

Lesson

Weekend

[Workshops \(/workshops\)](#)[/ Intro to Programming Workshop \(/workshops/intro-to-programming-workshop\)](#)[/ Publishing Your Site with GitHub Pages](#)

Text

This lesson is set up to take *one* person through the process of publishing a website to github. As we're working in pairs today, each person should go through this lesson for themselves. Set up an account for one of you, go through the whole lesson, then switch and do the whole process again with the next pair.

Our site is looking good! We have HTML content formatting and styled using both our own custom CSS, and the Bootstrap CSS framework. And it even includes an interactive jQuery form to update the site on the fly! Great work. Next up, let's actually *publish* our new site for the world to see. In this lesson we'll briefly explore tools called **Git** and **GitHub** in order to publish our page online using a special tool known as **GitHub pages**.

## Git and GitHub

Professional web and software developers have a wide range of tools at their disposal to help build, monitor, and collaborate on projects. One of the most common types of software is called **version control software**. The goal of good version control software is to take micro-snapshots, frequently, as a project is being built. This helps track little changes along the way, and makes it easier if you need to go back to an old version, see how something was done "back when it was working" or something like that. Good version control software also has many other uses. One is the ability to move projects around from machine to machine using the Terminal.

The most common version control software in use is called **Git**. It's easy to access via the command line or Terminal, and ties easily to one of the most common open-source code-sharing websites, **GitHub**. Our goal in this section is to help you get your code saved with Git, and hosted on GitHub so that you can view your code, or even visit your site from any computer.

Let's open the Terminal and enter a few commands. Git is complex software, that does a lot more than we need it for today. Stick to these commands, and you can't go wrong.

Make sure you're in the `My_Project` directory with `pwd` , then enter the following.

```
git init
```

```
git config user.name your_name (use quotes if you want to use a multiple word name)
```


```
git config user.email your_email
```

You're now set up as the author of this project.

Now we need to create a GitHub account for you. Good news, they're free, and are a staple of the developer's life. You can create an account at [github.com](http://github.com) (<http://github.com>). After creating the account you'll need to visit the email address you provided, and click the link in the confirmation email GitHub has sent you.

Pick a good user name! As a developer, it might end up on your resume! GitHub also functions as a social network for developers. You can connect with people, collaborate on projects, and even find cool new libraries and tools to use.

Once you successfully register for a GitHub account, you should see a welcome message like this:

 Search GitHub


Pull requestsIssuesGist


+

# Welcome to GitHub

You've taken your first step into a larger world, @EpicodusWorkshopStudent.

✓ Completed  
Set up a personal account

 Step 2:  
Choose your plan

 Step 3:  
Tailor your experience

## Choose your personal plan

☒ Unlimited public repositories for free.

☐ Unlimited private repositories for \$7/month.

Don't worry, you can cancel or upgrade at any time.

☐ **Help me set up an organization next**  
Organizations are separate from personal accounts and are best suited for businesses who need to manage permissions for many employees.  
[Learn more about organizations.](#)

Continue

**Both plans include:**

- ✓ Collaborative code review
- ✓ Issue tracking
- ✓ Open source community
- ✓ Unlimited public repositories
- ✓ Join any organization

Now that you have an account, let's create a **repository** on GitHub. A **repository** is like creating a place to store and single project on your GitHub account. To create a new repository (almost always called a **repo**) select the + option in the upper-right corner of the GitHub welcome page. Then, select *New repository* from the dropdown menu, as seen in the image below:

The screenshot shows the GitHub 'Welcome to GitHub' page. At the top, there's a dark header with the GitHub logo, a search bar, and links for 'Pull requests', 'Issues', and 'Gist'. A red box highlights a '+' button in the top right corner, which has opened a dropdown menu with options: 'New repository', 'Import repository', 'New gist', and 'New organization'. The main content area has a large 'Welcome to GitHub' heading and a sub-header 'You've taken your first step into a larger world, @EpicodusWorkshopStu'. Below this is a progress bar with three steps: 'Step 1: Completed Set up a personal account', 'Step 2: Choose your plan' (the current step), and 'Step 3: Tailor your experience'. Under 'Choose your personal plan', there are two radio button options: 'Unlimited public repositories for free.' (selected) and 'Unlimited private repositories for \$7/month.'. Below these options is a note: 'Don't worry, you can cancel or upgrade at any time.' and a checkbox for 'Help me set up an organization next' with a brief explanation and a link to 'Learn more about organizations.'. A green 'Continue' button is at the bottom left.

