How to use Git and the Terminal

Ebonie Gadsone

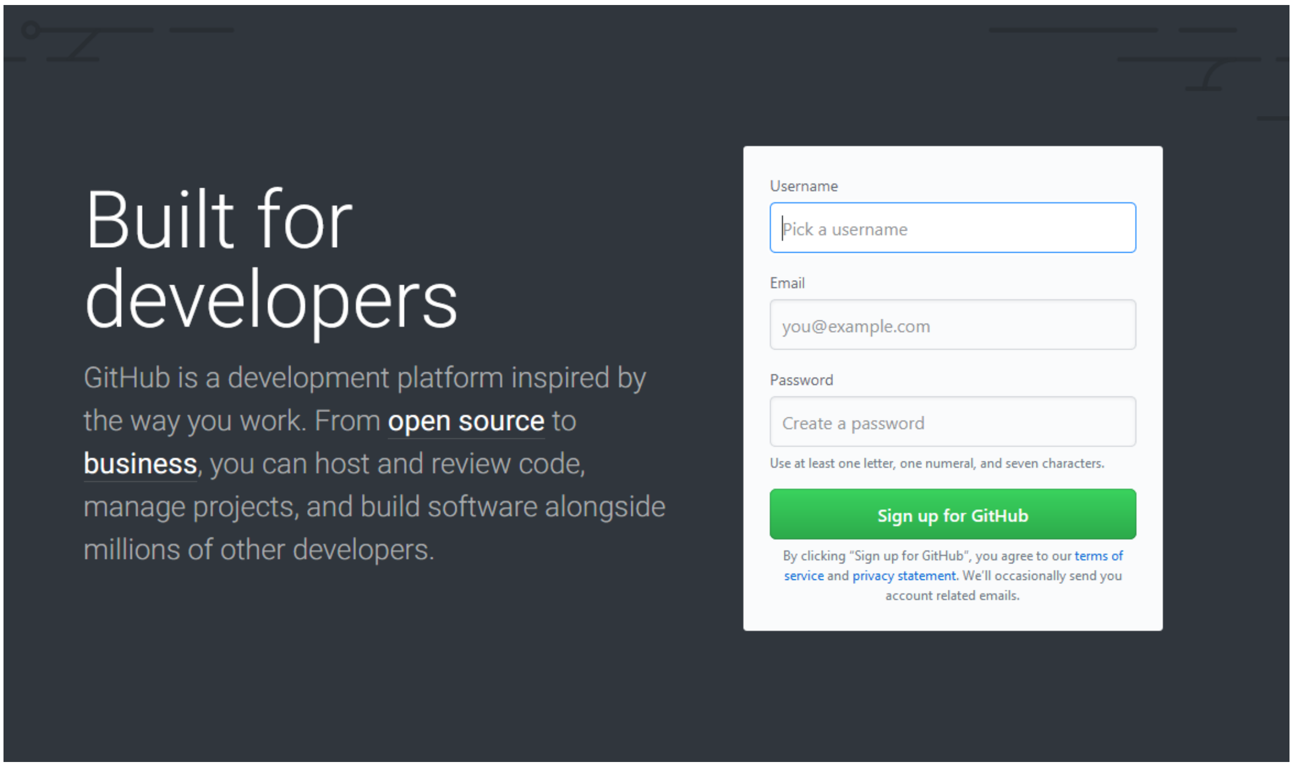
12 August 2018

Table of Contents

1. Getting Started

1.1. Github

1.1.1. Creating a Github account



Built for developers

GitHub is a development platform inspired by the way you work. From **open source** to **business**, you can host and review code, manage projects, and build software alongside millions of other developers.

Username

Pick a username

Email

you@example.com

Password

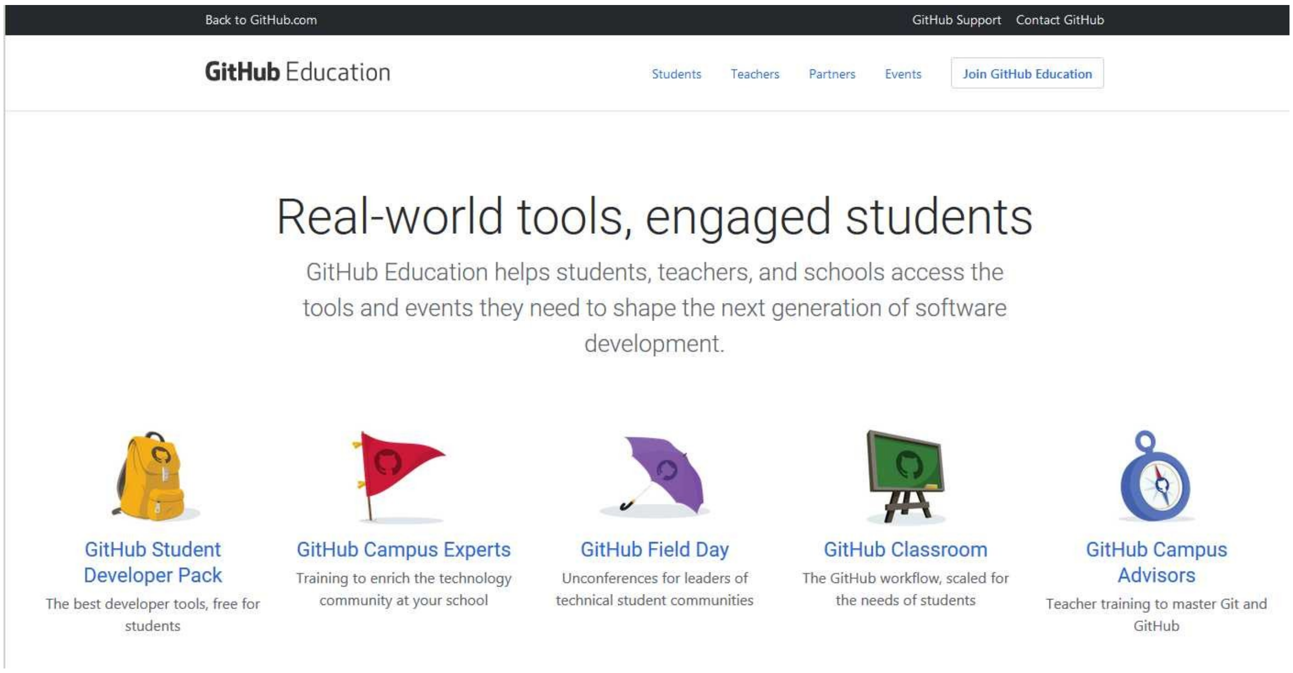
Create a password

Use at least one letter, one numeral, and seven characters.

By clicking "Sign up for GitHub", you agree to our terms of service and privacy statement. We'll occasionally send you account related emails.

Figure 1.1.1: In order to have access to an unlimited amount of private repositories, sign up without your MCs student email address (MYMCID@montgomerycollege.edu).

1.1.2. Request a discount



Back to GitHub.com GitHub Support Contact GitHub

GitHub Education Students Teachers Partners Events Join GitHub Education

Real-world tools, engaged students

GitHub Education helps students, teachers, and schools access the tools and events they need to shape the next generation of software development.

GitHub Student Developer Pack
The best developer tools, free for students

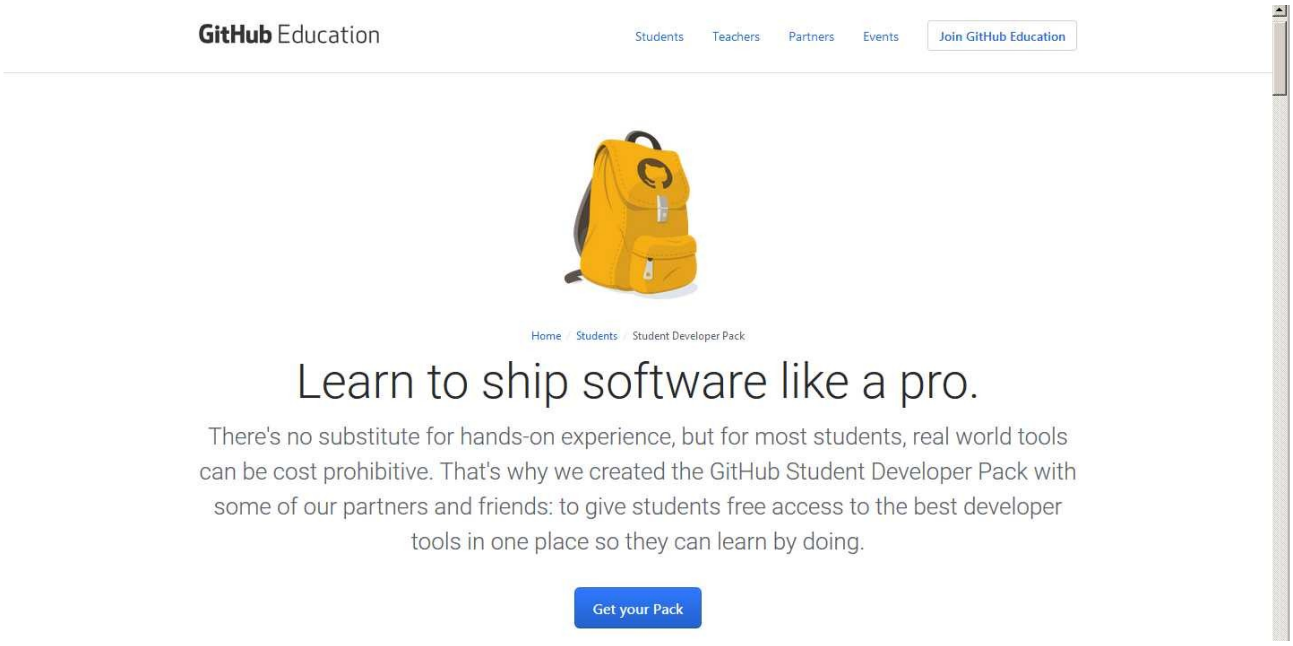
GitHub Campus Experts
Training to enrich the technology community at your school

GitHub Field Day
Unconferences for leaders of technical student communities

GitHub Classroom
The GitHub workflow, scaled for the needs of students

GitHub Campus Advisors
Teacher training to master Git and GitHub

Figure 1.1.2: After signing up for github go to https://education.github.com to request a free 2 year discount. Select "GitHub Student Developer Pack"



GitHub Education Students Teachers Partners Events Join GitHub Education

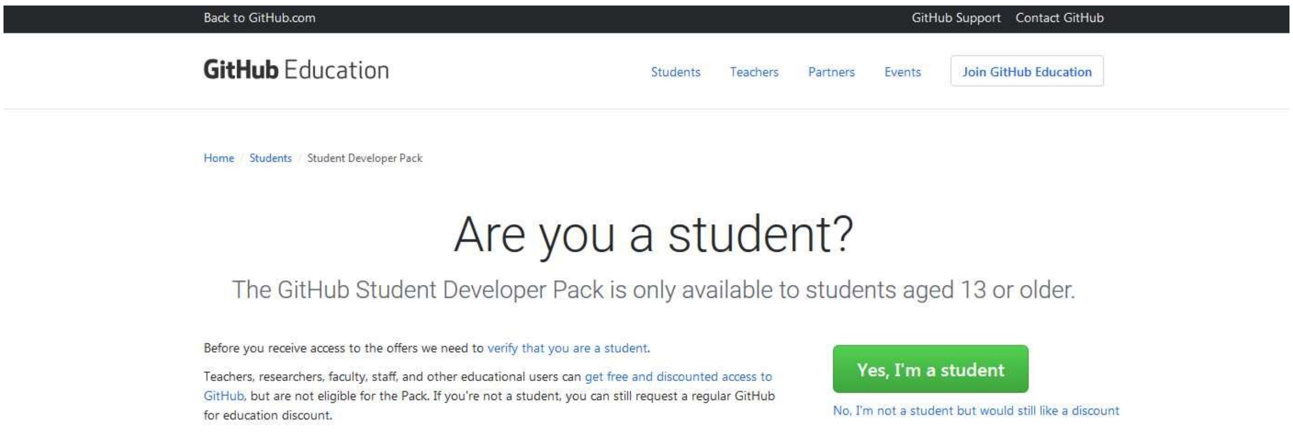
Home / Students / Student Developer Pack

Learn to ship software like a pro.

There's no substitute for hands-on experience, but for most students, real world tools can be cost prohibitive. That's why we created the GitHub Student Developer Pack with some of our partners and friends: to give students free access to the best developer tools in one place so they can learn by doing.

