OSSDev: Advanced Git

Mind your Git Manners

Irina Gulina
Senior Quality Engineer

Tomas Tomecek
Principal Software Engineer

Red Hat

Happy Birthday, Git!

Source: https://octodex.github.com/

Agenda

- Commit
- Push
- PR/MR submitting and review
- Merge
- Rebase
- Quiz

Git Commit

Commit content

- Don't: **Two and more changes in one commit**

  1e4faa0 Fix login timeout BZ, add logout step

- Do: **One commit = One logical change**

  1e4faa0 Fix login timeout BZ
  2r5asy8 Add logout step

Commit content

- Separate whitespace changes from code changes, especially unrelated.
  - Mixing those is a great way to introduce a bug and
  - Complicates code review

What is a commit message?

- Title/subject line
- Body

Commit message example

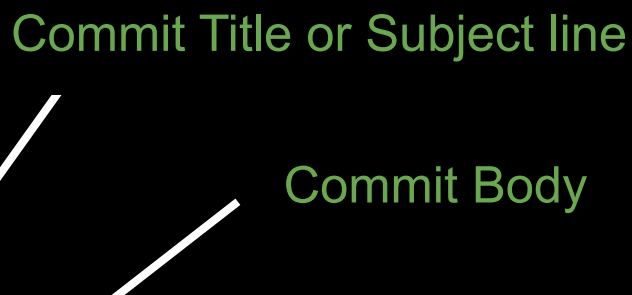
```
$ git log

commit <commit_id>
Author: <author_name> <author_email>
Date:   Mon Apr 2 15:10:03 2020 -0400

    Change how workers are represented

    * Don't serialize the 'gracefully_shutdown' field
    * Create a new 'missing' property and serialize it
    * In the status API, list both online and missing workers

    Requires PR: https://github.com/<project>/pull/921
    closes #354498
    https://bugzilla.redhat.com/show_bug.cgi?id=354498
```



Red Hat

What is a “bad” message?

```
$ git log --oneline -5 --author irina --before "Wed Apr 7 2021"

dcc2d35         address comments
b7aac30         fix issue #123
0b7a4e4         various docs fixes
1e4faa0         ui bug fix
fc3d081         readme  update
d21660dc        ToDo
0b7a4e4         Mix fixes and cleanups
5h3d28g         refactoring
```

What is a “bad” message?

```
$ git log --oneline -5 --author irina --before "Wed Apr 7 2021"

dcc2d35     address comments          <- what comments?
b7aac30     fix issue #123            <- of what project?
0b7a4e4     various docs fixes        <- what docs? why?
1e4faa0     ui bug fix                <- what was the bug?
fc3d081     readme  update            <- why?
d21660dc    ToDo                      <- 😩😩😩😩
0b7a4e4     Mix fixes and cleanups    <-🙈 🙊 🙉
5h3d28g     refactoring               <- 😭
```