Search GitHub Pull requests Issues Gist

New repository  
Import repository  
New gist  
New organization

# Welcome to GitHub

You've taken your first step into a larger world, @EpicodusWorkshopStu

Completed Set up a personal account

Step 2: Choose your plan

Step 3: Tailor your experience

## Choose your personal plan

☒ Unlimited public repositories for free.

☐ Unlimited private repositories for \$7/month.

Don't worry, you can cancel or upgrade at any time.


☐ Help me set up an organization next  
Organizations are separate from personal accounts and are best suited for businesses who need to manage permissions for many employees.  
[Learn more about organizations.](#)

Continue


Both plans include:

- ✓ Collaborative code review
- ✓ Issue tracking
- ✓ Open source community
- ✓ Unlimited public repositories
- ✓ Join any organization

This should take you to a screen that looks like this:

 Search GitHub

Pull requestsIssuesGist



---


## Create a new repository

A repository contains all the files for your project, including the revision history.

---

Owner


Repository name


 EpicodusWorkshopStudent /

Great repository names are short and memorable. Need inspiration? How about **redesigned-octo-memory**.

Description (optional)

---


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

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

---

☐ **Initialize this repository with a README**  
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

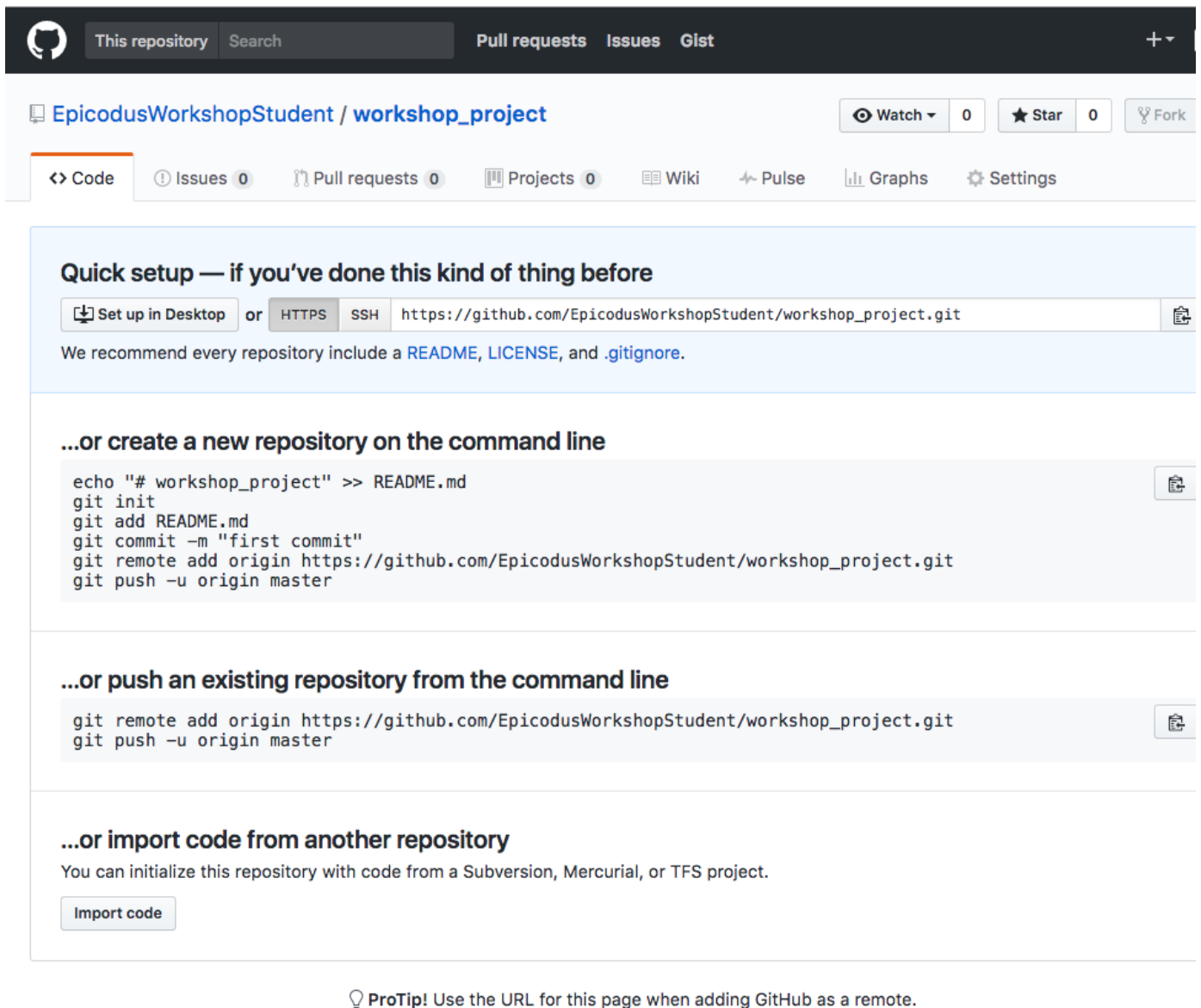
Add .gitignore: **None**

Add a license: **None** 

---

Create repository

When it asks for the name of the repo, use something easy to identify, like *workshop\_project*. You don't need to alter any of the other options here, besides your *Repository name*. After entering a name, click the *Create repository* button. This should take you to a page that looks like this:



