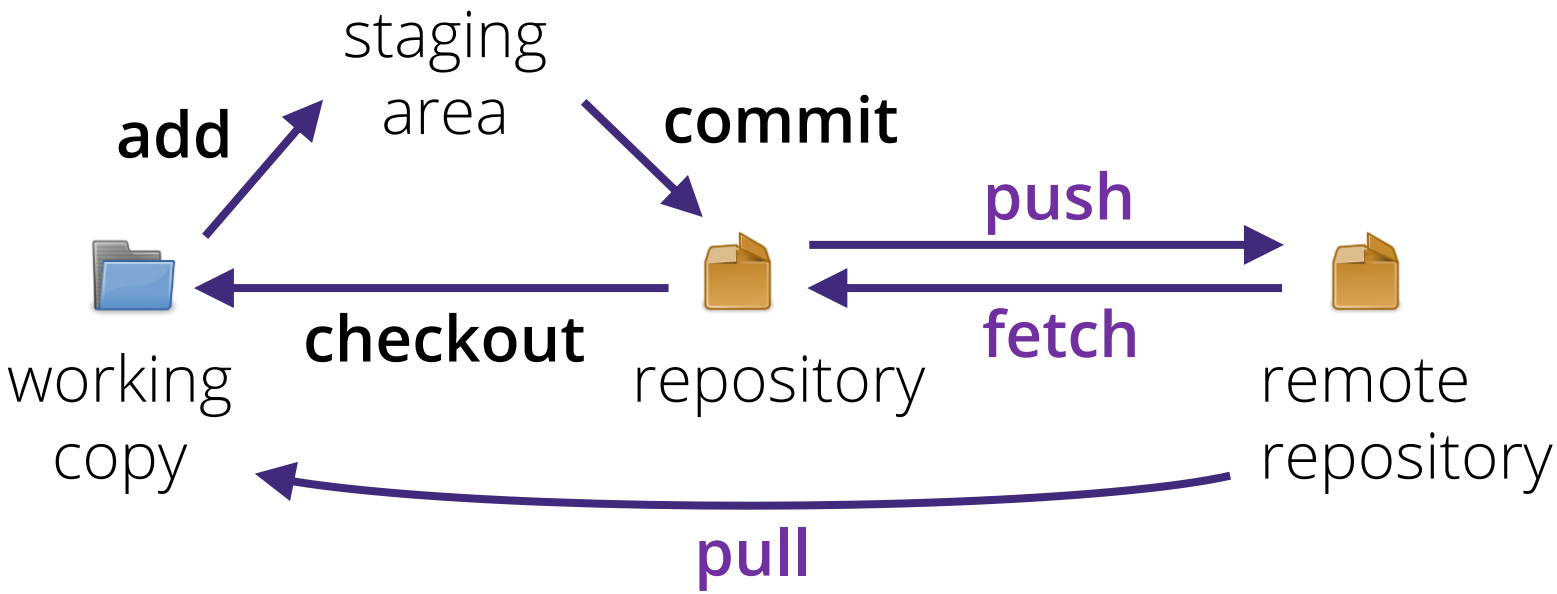
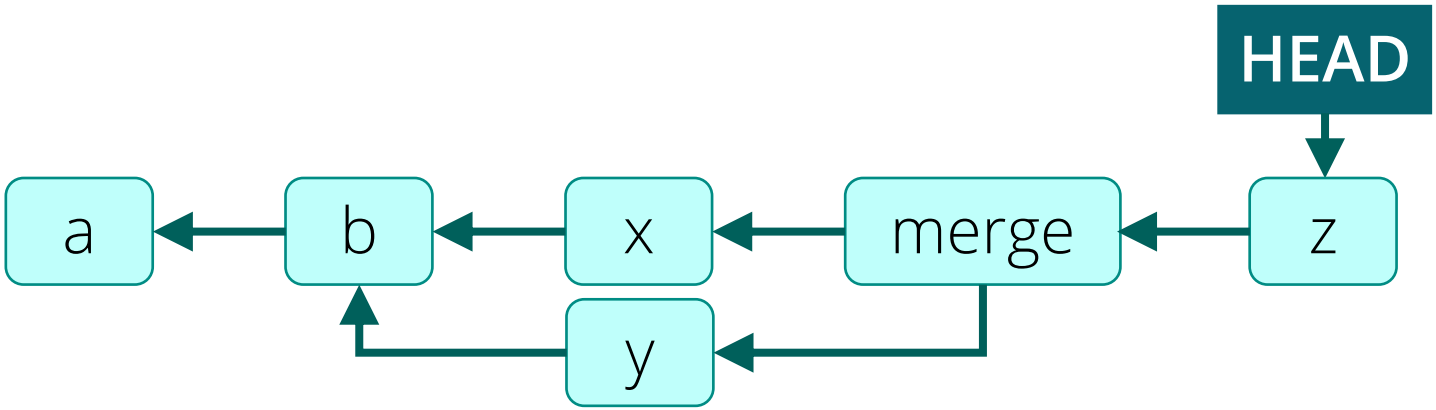
Git 3

