How to not destroy all our work

AN INTRODUCTION TO GIT FOR UMSAE FORMULA ELECTRIC

(No pressure)

Version control (meta)

1. Authored
    - UMSAE Chair Brett Stevens, November 2022

Wtf is git?

From Wikipedia:

- "Git (/gɪt/) is free and open-source software for **distributed version control**: tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows (thousands of parallel branches running on different systems)"

Authored by Linus Torvalds in 2005 for the development of the Linux kernel.

Thank Mr. Linux!

Notable Git Services

Github
Hosts git repos

GitKraken
Desktop interface for remote repos
(Sparkies love/hate this shit)

GitLab
Another interface for remote repos
(Less used but useful)

VSCode / Intellij
Or just about any general purpose IDE will have built in git support.

Definitions

Repository (Repo): A code base or project that you want to improve upon

Commit: A version of the repo that *should be* bug free

Branch: A separate development path of the repo that can be used to work in parallel with peers

Origin: The URL that links to your remote repo

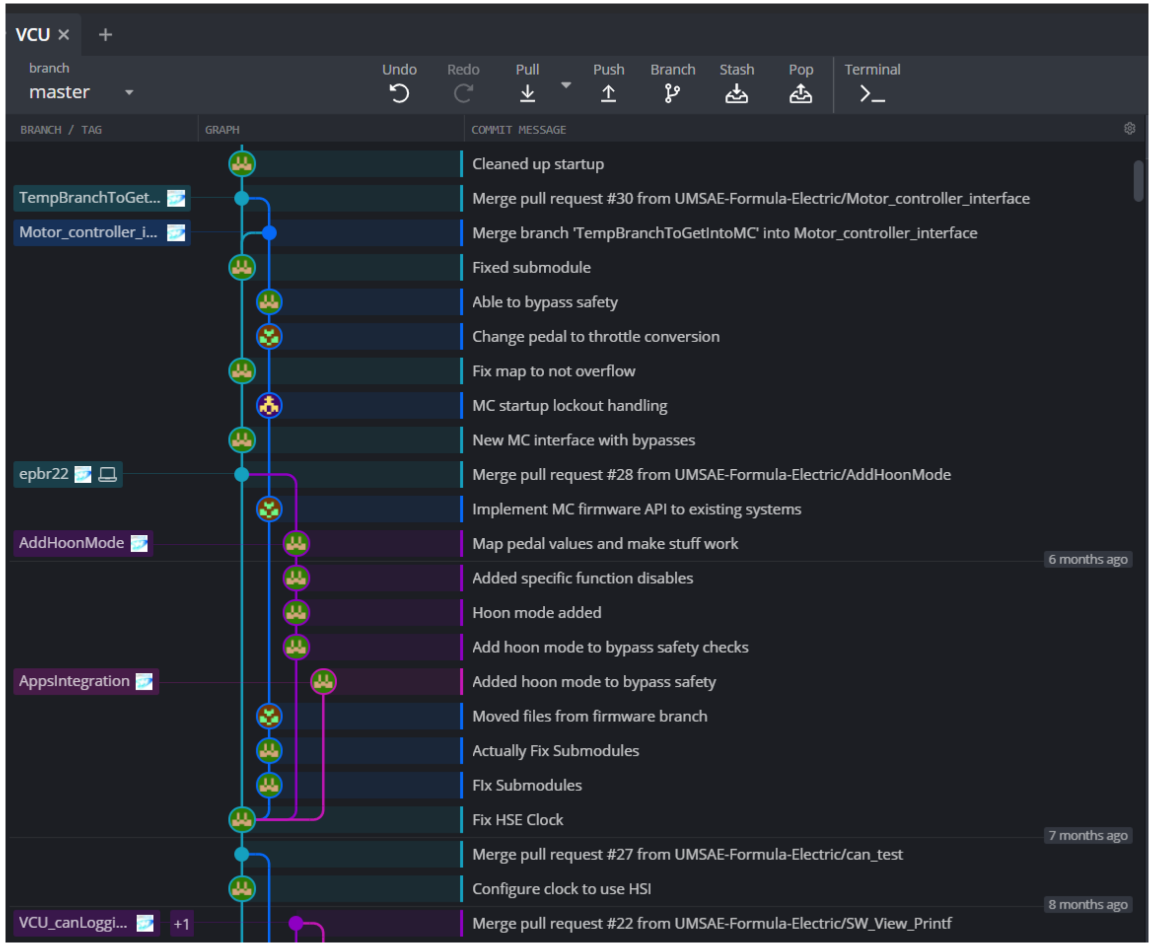
HEAD: The most recent commit in a branch

Path: A map to navigate within your local storage to a directory

Ex: (i.e. C:\Users\Ozone\HomeworkFolder)

Directory: Programmer word for folder

Structure



BRANCH / TAG	COMMIT MESSAGE	
	Cleaned up startup	
TempBranchToGet...	Merge pull request #30 from UMSAE-Formula-Electric/Motor_controller_interface	
Motor_controller_i...	Merge branch 'TempBranchToGetIntoMC' into Motor_controller_interface	
	Fixed submodule	
	Able to bypass safety	
	Change pedal to throttle conversion	
	Fix map to not overflow	
	MC startup lockout handling	
	New MC interface with bypasses	
epbr22	Merge pull request #28 from UMSAE-Formula-Electric/AddHoonMode	
	Implement MC firmware API to existing systems	
AddHoonMode	Map pedal values and make stuff work	6 months ago
	Added specific function disables	
	Hoon mode added	
	Add hoon mode to bypass safety checks	
AppsIntegration	Added hoon mode to bypass safety	
	Moved files from firmware branch	
	Actually Fix Submodules	
	FIx Submodules	
	Fix HSE Clock	
	Merge pull request #27 from UMSAE-Formula-Electric/can_test	7 months ago
	Configure clock to use HSI	
VCU_canLoggi... +1	Merge pull request #22 from UMSAE-Formula-Electric/SW_View_Printf	8 months ago

- A series of small commits within branches
- Ideally there is a “main” branch and development (dev) branches
- Dev branches get **merged** into main branch after the code is reviewed and confirmed to be functional

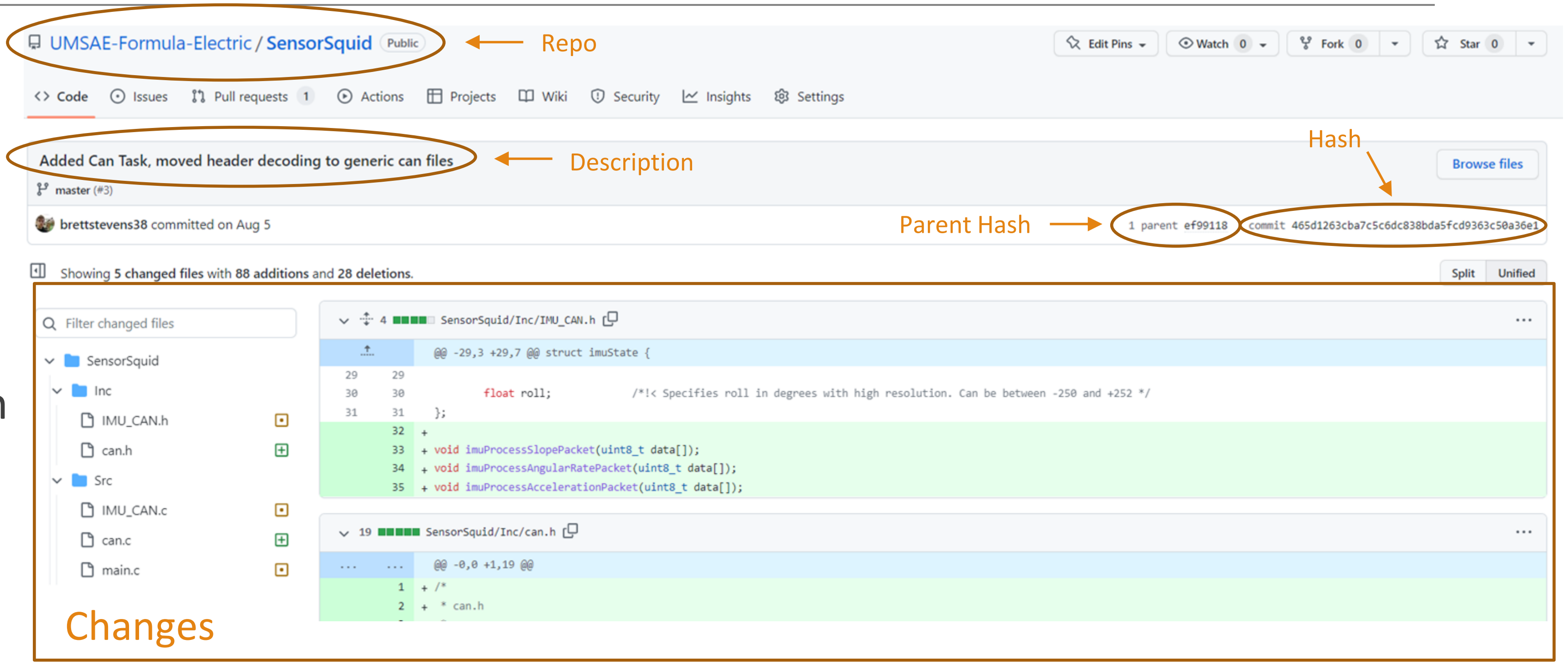
What’s in a commit?

Parent: The previous commit that you are modifying

Changes: The changes that you have made to the parent

Description: A concise but accurate description of what you changed in the repo

Hash: A 20 byte identifier that is generated by the git software



Added Can Task, moved header decoding to generic can files

master (#3)

brettstevens38 committed on Aug 5

1 parent ef99118 commit 465d1263cba7c5c6dc838bda5fcd9363c50a36e1

Showing 5 changed files with 88 additions and 28 deletions.

```
4 SensorSquid/Inc/IMU_CAN.h
@@ -29,3 +29,7 @@ struct imuState {

        float roll;             /*!< Specifies roll in degrees with high resolution. Can be between -250 and +252 */
 };
+
+ void imuProcessSlopePacket(uint8_t data[]);
+ void imuProcessAngularRatePacket(uint8_t data[]);
+ void imuProcessAccelerationPacket(uint8_t data[]);
```

```
19 SensorSquid/Inc/can.h
@@ -0,0 +1,19 @@
+ /*
+  * can.h
```

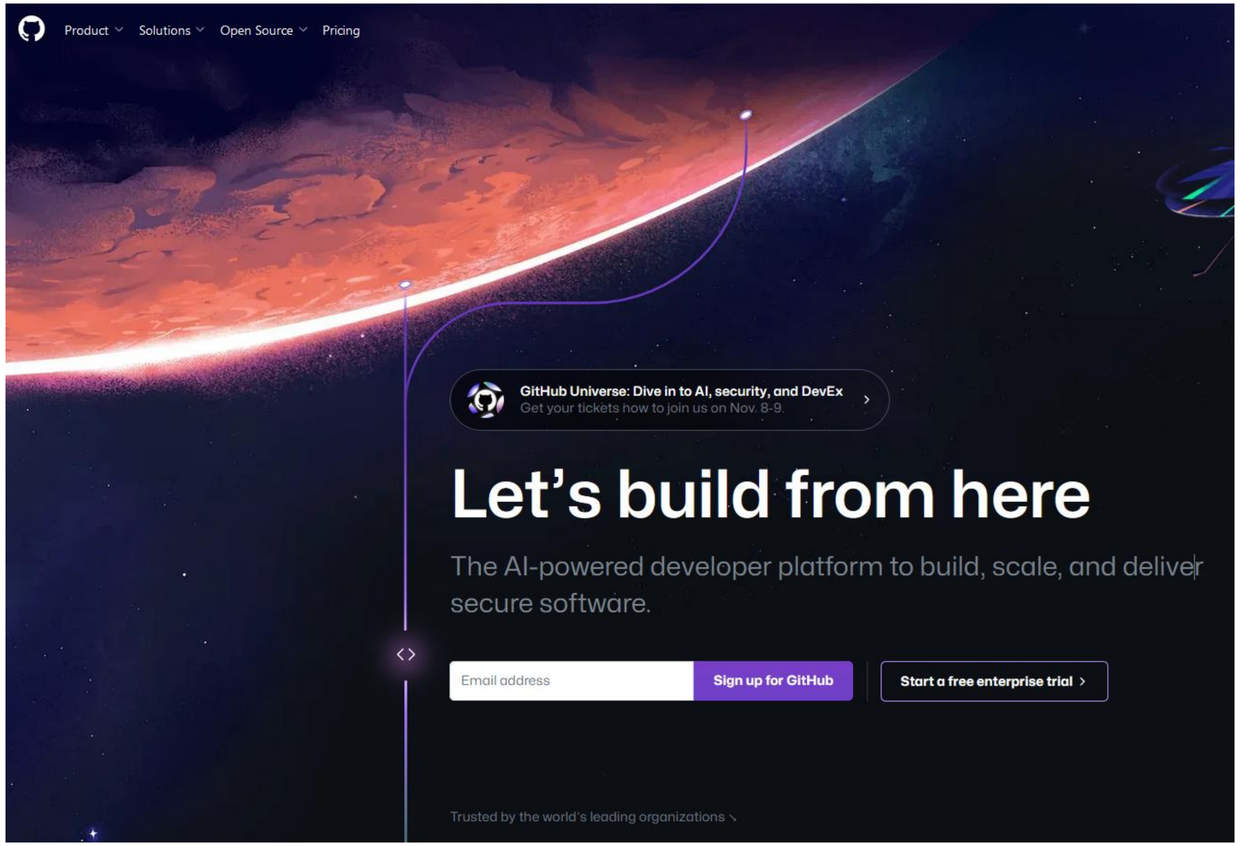
Install / Setup

WAY TOO MANY INSTALL OPTIONS

Make a Github account

Go to https://github.com/

Make an account using your personal information & email. Do not use your student email.



