CS304
Software Engineering

TAN, Shin Hwei

陈馨慧

Southern University of Science and Technology

https://lab.github.com/githubtraining/introduction-to-github

Lab today

- Learn about Git commands
- Take a short course on Introduction to GitHub

Git commands

command	description
`git clone` ***`url [dir]`***	copy a git repository so you can add to it
`git add` ***`files`***	adds file contents to the staging area
`git commit`	records a snapshot of the staging area
`git status`	view the status of your files in the working directory and staging area
`git diff`	shows diff of what is staged and what is modified but unstaged
`git help` ***`[command]`***	get help info about a particular command
`git pull`	fetch from a remote repo and try to merge into the current branch
`git push`	push your new branches and data to a remote repository
others: `init`, `reset`, `branch`, `checkout`, `merge`, `log`, `tag`	

Get ready to use Git!

1. Set the name and email for Git to use when you commit:

```
$ git config --global user.name “Bugs Bunny”
$ git config --global user.email bugs@gmail.com
```

- You can call **`git config –list`** to verify these are set.
- These will be set globally for all Git projects you work with.
- You can also set variables on a project-only basis by not using the **`--global`** flag.
- You can also set the editor that is used for writing commit messages:
  $ git config --global core.editor emacs (it is vim by default)

Create a local copy of a repo

2. Two common scenarios: (only do one of these)

   a) To **clone an already existing repo** to your current directory:

   ```
   $ git clone <url> [local dir name]
   ```

   This will create a directory named *local dir name*, containing a working copy of the files from the repo, and a **.git** directory (used to hold the staging area and your actual repo)

   b) To **create a Git repo** in your current directory:

   ```
   $ git init
   ```

   This will create a **.git** directory in your current directory.

   Then you can commit files in that directory into the repo:

   ```
   $ git add file1.java
   $ git commit –m “initial project version”
   ```

Committing files

- The first time we ask a file to be tracked, *and* **every time before we commit a file** we must add it to the staging area:

```
$ git add README.txt hello.java
```

This takes a snapshot of these files at this point in time and adds it to the staging area.

- To move staged changes into the repo we commit:

```
$ git commit –m “Fixing bug #22”
```

Note: To unstage a change on a file before you have committed it:

```
$ git reset HEAD  --  filename
```

Note: To unmodify a modified file:

```
$ git checkout  --  filename
```

Note: These commands are just acting on **your local version of repo**.

Status and Diff

- To view the **status** of your files in the working directory and staging area:

```
$ git status
```

or

```
$ git status -s
```

(`-s` shows a short one line version similar to svn)

- To see what is modified but unstaged:

```
$ git diff
```

- To see staged changes:

```
$ git diff --cached
```

After editing a file…

```
[rea@attu1 superstar]$ vim rea.txt
[rea@attu1 superstar]$ git status
# On branch master
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working directory)
#
#       modified:   rea.txt
#
no changes added to commit (use "git add" and/or "git commit -a")
[rea@attu1 superstar]$ git status -s
 M rea.txt                                  ← Note: M is in second column = “working  tree”
[rea@attu1 superstar]$ git diff             ← Shows modifications that have not been staged.
diff --git a/rea.txt b/rea.txt
index 66b293d..90b65fd 100644
--- a/rea.txt
+++ b/rea.txt
@@ -1,2 +1,4 @@
 Here is rea's file.
+
+One new line added.
[rea@attu1 superstar]$ git diff --cached    ← Shows nothing, no modifications have been staged yet.
```

After adding file to staging area…

```
[rea@attu1 superstar]$ git add rea.txt
[rea@attu1 superstar]$ git status
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   rea.txt
#
[rea@attu1 superstar]$ git status -s
M  rea.txt                                          ← Note: M is in first column = “staging area”
[rea@attu1 superstar]$ git diff                     ← Note: Shows nothing, no modifications that have not been staged.
[rea@attu1 superstar]$ git diff --cached            ← Note: Shows staged modifications.
diff --git a/rea.txt b/rea.txt
index 66b293d..90b65fd 100644
--- a/rea.txt
+++ b/rea.txt
@@ -1,2 +1,4 @@
 Here is rea's file.
+
+One new line added.
```

Viewing logs

To see a log of all changes in your local repo:

- `$ git log` or
- `$ git log --oneline` (to show a shorter version)

1677b2d Edited first line of readme

258efa7 Added line to readme

0e52da7 Initial commit

- `git log -5` (to show only the 5 most recent updates, etc.)

Note: changes will be listed by commitID #, (SHA-1 hash)

Note: changes made to the remote repo before the last time you cloned/pulled from it will also be included here

Pulling and Pushing

Good practice:

1. **Add** and **Commit** your changes to your local repo
2. **Pull** from remote repo to get most recent changes (fix conflicts if necessary, add and commit them to your local repo)
3. **Push** your changes to the remote repo

~~To fetch the most recent updates from the remote repo into your local~~ repo, and put them into your working directory:

```
$ git pull origin master
```

To push your changes from your local repo to the remote repo:

```
$ git push origin master
```

Notes: **`origin`** = an alias for the URL you cloned from

`master` = the remote branch you are pulling from/pushing to, (the local branch you are pulling to/pushing from is your current branch)