Get your Pack

Figure 1.1.3: Choose: "Get your pack"



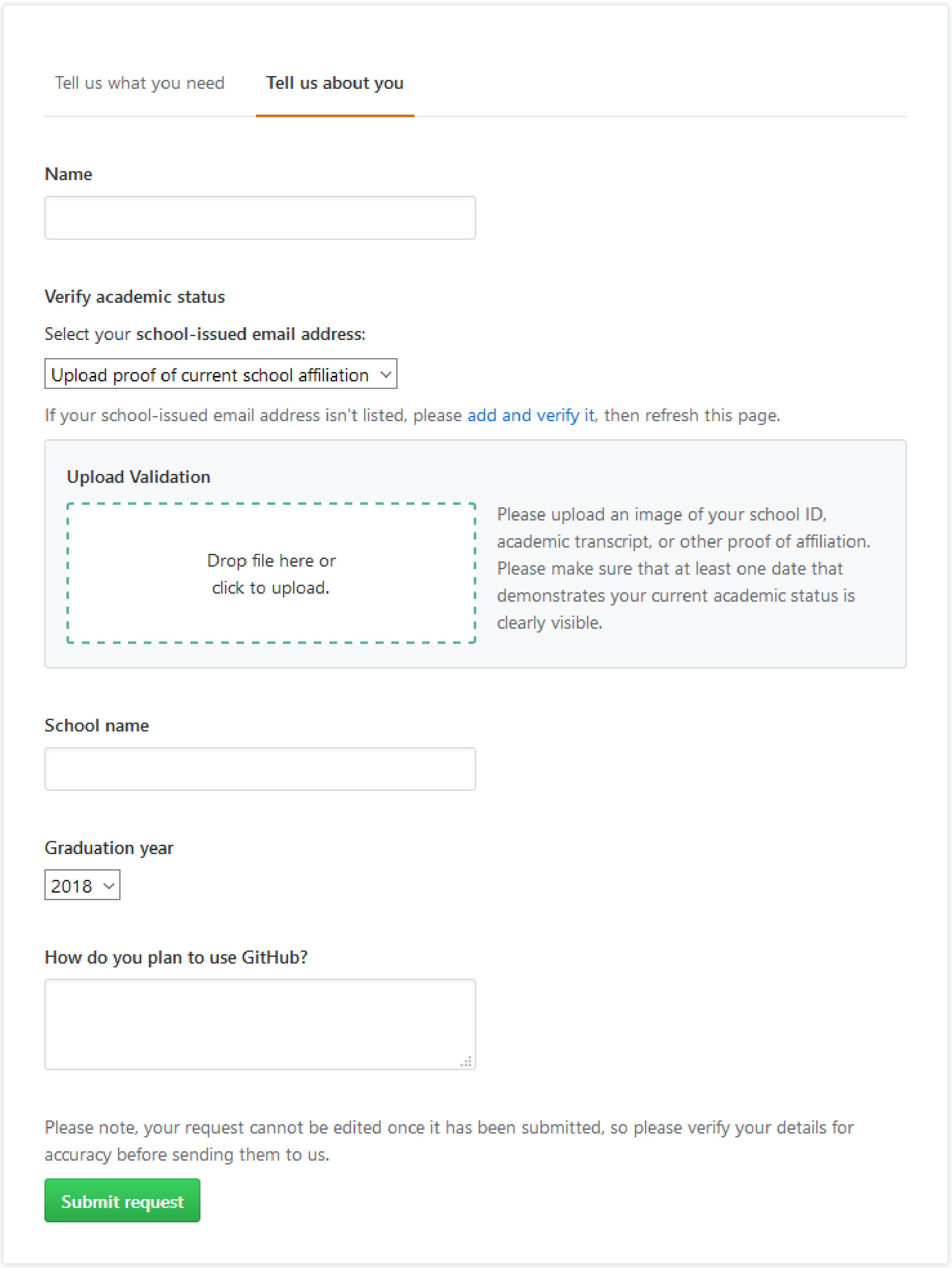
Are you a student?

The GitHub Student Developer Pack is only available to students aged 13 or older.

Before you receive access to the offers we need to verify that you are a student.

Teachers, researchers, faculty, staff, and other educational users can get free and discounted access to GitHub, but are not eligible for the Pack. If you're not a student, you can still request a regular GitHub for education discount.

Figure 1.1.4: Select: "Yes, I'm a student"


Tell us what you need | Tell us about you

Name

Verify academic status

Select your school-issued email address:

Upload proof of current school affiliation

If your school-issued email address isn't listed, please add and verify it, then refresh this page.

Upload Validation

Drop file here or click to upload.

Please upload an image of your school ID, academic transcript, or other proof of affiliation. Please make sure that at least one date that demonstrates your current academic status is clearly visible.

School name

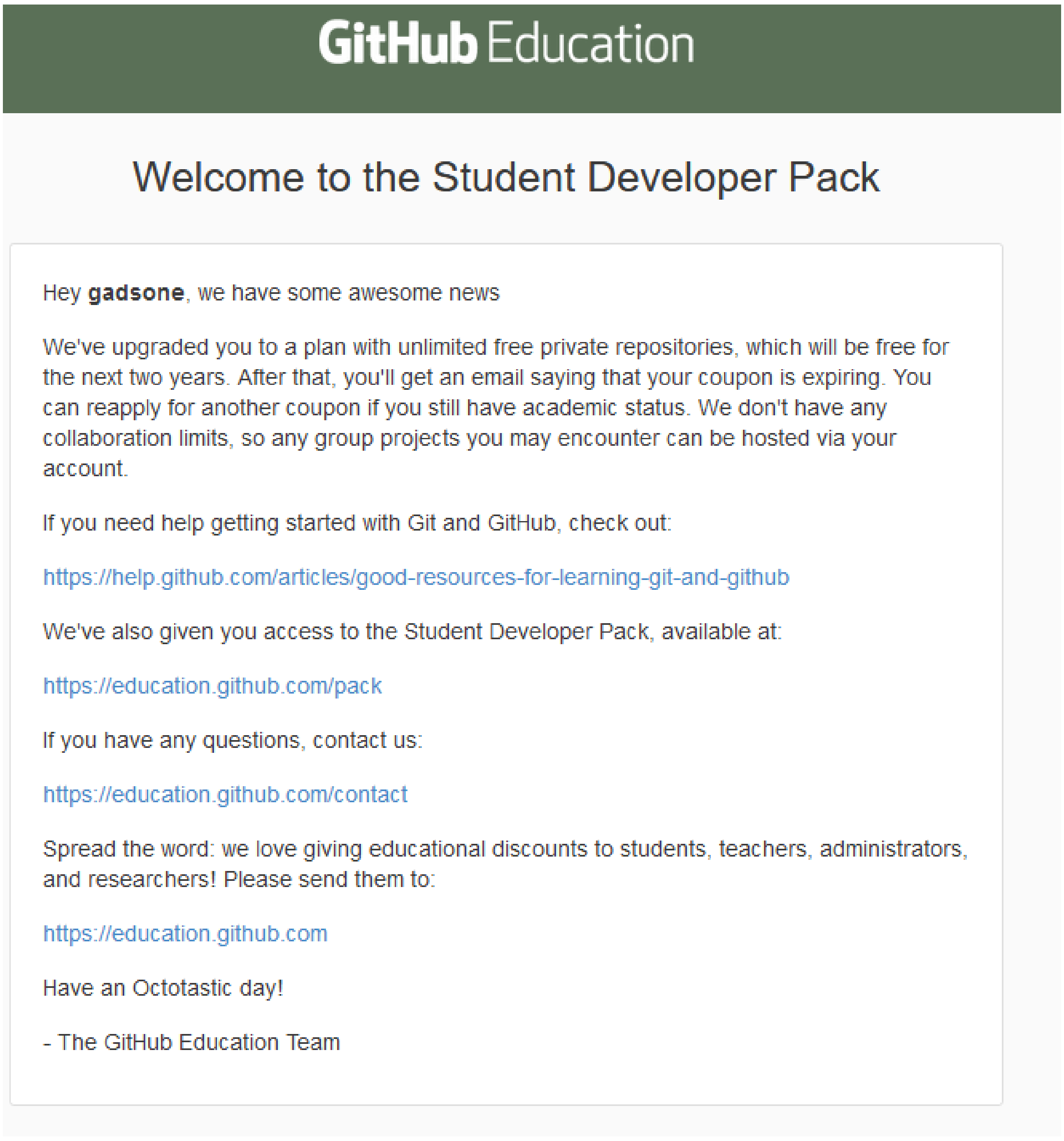
Graduation year

2018

How do you plan to use GitHub?

Please note, your request cannot be edited once it has been submitted, so please verify your details for accuracy before sending them to us.

Figure 1.1.5: Fill out the information above and upload a photo of your Montgomery College ID. Then, submit your request.



GitHub Education

Welcome to the Student Developer Pack

Hey **gadsone**, we have some awesome news

We've upgraded you to a plan with unlimited free private repositories, which will be free for the next two years. After that, you'll get an email saying that your coupon is expiring. You can reapply for another coupon if you still have academic status. We don't have any collaboration limits, so any group projects you may encounter can be hosted via your account.

If you need help getting started with Git and GitHub, check out:

https://help.github.com/articles/good-resources-for-learning-git-and-github

We've also given you access to the Student Developer Pack, available at:

https://education.github.com/pack

If you have any questions, contact us:

https://education.github.com/contact

Spread the word: we love giving educational discounts to students, teachers, administrators, and researchers! Please send them to:

https://education.github.com

Have an Octotastic day!

- The GitHub Education Team

Figure 1.1.6: Check college email to see when you have leveled up to student pack. You should receive a similar email to the one above typically in minutes or 3 to 5 business days. After submitting your request go to the billings tab on github and you should see your coupon.

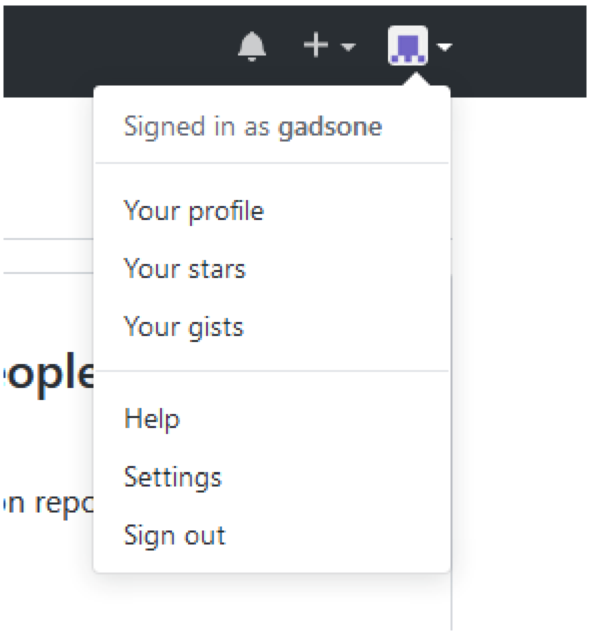
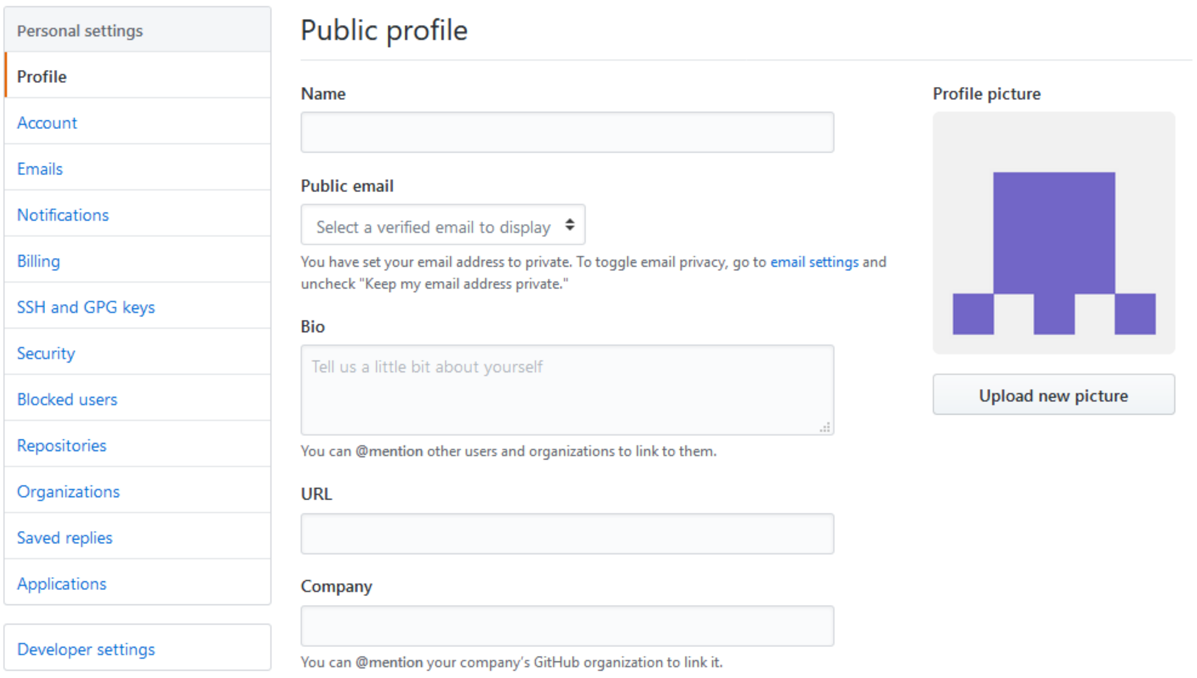


Figure 1.1.7: You can view your coupon by clicking on the down arrow next to your profile pic and selecting the "Settings" option.



