Advanced Git

Beyond basic backup

Jerry Chen

Agenda

- Recap of last time
- Useful daily tools
- More on branches
- Changing history
- GitHub collaboration tools

The Basics

- Git — a **version control system** that gives you control
  - Alternative VCS exist, but have various issues
- GitHub — a cloud service that hosts your Git repos

File Home Insert Page Layout References Mailings Re

Paste Cut Copy Format Painter

Clipboard

Calibri (Body) 11

Font

Document Recovery

Word has recovered the following files. Save the ones you wish to keep.

Available files

...w_v2.docx [Autosav...
Version created from the last ...
14:25 11 мая 2016 г.

_ v_v2.docx [Original]
Version created last time the ...
14:17 11 мая 2016 г.

-ready11.doc [Original]
Version created last time the ...
12:26 11 мая 2016 г.
Repaired

Manage versions

Drive keeps older versions of 'disc11_sol.pdf' for 30 days. Learn more

UPLOAD NEW VERSION

Current version disc11_sol.pdf
Nov 30, 12:20 PM JERRY CHEN

Version 4 disc11_sol.pdf
Nov 27, 7:25 PM JERRY CHEN

Version 3 disc11_sol.pdf
Nov 26, 9:57 PM JERRY CHEN

Version 2 disc11_sol.pdf
Nov 26, 9:41 PM JERRY CHEN

Version history

Only show named versions

Thursday

Past_Versions

Search

Past_Versions

Name	Date Modified	Size
Jerry Chen Resume v2.pdf	Jan 4, 2015 at 23:59	55 KB
Jerry Chen Resume v3	Jan 12, 2015 at 13:02	55 KB
Jerry Chen Resume v4.pdf	Jan 26, 2015 at 21:35	55 KB
Jerry-Chen-Resume-1_26_17.pdf	Aug 27, 2016 at 22:09	89 KB
Jerry-Chen-Resume-R-v1.pdf	Jul 31, 2015 at 16:53	67 KB
Jerry-Chen-Resume-R-v15.pdf	Oct 12, 2015 at 19:45	85 KB
Jerry-Chen-Resume-v5.pdf	Jan 26, 2015 at 21:35	55 KB
Jerry-Chen-Resume-v6.pdf	Mar 19, 2015 at 11:19	56 KB
Jerry-Chen-Resume-v7.pdf	Apr 16, 2015 at 14:47	55 KB
Jerry-Chen-Resume-v8.pdf	Apr 16, 2015 at 14:51	55 KB
Jerry-Chen-Resume-v9.pdf	Jul 5, 2015 at 15:44	57 KB
Jerry-Chen-Resume-v10.1.pdf	Jul 26, 2015 at 21:50	85 KB
Jerry-Chen-Resume-v10.pdf	Jul 6, 2015 at 18:16	57 KB
Jerry-Chen-Resume-v11.pdf	Jul 28, 2015 at 21:10	67 KB
Jerry-Chen-Resume-v12.pdf	Jul 30, 2015 at 14:56	67 KB
Jerry-Chen-Resume-v13.pdf	Sep 4, 2015 at 18:13	84 KB

19 items, 122.67 GB available

Getting Started

- `git init` creates a new Git repository
- `git clone` will download a remote repo
- Create your commits/branches locally
- `git push` and `git pull` to stay in touch w/ remote

Other Commands

- `add, commit`
- `log`
- `branch`
- `merge`
- `checkout, reset`
- `diff, show, grep`
- `master~1, HEAD^2`

There's a lot more to learn

Cherry-pick

Interactive rebase

Rebasing a branch

Clean

Bisect

Reflog

Creating patches

.gitignore

Tag

Revert

Stashing changes

Blame

Cherry-pick

Interactive rebase

Rebasing a branch

Clean

Bisect

Reflog

Creating patches

.gitignore

Tag

Revert

Stashing changes

Blame

Useful Tools

Daily Tools

Preview

- The `.gitignore` file
- Clean
- Stash
- Tags
- Fast-forward

`.gitignore`

- It might not make sense to track all files
- Examples (usually anything auto-generated)
  - LaTeX build products
  - PDFs (maybe)
  - Application binaries (prefer to build from source)
  - Vim `.swp` files (has happened to me before 😐)

`.gitignore`

Three (actually two) easy steps

1. Create a `.gitignore` file in the base of your directory
2. Add stuff you want to ignore
3. Done. Forget about `.gitignore` for the rest of your life

and carry on with your repository

Clean

- The `git clean` command will removes untracked files
- Different than `git checkout -- filename`
- This is dangerous! You will lose changes
- So dangerous, the command doesn't work (huh?)

Stash

Save work for later

- `git stash` changes the directory to match HEAD
- Store all the changes made
- Can apply the changes later (even on a different branch)
- Generally safe (unless you drop your changes)

Tags

Stickier branches

- Like branches, tags are used to name commits
- Tags do not move to follow new commits
- You can make tags the same name as branches
    - Don't do this
- Tags generally stick around forever, unlike branches

Let's get started

Please get your computer ready

- This part is best if you follow along on a computer
- You can also view the slides & notes later
- https://v.gd/lec_3

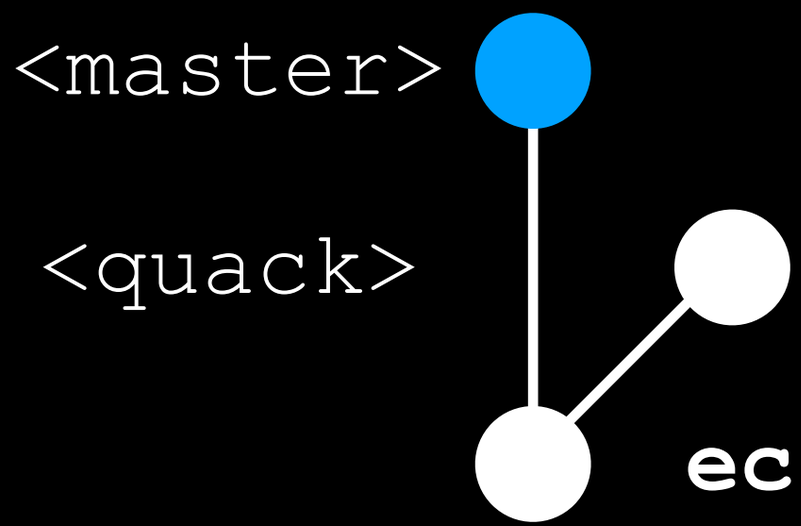
Demo

Daily Tools

Recap

- Started with a source LaTeX file
- Update `.gitignore` so we didn't track build products
- Removed junk files using `git clean`
- Made some changes, saved them with `git stash` and applied them laster
- Tag a v0.1 commit