GitHub Universe: Dive in to AI, security, and DevEx
Get your tickets now to join us on Nov. 8-9.

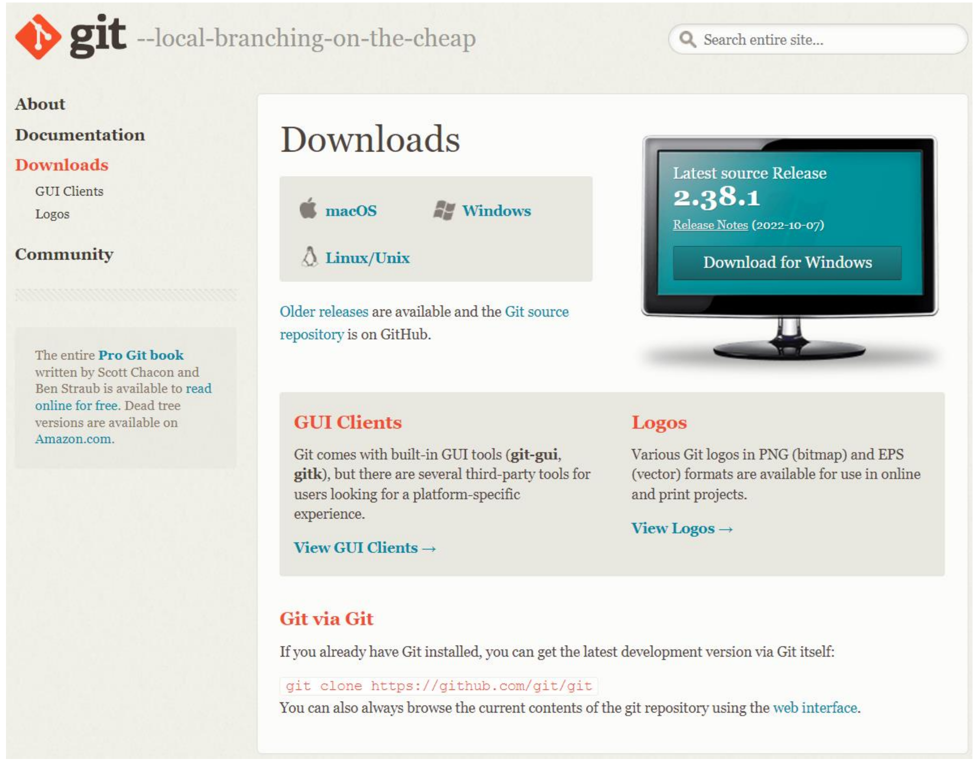
Let's build from here

The AI-powered developer platform to build, scale, and deliver secure software.

Trusted by the world's leading organizations

Download Git

Go to https://git-scm.com/downloads, or just google "download git" and choose your platform.



git --local-branching-on-the-cheap

Search entire site...

About
Documentation
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Downloads

macOS Windows
Linux/Unix

Older releases are available and the Git source repository is on GitHub.

Latest source Release
2.38.1
Release Notes (2022-10-07)
Download for Windows

GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

View GUI Clients →

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

View Logos →

Git via Git

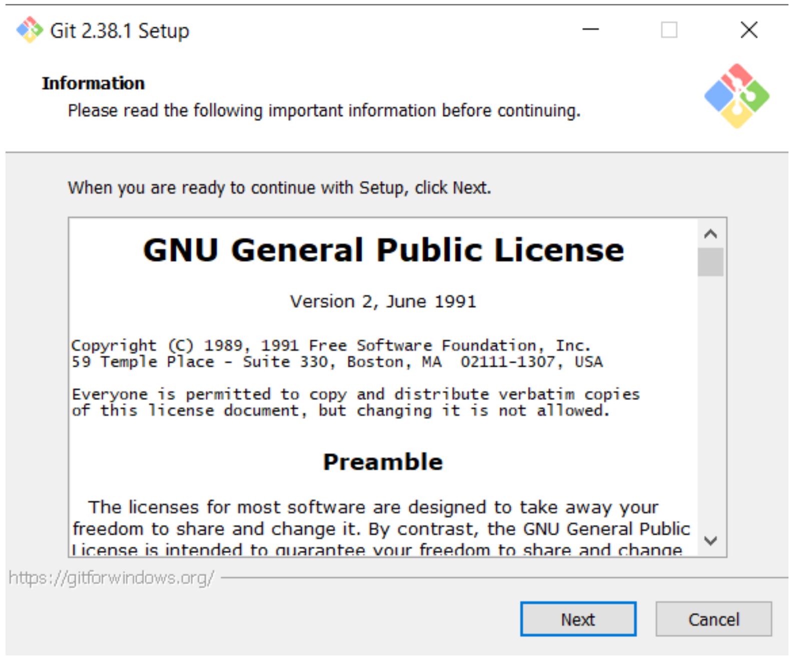
If you already have Git installed, you can get the latest development version via Git itself:

```
git clone https://github.com/git/git
```

You can also always browse the current contents of the git repository using the web interface.