Public profile

Name

Public email

Select a verified email to display

You have set your email address to private. To toggle email privacy, go to email settings and uncheck "Keep my email address private."

Bio

Tell us a little bit about yourself

You can @mention other users and organizations to link to them.

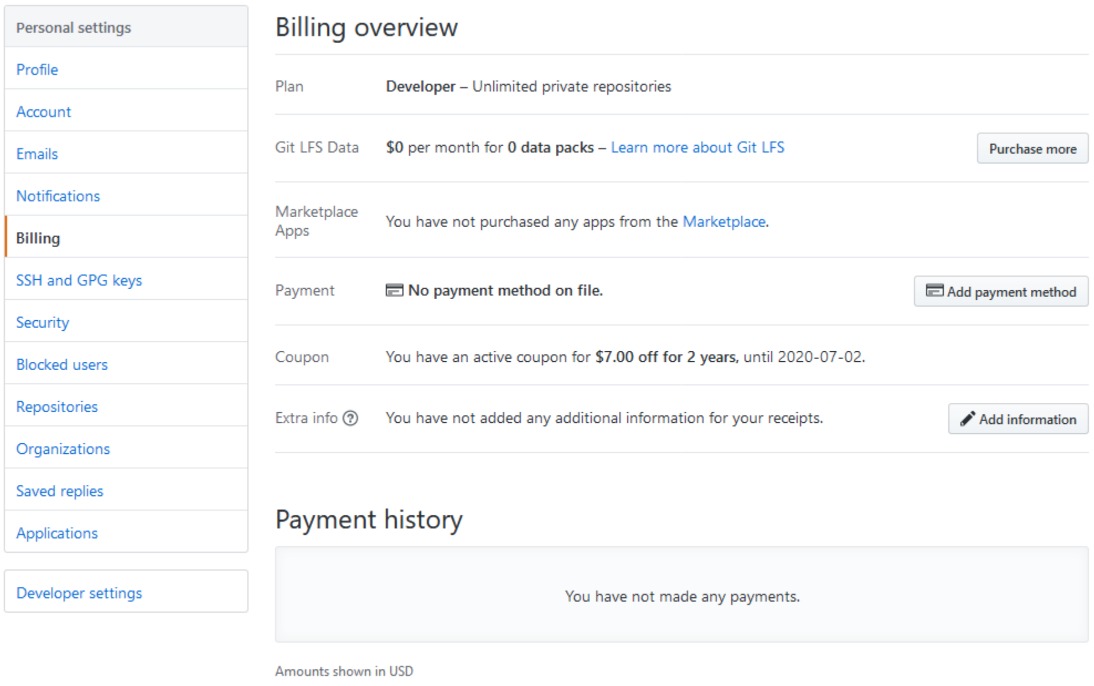
URL

Company

You can @mention your company's GitHub organization to link it.

Profile picture

Figure 1.1.8: Next, Select the "Billing" option



Billing overview

Plan	**Developer** – Unlimited private repositories	
Git LFS Data	**$0** per month for **0 data packs** – Learn more about Git LFS	Purchase more
Marketplace Apps	You have not purchased any apps from the Marketplace.	
Payment	**No payment method on file.**	Add payment method
Coupon	You have an active coupon for **$7.00 off for 2 years**, until 2020-07-02.	
Extra info	You have not added any additional information for your receipts.	Add information

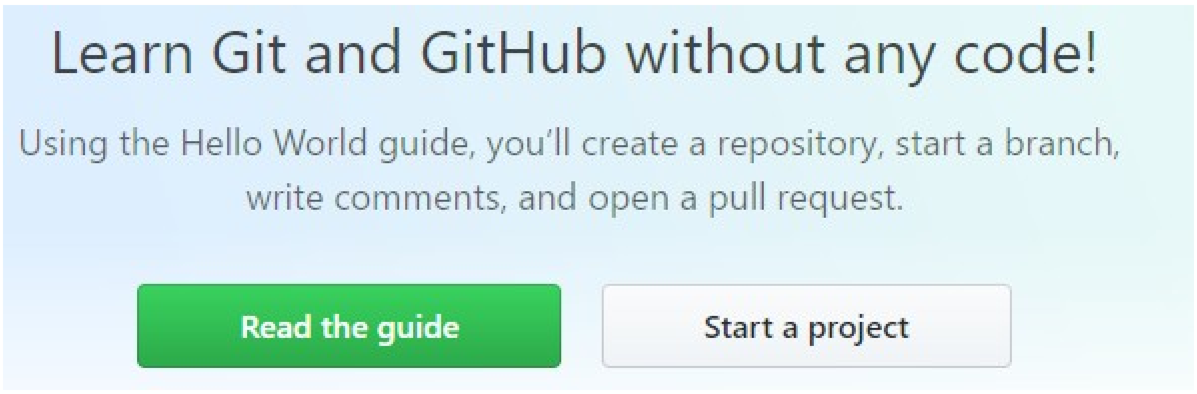
Payment history

You have not made any payments.

Amounts shown in USD

Figure 1.1.9: After, check to see if your plan includes unlimited private repositories and that you have an active coupon for $7.00.

1.1.3. Creating a Project in Github

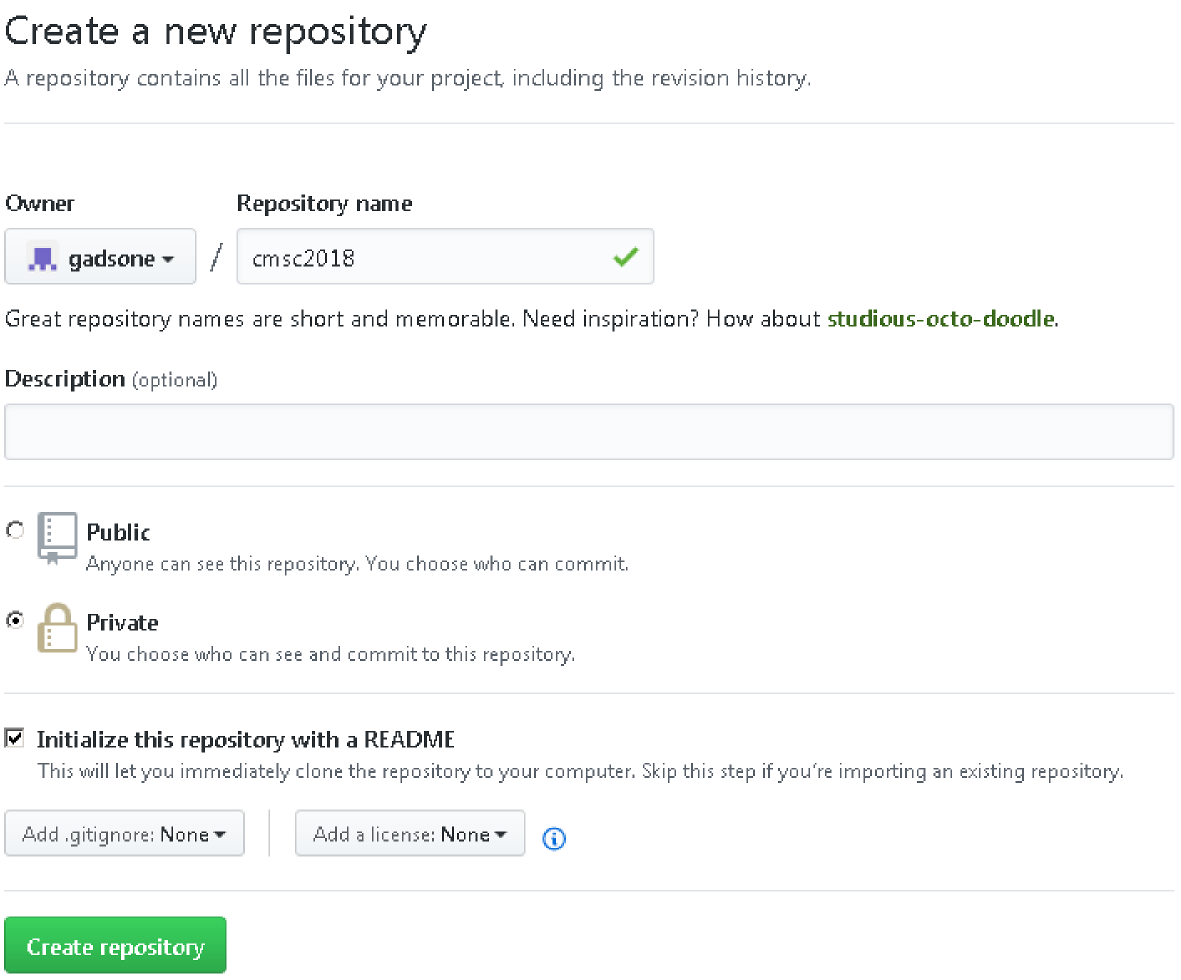


Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Figure 1.1.10: When we are logged into GitHub.com, we can store the project by first clicking on "Start a project."

1. Creating a repository


Create a new repository

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

Owner: gadsone / Repository name: cmsc2018

Great repository names are short and memorable. Need inspiration? How about studious-octo-doodle.

Description (optional)

( ) Public
Anyone can see this repository. You choose who can commit.

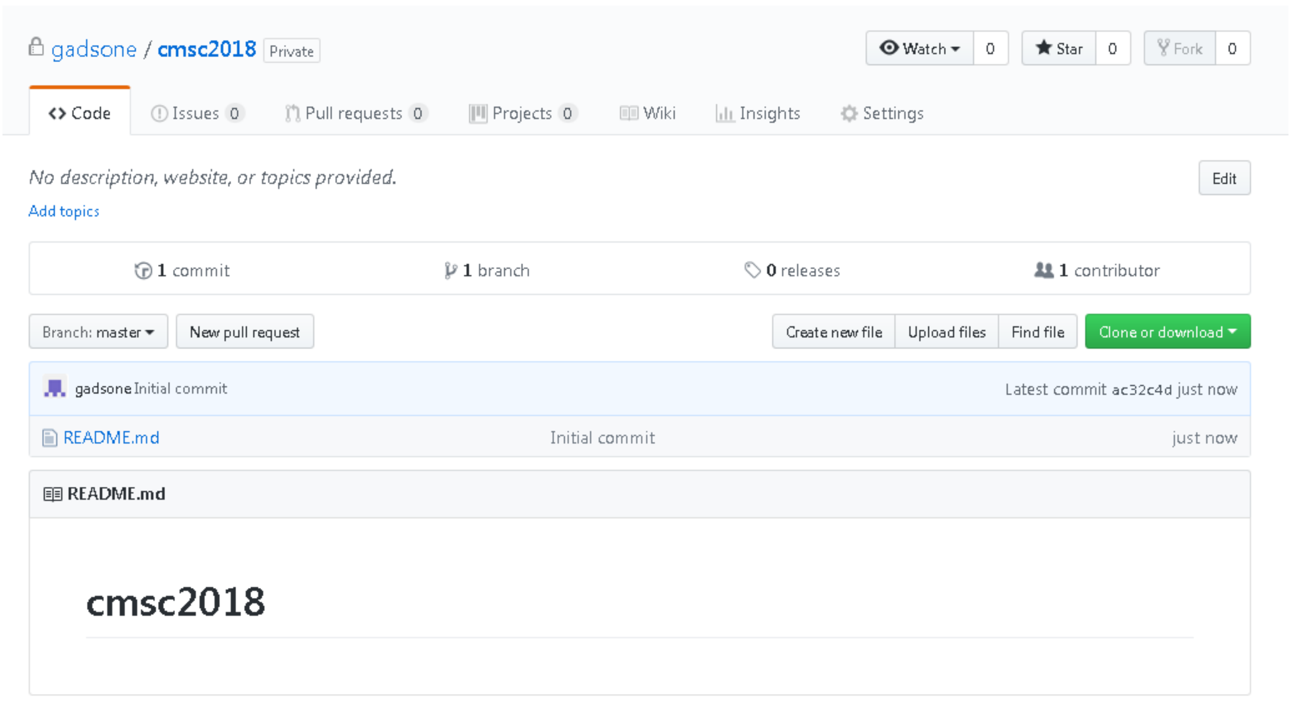
(●) Private
You choose who can see and commit to this repository.

