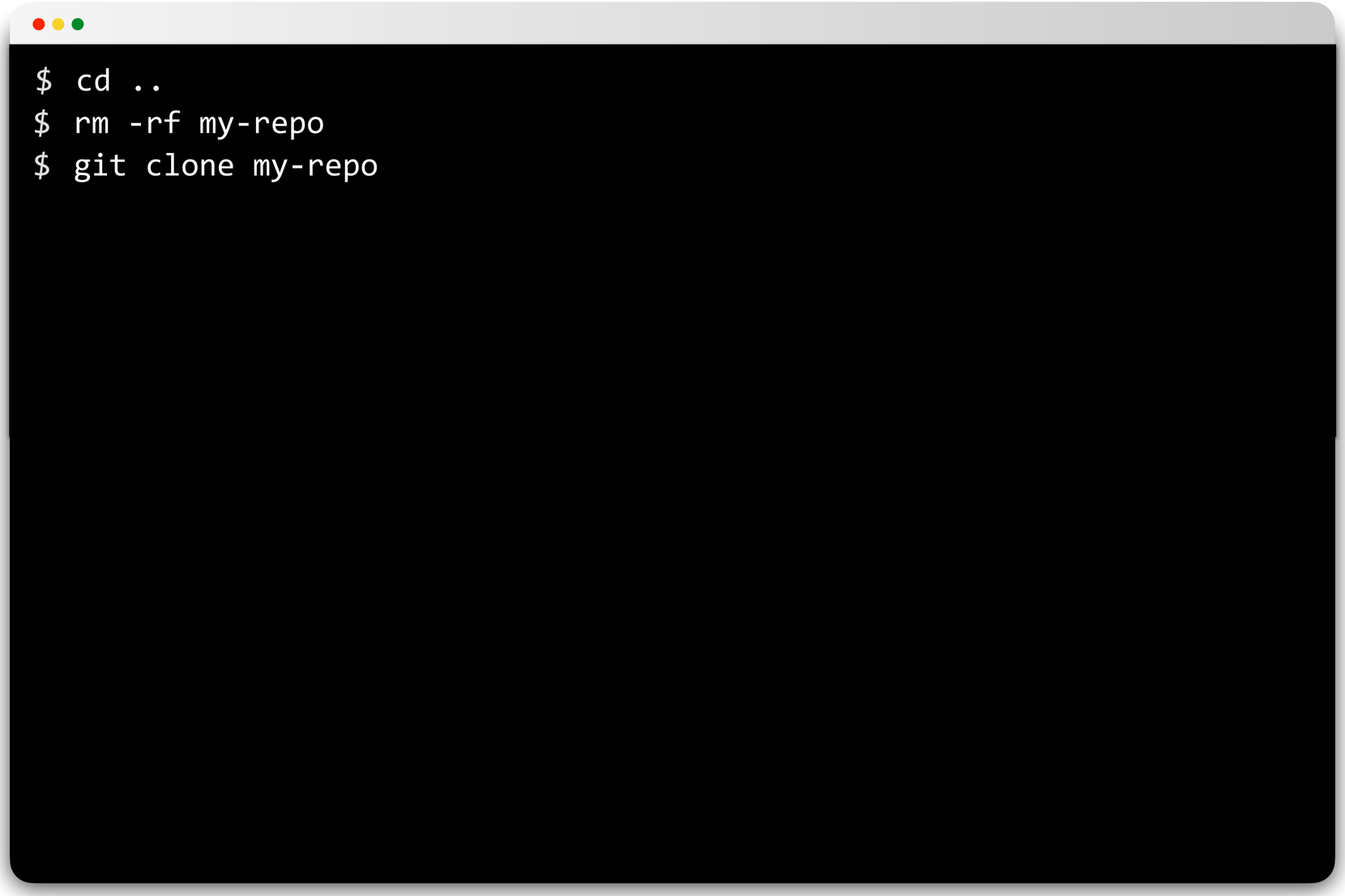
MULTI WORKTREE



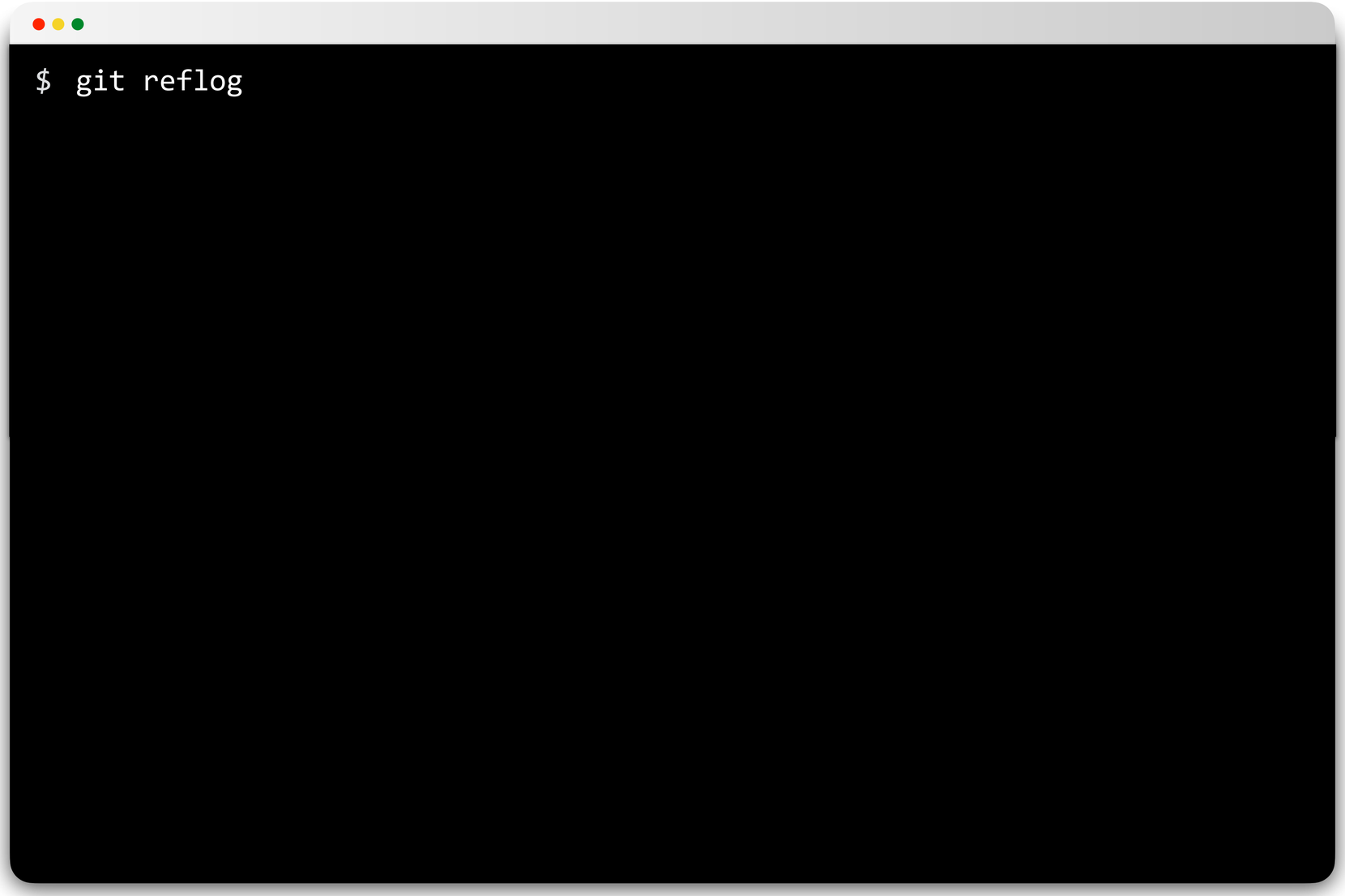
GLOBAL GITIGNORE



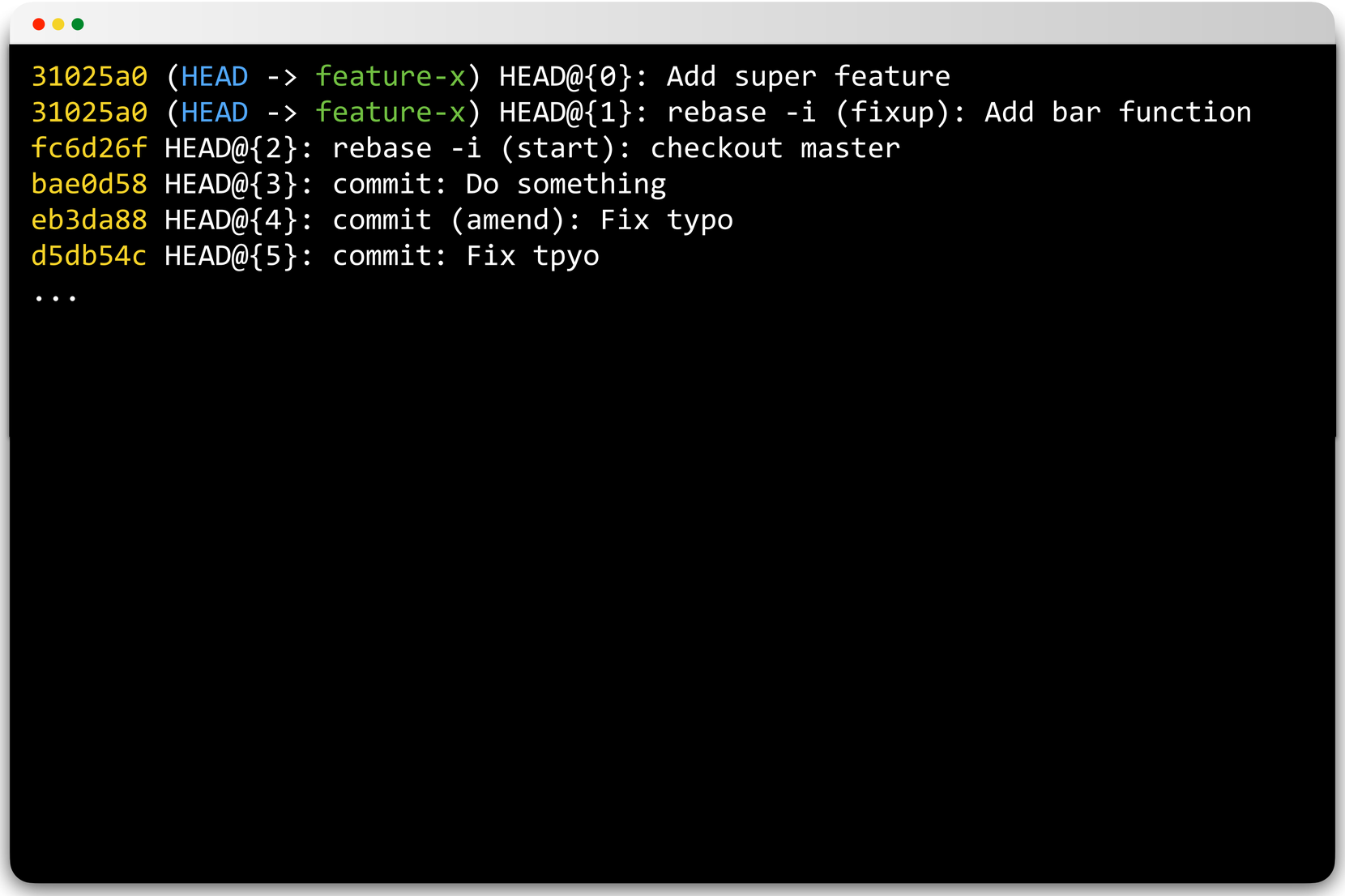
MISTAKES



GIT REFLOG



GIT REFLOG


31025a0 (HEAD -> feature-x) HEAD@{0}: Add super feature
31025a0 (HEAD -> feature-x) HEAD@{1}: rebase -i (fixup): Add bar function
fc6d26f HEAD@{2}: rebase -i (start): checkout master
bae0d58 HEAD@{3}: commit: Do something
eb3da88 HEAD@{4}: commit (amend): Fix typo
d5db54c HEAD@{5}: commit: Fix tpyo
...