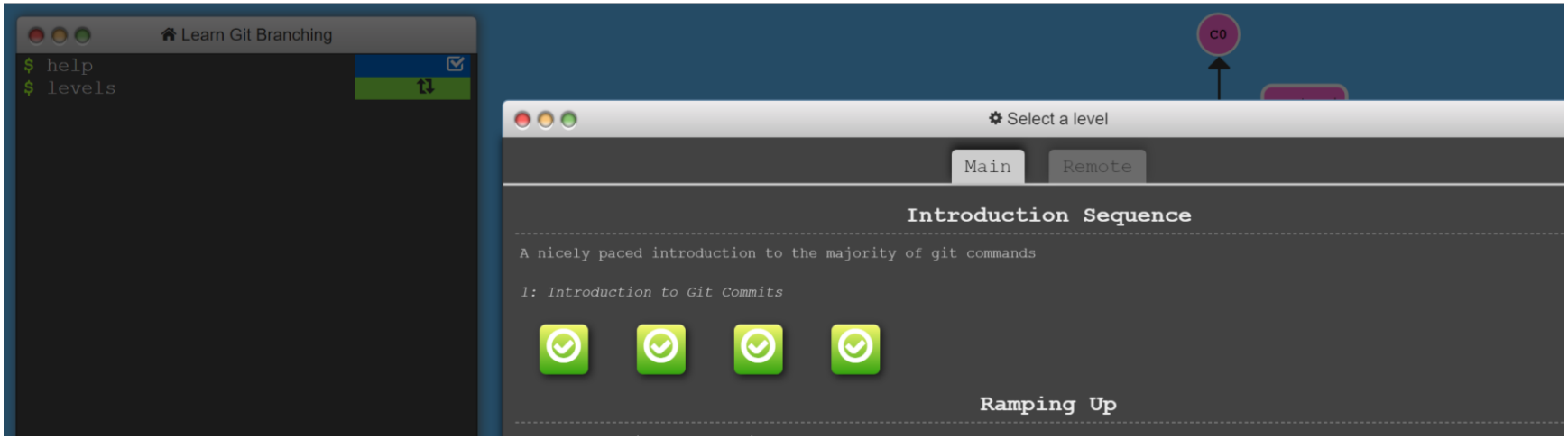
Try Git

- Go to https://learngitbranching.js.org/
- Follow the instruction of the game

You have ~20 minutes

Finish the Introduction Sequence

- Show this screen to me

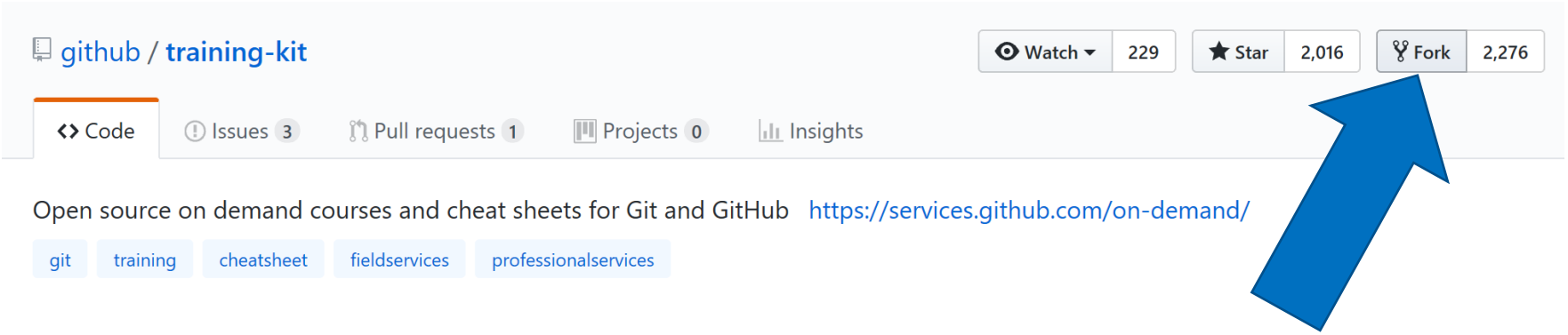


Part 2: Lab today

- You will take a short course on Introduction to GitHub!
- In this course, you'll learn how to:
  - Communicate in issues
  - Manage notifications
  - Create branches
  - Make commits
  - Introduce changes with pull requests

Step 1: Fork

- Login to GitHub: https://github.com/
- Go to: https://github.com/github/training-kit



Fork the training-kit repository

Step 2: Go to link

- Go to the link: https://lab.github.com/githubtraining/introduction-to-github

Install GitHub Learning Lab

- Install GitHub Learning Lab on “training-kit”

- Need tighter permissions? You can also install the app each time you register for a new course/repository.

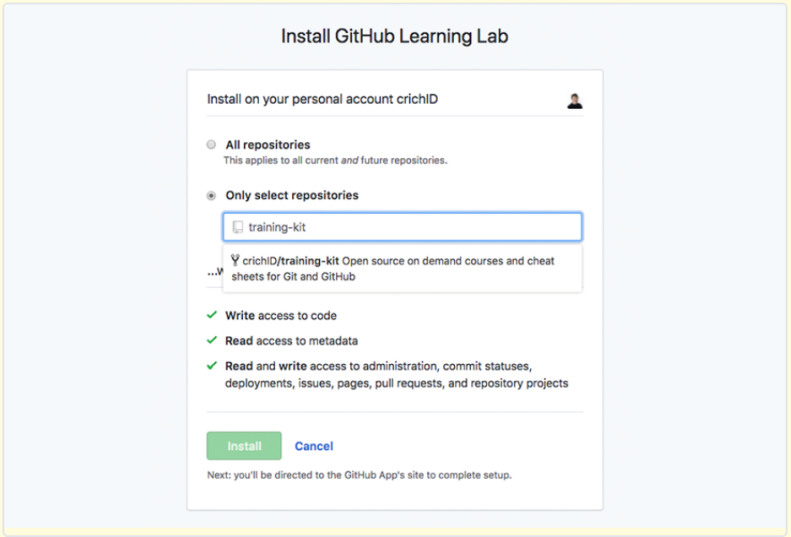
Limit GitHub Learning Lab to Course Repositories

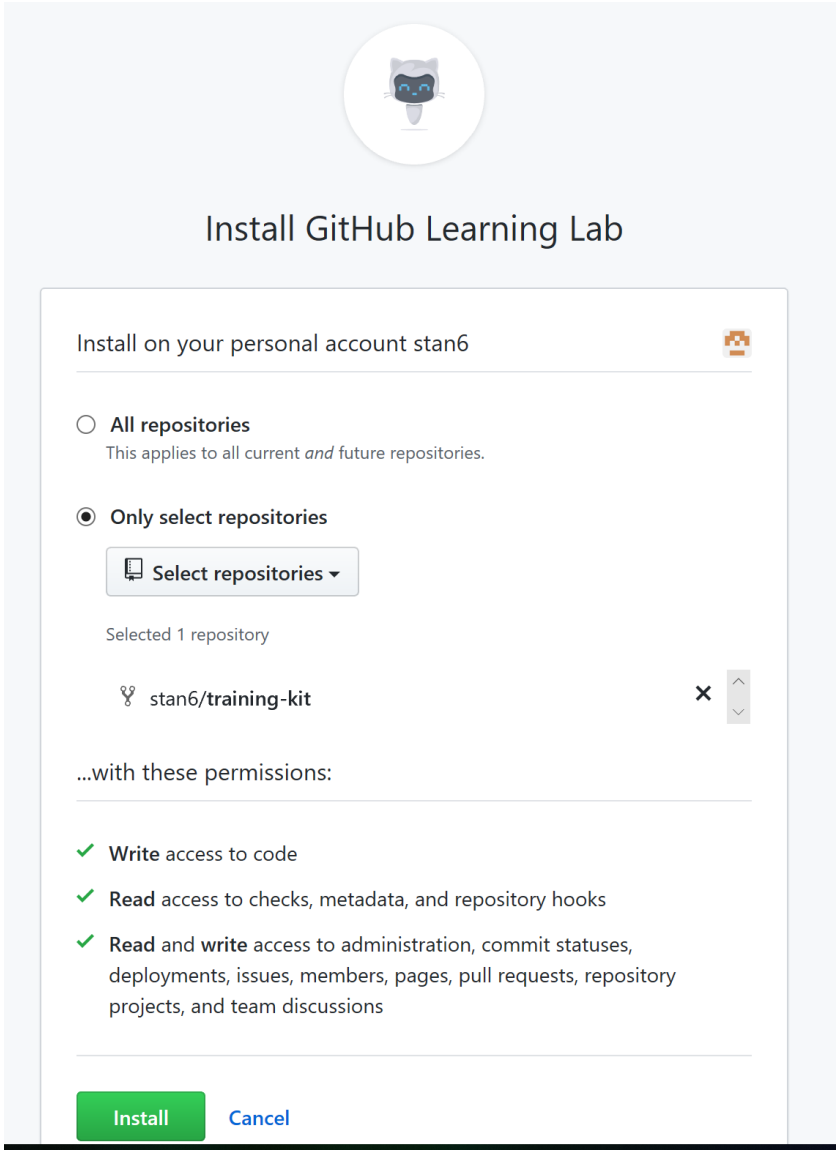
You can install the app on any repository you want. Feel free to fork the `github/training-kit` repository as a stand-in.

