GitHub

GitHub Developer Training

Presented by GitHub Professional Services

Brian Blackman

CSA – E

brianbla@Microsoft.com

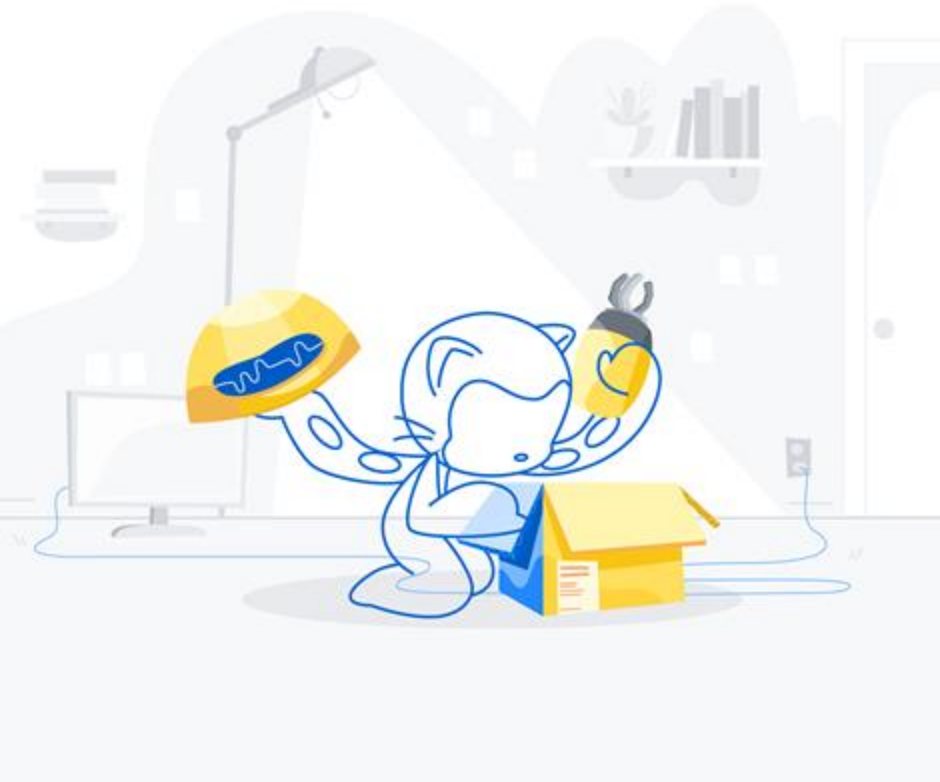
Microsoft

Objectives

- Have a hands-on training in how to **collaborate** effectively using the GitHub's toolset
- Understand how to **navigate** around GitHub
- Maximise **Git** and **GitHub** usage to streamline your development lifecycle
- Implement the **GitHub flow** in building software
- Resolve various types of **merge conflict** as they arise
- **Protect branches** with rules that work for your team using the **CODEOWNERS** file
- Mark moments in time within your repo using **tags** and **releases**
- **Rewrite the history** of your project in a way that suits you

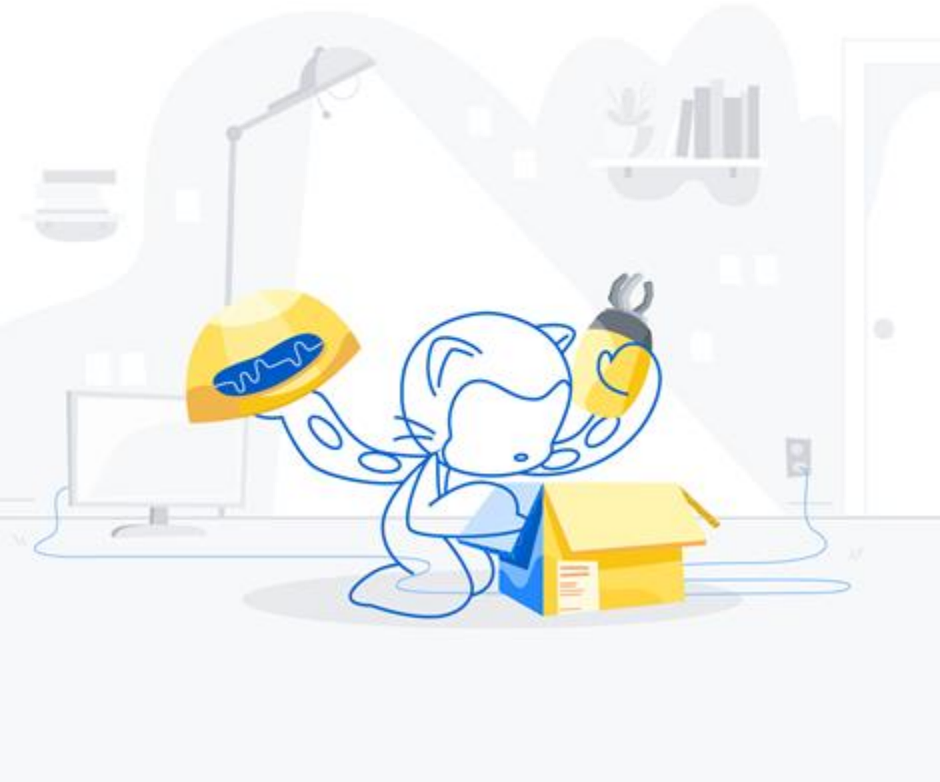
Agenda: Part 1

- **Getting started**
  - Getting ready for class
  - Getting started
  - GitHub flow
- **Project 1: caption this**
  - Branching with Git
  - Local Git Configs
  - Working locally
  - Collaborating on code
  - Editing on GitHub
  - Merging Pull Requests
  - Local History
  - Streamline workflow with aliases
- **Project 2: Merge Conflicts**
  - Defining a merge conflict
  - Resolving a merge conflict



Agenda: Part 2

- **Project 3: GitHub Games**
    - Workflow review
    - Protected branches & CODEOWNERS
    - Git bisect
    - Reverting commits
    - Helpful Git commands
    - Viewing local changes
    - Tags and releases
    - Workflow discussion
- **Project 4: Local repository**
    - Create a local repository
    - Fixing commit mistakes
    - Git reset
    - Cherry picking
    - Merge strategies



Getting ready for class

- User account
- Install Git
- Cloning and proxy configuration
- Personal Access Tokens

Setting up a GitHub.com user account

1. Access GitHub.com and click Sign up.
2. Choose the free account.
3. You will receive a verification email at the address provided.
4. Click the link to complete the verification process.

Install Git

- `git --version`
- If you don't already have Git installed, you can download Git at www.git-scm.com.

Cloning and proxy configuration

- `git clone https://github.com/githubschool/scratch`

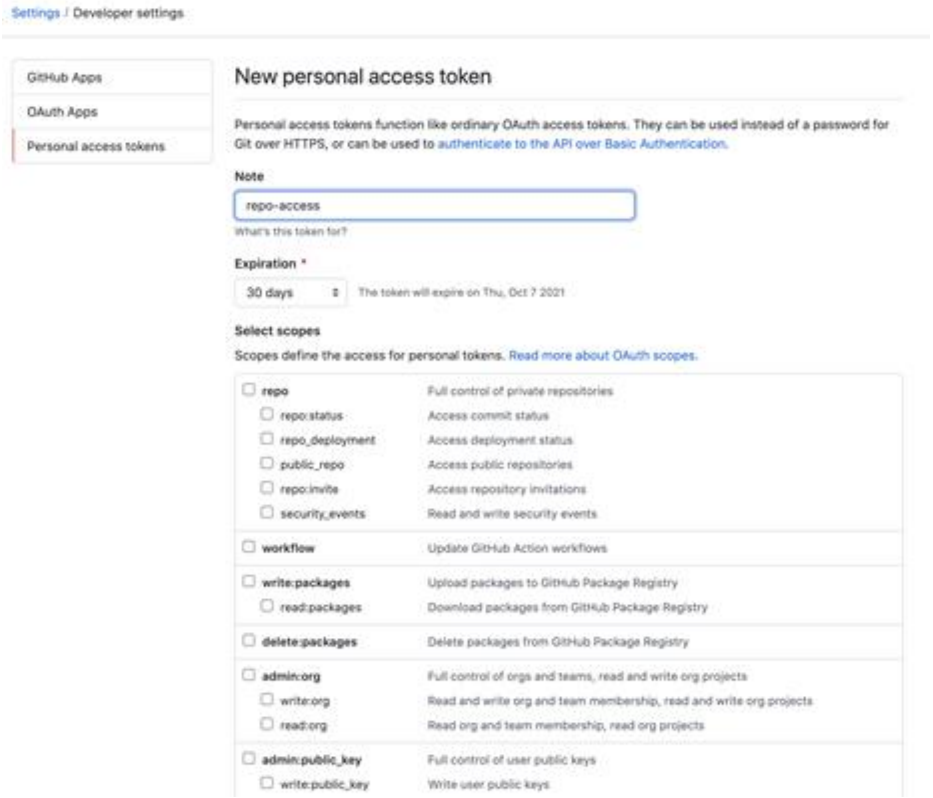
Proxy configuration if no authentication required

- `git config --global http.proxy https://YOUR.PROXY.SERVER:8080`

Proxy configuration if authentication required

- `git config --global http.proxy https://YOUR_PROXY_USERNAME:YOUR_PROXY_PASSWORD@YOUR.PROXY.SERVER:8080`

Personal Access Tokens



New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Note

repo-access

What's this token for?

Expiration *

30 days

The token will expire on Thu, Oct 7 2021

Select scopes

Scopes define the access for personal tokens. Read more about OAuth scopes.

□ **repo**	Full control of private repositories
□ repo:status	Access commit status
□ repo_deployment	Access deployment status
□ public_repo	Access public repositories
□ repo:invite	Access repository invitations
□ security_events	Read and write security events
□ **workflow**	Update GitHub Action workflows
□ **write:packages**	Upload packages to GitHub Package Registry
□ read:packages	Download packages from GitHub Package Registry
□ **delete:packages**	Delete packages from GitHub Package Registry
□ **admin:org**	Full control of orgs and teams, read and write org projects
□ write:org	Read and write org and team membership, read and write org projects
□ read:org	Read org and team membership, read org projects
□ **admin:public_key**	Full control of user public keys
□ write:public_key	Write user public keys

Demo

Getting started

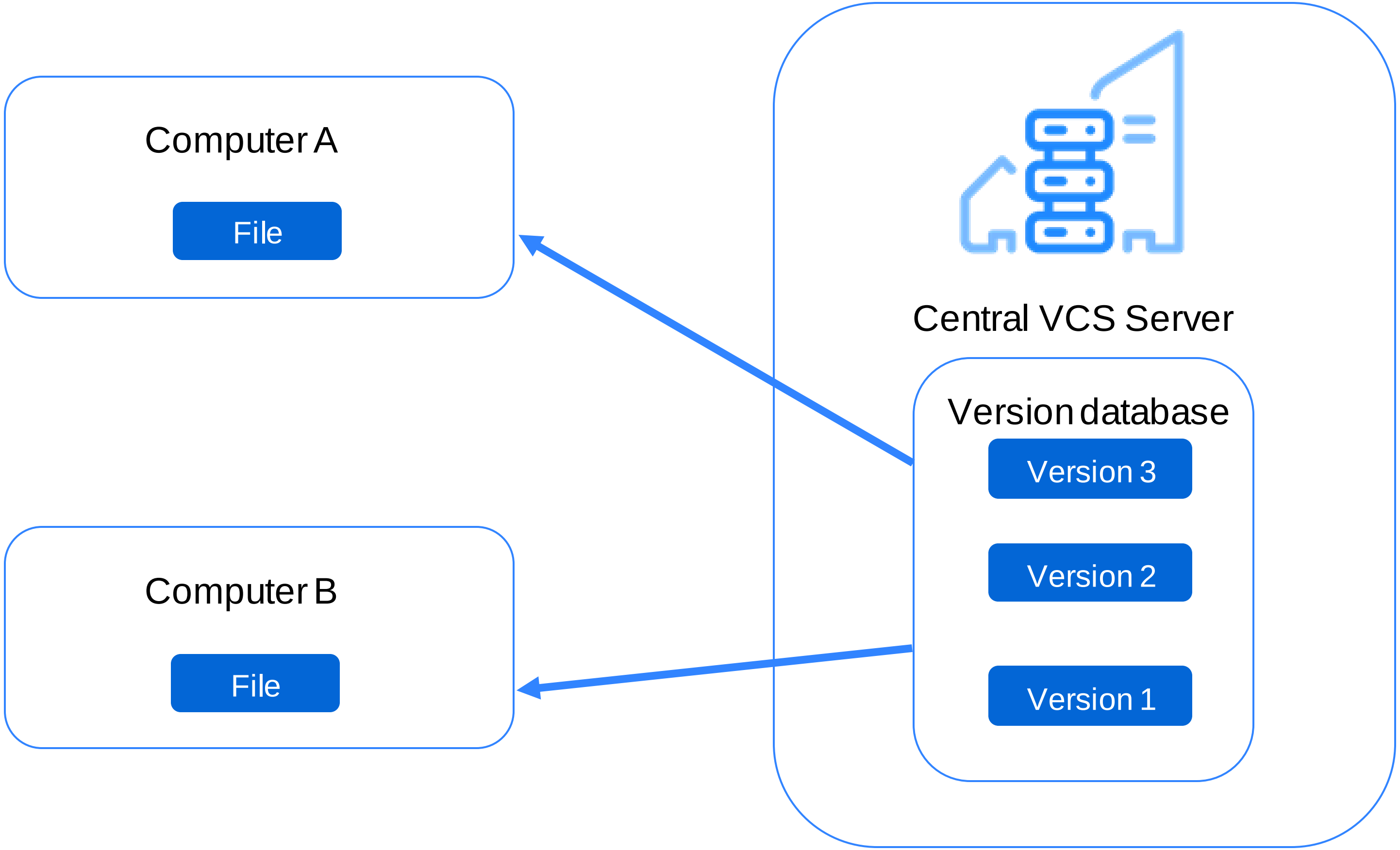
Version control systems ● Git ● GitHub

What is version control?

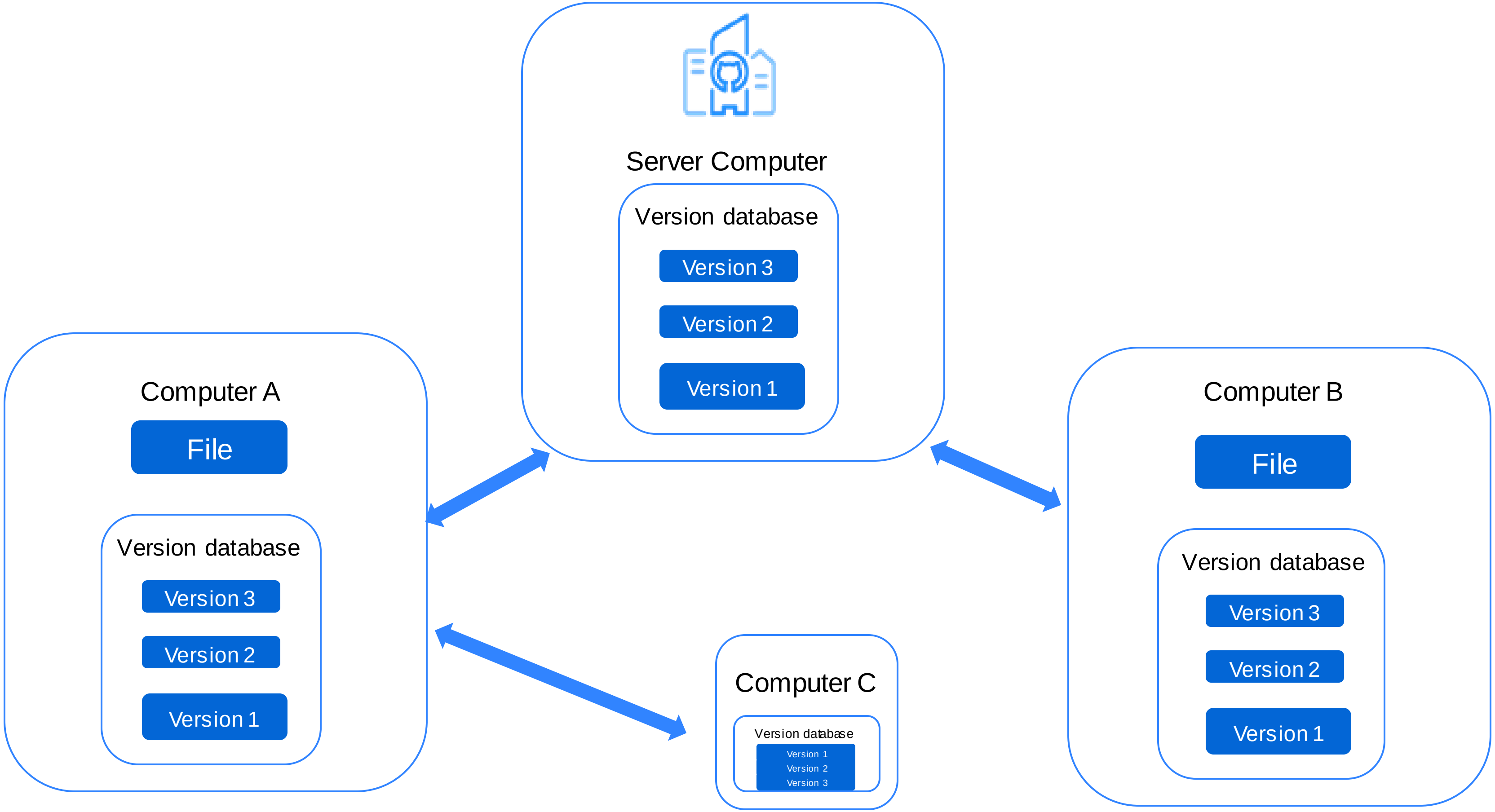
“Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.”

https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control

Centralized Version Control



Distributed Version Control



Server Computer

Version database

Version 3

Version 2

Version 1

Computer A

File

Version database

Version 3

Version 2

Version 1

Computer B

File

Version database

Version 3

Version 2

Version 1

Computer C

Version database

Version 1

Version 2

Version 3

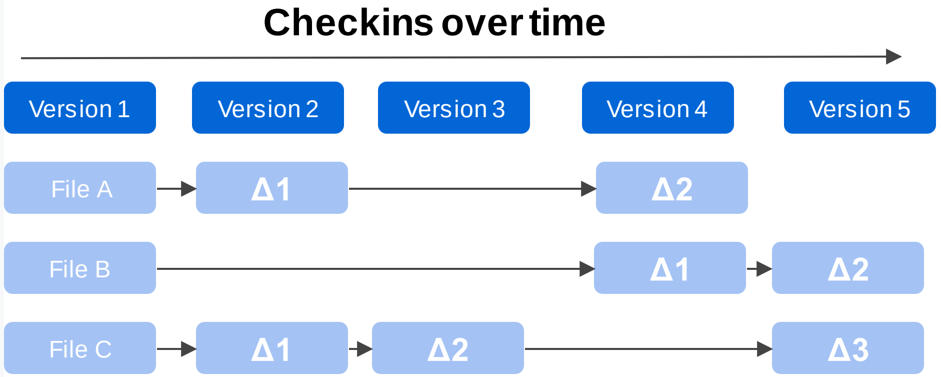
Getting started

Version control systems ● Git ● GitHub

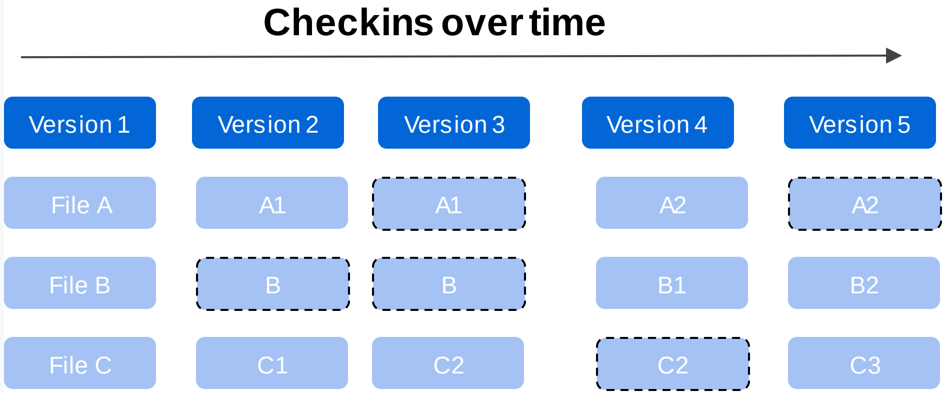
git

- A **distributed** version control system (DVCS)
- **Free** and **open source**
- Designed to handle everything from **small** to very **large** projects with speed and efficiency
- Both small in size and **fast**
- Optimized for **local operations**
- **Branching**
- **Snapshots**, not deltas

Snapshots vs deltas



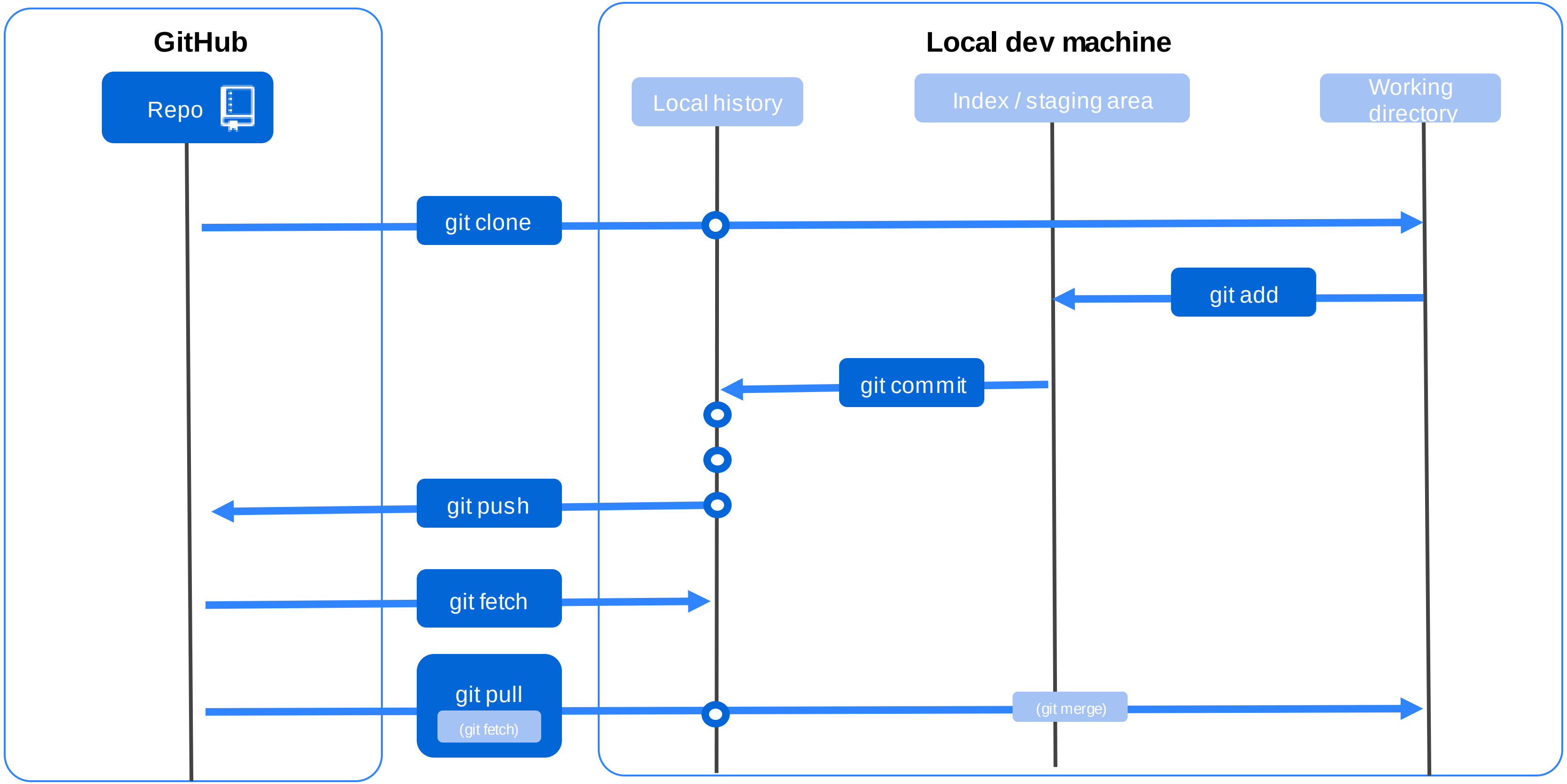
Centralised Version Control System



Distributed Version Control System

Image source: ProGit v2 by Scott Chacon

How Git interacts with GitHub



Getting started

- Version control systems
- Git
- GitHub
  - Ecosystem
  - Repository navigation

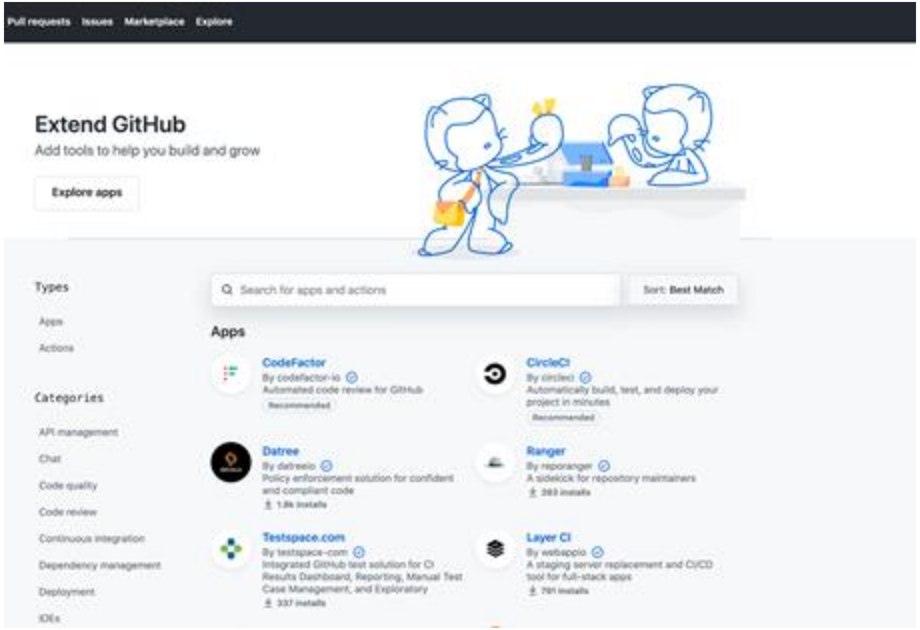
GitHub

- Technology platform built **on top** of **DVCS**
- **Social** side of Git
- **Code hosting** and **collaboration** platform
- **Community** of developers
- **Ecosystem** of integrations
- **Team** collaboration through:
    - **Issues**
    - **Pull requests**
    - **Organisation** and **teams**
    - **Projects**

Getting started

- Version control systems
- Git
- GitHub
  - Ecosystem
  - Repository navigation

The GitHub ecosystem



Extend GitHub

Add tools to help you build and grow

Apps

CodeFactor
By codefactor-io
Automated code review for GitHub

CircleCI
By circleci
Automatically build, test, and deploy your project in minutes

Datree
By datreeio
Policy enforcement solution for confident and compliant code
1.8k installs

Ranger
By reporanger
A sidekick for repository maintainers
283 installs

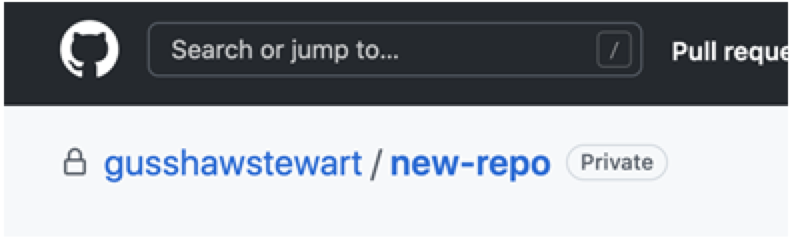
Testspace.com
By testspace-com
Integrated GitHub test solution for CI Results Dashboard, Reporting, Manual Test Case Management, and Exploratory
337 installs

Layer CI
By webappio
A staging server replacement and CI/CD tool for full-stack apps
781 installs

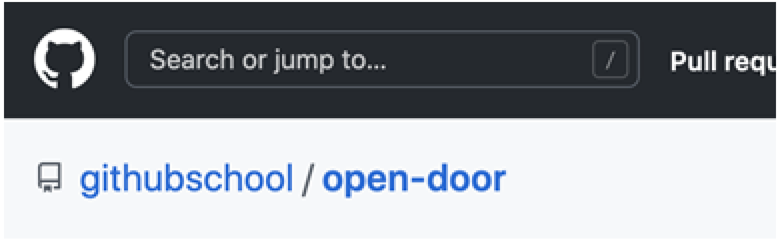
Getting started

- Version control systems
- Git
- GitHub
  - Ecosystem
  - Repository navigation

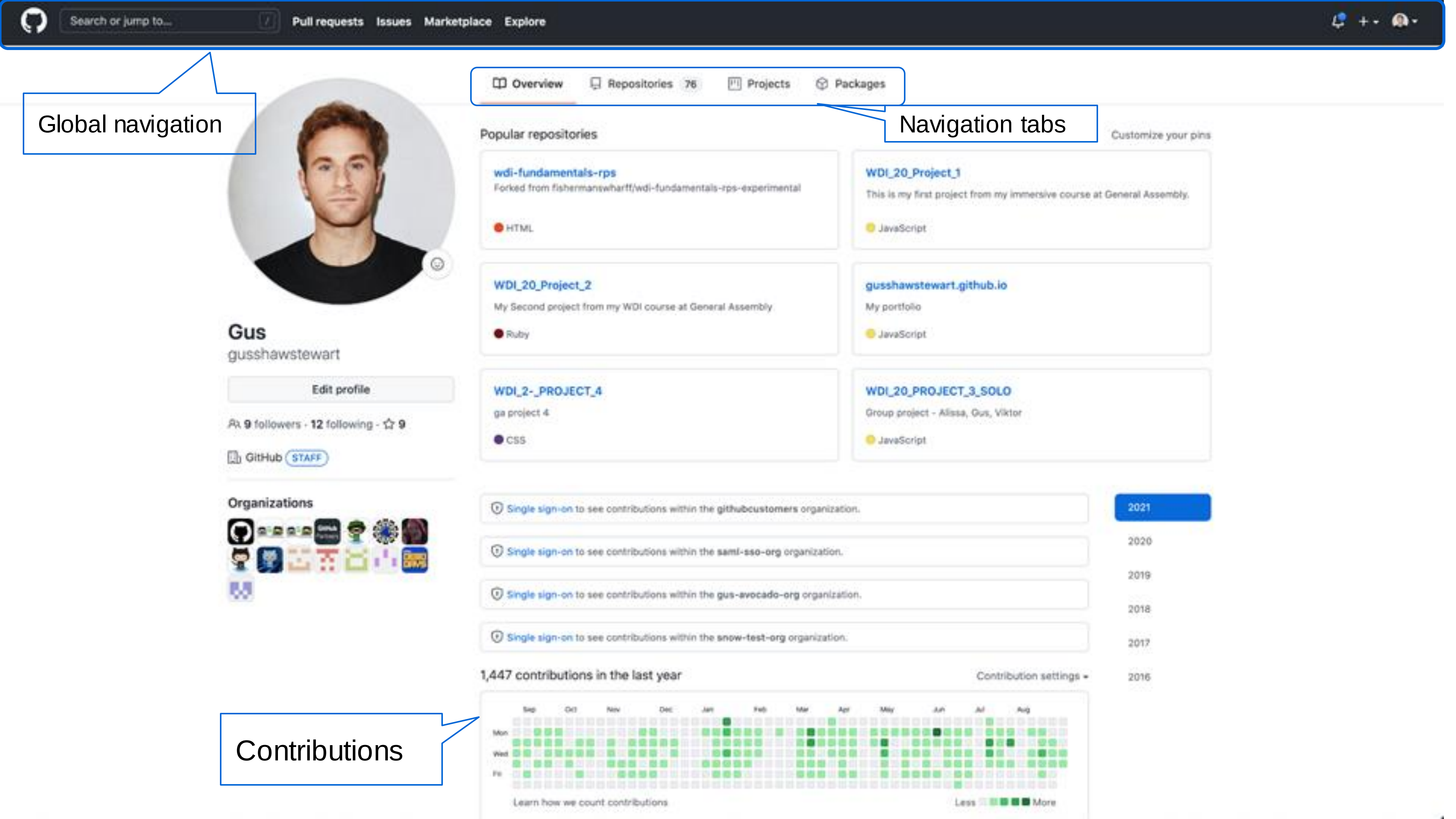
User accounts vs organization accounts



gusshawstewart / new-repo Private



githubschool / open-door



Global navigation

Navigation tabs

Contributions

Search or jump to...