GIT REFLOG

INVESTIGATE

GIT BISECT

```
$ git bisect start
$ git bisect bad HEAD
$ git bisect good d508d39
Bisecting: 3 revisions left to test after this (roughly 2 steps)
[fd3d3c4f366ba36673458dc56a3d20495cabcb18] Add more unit tests
$ git bisect bad
Bisecting: 0 revisions left to test after this (roughly 1 step)
[76ea3b61b2bba31e893f7aeb420556db83f61cc6] Make everything burn
$ git bisect bad
Bisecting: 0 revisions left to test after this (roughly 0 steps)
[8bb307ada4ce74ac69662406ca6fc8df16b1ac05] Optimize Messaging Queue
$ git bisect good
76ea3b61b2bba31e893f7aeb420556db83f61cc6 is the first bad commit
commit 76ea3b61b2bba31e893f7aeb420556db83f61cc6
Author: Sebastian Feldmann <sf@github.com>
Date:   Fri Mar 17 14:37:32 2024 +0100

    Make everything burn
$ git bisect reset
```

GIT BISECT

```
$ git bisect start
$ git bisect bad HEAD
$ git bisect good d508d39
Bisecting: 3 revisions left to test after this (roughly 2 steps)
[fd3d3c4f366ba36673458dc56a3d20495cabcb18] Add more unit tests
$ git bisect run unittests
fd3d3c4 - Unittests failed
76ea3b6 - Unittests failed
8bb307a - Unittests passed

git bisect successful

76ea3b61b2bba31e893f7aeb420556db83f61cc6 is the first bad commit
commit 76ea3b61b2bba31e893f7aeb420556db83f61cc6
Author: Sebastian Feldmann <sf@github.com>
Date:   Fri Mar 17 14:37:32 2024 +0100

    Make everything burn
$ git bisect reset
```

THANK YOU

❤️ FEEDBACK

sebastianfeldmann

@movetodevnull

phpc.social/@movetodevnull

TIPS & TRICKS

GIT ALIASES

Add via terminal or edit ~/.gitconfig

```
$ git config --global alias.unstage "reset HEAD"
$ git config --global alias.undo-commit "reset --soft"
$ git config --global alias.clean "reset --hard"
$ git config --global alias.b "branch"
$ git config --global alias.c "commit"
$ git config --global alias.p "pull"
```