Recap of Merging



b676eb2 Add more fox

19bccb7 Quacking

ec6d394 Add Trebek pangram

<master>

b676eb2 Add more fox

+fox

pangram1.txt

<quack>

19bccb7 Quacking

+quack

pangram1.txt

+quack

pangram3.txt

<master>

b676eb2 Add more fox

+fox

pangram1.txt

<quack>

19bccb7 Quacking

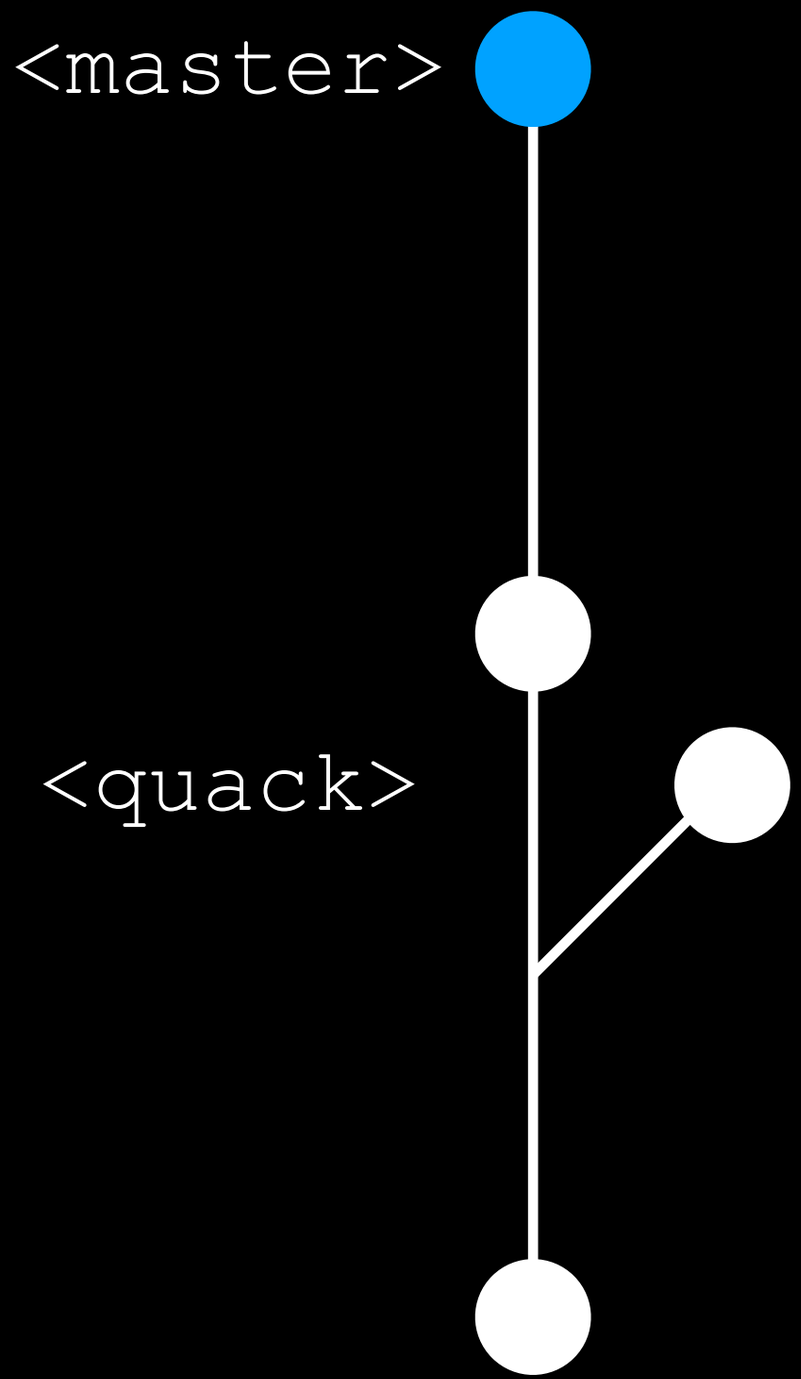
+quack

pangram1.txt

+quack

pangram3.txt

git merge quack

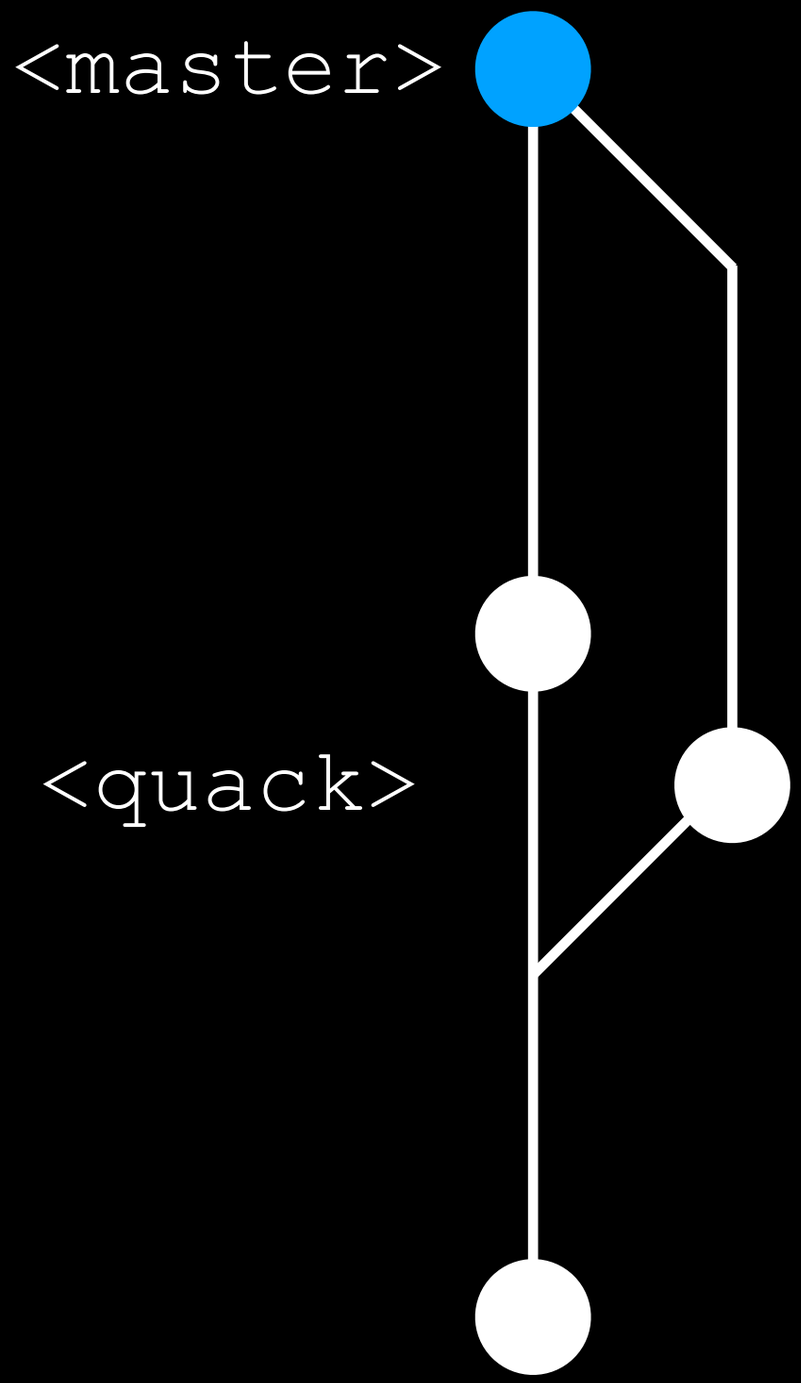


ff2ee03 Merge branch 'quack'

b676eb2 Add more fox

19bccb7 Quacking

git merge quack



ff2ee03 Merge branch 'quack'