[✓] 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

Figure 1.1.11: Next, we name the repository, select either public or private, and check the initialize README box. Then, click "Create repository."


gadsone / cmsc2018 Private
Watch 0 Star 0 Fork 0
Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings
Edit
Add topics
Branch: master
New pull request
Create new file Upload files Find file Clone or download
README.md
cmsc2018

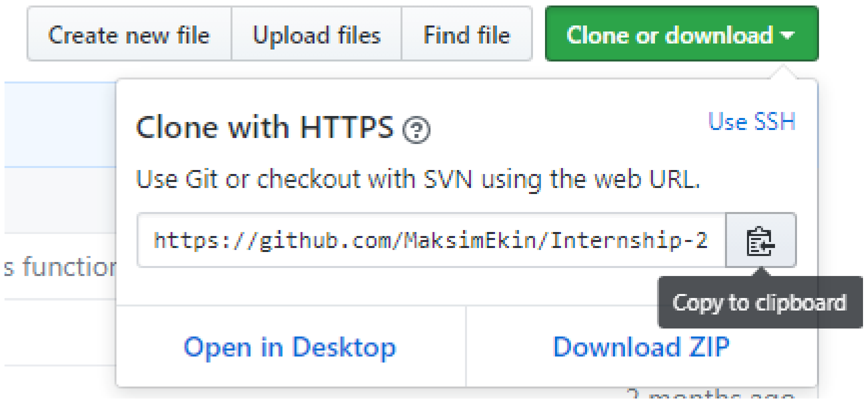
No description, website, or topics provided.

1 commit 1 branch 0 releases 1 contributor

gadsone Initial commit Latest commit ac32c4d just now

README.md Initial commit just now

Figure 1.1.12: We can click on “Clone or download” to open our repository locally.



Use Git or checkout with SVN using the web URL.

```
https://github.com/MaksimEkin/Internship-2
```

Figure 1.1.13: This will give us an HTTPS link we can use to clone our repository anywhere.

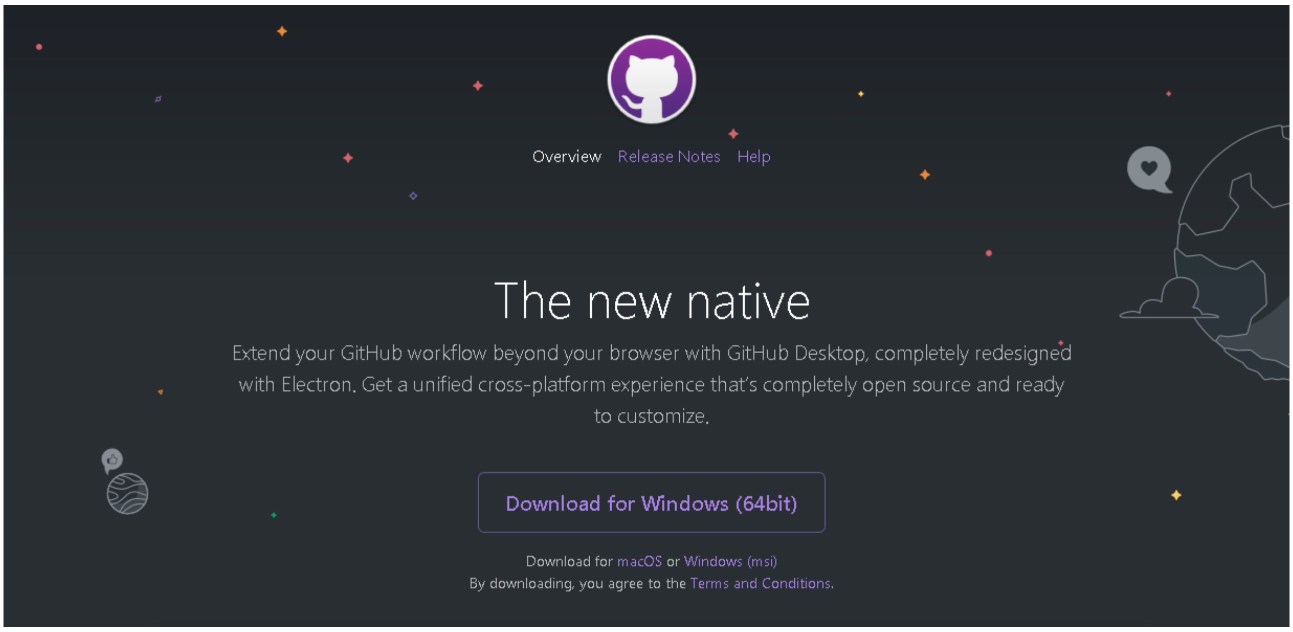
2. How to begin using Git

2.1. Git Desktop

2.1.1. What is Git Desktop?

Git Desktop is a graphical environment for using the git version-control software. It allows us to do all operations that can be done from Git Bash or the Terminal through a graphical interface.

2.1.2. How to download and install Git Desktop



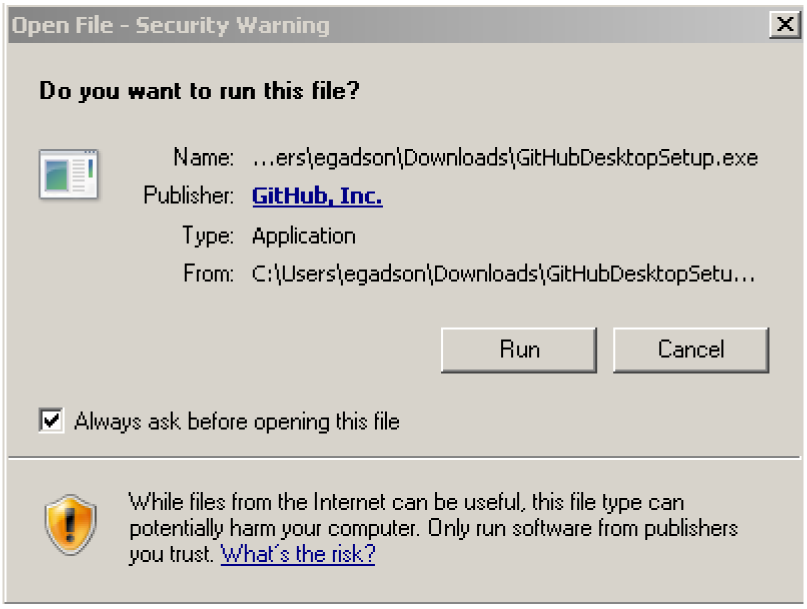
The new native

Extend your GitHub workflow beyond your browser with GitHub Desktop, completely redesigned with Electron. Get a unified cross-platform experience that's completely open source and ready to customize.

Download for macOS or Windows (msi)
By downloading, you agree to the Terms and Conditions.

Figure 2.1.1: We can download GitHub desktop to manage our workflow for Windows and macOS. https://desktop.github.com/ Choose "Download for Windows (64bit)" or "Download for macOS."

1. Installation



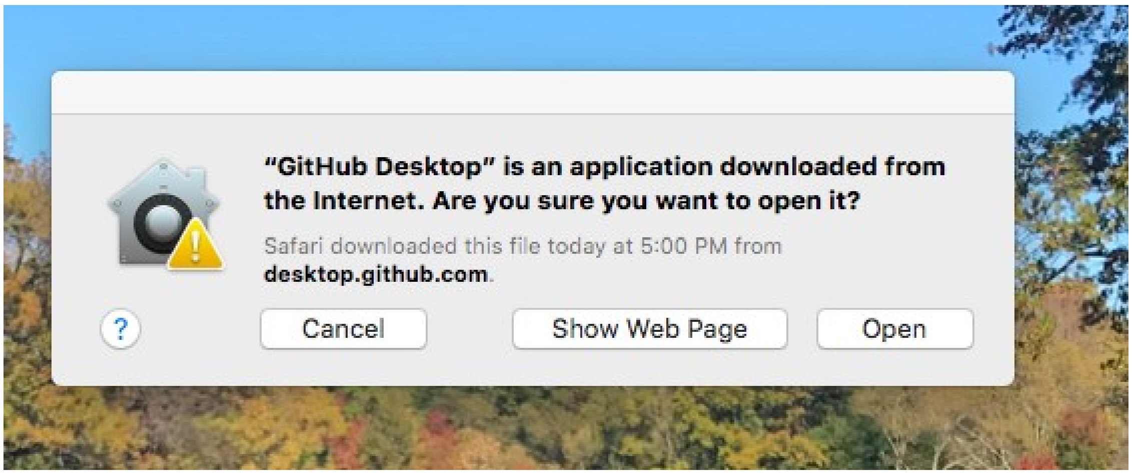
Do you want to run this file?

Name: ...ers\egadson\Downloads\GitHubDesktopSetup.exe
Publisher: GitHub, Inc.
Type: Application
From: C:\Users\egadson\Downloads\GitHubDesktopSetu...

☑ Always ask before opening this file

While files from the Internet can be useful, this file type can potentially harm your computer. Only run software from publishers you trust. What's the risk?

Figure 2.1.2: (WINDOWS) Then, click run in order to start the installation of the program.


"GitHub Desktop" is an application downloaded from the Internet. Are you sure you want to open it?

Safari downloaded this file today at 5:00 PM from **desktop.github.com**.

Figure 2.1.3: (macOS) Click "Open" when a notification pops up asking to download GitHub Desktop.