The screenshot shows the GitHub interface for a repository named 'workshop\_project' by user 'EpicodusWorkshopStudent'. The top navigation bar includes links for 'This repository', 'Search', 'Pull requests', 'Issues', and 'Gist'. Below the repository name, there are buttons for 'Watch' (0), 'Star' (0), and 'Fork'. A secondary navigation bar contains links for 'Code', 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Pulse', 'Graphs', and 'Settings'. The main content area features a 'Quick setup' section with instructions for setting up the repository on a desktop or via HTTPS/SSH. It provides the repository URL: `https://github.com/EpicodusWorkshopStudent/workshop_project.git`. Below this, there are two sections for creating a new repository or pushing an existing one from the command line, each with a list of git commands. The first section includes commands to create a README, initialize git, add the README, commit, add the remote origin, and push. The second section shows commands to add the remote origin and push. A third section, '...or import code from another repository', mentions that code from Subversion, Mercurial, or TFS can be imported, with an 'Import code' button. A 'ProTip!' note at the bottom advises using the URL for adding GitHub as a remote.

**Quick setup — if you've done this kind of thing before**

Set up in Desktop or **HTTPS** **SSH** `https://github.com/EpicodusWorkshopStudent/workshop_project.git`

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

**...or create a new repository on the command line**

```
echo "# workshop_project" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/EpicodusWorkshopStudent/workshop_project.git
git push -u origin master
```

**...or push an existing repository from the command line**

```
git remote add origin https://github.com/EpicodusWorkshopStudent/workshop_project.git
git push -u origin master
```