COMS10012 Software Tools

reminder

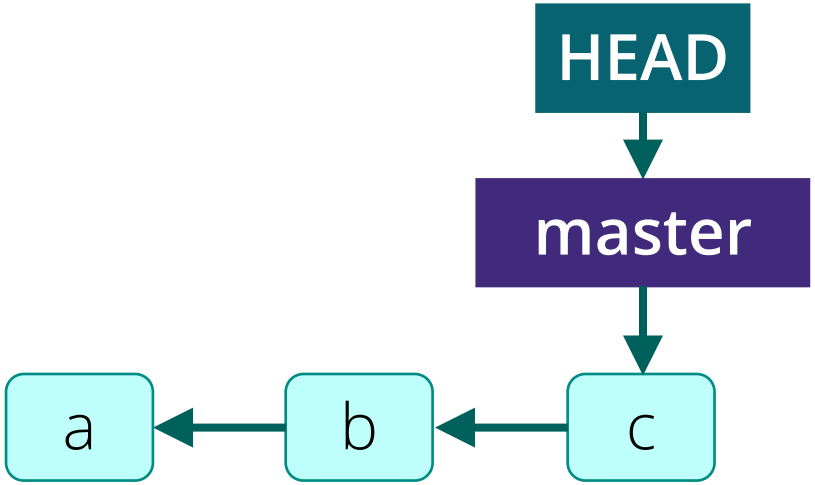

staging
area
add
commit
push
checkout
fetch
working
copy
repository
remote
repository
pull