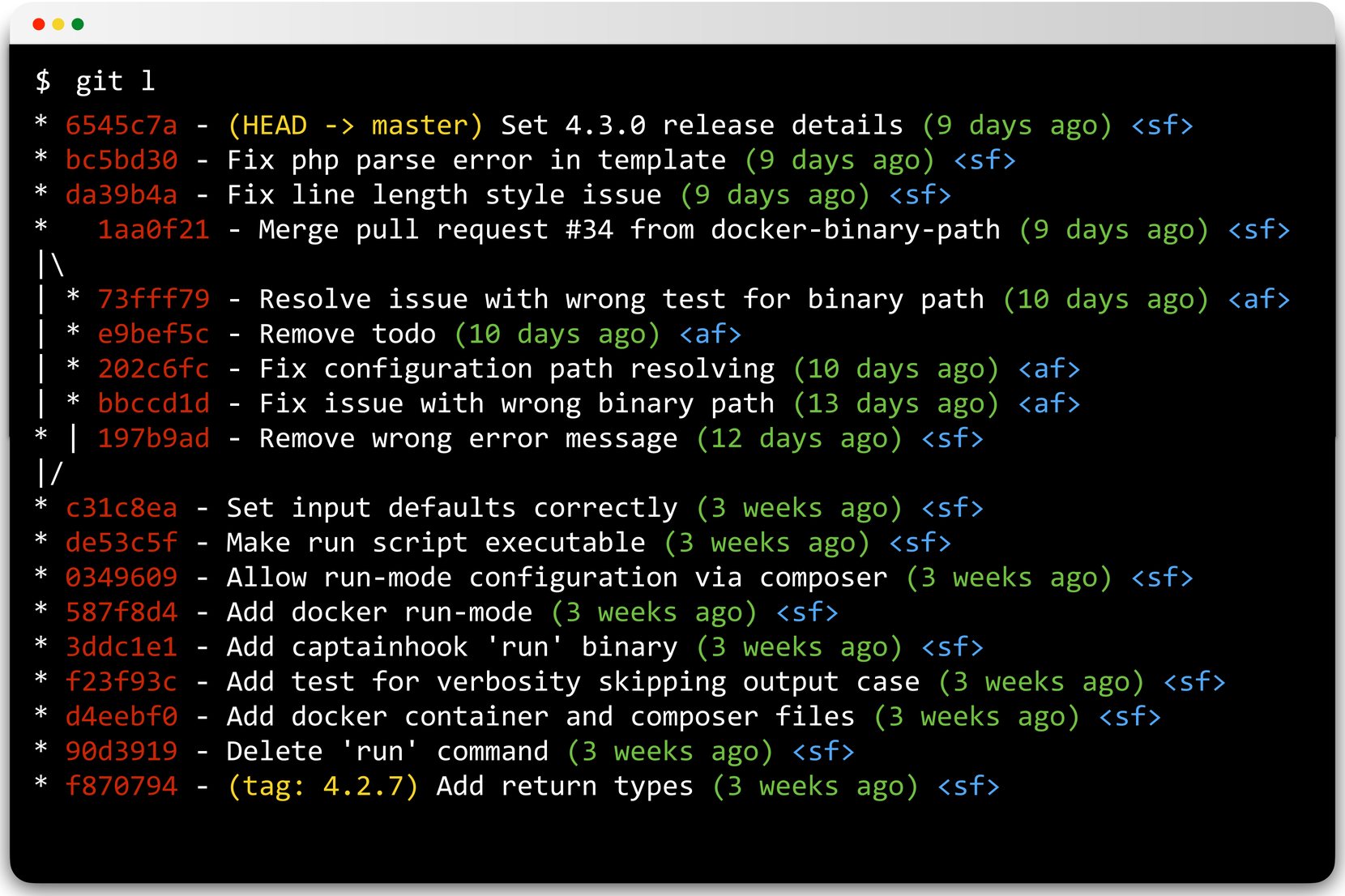
GIT LOG

`--author="Sebastian"`	Only show commits made by a certain author
`--name-only`	Only show names of files that changed
`--oneline`	Show commit data compressed to one line
`--graph`	Show dependency tree for all commits
`--reverse`	Show commits in reverse order (Oldest commit first)
`--after`	Show all commits that happened after certain date
`--before`	Show all commits that happened before certain data
`--stat`	Show all commits with some statistics

GIT LOG


$ git log --color --graph --pretty=format:'%Cred%h%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset' --abbrev-commit

GIT LOG


$ git l
* 6545c7a - (HEAD -> master) Set 4.3.0 release details (9 days ago) <sf>
* bc5bd30 - Fix php parse error in template (9 days ago) <sf>
* da39b4a - Fix line length style issue (9 days ago) <sf>
*   1aa0f21 - Merge pull request #34 from docker-binary-path (9 days ago) <sf>
|\
| * 73fff79 - Resolve issue with wrong test for binary path (10 days ago) <af>
| * e9bef5c - Remove todo (10 days ago) <af>
| * 202c6fc - Fix configuration path resolving (10 days ago) <af>
| * bbccd1d - Fix issue with wrong binary path (13 days ago) <af>
* | 197b9ad - Remove wrong error message (12 days ago) <sf>
|/
* c31c8ea - Set input defaults correctly (3 weeks ago) <sf>
* de53c5f - Make run script executable (3 weeks ago) <sf>
* 0349609 - Allow run-mode configuration via composer (3 weeks ago) <sf>
* 587f8d4 - Add docker run-mode (3 weeks ago) <sf>
* 3ddc1e1 - Add captainhook 'run' binary (3 weeks ago) <sf>
* f23f93c - Add test for verbosity skipping output case (3 weeks ago) <sf>
* d4eebf0 - Add docker container and composer files (3 weeks ago) <sf>
* 90d3919 - Delete 'run' command (3 weeks ago) <sf>
* f870794 - (tag: 4.2.7) Add return types (3 weeks ago) <sf>

GIT LOG

You can use the regular less command to search

```
/{{your-search-here}}
```

Use lower case `n` to navigate to the next occurrence and upper case `N` to the previous one.