Pull requests Issues Marketplace Explore

Overview | Repositories 76 | Projects | Packages

Gus

gusshawstewart

Edit profile

9 followers · 12 following · ☆ 9

GitHub STAFF

Organizations

Popular repositories

Customize your pins

wdi-fundamentals-rps
Forked from fishermanswharff/wdi-fundamentals-rps-experimental
● HTML

WDI_20_Project_1
This is my first project from my immersive course at General Assembly.
● JavaScript

WDI_20_Project_2
My Second project from my WDI course at General Assembly
● Ruby

gusshawstewart.github.io
My portfolio
● JavaScript

WDI_2-_PROJECT_4
ga project 4
● CSS

WDI_20_PROJECT_3_SOLO
Group project - Alissa, Gus, Viktor
● JavaScript

Single sign-on to see contributions within the **githubcustomers** organization.

Single sign-on to see contributions within the **saml-sso-org** organization.

Single sign-on to see contributions within the **gus-avocado-org** organization.

Single sign-on to see contributions within the **snow-test-org** organization.

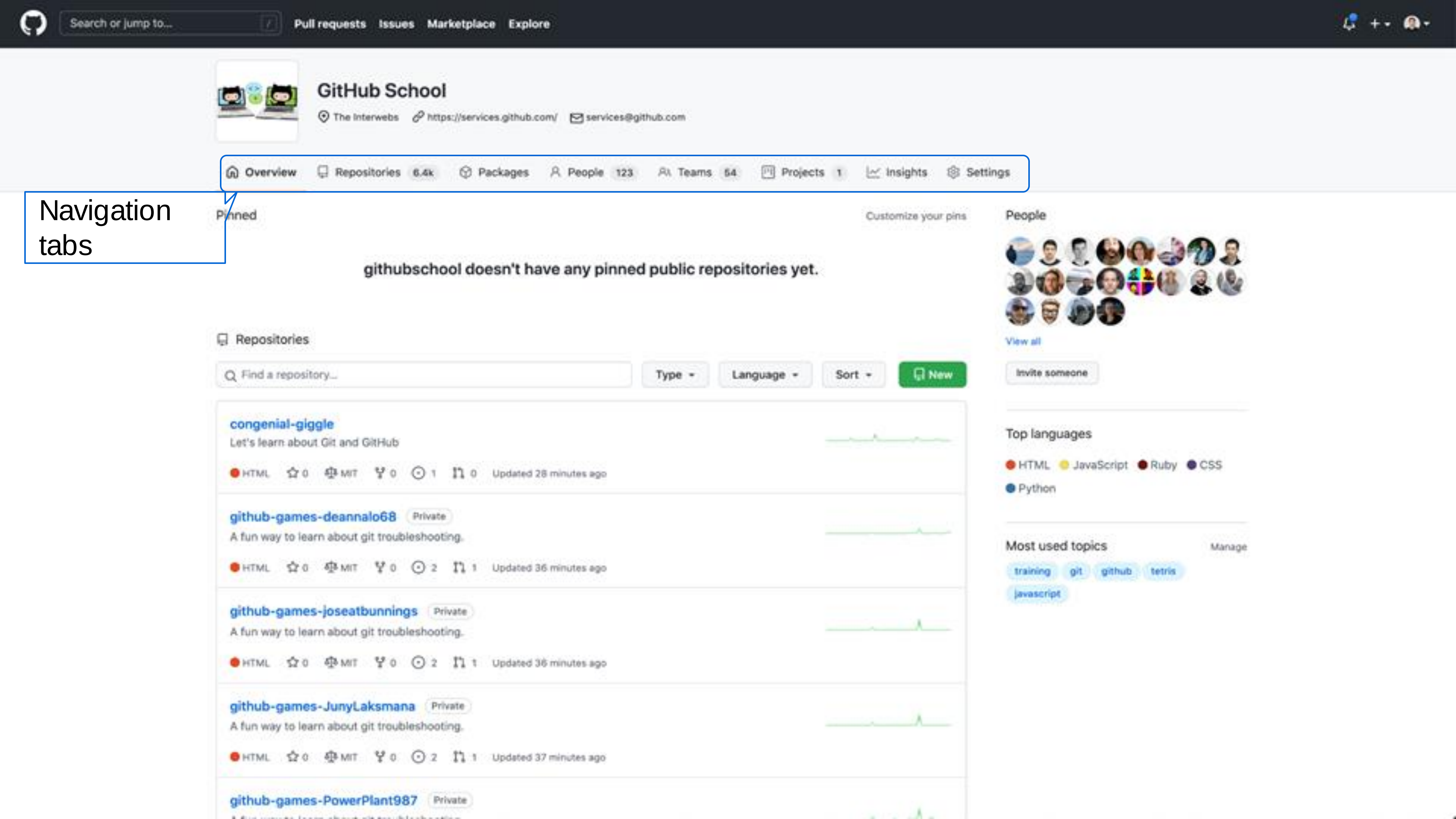
2021
2020
2019
2018
2017
2016

1,447 contributions in the last year

Contribution settings ▾

Learn how we count contributions

Less 🟩🟩🟩🟩 More



Navigation tabs

Search or jump to...

Pull requests Issues Marketplace Explore

GitHub School

The Interwebs https://services.github.com/ services@github.com

Overview | Repositories 6.4k | Packages | People 123 | Teams 54 | Projects 1 | Insights | Settings

Pinned

Customize your pins

githubschool doesn't have any pinned public repositories yet.

Repositories

Find a repository...

Type ▾ Language ▾ Sort ▾ New

congenial-giggle

Let's learn about Git and GitHub

● HTML ☆ 0 MIT 0 1 0 Updated 28 minutes ago

github-games-deannalo68 Private

A fun way to learn about git troubleshooting.

● HTML ☆ 0 MIT 0 2 1 Updated 36 minutes ago

github-games-joseatbunnings Private

A fun way to learn about git troubleshooting.

● HTML ☆ 0 MIT 0 2 1 Updated 36 minutes ago

github-games-JunyLaksmana Private

A fun way to learn about git troubleshooting.

● HTML ☆ 0 MIT 0 2 1 Updated 37 minutes ago

github-games-PowerPlant987 Private

People

View all

Invite someone

Top languages

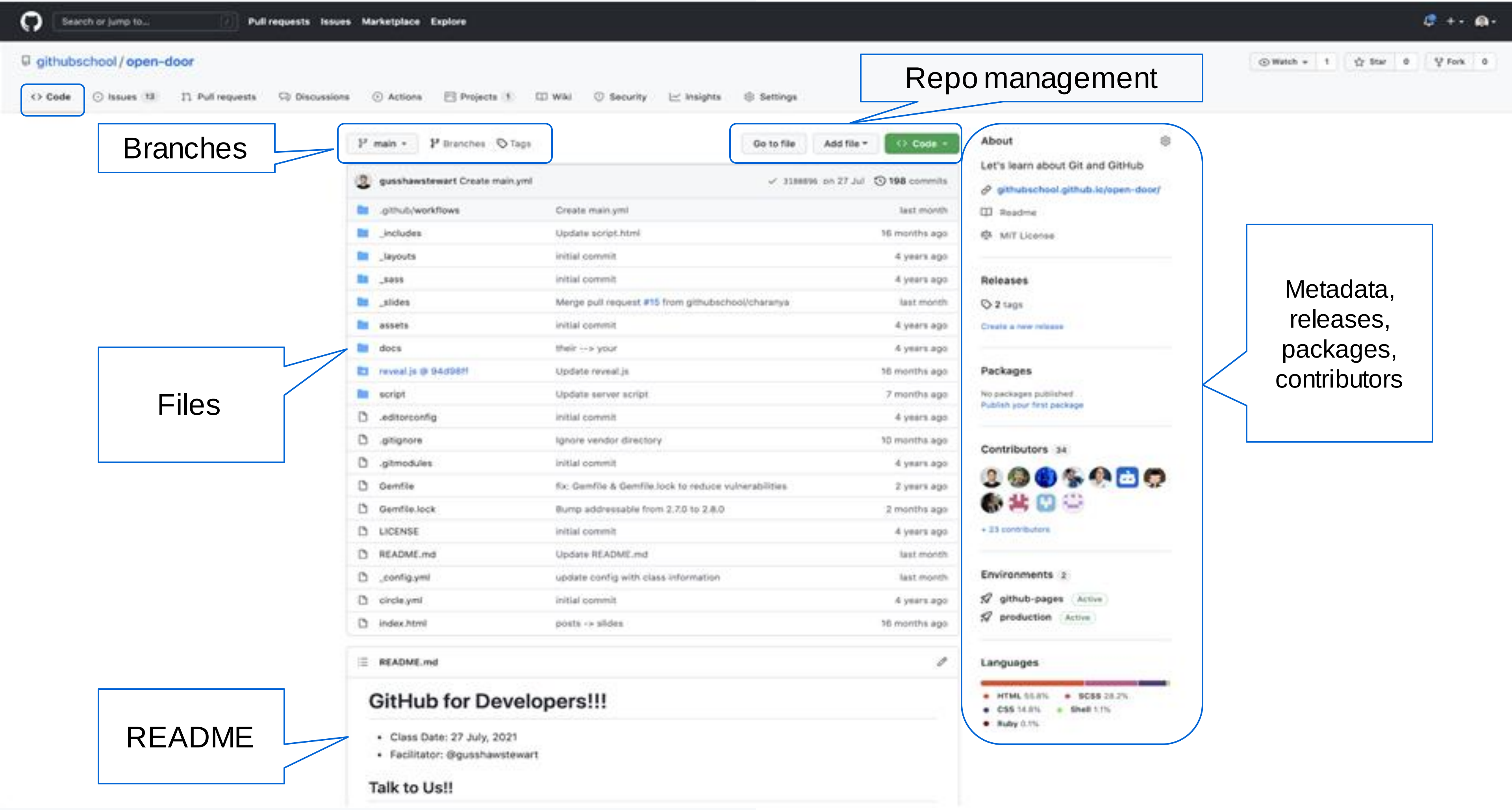
● HTML ● JavaScript ● Ruby ● CSS ● Python

Most used topics

Manage

training git github tetris javascript

Demo



Repo management

Branches

Files

README

Metadata, releases, packages, contributors

gusshawstewart Create main.yml ✓ 3188896 on 27 Jul 198 commits

.github/workflows	Create main.yml	last month
_includes	Update script.html	16 months ago
_layouts	initial commit	4 years ago
_sass	initial commit	4 years ago
_slides	Merge pull request #15 from githubschool/charanya	last month
assets	initial commit	4 years ago
docs	their --> your	4 years ago
reveal.js @ 94d98ff	Update reveal.js	16 months ago
script	Update server script	7 months ago
.editorconfig	initial commit	4 years ago
.gitignore	ignore vendor directory	10 months ago
.gitmodules	initial commit	4 years ago
Gemfile	fix: Gemfile & Gemfile.lock to reduce vulnerabilities	2 years ago
Gemfile.lock	Bump addressable from 2.7.0 to 2.8.0	2 months ago
LICENSE	initial commit	4 years ago
README.md	Update README.md	last month
_config.yml	update config with class information	last month
circle.yml	initial commit	4 years ago
index.html	posts -> slides	16 months ago

README.md

GitHub for Developers!!!

- Class Date: 27 July, 2021
- Facilitator: @gusshawstewart

Talk to Us!!

About

Let's learn about Git and GitHub

githubschool.github.io/open-door/

Readme

MIT License

Releases

2 tags

Create a new release

Packages

No packages published
Publish your first package

Contributors 34

+ 23 contributors

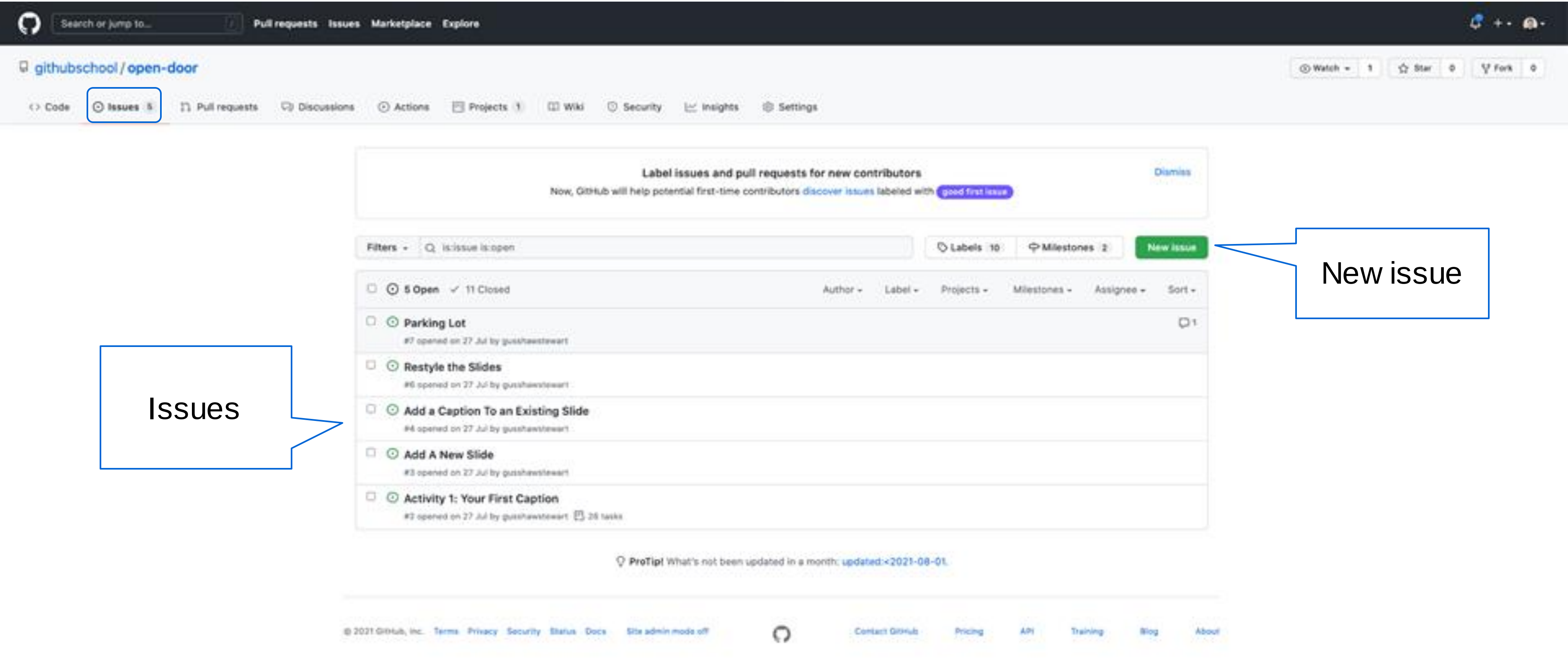
Environments 2

github-pages Active

production Active

Languages

- HTML 55.8%
- SCSS 28.2%
- CSS 14.8%
- Shell 1.1%
- Ruby 0.1%



Label issues and pull requests for new contributors

Now, GitHub will help potential first-time contributors discover issues labeled with good first issue

○ 5 Open ✓ 11 Closed

□ ○ **Parking Lot** 1
#7 opened on 27 Jul by gusshawstewart

□ ○ **Restyle the Slides**
#6 opened on 27 Jul by gusshawstewart

□ ○ **Add a Caption To an Existing Slide**
#4 opened on 27 Jul by gusshawstewart

□ ○ **Add A New Slide**
#3 opened on 27 Jul by gusshawstewart

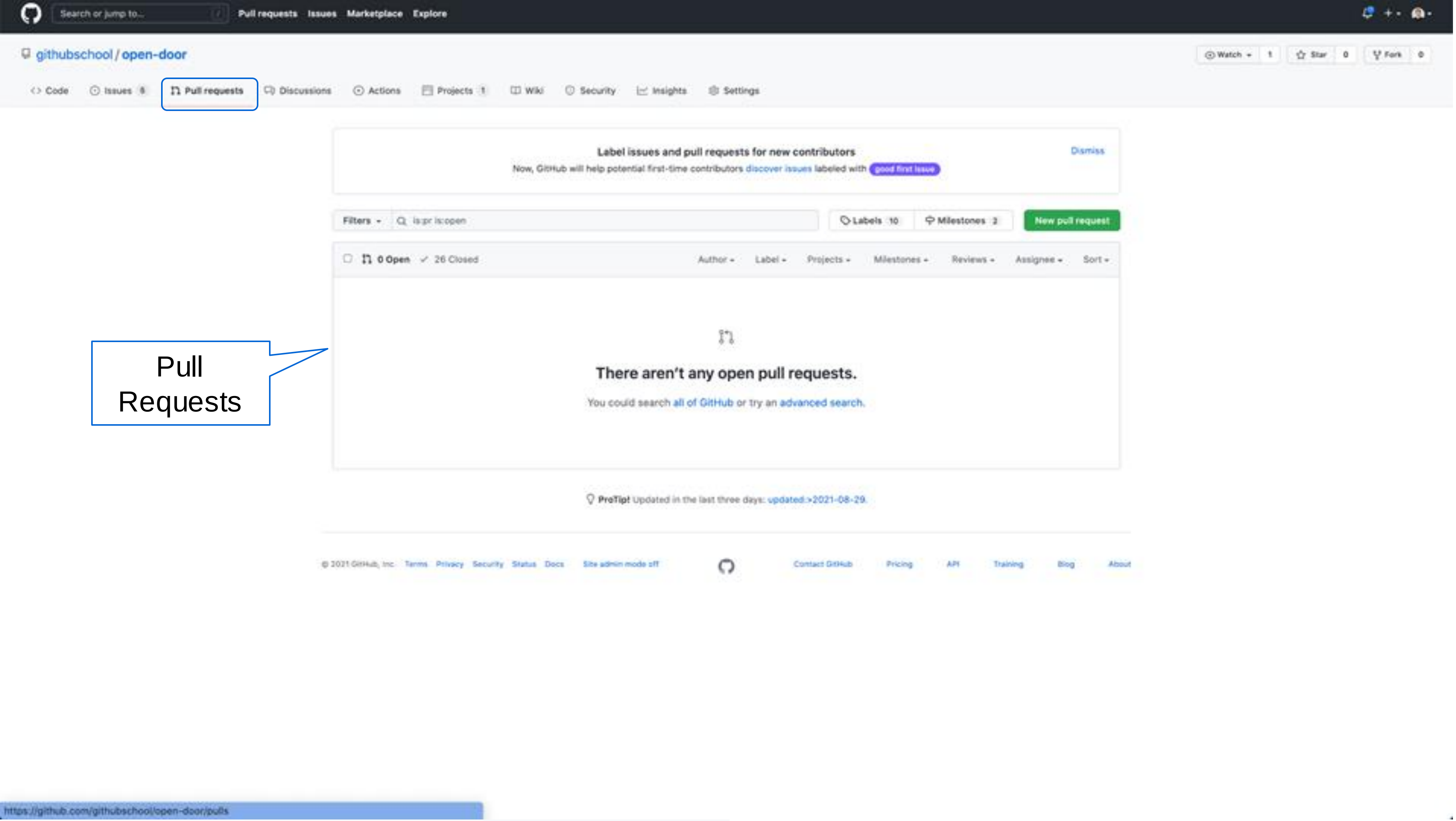
□ ○ **Activity 1: Your First Caption**
#2 opened on 27 Jul by gusshawstewart 26 tasks

ProTip! What's not been updated in a month: updated:<2021-08-01.

© 2021 GitHub, Inc. Terms Privacy Security Status Docs Site admin mode off Contact GitHub Pricing API Training Blog About

New issue

Issues



Label issues and pull requests for new contributors

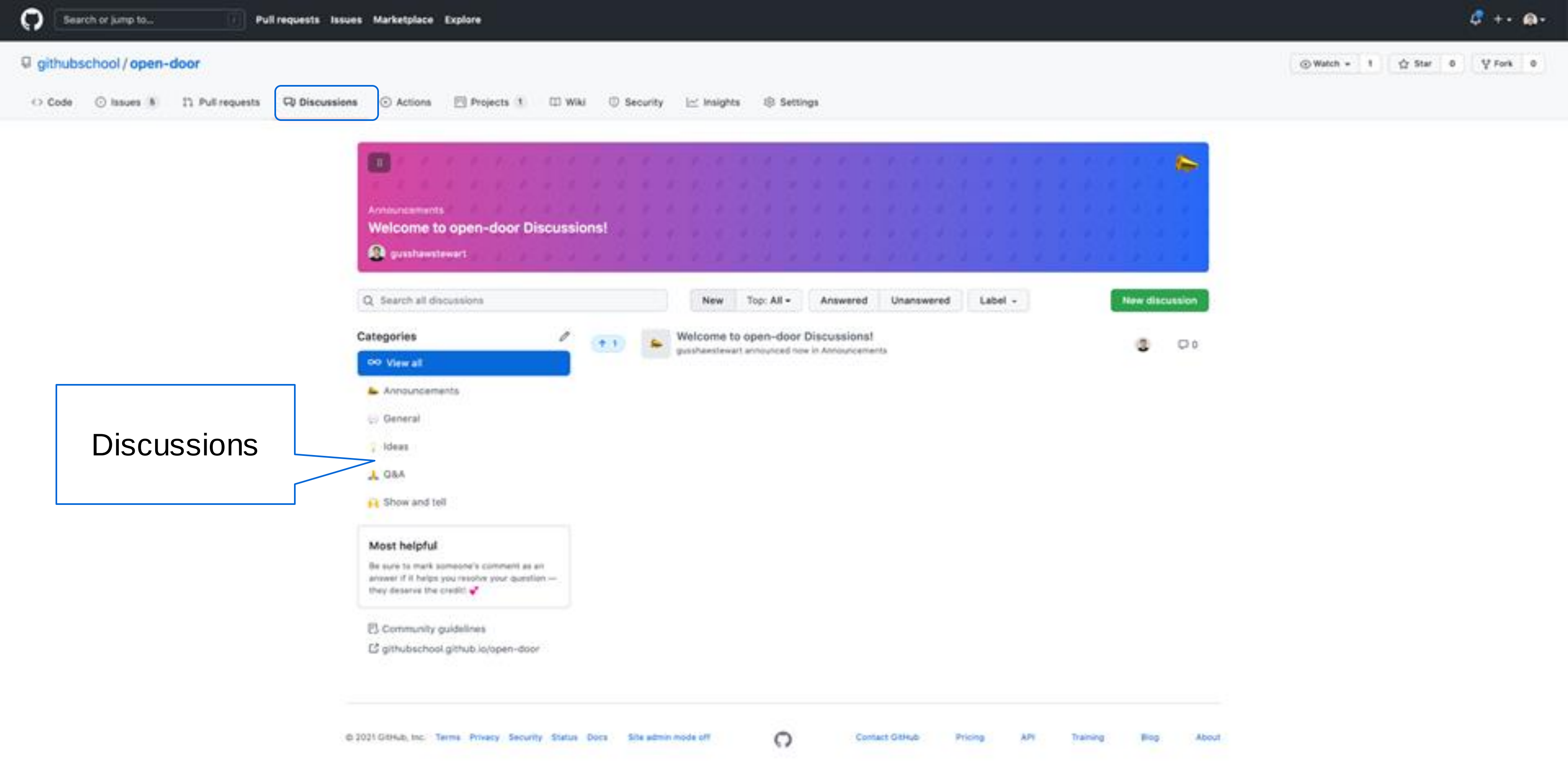
Now, GitHub will help potential first-time contributors discover issues labeled with good first issue

There aren't any open pull requests.

You could search all of GitHub or try an advanced search.

ProTip! Updated in the last three days: updated:>2021-08-29.

Pull Requests



Announcements

Welcome to open-door Discussions!

gusshawstewart

↑ 1 Welcome to open-door Discussions!

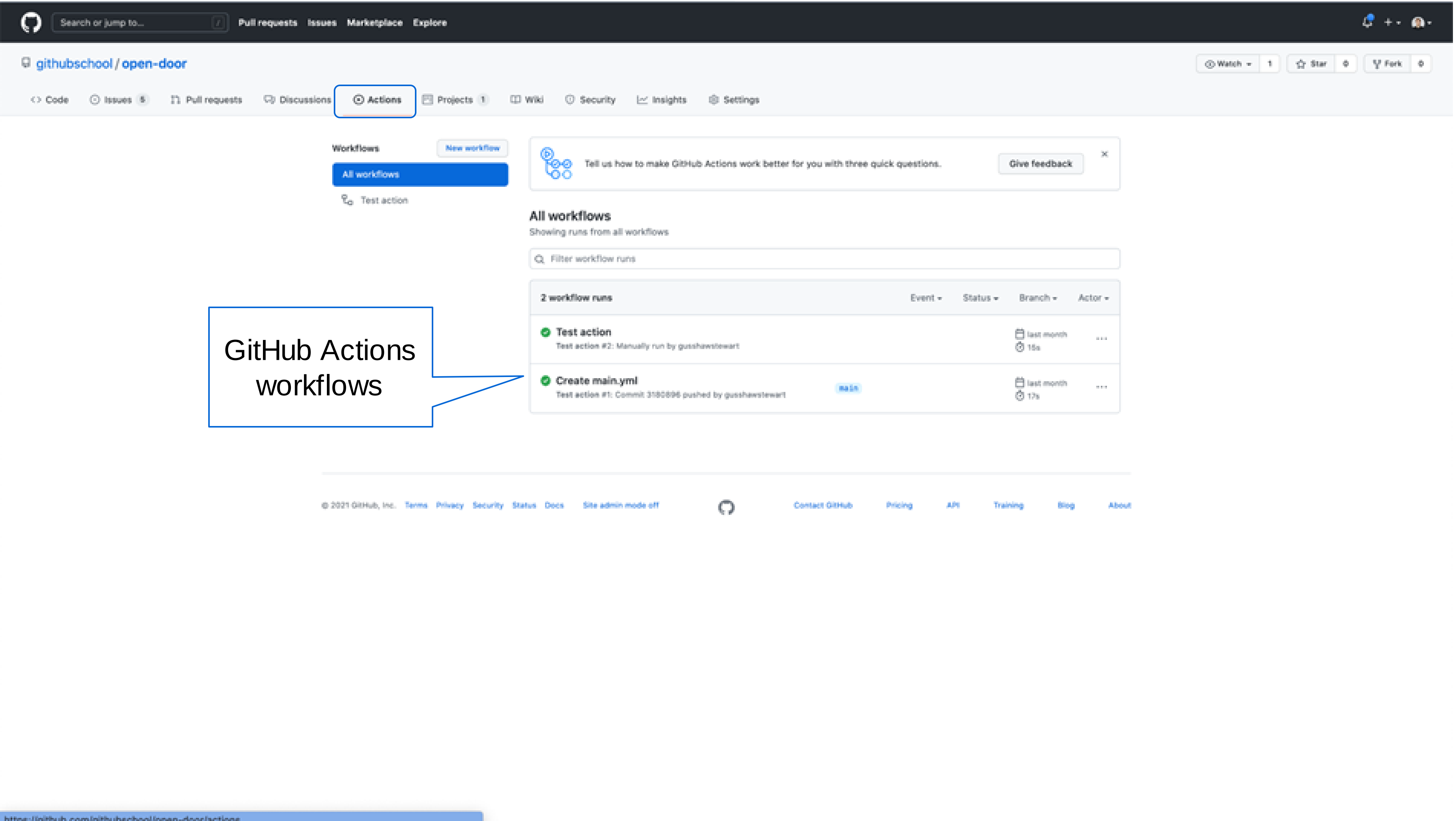
gusshawstewart announced now in Announcements

0

Most helpful

Be sure to mark someone's comment as an answer if it helps you resolve your question — they deserve the credit! 💕

Discussions



Tell us how to make GitHub Actions work better for you with three quick questions.

