PR Like a Pro with Advanced Techniques in Git

Jordan Leven

```
git rebase main

git rebase --continue

git rebase --abort
git push --force-with-lease
```

Nondestructive

```
git pull --rebase
```

Command Summary

Rebase on main	`git rebase`
Continue rebase	`git rebase --continue`
Abort rebase	`git rebase --abort`
Push after rebase	`git push --force-with-lease`
Pull after rebase	`git pull --rebase`

Command Summary

Rebase on main	`grb`
Continue rebase	`grbc`
Abort rebase	`grba`
Push after rebase	`gpf`
Pull after rebase	`gpr`

```
git rebase -i main
```

The branch to compare commits against.

```
git rebase -i main

git rebase -i HEAD~10
```

Number of commits in
current branch to show
(descending order)

Commands in Interactive Rebasing

pick (p)	Use existing commit	squash (s)	Combine this commit with previous commit, but preserve commit message of this commit
drop (d)	Remove commit from history	fixup (f)	Combine this commit with previous commit and do not preserve commit message of this commit
edit (e)	Rewind HEAD to commit and make changes	reword (r)	Reword commit message (no code changes)

Reordering Commits

When to use

- Related commits are not grouped together
- Commits cannot be individually checked out

```
7d39341 feat: Use my great feature
a16aa66 feat: Add my great feature
```

Pick (p)	Drop (d)	Edit (e)	Squash (s)	Fixup (f)	Reword (r)
Use existing commit	*Remove commit from history*	*Rewind HEAD to commit and make changes*	*Combine commit with previous and preserve commit message*	*Combine commit with previous and disregard commit message*	*Reword commit message (no code changes)*

Reordering Commits

How to use

- Re-order your commits in the preferred ascending order
- You may encounter merge conflicts
- Can be combined with all other commands

Pick (p)	Drop (d)	Edit (e)	Squash (s)	Fixup (f)	Reword (r)
Use existing commit	*Remove commit from history*	*Rewind HEAD to commit and make changes*	*Combine commit with previous and preserve commit message*	*Combine commit with previous and disregard commit message*	*Reword commit message (no code changes)*

Dropping Commits

When to use

- Made an accidental commit
- Delete a temporary commit

```
7d39341 drop: Attempt rebuild
```

Pick (p)	Drop (d)	Edit (e)	Squash (s)	Fixup (f)	Reword (r)
Use existing commit	*Remove commit from history*	*Rewind HEAD to commit and make changes*	*Combine commit with previous and preserve commit message*	*Combine commit with previous and disregard commit message*	*Reword commit message (no code changes)*

Dropping Commits

How to use

- Replace `pick` with `drop` or `d`
- Remove the line entirely
- Gone, but not forgotten

Pick (p)	**Drop (d)**	Edit (e)	Squash (s)	Fixup (f)	Reword (r)
Use existing commit	*Remove commit from history*	*Rewind HEAD to commit and make changes*	*Combine commit with previous and preserve commit message*	*Combine commit with previous and disregard commit message*	*Reword commit message (no code changes)*

Editing Commits

When to use

- A single commit encompasses too many ideas
- You made a mistake in a previous commit

```
a77779f feat: Add my great feature
7d39341 feat: Add missing part of feature
```

Pick (p)	Drop (d)	**Edit (e)**	Squash (s)	Fixup (f)	Reword (r)
Use existing commit	*Remove commit from history*	*Rewind HEAD to commit and make changes*	*Combine commit with previous and preserve commit message*	*Combine commit with previous and disregard commit message*	*Reword commit message (no code changes)*

