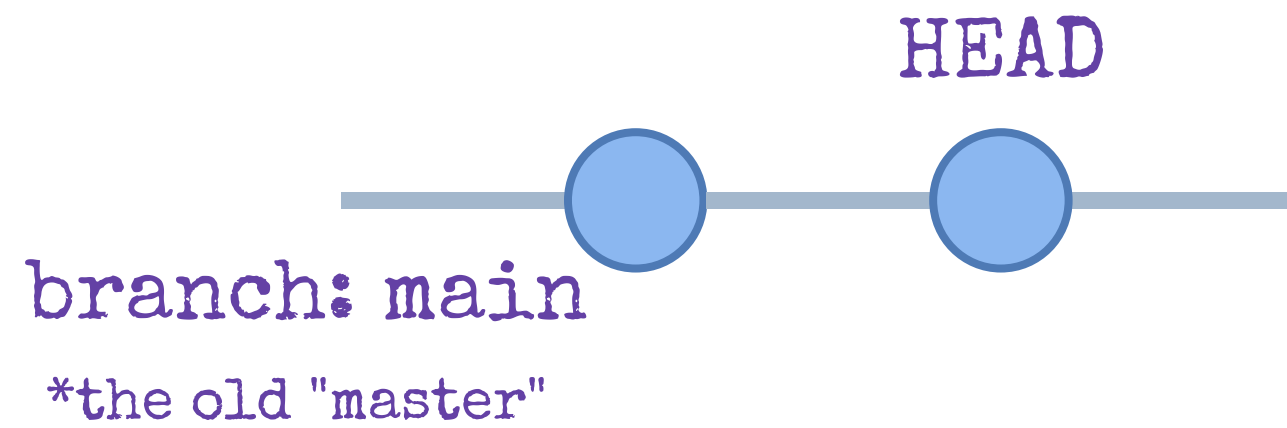
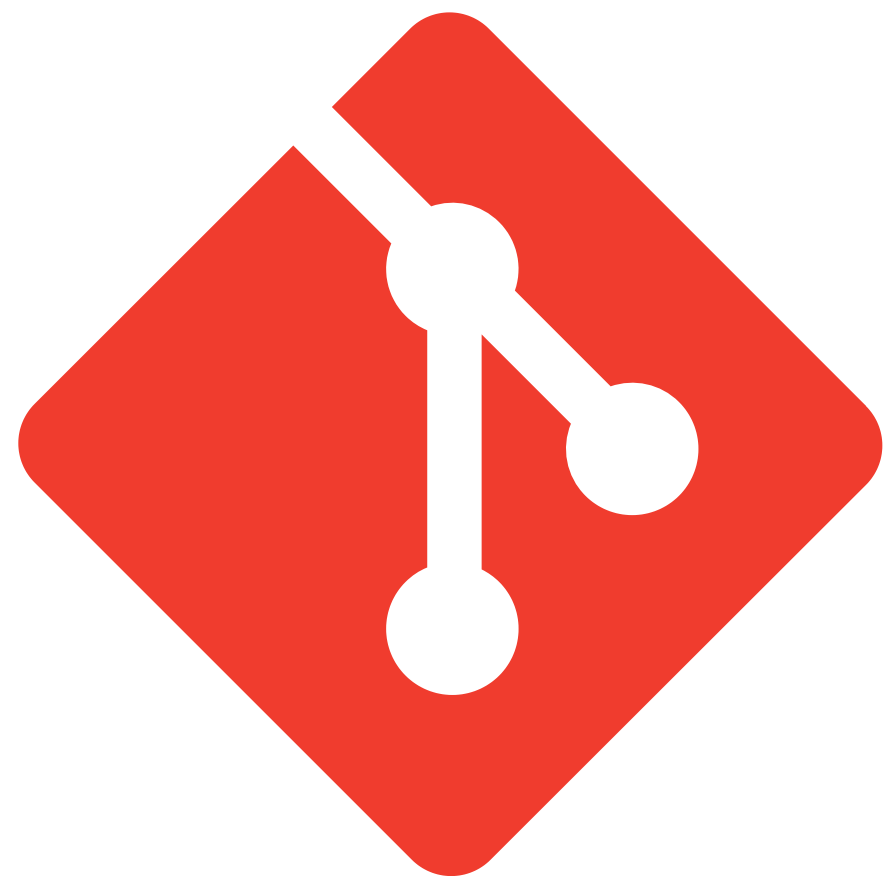