All workflows

Showing runs from all workflows

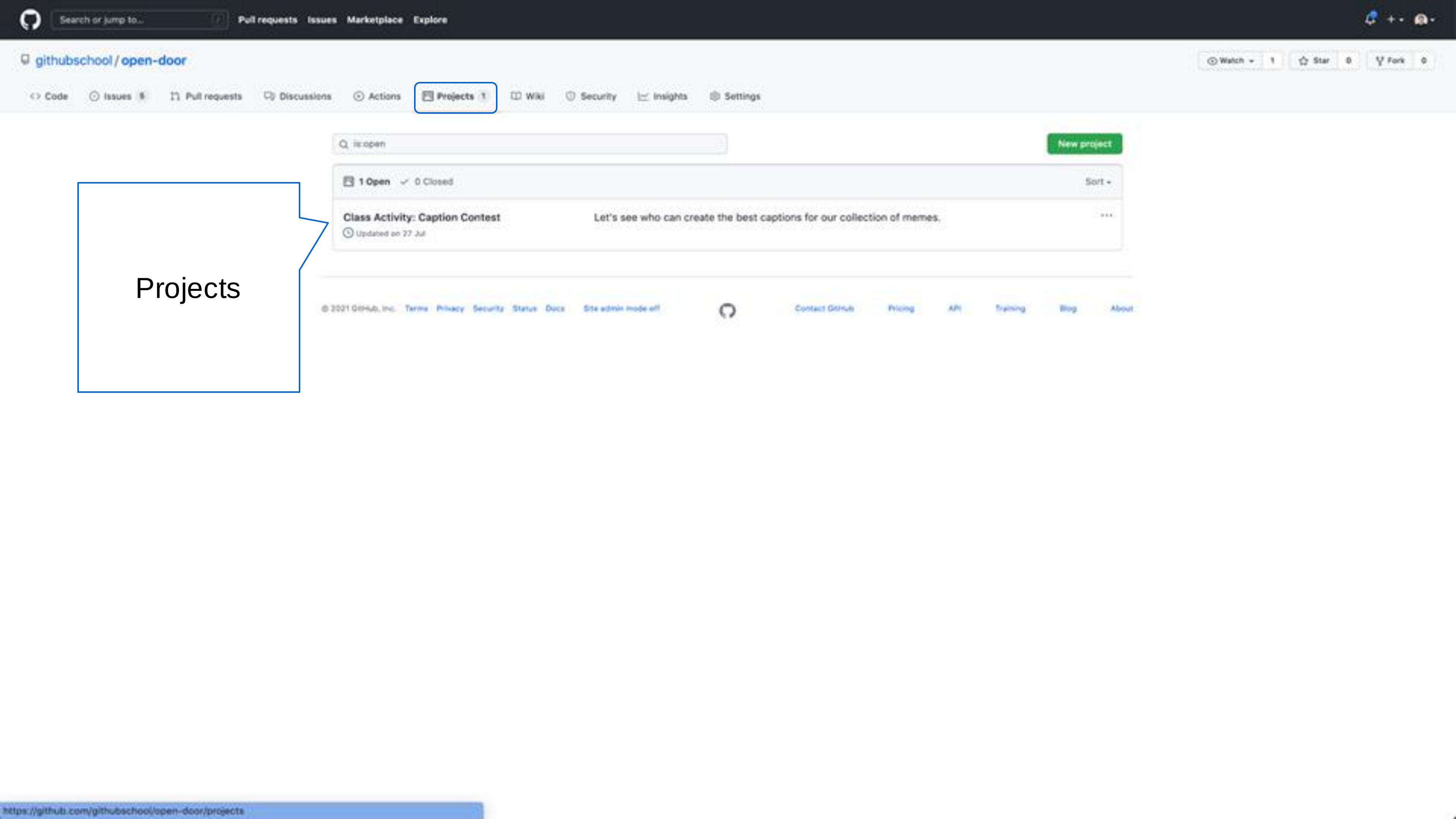
Filter workflow runs

2 workflow runs		Event ▾ Status ▾ Branch ▾ Actor ▾
✓ Test action Test action #2: Manually run by gusshawstewart		last month 15s
✓ Create main.yml Test action #1: Commit 3180896 pushed by gusshawstewart	main	last month 17s

© 2021 GitHub, Inc. Terms Privacy Security Status Docs Site admin mode off Contact GitHub Pricing API Training Blog About

https://github.com/githubschool/open-door/actions

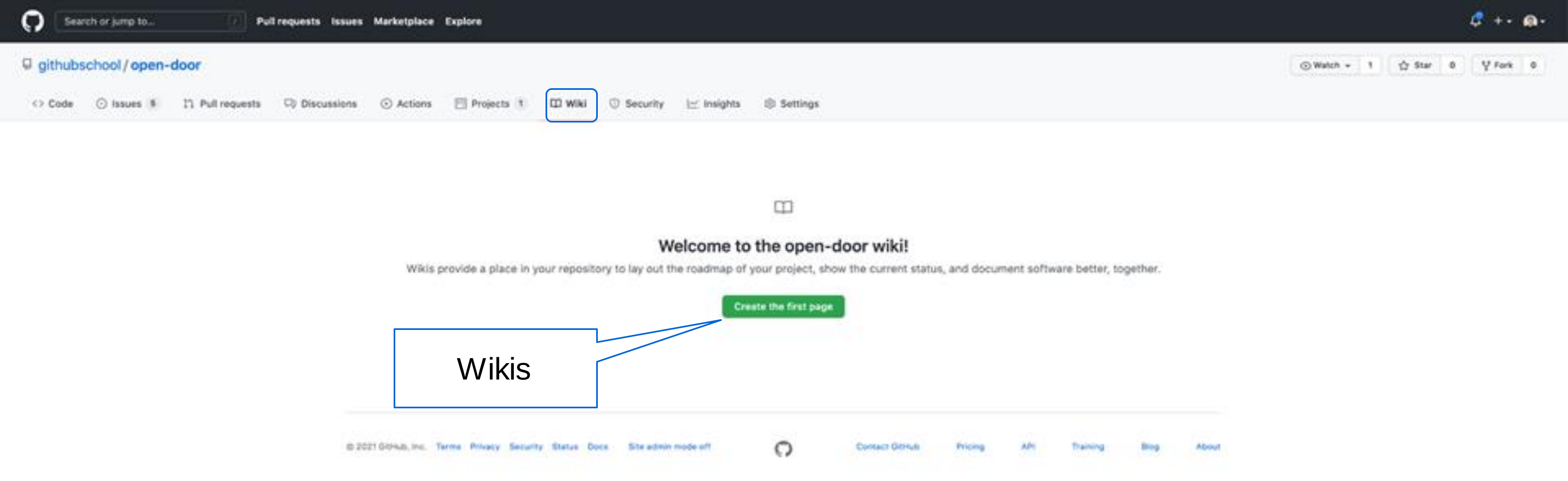
GitHub Actions workflows



Class Activity: Caption Contest
Updated on 27 Jul

Let's see who can create the best captions for our collection of memes.

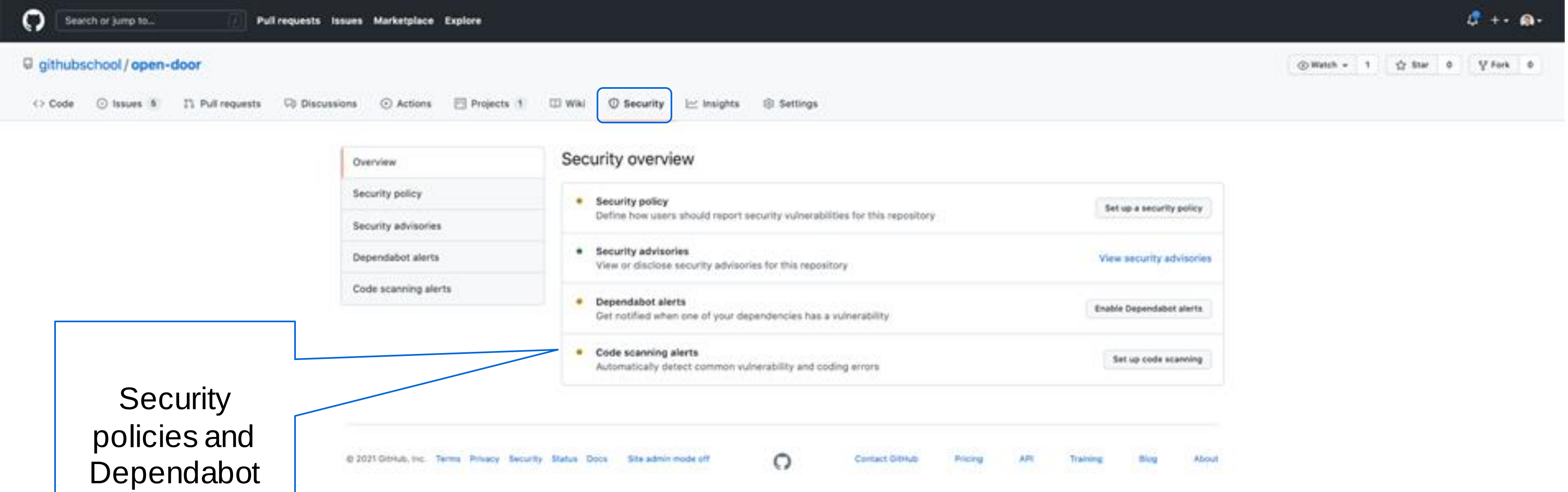
Projects



Welcome to the open-door wiki!

Wikis provide a place in your repository to lay out the roadmap of your project, show the current status, and document software better, together.

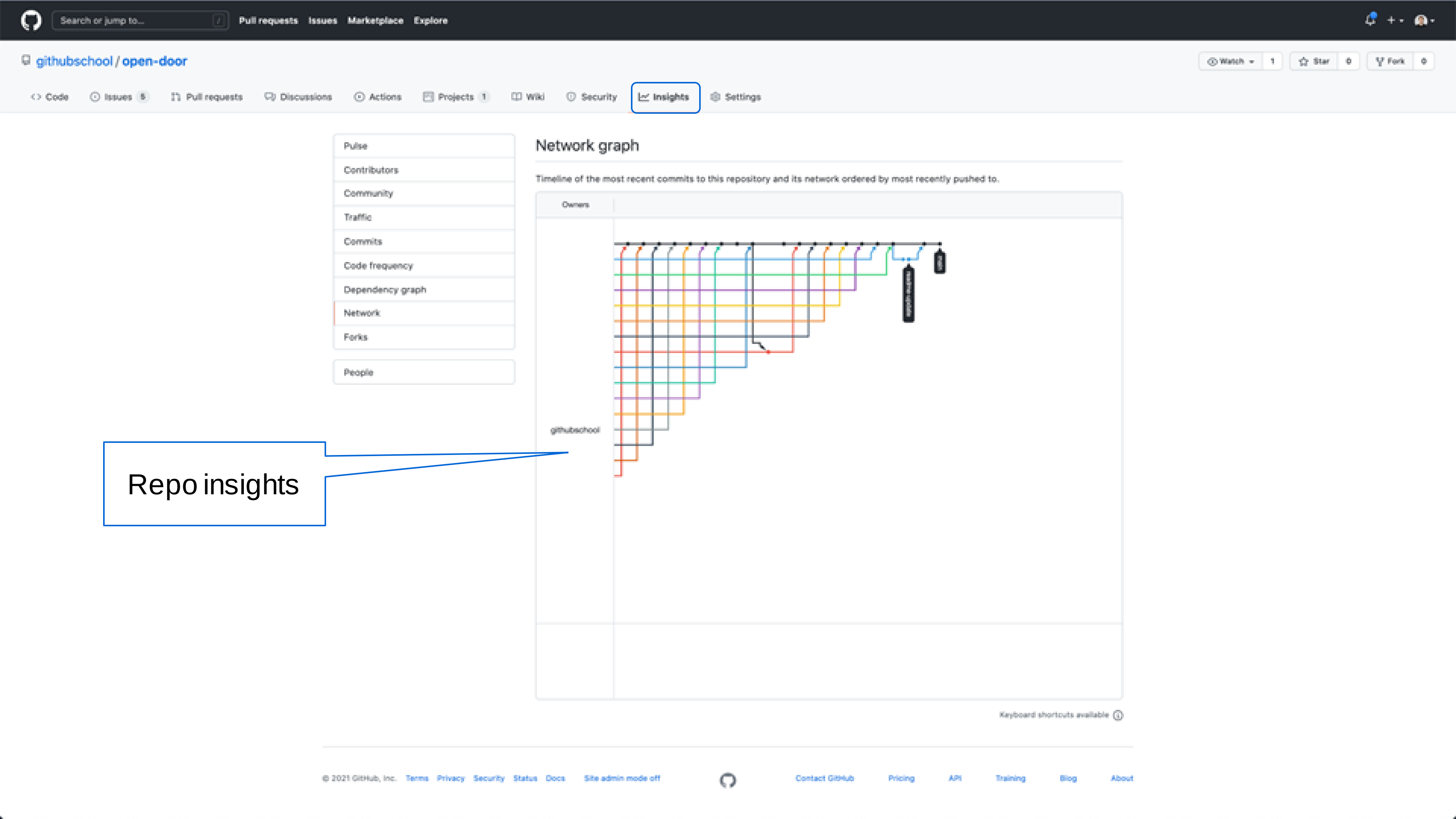
Wikis



Security overview

- **Security policy**
  Define how users should report security vulnerabilities for this repository
  Set up a security policy
- **Security advisories**
  View or disclose security advisories for this repository
  View security advisories
- **Dependabot alerts**
  Get notified when one of your dependencies has a vulnerability
  Enable Dependabot alerts
- **Code scanning alerts**
  Automatically detect common vulnerability and coding errors
  Set up code scanning

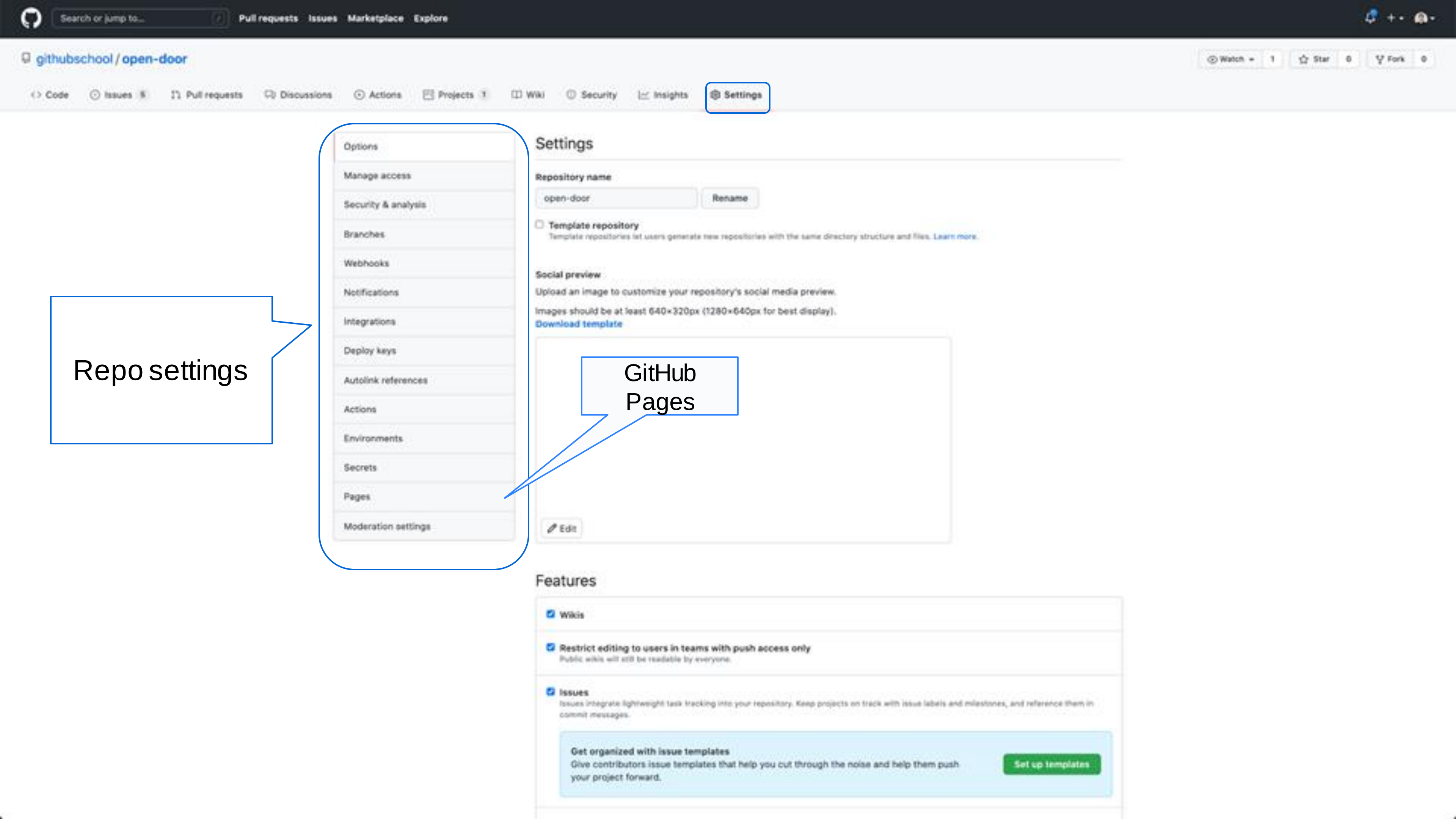
Security policies and Dependabot



Network graph

Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.

Repo insights



Repo settings

GitHub Pages

Settings

Repository name

open-door Rename

□ **Template repository**
Template repositories let users generate new repositories with the same directory structure and files. Learn more.

Social preview

Upload an image to customize your repository's social media preview.

Images should be at least 640×320px (1280×640px for best display).
Download template

Features

☑ **Wikis**

☑ **Restrict editing to users in teams with push access only**
Public wikis will still be readable by everyone.

☑ **Issues**
Issues integrate lightweight task tracking into your repository. Keep projects on track with issue labels and milestones, and reference them in commit messages.

Get organized with issue templates
Give contributors issue templates that help you cut through the noise and help them push your project forward.

Activity 1: comment on the issue

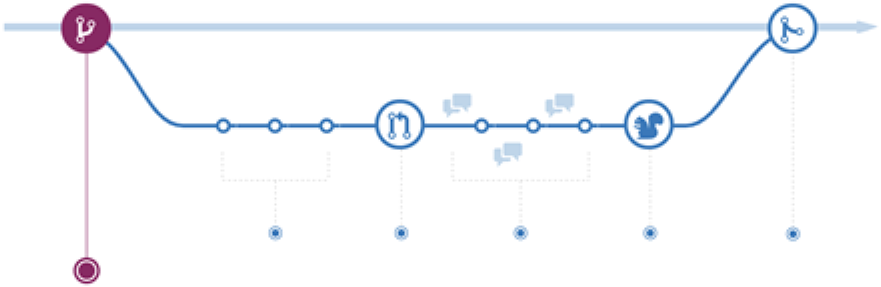
Repository: caption-this

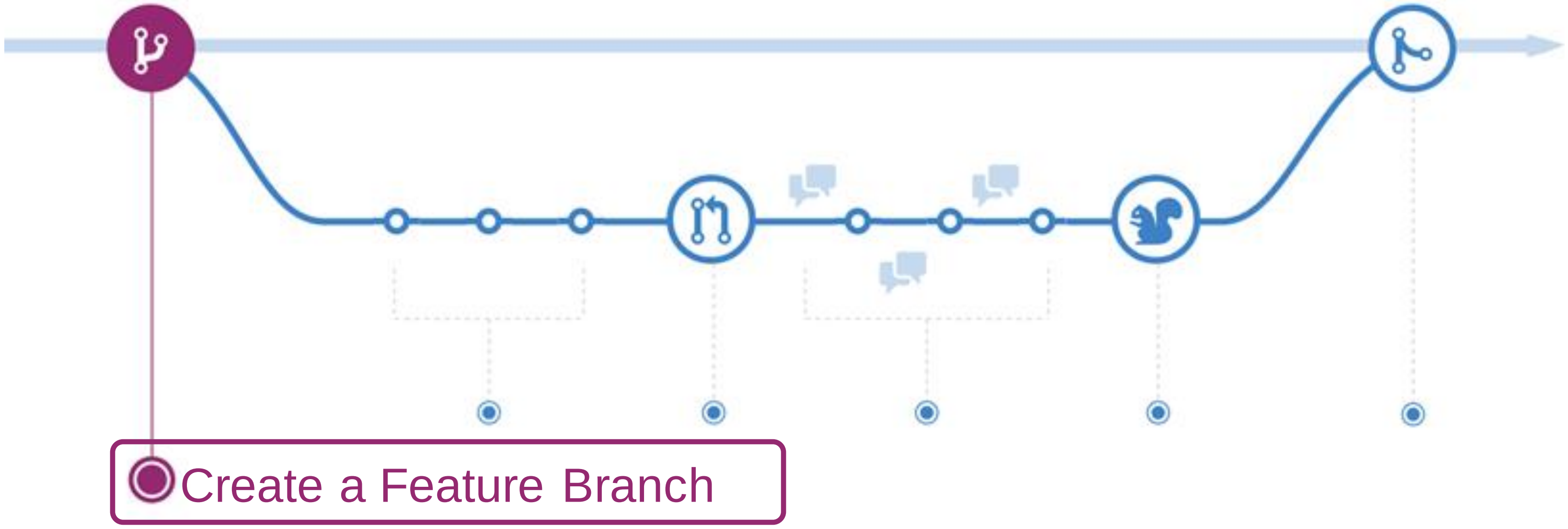
1. Navigate to the issues tab, and click on the issue entitled “Comment here to be added as a collaborator”
2. Leave a comment
3. Instructor needs IDs to run class scripts

