git recitation, htm 204

Nathan Perry (kerb = npry)

will be demoing – if interested in following along, install git cli ->

(can't stop to get people running, slides + site available)

goals

- basic functional background
- get you configured and committing to your repos if not already
  - follow along if you like, slides and website docs will be available
- conceptual backing at the end as/if we have time

what exists

one repo per child – these get deployed to the web (linked in people page)

site for each section (cba, harv, arch, csail)

individual assignments go in your site

group assignments go in section site – document your contributions on your site

sites hosted on gitlab

why version control?

paper final final revised (Copy).docx

why version control?

paper final final revised (Copy).docx

* first draft (-> paper.docx)

why version control?

paper final final revised (Copy).docx

* first draft (-> paper.docx)
* first round of edits, restructuring (-> final)

why version control?

paper final final revised (Copy).docx

* first draft (-> paper.docx)
* first round of edits, restructuring (-> final)
* grammar and spelling cleanup (-> final final)

why version control?

paper final final revised (Copy).docx

* first draft (-> paper.docx)
* first round of edits, restructuring (-> final)
* grammar and spelling cleanup (-> final final)
* more revisions (-> revised)

why version control?

paper final final revised (Copy).docx

* first draft (-> paper.docx)
* first round of edits, restructuring (-> final)
* grammar and spelling cleanup (-> final final)
* more revisions (-> revised)
* fix date (-> (Copy))

version control

- manages this versioning for you
- "time traveling"
- collaboration

* first draft (paper.tex, created by alice)

* first round of edits (paper.tex, modified by bob)

git?

ubiquitous version control for software development

initially developed for linux kernel dev (learning curve)

popular centralized hosting (GitHub, GitLab)

how does it work

```
$ git add paper.tex
$ git commit -m "first draft"
```

how does it work

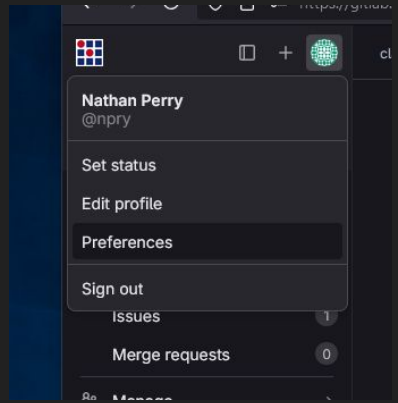
```
$ git add paper.tex

$ git commit -m "first draft"

$ git push # git@gitlab.cba.mit.edu:846/...
```

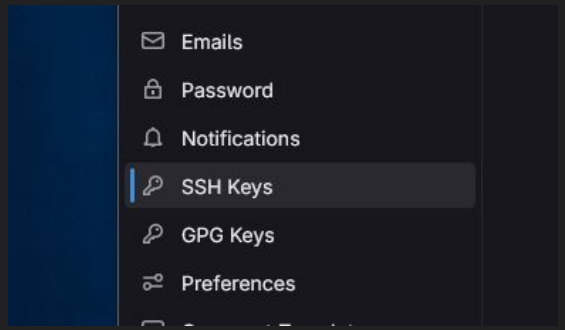
how does it work

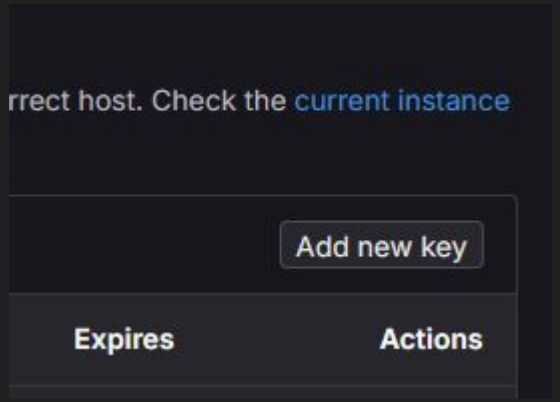
```
$ git add paper.tex
$ git commit -m "first draft"
$ git push # git@gitlab.cba.mit.edu:846/...
# make some changes
$ git add paper.tex
$ git commit -m "first round of edits"
$ git push
```

getting started (ssh keys)

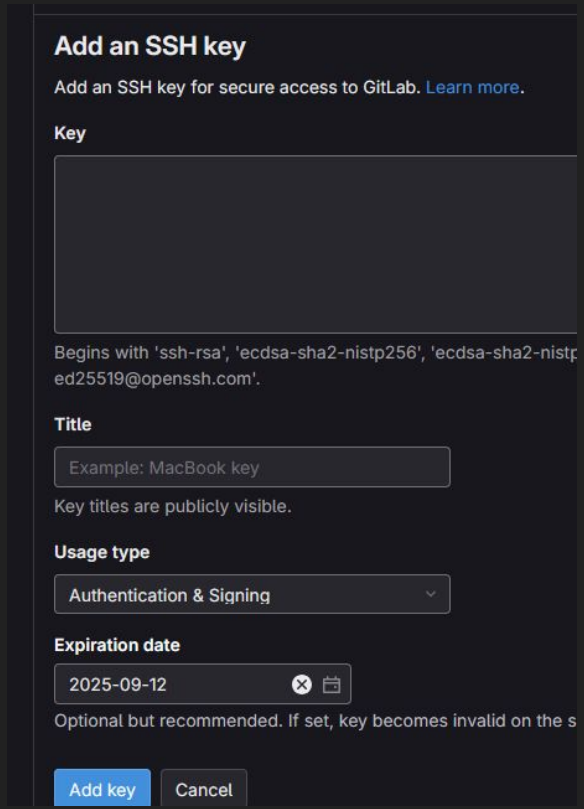
```
$ ls ~/.ssh  # id_rsa.pub, id_ed25519.pub?

$ ssh-keygen # if not

$ cat ~/.ssh/id_rsa.pub
```