What's a fork?

Choose to install the app on the repository you've chosen. Once you've done this, close these instructions out and try to register for a new course.

Install the app

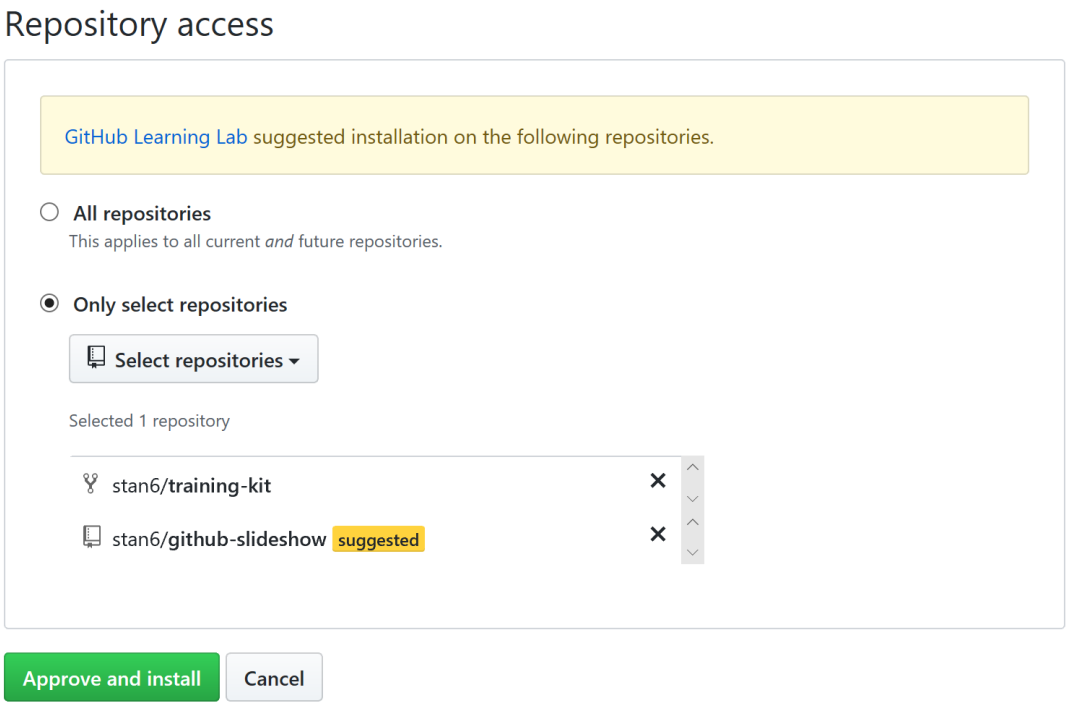



Install GitHub Learning Lab
Install on your personal account stan6
All repositories
This applies to all current and future repositories.
Only select repositories
Select repositories
Selected 1 repository
stan6/training-kit
...with these permissions:
Write access to code
Read access to checks, metadata, and repository hooks
Read and write access to administration, commit statuses, deployments, issues, members, pages, pull requests, repository projects, and team discussions
Install
Cancel

Step 3: Go to new course

- Go to the link: https://lab.github.com/githubtraining/introduction-to-github

Approve and Install



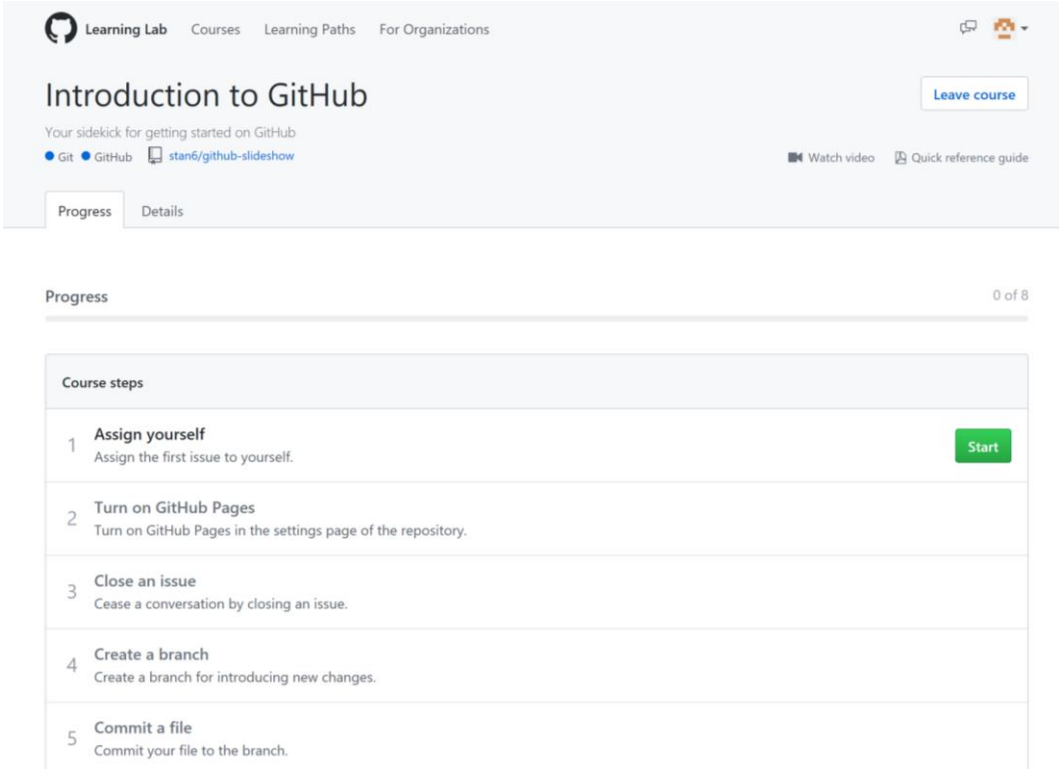
Uninstall GitHub Learning Lab

When you uninstall **GitHub Learning Lab**, it will be removed from all your repositories.