Download Git

Accept the license, you weren't gonna read it anyways…



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

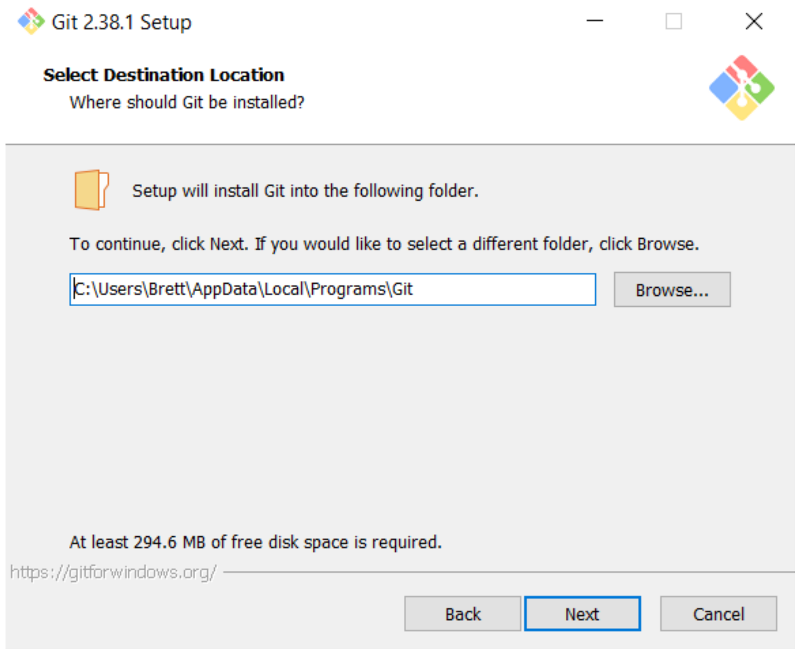
```
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA  02111-1307, USA

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

Download Git

Download location doesn't matter


Git 2.38.1 Setup

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\Brett\AppData\Local\Programs\Git

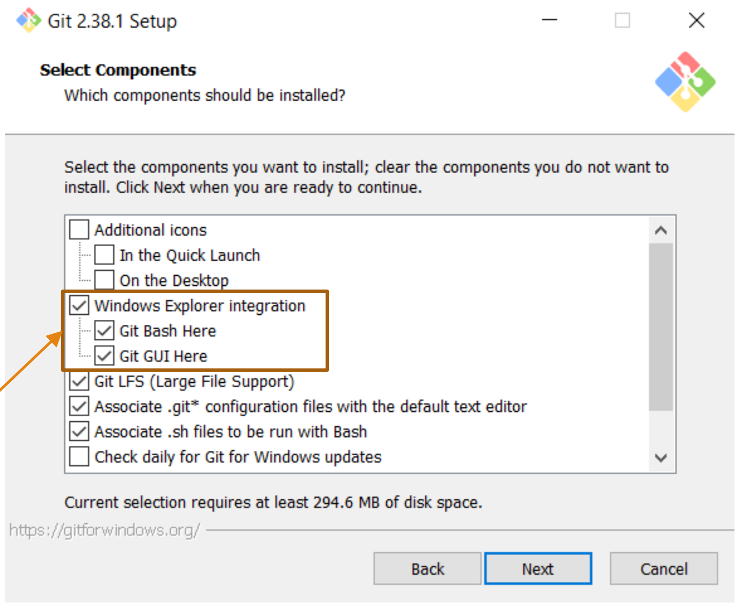
At least 294.6 MB of free disk space is required.

https://gitforwindows.org/

Download Git

You can keep all of these default

This is especially useful


Git 2.38.1 Setup

Select Components
Which components should be installed?

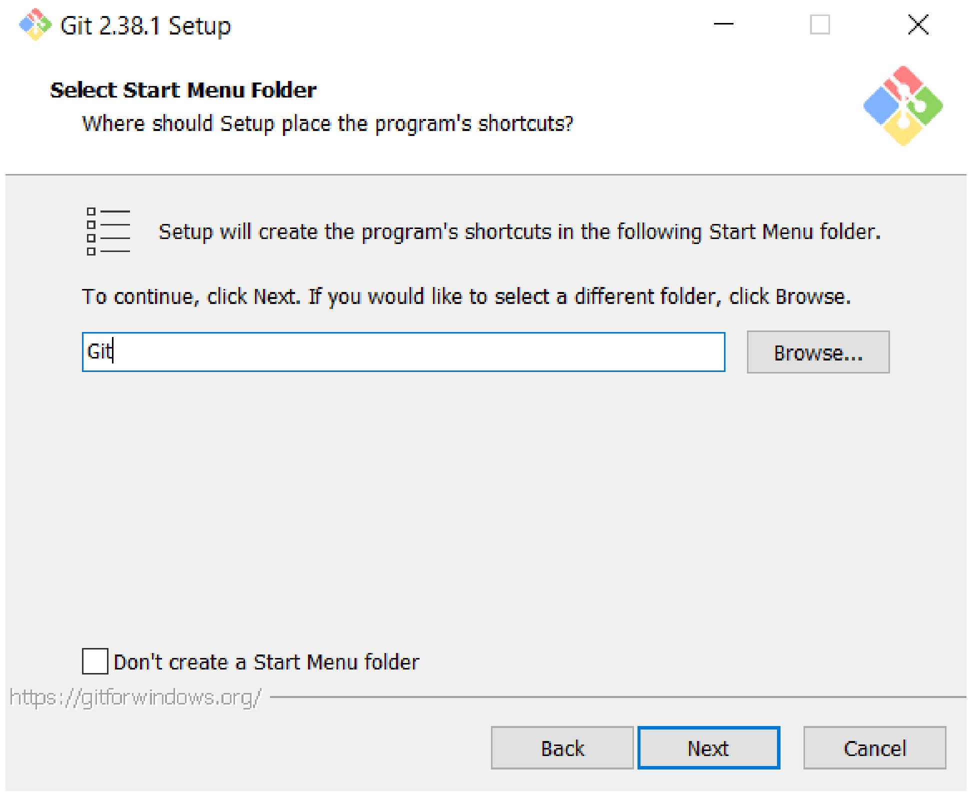
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- [ ] Additional icons
  - [ ] In the Quick Launch
  - [ ] On the Desktop
- [✓] Windows Explorer integration
  - [✓] Git Bash Here
  - [✓] Git GUI Here
- [✓] Git LFS (Large File Support)
- [✓] Associate .git* configuration files with the default text editor
- [✓] Associate .sh files to be run with Bash
- [ ] Check daily for Git for Windows updates

Current selection requires at least 294.6 MB of disk space.

Download Git

Start menu folder, doesn't matter



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.

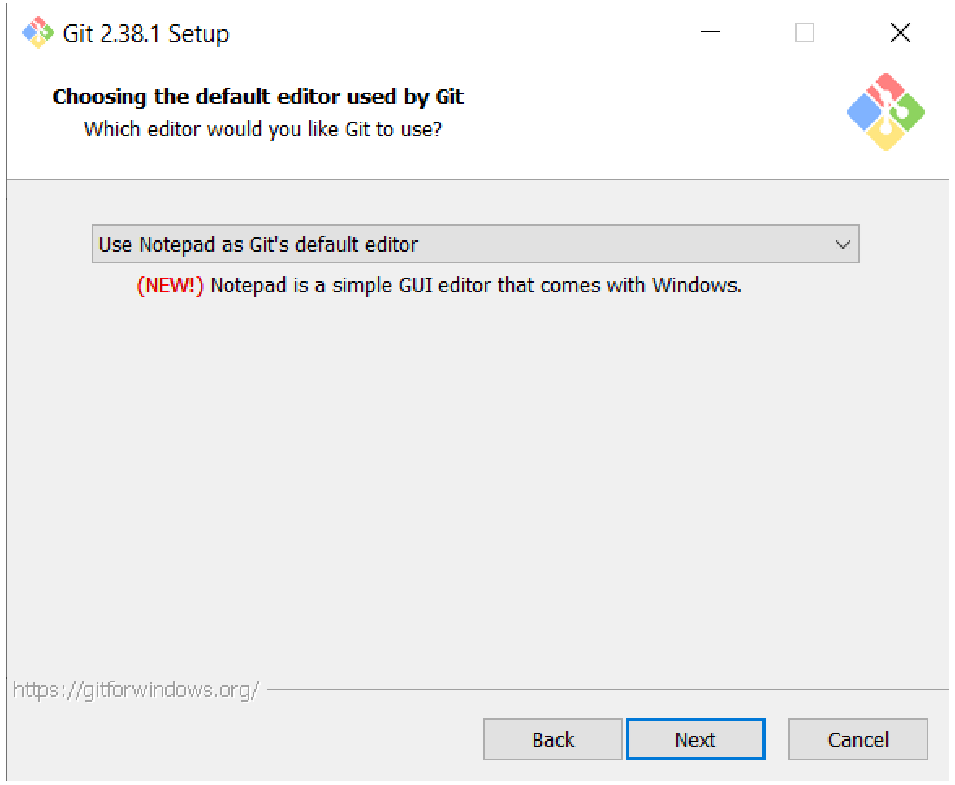
To continue, click Next. If you would like to select a different folder, click Browse.

Git

Download Git

Default text editor, change it to your preference

- Doesn't really matter either, just change it from Vim, Notepad is the simplest option


Git 2.38.1 Setup

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

Use Notepad as Git's default editor

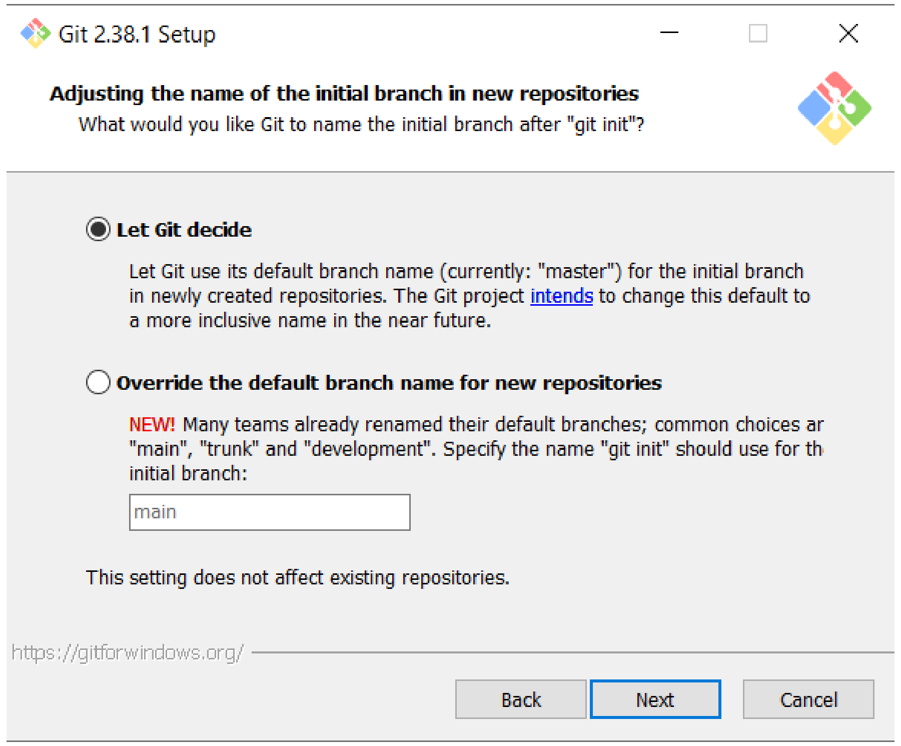
(NEW!) Notepad is a simple GUI editor that comes with Windows.

https://gitforwindows.org/

Download Git

Default branch name:

- Moving away from “master” and “slave” conventions, do what you like.


Git 2.38.1 Setup

Adjusting the name of the initial branch in new repositories

What would you like Git to name the initial branch after "git init"?

[✓ **Let Git decide**]

Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project intends to change this default to a more inclusive name in the near future.

Override the default branch name for new repositories

NEW! Many teams already renamed their default branches; common choices ar "main", "trunk" and "development". Specify the name "git init" should use for th initial branch:

main

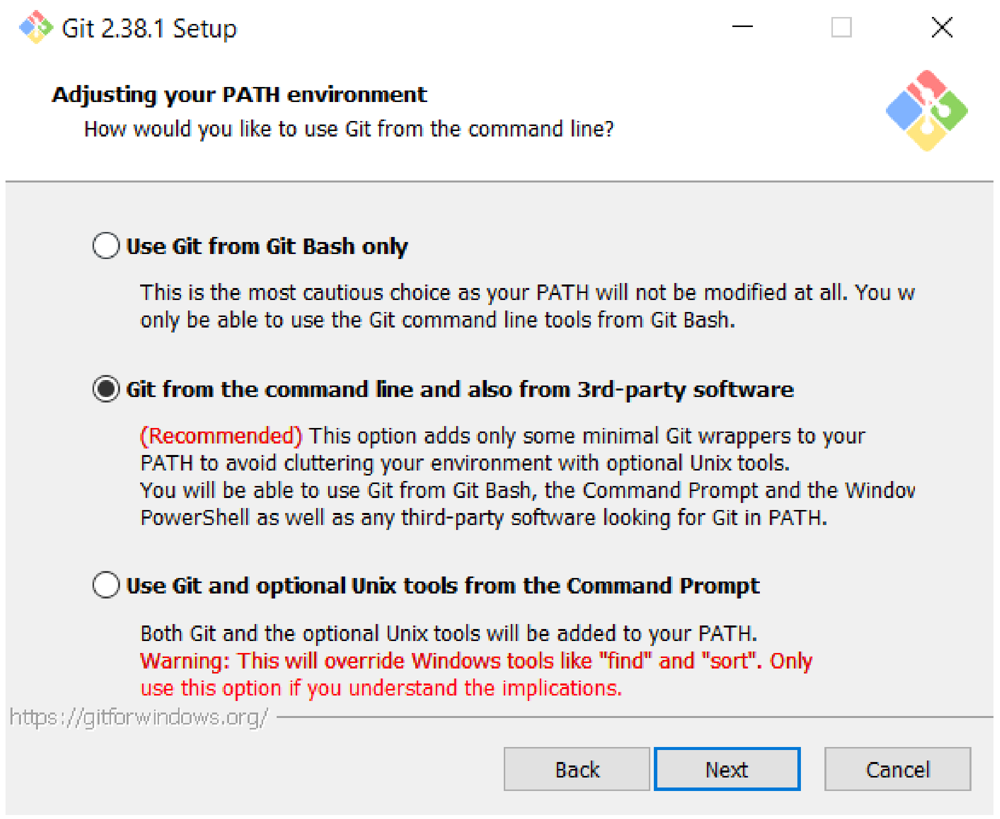
This setting does not affect existing repositories.

https://gitforwindows.org/

Download Git

Where to use Git:

- Keep your options open



Git 2.38.1 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 most cautious choice as your PATH will not be modified at all. You w
only be able to use the Git command line tools from Git Bash.

(●) **Git from the command line and also from 3rd-party software**

(Recommended) This option adds only some minimal Git wrappers to your
PATH to avoid cluttering your environment with optional Unix tools.
You will be able to use Git from Git Bash, the Command Prompt and the Window
PowerShell as well as any third-party software looking for Git in PATH.

( ) **Use Git and optional Unix tools from the Command Prompt**

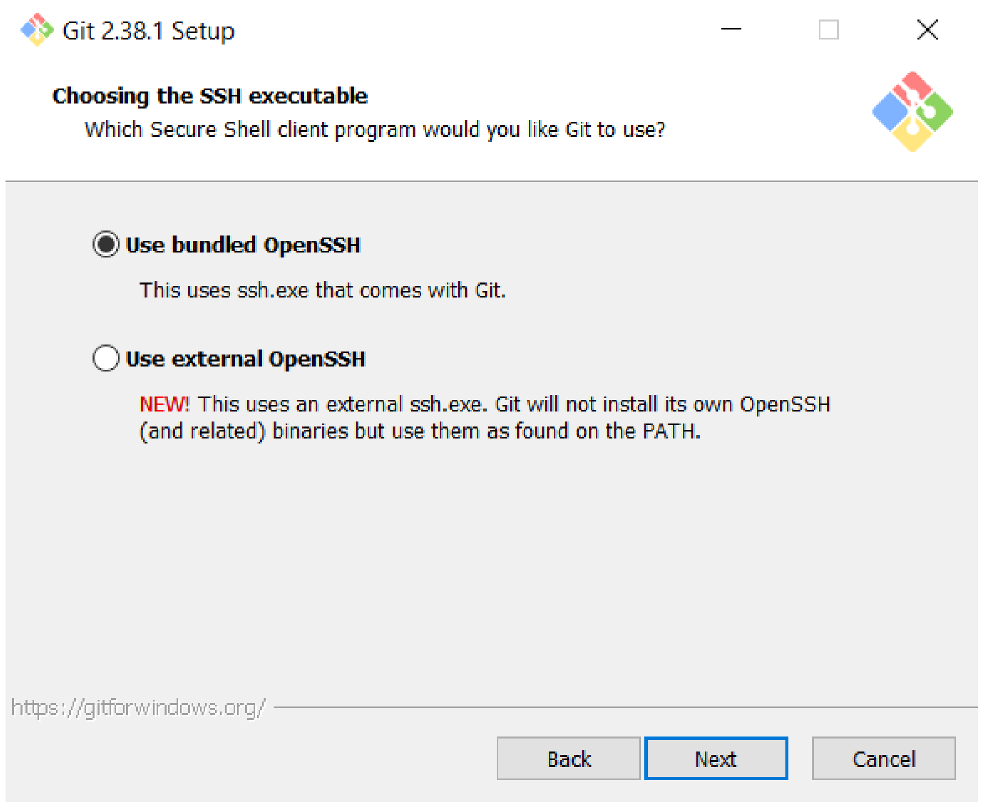
Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only
use this option if you understand the implications.

https://gitforwindows.org/

Download Git

OpenSSH

- Stick to the default



Git 2.38.1 Setup

Choosing the SSH executable

Which Secure Shell client program would you like Git to use?

[✓ **Use bundled OpenSSH**]

This uses ssh.exe that comes with Git.

Use external OpenSSH

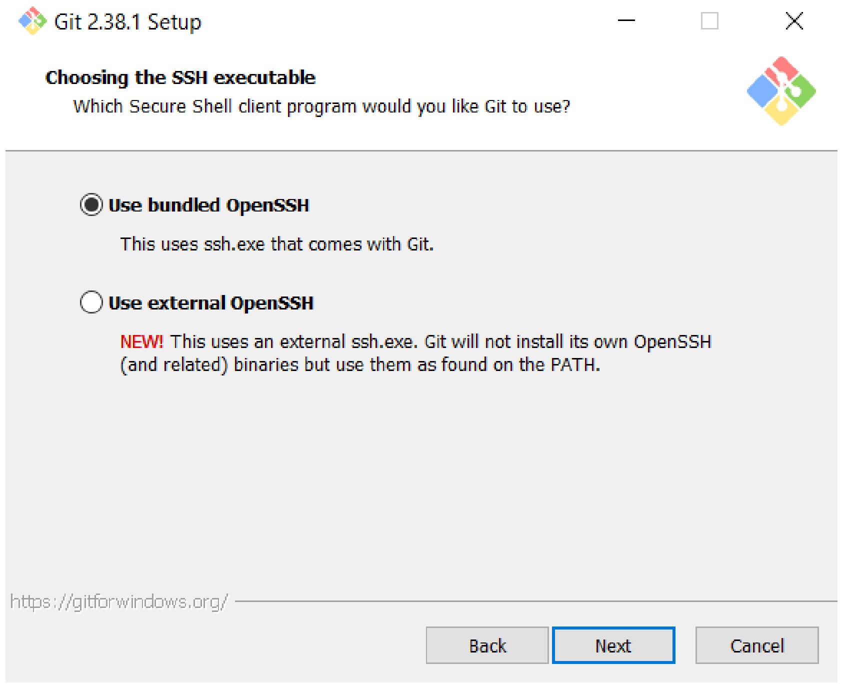
NEW! This uses an external ssh.exe. Git will not install its own OpenSSH (and related) binaries but use them as found on the PATH.

https://gitforwindows.org/

Download Git

OpenSSH and OpenSSL

- Stick to the defaults, tried and true



Git 2.38.1 Setup

Choosing the SSH executable

Which Secure Shell client program would you like Git to use?

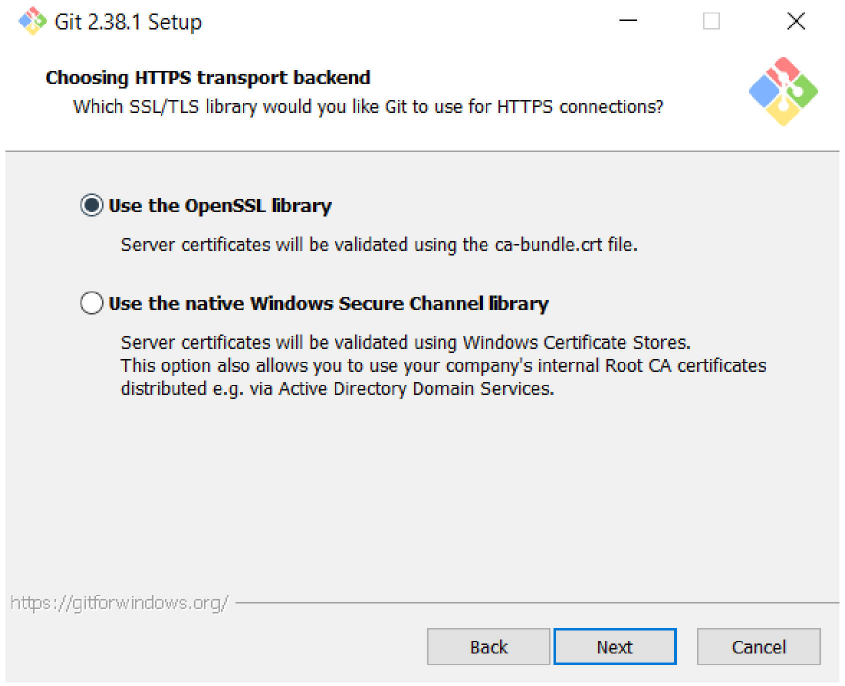
(● Use bundled OpenSSH) [✓]

This uses ssh.exe that comes with Git.

○ Use external OpenSSH

NEW! This uses an external ssh.exe. Git will not install its own OpenSSH (and related) binaries but use them as found on the PATH.

https://gitforwindows.org/



Git 2.38.1 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.

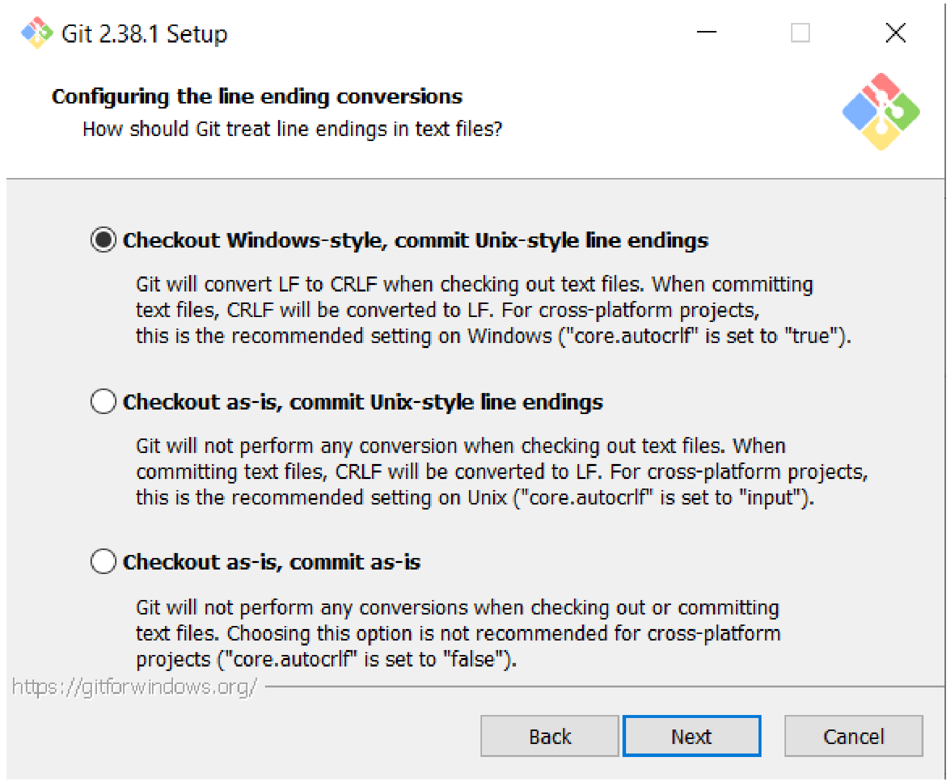
https://gitforwindows.org/

Download Git

Line ending conversions:

- Checkout Windows-style, commit Unix-style line endings

This will offer the most cross-platform compatibility



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").

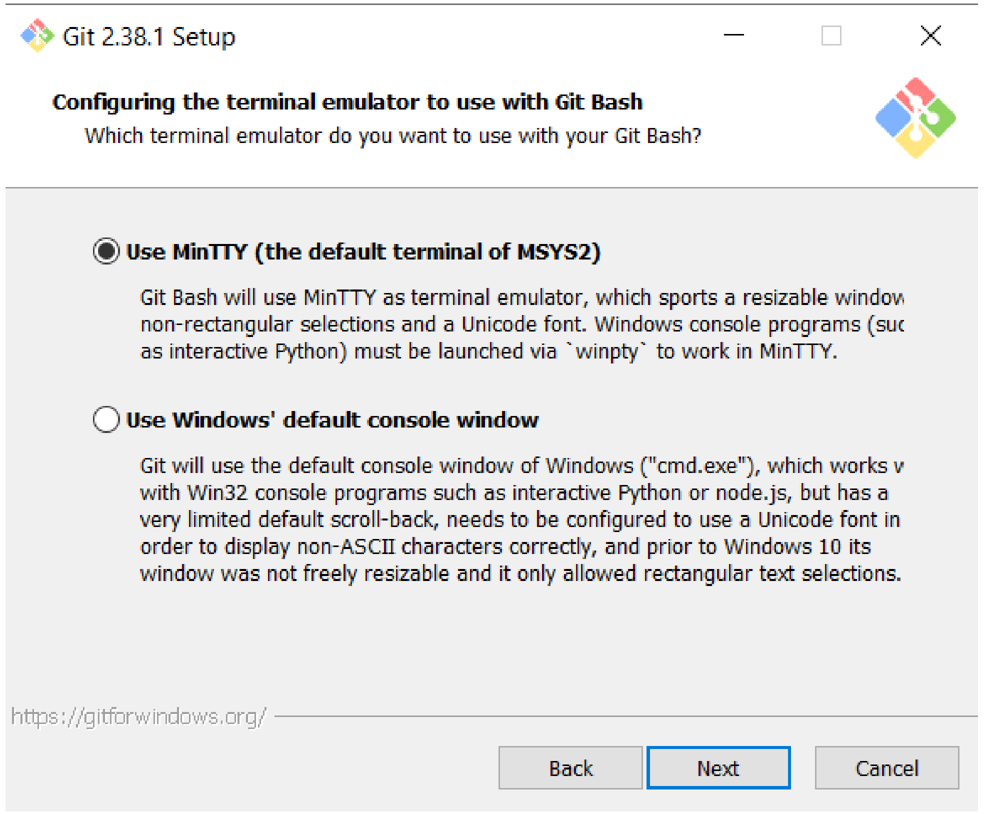
Checkout as-is, commit as-is

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").

Download Git

Terminal emulator:

- Linux is just better, choose MinTTY
- This doesn’t change any functionality, it just makes your command line window better



Git 2.38.1 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 (suc as interactive Python) must be launched via `winpty` to work in MinTTY.

Use Windows' default console window

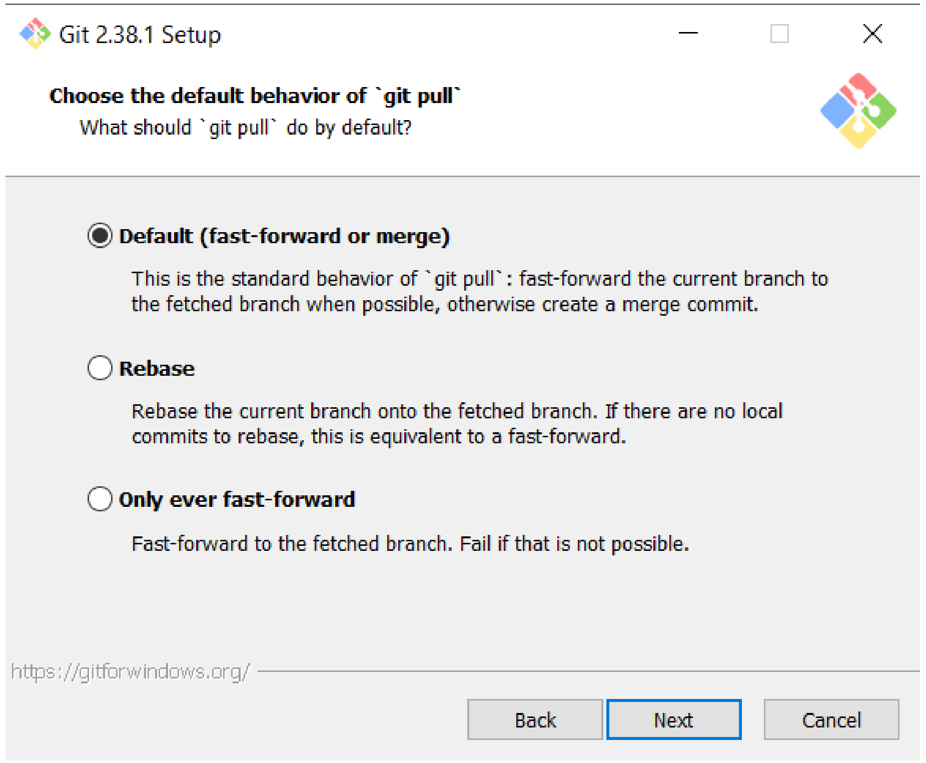
Git will use the default console window of Windows ("cmd.exe"), which works v with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows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

https://gitforwindows.org/

Download Git

Pull functionality:

- Keep the default, this provides the most functionality and speed



Choose the default behavior of `git pull`

What should `git pull` do by default?

[● **Default (fast-forward or merge)**]

This is the standard behavior of `git pull`: fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.

○ **Rebase**

Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.

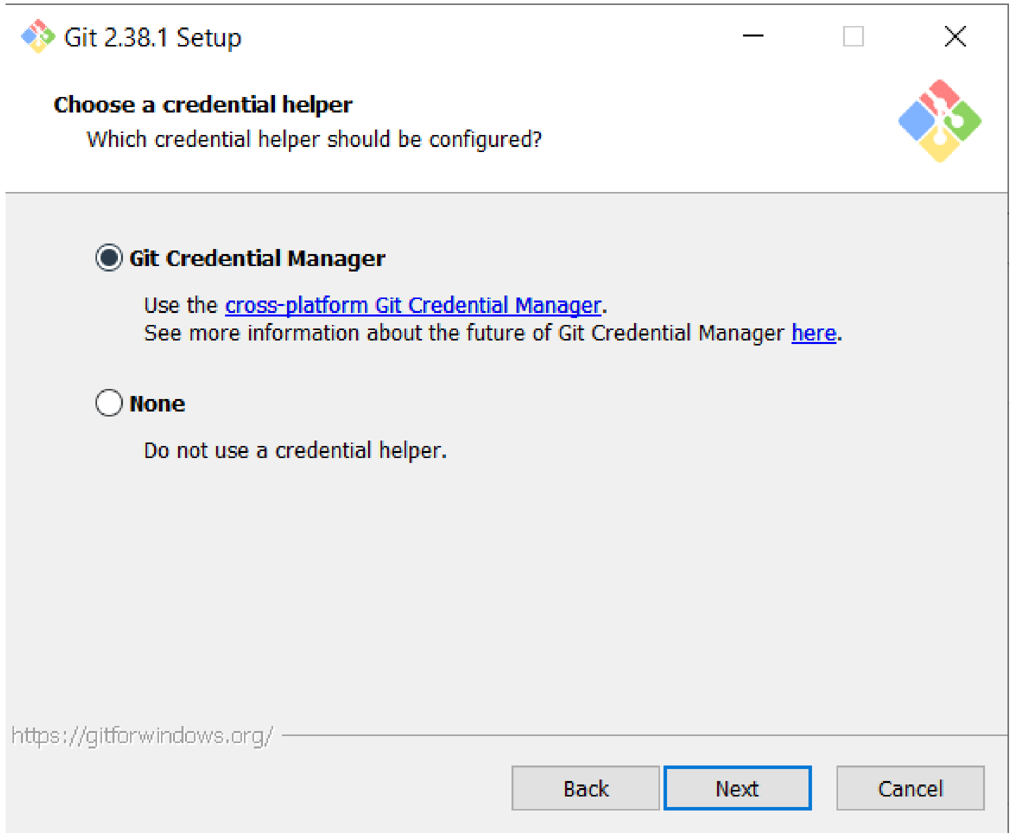
○ **Only ever fast-forward**

Fast-forward to the fetched branch. Fail if that is not possible.

Download Git

Credential Helper:

- Keep it cross-platform



Git 2.38.1 Setup

Choose a credential helper

Which credential helper should be configured?

[✓ **Git Credential Manager**]

Use the cross-platform Git Credential Manager.
See more information about the future of Git Credential Manager here.

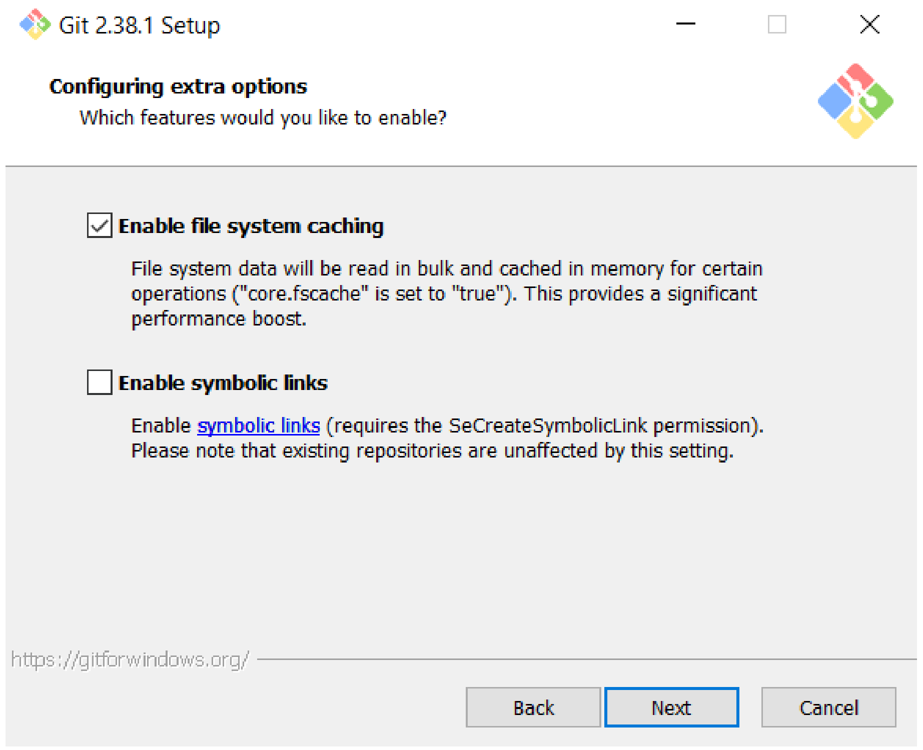
None

Do not use a credential helper.

https://gitforwindows.org/

Download Git

Caching is probably good, probably doesn't matter for our purposes. We don't need any other options.



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.

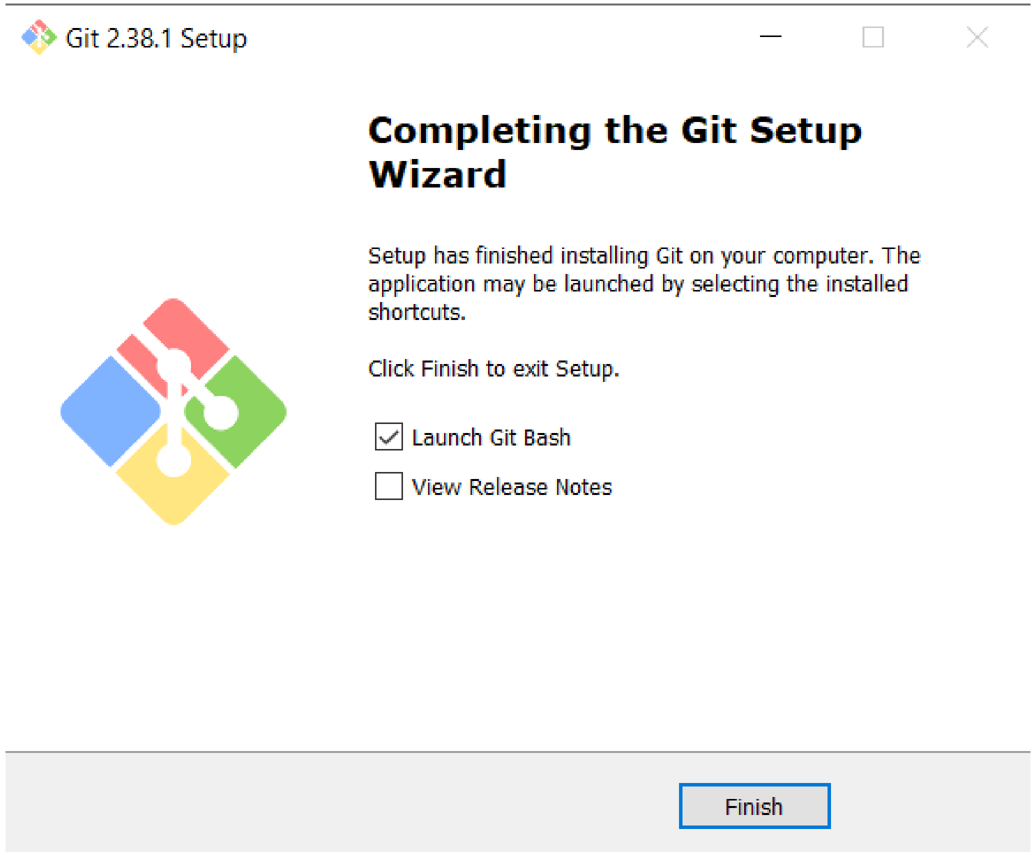
[ ] **Enable symbolic links**

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

https://gitforwindows.org/

Download Git

Done! Lets open Git Bash, a command line terminal.



Git 2.38.1 Setup

Completing the Git Setup Wizard

Setup has finished installing Git on your computer. The application may be launched by selecting the installed shortcuts.

Click Finish to exit Setup.

- [✓] Launch Git Bash
- [ ] View Release Notes

Setting up SSH

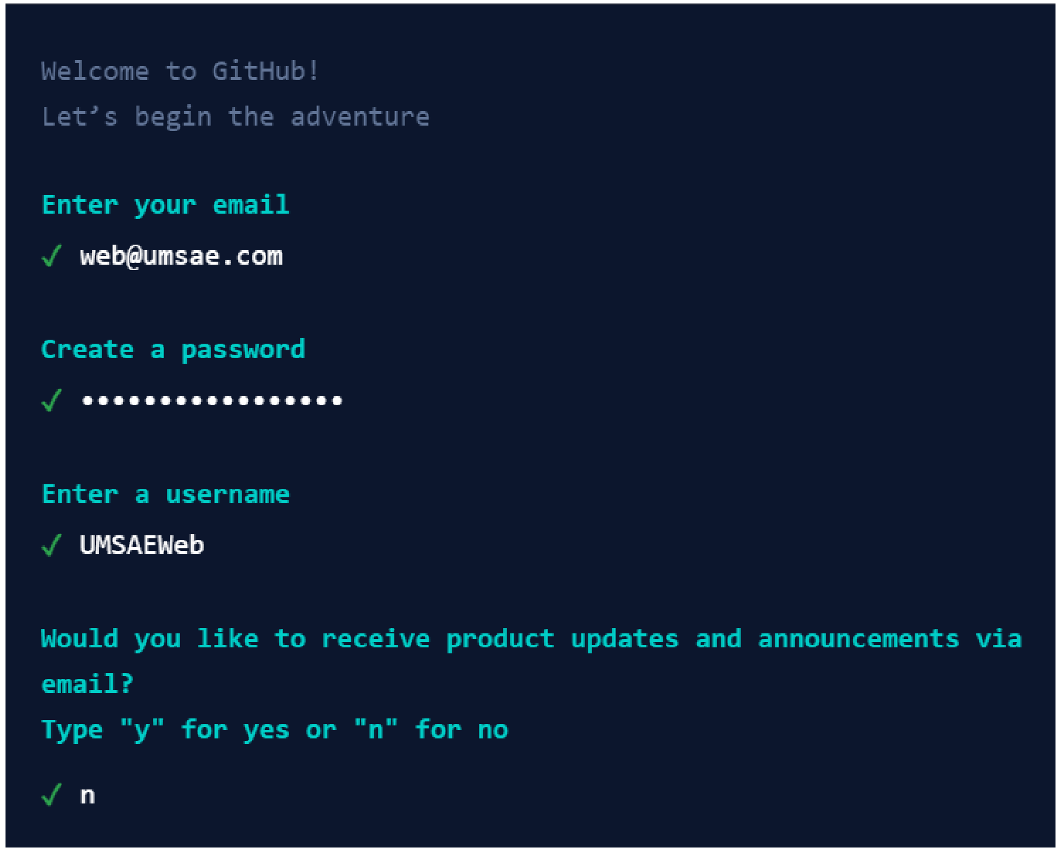
TELLING GITHUB THAT YOU ARE YOU

Setting up SSH

First off, you’ll need a GitHub account.

If you don’t have one, lets just make one now.

As a student, you can get GitHub pro for free. Let’s do that later and skip personalization for now.