Git

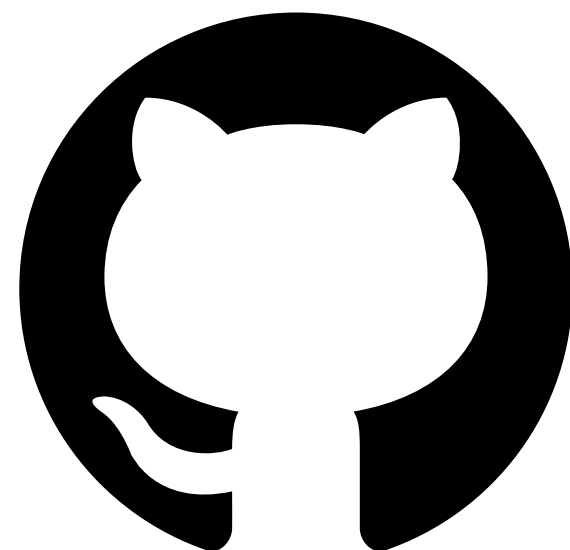


Concept of a "Stack"

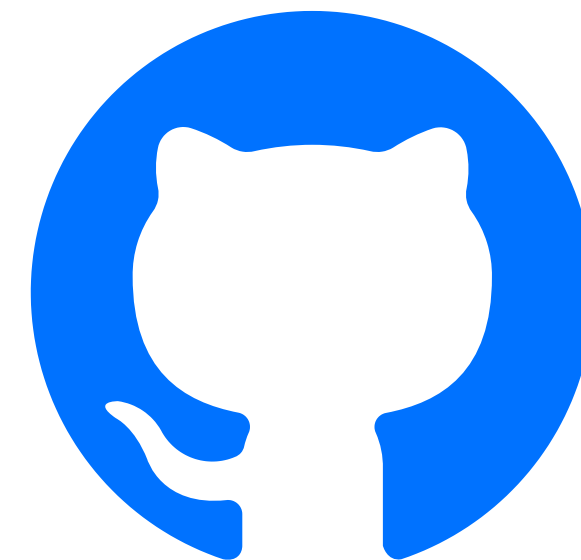




Git



GitHub



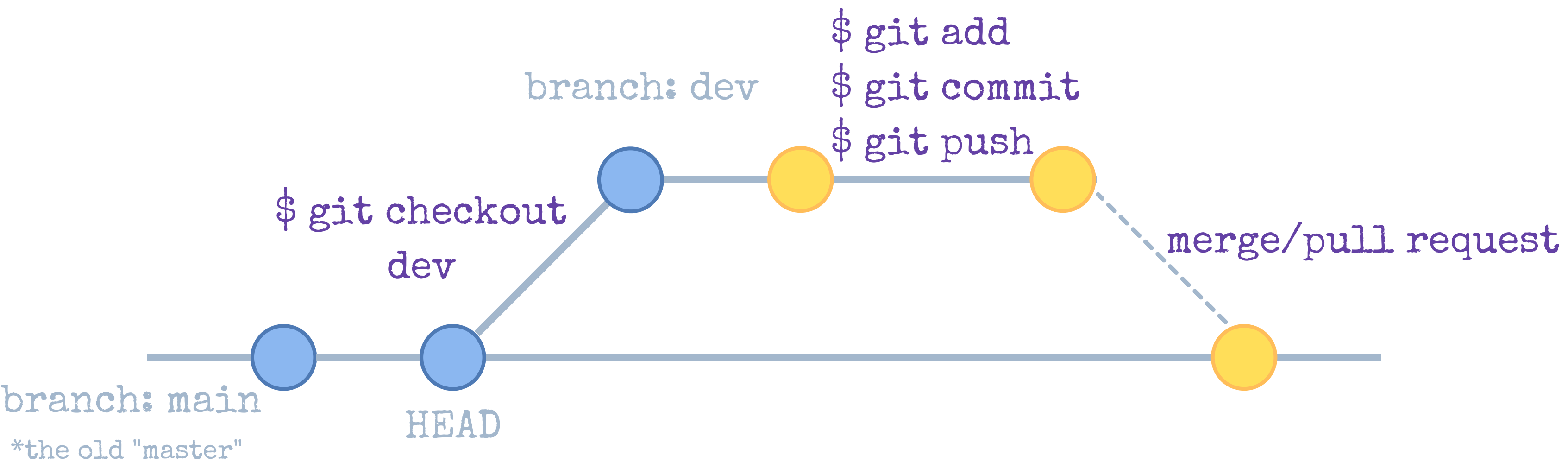
GitHub
Enterprise



GitLab

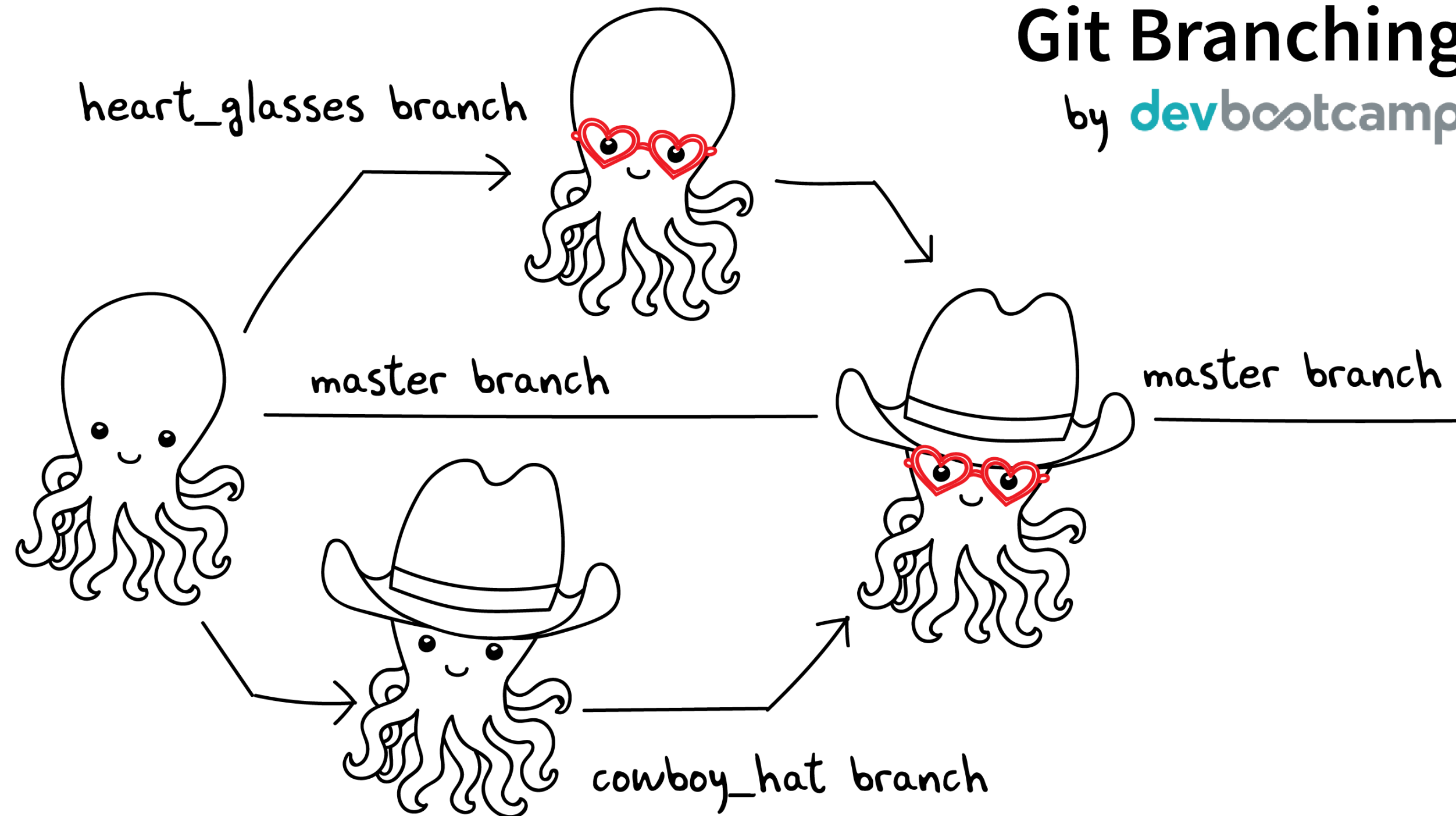


Azure
Repos

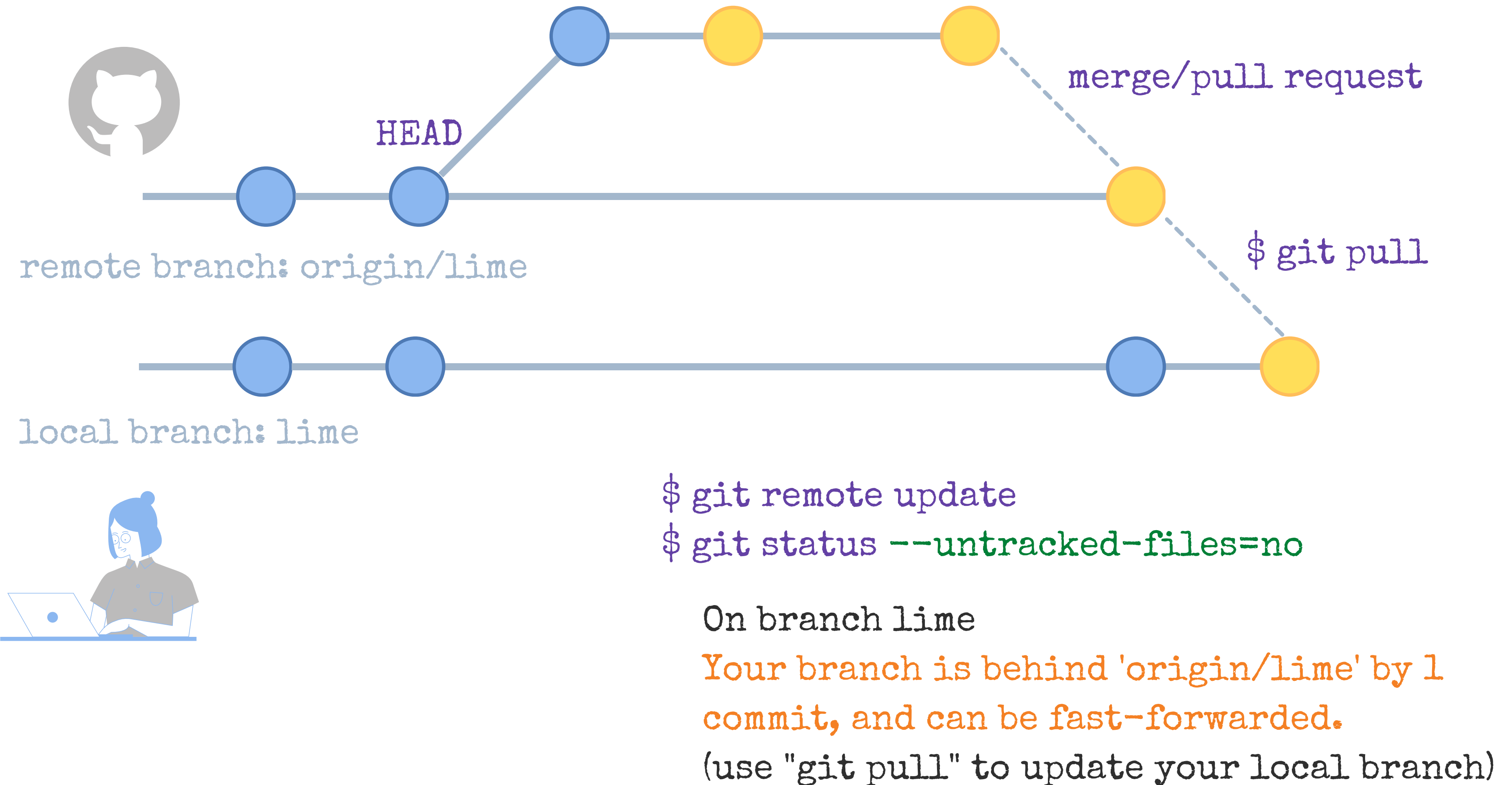


Git Branching

by **dev**bootcamp





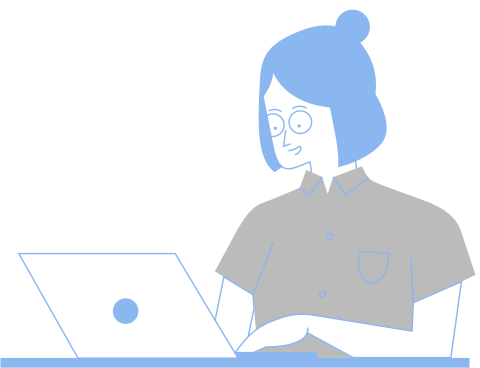




HEAD

remote branch: origin/apple

local branch: lime



```
$ git branch --set-upstream-  
to=origin/apple lime
```

Branch 'lime' set up
to track remote
branch 'apple' from
'origin'.

```
$ git pull
```



HEAD

remote branch: origin/apple

local branch: lime

```
$ git branch --  
set-upstream-  
to=origin/apple  
lime
```

```
$ git stash  
save
```