reminder

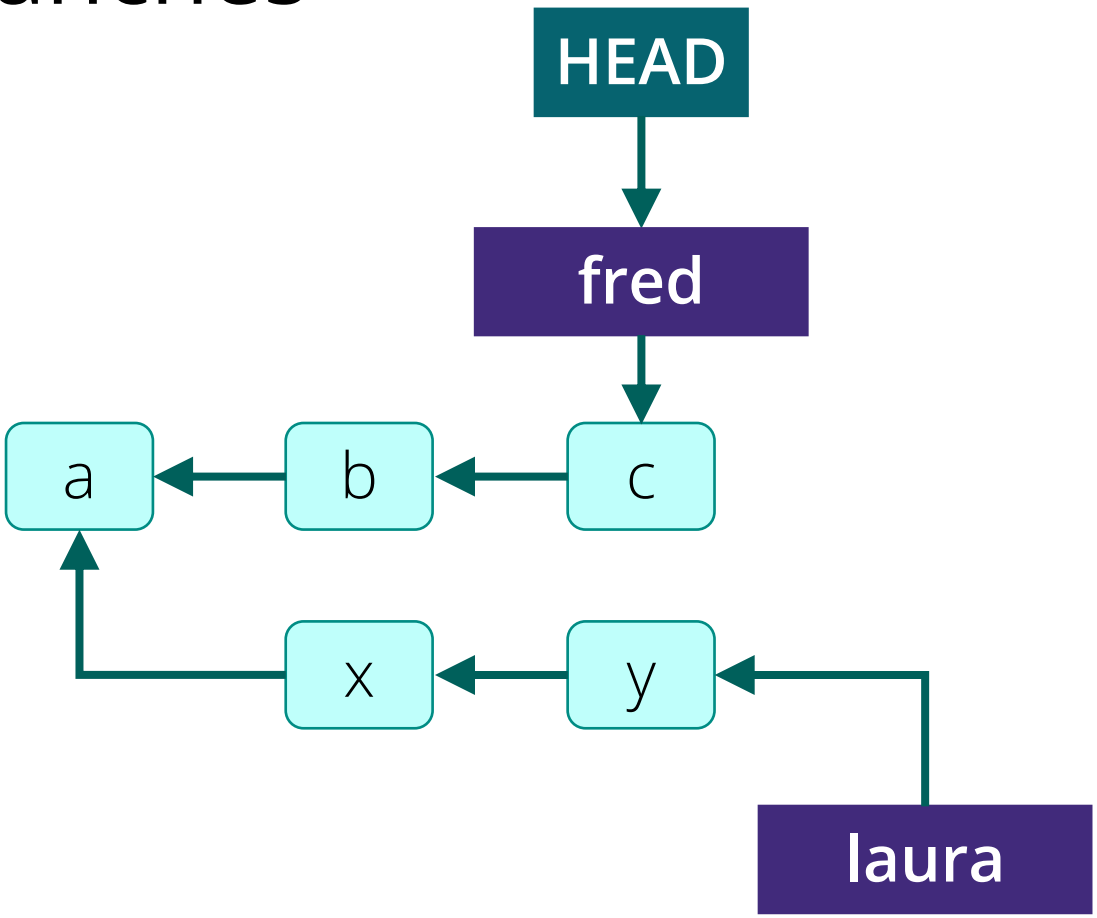


branches

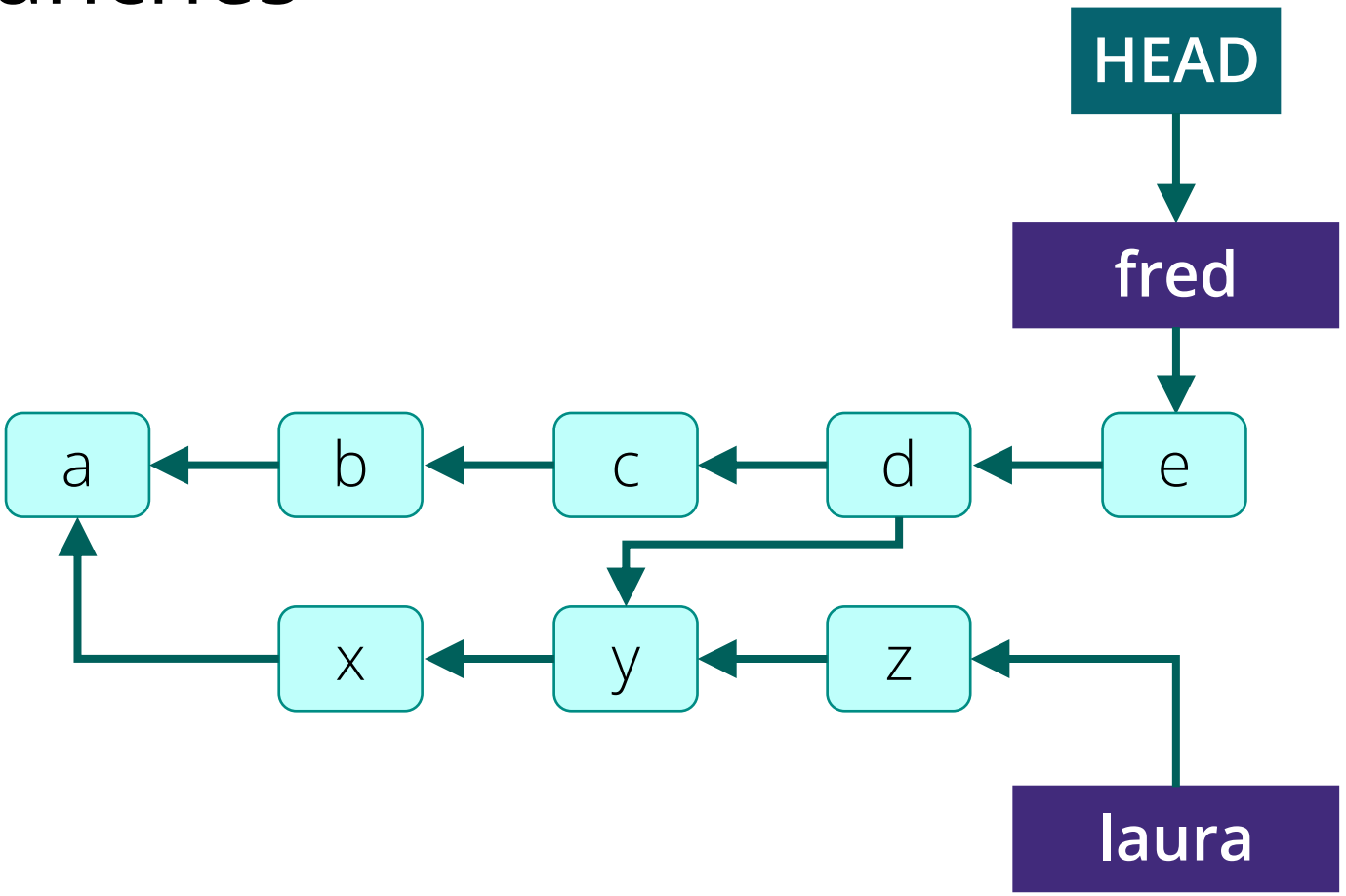
master



branches



branches



git multiplayer principles

- small commits
- frequent commits
- separate branches
- merge when you are ready

branches

```
$ git checkout -b BRANCHNAME
```

create a branch

```