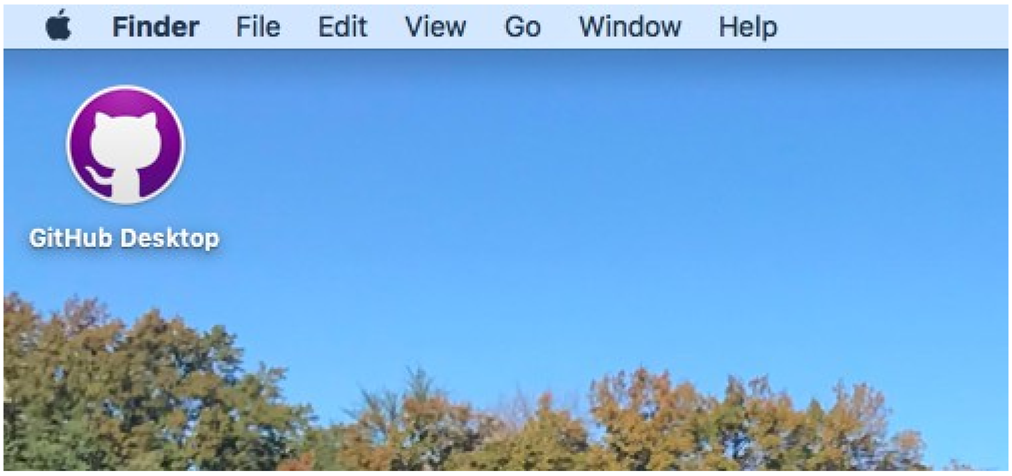
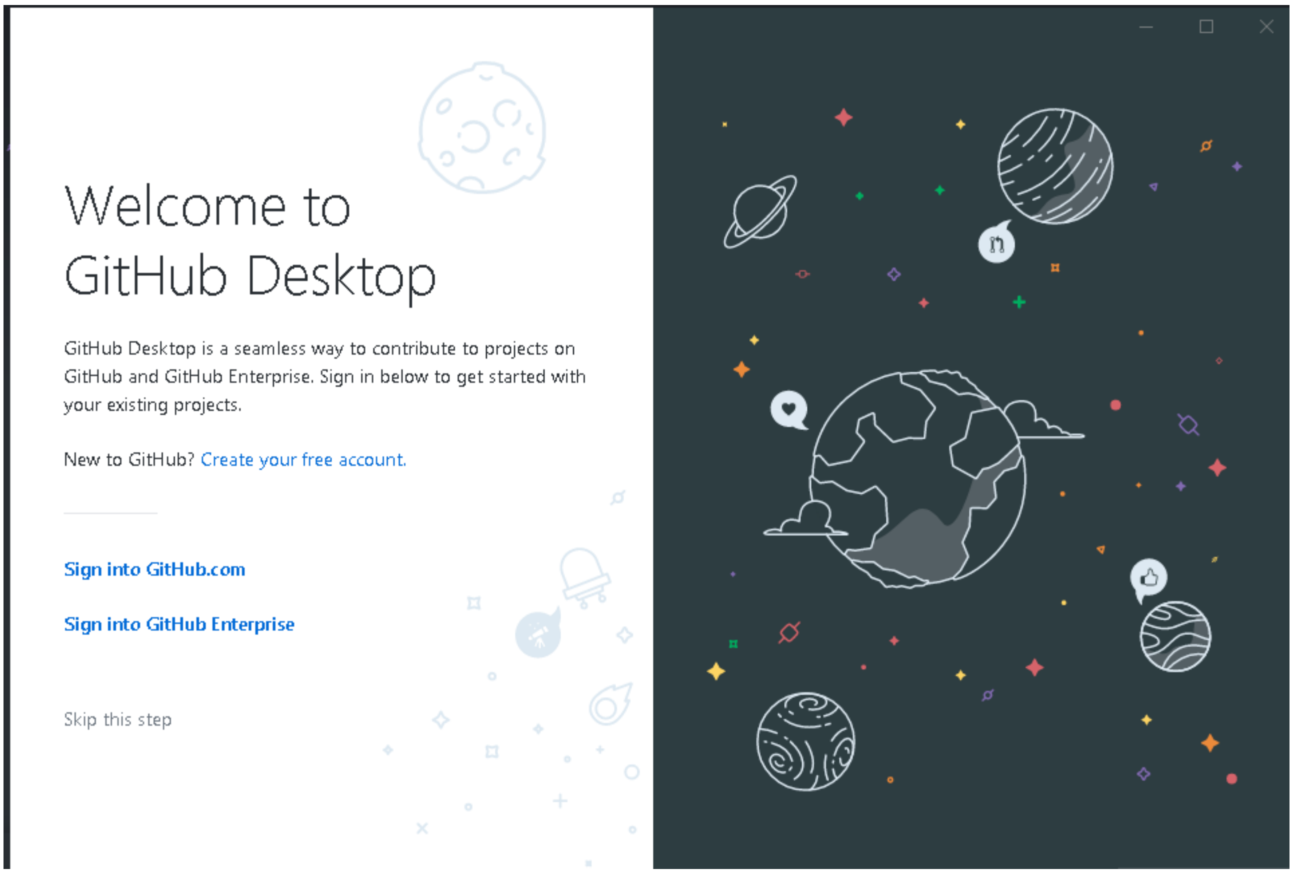


Figure 2.1.4: (macOS) Finally, open the application where it has been downloaded (in this case in the Desktop) in order to start the installation process.



Welcome to GitHub Desktop

GitHub Desktop is a seamless way to contribute to projects on GitHub and GitHub Enterprise. Sign in below to get started with your existing projects.

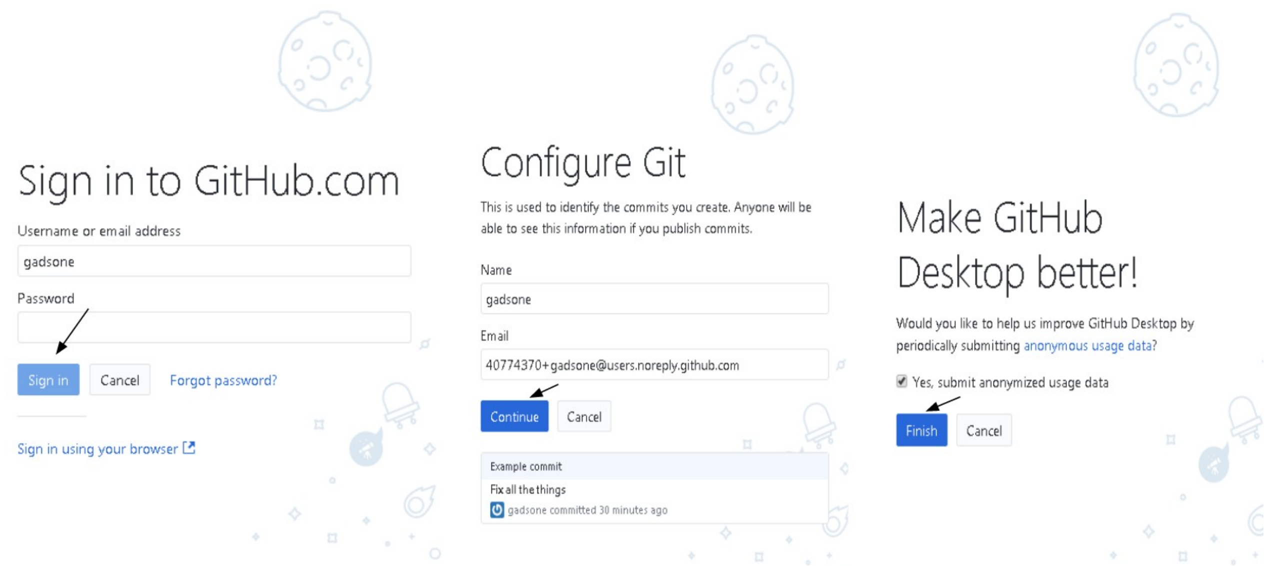
New to GitHub? Create your free account.

Sign into GitHub.com

Sign into GitHub Enterprise

Skip this step

Figure 2.1.5: Once we run GitHub desktop, we can login to our account by choosing "Sign into GitHub.com.



Sign in to GitHub.com

Username or email address: gadsone

Password:

Sign in | Cancel | Forgot password?

Sign in using your browser

Configure Git

This is used to identify the commits you create. Anyone will be able to see this information if you publish commits.

Name: gadsone

Email: 40774370+gadsone@users.noreply.github.com

Continue | Cancel

Example commit
Fix all the things
gadsone committed 30 minutes ago

Make GitHub Desktop better!

Would you like to help us improve GitHub Desktop by periodically submitting anonymous usage data?

☑ Yes, submit anonymized usage data

Finish | Cancel

Figure 2.1.6: Once we successfully sign in using our credentials, we click "Continue" and then "Finish".

2.1.3. How to clone a repository into Git Desktop

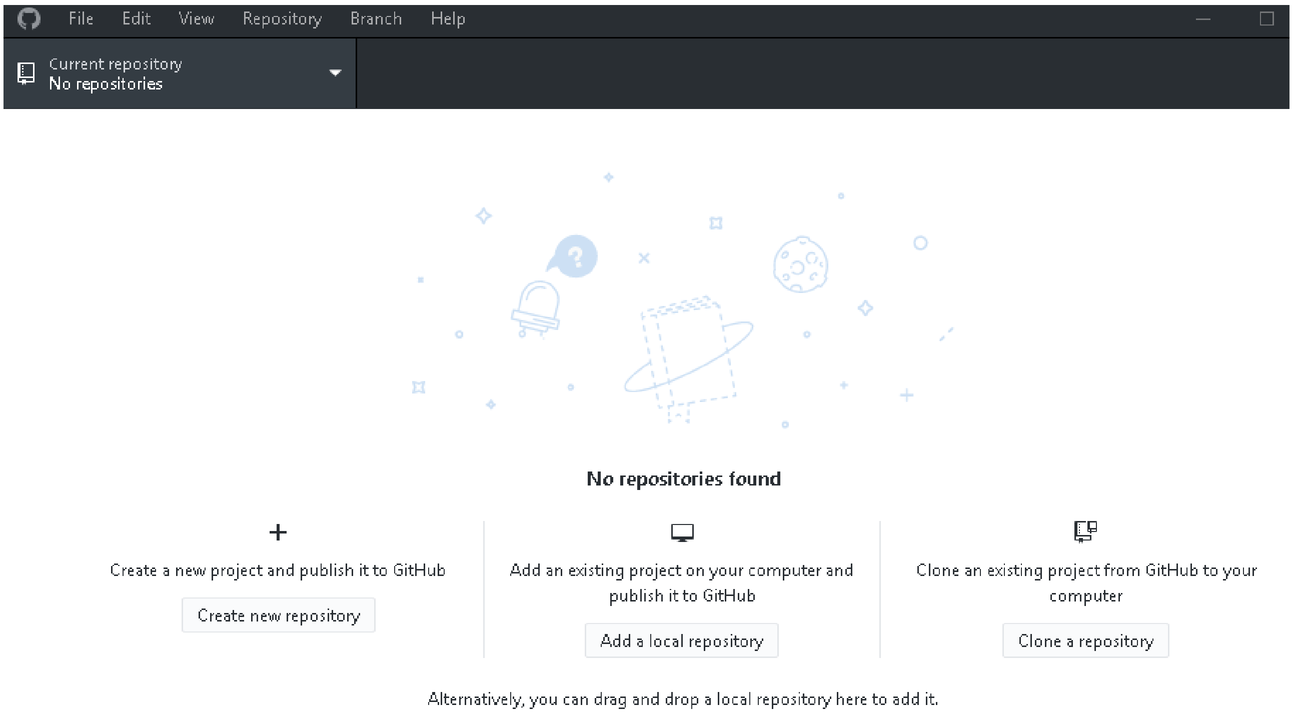

File
Edit
View
Repository
Branch
Help
Current repository
No repositories
No repositories found
Create a new project and publish it to GitHub
Create new repository
Add an existing project on your computer and publish it to GitHub
Add a local repository
Clone an existing project from GitHub to your computer
Clone a repository
Alternatively, you can drag and drop a local repository here to add it.

Figure 2.1.7: To download our repository in GitHub desktop, we select the "Add a local repository" option. Note: If we have not created a repository on GitHub.com, we should select "Create new repository" instead.

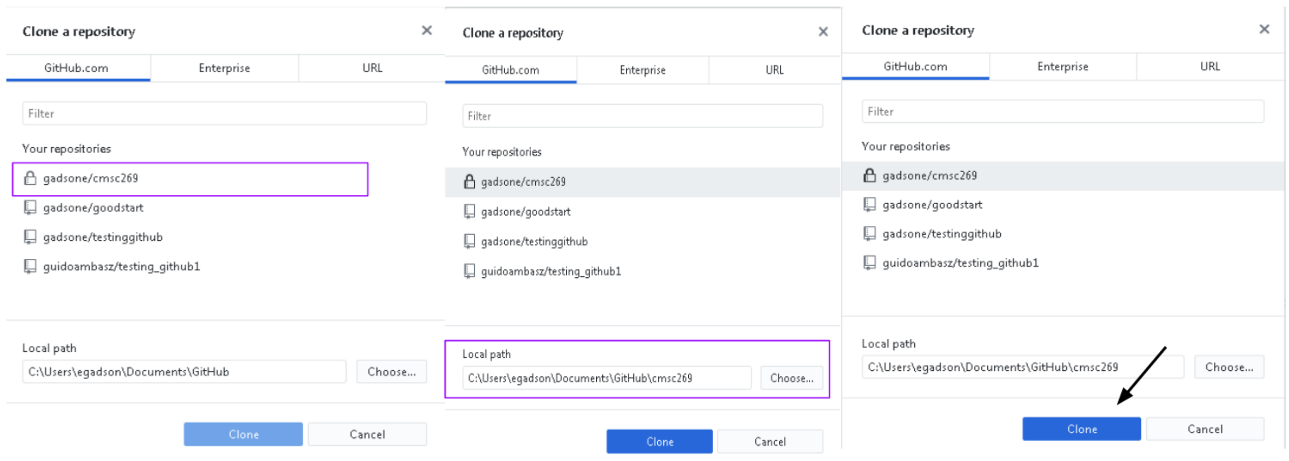


Figure 2.1.8: Next, we select a repository from GitHub.com, choose where to store it locally, and click "Clone."