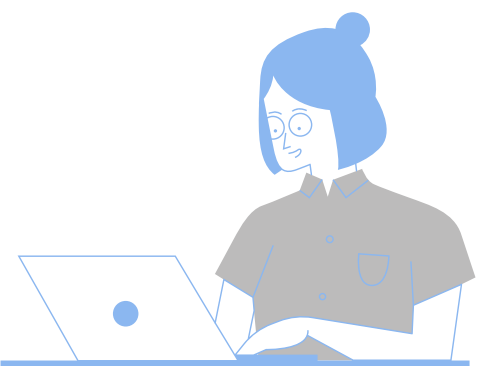
```
$ git pull
```

```
$ git stash  
pop
```

A stash is
a bkp stack



Auto-merging [file]
CONFLICT (content):
Merge conflict in [file]
The stash entry is kept
in case you need it
again.



(1) Solve inconsistencies manually where git issued warnings:

A

<<<<<< Updated upstream

B

=====

C

>>>>>> Stashed changes

(2) Add and push:

```
$ git add .
```

```
$ git push origin HEAD:apple
```

This is the safest method available!

Ignore certain files:

file.py

Wild cards:

*.c

Ensure specific files are sent:

!frob_*.c

!custom.c

Relative paths:

dir/file.py

Relative paths with wild cards:


dir/*

Dir and subdirs:


dir/*/**

How to ignore files!

Owner *

 felipepenha ▾

 /


test 