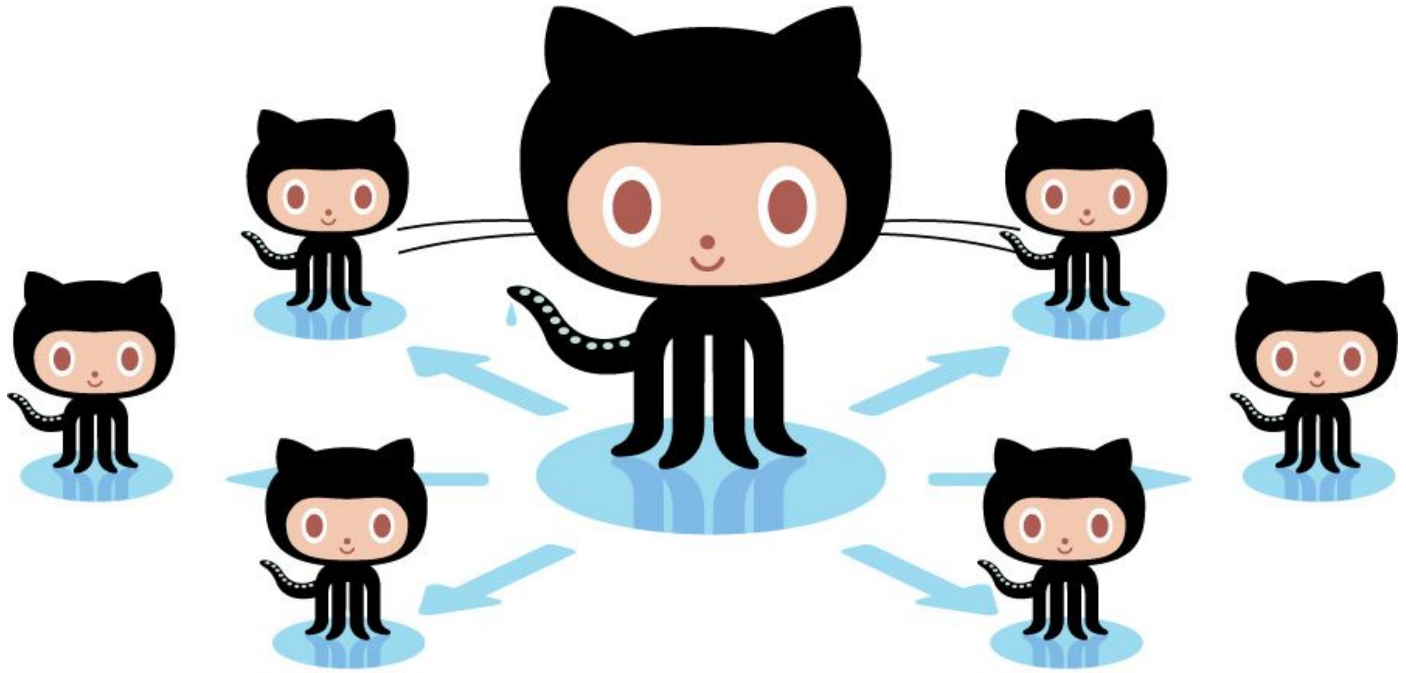
Uninformative, look-elsewhere commit messages (titles)

Usage of a commit title

- git log --pretty=oneline
- git rebase --interactive
- merge.summary
- git shortlog
- git format-patch, git send-email, ...
- reflogs
- GUI tools for committing and browsing
- GitHub, SourceForge, Bitbucket, GitLab, ... service

Poor quality code can be refactored.
A terrible commit message lasts **forever**.

For whom do you write commit messages?



Why should I write ‘good’ commit messages?

- To help to understand the code change
  - What has been changed?
  - Why is that change necessary?
- To speed up the reviewing process
- To help to locate a bug
- To write a good release note or script it

What constitutes a good commit message?

- git commit -m “Fix login timeout bug”
- git commit or git commit --verbose

```
Redirect user to the requested page after login

https://link/to/issue/tracker
```

What constitutes a good commit message?

- Capital letter, 50/72, no punctuation in the end

```
$ git commit
A brief summary of the commit

A paragraph describing what changed and its impact.
```

What constitutes a good commit message?

- Present Tense and Imperative Mood

```
cf31d12 Adds login unit tests
7a9kj4f Fixed login unit tests
101q2wd Update login unit tests
1b7hn61 Removing login unit test
```

"If accepted, this commit will *<your commit message goes here>*."

Ticket number in commit messages

- Ticketing system != git log
  - “TICKET-123456 add missing params to class”
  - “Add missing meta fields to response”
- ❑ Takes space in 50 chars limit title
- ❑ Look-elsewhere for details message, I’m lazy
- ❑ May be not available for interested user or reviewer (permissions, outage)

What constitutes a good commit message?