+fox +quack	+quack
pangram1.txt	pangram3.txt

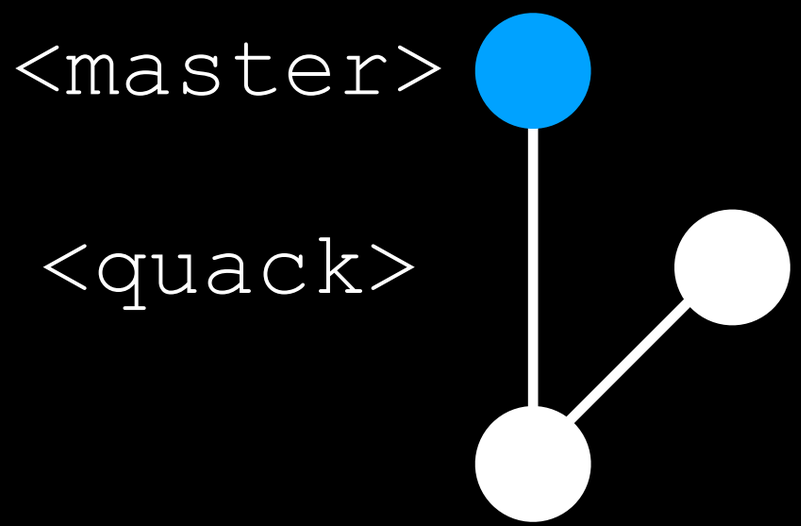
b676eb2 Add more fox

19bccb7 Quacking

Fast-Forward

Special case for merge

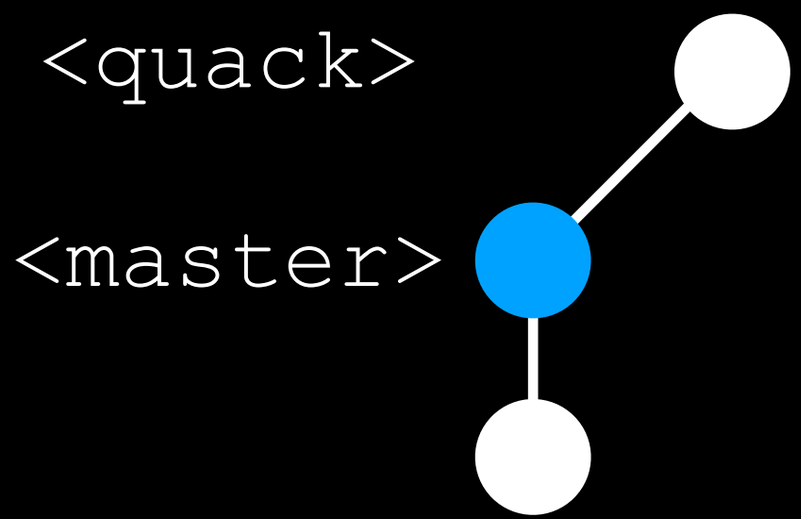
- In certain cases, a merge commit might not be necessary
- Heavily dependent on your commit order (for now)
- Git will automatically try to fast-forward