2.1.4. Comitting our changes

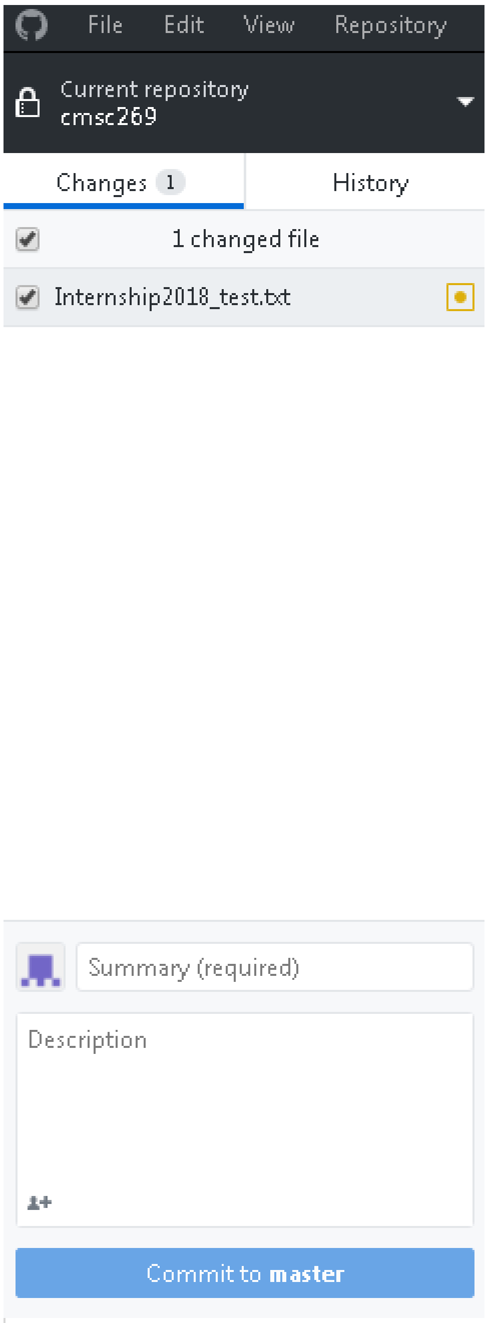


Figure 2.1.9: Add a summary of what was changed (a message). You can add a description but it is not necessary. Click "Commit to [branch]".

2.1.5. Pushing our changes

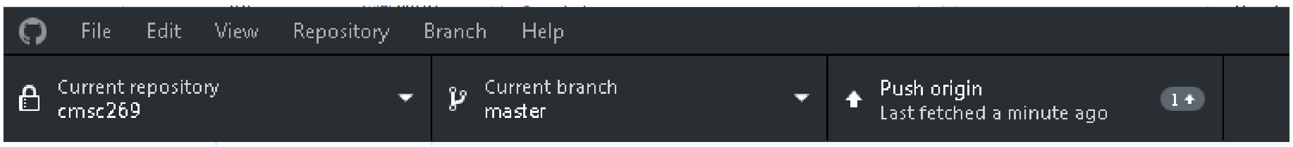
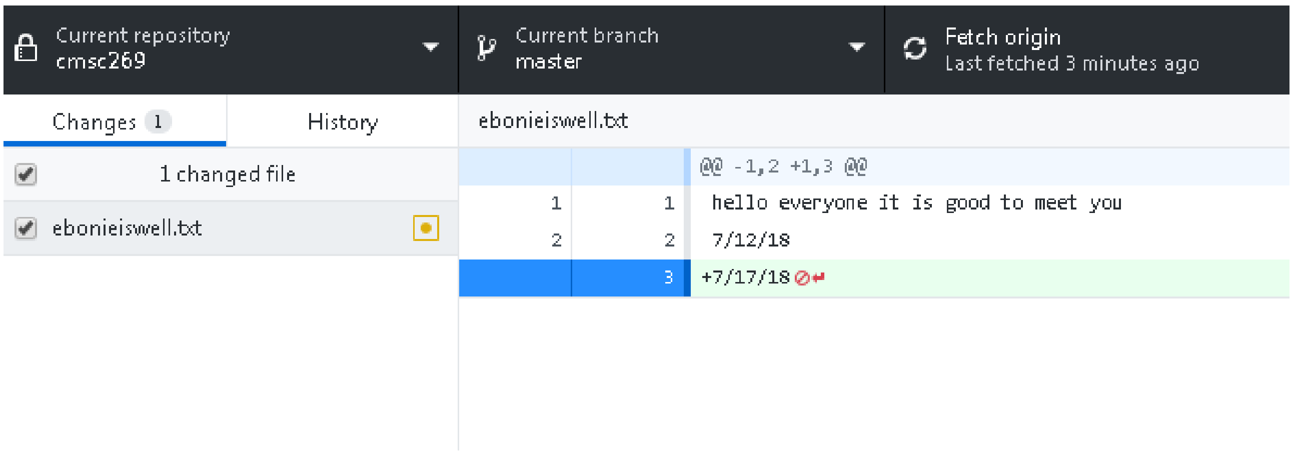


Figure 2.1.10: Select "Push Origin" to push changes to GitHub.com.

2.1.6. Pulling from the repository



```
@@ -1,2 +1,3 @@
1   1   hello everyone it is good to meet you
2   2   7/12/18
    3  +7/17/18
```

Figure 2.1.11:

2.2. Git Bash

2.2.1. What is Git Bash?

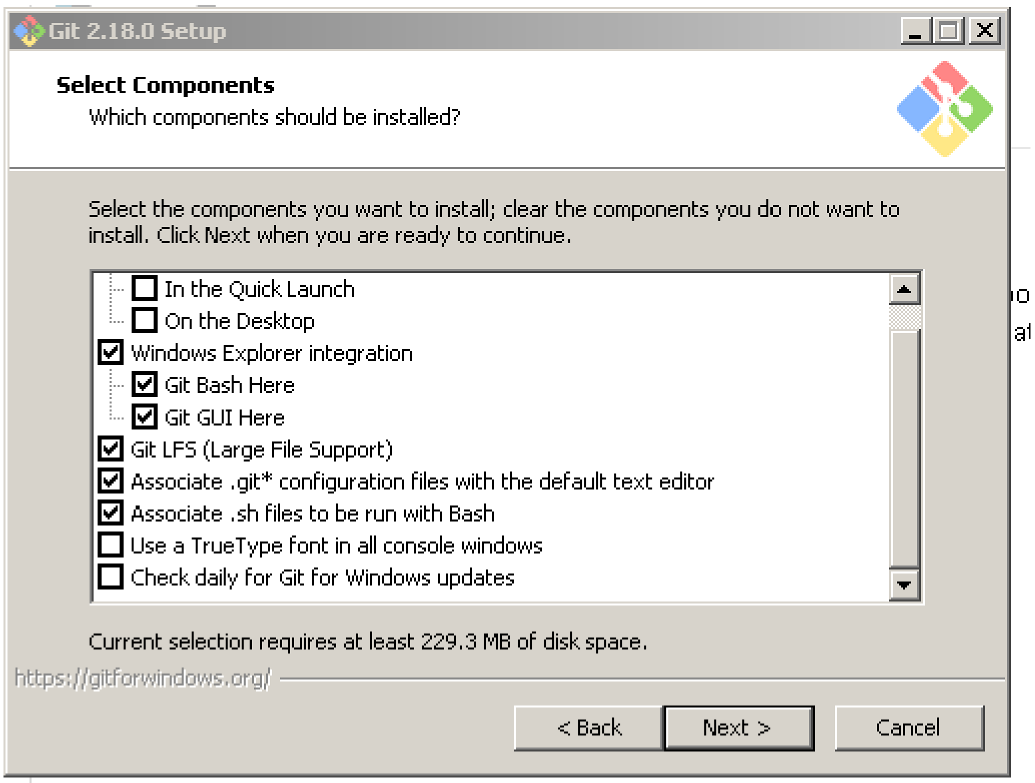
Git bash is a software that allows us to use the *git* software from a terminal emulator using the Windows OS. Most developers usually interact with the git version control system this way, either by using Git Bash or another terminal emulator (in macOS or Linux).

2.2.2. Download

You can download Git Bash here: https://git-scm.com/download/win

2.2.3. Installation

Instructions for installing Git Bash:


Git 2.18.0 Setup

Select Components

Which components should be installed?

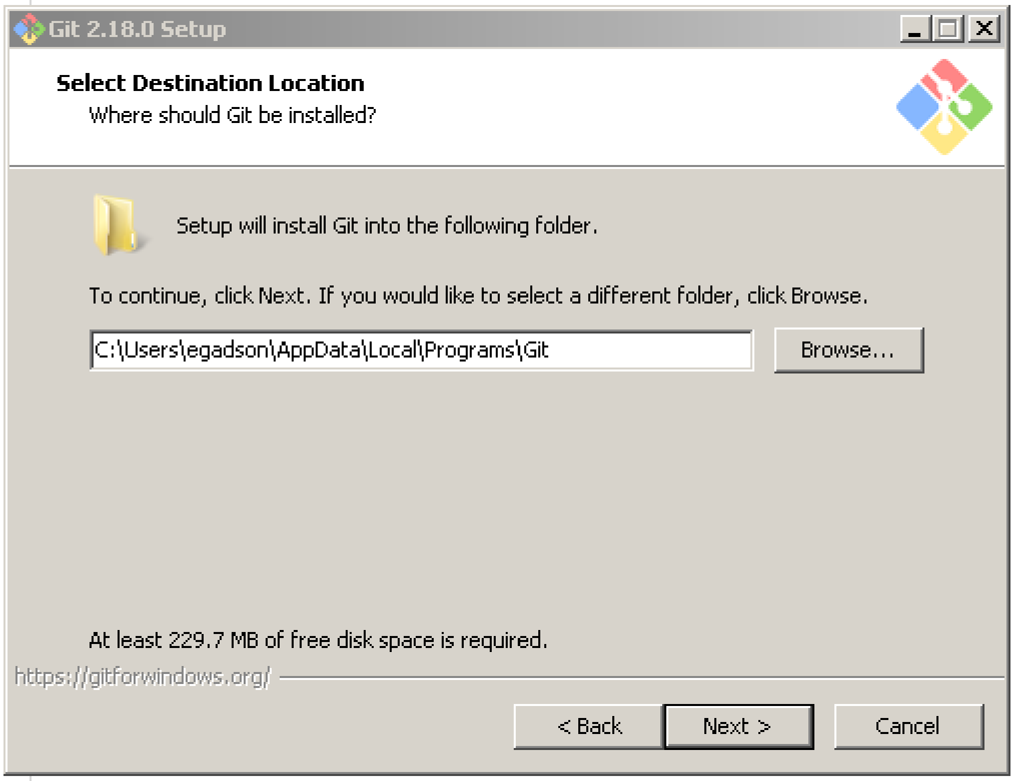
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- [ ] In the Quick Launch
- [ ] On the Desktop
- [x] Windows Explorer integration
  - [x] Git Bash Here
  - [x] Git GUI Here
- [x] Git LFS (Large File Support)
- [x] Associate .git* configuration files with the default text editor
- [x] Associate .sh files to be run with Bash
- [ ] Use a TrueType font in all console windows
- [ ] Check daily for Git for Windows updates

Current selection requires at least 229.3 MB of disk space.

https://gitforwindows.org/

Figure 2.2.1: Make sure all the boxes for "Windows Explorer integration", "Git LFS", "Associate .git configuration", and "Associate .s files are checked." Click "Next".



Select Destination Location

Where should Git be installed?

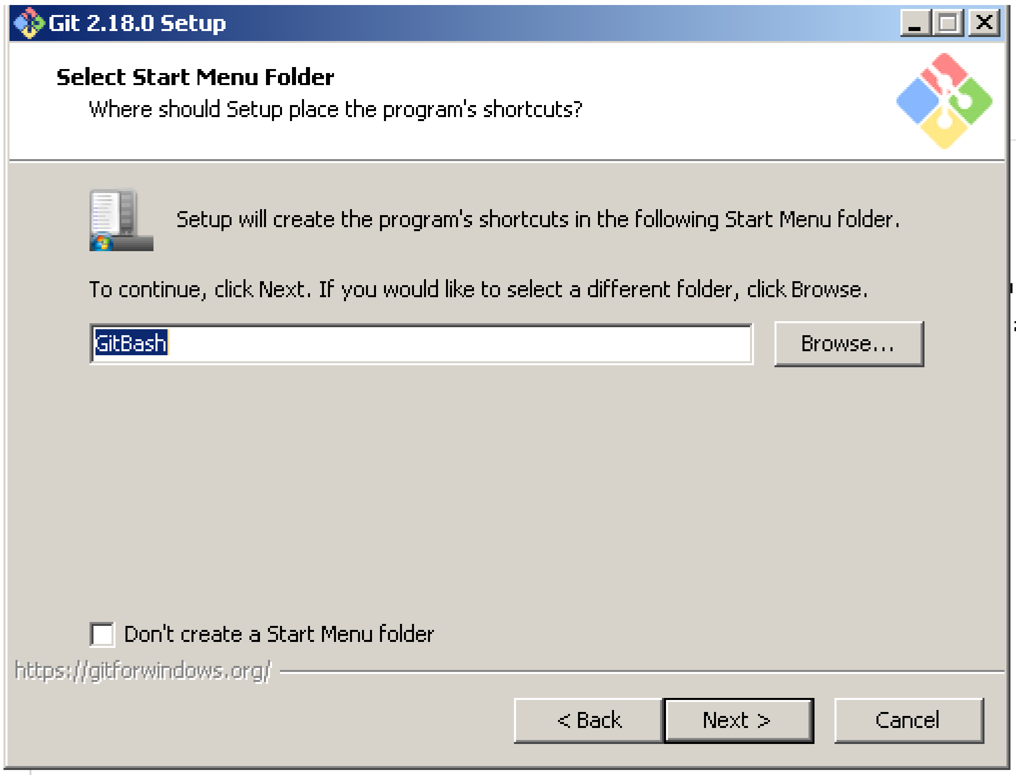
Setup will install Git into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

C:\Users\egadson\AppData\Local\Programs\Git

At least 229.7 MB of free disk space is required.