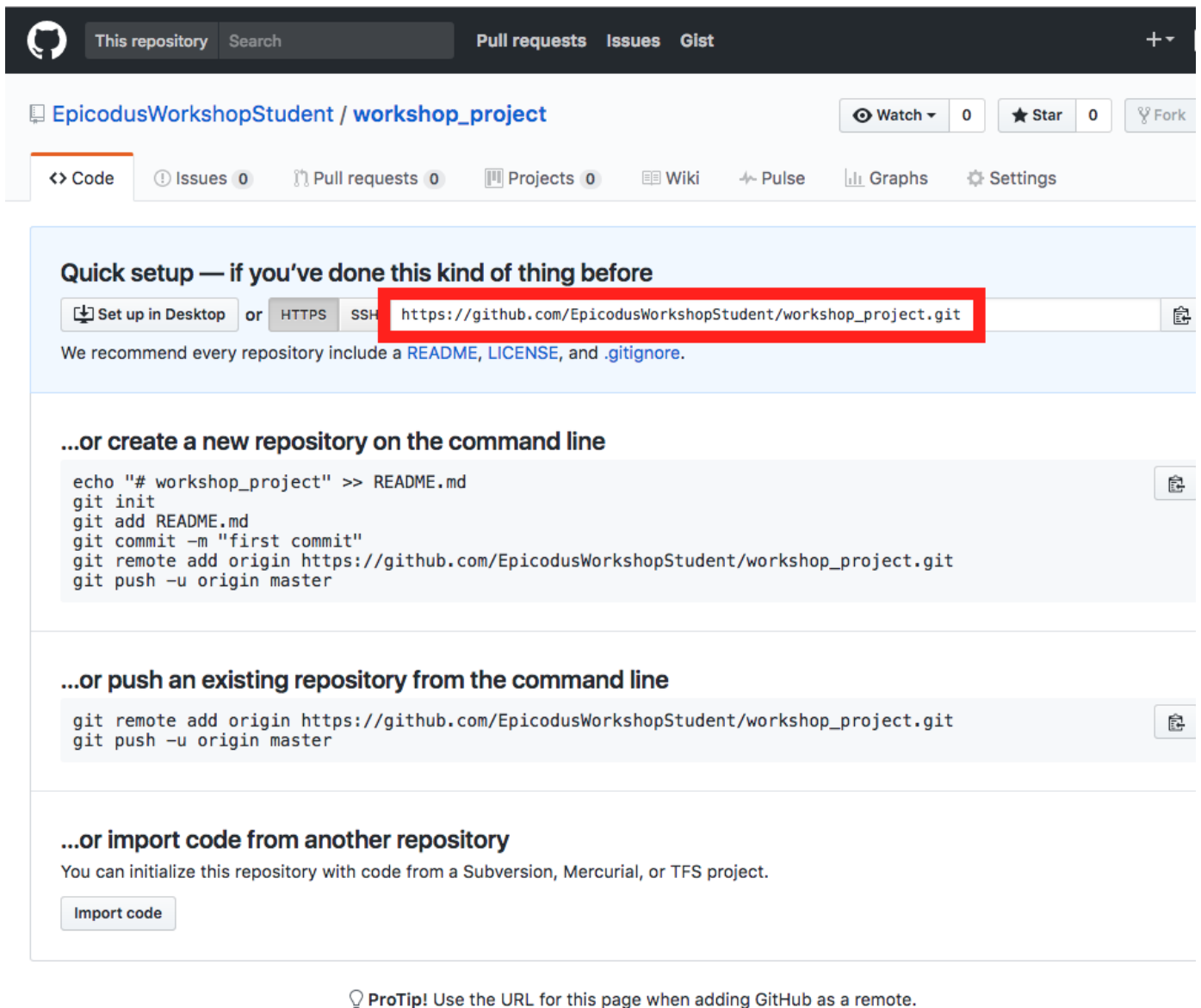
**...or import code from another repository**

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

[Import code](#)

**ProTip!** Use the URL for this page when adding GitHub as a remote.

Now that we have a GitHub account we need to link the copy of our project saved to our computer to our new repo we've just created for it on GitHub. Copy that new URL GitHub has given you, and move back to the Terminal.



The screenshot shows the GitHub interface for a repository named 'workshop\_project' by 'EpicodusWorkshopStudent'. The repository is empty, with 0 stars and 0 forks. The 'Code' tab is selected, showing options to set up the repository on a desktop or via HTTPS. The HTTPS URL is highlighted in a red box: `https://github.com/EpicodusWorkshopStudent/workshop_project.git`. Below this, there are instructions for creating a new repository or pushing an existing one from the command line. The command line instructions for creating a new repository are:

```
echo "# workshop_project" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/EpicodusWorkshopStudent/workshop_project.git
git push -u origin master
```

The command line instructions for pushing an existing repository are:

```
git remote add origin https://github.com/EpicodusWorkshopStudent/workshop_project.git
git push -u origin master
```

There is also an option to import code from another repository (Subversion, Mercurial, or TFS) and a 'ProTip!' to use the URL for this page when adding GitHub as a remote.

Enter the following line into the terminal.

```
git remote add github your_URL
```

This is setting up something called a **remote** within GitHub. A remote is simply an address of a repo.

Next, enter the following commands.

```
git add .
```

```
git commit -m "add workshop project to git"
```

This will tell Git to collect all the work we've done today, and add it to a package, called a **commit**. That commit is the thing we're going to send to our repo on GitHub.

Finally, you can enter the following:

```
git push github master
```

Git will ask you to enter your username and password for GitHub, just to verify that you have the right to store your project there.

Once you've entered your password, Git will send your the local version of your project from your computer to your repo you've just created on GitHub. You can view the files online by refreshing your repo.

The screenshot shows the GitHub interface for a repository named 'workshop\_project' by user 'EpicodusWorkshopStudent'. The repository has 1 commit, 1 branch, 0 releases, and 1 contributor. The 'master' branch is selected. Below the repository header, there is a list of files: 'css', 'js', and 'index.html', all with the commit message 'add workshop project to git.' and a timestamp of 'a minute ago'. At the bottom, there is a prompt to 'Add a README'.