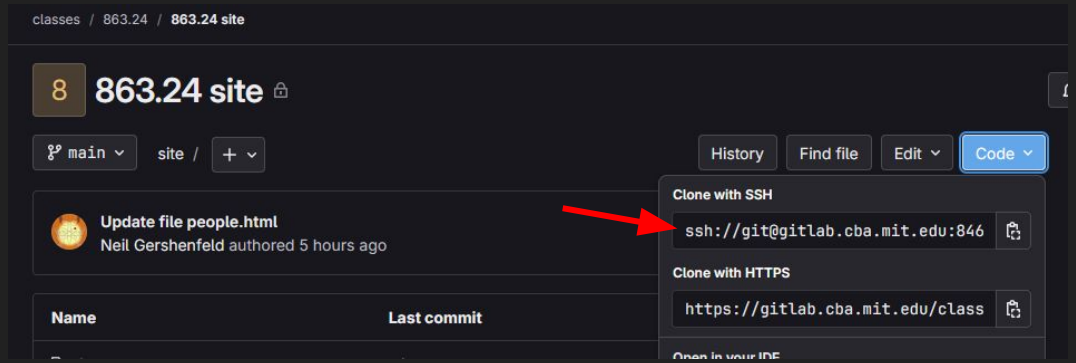


getting started (ssh keys)



getting started (git config)

```
$ git config --global user.name "$yourname"
$ git config --global user.email "$email"
$ git clone $REPO_URL
```



first commit

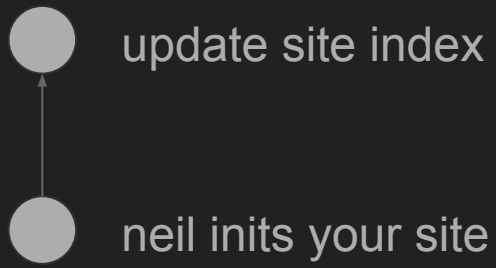
```
$ vim index.html
<!DOCTYPE html>
<html>
  <head></head>
  <body>I edited my site!</body>
</html>
```

first commit

```
$ git add index.html

$ git commit -m 'update site index'

$ git push

# go look at your site!

# (may take a minute to build)
```

people page ->

commit graph



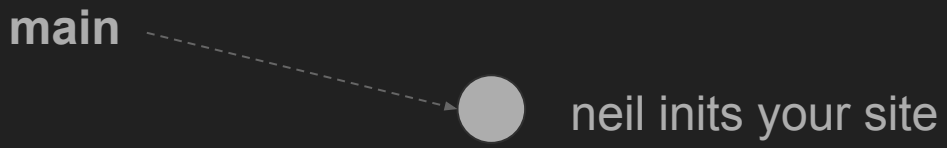
staging changes

```
$ git add index.html; touch ignored_file
$ git status
On branch master
Changes to be committed:
      modified: index.html
Untracked files:
      ignored_file
```

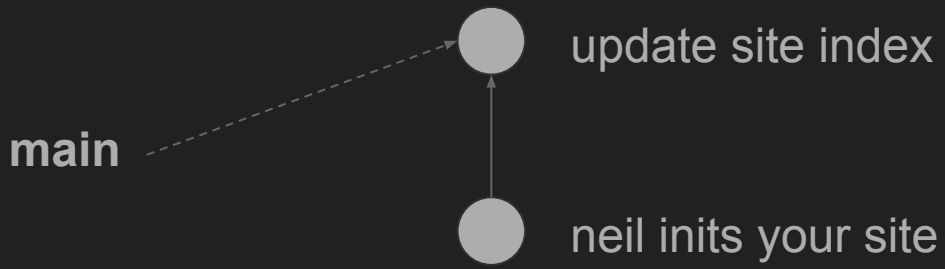
branches

```
$ git branch
* main
```

commit graph



commit graph



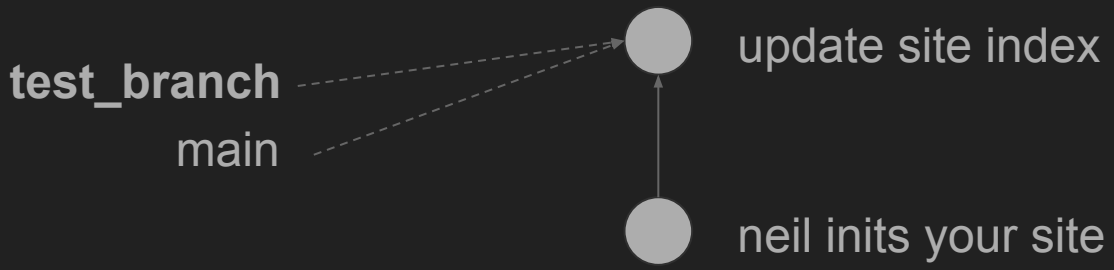
branches

```
$ git branch
* main
```

branches

```
$ git branch
* main
$ git checkout -b test_branch
$ git branch
* test_branch
  main
```