- Clear Title – What is commit about?
- Present Tense and Imperative Mood
- No punctuation in a title
- Clear Body - What and why is it needed/changed vs how?
- 50/72
- Reference to an issue in a body message
- Follow the commit convention defined by the team

Git Push

IF YOU DO FORCE PUSH...

May the force stay with you.

Git push --force trap

- It’s ok to force push to your local branch
- It’s ok to force push to your (unmerged/open) PR/MR
- It’s not ok to force push to a public branch

Git push --force consequences

Git push --force consequences

- Lost data
- Altered history
- Not happy colleagues
- Lost karma points

How to avoid unwanted force push

How to avoid unwanted force push

- Protect important branches
- Backup
- Use git checkout -b
- Use --force-with-lease, carefully
- Use PR revert

You have a great freedom...
to change your history **locally.**

Submitting a PR

Why do we use PR/MR workflow?

Why do we use PR/MR workflow?

- Share changes
- Get review and feedback
- Encourage quality

What constitutes a good PR/MR?

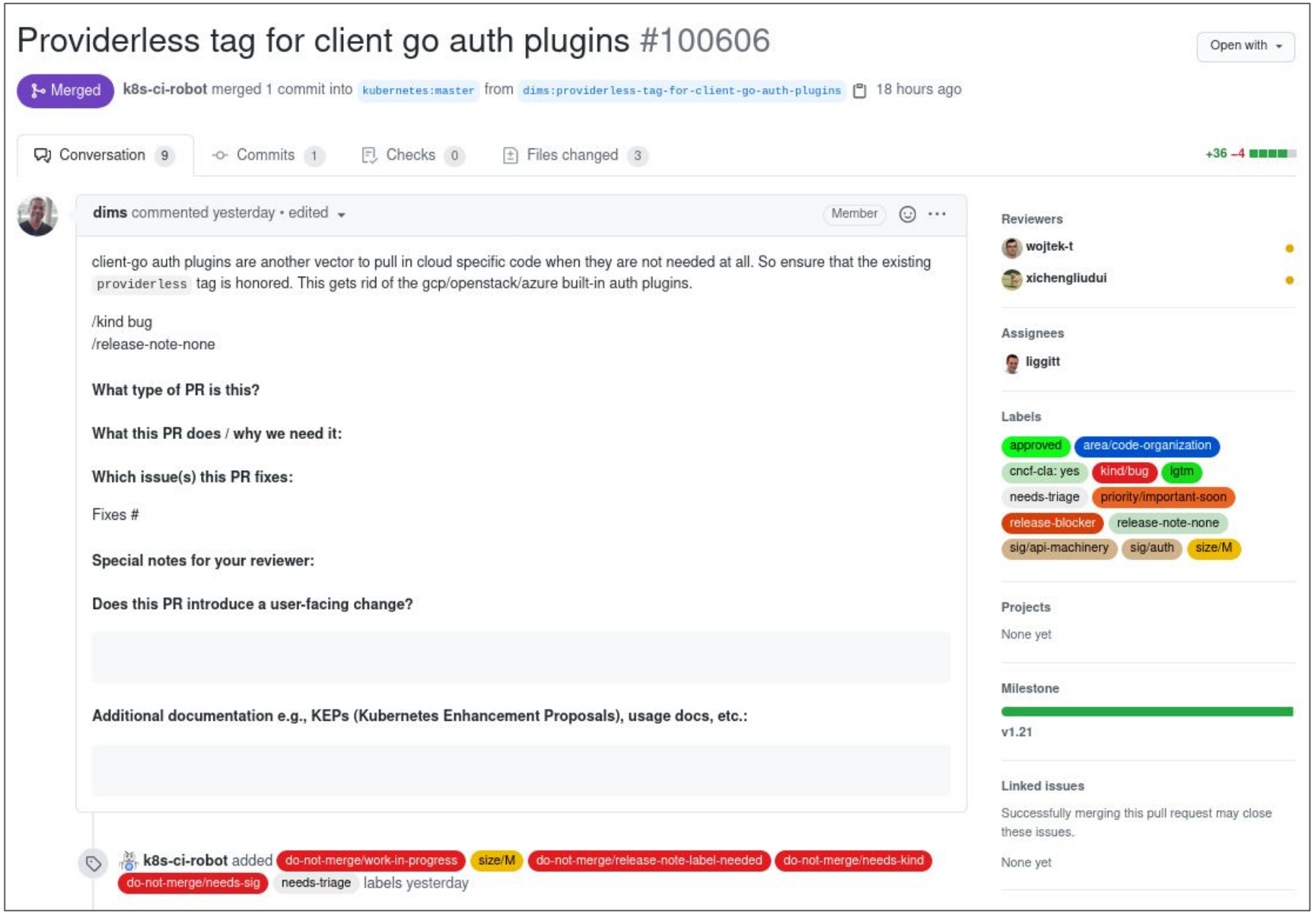
What constitutes a good PR/MR?