Branching Strategy

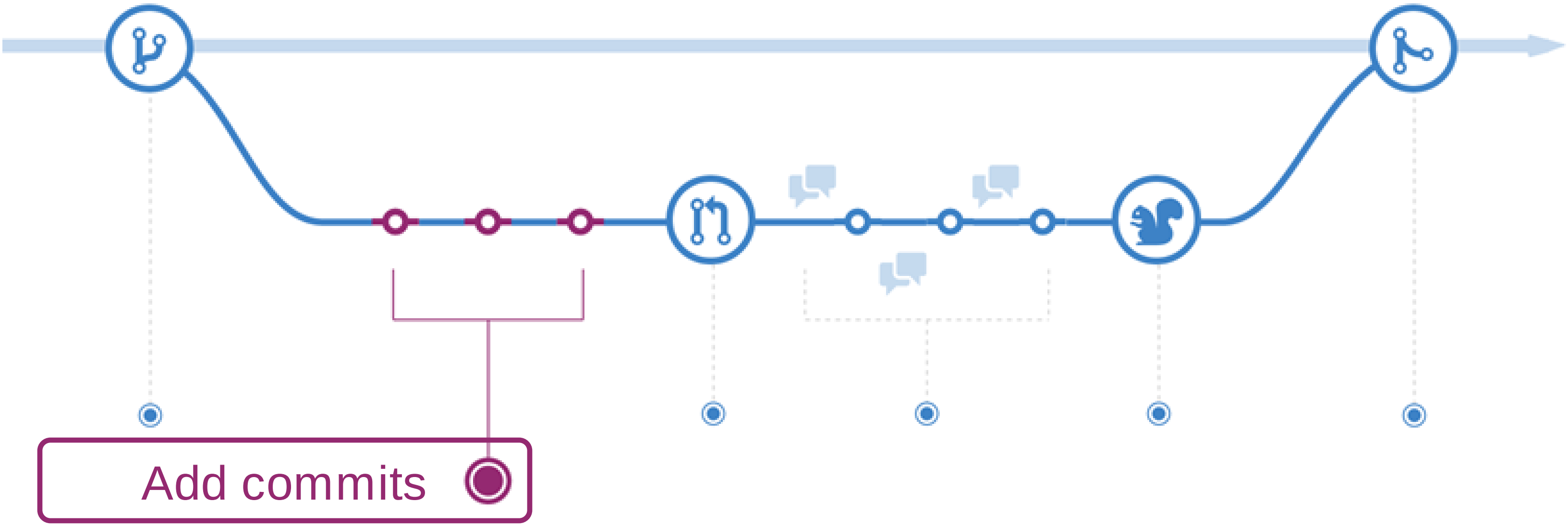
GitHub Flow

The GitHub Flow

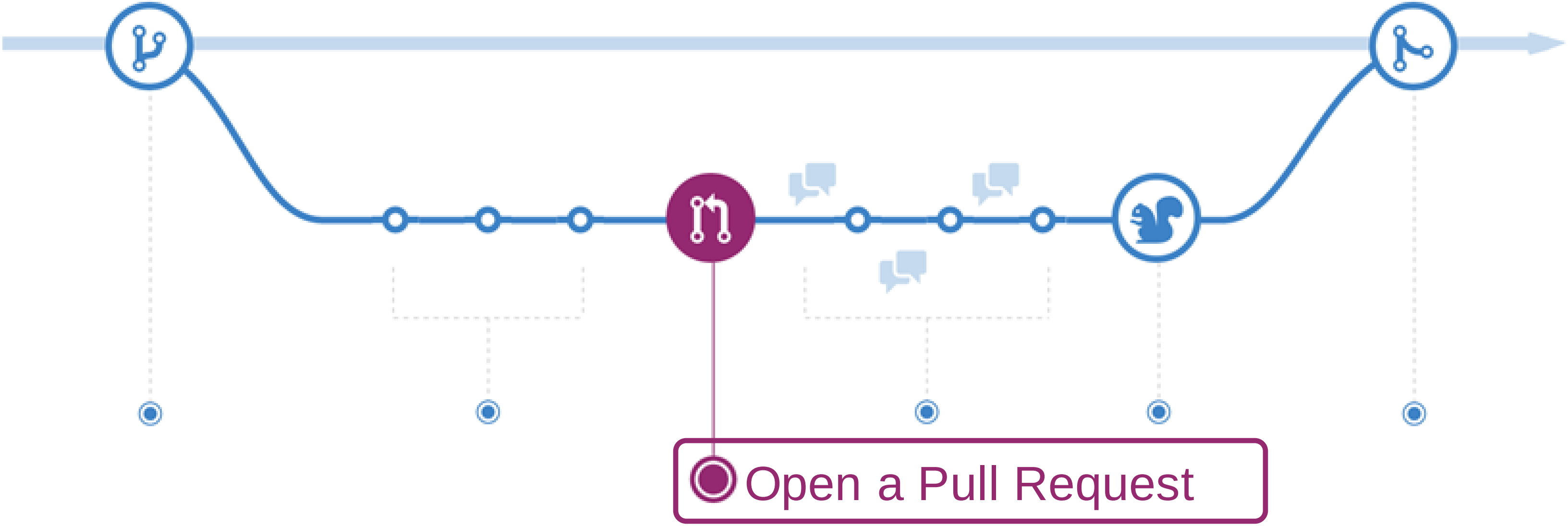




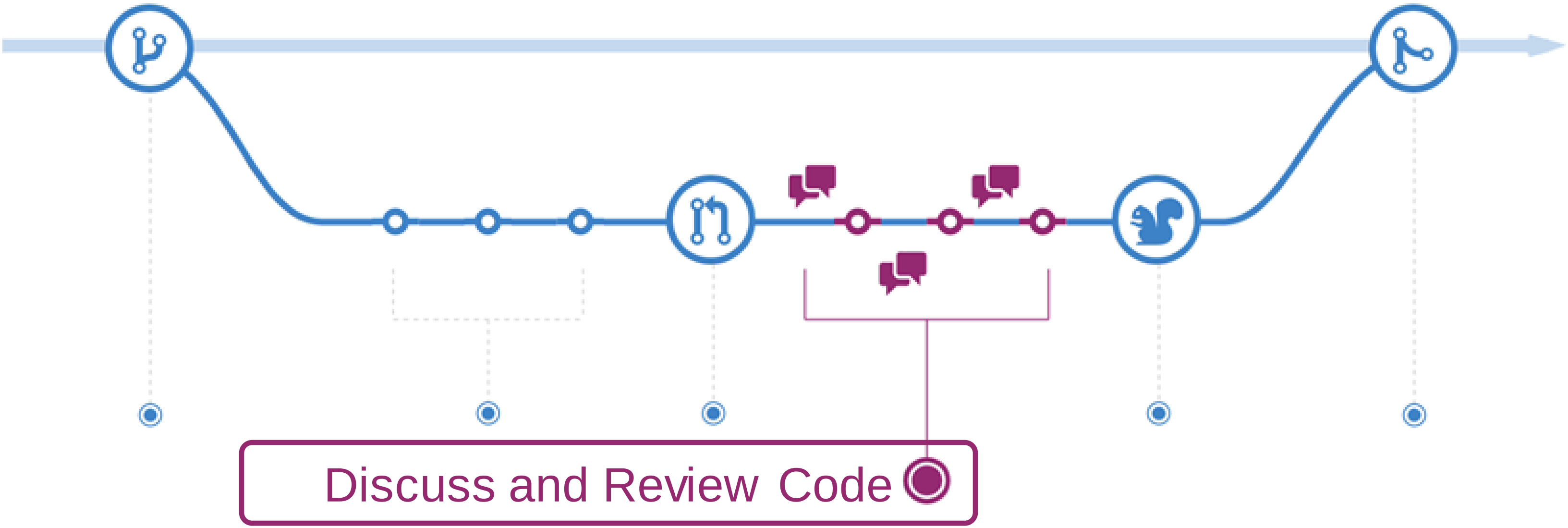
Create a Feature Branch



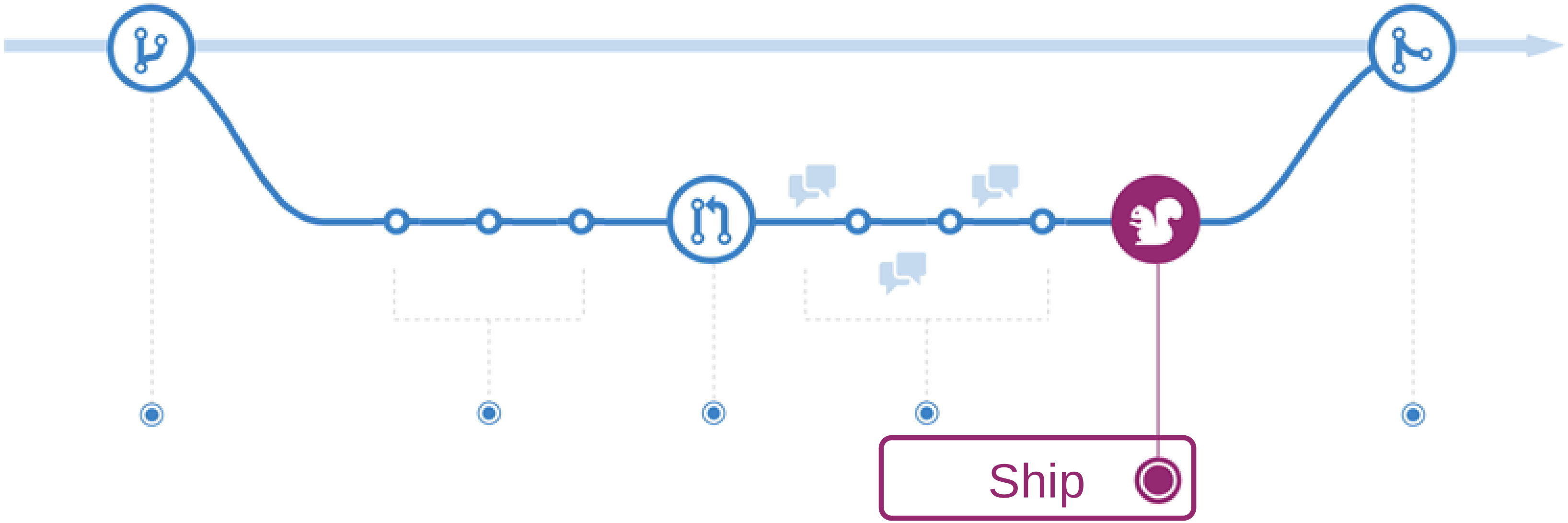
Add commits



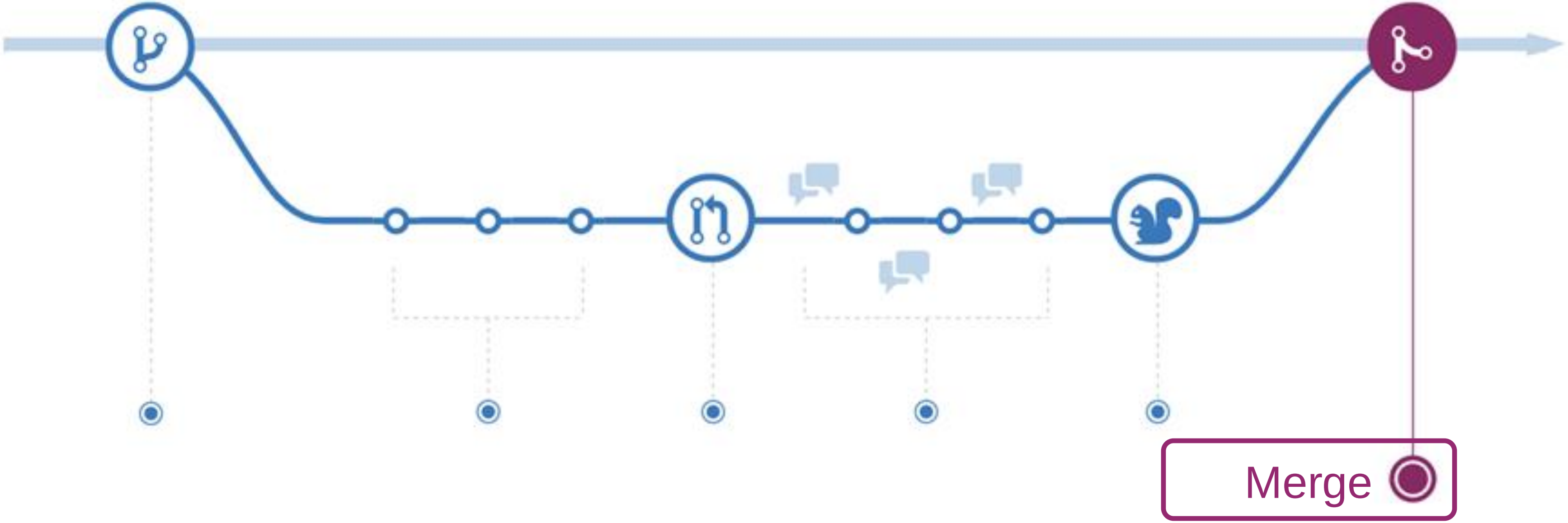
Open a Pull Request



Discuss and Review Code



Ship



Merge

Project 1: Caption this

- Branching with Git
- Collaborating on code
- Local history
- Local Git configs
- Editing on GitHub
- Streamlining workflows with aliases
- Working locally
- Merging Pull Requests

Demo

Activity 2: creating a branch with GitHub

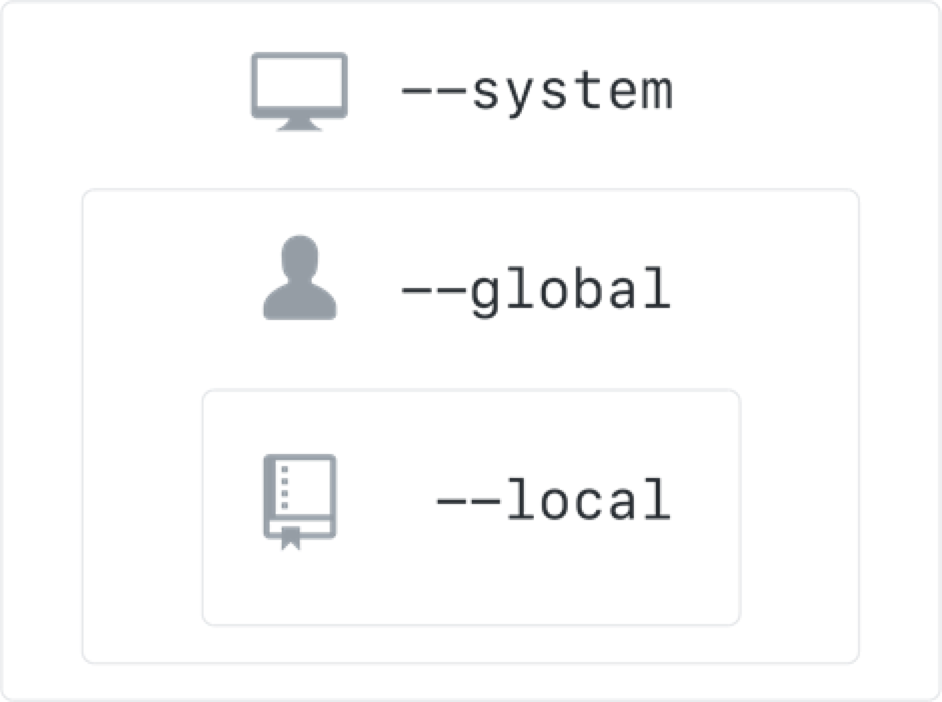
Repository: caption-this

1. Navigate to Code tab of the class repository.
2. Click the *branch dropdown*.
3. Enter the branch name ‘add-caption-YOUR-GITHUB-ID’.
4. Click the Create Branch link below

Project 1: Caption this

- Branching with Git
- Collaborating on code
- Local history
- Local Git configs
- Editing on GitHub
- Streamlining workflows with aliases
- Working locally
- Merging Pull Requests

Local Git configs



--system

--global

--local

Demo

Activity 3: configuring username and email

View Your User Name and Email

```
git config --global user.name
```

```
git config --global user.email
```

Viewing Your Configurations

```
git config –list
```

```
git config --global –list
```

Create a new value

```
git config --global user.nickname “MyNickName”
```

Remove value

```
git config –global --unset user.nickname
```

Project 1: Caption this

Branching with Git

Collaborating on code

Local history

Local Git configs

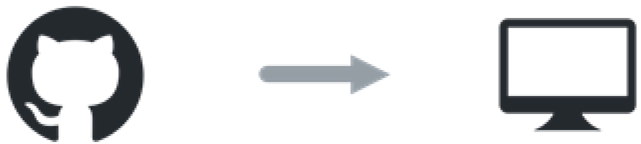
Editing on GitHub

Streamlining workflows with aliases

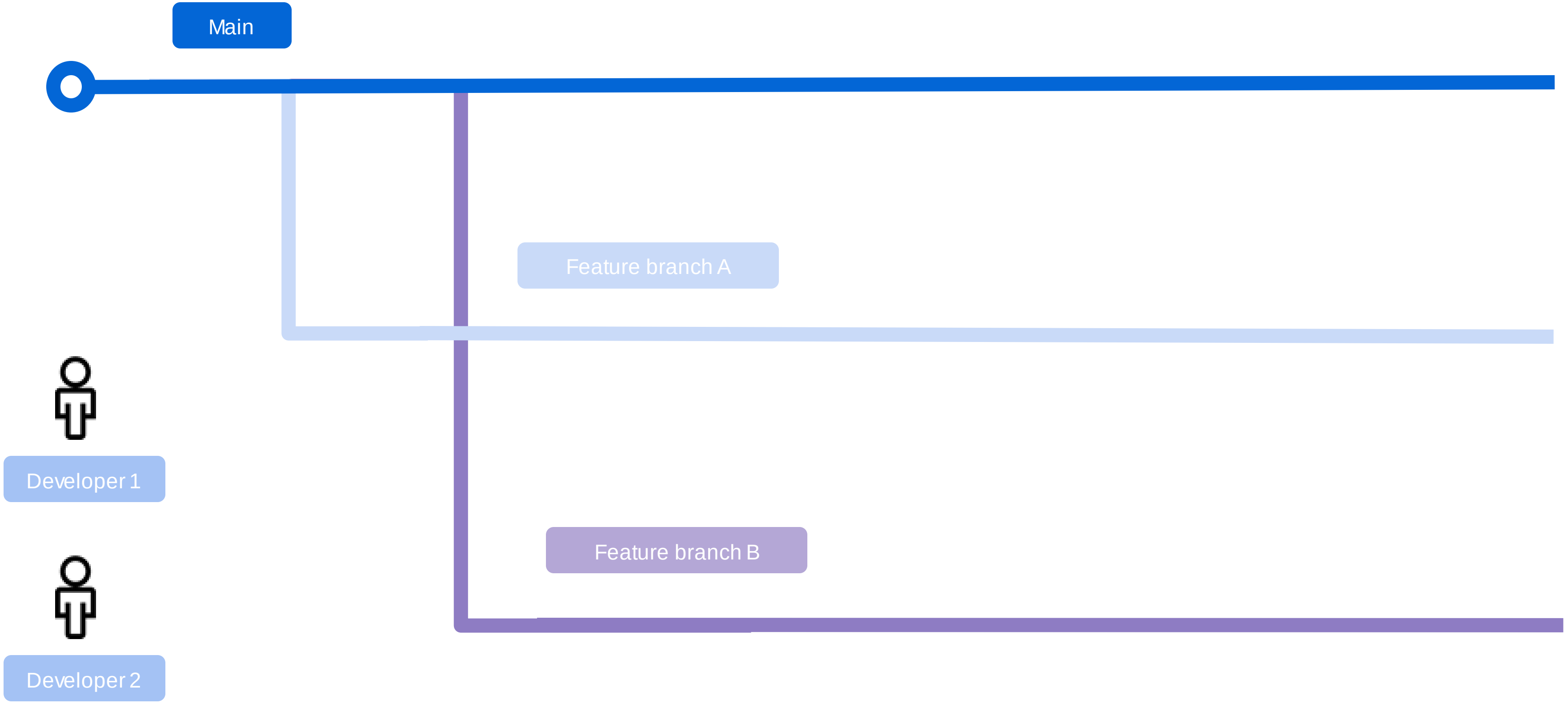
Working locally

Merging Pull Requests

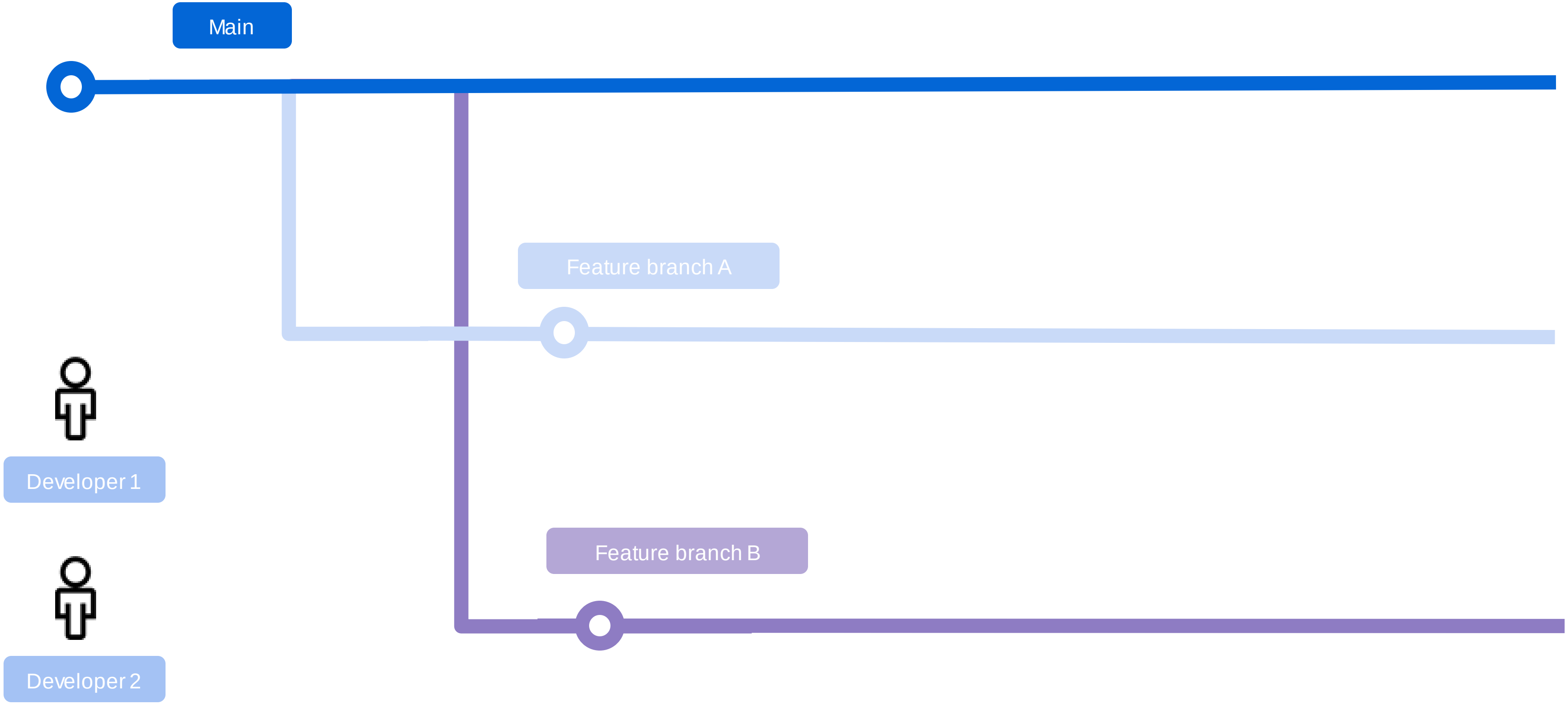
Working locally



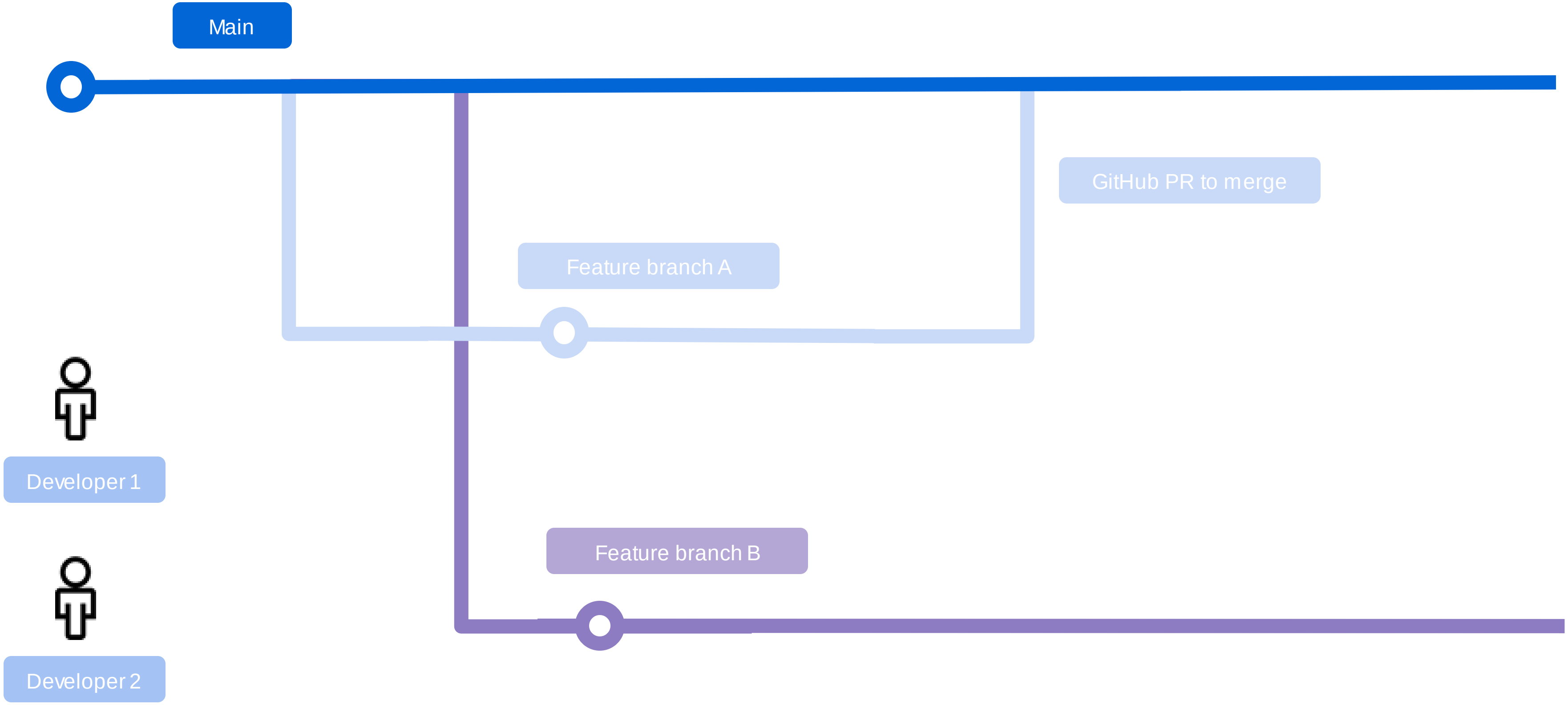
Branching with Git



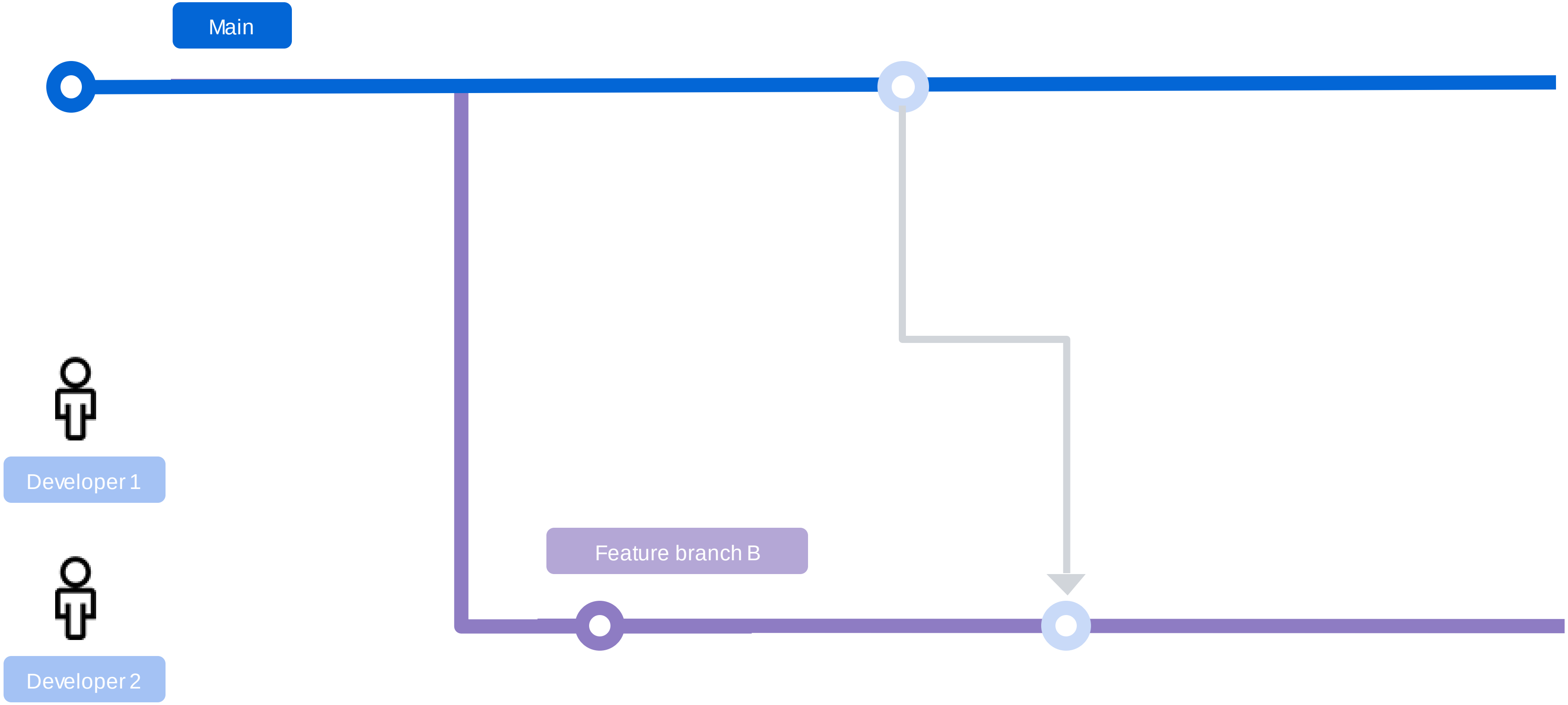
Branching with Git



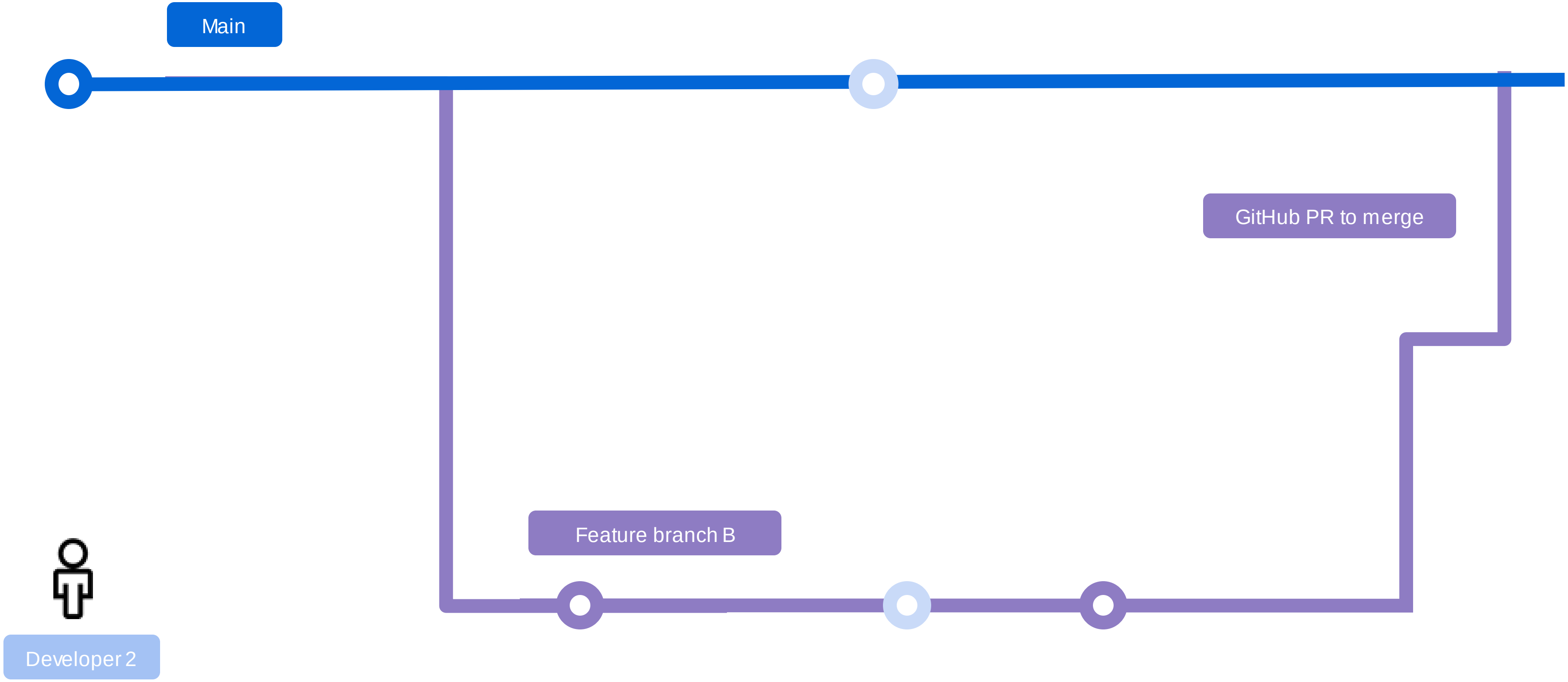
Branching with Git



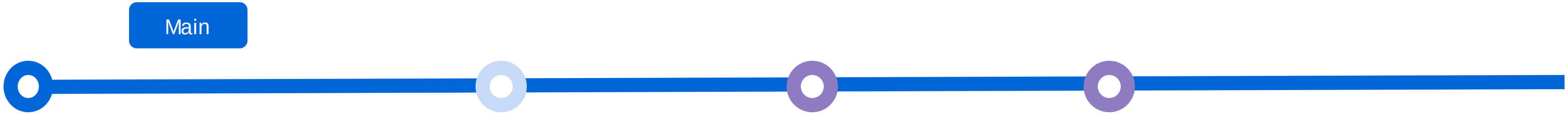
Branching with Git



Branching with Git



Branching with Git



Demo

Activity 4: create a local copy of the repo

Repository: caption-this

1. Navigate to the Code tab of the class repository on GitHub.
2. Click <>Code in green.
3. Copy the clone URL to your clipboard.
4. Open your command line application.
5. Retrieve a full copy of the repository from GitHub: `git clone <CLONE-URL>`
6. Once the clone is complete, cd into the new directory created by the clone operation: `cd <REPOSITORY-NAME>`

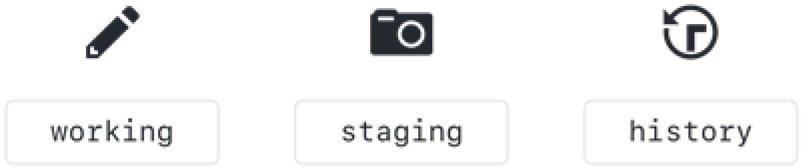
Activity 5:
Switching branches

using your local cloned repo of caption-this

1. `git branch`
2. `git branch --all`
3. `git branch -a`
4. `git switch <BRANCH-NAME>`

   `Older versions not support switch, use checkout`

The two stage commit

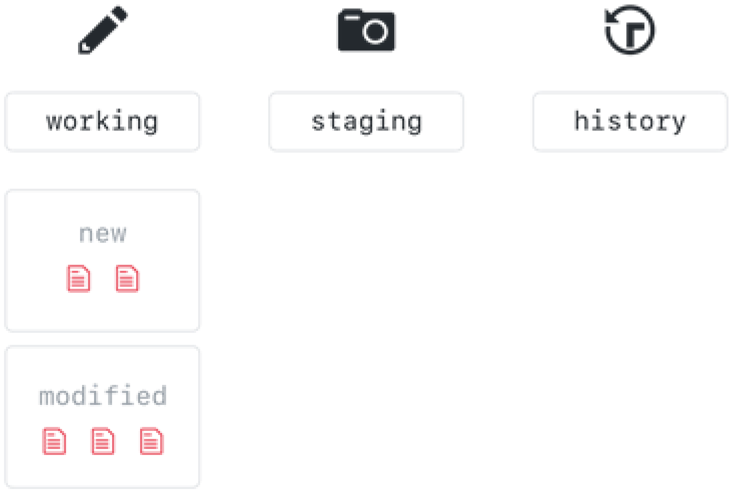


working

staging

history

The working tree



working

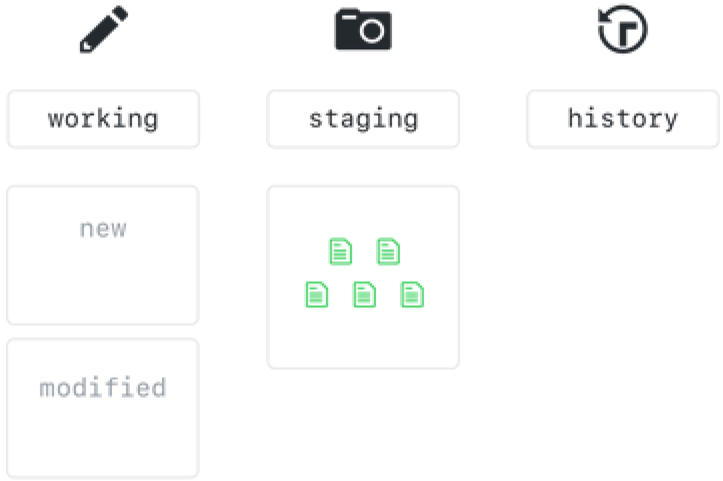
staging

history

new

modified

Staging area



working

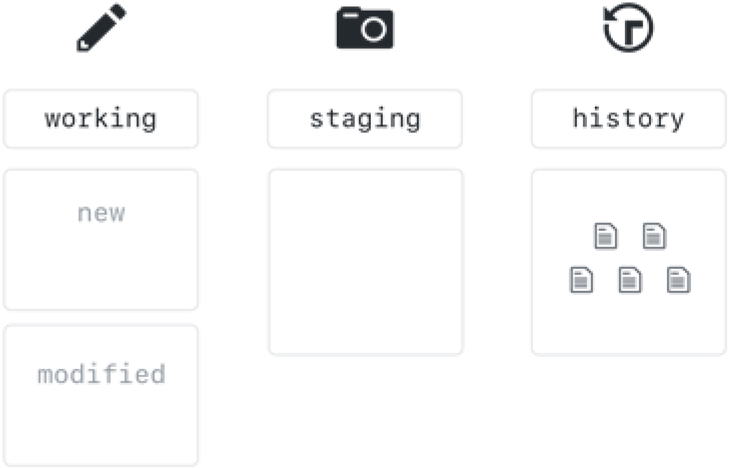
staging

history

new

modified

The commit



working

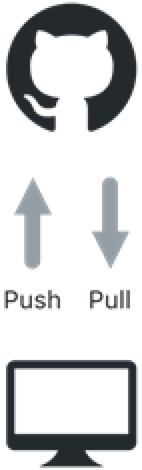
new

modified

staging

history

Pushing changes to Github



Demo

Activity 6: edit your file

In your local caption-this repo

1. Make sure you are still on the previously created branch
2. Within the _slides directory, find your file, e.g `05-USERNAME.md`.
3. Open your file in your favorite text editor.
4. On line 6, replace the text with your own caption.
5. *Save* your file.