b676eb2 Add more fox

19bccb7 Quacking

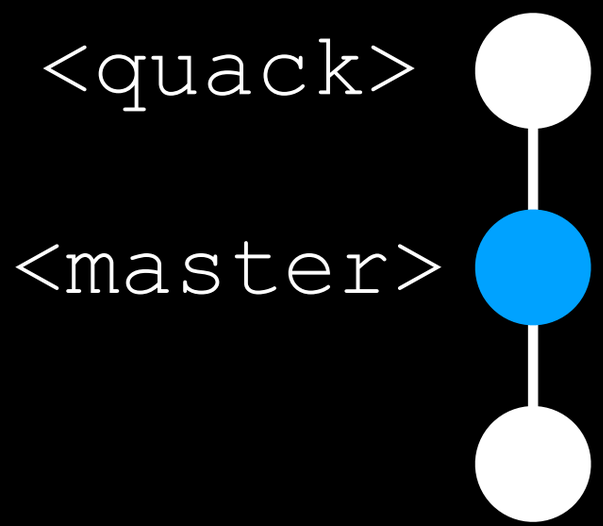
ec6d394 Add Trebek pangram



19bccb7 Quacking

b676eb2 Add more fox

ec6d394 Add Trebek pangram

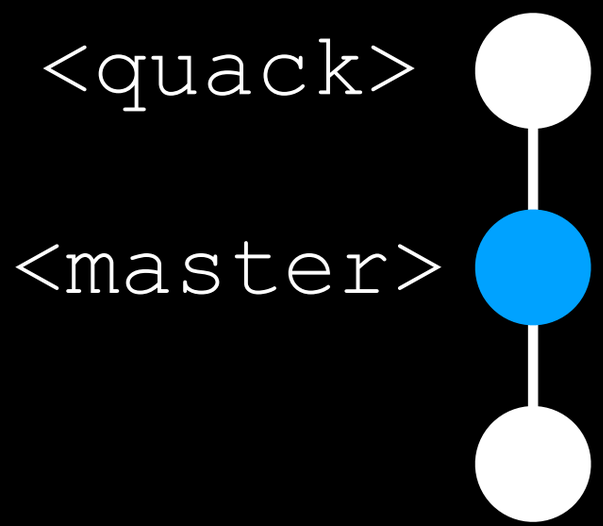


19bccb7 Quacking

b676eb2 Add more fox

ec6d394 Add Trebek pangram

git merge quack

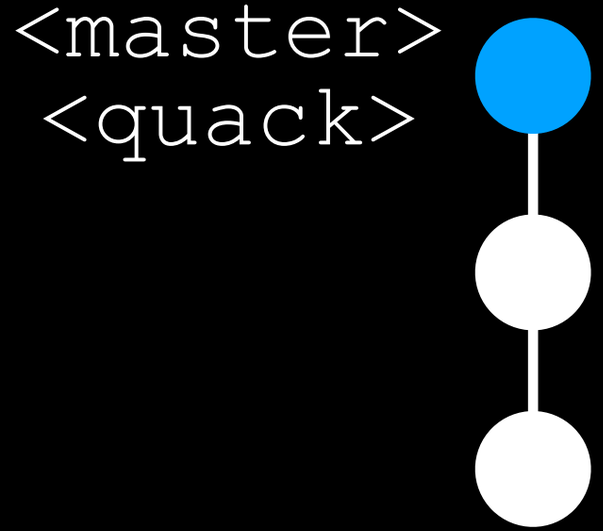


19bccb7 Quacking

b676eb2 Add more fox

ec6d394 Add Trebek pangram

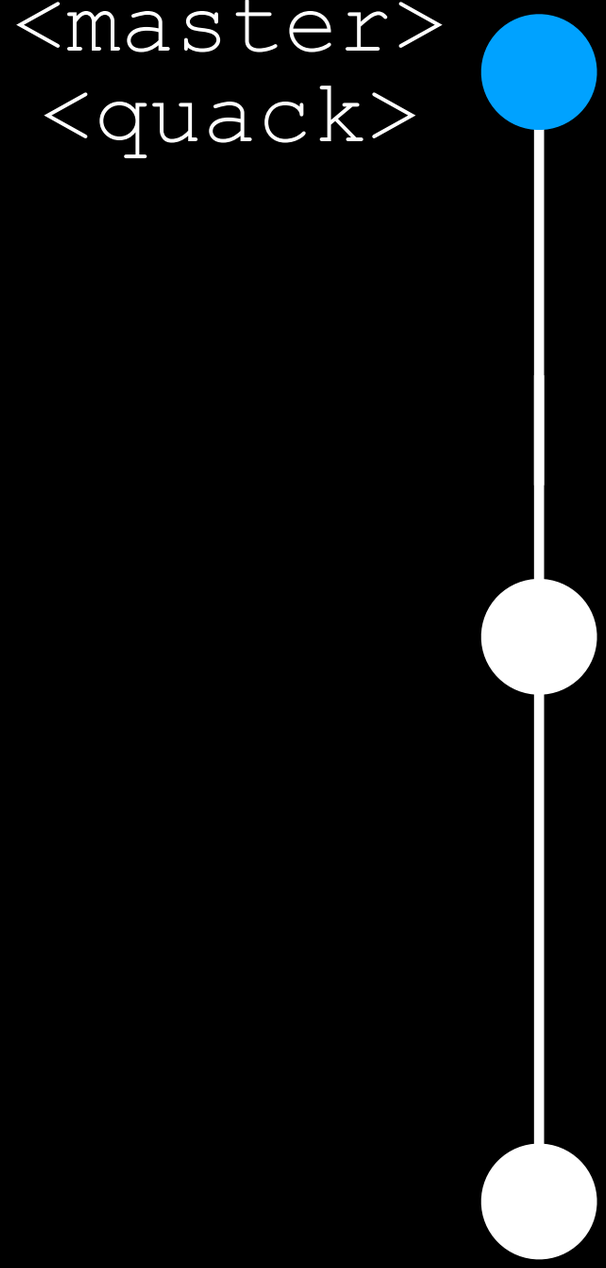
git merge quack



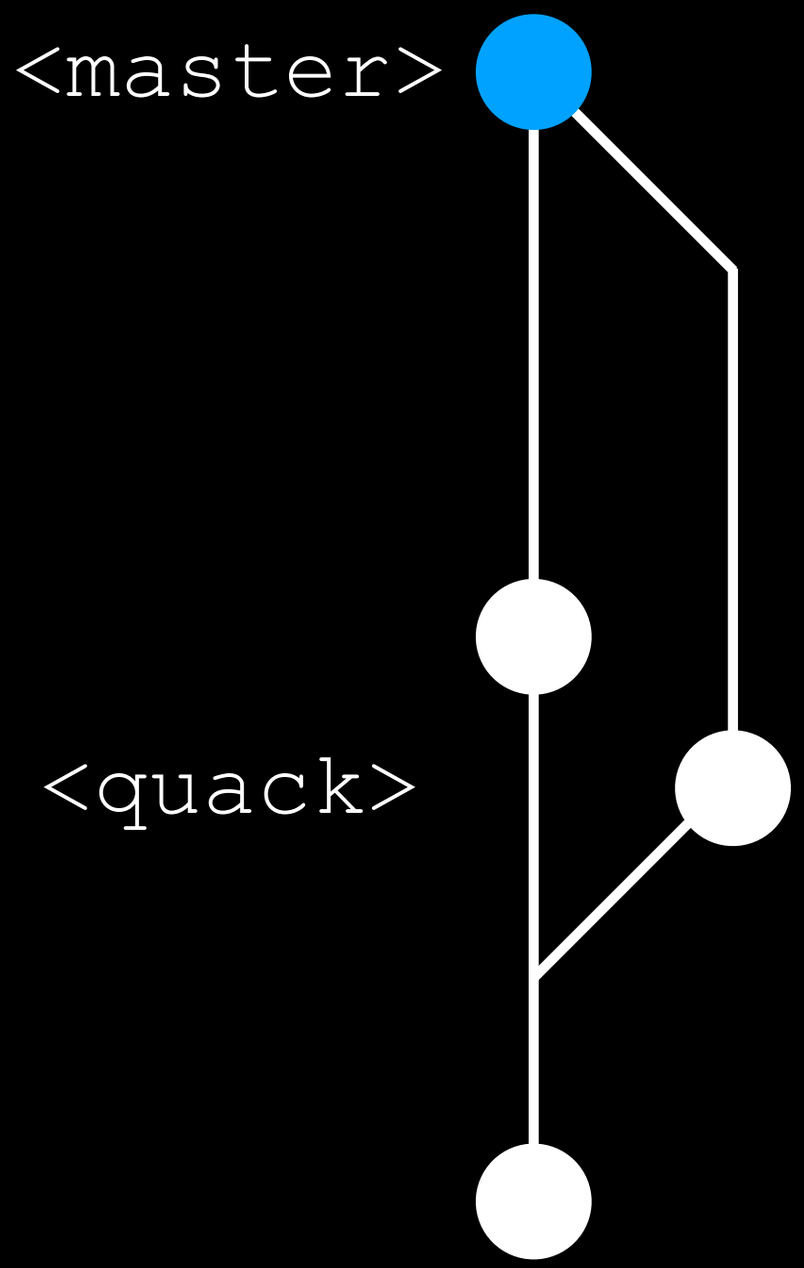
19bccb7 Quacking

b676eb2 Add more fox

ec6d394 Add Trebek pangram



Fast-Forward



Non Fast-Forward

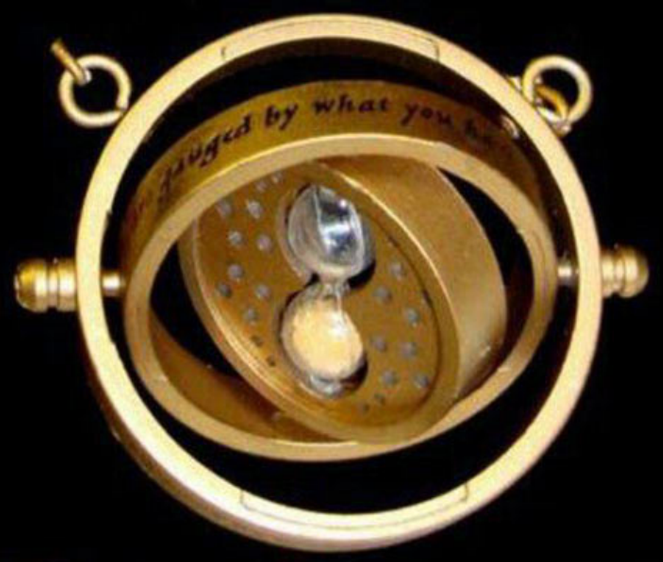
Demo

Fast-Forward

In Summary

- Fast-forwards will happen a lot (without you noticing)
- Preserves a linear commit history

Changing History



TIME-TURNER

SITUATIONS IT COULD HAVE BEEN USED

- Preventing the Potters murders;
- Witnessing Peter Pettigrew's muggle slaughter;
- Preventing escapes from Azkaban;
- Diminishing Auror casualties;
- Killing/Arresting the young Tom Riddle.

SITUATIONS IT HAS BEEN USED

- Taking an extra class.

Time Travel is Dangerous

Changing History is Dangerous

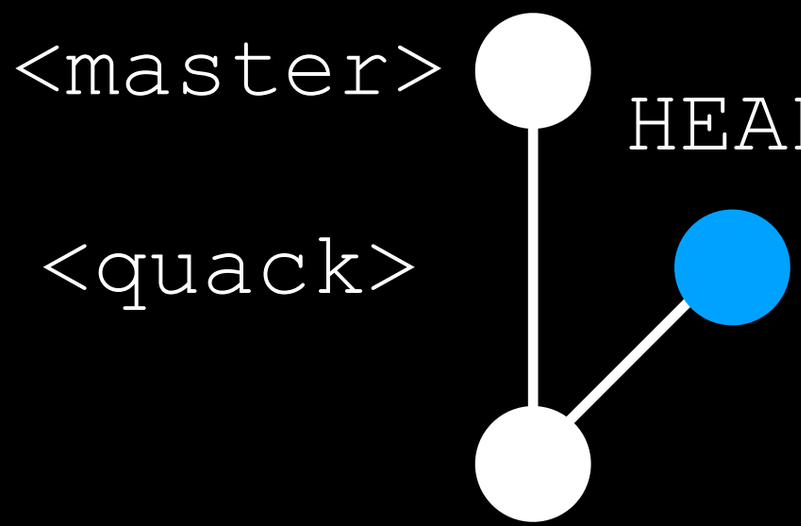
The plot of every time travel movie and Git

- Information can be permanently lost
- Undoing might not be obvious
- New history creates conflicts for everyone else on remote
    - More on this later
- You can live life without ever changing history
- Limit your risk by understanding the commands!

Rebase

Keep your branch up to date

- Changes the base commit of a branch
- Rebase will **change history**, albeit in a predictable way
- If used properly, makes it possible to fast-forward