- Complete piece of work
- Adds value in some way
- Solid title and body
- Clear commit history
- Small
- Meets project's contribution guidelines

Contributors (before submitting a PR/MR)

- Follow the repo’s conventions
- Double check your code (and ToDos)
- Add docs
- Keep changes small
- Separate branch
- Be clear and specific
- Check your ego and be polite,

Open Source Contribution (GitHub)



Providerless tag for client go auth plugins #100606

Open with

Merged **k8s-ci-robot** merged 1 commit into `kubernetes:master` from `dims:providerless-tag-for-client-go-auth-plugins` 18 hours ago

Conversation 9 Commits 1 Checks 0 Files changed 3 +36 −4

dims commented yesterday • edited Member

client-go auth plugins are another vector to pull in cloud specific code when they are not needed at all. So ensure that the existing `providerless` tag is honored. This gets rid of the gcp/openstack/azure built-in auth plugins.

/kind bug
/release-note-none

What type of PR is this?

What this PR does / why we need it:

Which issue(s) this PR fixes:

Fixes #

Special notes for your reviewer:

Does this PR introduce a user-facing change?

Additional documentation e.g., KEPs (Kubernetes Enhancement Proposals), usage docs, etc.:

k8s-ci-robot added do-not-merge/work-in-progress size/M do-not-merge/release-note-label-needed do-not-merge/needs-kind do-not-merge/needs-sig needs-triage labels yesterday