```
Welcome to GitHub!
Let’s begin the adventure

Enter your email
✓ web@umsae.com

Create a password
✓ ••••••••••••••••

Enter a username
✓ UMSAEWeb

Would you like to receive product updates and announcements via
email?
Type "y" for yes or "n" for no
✓ n
```

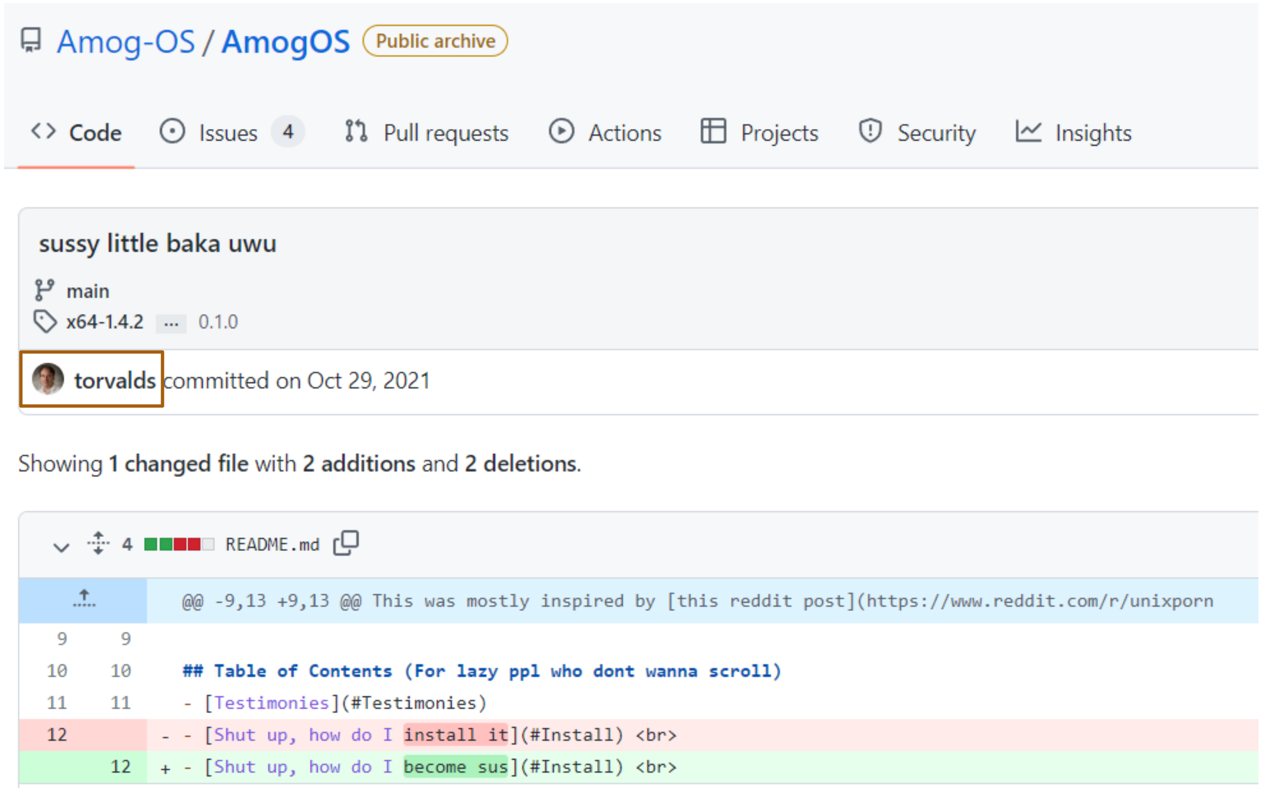
Setting up SSH

What is SSH?

From Wikipedia: The Secure Shell Protocol (SSH) is a cryptographic network protocol for operating network services securely over an unsecured network.

Secure cryptographic data transmission that makes sure no one is making commits where they aren't supposed to be.

(It can be spoofed though…)



sussy little baka uwu

main
x64-1.4.2 … 0.1.0

torvalds committed on Oct 29, 2021

Showing 1 changed file with 2 additions and 2 deletions.

4 README.md

```
@@ -9,13 +9,13 @@ This was mostly inspired by [this reddit post](https://www.reddit.com/r/unixporn

   ## Table of Contents (For lazy ppl who dont wanna scroll)
   - [Testimonies](#Testimonies)
-  - [Shut up, how do I install it](#Install) <br>
+  - [Shut up, how do I become sus](#Install) <br>
```

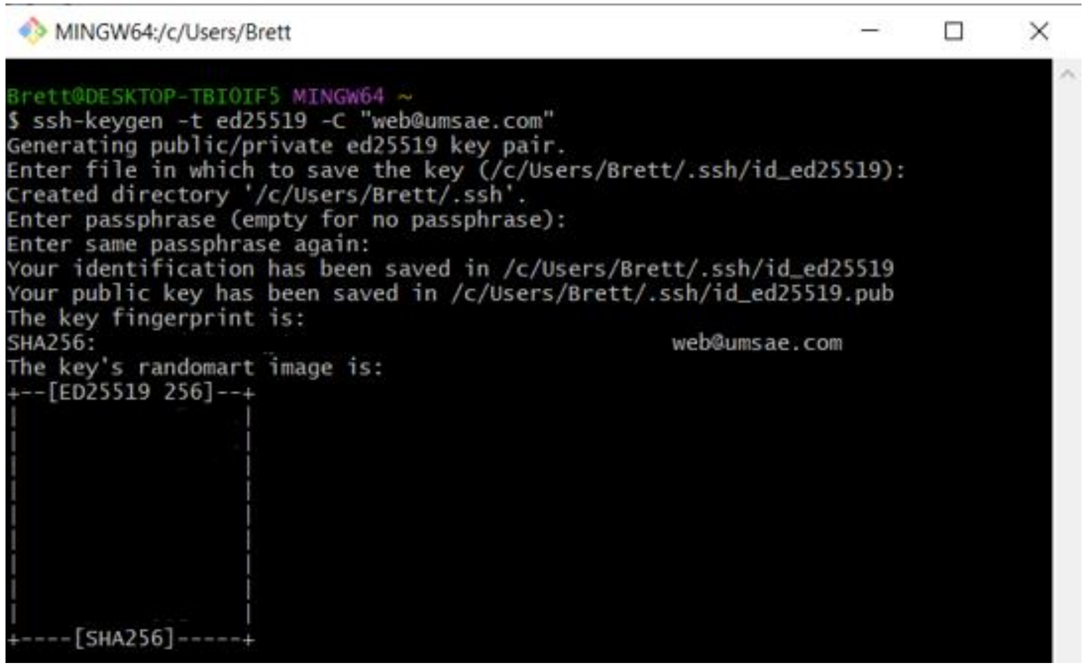
Setting up SSH

Generate a new SSH key:

In git bash (or any command line),

The command:
ssh-keygen –t ed25519 –C "<your@email.com>"

When prompted for directory and password, just leave everything empty.



MINGW64:/c/Users/Brett

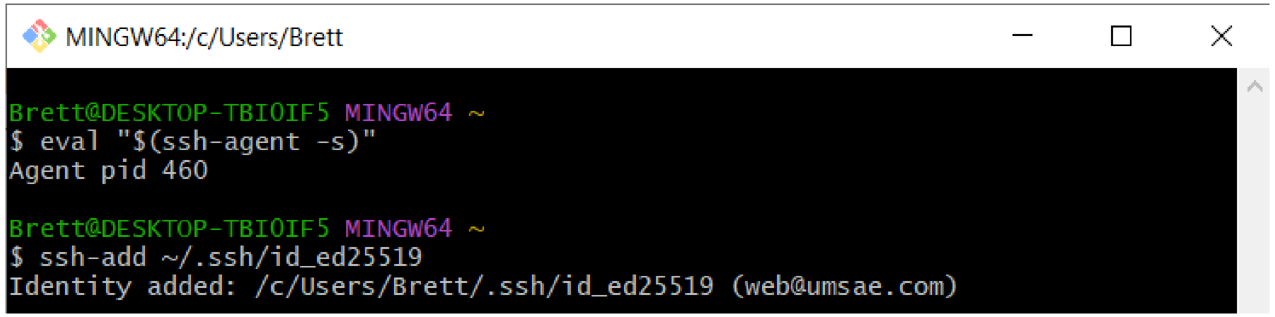
```
Brett@DESKTOP-TBIOIF5 MINGW64 ~
$ ssh-keygen -t ed25519 -C "web@umsae.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/c/Users/Brett/.ssh/id_ed25519):
Created directory '/c/Users/Brett/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/Brett/.ssh/id_ed25519
Your public key has been saved in /c/Users/Brett/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:                                            web@umsae.com
The key's randomart image is:
+--[ED25519 256]--+
|                 |
|                 |
|                 |
|                 |
|                 |
|                 |
|                 |
|                 |
|                 |
+----[SHA256]-----+
```

Setting up SSH

Add ssh key to ssh client:

Start the client:
eval “$(ssh-agent –s)”

Add your new ssh key:
ssh-add ~/.ssh/id_ed25519



```
Brett@DESKTOP-TBIOIF5 MINGW64 ~
$ eval "$(ssh-agent -s)"
Agent pid 460

Brett@DESKTOP-TBIOIF5 MINGW64 ~
$ ssh-add ~/.ssh/id_ed25519
Identity added: /c/Users/Brett/.ssh/id_ed25519 (web@umsae.com)
```

Setting up SSH

Copy the key:

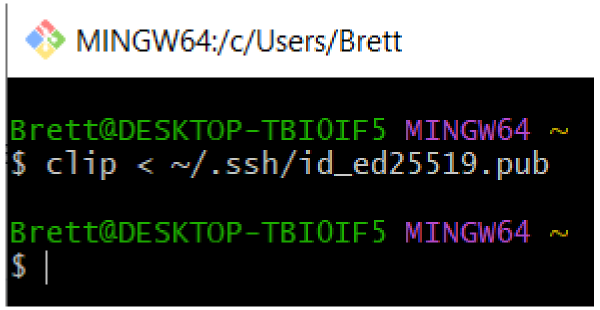
clip < ~/.ssh/id_ed25519.pub

Go to github -> settings
-> SSH and GPG keys -> New SSH Key

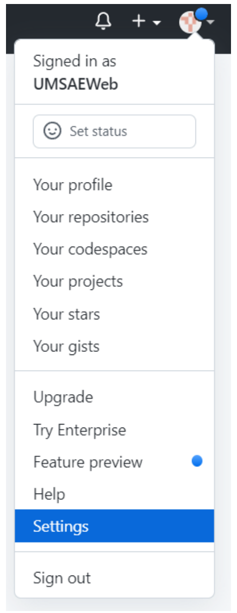
Give a descriptive name and paste your key

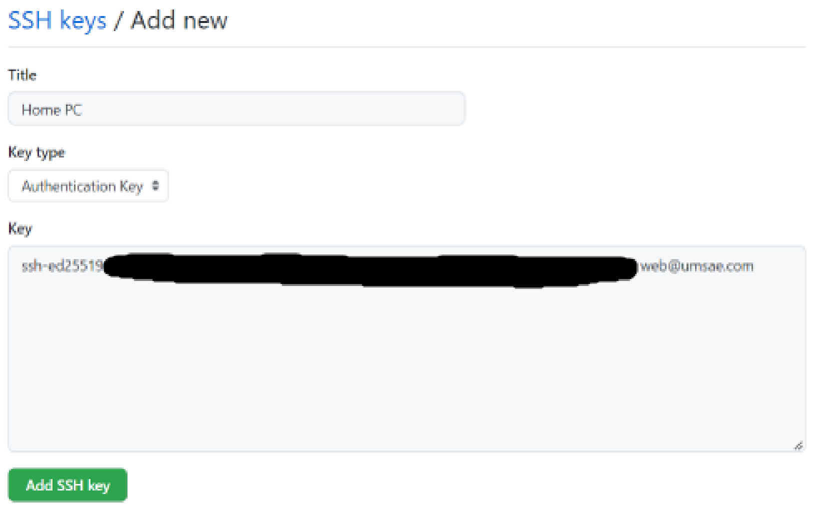
Now you can interface with GitHub from the computer you are working on right now.

You have to repeat this process for any other computer