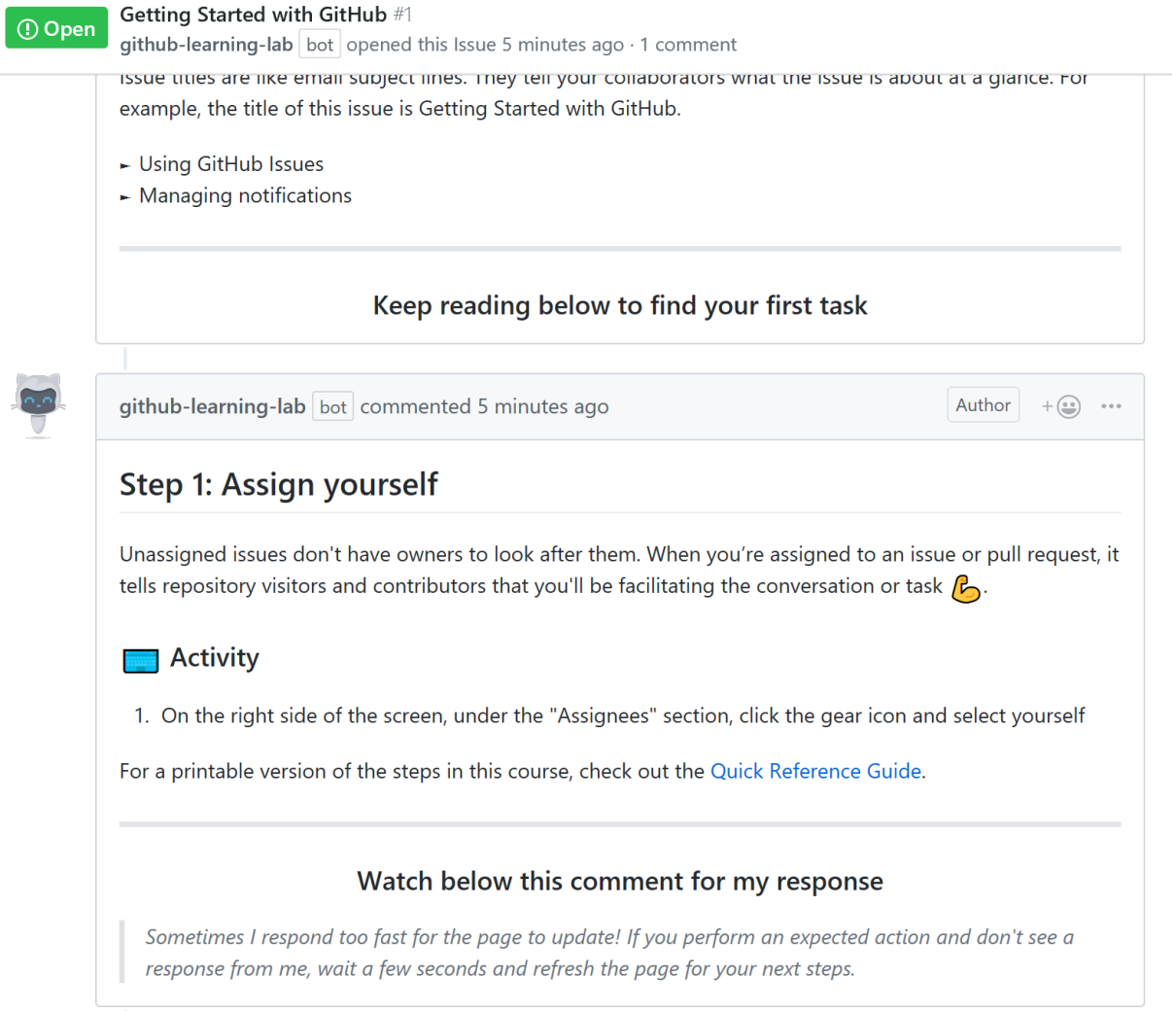
Uninstall

Start the course

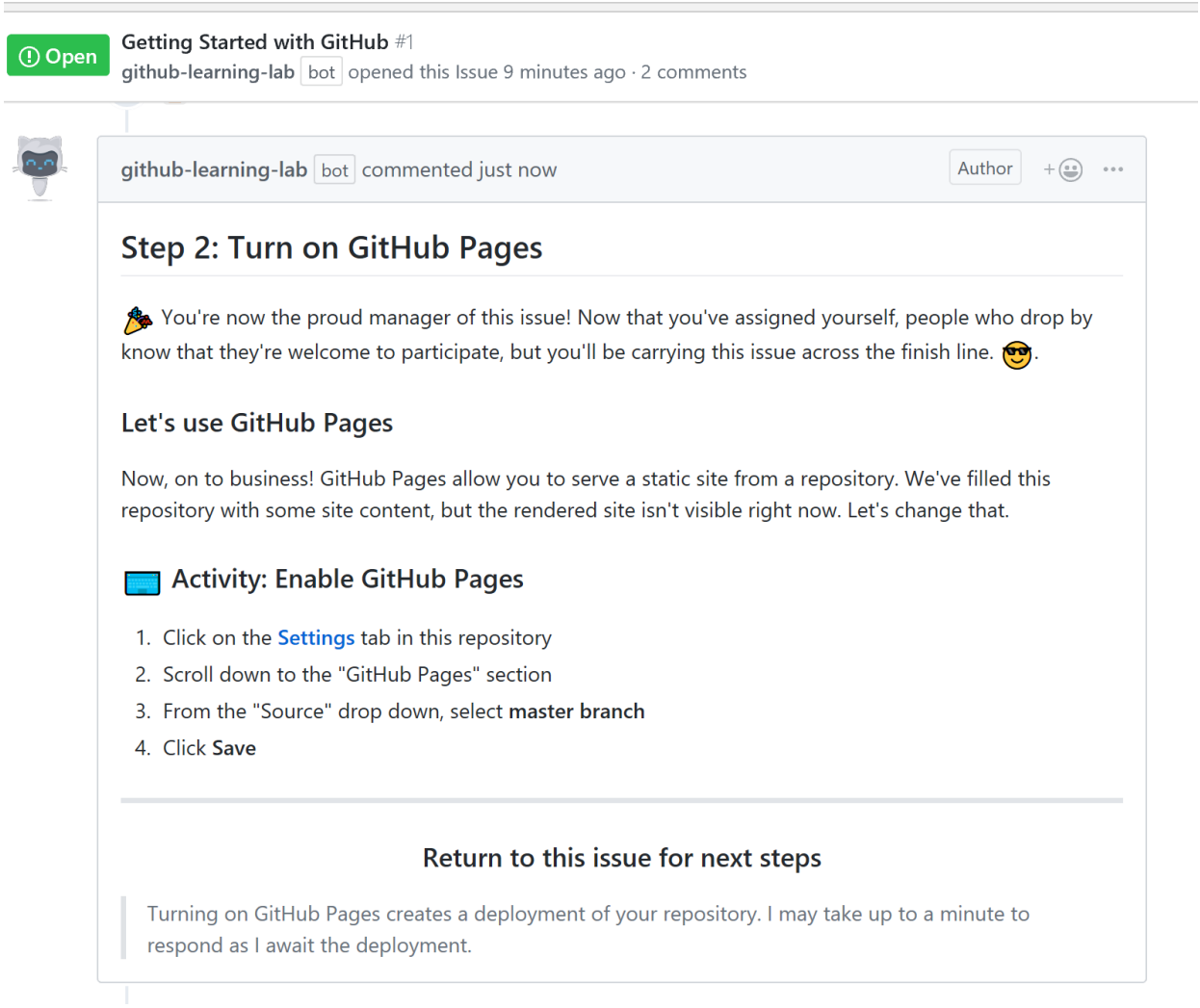
- Go to the link again to start the course: https://lab.github.com/githubtraining/introduction-to-github