Activity 7: make a commit

1. `git status`
2. `git add my-file.md`
3. `git status`
4. `git commit -m “updating caption”`
5. `git status`

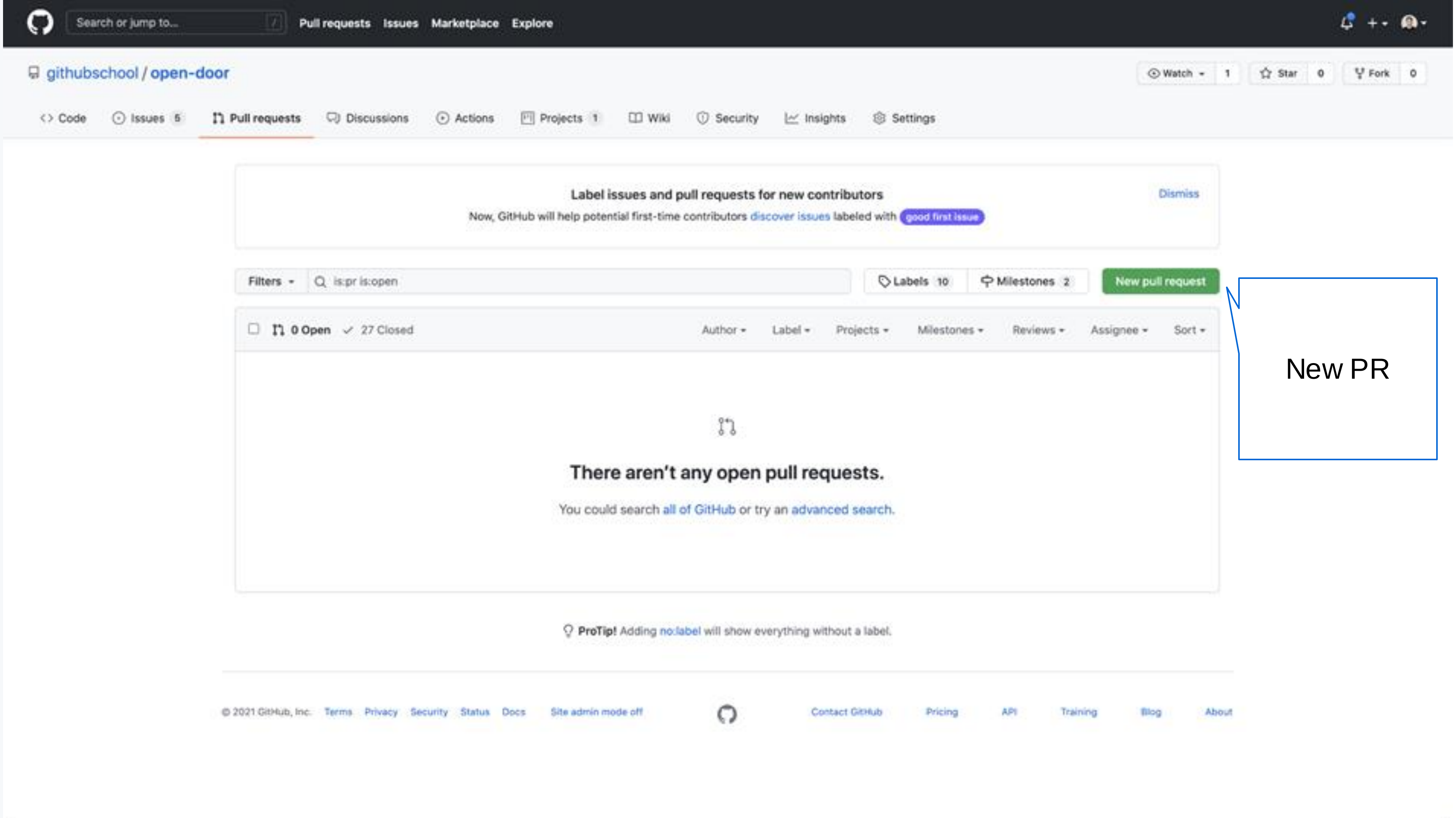
Activity 8: push to GitHub

1. `git push`

Project 1:
Caption this

- Branching with Git
- Collaborating on code
- Local history
- Local Git configs
- Editing on GitHub
- Streamlining workflows with aliases
- Working locally
- Merging Pull Requests

Creating a PR



Label issues and pull requests for new contributors

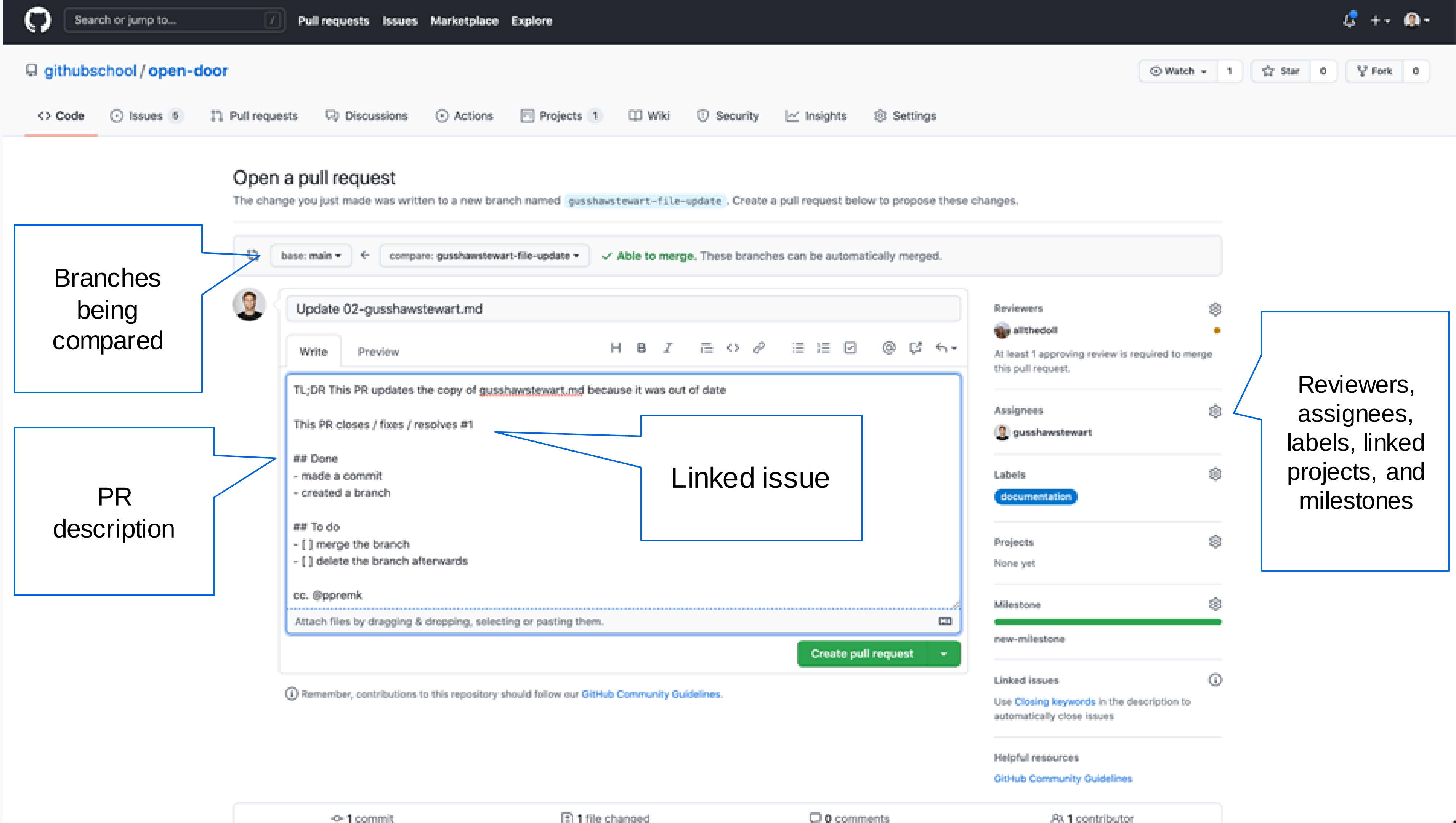
Now, GitHub will help potential first-time contributors discover issues labeled with good first issue

There aren't any open pull requests.

You could search all of GitHub or try an advanced search.

ProTip! Adding no:label will show everything without a label.

New PR



Open a pull request

The change you just made was written to a new branch named `gusshawstewart-file-update`. Create a pull request below to propose these changes.

base: main ← compare: gusshawstewart-file-update ✓ **Able to merge.** These branches can be automatically merged.

Update 02-gusshawstewart.md

```
TL;DR This PR updates the copy of gusshawstewart.md because it was out of date

This PR closes / fixes / resolves #1

## Done
- made a commit
- created a branch

## To do
- [ ] merge the branch
- [ ] delete the branch afterwards

cc. @ppremk
```

Attach files by dragging & dropping, selecting or pasting them.

Remember, contributions to this repository should follow our GitHub Community Guidelines.

Reviewers
allthedoll
At least 1 approving review is required to merge this pull request.

Assignees
gusshawstewart

Labels
documentation

Projects
None yet

Milestone
new-milestone

Linked issues
Use Closing keywords in the description to automatically close issues

Helpful resources
GitHub Community Guidelines

Branches being compared

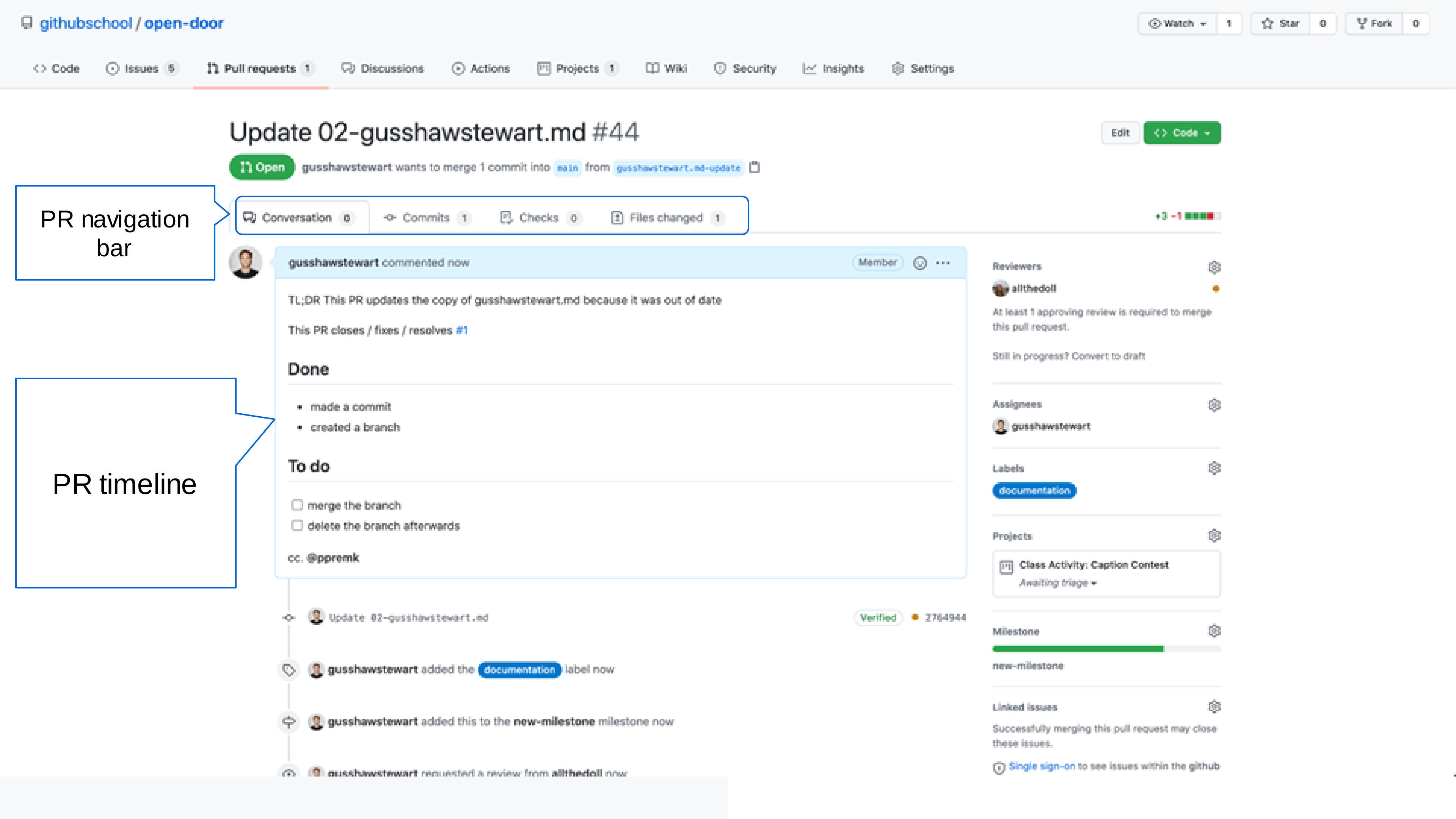
PR description

Linked issue

Reviewers, assignees, labels, linked projects, and milestones

PR navigation bar

PR timeline



Update 02-gusshawstewart.md #44

Open gusshawstewart wants to merge 1 commit into `main` from `gusshawstewart.md-update`

gusshawstewart commented now

TL;DR This PR updates the copy of gusshawstewart.md because it was out of date

This PR closes / fixes / resolves #1

Done

- made a commit
- created a branch

To do

- ☐ merge the branch
- ☐ delete the branch afterwards

cc. @ppremk

`Update 02-gusshawstewart.md` Verified 2764944

gusshawstewart added the documentation label now

gusshawstewart added this to the new-milestone milestone now

gusshawstewart requested a review from allthedoll now

Reviewers
allthedoll
At least 1 approving review is required to merge this pull request.

Still in progress? Convert to draft

Assignees
gusshawstewart

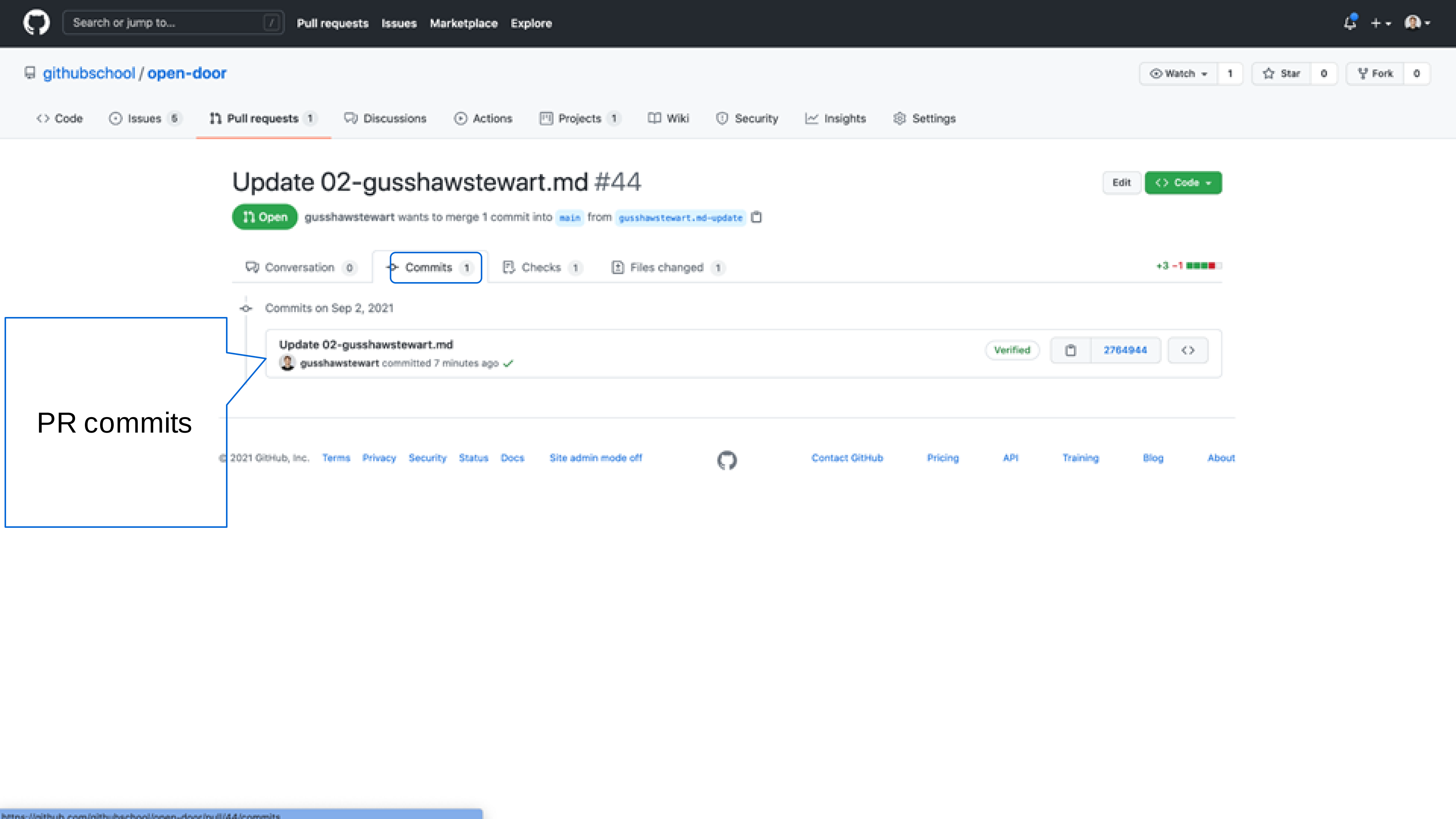
Labels
documentation

Projects
Class Activity: Caption Contest
Awaiting triage

Milestone
new-milestone

Linked issues
Successfully merging this pull request may close these issues.

Single sign-on to see issues within the github



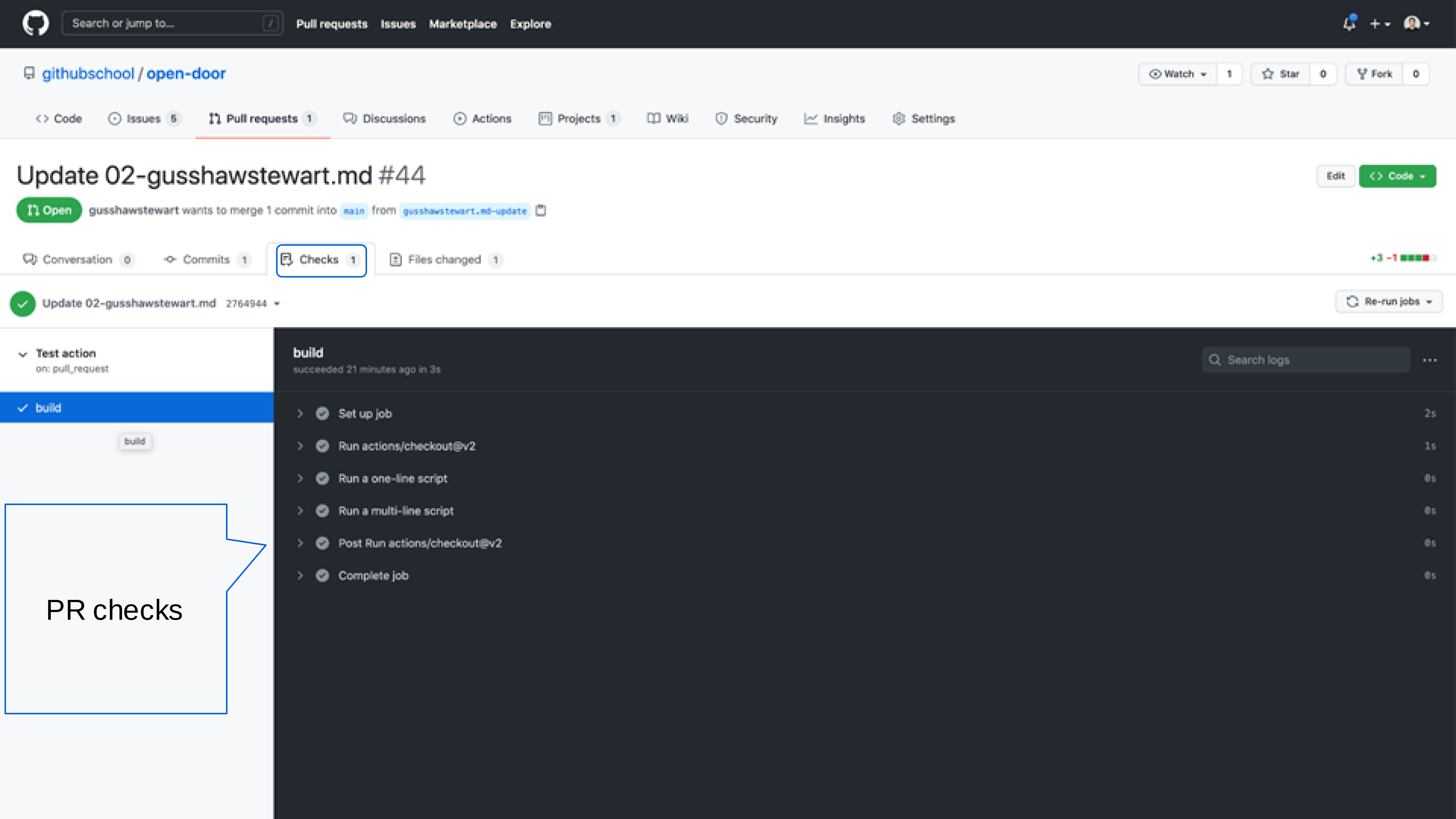
Update 02-gusshawstewart.md #44

Open gusshawstewart wants to merge 1 commit into `main` from `gusshawstewart.md-update`

Commits on Sep 2, 2021

Update 02-gusshawstewart.md
gusshawstewart committed 7 minutes ago ✓

PR commits



githubschool / open-door

Update 02-gusshawstewart.md #44

Open gusshawstewart wants to merge 1 commit into `main` from `gusshawstewart.md-update`

Update 02-gusshawstewart.md 2764944

Test action
on: pull_request

✓ build

build

build
succeeded 21 minutes ago in 3s

```
Set up job                     2s
Run actions/checkout@v2        1s
Run a one-line script          0s
Run a multi-line script        0s
Post Run actions/checkout@v2   0s
Complete job                   0s
```

PR checks

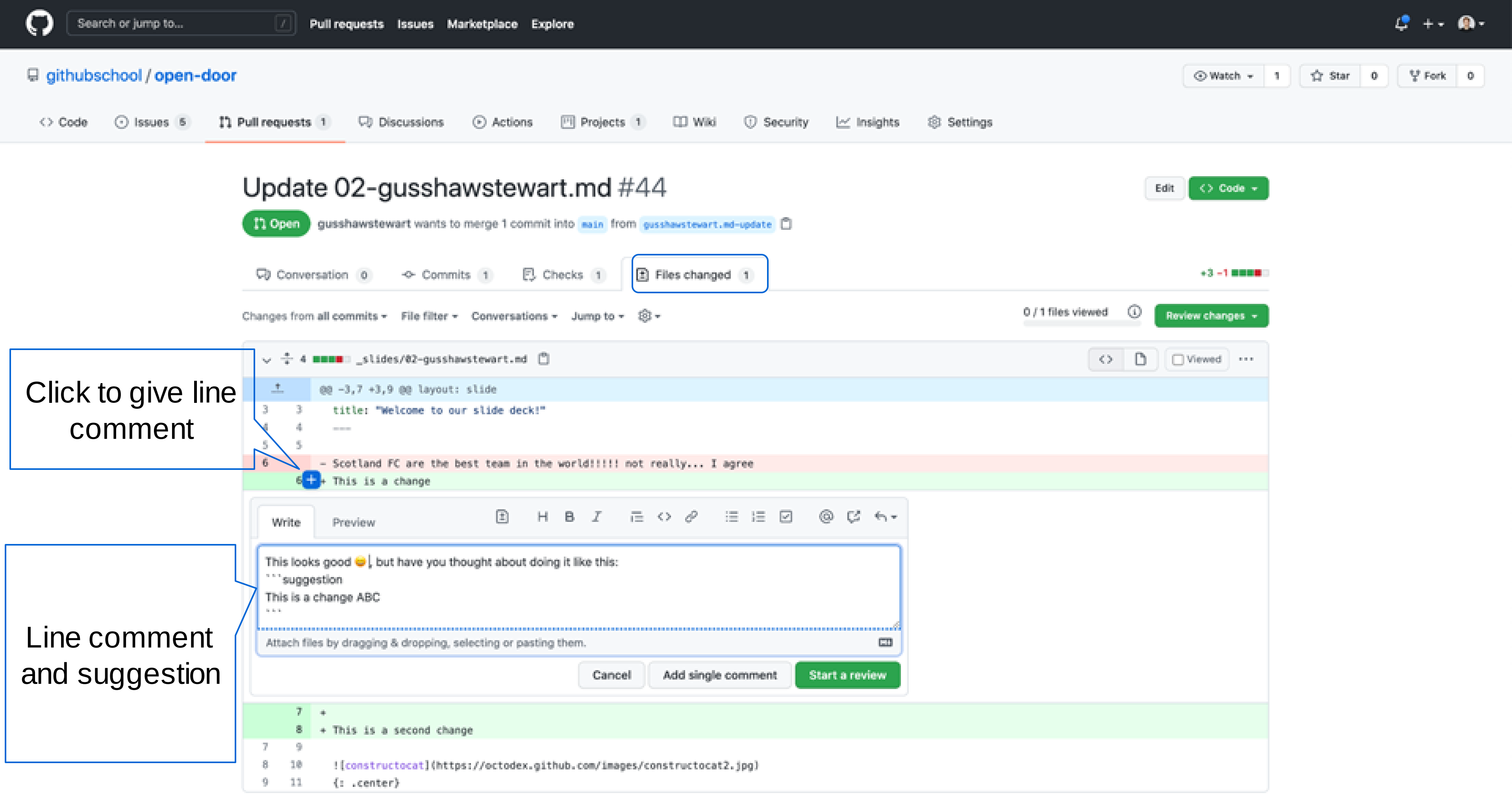
Demo

Activity 9: create a Pull Request

Repository: caption-this

1. Click the ***Pull Request*** tab.
2. Click ***New Pull*** *Request*.
3. In the ***base*** dropdown, choose **`main`**
4. In the ***compare*** dropdown, choose **your branch**.
5. Type a subject line and enter a comment.
6. Use markdown formatting to add a header and a checklist to your Pull Request.
7. Include one of the keywords: `closes`, `fixes`, or `resolves` followed by the issue number you created earlier to note which Issue the Pull Request should close. Example: `This resolves #3`
8. Click ***Preview*** to see how your Pull Request will look.
9. Assign the Pull Request to yourself.
10. Select your partner as a Reviewer for the Pull Request.
11. Click *Create pull request*.