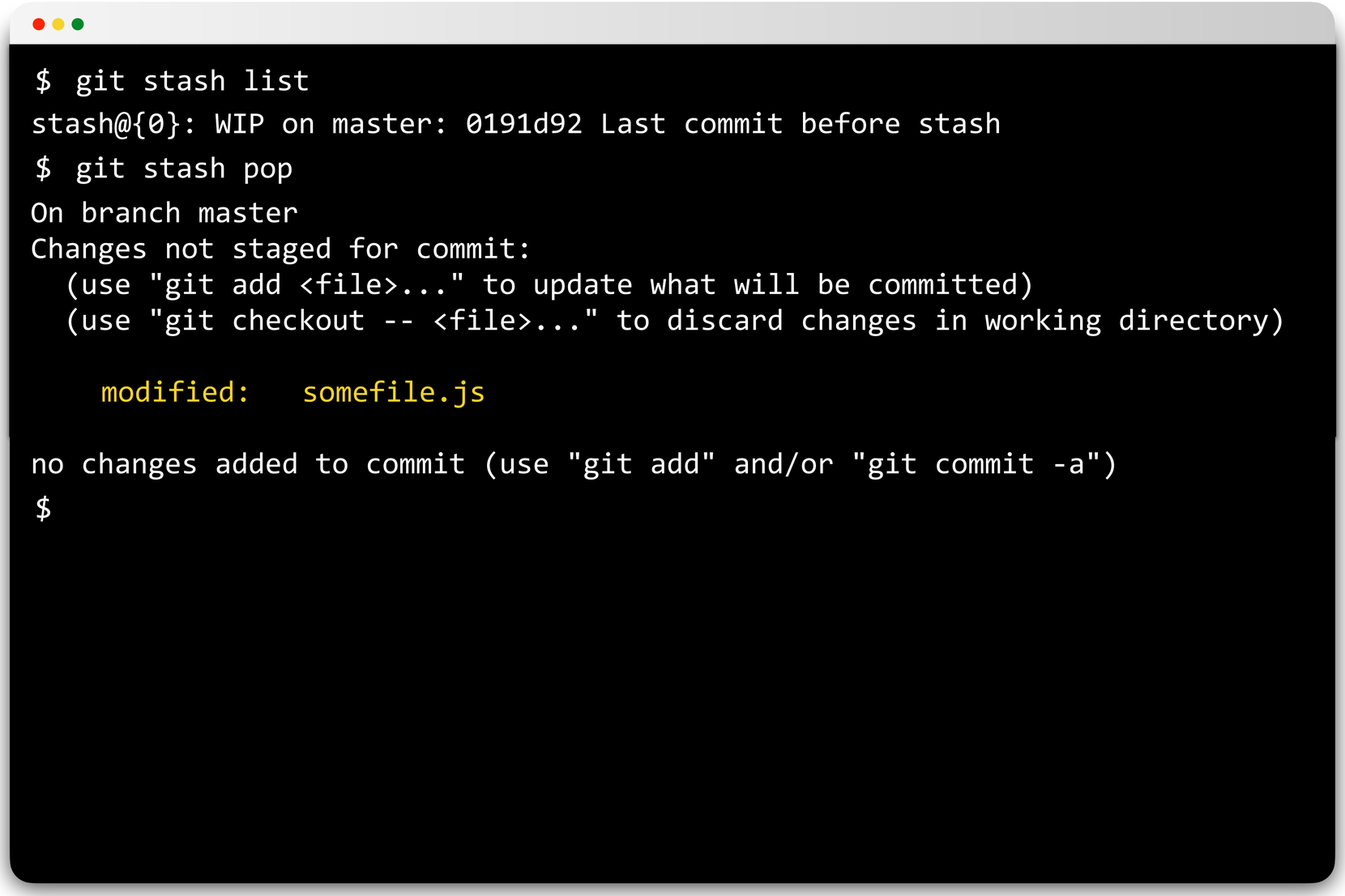
FORCE PUSH

```
$ git push --force-with-lease
```

GIT STASH

```
$ git status

On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   somefile.js

no changes added to commit (use "git add" and/or "git commit -a")

$ git stash
$ git status
On branch master

nothing to commit, working tree clean
$
```

GIT STASH


$ git stash list
stash@{0}: WIP on master: 0191d92 Last commit before stash
$ git stash pop
On branch master
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git checkout -- <file>..." to discard changes in working directory)
modified: somefile.js
no changes added to commit (use "git add" and/or "git commit -a")
$

GIT STASH

Apply latest stashed state and remove from list

```
$ git stash pop
```

Apply any stashed state and keep in list

```
$ git stash apply stash@{n}
```

Clean up the stash

```
$ git stash drop stash@{n}
$ git stash clear
```