Reviewers
wojtek-t
xichengliudui

Assignees
liggitt

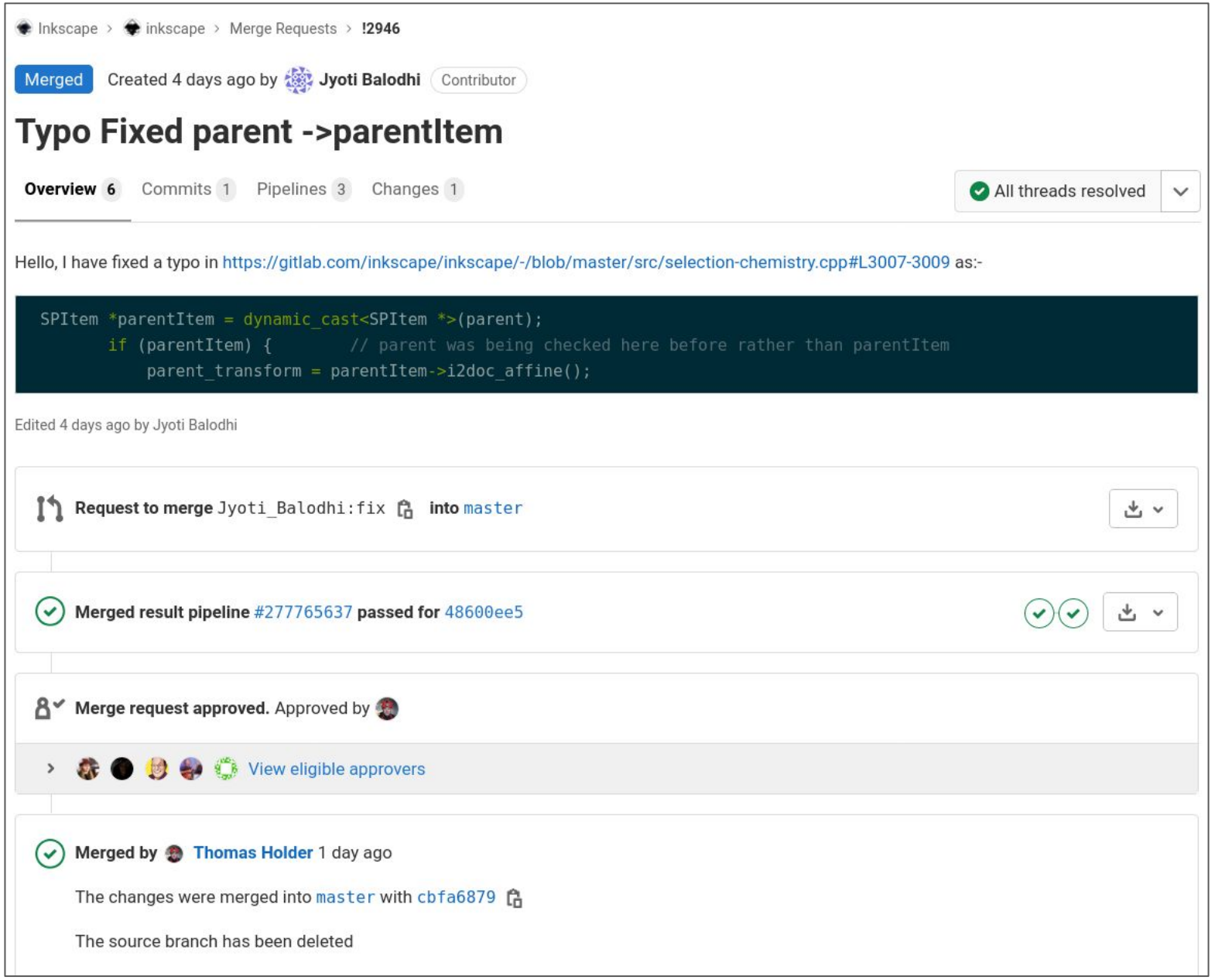
Labels
approved area/code-organization cncf-cla: yes kind/bug lgtm needs-triage priority/important-soon release-blocker release-note-none sig/api-machinery sig/auth size/M

Projects
None yet

Milestone
v1.21

Linked issues
Successfully merging this pull request may close these issues.
None yet

Open Source Contribution (GitLab)



Red Hat

Contributors (after submitting a PR/MR)

- Check your ego and be polite
  - **@username ping!**
  - **@username review please**
- Ensure your branch merge and tests pass
- Use --amend, --fixup or rebase -i
- Don’t merge your own PR

WIP PR/MR

- Don’t overuse WIP label
- Remove WIP label when ready
- “This is ready for review, please.”