536f67a Add more fox

ded1408 Quacking

f0f78cf Add Trebek pangram

<master>

536f67a Add more fox

+fox

pangram1.txt

<quack>

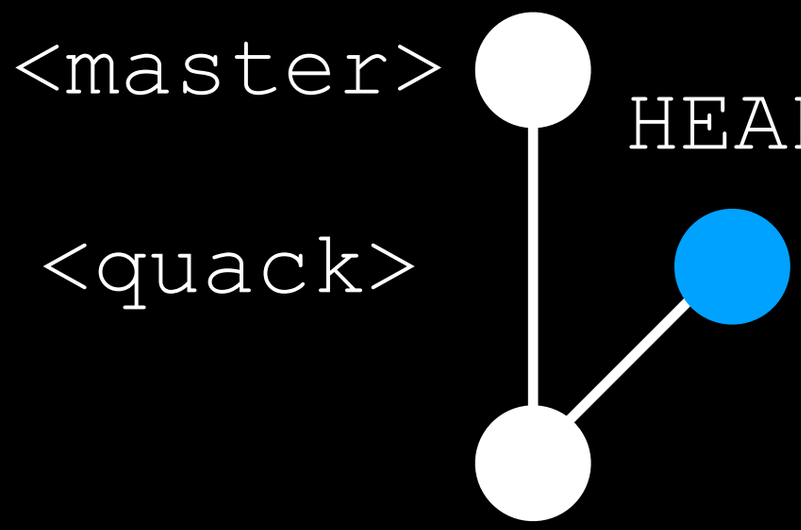
ded1408 Quacking

+quack

pangram1.txt

+quack

pangram3.txt

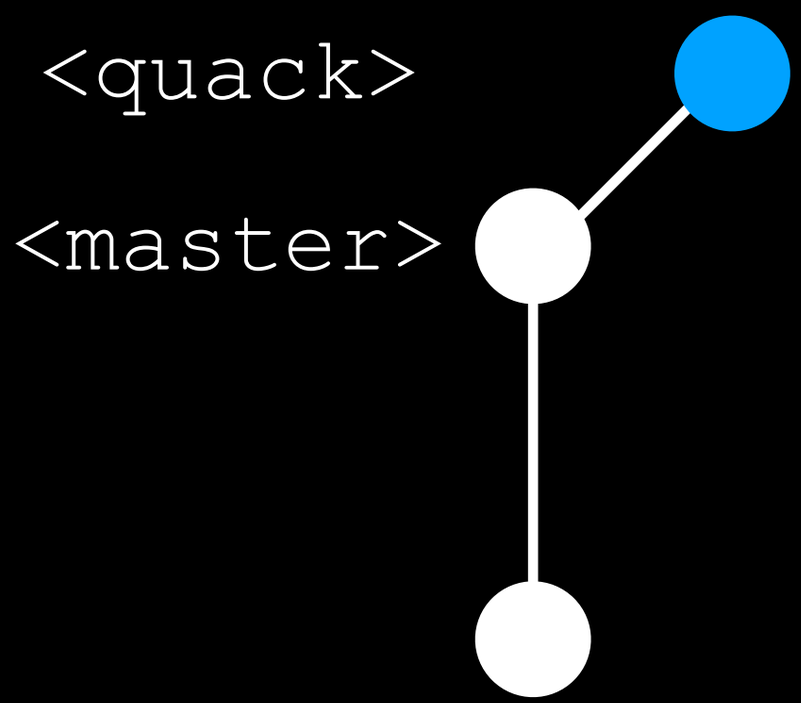


536f67a Add more fox

ded1408 Quacking

f0f78cf Add Trebek pangram

```
git rebase master
```

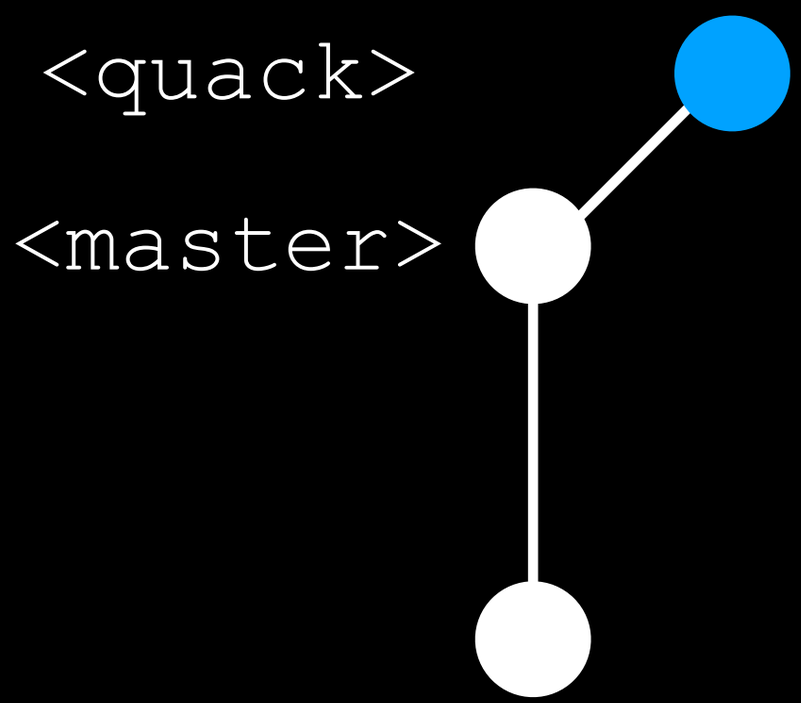


ded1408 Quacking

536f67a Add more fox

f0f78cf Add Trebek pangram

git rebase master



23c17e7 Quacking

536f67a Add more fox

f0f78cf Add Trebek pangram

Demo

Rebase vs Merge

- Merge preserves (a messy) history
- Rebase is a bit riskier, but creates **linear** history
- Use case depends very much on your team

Rebase vs. Merge

One point of view on this is that your repository's commit history is a record of what actually happened.

...

The opposing point of view is that the commit history is the story of how your project was made.

– Git.scm Website

Amend

Make a mistake? Me?

- Edit the last commit
- Add content
- Change the commit message

Interactive Rebase

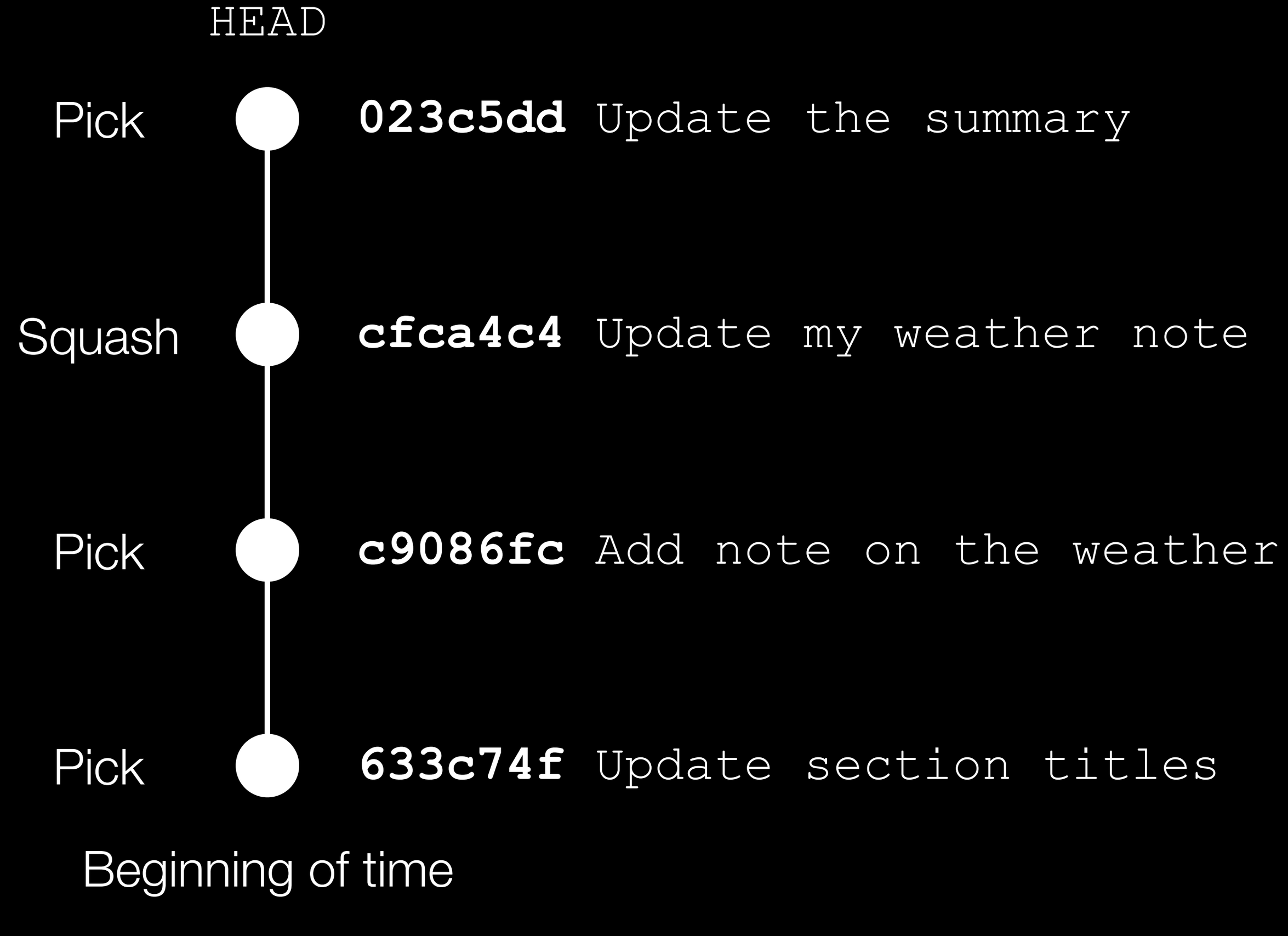
Superpowered commit editing

- An extension of regular rebase
- Examine commits before putting them on new base
- Remove/reorder/combine/edit commits