Reviewing a PR



Update 02-gusshawstewart.md #44

gusshawstewart wants to merge 1 commit into `main` from `gusshawstewart.md-update`

`4 _slides/02-gusshawstewart.md`

```
@@ -3,7 +3,9 @@ layout: slide
3   3    title: "Welcome to our slide deck!"
4   4    ---
5   5
6      - Scotland FC are the best team in the world!!!!! not really... I agree
    6  + This is a change
```

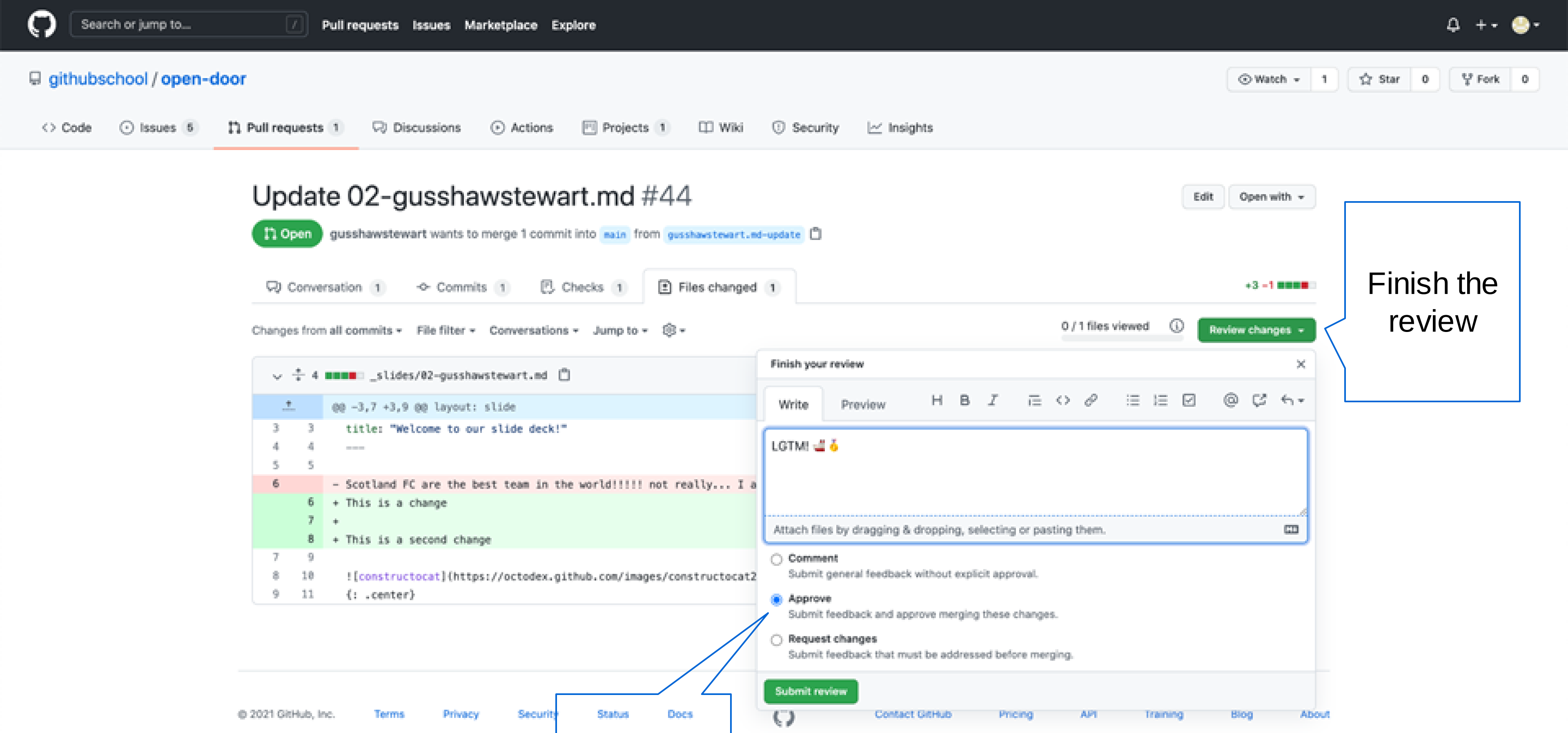
This looks good 😀, but have you thought about doing it like this:

```suggestion
This is a change ABC
```

```
    7  +
    8  + This is a second change
7   9
8   10   ![constructocat](https://octodex.github.com/images/constructocat2.jpg)
9   11   {: .center}
```

Click to give line comment

Line comment and suggestion



githubschool / open-door

Update 02-gusshawstewart.md #44

Open gusshawstewart wants to merge 1 commit into `main` from `gusshawstewart.md-update`

+3 −1

```
4 _slides/02-gusshawstewart.md
@@ -3,7 +3,9 @@ layout: slide
3  3    title: "Welcome to our slide deck!"
4  4    ---
5  5
6     - Scotland FC are the best team in the world!!!!! not really... I a
   6  + This is a change
   7  +
   8  + This is a second change
7  9
8  10   ![constructocat](https://octodex.github.com/images/constructocat2
9  11   {: .center}
```

Finish your review

LGTM! 🚢 👌

Attach files by dragging & dropping, selecting or pasting them.

○ Comment
Submit general feedback without explicit approval.

● Approve
Submit feedback and approve merging these changes.

○ Request changes
Submit feedback that must be addressed before merging.

Finish the review

Approve PR

Demo

Activity 10: review a Pull Request

Repository: caption-this

1. Click the **Pull Request** tab.
2. Review the PR directly **below** your own
3. Click the **‘Files Changed**’ tab.
4. Hover over a single line in the file to see the blue +. Click the + to add a line comment.
5. Comment on the line and click Start a review.
6. Click Review changes in the top right corner.
7. ‘Approve’ their changes
8. Enter a general comment for the review.
9. Click Submit review

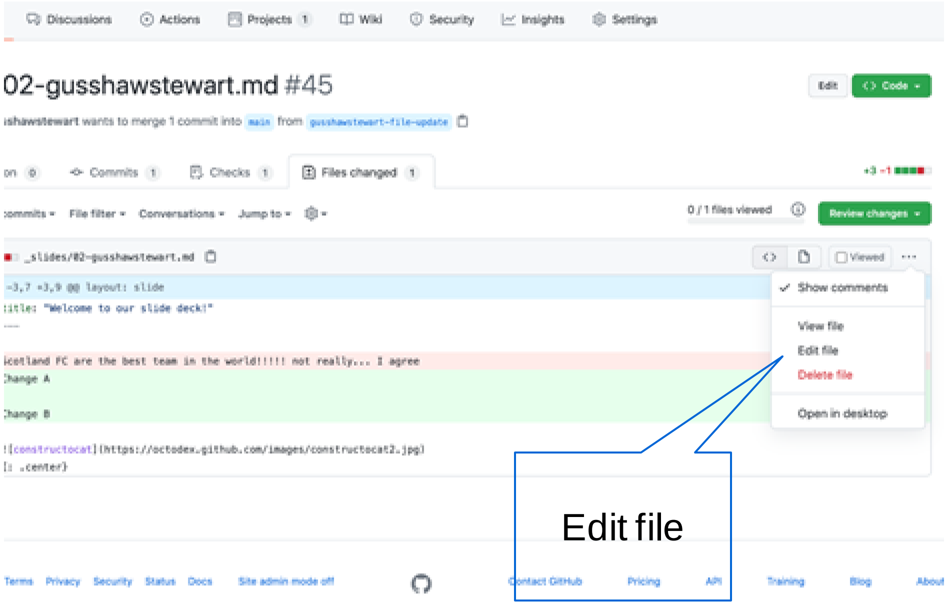
Project 1: Caption this

- Branching with Git
- Collaborating on code
- Local history

- Local Git configs
- Editing on GitHub
- Streamlining workflows with aliases

- Working locally
- Merging Pull Requests

Editing on GitHub



02-gusshawstewart.md #45

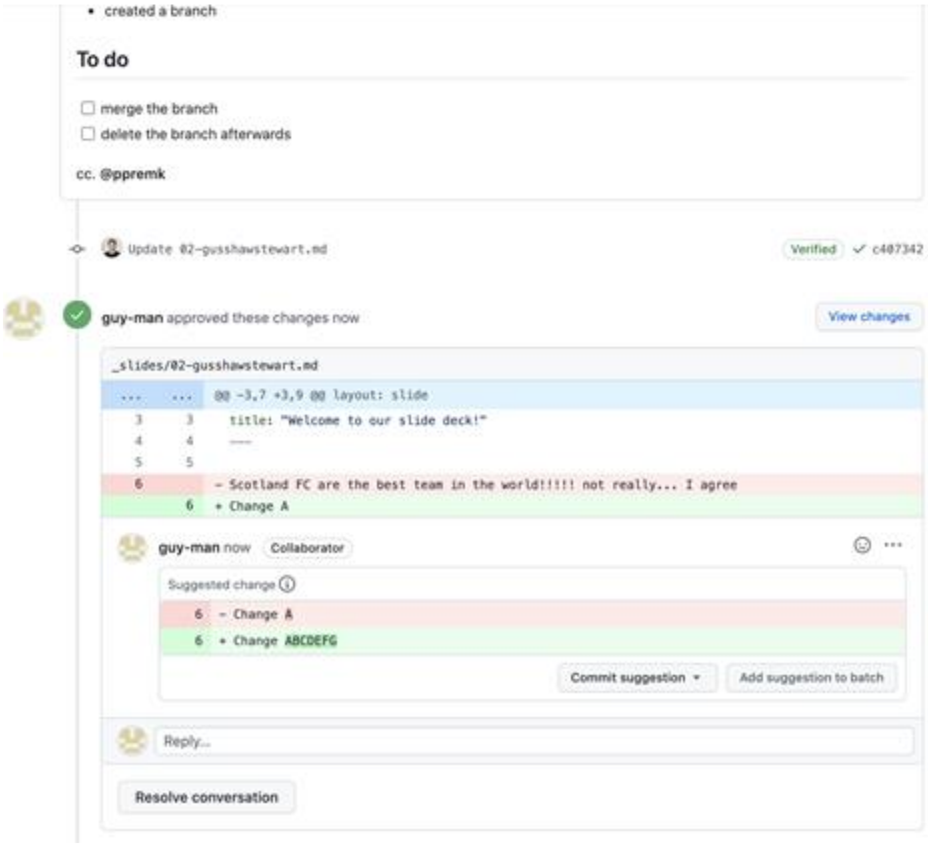
shawstewart wants to merge 1 commit into main from gusshawstewart-file-update

_slides/02-gusshawstewart.md

```
-3,7 +3,9 @@ layout: slide
itle: "Welcome to our slide deck!"
---

cotland FC are the best team in the world!!!!! not really... I agree
hange A

hange B

![constructocat](https://octodex.github.com/images/constructocat2.jpg)
{: .center}
```

Edit file

Commit suggested changes



- created a branch

To do

□ merge the branch
□ delete the branch afterwards

cc. @ppremk

Update 02-gusshawstewart.md Verified ✓ c407342

guy-man approved these changes now

_slides/02-gusshawstewart.md

```
...  ...  @@ -3,7 +3,9 @@ layout: slide
3    3      title: "Welcome to our slide deck!"
4    4      ---
5    5
6         - Scotland FC are the best team in the world!!!!! not really... I agree
     6    + Change A
```

guy-man now Collaborator

Suggested change

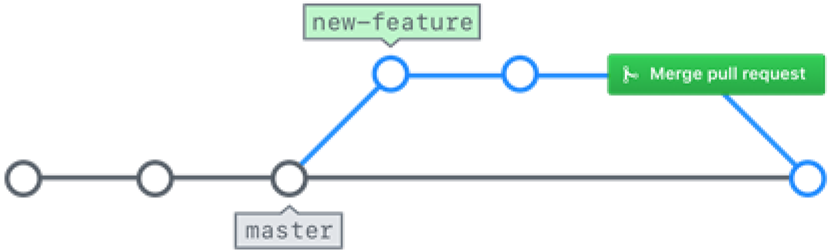
```
6  - Change A
6  + Change ABCDEFG
```

Demo

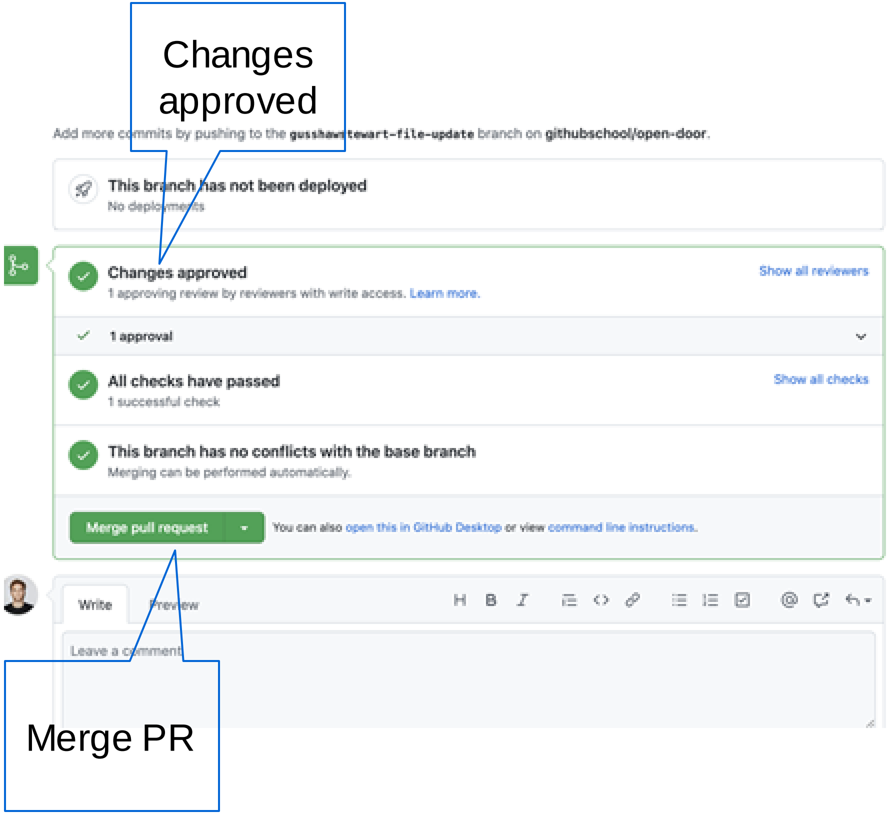
Project 1: Caption this

- Branching with Git
- Collaborating on code
- Local history
- Local Git configs
- Editing on GitHub
- Streamlining workflows with aliases
- Working locally
- Merging Pull Requests

Merging a PR



Merging on GitHub



Changes approved

Merge PR

Add more commits by pushing to the gusshawstewart-file-update branch on githubschool/open-door.

This branch has not been deployed
No deployments

Changes approved
1 approving review by reviewers with write access. Learn more.

1 approval

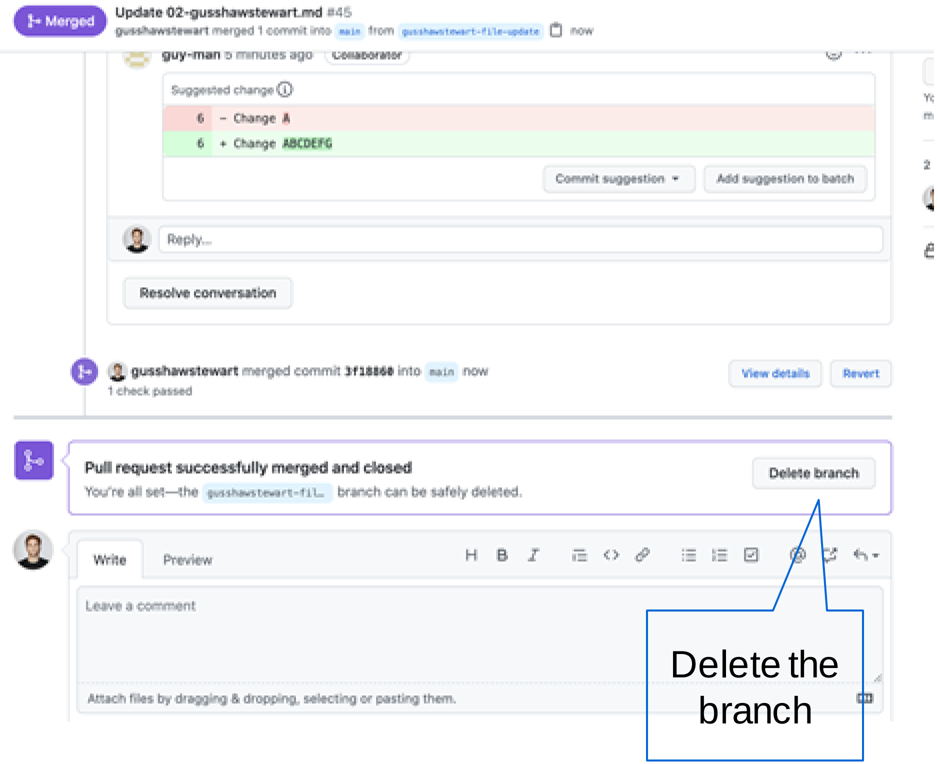
All checks have passed
1 successful check

This branch has no conflicts with the base branch
Merging can be performed automatically.

You can also open this in GitHub Desktop or view command line instructions.

Leave a comment

Delete the branch on GitHub, and locally



Merged Update 02-gusshawstewart.md #45

gusshawstewart merged 1 commit into main from gusshawstewart-file-update now

guy-man 5 minutes ago Collaborator

Suggested change

```
6  - Change A
6  + Change ABCDEFG
```

gusshawstewart merged commit 3f18868 into main now

1 check passed

Pull request successfully merged and closed

You're all set—the gusshawstewart-fil… branch can be safely deleted.

Attach files by dragging & dropping, selecting or pasting them.

Delete the branch

Demo

Activity 11: Merge Pull Request

Repository: caption-this

1. Navigate to your Pull Request (HINT: Use the Author or Assignee drop downs to find your Pull Request quickly)
2. Click *Conversation*
3. Scroll to the bottom of the Pull Request and click the *Merge pull request* button
4. Click *Confirm merge*
5. Click *Delete branch*

Update local repo

1. Start by switching back to your default branch: `git switch main`
2. Retrieve all of the changes from GitHub: `git pull`

Activity 12: Cleaning Up the Unneeded Branches

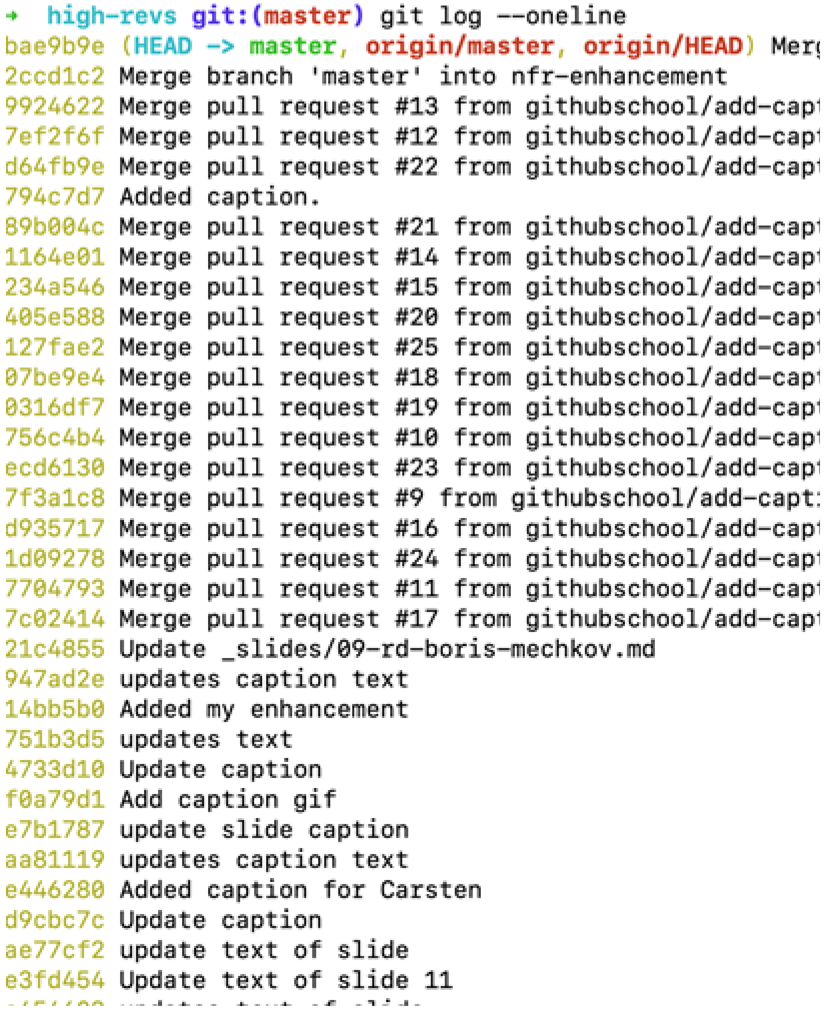
Your local copy of caption-this

1. `git branch --all`
2. `git branch --merged`
3. `git branch -d <branch-name>`
4. `git branch --all`
5. `git pull --prune`
6. `git config --global fetch.prune true`.

Project 1: Caption this

- Branching with Git
- Collaborating on code
- Local history
- Local Git configs
- Editing on GitHub
- Streamlining workflows with aliases
- Working locally
- Merging Pull Requests

Local history



```
➜  high-revs git:(master) git log --oneline
bae9b9e (HEAD -> master, origin/master, origin/HEAD) Merg
2ccd1c2 Merge branch 'master' into nfr-enhancement
9924622 Merge pull request #13 from githubschool/add-cap
7ef2f6f Merge pull request #12 from githubschool/add-cap
d64fb9e Merge pull request #22 from githubschool/add-cap
794c7d7 Added caption.
89b004c Merge pull request #21 from githubschool/add-cap
1164e01 Merge pull request #14 from githubschool/add-cap
234a546 Merge pull request #15 from githubschool/add-cap
405e588 Merge pull request #20 from githubschool/add-cap
127fae2 Merge pull request #25 from githubschool/add-cap
07be9e4 Merge pull request #18 from githubschool/add-cap
0316df7 Merge pull request #19 from githubschool/add-cap
756c4b4 Merge pull request #10 from githubschool/add-cap
ecd6130 Merge pull request #23 from githubschool/add-cap
7f3a1c8 Merge pull request #9 from githubschool/add-capt
d935717 Merge pull request #16 from githubschool/add-cap
1d09278 Merge pull request #24 from githubschool/add-cap
7704793 Merge pull request #11 from githubschool/add-cap
7c02414 Merge pull request #17 from githubschool/add-cap
21c4855 Update _slides/09-rd-boris-mechkov.md
947ad2e updates caption text
14bb5b0 Added my enhancement
751b3d5 updates text
4733d10 Update caption
f0a79d1 Add caption gif
e7b1787 update slide caption
aa81119 updates caption text
e446280 Added caption for Carsten
d9cbc7c Update caption
ae77cf2 update text of slide
e3fd454 Update text of slide 11
```

Demo

Activity 13: using Git Log

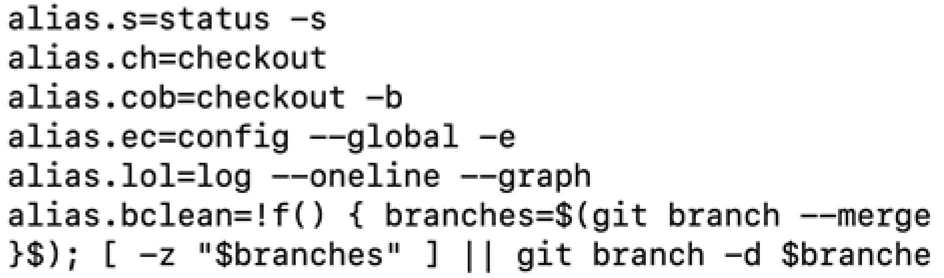
Your local copy of caption-this

- `git log`
- `git log --oneline`
- `git log --oneline --graph`
- `git log --oneline --graph --decorate`
- `git log --oneline --graph --decorate --all`

Project 1: Caption this

- Branching with Git
- Local Git configs
- Working locally
- Collaborating on code
- Editing on GitHub
- Merging Pull Requests
- Local history
- Streamlining workflows with aliases

Streamlining workflows with aliases



```
alias.s=status -s
alias.ch=checkout
alias.cob=checkout -b
alias.ec=config --global -e
alias.lol=log --oneline --graph
alias.bclean=!f() { branches=$(git branch --merge
}$); [ -z "$branches" ] || git branch -d $branche
```

Demo

Activity 14: creating custom aliases

Your local copy of caption-this

Original Command

```
git log --oneline --graph --
decorate --all
```

Creating the Alias

```
git config --global alias.lol
"log --oneline --graph --decorate
--all"
```

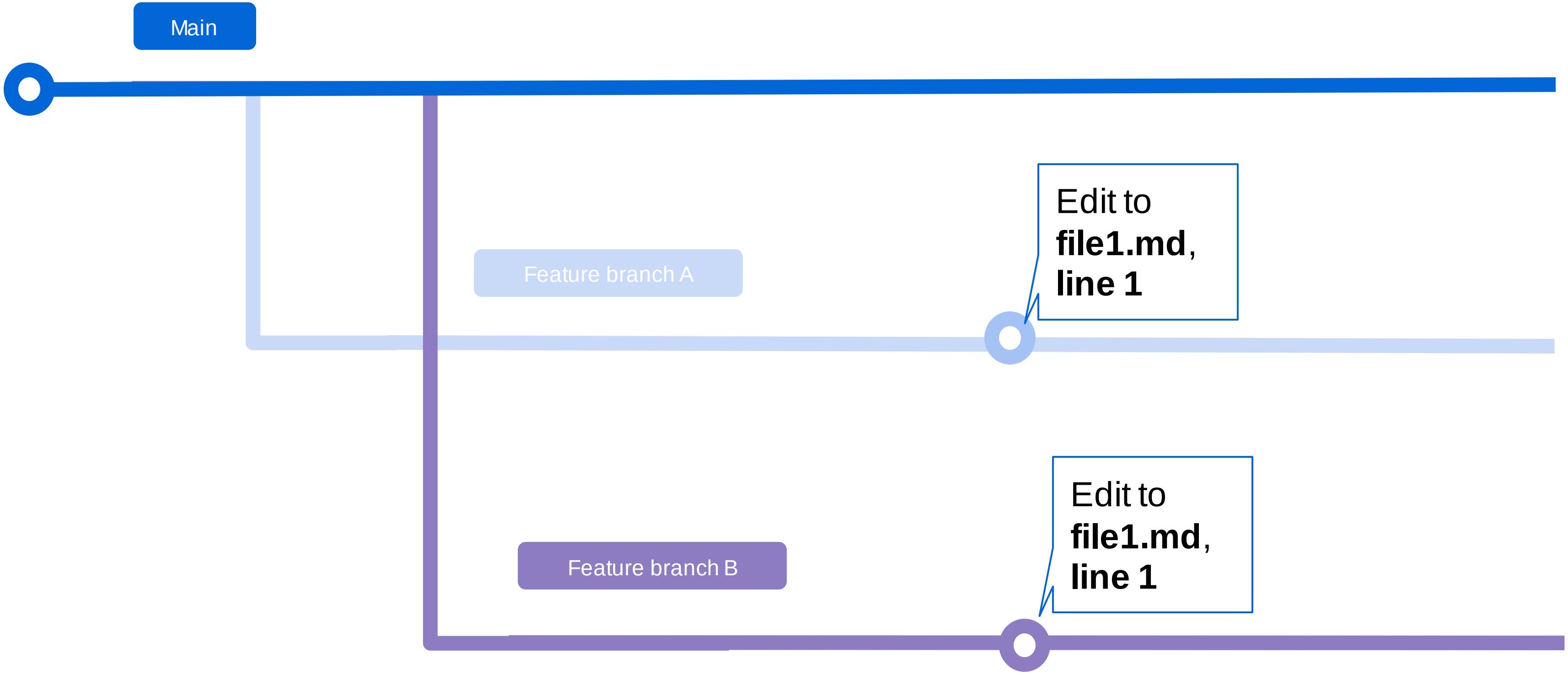
Using the Alias

```
git lol
```

Project 2:
Merge conflicts

Defining a merge conflict ● Resolving a merge conflict

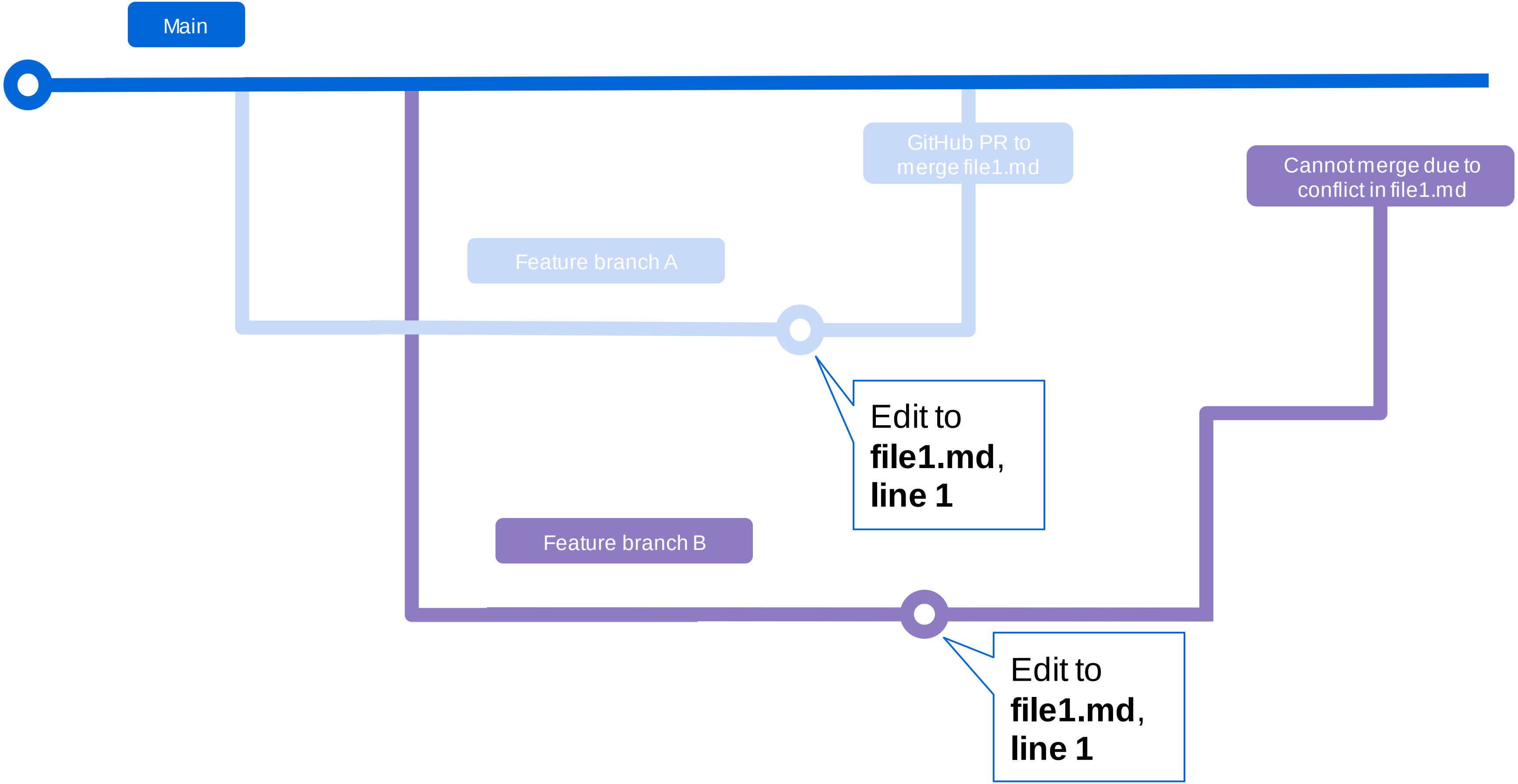
Defining a merge conflict



Edit to **file1.md**, **line 1**

Edit to **file1.md**, **line 1**

Defining a merge conflict

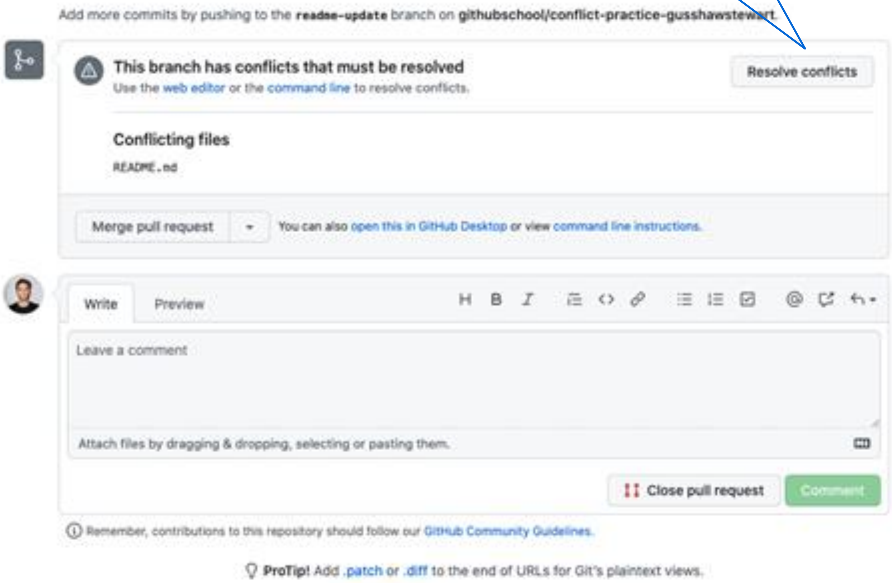

Main
GitHub PR to merge file1.md
Cannot merge due to conflict in file1.md
Feature branch A
Edit to **file1.md**, **line 1**
Feature branch B
Edit to **file1.md**, **line 1**

Project 2:
Merge conflicts

Defining a merge conflict ● Resolving a merge conflict

Fixing a merge conflict

Resolve conflict on UI



Add more commits by pushing to the `readme-update` branch on githubschool/conflict-practice-gusshawstewart.