GitHub repository page for **EpicodusWorkshopStudent / workshop\_project**. The page shows the repository details, including the commit history and the files in the repository.

The repository has 1 commit, 1 branch, 0 releases, and 1 contributor.

The commit history shows the following files:

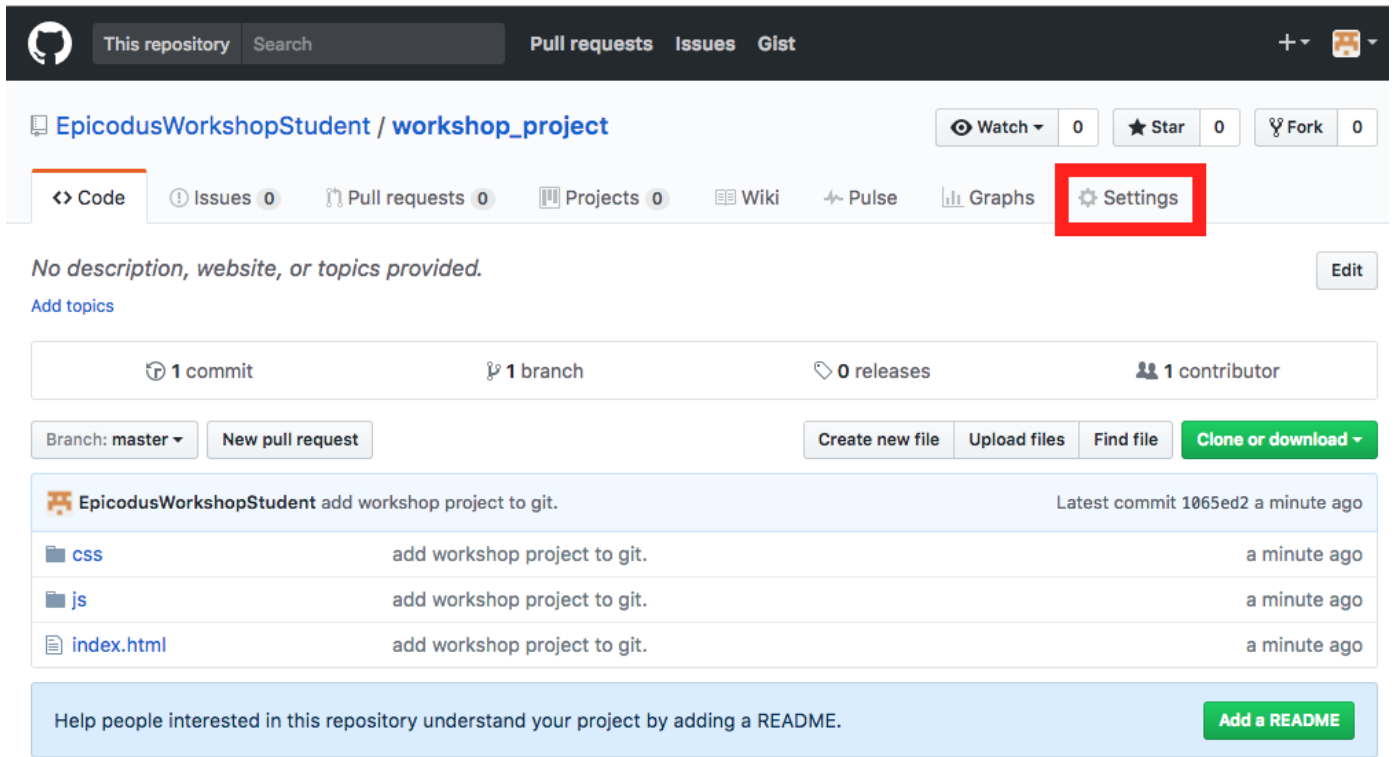
- css: add workshop project to git. (a minute ago)
- js: add workshop project to git. (a minute ago)
- index.html: add workshop project to git. (a minute ago)

At the bottom, there is a prompt to **Add a README**.

Let's take one last step. Let's let GitHub host our project, live, online.

Within your repo, click *Settings* at the top.