Break

Demo


HEAD
023c5dd Update the summary
cfca4c4 Update my weather note
c9086fc Add note on the weather
633c74f Update section titles
Beginning of time


HEAD
Pick
023c5dd Update the summary
Squash
cfca4c4 Update my weather note
Pick
c9086fc Add note on the weather
Pick
633c74f Update section titles
Beginning of time

HEAD

Pick **023c5dd** Update the summary

Squash **cfca4c4** Update my weather note

Pick **c9086fc** Add note on the weather

Pick **633c74f** Update section titles

Beginning of time

HEAD

Squash **cfca4c4** Update my weather note

Pick **c9086fc** Add note on the weather

Pick **023c5dd** Update the summary

Pick **633c74f** Update section titles

Beginning of time

HEAD

08862bd Add note on the weather...

Pick **8155cd2** Update the summary

Pick **633c74f** Update section titles

Beginning of time

HEAD

08862bd Add note on the weather...

Pick **8155cd2** Update the summary

Pick **633c74f** Update section titles

Beginning of time

Revert

Made a mistake? Me?

- Revert **does not change history**
- Create a new commit undoing previous commit

Reference Log

Aka "reflog"

- Regular log shows history of commits
- Reference log ("reflog") shows all history (HEAD positions)

Reference Log

- Both checkout and reset work the same
  - Checkout -- updates HEAD only
  - Reset -- updates HEAD and current branch
  - `HEAD@{n}` represents `HEAD` n steps ago