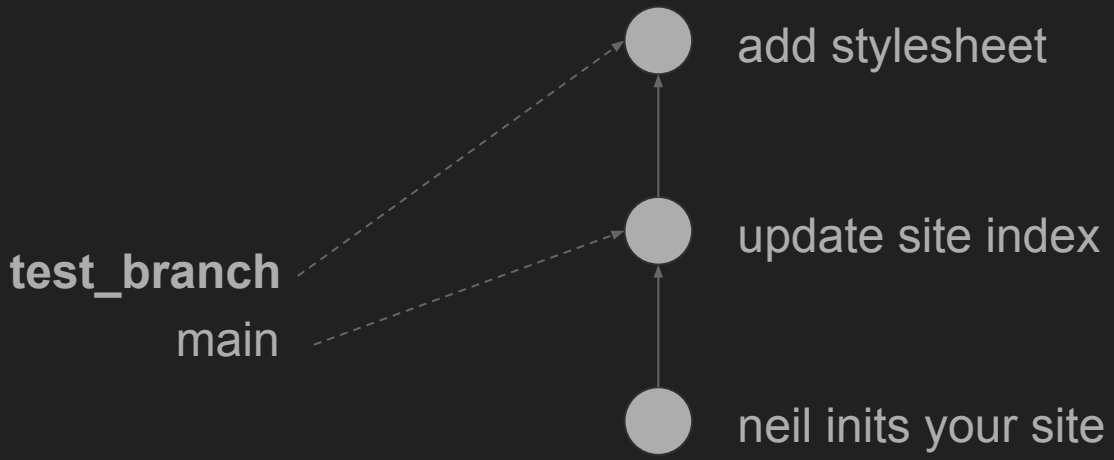
commit graph



branches

```
$ touch style.css

$ git add style.css

$ git commit -m 'add stylesheet'
```

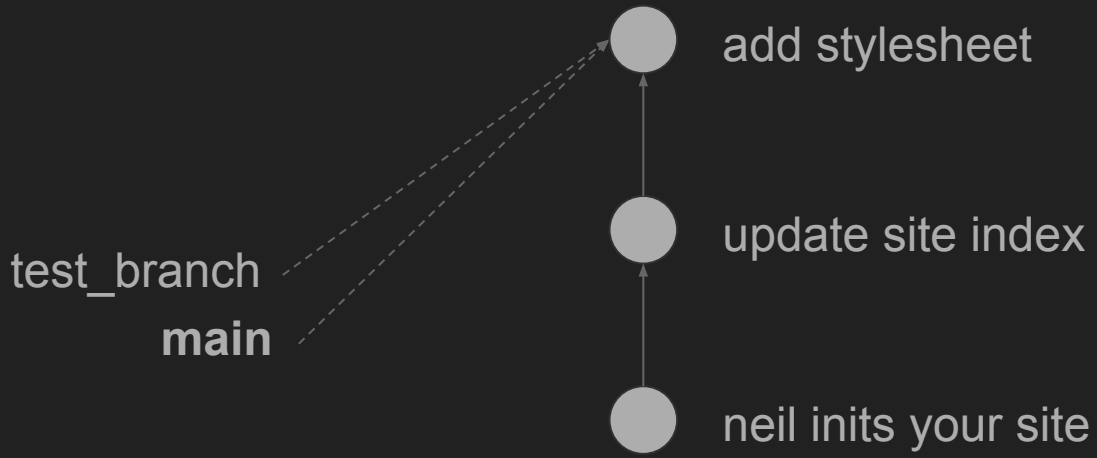
commit graph



branches

```
$ touch style.css

$ git add style.css

$ git commit -m 'add stylesheet'

$ git checkout main

$ git merge test_branch
```

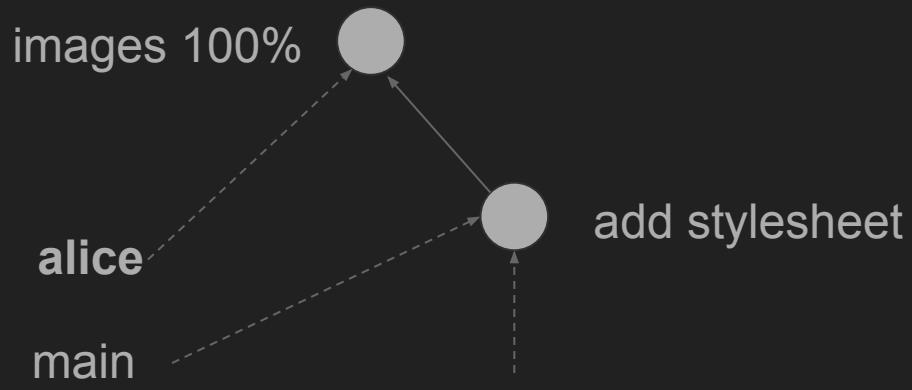
commit graph



branches (merge conflict)

```
$ git checkout -b alice

$ echo 'img { width: 100%; }' > style.css

$ git commit -m 'set images to 100% width'
```