This branch has conflicts that must be resolved

Use the web editor or the command line to resolve conflicts.

Conflicting files

`README.md`

You can also open this in GitHub Desktop or view command line instructions.

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

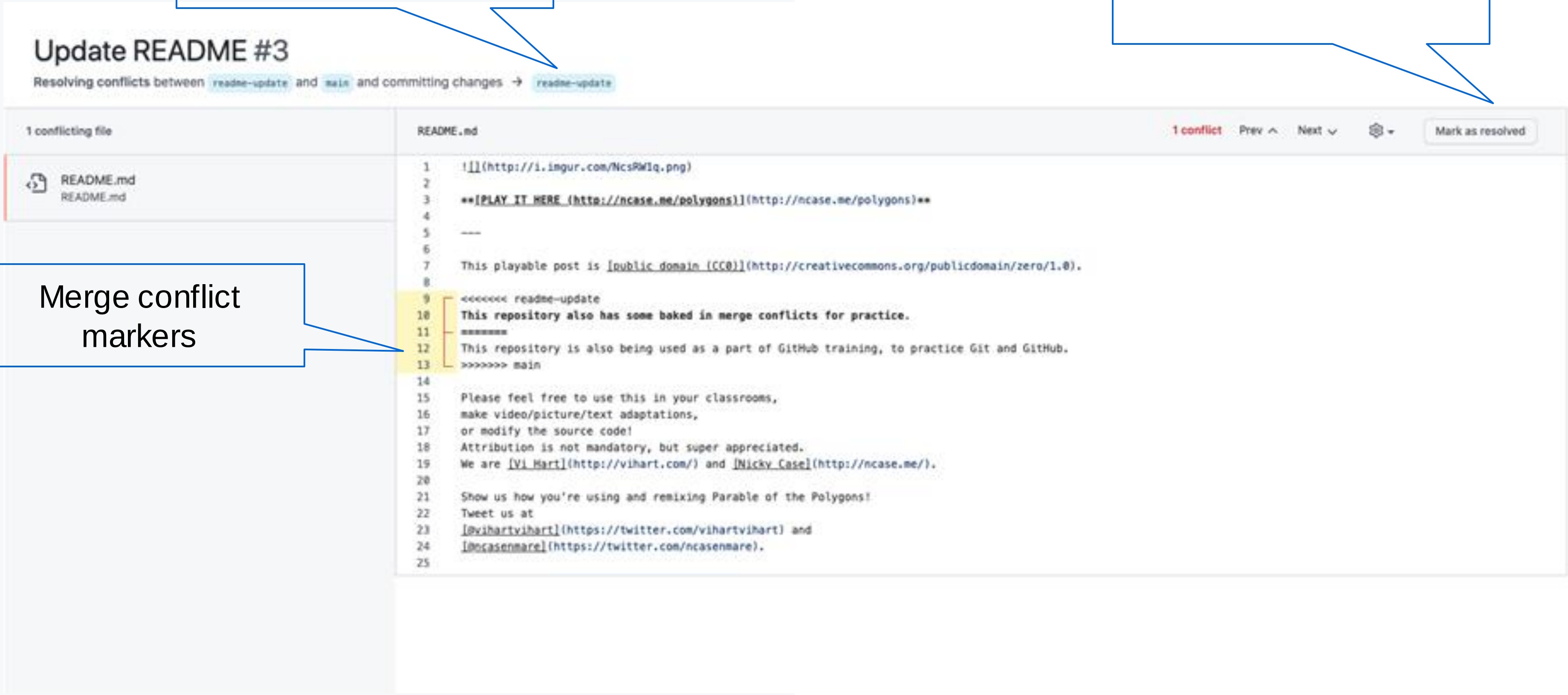
Remember, contributions to this repository should follow our GitHub Community Guidelines.

ProTip! Add .patch or .diff to the end of URLs for Git's plaintext views.

Conflict resolved in feature branch

Mark as resolved

Merge conflict markers



Update README #3

Resolving conflicts between `readme-update` and `main` and committing changes → `readme-update`

```
![](http://i.imgur.com/NcsRW1q.png)

**[PLAY IT HERE (http://ncase.me/polygons)](http://ncase.me/polygons)**

---

This playable post is [public domain (CC0)](http://creativecommons.org/publicdomain/zero/1.0).

<<<<<<< readme-update
This repository also has some baked in merge conflicts for practice.
=======
This repository is also being used as a part of GitHub training, to practice Git and GitHub.
>>>>>>> main

Please feel free to use this in your classrooms,
make video/picture/text adaptations,
or modify the source code!
Attribution is not mandatory, but super appreciated.
We are [Vi Hart](http://vihart.com/) and [Nicky Case](http://ncase.me/).

Show us how you're using and remixing Parable of the Polygons!
Tweet us at
[@vihartvihart](https://twitter.com/vihartvihart) and
[@ncasenmare](https://twitter.com/ncasenmare).

```

Demo

Activity 15: resolve a merge conflict

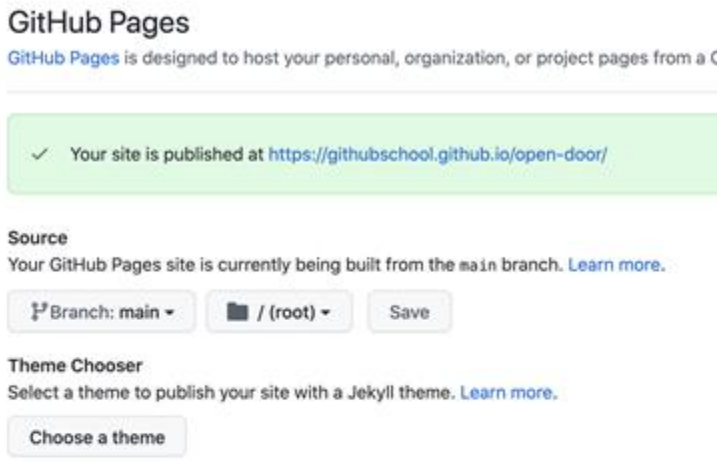
Repository: conflict-practice-”your-github-id”

1. Use the GitHub UI to resolve your conflict – see PR “Minor CSS fixes” and/or “Update README”
2. Use your command line to resolve the third merge conflict – see PR “Updates to game manual”

Project 3:
GitHub Games

- Workflow review
- Reverting Commits
- Tags and Releases
- Protected Branches and CODEOWNERS
- Helpful Git commands
- Branching strategy (Git flow)
- Git Bisect
- Viewing local changes

GitHub Pages



GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a G

✓ Your site is published at https://githubschool.github.io/open-door/

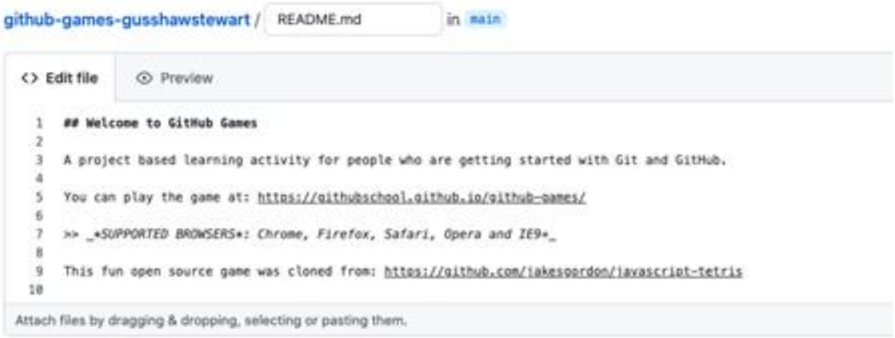
Source

Your GitHub Pages site is currently being built from the `main` branch. Learn more.

Theme Chooser

Select a theme to publish your site with a Jekyll theme. Learn more.

Workflow review



github-games-gusshawstewart / README.md in main
<> Edit file
Preview

```
## Welcome to GitHub Games

A project based learning activity for people who are getting started with Git and GitHub.

You can play the game at: https://githubschool.github.io/github-games/

>> _*SUPPORTED BROWSERS*: Chrome, Firefox, Safari, Opera and IE9+_

This fun open source game was cloned from: https://github.com/jakesgordon/javascript-tetris

```

Attach files by dragging & dropping, selecting or pasting them.

Demo

Activity 16: updating the README.md

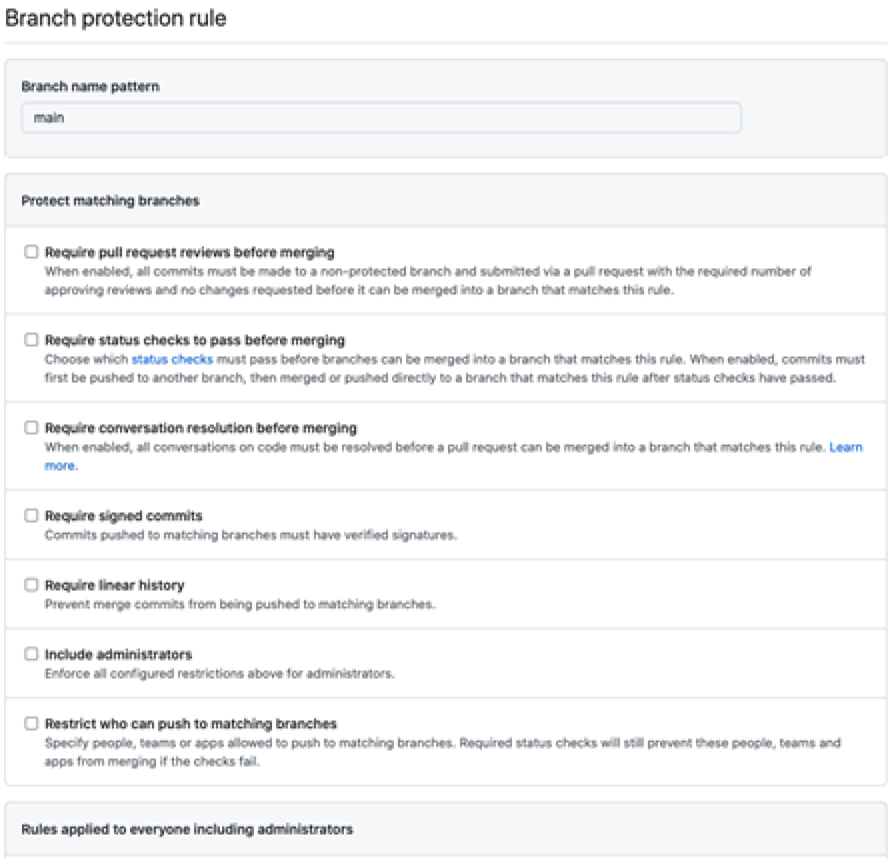
Repository: github-games-”your-github-id”

1. `git clone URL`
2. `git switch -c readme-update`
3. Edit the URL in the README.md.
4. Commit the changes to your branch.
5. `git push -u origin readme-update`
6. Create a Pull Request *in your repository* (base: `main`, compare: `readme-update`)
7. Merge your Pull Request.
8. Delete the branch on GitHub.
9. Update your local copy of the repository: `git pull --prune`

Project 3:
GitHub Games

- Workflow review
- Reverting Commits
- Tags and Releases
- Protected Branches and CODEOWNERS
- Helpful Git commands
- Branching strategy (Git flow)
- Git Bisect
- Viewing local changes

Protected Branches



Branch protection rule

Branch name pattern

main

Protect matching branches

□ **Require pull request reviews before merging**
When enabled, all commits must be made to a non-protected branch and submitted via a pull request with the required number of approving reviews and no changes requested before it can be merged into a branch that matches this rule.

□ **Require status checks to pass before merging**
Choose which status checks must pass before branches can be merged into a branch that matches this rule. When enabled, commits must first be pushed to another branch, then merged or pushed directly to a branch that matches this rule after status checks have passed.

□ **Require conversation resolution before merging**
When enabled, all conversations on code must be resolved before a pull request can be merged into a branch that matches this rule. Learn more.

□ **Require signed commits**
Commits pushed to matching branches must have verified signatures.

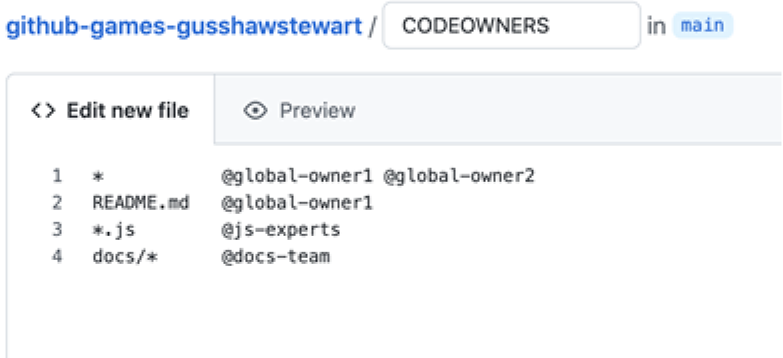
□ **Require linear history**
Prevent merge commits from being pushed to matching branches.

□ **Include administrators**
Enforce all configured restrictions above for administrators.

□ **Restrict who can push to matching branches**
Specify people, teams or apps allowed to push to matching branches. Required status checks will still prevent these people, teams and apps from merging if the checks fail.

Rules applied to everyone including administrators

CODEOWNERS



```
*           @global-owner1 @global-owner2
README.md   @global-owner1
*.js        @js-experts
docs/*      @docs-team
```

Demo

Activity 17: Protected Branches

Repository: github-games-your-github-id

1. Select the Settings tab.
2. Select Branches from the menu on the left side of the screen.
3. Click the Add rule button next to Branch protection rules.
4. In the Apply rule to textbox type the name of the branch you would like to protect, for example, `main`.
5. **Check Do not allow bypassing the above settings**
6. Click the Create button.

Activity 18: create a CODEOWNERS file

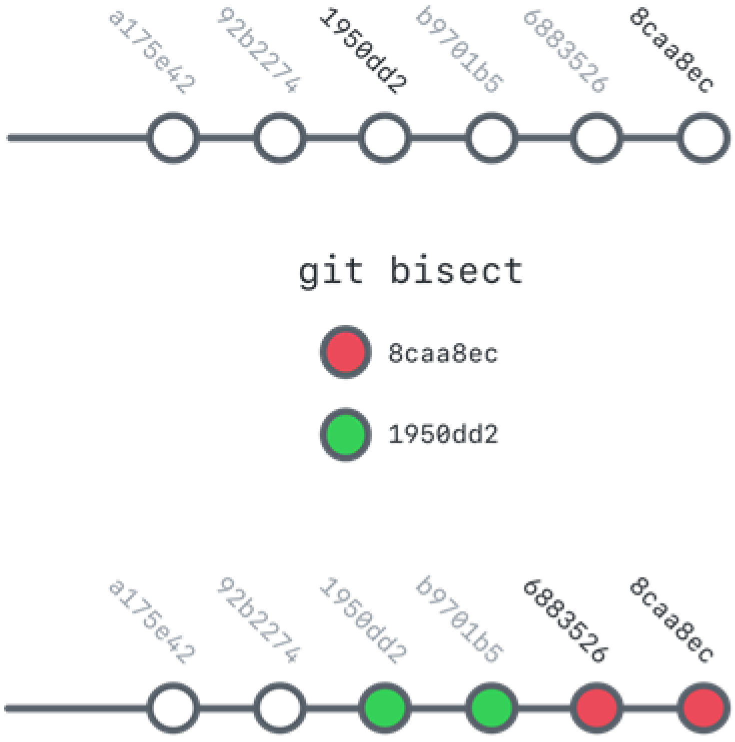
Repository: github-games-your-github-id

1. Go out to the Code tab of your repository.
2. Click the Create new file button.
3. In the Name your file... textbox enter `CODEOWNERS` (no extension necessary). You can add this to a `.github/` directory if desired by entering `.github/CODEOWNERS`.
4. On the first line, type `*          @YOUR_USERNAME`
   - This means that you will be the default owner for everything in the repo, unless a later match takes preference.
5. On the next line, type `*.js       @GITHUBTEACHER`
   - Order is important. The last matching pattern for a given change takes precedence.
6. Scroll down, and type a commit message into the Commit new file dialog box.
7. Click the Commit new file button to save your changes.
8. Now that you have created a CODEOWNERS file, go back to your branch protection settings and click the Edit button next to `main`.
9. Under Rule settings, select the option to Require pull request reviews before merging and Require review from Code Owners. Remember to click Save changes.

Project 3: GitHub Games

- Workflow review
- Reverting Commits
- Tags and Releases
- Protected Branches and CODEOWNERS
- Helpful Git commands
- Branching strategy (Git flow)
- Git Bisect
- Viewing local changes

Git Bisect


a175e42
92b2274
1950dd2
b9701b5
6883526
8caa8ec
git bisect
8caa8ec
1950dd2
a175e42
92b2274
1950dd2
b9701b5
6883526
8caa8ec

Demo

Activity 19: Git Bisect- the long way

Your local copy of Repository: github-games-yourGitHubId

1. **You will need to have cloned the github-games-YourUserId repo**
2. Initiate the binary search: `git bisect start`.
3. Specify the commit where you noticed the code was broken: `git bisect bad <SHA>`.
4. Specify the commit where you knew things were working: `git bisect good <SHA>`.
5. Bisect will check you out to the midpoint between good and bad.
6. Run a test to see if the game would work at this point. Our test is to use `ls` to see if an `index.html` file exists.
7. If the game is still broken (there *is no* `index.html` file), type: `git bisect bad`.
8. If the game works (and there *is* an `index.html` file), type: `git bisect good`.
9. Git will bisect again and wait for you to test. This will happen until Git has enough information to pinpoint the first bad commit.
10. When Git has detected the error, it will provide a message that `SHA is the first bad commit.`
11. Exit the bisect process: `git bisect reset`.

Activity 20: Git Bisect- the short way

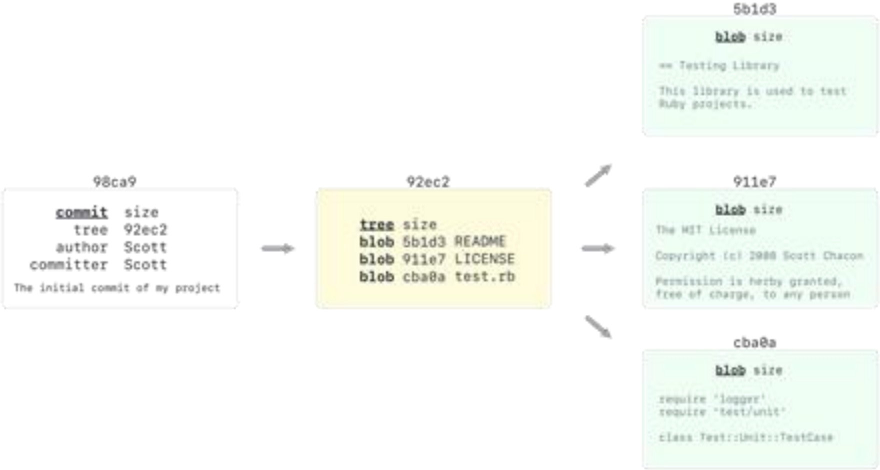
Your local copy of Repository: github-games-yourGitHubId

1. `git bisect start <bad-SHA> <good-SHA>`
2. `git bisect run ls index.html`
3. `git bisect reset`

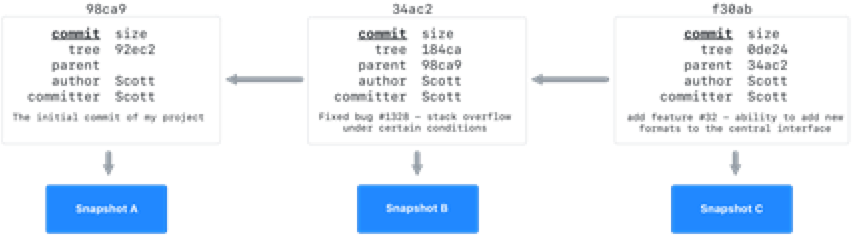
Project 3:
GitHub Games

- Workflow review
- Reverting Commits
- Tags and Releases
- Protected Branches and CODEOWNERS
- Helpful Git commands
- Branching strategy (Git flow)
- Git Bisect
- Viewing local changes

Reverting commits



Commit object structure



Each commit pointing to its parent

*Image source: ProGit v2 by Scott Chacon

Demo

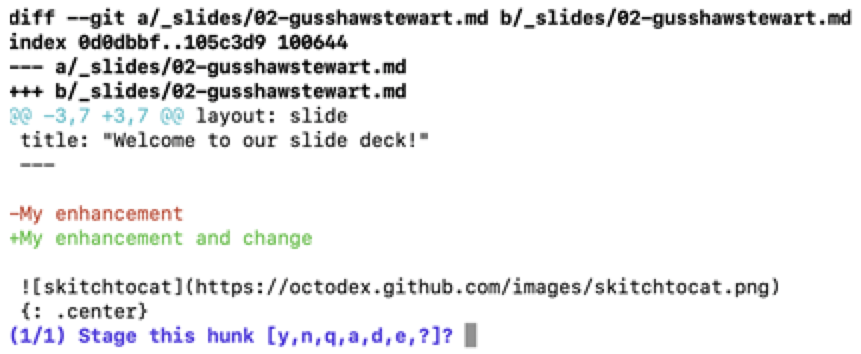
Activity: Revert a commit

1. Initialize the revert: `git revert <SHA>`
2. Type a commit message.
3. Push your changes to GitHub.

Project 3:
GitHub Games

- Workflow review
- Reverting Commits
- Tags and Releases
- Protected Branches and CODEOWNERS
- Helpful Git commands
- Branching strategy (Git flow)
- Git Bisect
- Viewing local changes

Helpful Git commands



```
diff --git a/_slides/02-gusshawstewart.md b/_slides/02-gusshawstewart.md
index 0d0dbbf..105c3d9 100644
--- a/_slides/02-gusshawstewart.md
+++ b/_slides/02-gusshawstewart.md
@@ -3,7 +3,7 @@ layout: slide
 title: "Welcome to our slide deck!"
 ---

-My enhancement
+My enhancement and change

 ![skitchtocat](https://octodex.github.com/images/skitchtocat.png)
 {: .center}
(1/1) Stage this hunk [y,n,q,a,d,e,?]?
```

Demo

Activity 21: moving and renaming files with Git

Your local copy of Repository: github-games-yourGitHubId

1. Create a new branch named `slow-down`.
2. On *line 9* of the `index.html` file, change the background url to `images/texture.jpg`.
3. On *line 78*, change the timing for the game to speed it up or slow it down.
4. Save your changes.
5. See what git is tracking:
6. `git status`
7. Create a new, empty directory:
8. `mkdir images`
9. Move the texture file into the directory with git:
10. `git mv texture.jpg images/texture.jpg`

Activity 22: staging chunks of changes

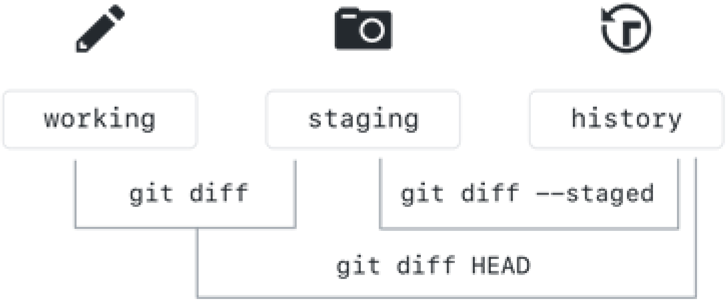
Your local copy of Repository: github-games-yourGitHubId

1. See what git is tracking:
2. `git status`
3. Move some parts of some files to the staging area with the `--patch` flag:
4. `git add -p`
5. Stage the hunk related to the image move: y
6. Leave the hunk related to the speed change in the working area: ‘n’

Project 3:
GitHub Games

- Workflow review
- Reverting Commits
- Tags and Releases
- Protected Branches and CODEOWNERS
- Helpful Git commands
- Branching strategy (Git flow)
- Git Bisect
- Viewing local changes

Viewing local changes



working

staging

history

git diff

git diff --staged

git diff HEAD

Demo

Activity 23: comparing changes within the local repository

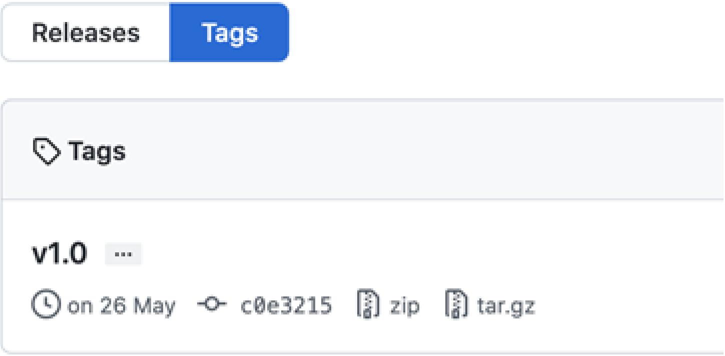
Your local copy of Repository: github-games-yourGitHubId

1. `git diff`
2. `git diff --staged`
3. `git diff HEAD`
4. `git diff --color-words`
5. `git diff <REF-1> <REF-2>`
6. `git diff main slow-down`
7. `git diff origin/main main`
8. `git diff 2710 b745`

Project 3:
GitHub Games

- Workflow review
- Protected Branches and CODEOWNERS
- Git Bisect
- Reverting Commits
- Helpful Git commands
- Viewing local changes
- Tags and Releases
- Branching strategy (Git flow)

Tags

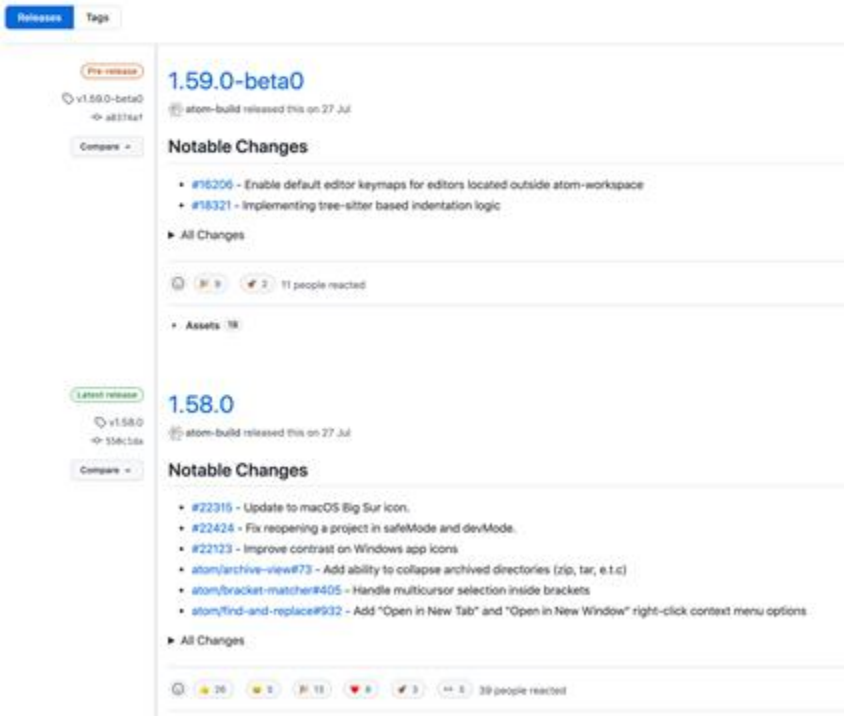


Tags

v1.0 ···

on 26 May c0e3215 zip tar.gz

Releases



1.59.0-beta0

atom-build released this on 27 Jul

Notable Changes

- #16206 - Enable default editor keymaps for editors located outside atom-workspace
- #18321 - Implementing tree-sitter based indentation logic

▶ All Changes

9 2 11 people reacted

Assets 18

1.58.0

atom-build released this on 27 Jul

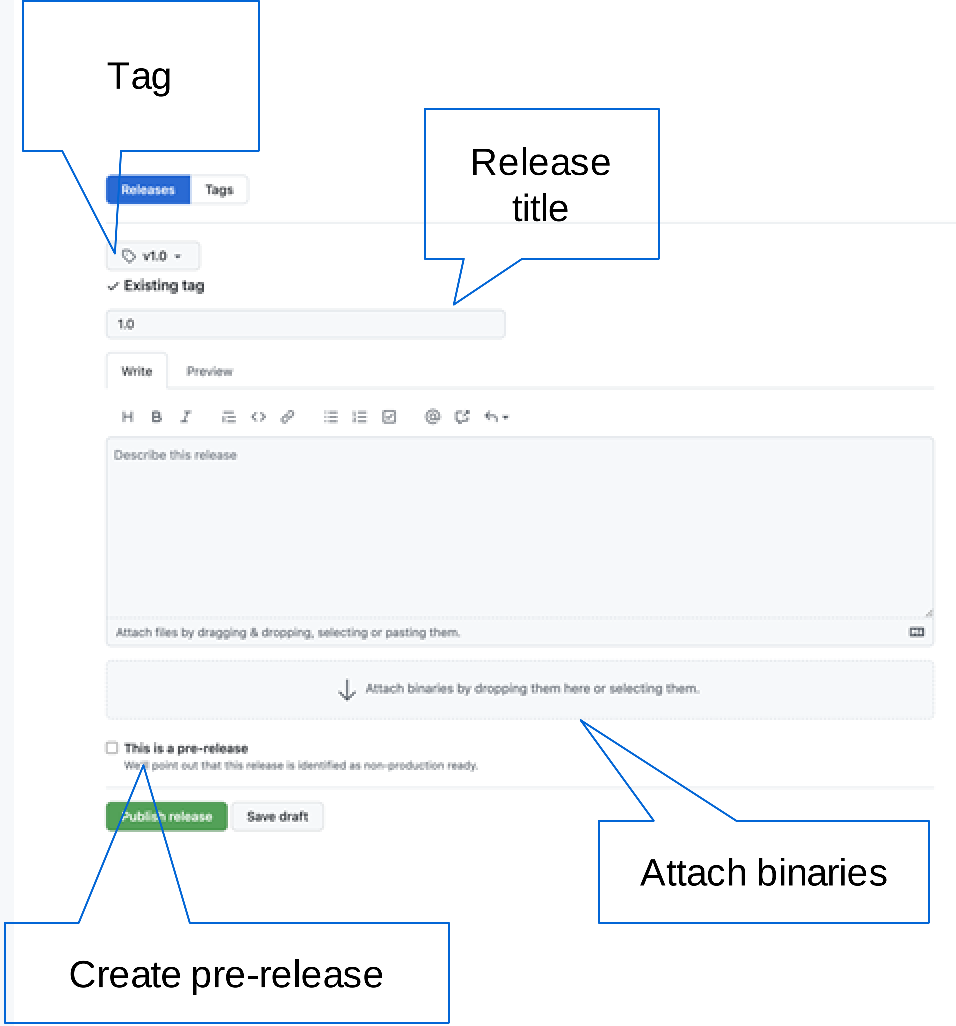
Notable Changes

- #22316 - Update to macOS Big Sur icon.
- #22424 - Fix reopening a project in safeMode and devMode.
- #22123 - Improve contrast on Windows app icons
- atom/archive-view#73 - Add ability to collapse archived directories (zip, tar, e.t.c)
- atom/bracket-matcher#405 - Handle multicursor selection inside brackets
- atom/find-and-replace#932 - Add "Open in New Tab" and "Open in New Window" right-click context menu options

▶ All Changes

26 5 13 6 3 5 39 people reacted

Creating a release



Tag

Release title

v1.0

✓ Existing tag

1.0

Describe this release

Attach files by dragging & dropping, selecting or pasting them.