$ git branch
```

list all branches

```
$ git checkout BRANCHNAME
```

switch to a branch

branch merge

Fred wants to merge Laura's work:

```
(fred)$ git merge laura
```

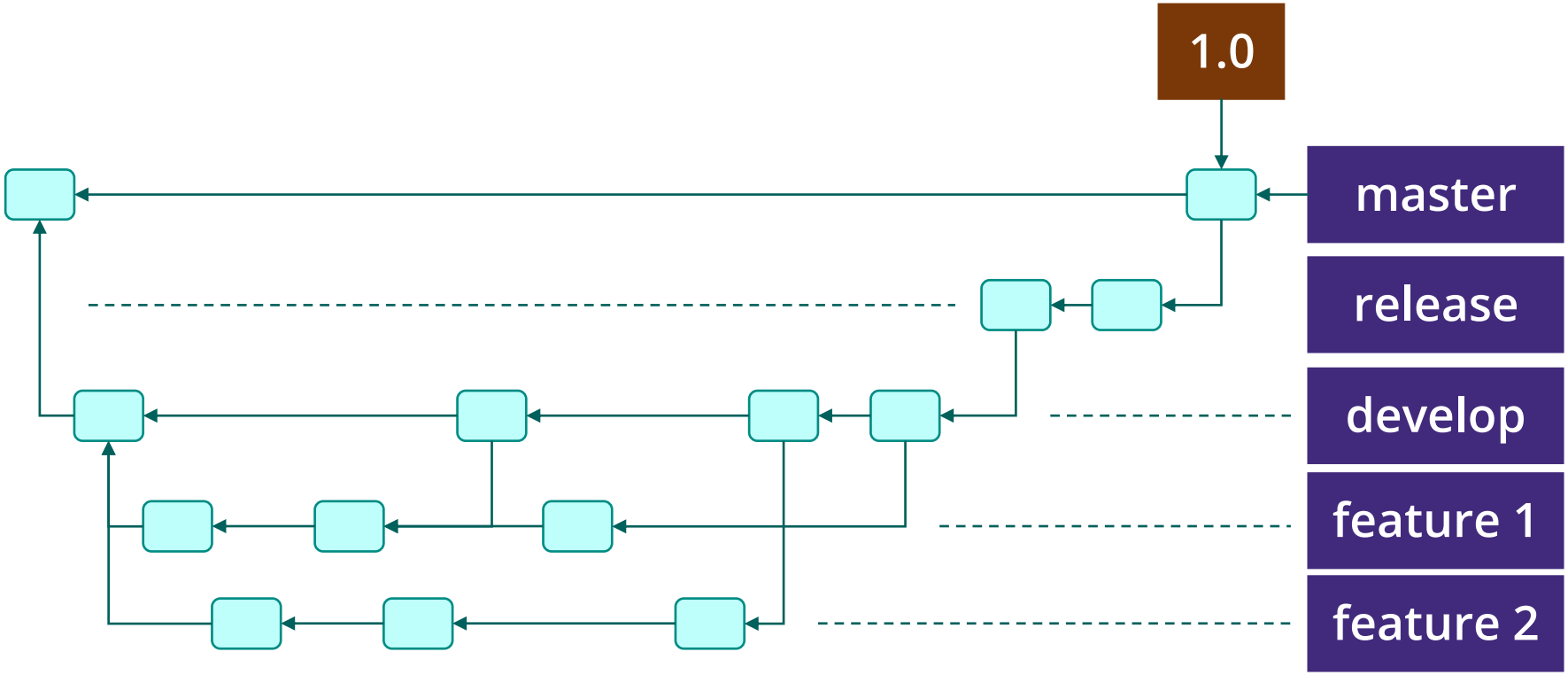
Creates a new commit on the current branch (fred) that includes the latest commit from the other (laura) branch.