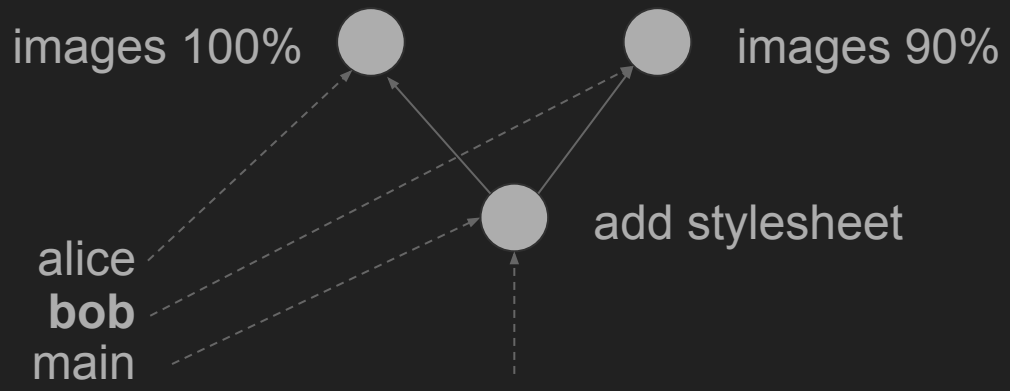
commit graph



branches (merge conflict)

```
$ git checkout -b alice
$ echo 'img { width: 100%; }' > style.css
$ git commit -m 'set images to 100% width'
$ git checkout main; git checkout -b bob
$ echo 'img { width: 90%; }' > style.css
$ git commit -m 'set images to 90% width'
```

commit graph



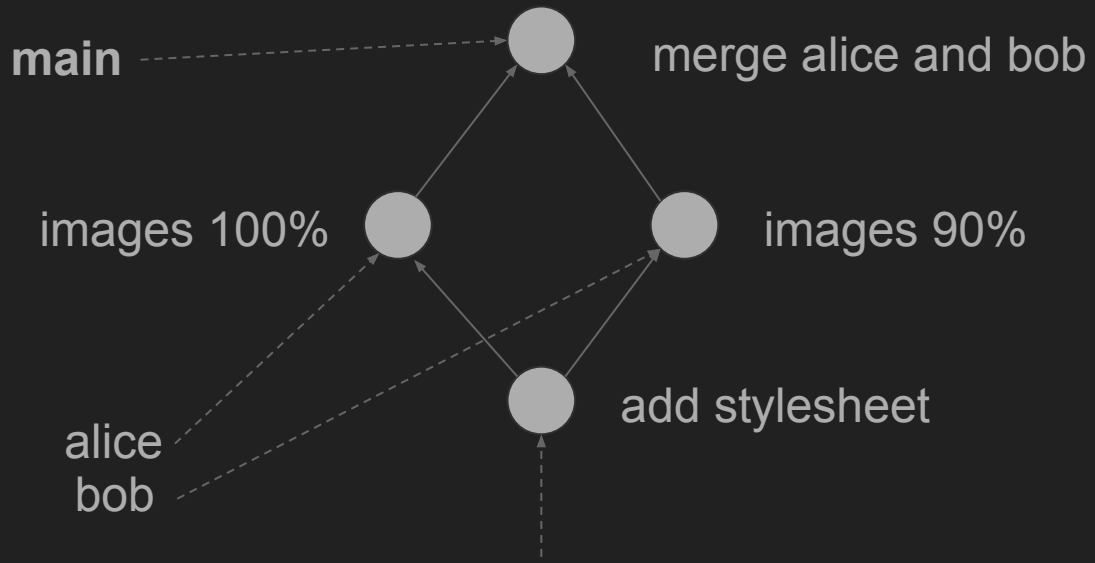
branches (merge conflict)

```
$ git merge alice

$ git merge bob

Auto-merging style.css

CONFLICT (content): Merge conflict in
style.css

Automatic merge failed; fix conflicts and
then commit the result.
```

merge conflict

```
$ cat style.css

<<<<<<< HEAD

img { width: 100%; }

=======

img { width: 90%; }

>>>>>>> bob
```

commit graph



git log --graph

```
$ git log --graph --oneline --all
*   40de3c5 (HEAD -> master) Merge branch 'bob'
|\
| * 07f637b (bob) images 90%
* | 6916dcc (alice) images 100%
|/
* 5b5d349 add stylesheet
* c3ad4e1 initial commit
```