```
MINGW64:/c/Users/Brett

Brett@DESKTOP-TBIOIF5 MINGW64 ~
$ clip < ~/.ssh/id_ed25519.pub

Brett@DESKTOP-TBIOIF5 MINGW64 ~
$ |
```





SSH keys / Add new

Title

Home PC

Key type

Authentication Key

Key

ssh-ed25519[redacted]web@umsae.com

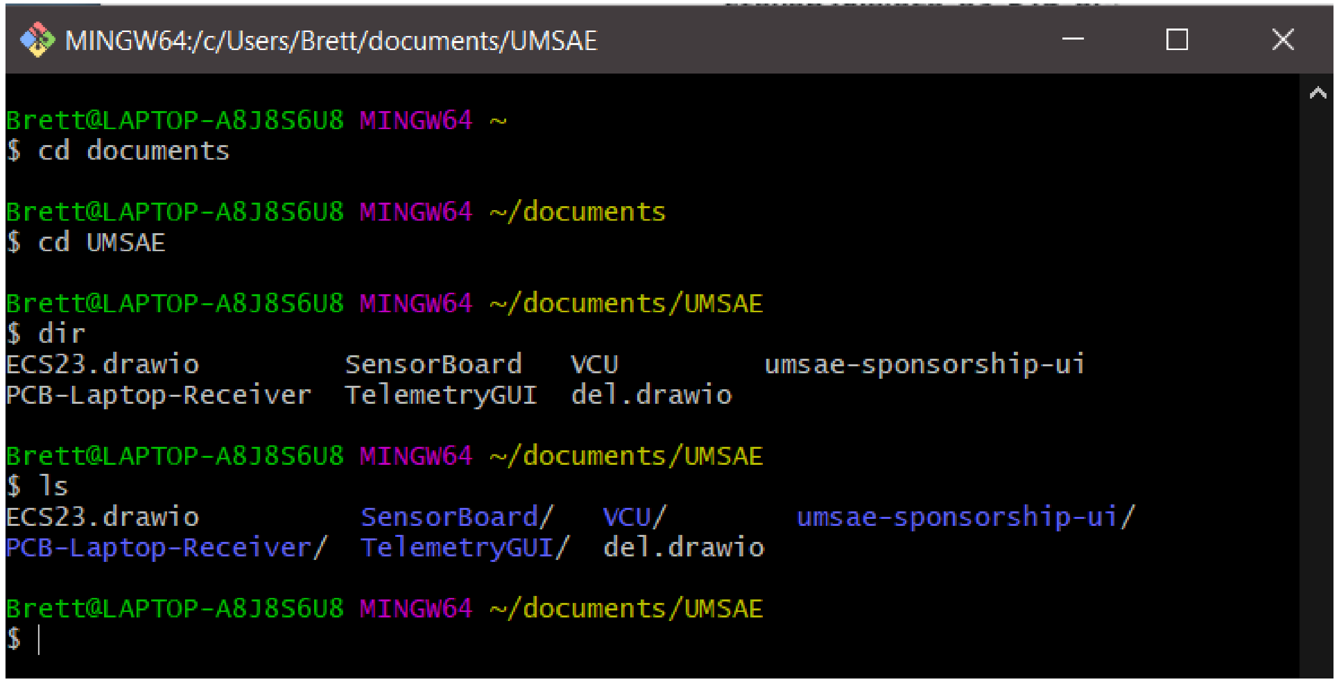
The cool way that cool people use

How 2 git

COMMAND LINE EDITION

How 2 git (command line 😎)

1. Navigate to your directory
   - ls shows files in current directory but colors other directories and doesn’t color files
   - cd sets current directory to a new directory

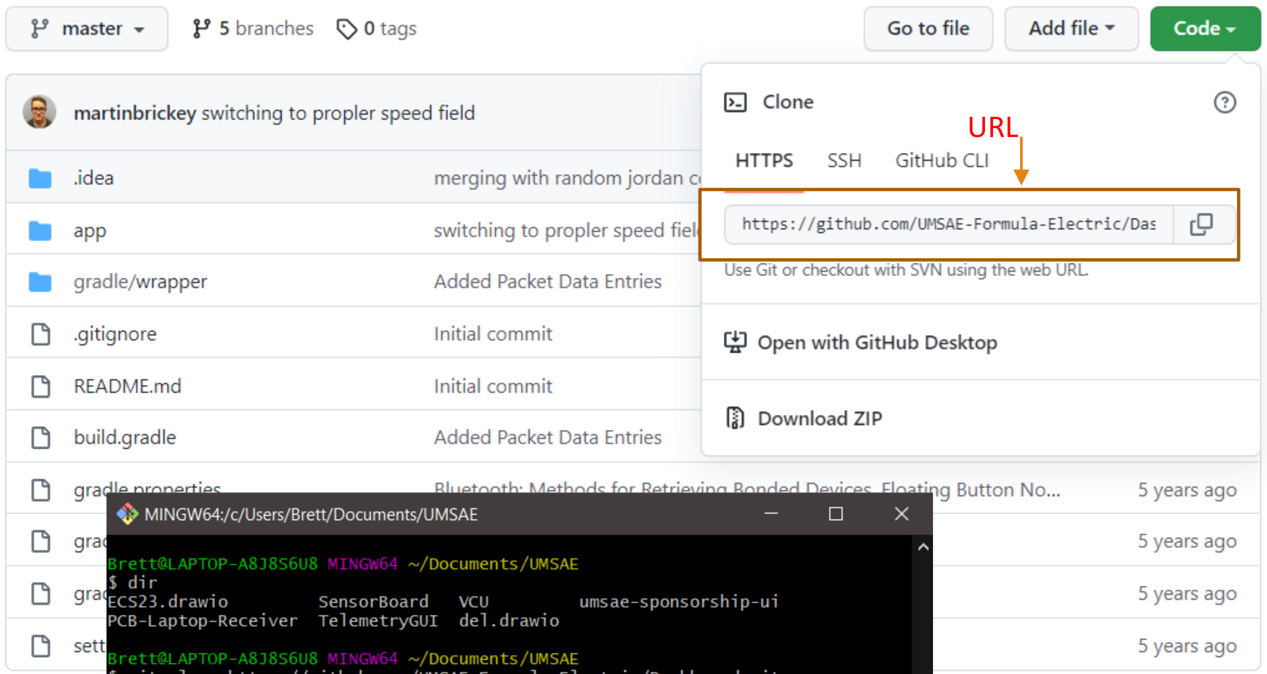


```
MINGW64:/c/Users/Brett/documents/UMSAE

Brett@LAPTOP-A8J8S6U8 MINGW64 ~
$ cd documents

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/documents
$ cd UMSAE

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/documents/UMSAE
$ dir
ECS23.drawio         SensorBoard   VCU          umsae-sponsorship-ui
PCB-Laptop-Receiver  TelemetryGUI  del.drawio

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/documents/UMSAE
$ ls
ECS23.drawio          SensorBoard/   VCU/         umsae-sponsorship-ui/
PCB-Laptop-Receiver/  TelemetryGUI/  del.drawio

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/documents/UMSAE
$
```

This isn’t git just yet, this is regular cmd line stuff

How 2 git (command line 😎)

2. Clone the repo from github
   - Go to the github page where the repo is hosted
   - Get the url
   - Run “git clone <url>”


master · 5 branches · 0 tags · Go to file · Add file · Code

martinbrickey switching to propler speed field

.idea	merging with random jordan c
app	switching to propler speed fiel
gradle/wrapper	Added Packet Data Entries
.gitignore	Initial commit
README.md	Initial commit
build.gradle	Added Packet Data Entries
gradle.properties	Bluetooth: Methods for Retrieving Bonded Devices, Floating Button No... 5 years ago
grad	5 years ago
grad	5 years ago
sett	5 years ago

Clone

HTTPS · SSH · GitHub CLI

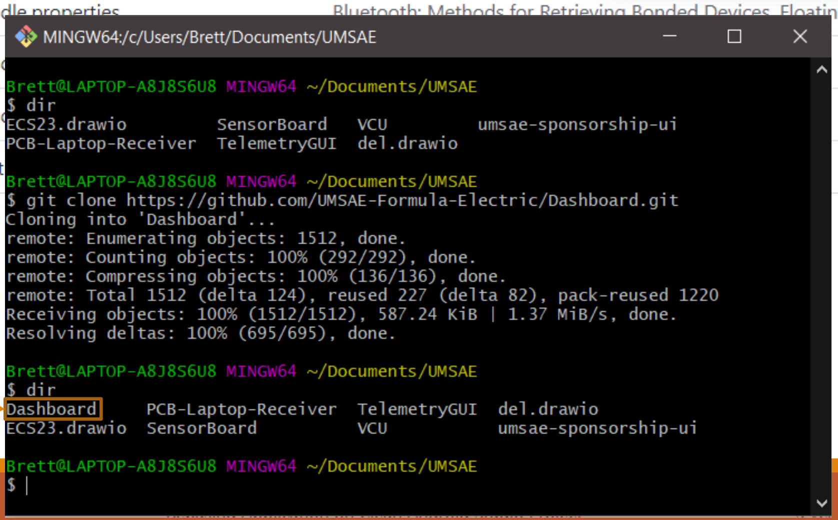
URL

https://github.com/UMSAE-Formula-Electric/Das

Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Download ZIP


MINGW64:/c/Users/Brett/Documents/UMSAE

```
Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE
$ dir
ECS23.drawio         SensorBoard   VCU         umsae-sponsorship-ui
PCB-Laptop-Receiver  TelemetryGUI  del.drawio

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE
$ git clone https://github.com/UMSAE-Formula-Electric/Dashboard.git
Cloning into 'Dashboard'...
remote: Enumerating objects: 1512, done.
remote: Counting objects: 100% (292/292), done.
remote: Compressing objects: 100% (136/136), done.
remote: Total 1512 (delta 124), reused 227 (delta 82), pack-reused 1220
Receiving objects: 100% (1512/1512), 587.24 KiB | 1.37 MiB/s, done.
Resolving deltas: 100% (695/695), done.

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE
$ dir
Dashboard     PCB-Laptop-Receiver  TelemetryGUI  del.drawio
ECS23.drawio  SensorBoard          VCU           umsae-sponsorship-ui

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE
$ |
```

We now have Dashboard locally

How 2 git (command line 😎)

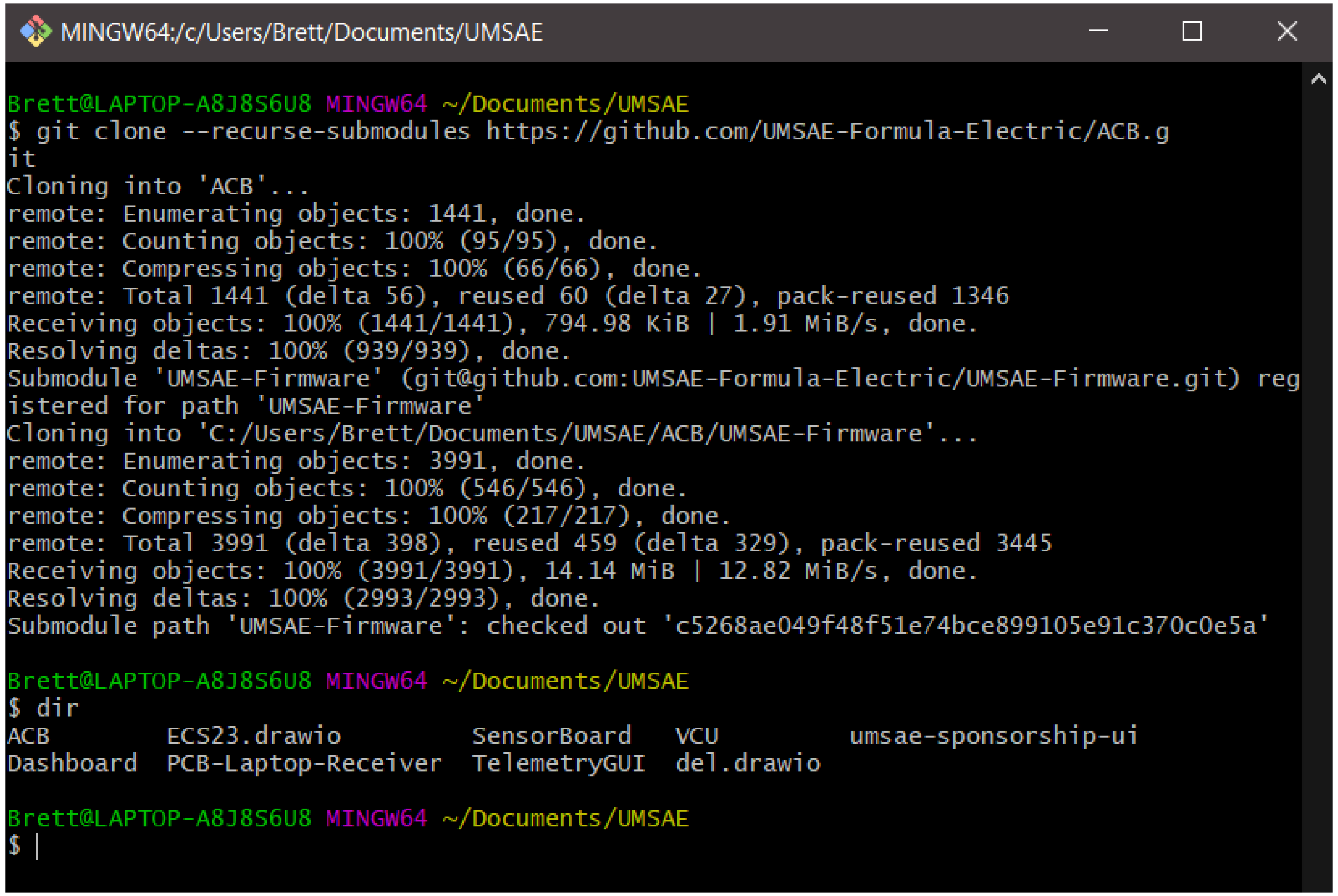
If your repository has submodules (ACB and VCU), use the option --recurse-submodules. This allows you to treat your submodules as separate repos.

“git clone --recurse-submodules <URL>”

Clones ACB

Clones UMSAE-Firmware submodule

Checks out submodule to the appropriate commit