Editing Commits

How to use

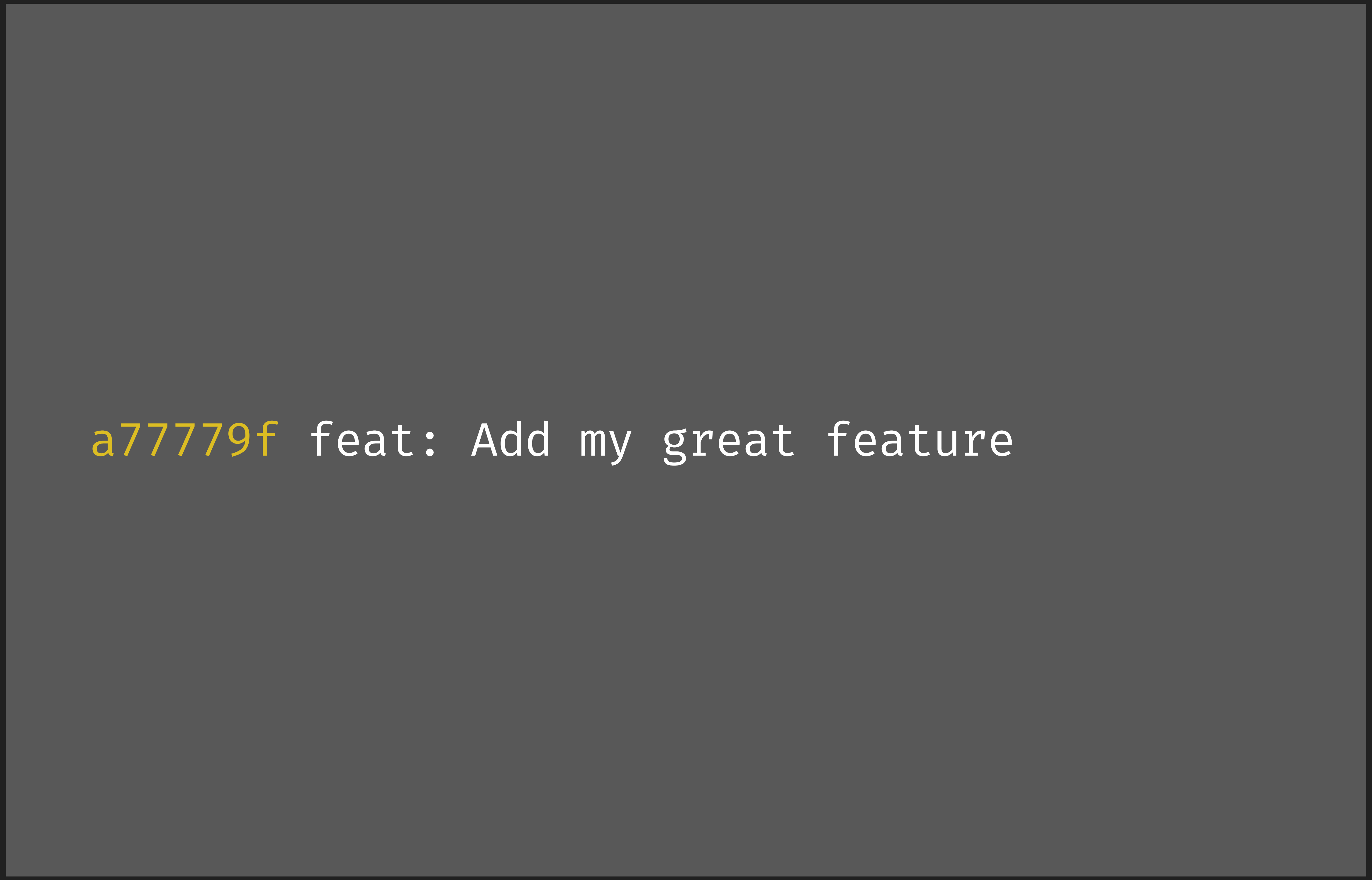
- Replace `pick` with `edit` or `e`
- Stage your changes (`ga -p`), and continue the rebase
- You may encounter merge conflicts

Pick (p)	Drop (d)	Edit (e)	Squash (s)	Fixup (f)	Reword (r)
Use existing commit	*Remove commit from history*	*Rewind HEAD to commit and make changes*	*Combine commit with previous and preserve commit message*	*Combine commit with previous and disregard commit message*	*Reword commit message (no code changes)*

Squashing and Fixing Up Commits

When to use

- Two commits express a singular idea
- You forgot to stage a change to your previous commit
- You're preparing to merge a PR



```
a77779f feat: Add my great feature
```

Pick (p)	Drop (d)	Edit (e)	Squash (s)	Fixup (f)	Reword (r)
Use existing commit	*Remove commit from history*	*Rewind HEAD to commit and make changes*	*Combine commit with previous and preserve commit message*	*Combine commit with previous and disregard commit message*	*Reword commit message (no code changes)*

Squashing and Fixing Up Commits

How to use

- Replace `pick` with `squash` or `s`
- Replace `pick` with `fixup` or `f`
- You're not modifying history — you're compacting it

Pick (p)	Drop (d)	Edit (e)	Squash (s)	Fixup (f)	Reword (r)
Use existing commit	*Remove commit from history*	*Rewind HEAD to commit and make changes*	*Combine commit with previous and preserve commit message*	*Combine commit with previous and disregard commit message*	*Reword commit message (no code changes)*

Rewording Commits

When to use

- You mkae a typo
- You forgot important details

```
a77779f feat: Do something
```

Pick (p)	Drop (d)	Edit (e)	Squash (s)	Fixup (f)	**Reword (r)**
Use existing commit	*Remove commit from history*	*Rewind HEAD to commit and make changes*	*Combine commit with previous and preserve commit message*	*Combine commit with previous and disregard commit message*	*Reword commit message (no code changes)*

Rewording Commits

How to use

- Replace `pick` with `reword` or `r`
- No code changes, but the SHAs will change

Pick (p)	Drop (d)	Edit (e)	Squash (s)	Fixup (f)	Reword (r)
Use existing commit	*Remove commit from history*	*Rewind HEAD to commit and make changes*	*Combine commit with previous and preserve commit message*	*Combine commit with previous and disregard commit message*	*Reword commit message (no code changes)*

github.com/jordanleven/advanced-techniques-in-git

Words of Advice

When using interactive rebasing, it’s easier to combine two commits than to split one.

```
git config --global core.editor "code --wait"

git config --global core.editor "goland --wait"

git config --global core.editor "idea --wait"
git reflog
```

Git reference log

SPECIAL THANKS TO ALL OUR AWESOME CAMP SPONSORS!

TruStage™

HEARTLAND BUSINESS SYSTEMS

Intellibus

aws

Unspecified SOFTWARE CO

Architect.io

ionic

CAMUNDA

alexa

Google Cloud

CORE BTS

GrapeCity®

Join Us Next Year!

WI 24

THAT®

CONFERENCE

JULY 29TH - AUG. 1ST

Session Feedback

PR Like a Pro with Advanced Techniques in Git

Jordan Leven

linkedin.com/in/jordanleven