git history browsers

gitk

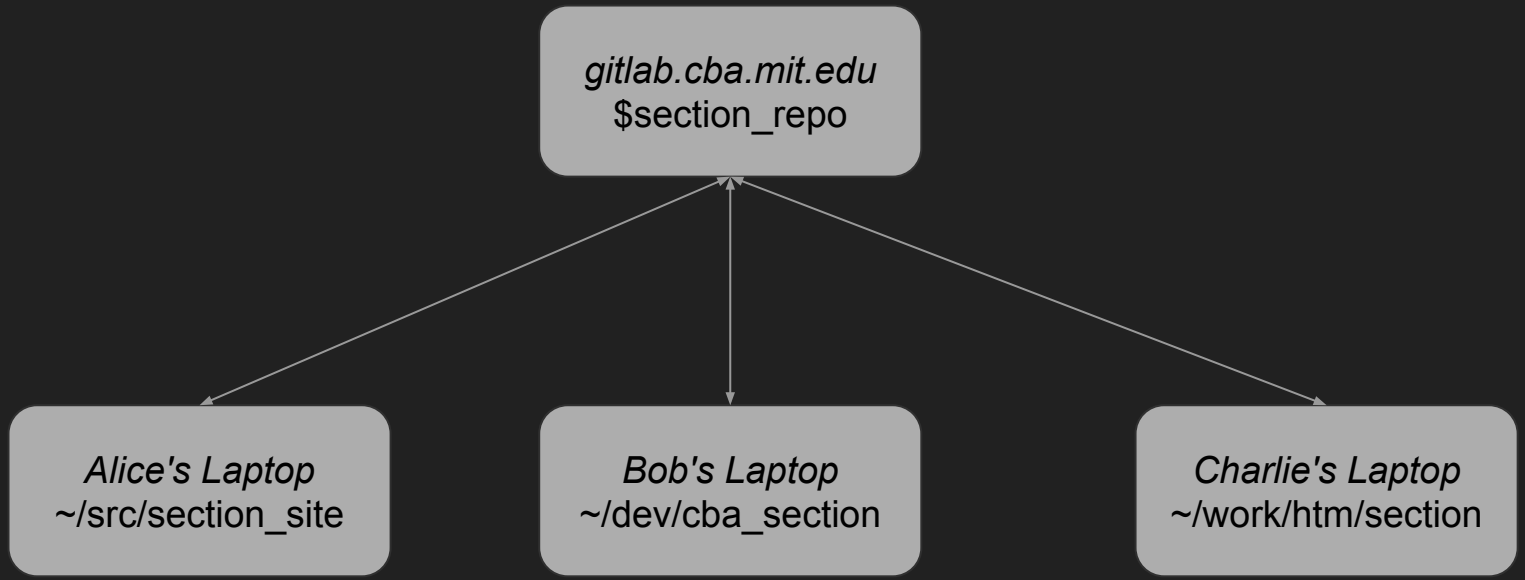
tig

gitkraken

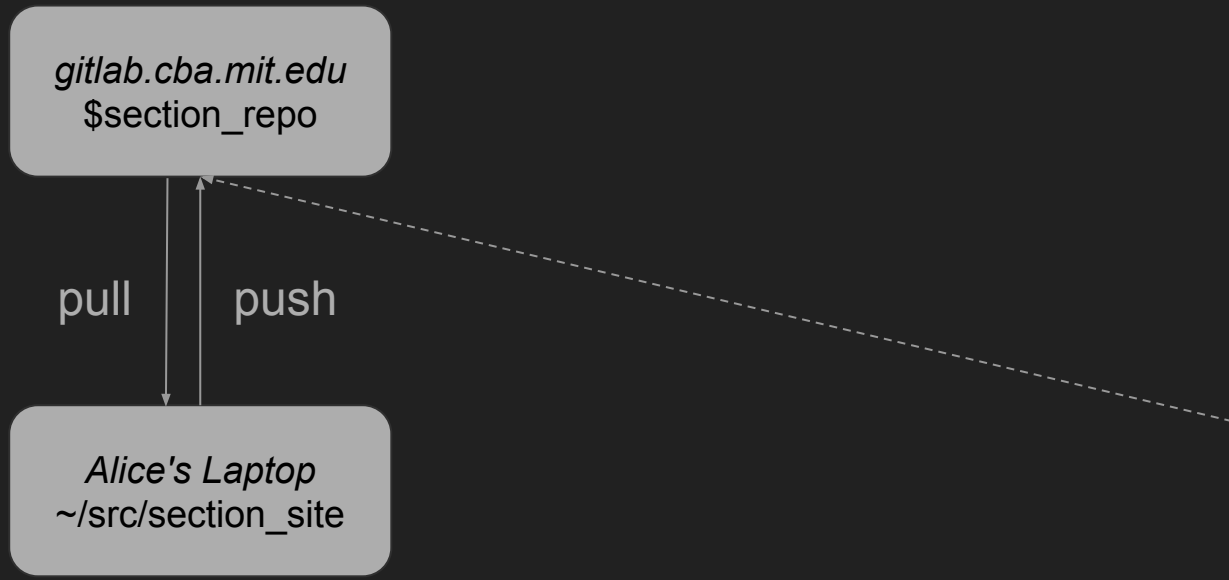
vs code

gitlab ui

remotes


gitlab.cba.mit.edu
$section_repo
Alice's Laptop
~/src/section_site
Bob's Laptop
~/dev/cba_section
Charlie's Laptop
~/work/htm/section