Demo

Revisiting Remotes

Projects

GitLab

Signatures

Pull requests

Fork

Fetch

Issues

Organizations

Customizing remotes

ssh keys

Bitbucket

Projects

GitLab

Signatures

Pull requests

Fork

Fetch

Issues

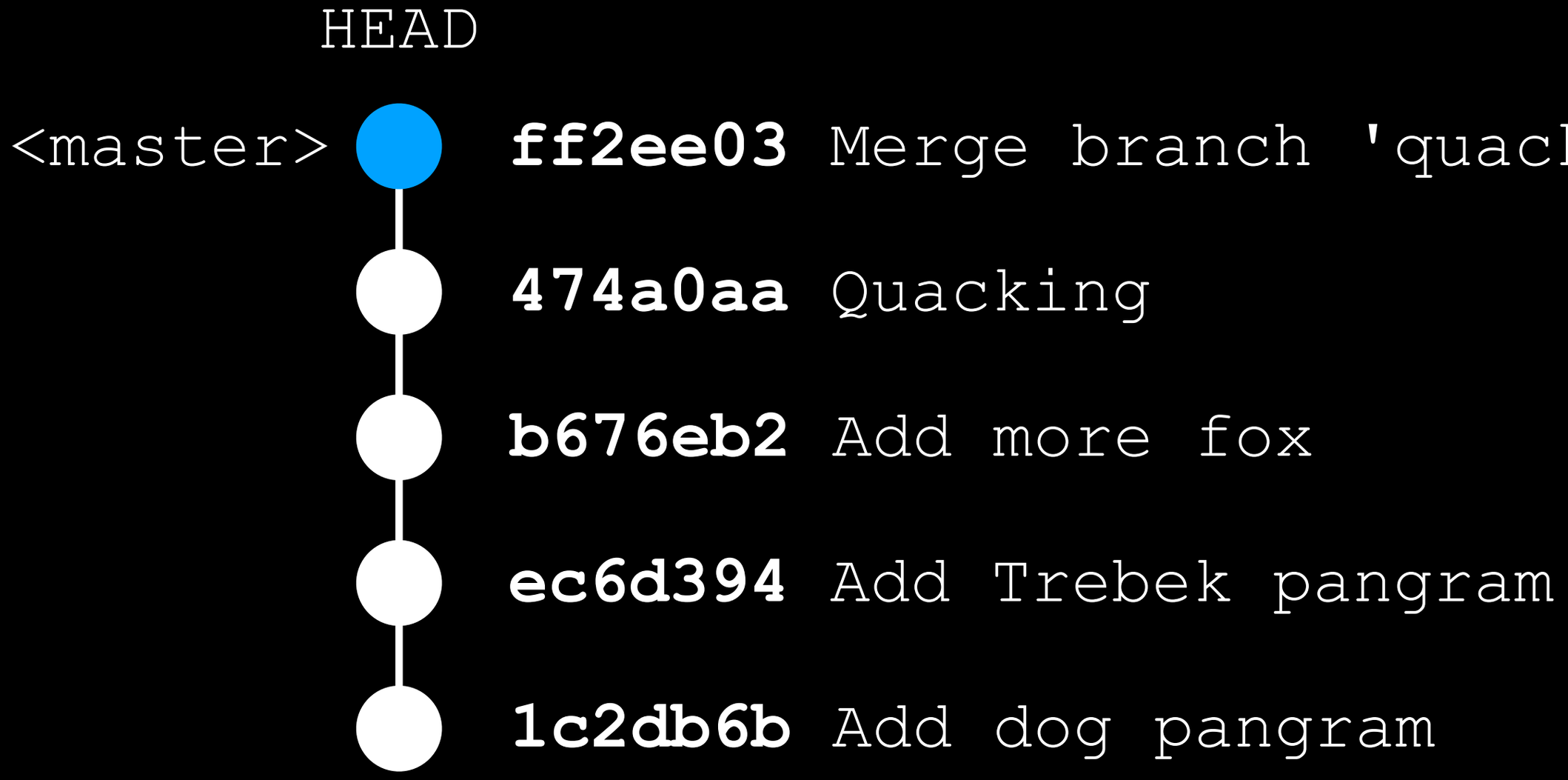
Organizations

Customizing remotes

ssh keys

Bitbucket

Push/Pull example from last time

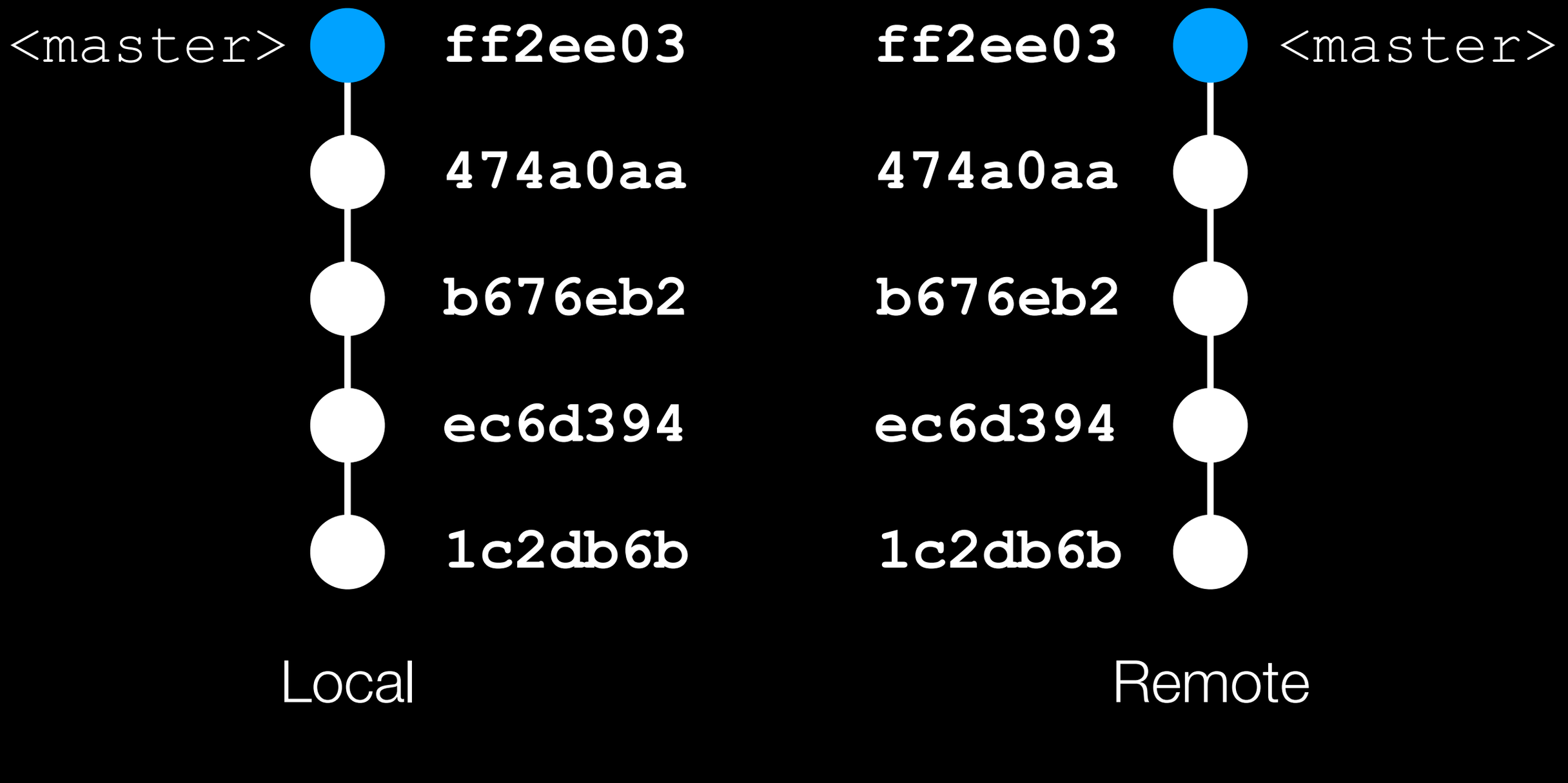


Local

<master>

Remote

git push origin master



git commit (on another machine)

git pull origin master

Fetch

The secret behind pull

- The pull command is actually a fetch + a merge
- **Local remote tracking branch** for each remote branch
- Fetch only updates the remote tracking branches


<master>
ff2ee03
474a0aa
b676eb2
ec6d394
1c2db6b
Local
ff2ee03
<master>
474a0aa
b676eb2
ec6d394
1c2db6b
Remote: origin


<origin/master>
<master>
ff2ee03
474a0aa
b676eb2
ec6d394
1c2db6b
ff2ee03
<master>
474a0aa
b676eb2
ec6d394
1c2db6b
Local
Remote: origin