Course Step 1: Assign yourself



Course Step 2: Turn on GitHub Pages



Open **Getting Started with GitHub** #1

github-learning-lab bot opened this Issue 9 minutes ago · 2 comments

github-learning-lab bot commented just now — Author

Step 2: Turn on GitHub Pages

You're now the proud manager of this issue! Now that you've assigned yourself, people who drop by know that they're welcome to participate, but you'll be carrying this issue across the finish line. .

Let's use GitHub Pages

Now, on to business! GitHub Pages allow you to serve a static site from a repository. We've filled this repository with some site content, but the rendered site isn't visible right now. Let's change that.

Activity: Enable GitHub Pages

1. Click on the Settings tab in this repository
2. Scroll down to the "GitHub Pages" section
3. From the "Source" drop down, select **master branch**
4. Click **Save**

Return to this issue for next steps

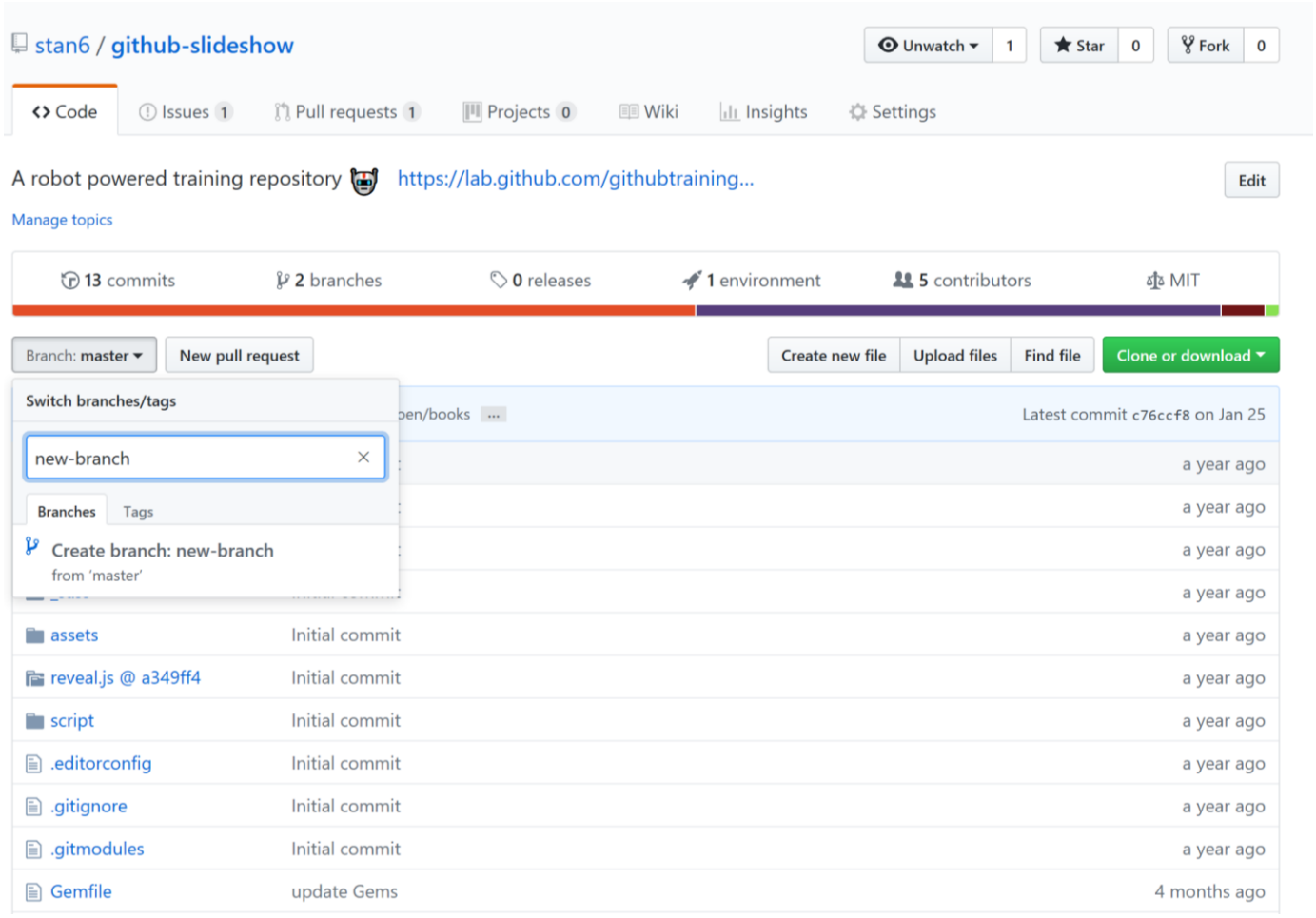
> Turning on GitHub Pages creates a deployment of your repository. I may take up to a minute to respond as I await the deployment.