remotes

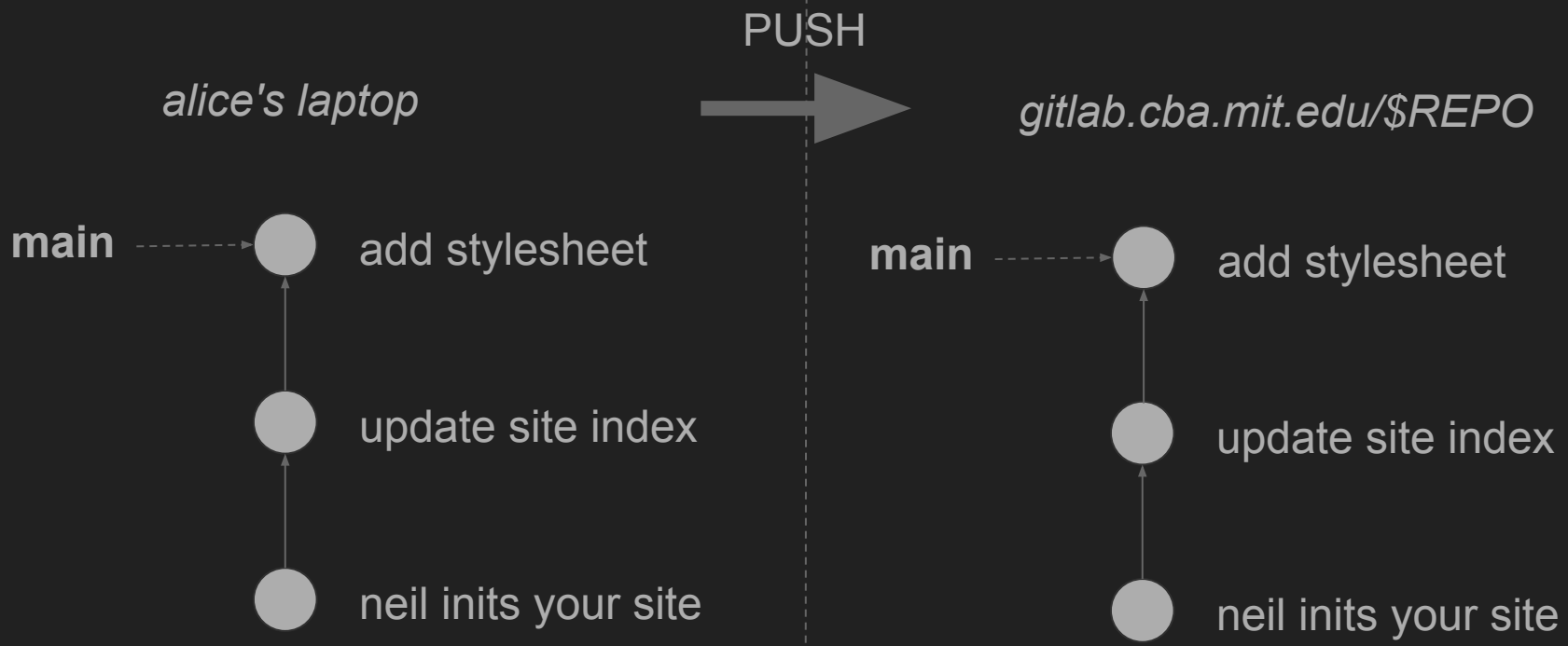

gitlab.cba.mit.edu
$section_repo
pull
push
Alice's Laptop
~/src/section_site

push

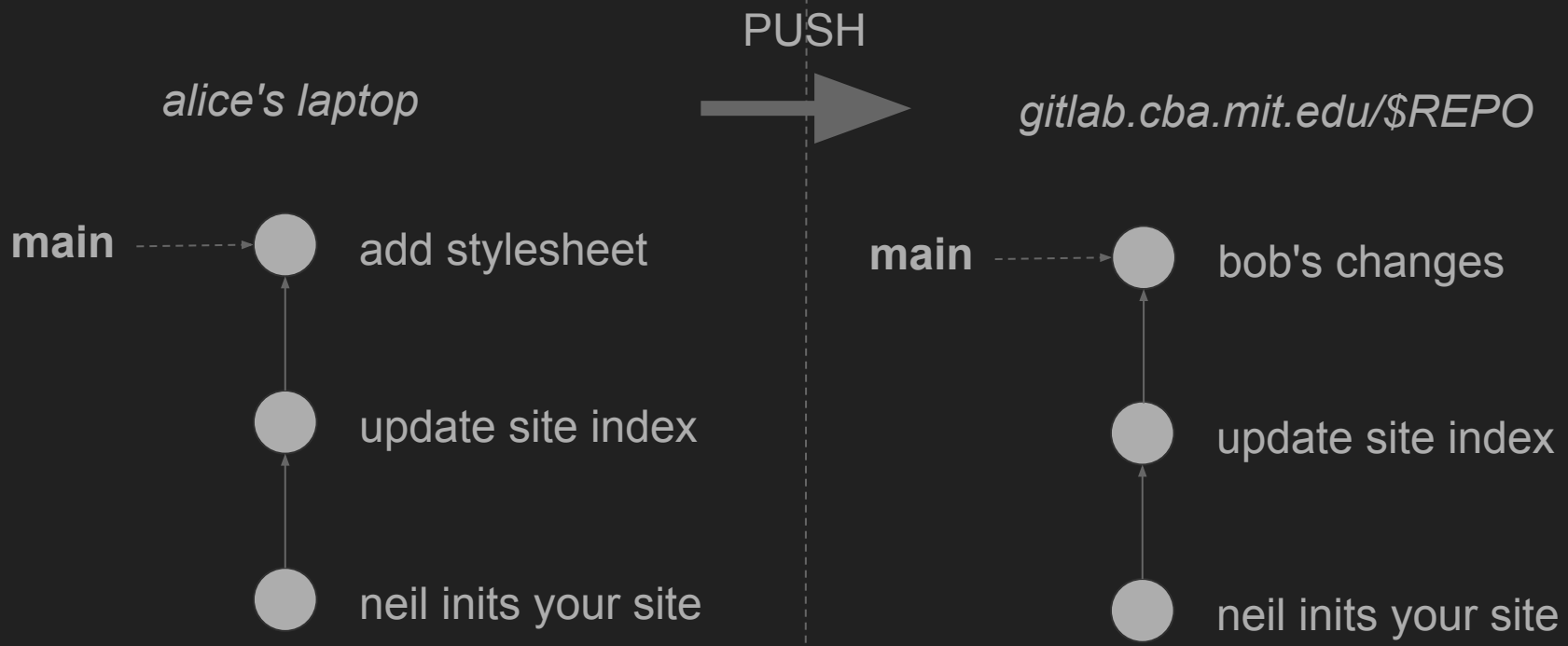
alice's laptop

gitlab.cba.mit.edu/$REPO

push


PUSH
alice's laptop
gitlab.cba.mit.edu/$REPO
main
add stylesheet
update site index
neil inits your site
main
add stylesheet
update site index
neil inits your site