<o/master>

<master> **ff2ee03**

474a0aa

b676eb2

ec6d394

1c2db6b

Local

ff2ee03 <master>

474a0aa

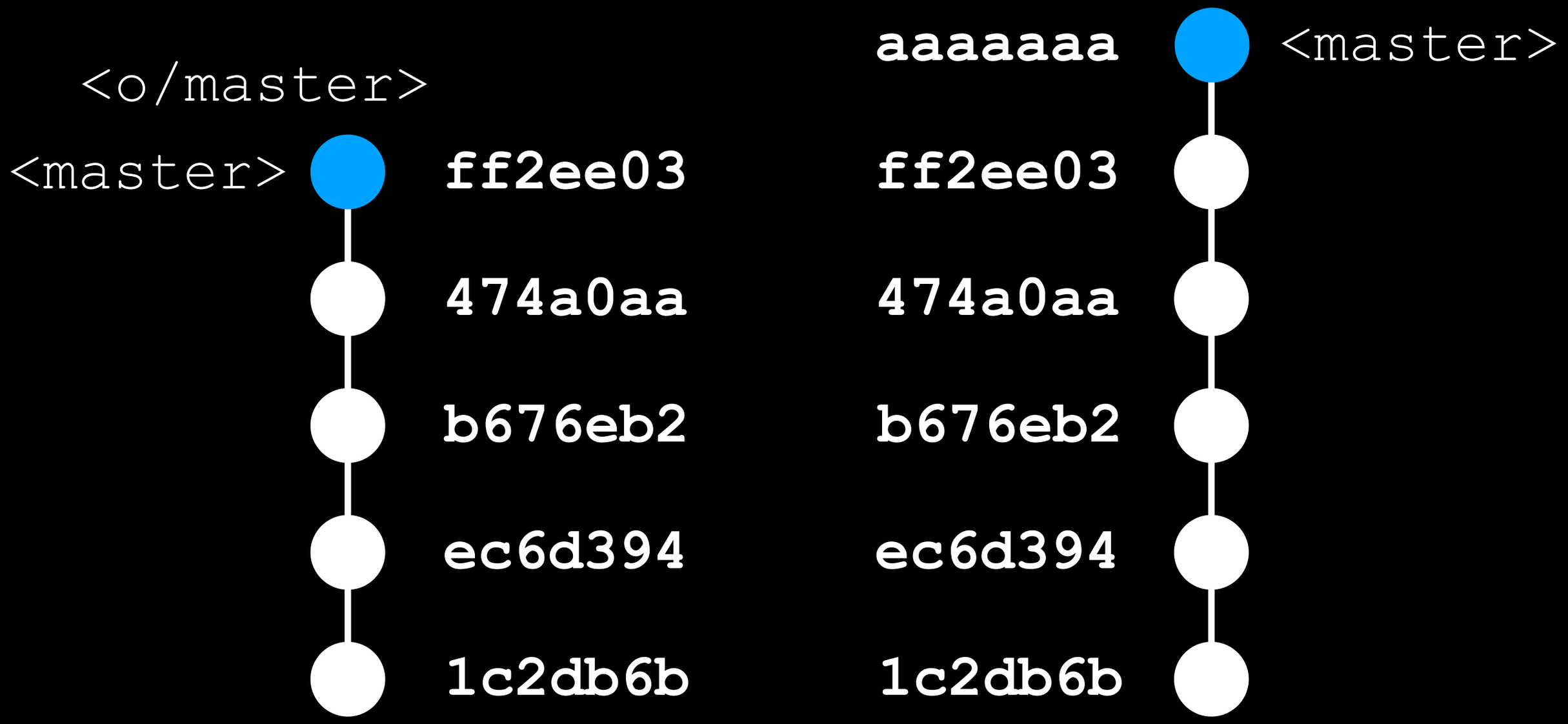
b676eb2

ec6d394

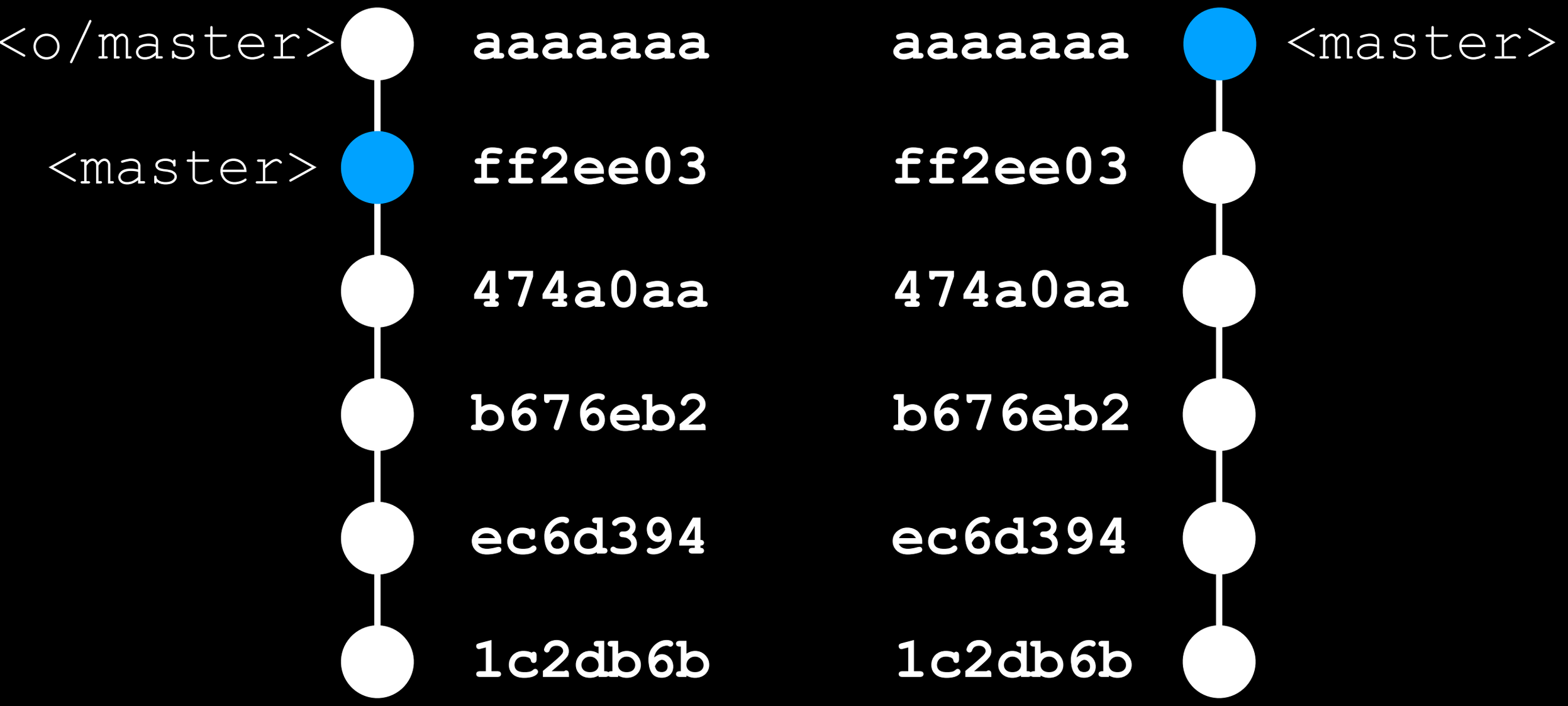
1c2db6b

Remote: origin

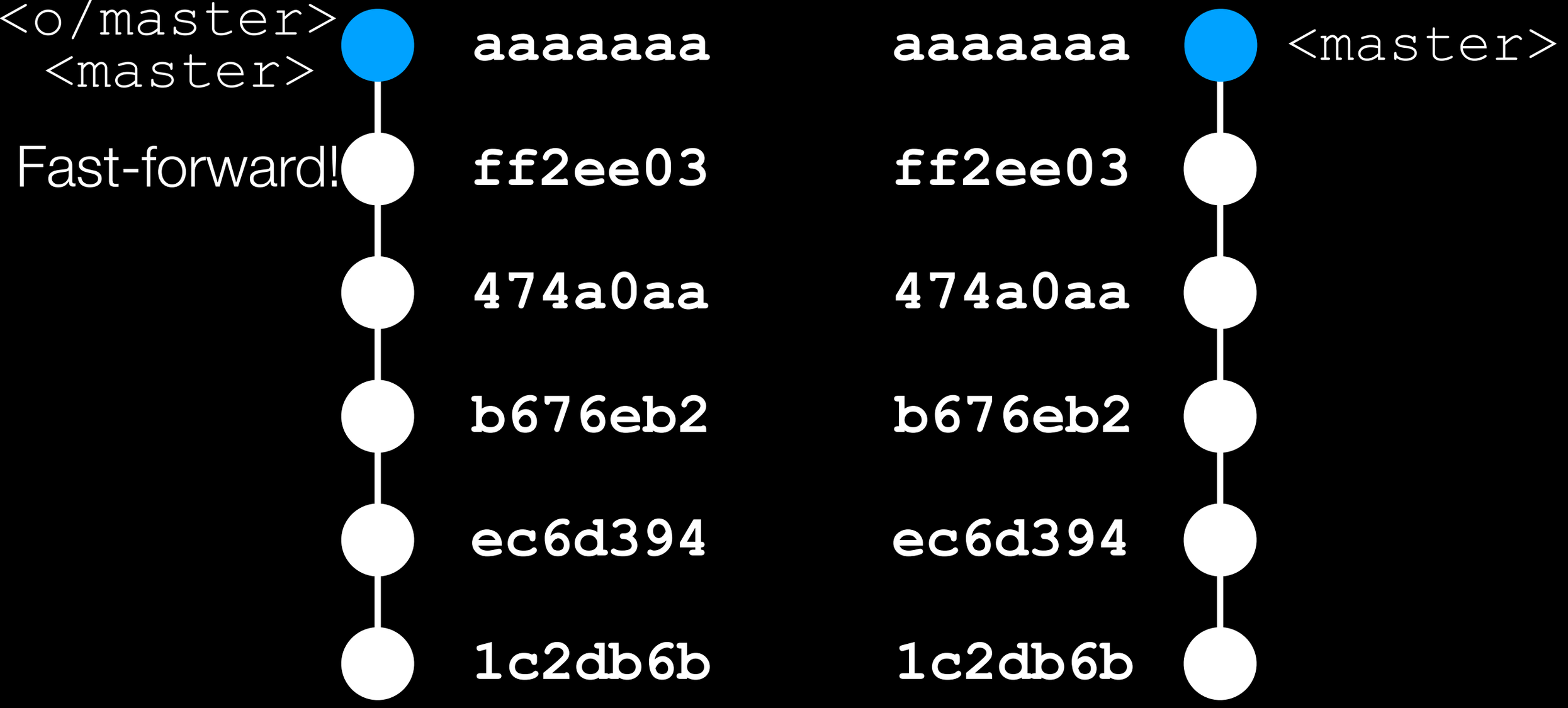
git commit (on another machine)



git fetch origin



git merge origin/master



Fetch

Why should you care?

Fetch

Why should you care?

- You probably shouldn't
- Pull is fine for the majority of use cases

Fetch

Why should you care?

- Fetch is fairly harmless
- Fetch all you want!

Fetch

Why should you care?

- Fetch is fairly harmless
- Fetch all you want!
- Or not...

Fetch

Why should you care?

- Fetch is fairly harmless
- Fetch all you want!
- Look at changes before merging in

Pull Requests

Aka: "Please merge my branch!"

- This is not a Git feature (GH, Bitbucket, etc. may differ)
- A **request** to **pull** in changes from a branch
- Allows a more formal review process
- GitHub can help you rebase and/or squash before merge

Push

Adding tags to your remote

- Use push to update branches
- Can also use it to push tags

Changing History, Again

Public vs Private

- Pushed commits are "**public**"
- Try to limit history changes to private history only
- This also applies to **tags**
- Use force push to overwrite public history

Standard Workflow

Three (and actually three) easy steps

1. Work on your own branch
2. Keep up to date (fetch/pull)
3. Push work regularly

Demo

Summary

Last week

- Create local repos
- Make backups (commits), create and join branches
- Backup to the cloud

Summary

This week

- Clean up your repo
- Create linear commit history
- Fix mistakes in the past
- Learn the truth behind remotes
- Use GitHub collaboration tools

Odds & Ends

- ssh keys
- GitHub Issues

```
git commit -m
"End of workshop"
```