Course Step 3: Close an issue

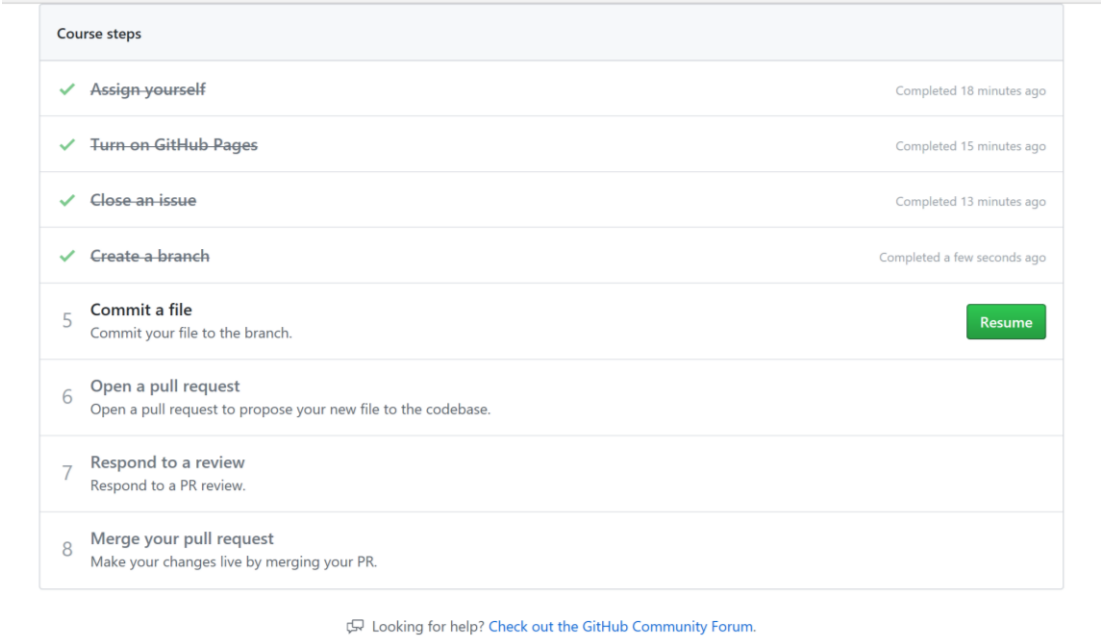
Course steps

- ✓ ~~Assign yourself~~ — Completed 4 minutes ago
- ✓ ~~Turn on GitHub Pages~~ — Completed a few seconds ago
- 3 **Close an issue** — Cease a conversation by closing an issue. [Resume]
- 4 **Create a branch** — Create a branch for introducing new changes.
- 5 **Commit a file** — Commit your file to the branch.
- 6 **Open a pull request** — Open a pull request to propose your new file to the codebase.
- 7 **Respond to a review** — Respond to a PR review.
- 8 **Merge your pull request** — Make your changes live by merging your PR.