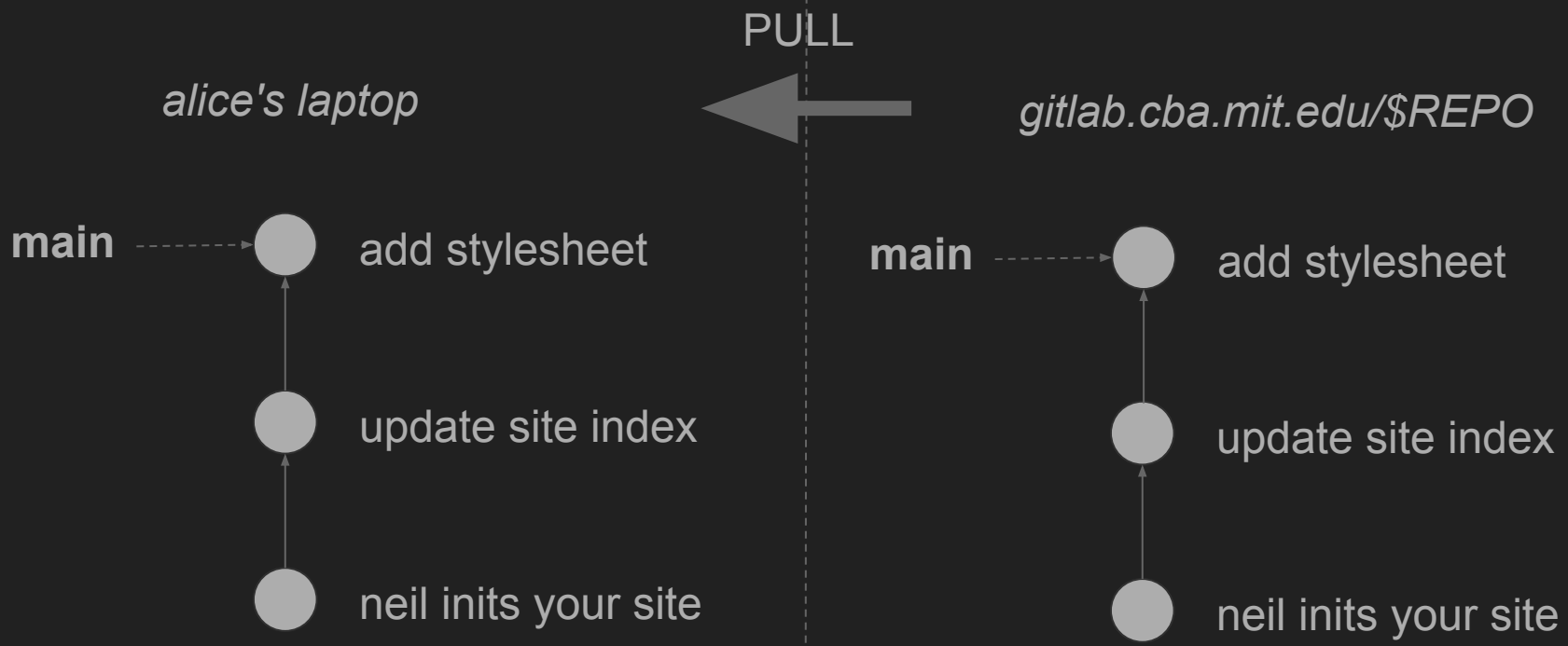
push (merge conflict)