Repository name *

Great repository names are short and memorable. Need inspiration? How about **legendary-succotash**?

Description (optional)

Test

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)


☒ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)

☒ **.gitignore template**

Filter ignores...

Processing

PureScript

 Python

Qooxdoo

or code. [Learn more.](#)

the default name in your [settings](#).

tabela.csv
credentials.json

.gitignore

tabela.csv
credentials.json

\$ git rm --cached [file]

\$ git rm --cached tabela.csv

\$ git rm --cached credentials.json

.git

Only after, you include [file] in .gitignore

Tags!

- tags are unique but anything really
- conventions:
 - Semantic Versioning 2.0.0
 - <https://semver.org/>

Given a version number MAJOR.MINOR.PATCH, increment the:

MAJOR version when you make incompatible API changes,
MINOR version when you add functionality in a backwards compatible manner, and
PATCH version when you make backwards compatible bug fixes.

Major version zero (0.y.z) is for initial development. Anything MAY change at any time. The public API SHOULD NOT be considered stable.

Tags!

```
$ git tag --annotate [x.y.z]  
    --message "[message]"
```

How to push with the tag?

```
$ git push origin [branch_name] tag [x.y.z]
```

```
git clone --branch [tag_name] [clone_url]
```

```
git checkout tags/[tag_name] -b [new_branch]
```

```
git fetch --tags
```

```
git tag --list
```

•git


```
git [command] --help
```