Reviewing a PR

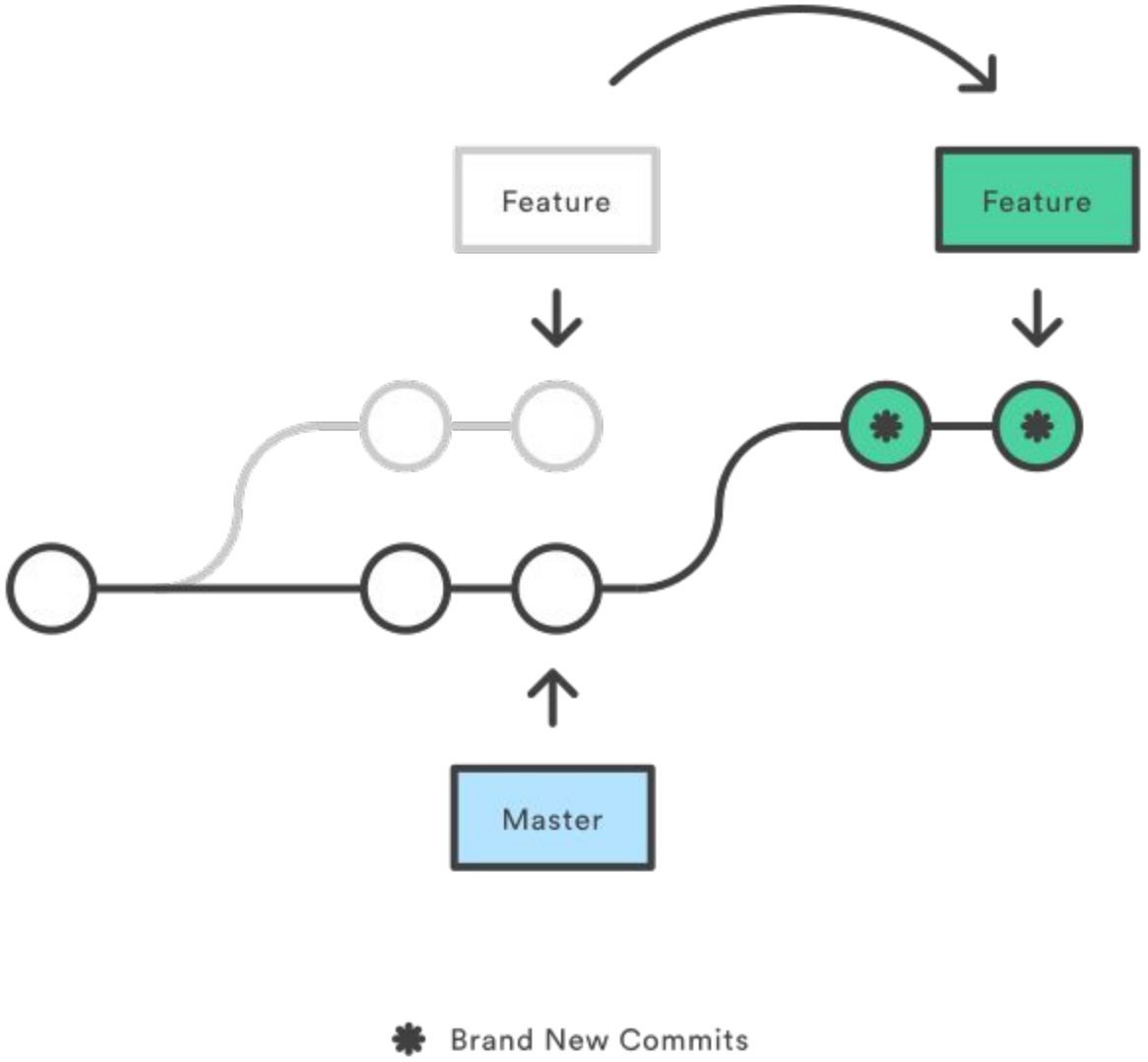
PR Reviewers

- Be kind and polite
  - **@username ping, error here!**
  - **@username s/foo/bar**
- Check commit history
- Don’t fix issues
- Ensure the branch can be merged
- CI Tests pass
- Don’t merge WIPs
- Squash
- Delete branch