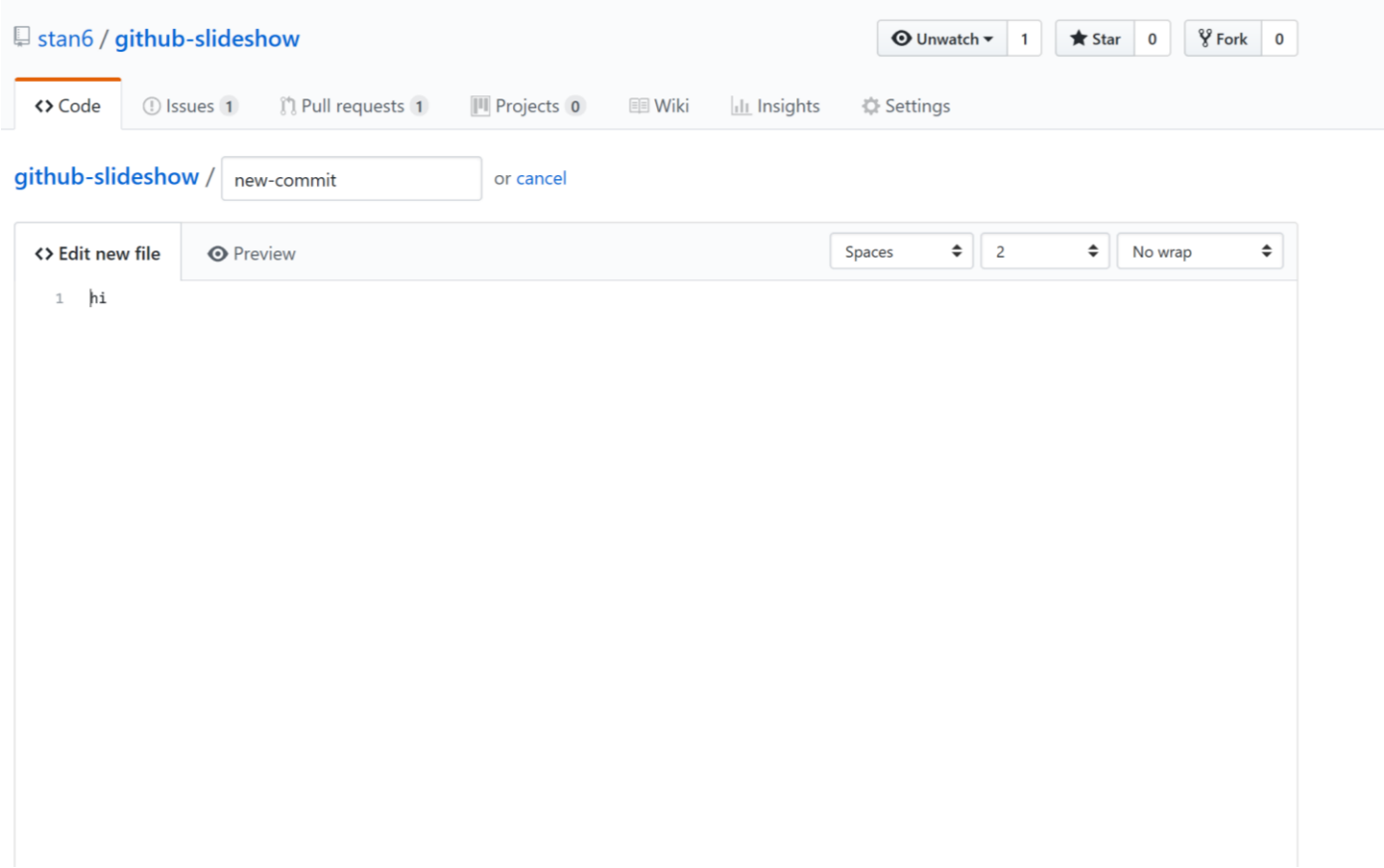
Course Step 4:
Create New Branch



Course Step 5: Commit a file



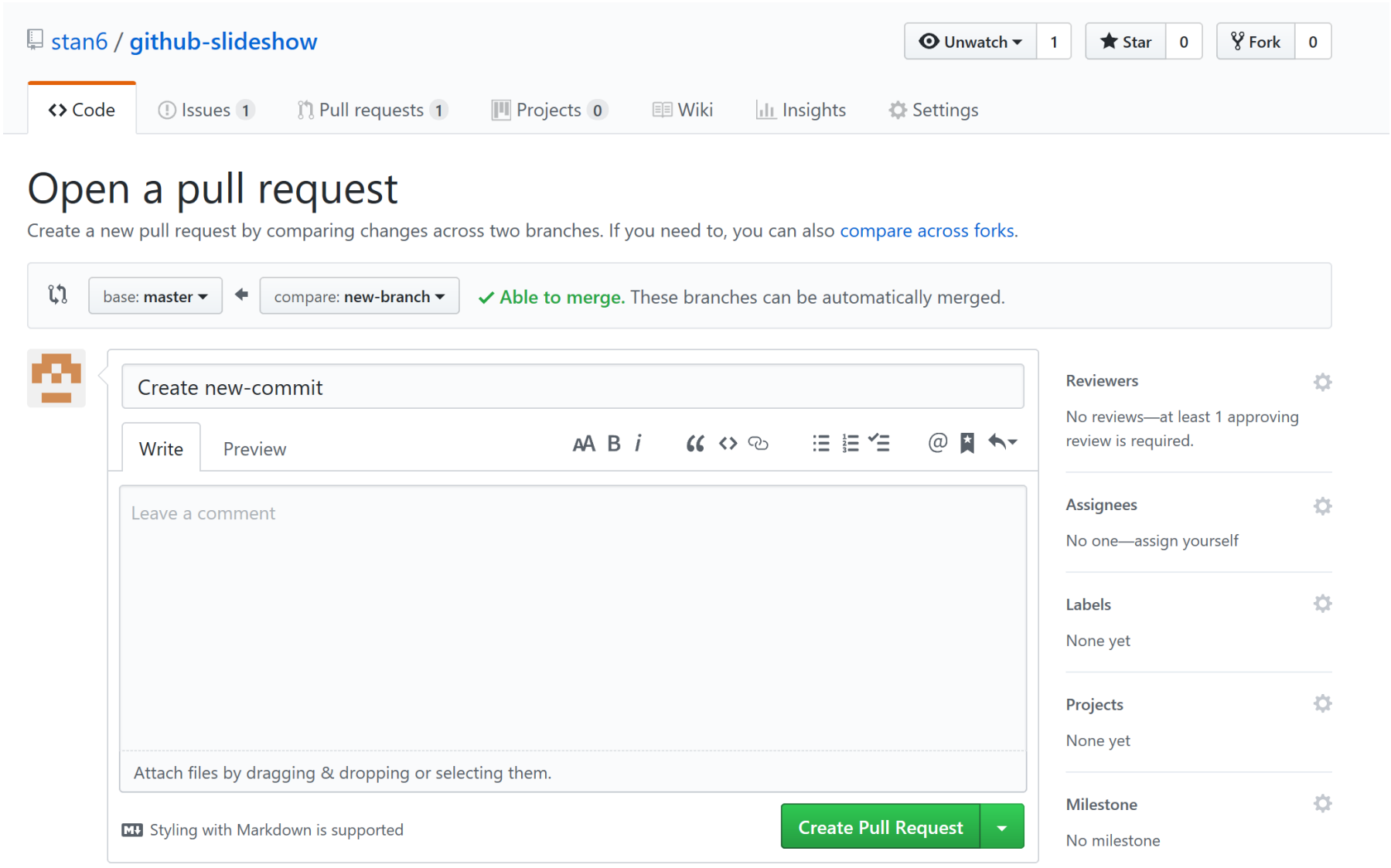
Commit a new file



Course Step 6: Open Pull Request

Course steps

✓	~~Assign yourself~~	Completed 23 minutes ago
✓	~~Turn on GitHub Pages~~	Completed 19 minutes ago
✓	~~Close an issue~~	Completed 17 minutes ago
✓	~~Create a branch~~	Completed 5 minutes ago
✓	~~Commit a file~~	Completed a few seconds ago
6	**Open a pull request** Open a pull request to propose your new file to the codebase.	Resume
7	**Respond to a review** Respond to a PR review.	
8	**Merge your pull request** Make your changes live by merging your PR.	


stan6 / github-slideshow
Unwatch 1
Star 0
Fork 0
Code
Issues 1
Pull requests 1
Projects 0
Wiki
Insights
Settings
Open a pull request
Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.
base: master
compare: new-branch
Able to merge. These branches can be automatically merged.
Create new-commit
Write
Preview
Leave a comment
Attach files by dragging & dropping or selecting them.
Styling with Markdown is supported
Create Pull Request
Reviewers
No reviews—at least 1 approving review is required.
Assignees
No one—assign yourself
Labels
None yet
Projects
None yet
Milestone
No milestone

Open **Create new-commit** #4

stan6 wants to merge 1 commit into `master` from `new-branch`

Create new-commit Verified 7596d80

github-learning-lab bot requested changes just now View changes

github-learning-lab bot left a comment

Good pull requests have a body description that tells other contributors about the change you're suggesting, so they understand the context.

Let's edit this pull request to add a body description.

Activity: Fixing your pull request

1. The first comment on your pull request will have the default text of **No description provided**. Click on the **...** icon located at the top right corner of the comment box, then click on **Edit** to make an edit
2. Add a description of the changes you've made in the comment box. Feel free to add a description of what you've accomplished so far. As a reminder, you have: created a branch, created a file and made a commit, and opened a pull request
3. Click the green **Update comment** button at the bottom right of the comment box when done

If you would like assistance troubleshooting the issue you are encountering, create a post on the GitHub Community board. You might also want to search for your issue to see if other people have resolved it in the past.