5/6/2017 Publishing Your Site with GitHub Pages

GitHub repository: EpicodusWorkshopStudent / workshop\_project

Buttons: Watch (0), Star (0), Fork (0)

Navigation: Code, Issues (0), Pull requests (0), Projects (0), Wiki, Pulse, Graphs, **Settings**

No description, website, or topics provided. [Edit](#)

Add topics

1 commit, 1 branch, 0 releases, 1 contributor

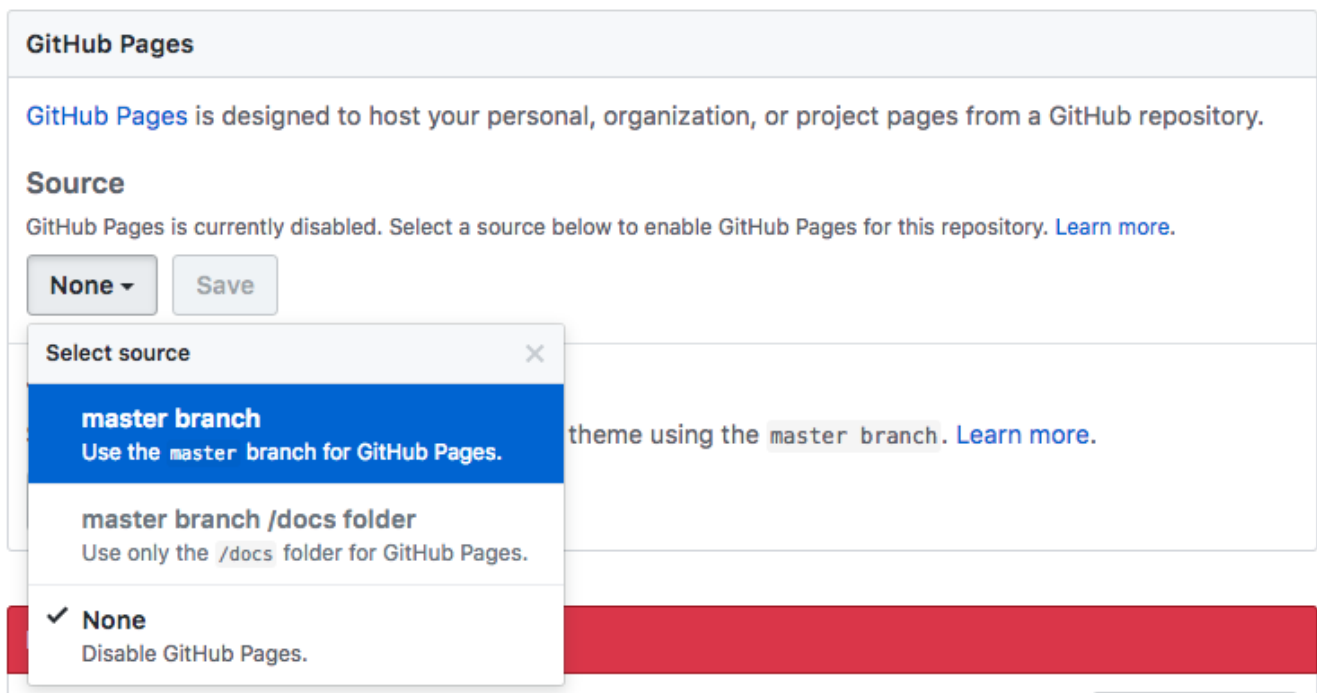
Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

Commit history:

File	Commit Message	Time
css	add workshop project to git.	a minute ago
js	add workshop project to git.	a minute ago
index.html	add workshop project to git.	a minute ago

Help people interested in this repository understand your project by adding a README. [Add a README](#)

In the settings area, scroll down until you see a box labelled GitHub Pages. Within this box, you'll see a dropdown reading *None*. From that, select *master branch*. Then click Save.



**GitHub Pages**

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

**Source**

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. [Learn more.](#)

None | Save

Select source

- master branch**  
Use the master branch for GitHub Pages.
- master branch /docs folder  
Use only the /docs folder for GitHub Pages.
- ✓ None  
Disable GitHub Pages.

theme using the master branch. [Learn more.](#)

It may take a minute to compile, but you can now scroll down to see the URL of your new hosted site! Try it out!

### GitHub Pages

Your site is ready to be published at [https://epicodusworkshopstudent.github.io/workshop\\_project/](https://epicodusworkshopstudent.github.io/workshop_project/).

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

### Source

Your GitHub Pages site is currently being built from the `master` branch. [Learn more.](#)

master branch ▼

Save

© 2017 Epicodus (<http://www.epicodus.com/>), Inc.