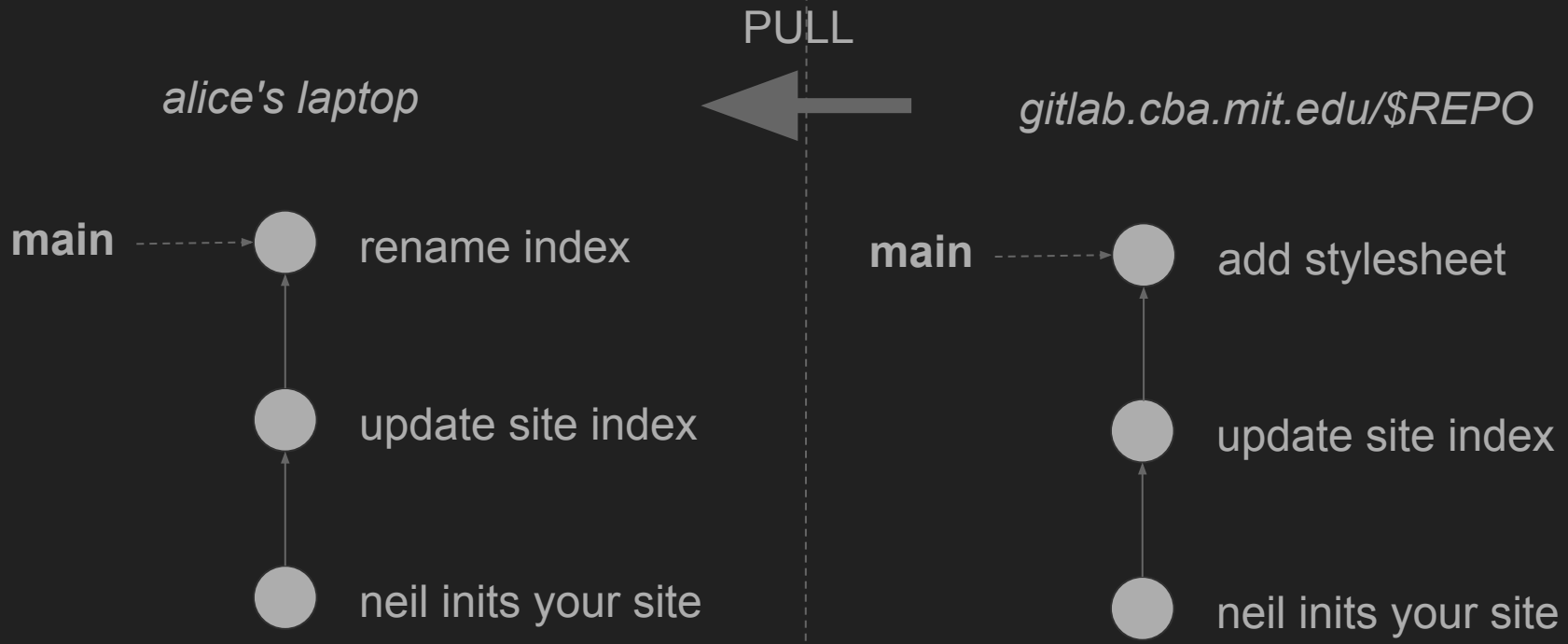
pull


alice's laptop
main
update site index
neil inits your site
gitlab.cba.mit.edu/$REPO
main
add stylesheet
update site index
neil inits your site

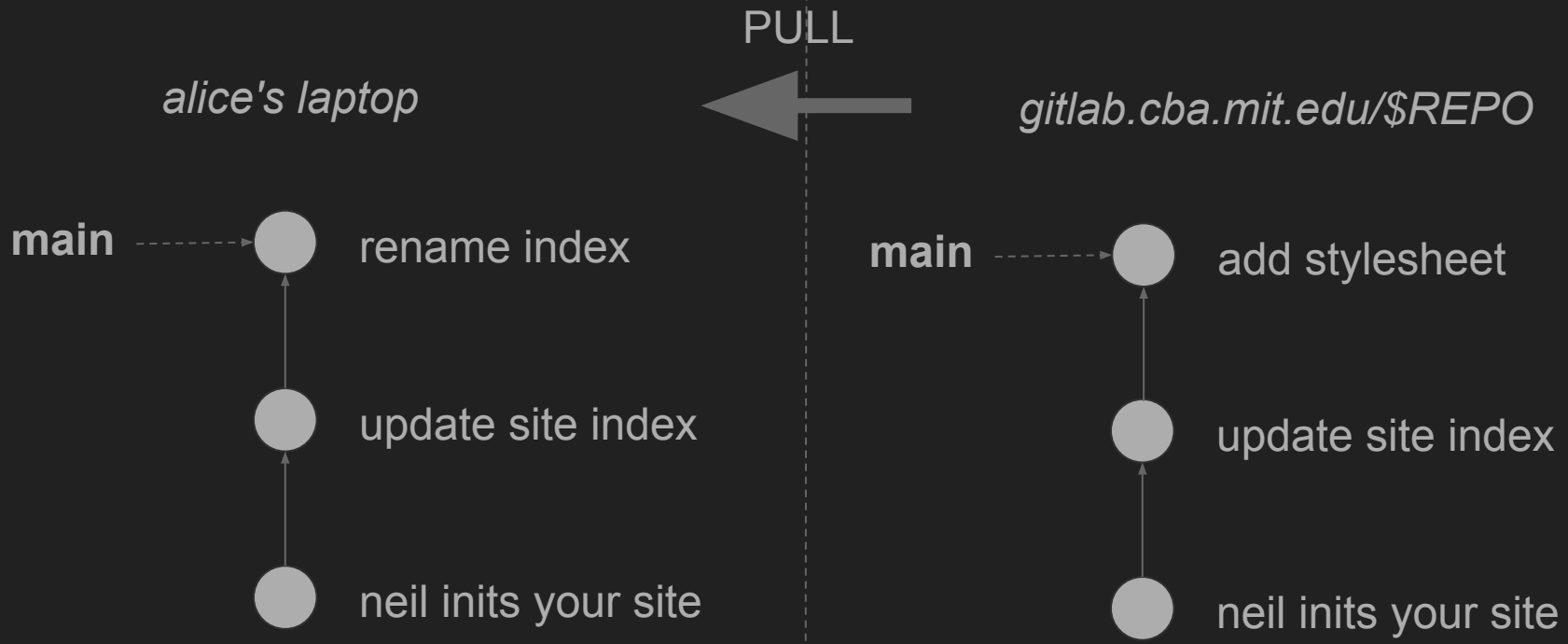
pull


PULL
alice's laptop
gitlab.cba.mit.edu/$REPO
main
add stylesheet
update site index
neil inits your site
main
add stylesheet
update site index
neil inits your site

pull (merge conflict)



pull (merge conflict)



troubleshooting

If you run into errors you don't know how to resolve, don't panic! It is hard to actually lose (committed) data in git.

Easiest ugly fix: reclone repo and copy files from broken one, recommit changes, push