feature branches

principles

- **master**: releases, snapshots
- **develop**: for developers to share code
- actual work happens on **feature branches** (small, forked off develop)
- **release**: tidying up before a release

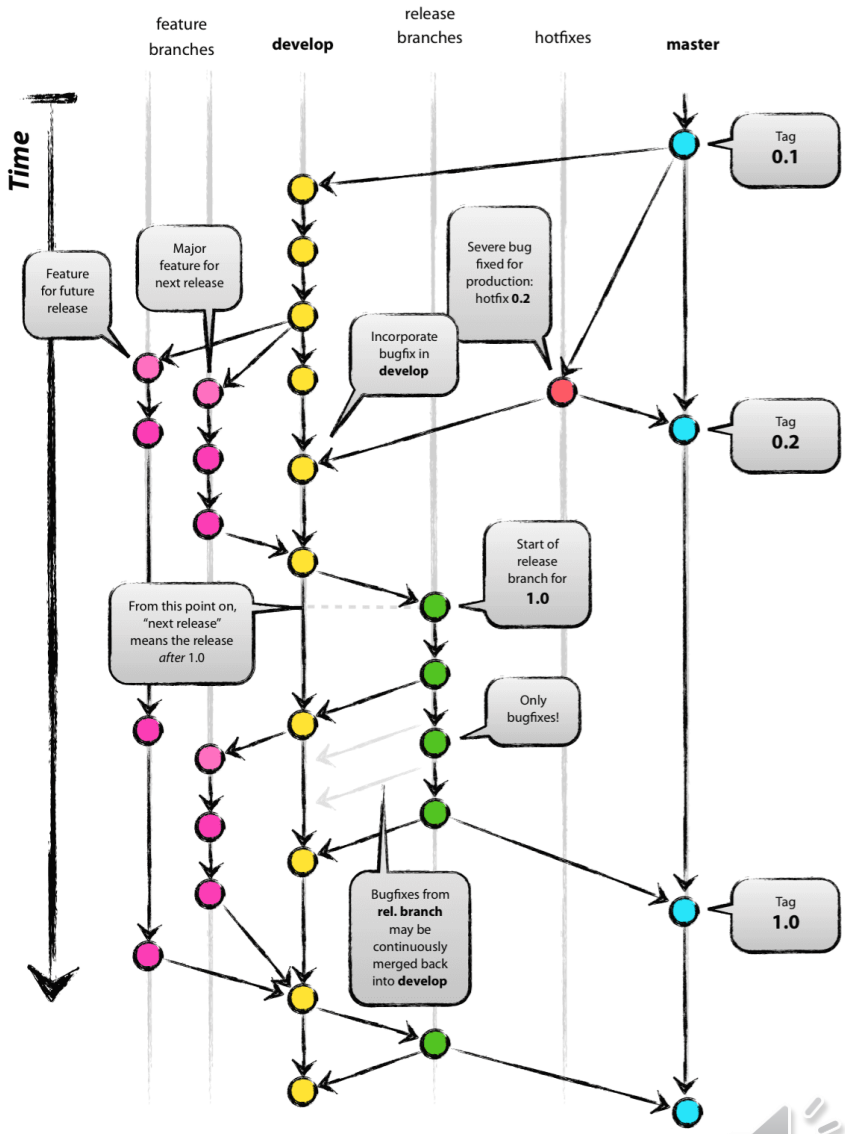
diagram



gitflow

https://nvie.com/posts/a-successful-git-branching-model/

(2010)



principles, again

- small, frequent commits
- work happens on lots of small branches
- conflicts only happen when you choose to merge into develop/release/master

pull requests

In some flows, only branch owners can merge – for master (develop) this is project managers.

Developers do work on their own branches and submit **pull requests.**

This lets you do a code review before merging.

open source

To fix a bug or add a feature in someone's code:

1. Clone their repository.
2. Make changes in your own copy.
3. Submit PR back to the original.