Watch below for my response

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Notifications

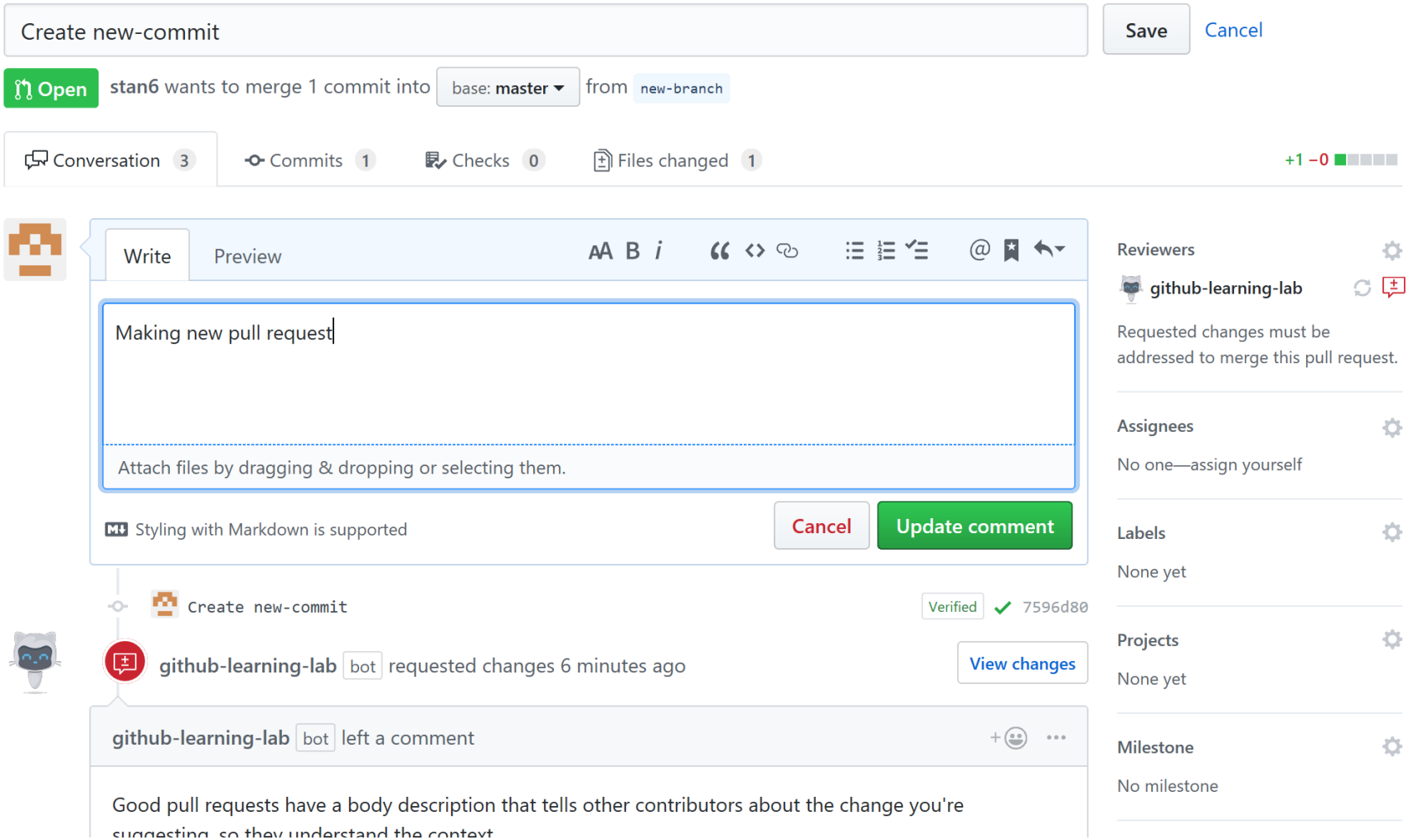
Unsubscribe

You're receiving notifications because you authored the thread.

1 participant

Lock conversation

Edit your pull request



Course Step 7: Respond to a review

Course steps

- ~~Assign yourself~~ — Completed 35 minutes ago
- ~~Turn on GitHub Pages~~ — Completed 31 minutes ago
- ~~Close an issue~~ — Completed 29 minutes ago
- ~~Create a branch~~ — Completed 17 minutes ago
- ~~Commit a file~~ — Completed 12 minutes ago
- ~~Open a pull request~~ — Completed a few seconds ago
- 7 **Respond to a review** — Respond to a PR review. — Resume
- 8 **Merge your pull request** — Make your changes live by merging your PR.

github-learning-lab bot reviewed an hour ago

View changes

github-learning-lab bot left a comment

Step 7: Respond to a review

Your pull request is looking great!

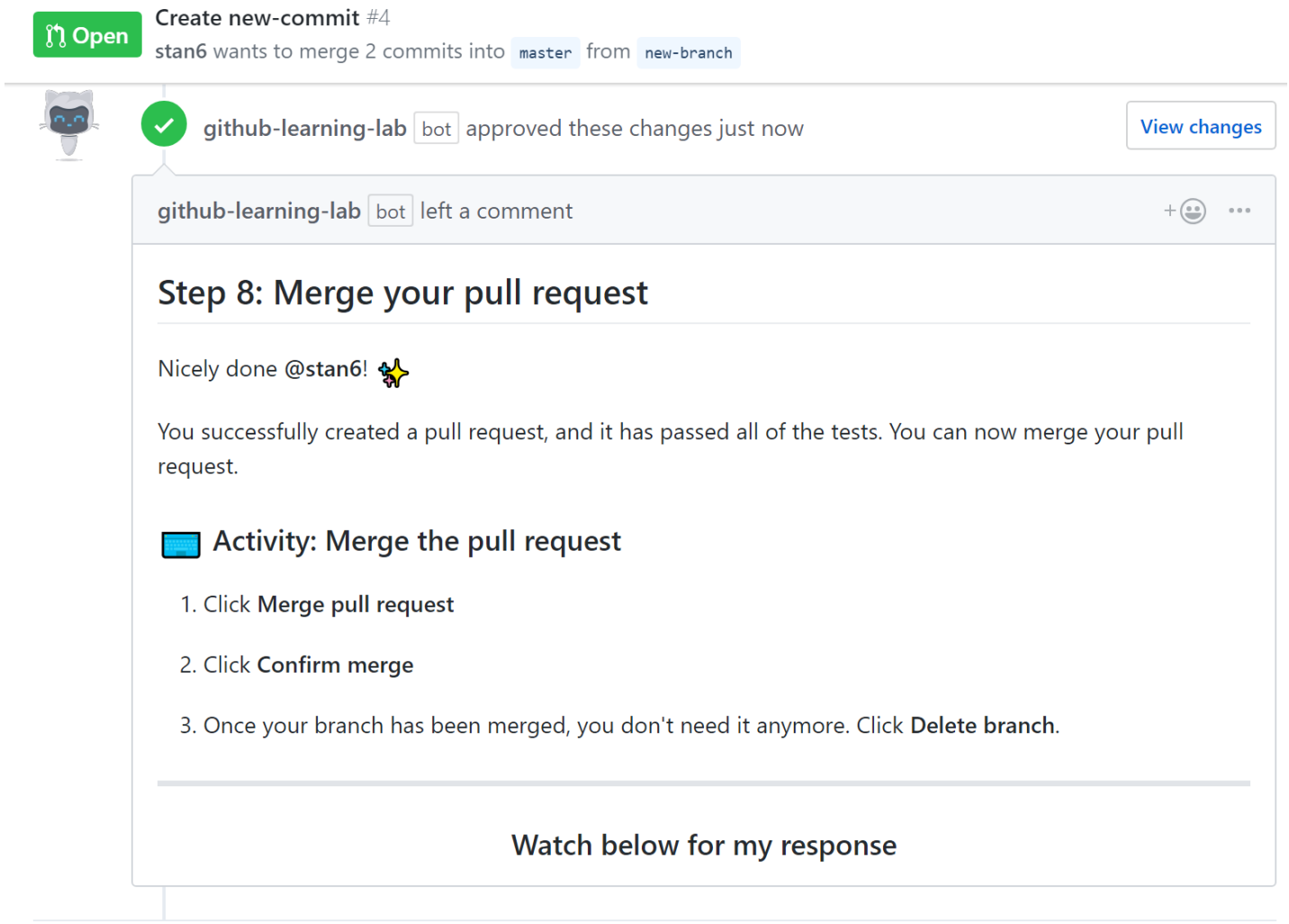
Let's add some content to your file. Replace line 5 of your file with a quotation or meme and witty caption. Remember: Markdown is supported.

Activity: Change your file

1. Click the "Files Changed" tab in this pull request
2. Click on the pencil icon found on the right side of the screen
3. Replace line 5 with something new
4. Scroll to the bottom and click **Commit Changes**

Watch below for my response

Course Step 8: Merge pull request



Go Back to the Link

- Go to the link: https://lab.github.com/githubtraining/introduction-to-github

Course Completed!

- Show this screen to the TA