Figure 2.2.2: After choose a local path where Git will be installed. Click "Next".



Select Start Menu Folder

Where should Setup place the program's shortcuts?

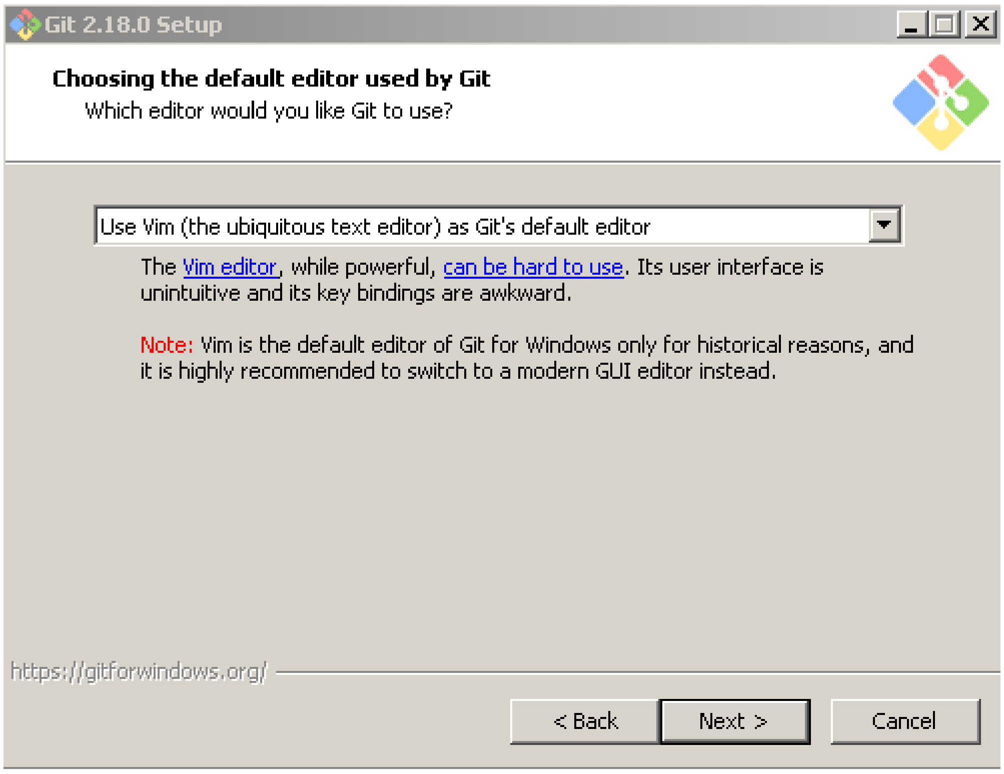
Setup will create the program's shortcuts in the following Start Menu folder.

To continue, click Next. If you would like to select a different folder, click Browse.

GitBash

☐ Don't create a Start Menu folder

Figure 2.2.3: Save the name for the program shortcut in the start menu as "Git Bash" and click "Next".



Choosing the default editor used by Git
Which editor would you like Git to use?

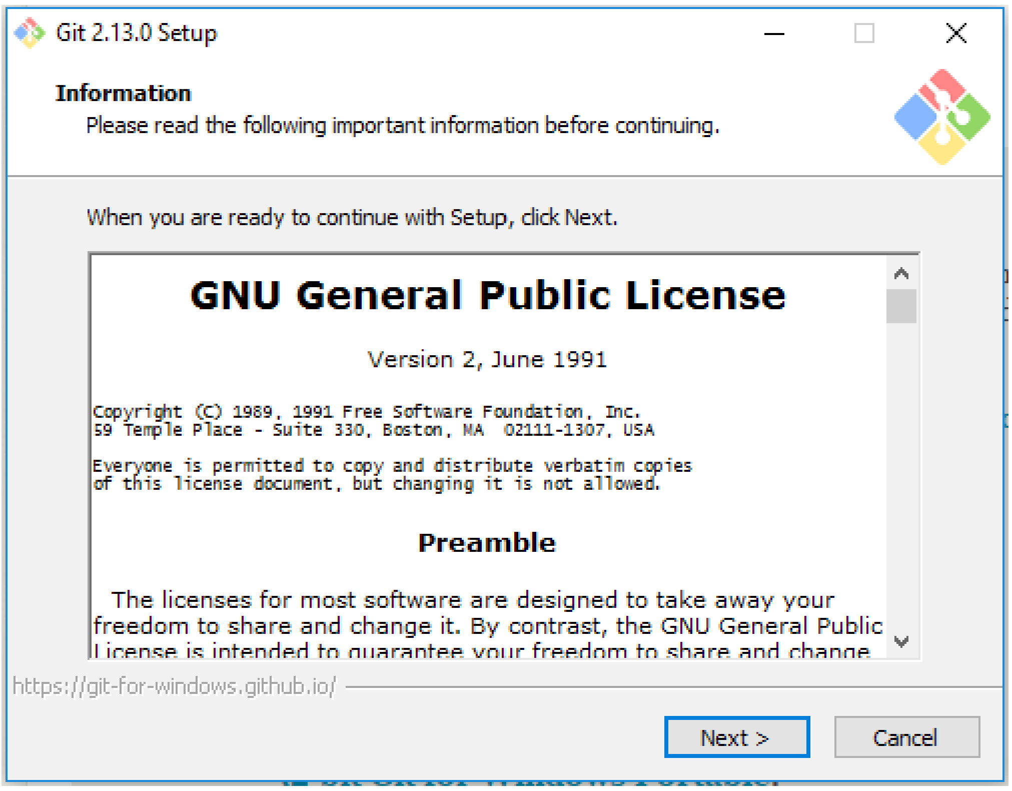
Use Vim (the ubiquitous text editor) as Git's default editor

The Vim editor, while powerful, can be hard to use. Its user interface is unintuitive and its key bindings are awkward.

Note: Vim is the default editor of Git for Windows only for historical reasons, and it is highly recommended to switch to a modern GUI editor instead.

https://gitforwindows.org/

Figure 2.2.4: Choose Git's default editor as "Use Vim (the ubiquitous text editor) as Git's default editor" and click "Next".



Information
Please read the following important information before continuing.

When you are ready to continue with Setup, click Next.

GNU General Public License

Version 2, June 1991

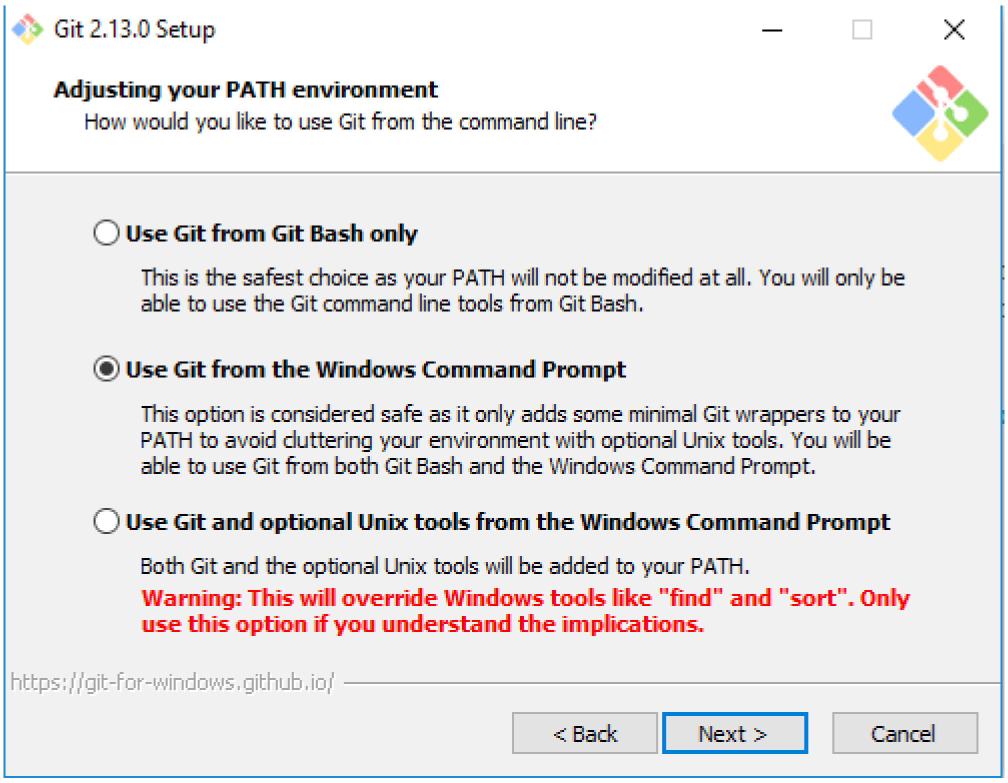
```
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA  02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
```

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change

https://git-for-windows.github.io/

Figure 2.2.5: After reading the "GNU General Public License", click "Next".



Git 2.13.0 Setup

Adjusting your PATH environment

How would you like to use Git from the command line?

○ **Use Git from Git Bash only**

This is the safest choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

[✓ **Use Git from the Windows Command Prompt**]

This option is considered safe as it only adds some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from both Git Bash and the Windows Command Prompt.

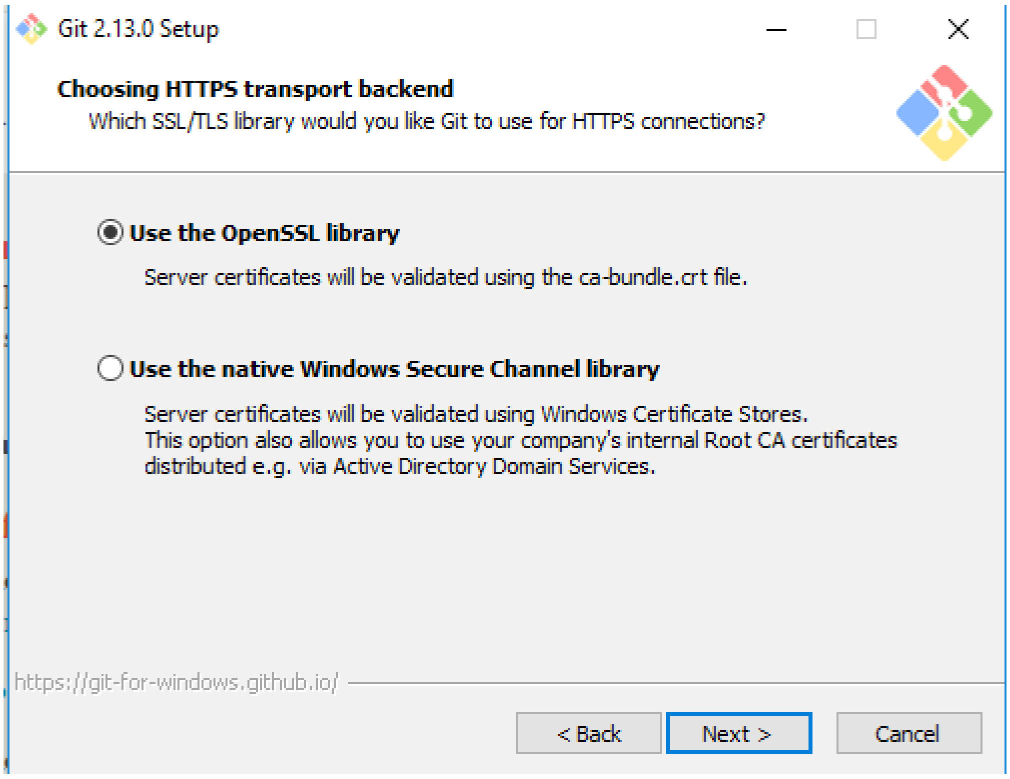
○ **Use Git and optional Unix tools from the Windows Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.

Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

https://git-for-windows.github.io/

Figure 2.2.6: Make sure the "Use Git from Windows Command Prompt" before clicking "Next".



Git 2.13.0 Setup

Choosing HTTPS transport backend

Which SSL/TLS library would you like Git to use for HTTPS connections?

[✓ **Use the OpenSSL library**]

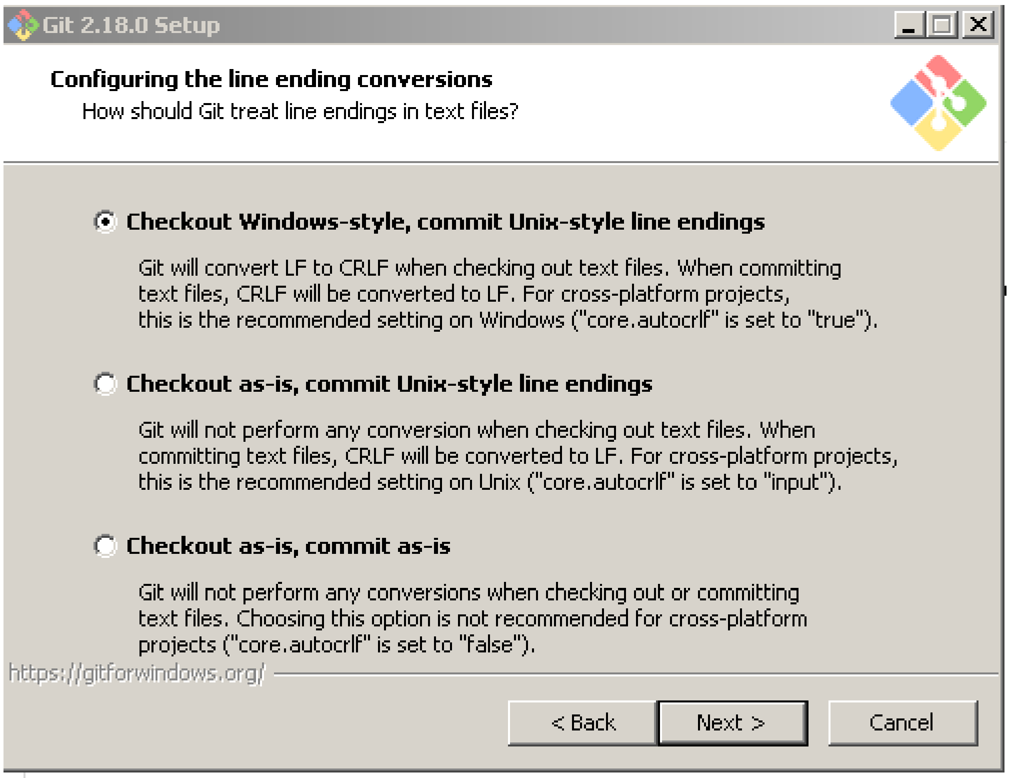
Server certificates will be validated using the ca-bundle.crt file.

○ **Use the native Windows Secure Channel library**

Server certificates will be validated using Windows Certificate Stores. This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.

https://git-for-windows.github.io/

Figure 2.2.7: Choose "Use Open SSL library" and click "Next".


Git 2.18.0 Setup

Configuring the line ending conversions

How should Git treat line endings in text files?

[✓ **Checkout Windows-style, commit Unix-style line endings**]

Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

Checkout as-is, commit Unix-style line endings

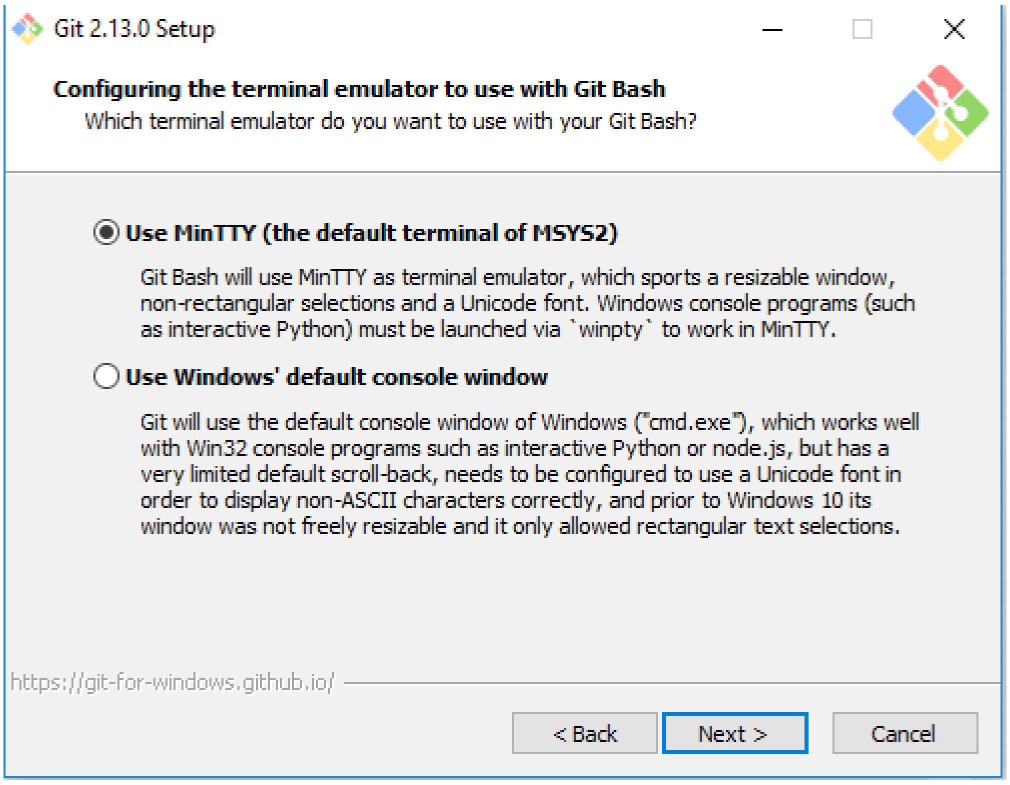
Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").

Checkout as-is, commit as-is

Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").

https://gitforwindows.org/

Figure 2.2.8: Make sure "Checkout Window-style, commit Unix-style line" is the selected option before clicking "Next".


Git 2.13.0 Setup

Configuring the terminal emulator to use with Git Bash

Which terminal emulator do you want to use with your Git Bash?

[✓ **Use MinTTY (the default terminal of MSYS2)**]

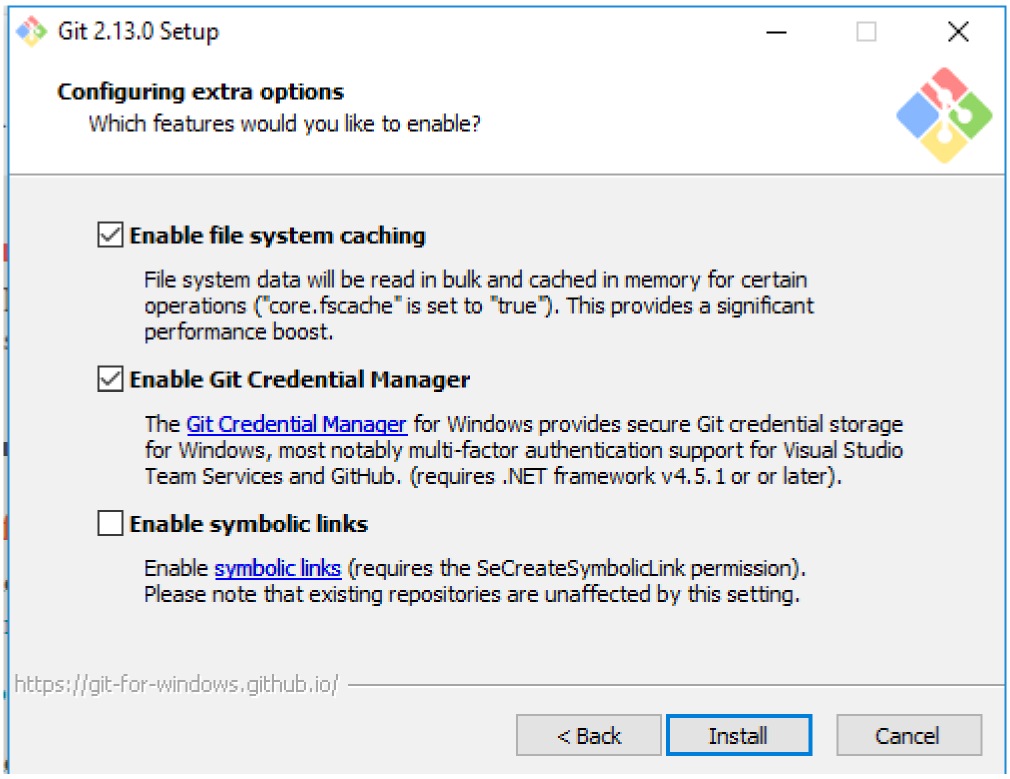
Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `winpty` to work in MinTTY.

Use Windows' default console window

Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.

https://git-for-windows.github.io/

Figure 2.2.9: Select "Use MinTTY (default terminal of MSYS2)" as the GitBash terminal, then click "Next".


Git 2.13.0 Setup

Configuring extra options

Which features would you like to enable?

☑ **Enable file system caching**

File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.

☑ **Enable Git Credential Manager**

The Git Credential Manager for Windows provides secure Git credential storage for Windows, most notably multi-factor authentication support for Visual Studio Team Services and GitHub. (requires .NET framework v4.5.1 or or later).

☐ **Enable symbolic links**

Enable symbolic links (requires the SeCreateSymbolicLink permission). Please note that existing repositories are unaffected by this setting.

Figure 2.2.10: Make sure all but the "Enable symbolic links" box are checked before clicking "Install".

2.3. *Terminal*

2.3.1. macOS

1. Download

   Download *git* from the official website here: https://git-scm.com/download/mac

2. Installation

   Open the downloaded files and open the *package* (.pkg) file. Then, click on "Install".

2.3.2. Linux

Installing *git* on different Linux-based distributions

1. Ubuntu

   Open a terminal and type in the following command:

```
sudo apt install git-all
```

2. Fedora

```
sudo dnf install git-all
```

3. On Git commands in general:

The instructions shown below are the same for *all* methods shown (except for *Git Desktop*). This means, that whether you are using *Git Bash* or a terminal in either *macOS* or *Linux*, the commands will be the same.

4. Git Status

4.1. What is it for?

The *git status* shows us the status of our files within our working project. This means that it will show whether changes have been made from the previous version of it.

4.2. Usage

In order to use it, we execute the following command:

```
git status
```

This will show something similar to the following:

```
Untracked files:
    (use git add <file>... to include in what will be committed)

    file_name.txt
```

4.3. Git status - Add

The add command will allows us to *add* files in order to later have them change the latest version of our project (essentially, adding our new changes). This can be done with the following command:

```
git add <file>
```

Once doing so, the file will be added. The status will know look like this:

```
Changes to be commited:
   (use git reset HEAD <file>..." to unstage)

      new file: hello_there.txt
```

4.4. Git status - Remove

The remove command has the opposite effect to the add command. This will revoke changes (not add them) once these want to be made.

```
git remove <file>
```

5. Git Commit

5.1. What is it for?

The *git commit* command allows us to tell the git software that we want the added changes to be later added into the latest version of our project. The way we do this is by *comitting* our changes, and adding a message explaining what changes we have made.

```
git commit -m "Our message goes here. We briefly explain what it is we have
changed."
```

Example case:

```
$ git commit -m "Added a welcome message to the website's homepage"
[master 74a4776] made changes
```

6. Git Push

6.1. What is it for?

The *git push* command is the final step towards making an effective change in our project. It tells git to grab all the different changes we have made and update the repository stored in github.

Usage:

```
git push
```

This will prompt us for our username and password for github. This is a security measure, in order to make sure only a valid user can edit the project.

6.2. Git Pull

6.3. What is it for?

The *git pull* command does the opposite of the *git push* command. It gets all the changes from our project stored in github in order to have the latest changes to the project.

This means that we should either get a confirmation message stating that the local (our) version of the project has been updated, or that it already was upto date.