Source: https://octodex.github.com/

Rebasing

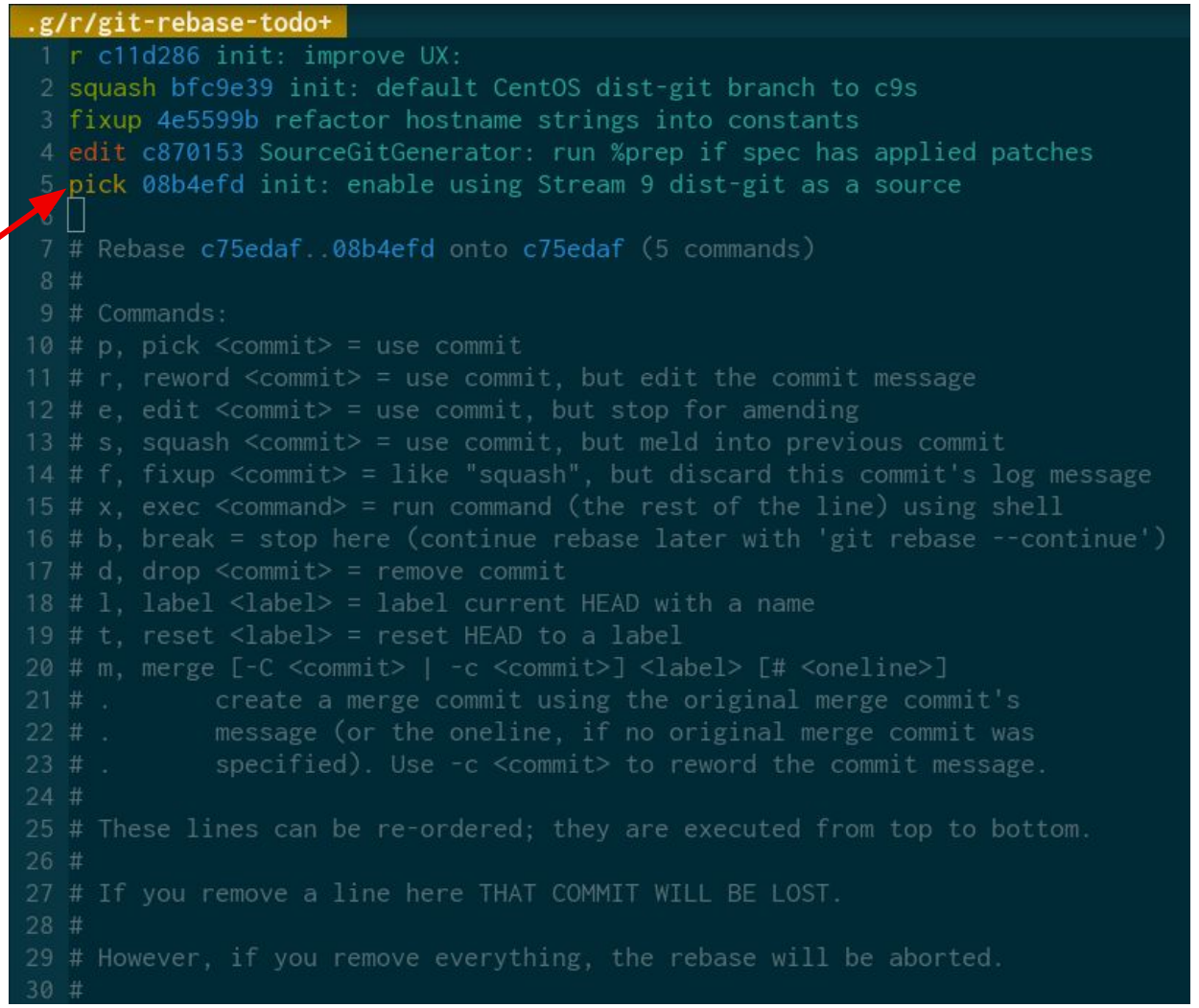
Rebase



Source: https://www.atlassian.com/git/tutorials/rewriting-history/git-rebase

Interactive rebase

HEAD



Source: Tomas' laptop

Red Hat

DEMO

QUIZ

QUESTIONS?

Irina Gulina, QE
igulina@redhat.com

Tomas Tomecek, Dev
ttomecek@redhat.com

Red Hat

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.


linkedin.com/company/red-hat


youtube.com/user/RedHatVideos


facebook.com/redhatinc


twitter.com/RedHat

Red Hat