Attach binaries by dropping them here or selecting them.

☐ This is a pre-release

We'll point out that this release is identified as non-production ready.

Attach binaries

Create pre-release

Demo

Activity 24: Tags

Your local copy of Repository: github-games-yourGitHubId

1. git log --oneline
2. git tag v1.0 <SHA>
3. git show v1.0
4. git tag -a v2.0 <SHA>
5. git push --tags
6. git config --global push.followTags true

Activity 25: Releases

Repository: github-games-YourGitHubID

1. On GitHub, navigate to the Code tab of the repository.
2. Under your repository name, click Releases.
3. Click Create new release.
4. Type a name for the tag. We recommend you use semantic versioning.
5. Select a branch that contains the project you want to release. Usually, you'll want to release against your main branch, unless you're releasing beta software. You can also select a recent commit by choosing the recent commits tab.
6. Type a title and description that describes your release.
7. If you're ready to publicize your release, click Publish release. Otherwise, click Save draft to work on it later.

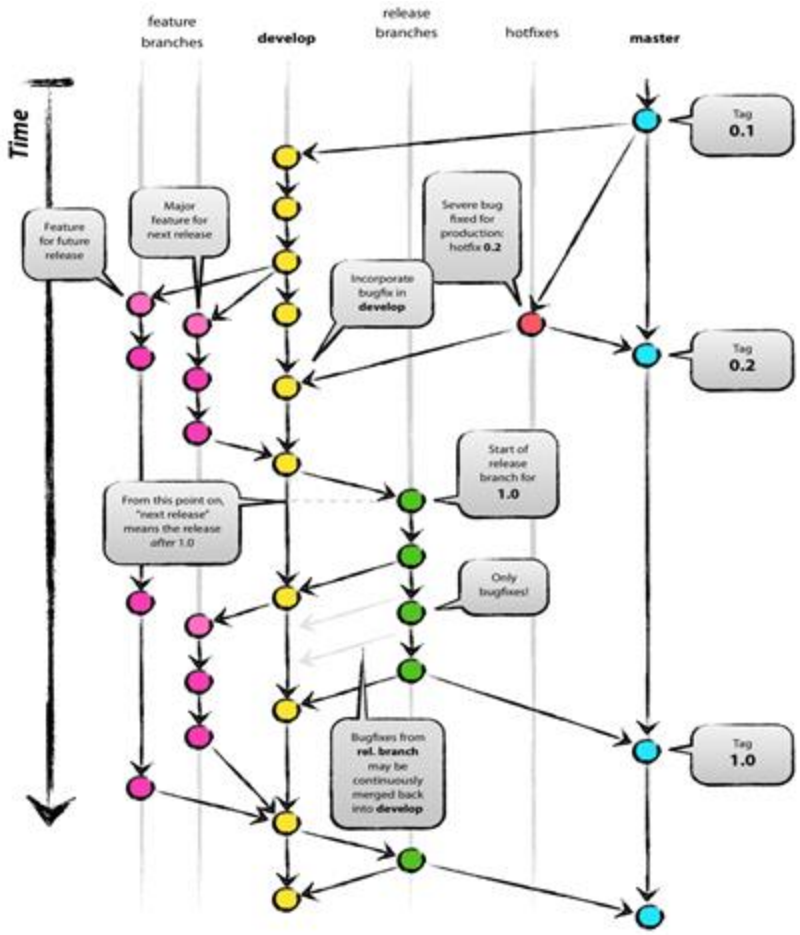
Project 3:
GitHub Games

- Workflow review
- Reverting Commits
- Tags and Releases
- Protected Branches and CODEOWNERS
- Helpful Git commands
- Branching strategy (Git flow)
- Git Bisect
- Viewing local changes

Branching Strategy

Git Flow

Git Flow


feature branches
develop
release branches
hotfixes
master
Time
Tag 0.1
Feature for future release
Major feature for next release
Severe bug fixed for production: hotfix 0.2
Incorporate bugfix in develop
Tag 0.2
Start of release branch for 1.0
From this point on, "next release" means the release after 1.0
Only bugfixes!
Bugfixes from rel. branch may be continuously merged back into develop
Tag 1.0

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

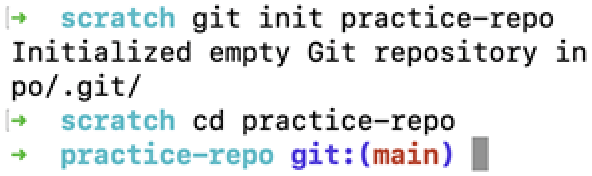
Activity 26: workflow discussion

1. Which branching strategy will we use?
2. Which branch will serve as our "main" or deployed code?
3. How will you protect your code?
4. Will we use naming conventions for our branches?
5. How will we use labels and assignees?
6. Will we use milestones?
7. Will we have required elements of Issues or Pull Requests (e.g. shipping checklists)?
8. Who is expected to review your work? Do you plan to involve other teams?
9. How will we indicate sign-off on Pull Requests?
10. Who will merge pull requests?
11. How will you teach your workflow to your team? If it already exists, how is it taught to new hires?
12. What integrations will be used in different stages of development? Will all teams be using the same tools?
13. If users have questions about Git, GitHub, or their workflows, who do they ask? How do they know who to ask?

Project 4:
Local repository

- Create a local repo
- Fixing commit mistakes
- Rewriting history with Git Reset
- Cherry picking
- Merge strategies

Create a local repo



```
➜  scratch git init practice-repo
Initialized empty Git repository in
po/.git/
➜  scratch cd practice-repo
➜  practice-repo git:(main)
```

Demo

Activity 27: initialize a new local repository

1. Navigate to the directory where you will place your practice repo (`cd ..` to get back to the parent folder).
2. Create a new directory and initialize it as a git repository: `git init practice-repo`
3. CD into your new repository: `cd practice-repo`
4. Create an empty new file named `README.md`:
   - Bash: `touch README.md`
   - PowerShell: `Out-File README.md`
5. Add and commit the README.md file.

Bash:

```
for d in {1..6}; do touch "file${d}.md";
git add "file${d}.md"; git commit -m
"adding file ${d}"; done
```

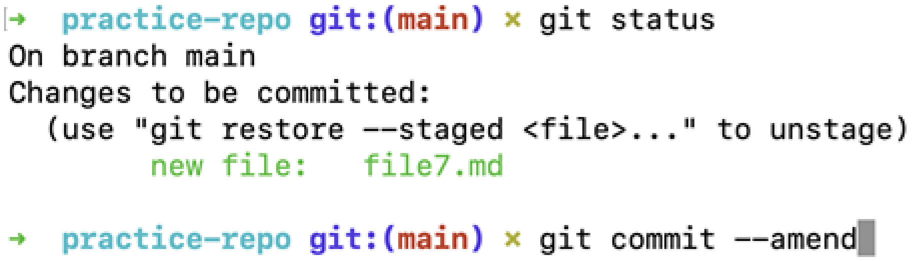
PowerShell:

```
for ($d=1; $d -le 6; $d++) { Out-File
file$d.md; git add file$d.md; git commit -
m "adding file$d.md"; }
```

Project 4:
Local repository

- Create a local repo
- Fixing commit mistakes
- Rewriting history with Git Reset
- Cherry picking
- Merge strategies

Fixing commit mistakes



```
➜  practice-repo git:(main) ✗ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   file7.md

➜  practice-repo git:(main) ✗ git commit --amend
```

Demo

Activity 28: revising your last commit

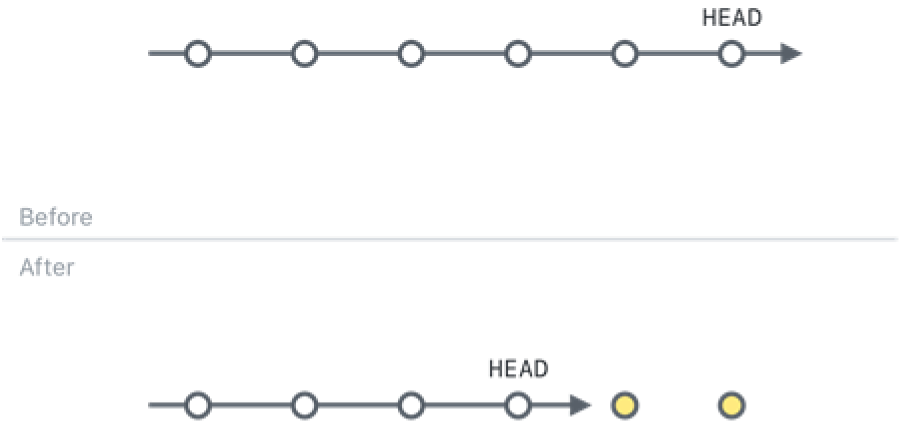
Use your newly created local repo

1. Create a new file:
   - Bash: `touch file7.md`
   - PowerShell: `Out-File file7.md`
2. When you are adding files to the previous commit, they should be in the staging area. Move your file to the staging area: `git add file7.md`
3. `git commit --amend`
4. The text editor will open, allowing you to edit your commit message.

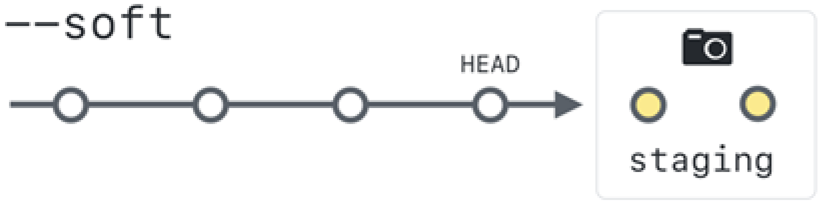
Project 4: Local repository

- Create a local repo
- Fixing commit mistakes
- Rewriting history with Git Reset
- Cherry picking
- Merge strategies

Rewriting history with Git Reset



Reset Soft



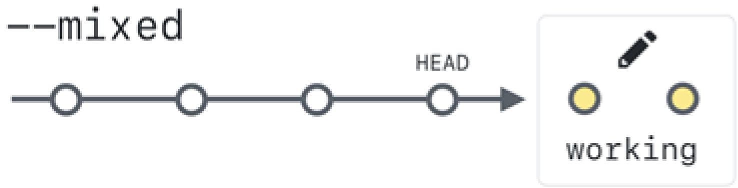
Demo

Activity 29: Reset Soft

Use your newly created local repo

1. View the history of our project: `git log --oneline --decorate`
2. Identify the current location of `HEAD`.
3. Go back two commits in history: `git reset --soft HEAD~2`
4. See the tip of our branch (and `HEAD`) is now sitting two commits earlier than it was before: `git log --oneline --decorate`
5. The changes we made in the last two commits should be in the staging area: `git status`
6. All the files still exist locally: `ls or dir`
7. Let's remove the extra file we created earlier: `git rm --cached file7.md`
8. Now, we'll re-commit these changes without the extra file: `git commit -m "re-add file 5 and 6"`

Reset Mixed



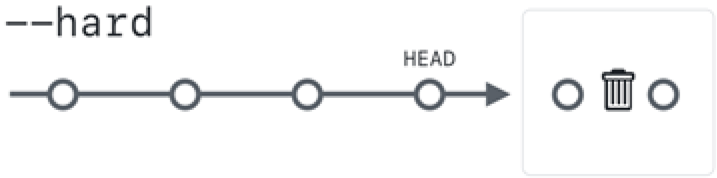
Demo

Activity 30: Reset Mixed

Use your newly created local repo

1. Once again, we will start by viewing the history of our project: `git log --oneline`
2. Go back one commit in history: `git reset HEAD~`
3. See where the tip of the branch is pointing: `git log --oneline --decorate`
4. The changes we made in the last commit have been moved back to the working directory: `git status`
5. All the files still exist locally: `ls or dir`
6. Move the files to the staging area before we can commit them: `git add file5.md file6.md`
7. Re-commit the files: `git commit -m "re-add file 5 and 6"`

Reset Hard



Demo

Activity 31: Reset Hard

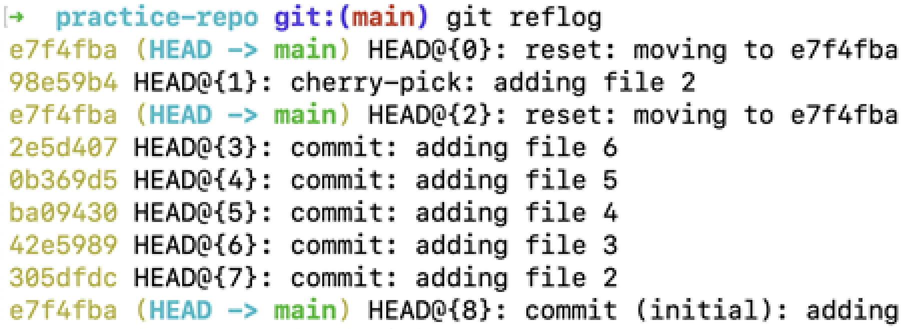
Use your newly created local repo

1. Start by viewing the history of our project with: `git log --oneline`
2. Reset to the point in time where the only file that existed was the README.md: `git reset --hard <SHA>`
3. See that all of the commits are gone: `git log --oneline`
4. Notice your working directory is clean: `git status`
5. See that the only files in your repository are the README.md and file7.md: `ls or dir`

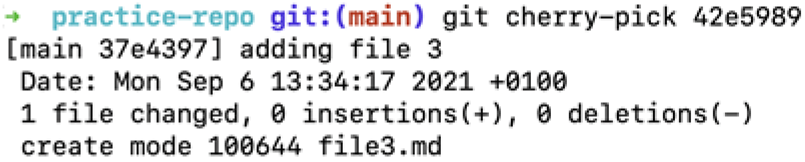
Project 4:
Local repository

- Create a local repo
- Fixing commit mistakes
- Rewriting history with Git Reset
- Cherry picking
- Merge strategies

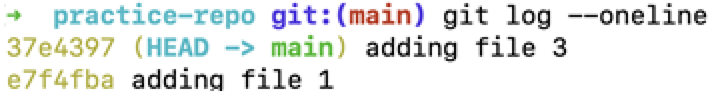
Cherry picking and reflog



```
➜  practice-repo git:(main) git reflog
e7f4fba (HEAD -> main) HEAD@{0}: reset: moving to e7f4fba
98e59b4 HEAD@{1}: cherry-pick: adding file 2
e7f4fba (HEAD -> main) HEAD@{2}: reset: moving to e7f4fba
2e5d407 HEAD@{3}: commit: adding file 6
0b369d5 HEAD@{4}: commit: adding file 5
ba09430 HEAD@{5}: commit: adding file 4
42e5989 HEAD@{6}: commit: adding file 3
305dfdc HEAD@{7}: commit: adding file 2
e7f4fba (HEAD -> main) HEAD@{8}: commit (initial): adding
```



```
➜  practice-repo git:(main) git cherry-pick 42e5989
[main 37e4397] adding file 3
 Date: Mon Sep 6 13:34:17 2021 +0100
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file3.md
```



```
➜  practice-repo git:(main) git log --oneline
37e4397 (HEAD -> main) adding file 3
e7f4fba adding file 1
```

Demo

Activity 32: cherry-pick a commit

Use your newly created local repo

1. Find the commit ID where you added file4.md: `git reflog`
2. Cherry-pick that commit: `git cherry-pick <SHA>`

Activity 33: directional reset

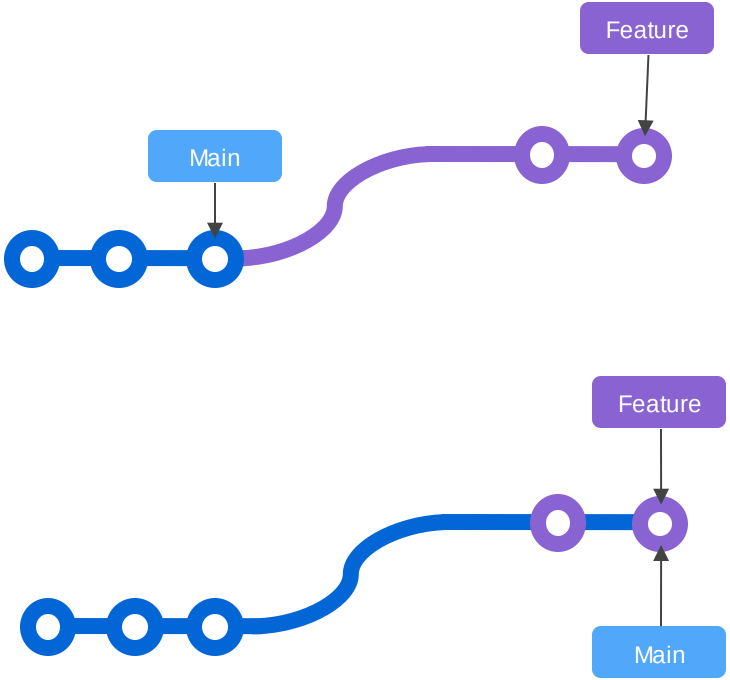
Use your newly created local repo

1. View the history of everywhere HEAD has pointed: `git reflog`
2. Reset to the point in time where the original `file6.md` was created: `git reset --hard <SHA>`
3. See your restored history: `git log --oneline`

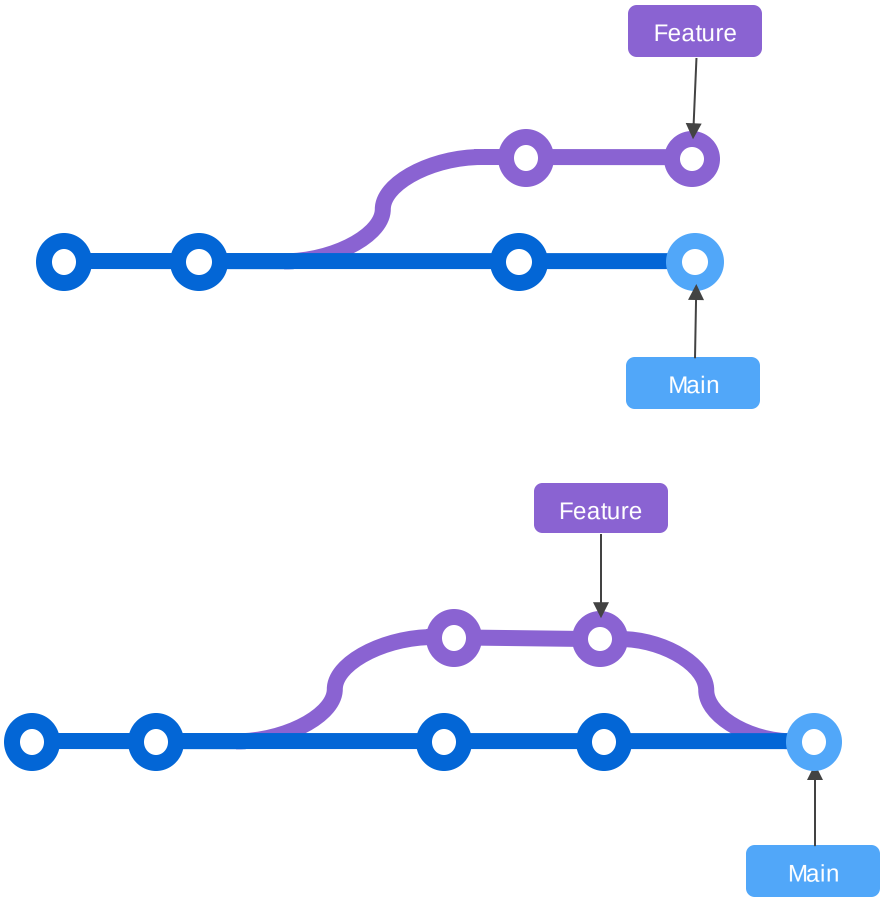
Project 4:
Local repository

- Create a local repo
- Fixing commit mistakes
- Rewriting history with Git Reset
- Cherry picking
- Merge strategies

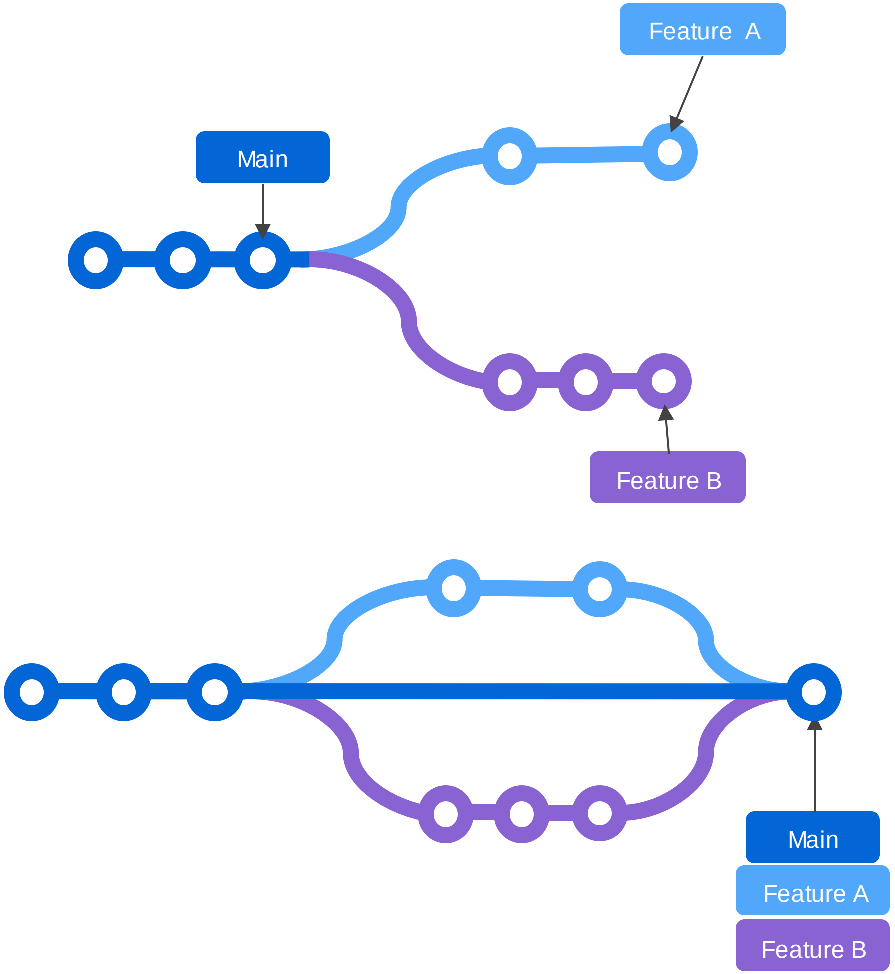
Merge strategies: fast-forward merge



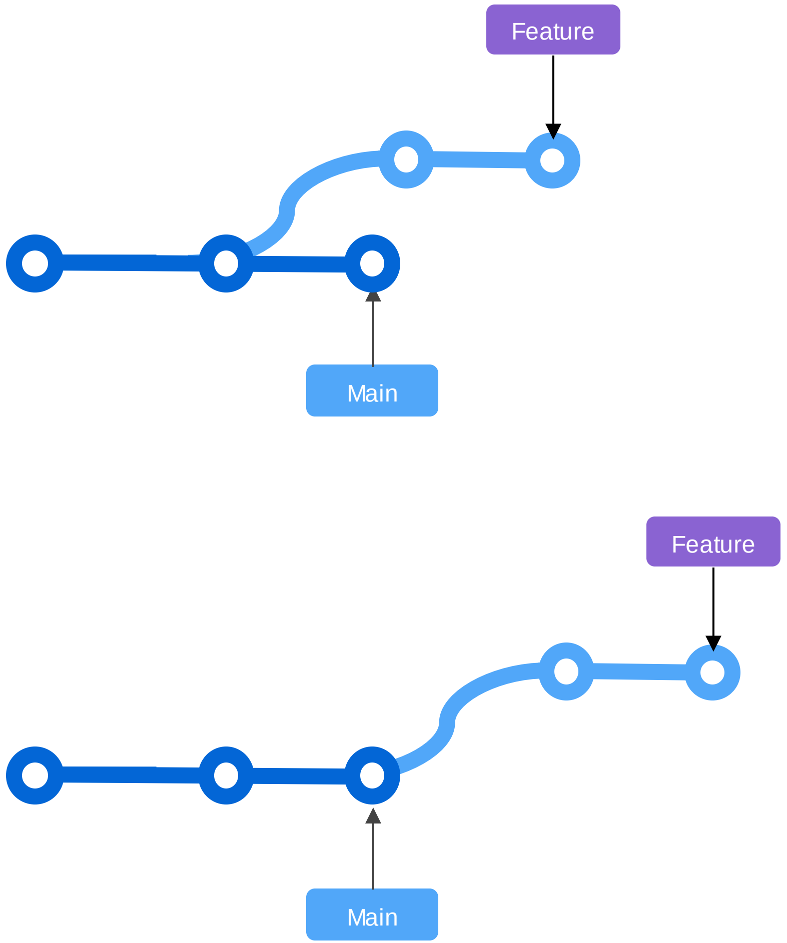
Merge strategies: Recursive merge



Merge strategies: Octopus merge



Git Rebase



Demo

Activity 34: setup to rebase

Use your newly created local repo

1. Find the SHA of the initial commit: `git log --oneline`
2. Reset to the SHA of the initial commit: `git reset --hard SHA`
3. Create a new branch and check out to it: `git switch -c rebase-me`
4. Cherry-pick files 4-6 onto the `rebase-me` branch using the reflog.
5. Switch to main: `git switch main`
6. Cherry-pick files 1-3 onto the `main` branch using the reflog.
7. Look at your history: `git log --oneline --graph --decorate --all`
8. If you merged now, it would be a recursive merge.

Activity 35: Begin the rebase

Use your newly created local repo

1. Switch to the `rebase-me` branch: `git switch rebase-me`
2. Start the merge: `git rebase -i main`
3. Your text editor will open, allowing you to see the commits to be rebased.
4. Save and close the `rebase-todo`.
5. Watch your rebase happen on the command line.
6. Take another look at your history: `git log --oneline --graph --decorate --all`
7. If you merged now, it would be a fast-forward merge.

Finish the Merge

1. Switch to main, the branch you will merge into: `git switch main`
2. Merge your changes in to main: `git merge rebase-me`

Recap

Q&A

Appendix

1. https://githubtraining.github.io/training-manual/#/01_getting_ready_for_class
2. https://guides.github.com/introduction/flow/
3. https://training.github.com/downloads/github-git-cheat-sheet.pdf
4. http://git-school.github.io/visualizing-git/

Thank you