```
Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE
$ git clone --recurse-submodules https://github.com/UMSAE-Formula-Electric/ACB.g
it
Cloning into 'ACB'...
remote: Enumerating objects: 1441, done.
remote: Counting objects: 100% (95/95), done.
remote: Compressing objects: 100% (66/66), done.
remote: Total 1441 (delta 56), reused 60 (delta 27), pack-reused 1346
Receiving objects: 100% (1441/1441), 794.98 KiB | 1.91 MiB/s, done.
Resolving deltas: 100% (939/939), done.
Submodule 'UMSAE-Firmware' (git@github.com:UMSAE-Formula-Electric/UMSAE-Firmware.git) reg
istered for path 'UMSAE-Firmware'
Cloning into 'C:/Users/Brett/Documents/UMSAE/ACB/UMSAE-Firmware'...
remote: Enumerating objects: 3991, done.
remote: Counting objects: 100% (546/546), done.
remote: Compressing objects: 100% (217/217), done.
remote: Total 3991 (delta 398), reused 459 (delta 329), pack-reused 3445
Receiving objects: 100% (3991/3991), 14.14 MiB | 12.82 MiB/s, done.
Resolving deltas: 100% (2993/2993), done.
Submodule path 'UMSAE-Firmware': checked out 'c5268ae049f48f51e74bce899105e91c370c0e5a'

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE
$ dir
ACB        ECS23.drawio         SensorBoard    VCU          umsae-sponsorship-ui
Dashboard  PCB-Laptop-Receiver  TelemetryGUI   del.drawio

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE
$ |
```

We are trying to phase out submodules because they are a pain to use. If you are reading this in 2024 or later and you’re still using submodules, then I am very disappointed…

How 2 git (command line 😎)

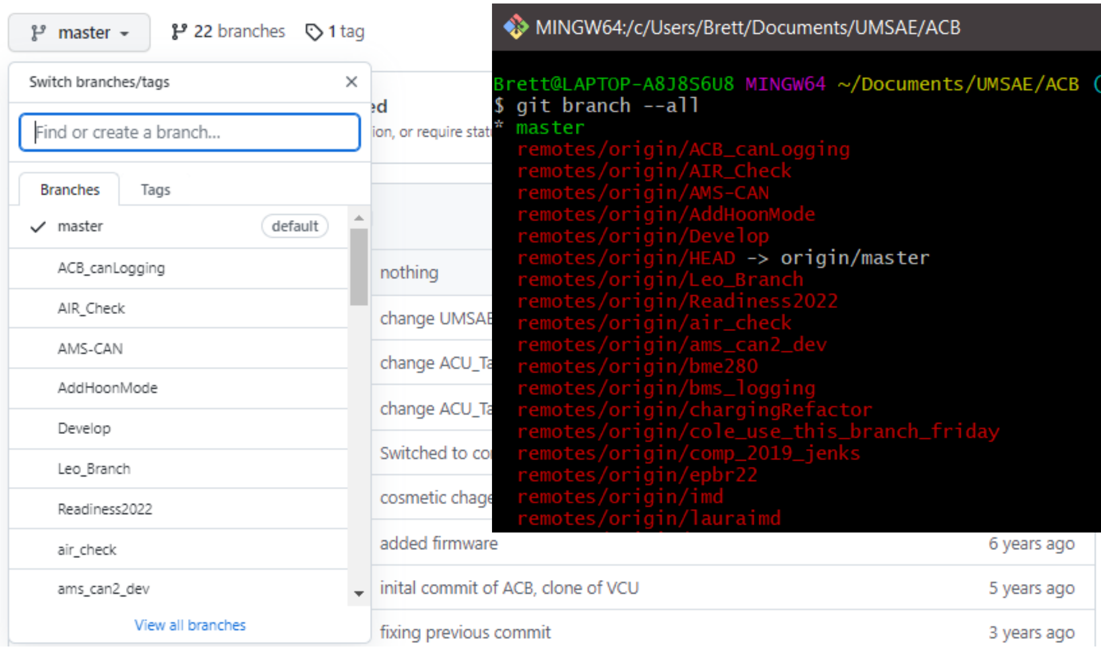
3. Checkout the branch you want to work on
   - Look at github to get the current branches or call “git branch --all”
   - Call “git checkout <branch>”

<u>This will change your local files to match the ones stored on the remote.</u>

- The repo is directly connected to your file when you git clone. Which means you have access to every branch and commit with a simple checkout call.

- **This is why git is useful**

If you want to create a new branch, use the –b option: “git checkout –b <new branch name>”



master · 22 branches · 1 tag

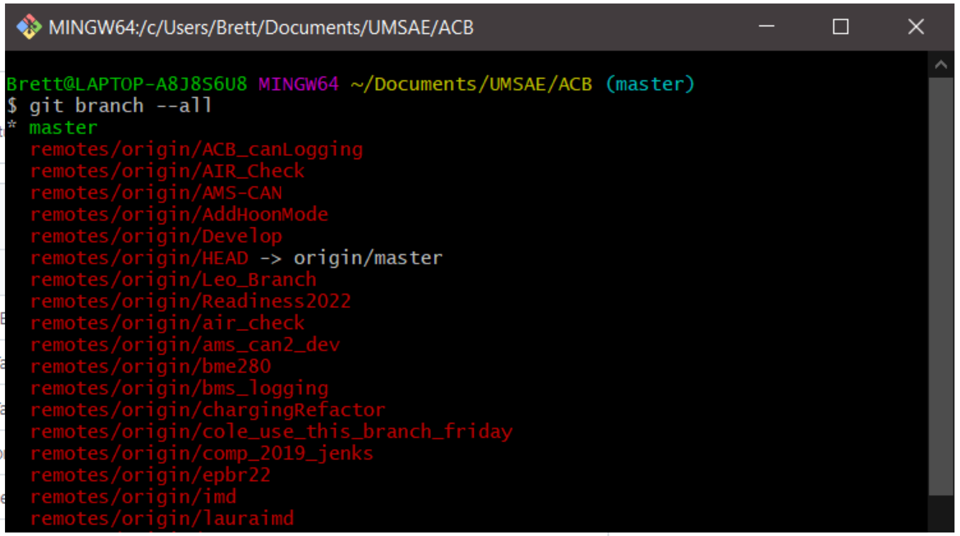
Switch branches/tags

Find or create a branch...

Branches | Tags

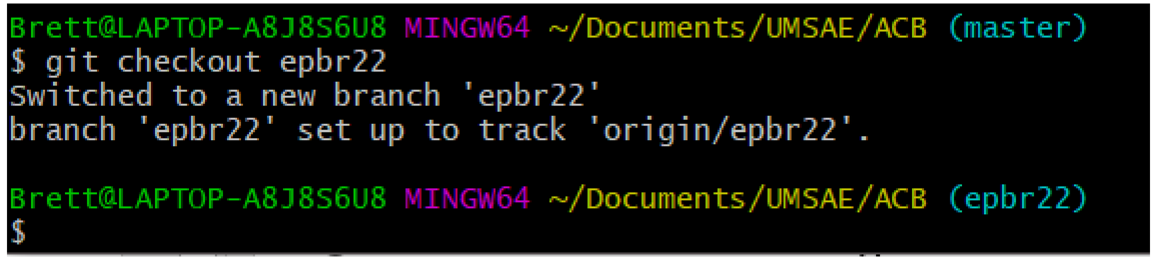
```
✓ master   default
ACB_canLogging
AIR_Check
AMS-CAN
AddHoonMode
Develop
Leo_Branch
Readiness2022
air_check
ams_can2_dev
View all branches
```

```
nothing
change UMSAE
change ACU_Ta
change ACU_Ta
Switched to co
cosmetic chage
added firmware                          6 years ago
inital commit of ACB, clone of VCU      5 years ago
fixing previous commit                  3 years ago
```



MINGW64:/c/Users/Brett/Documents/UMSAE/ACB

```
Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/ACB (master)
$ git branch --all
* master
  remotes/origin/ACB_canLogging
  remotes/origin/AIR_Check
  remotes/origin/AMS-CAN
  remotes/origin/AddHoonMode
  remotes/origin/Develop
  remotes/origin/HEAD -> origin/master
  remotes/origin/Leo_Branch
  remotes/origin/Readiness2022
  remotes/origin/air_check
  remotes/origin/ams_can2_dev
  remotes/origin/bme280
  remotes/origin/bms_logging
  remotes/origin/chargingRefactor
  remotes/origin/cole_use_this_branch_friday
  remotes/origin/comp_2019_jenks
  remotes/origin/epbr22
  remotes/origin/imd
  remotes/origin/lauraimd
```



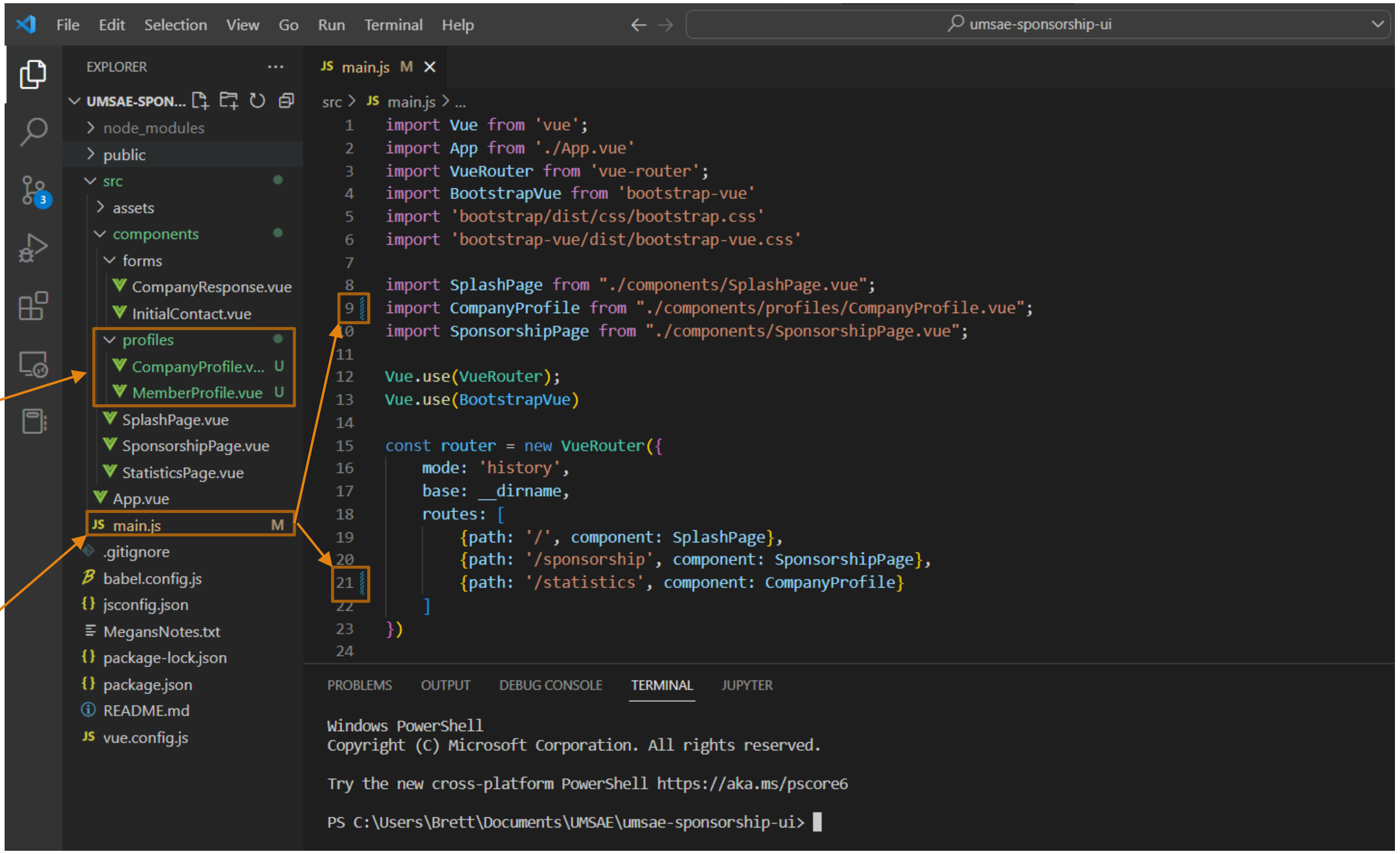
```
Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/ACB (master)
$ git checkout epbr22
Switched to a new branch 'epbr22'
branch 'epbr22' set up to track 'origin/epbr22'.

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/ACB (epbr22)
$
```

How 2 git (command line 😎)

4. Make some changes!
   - In this example, I have modified main.js and added a folder containing two profile components.
   - Most IDEs will show you what changes you have made since the last commit

U means "untracked": Git doesn't see a history for this file (It's new)

M means "modified": Git can tell you have changed this file



```
import Vue from 'vue';
import App from './App.vue'
import VueRouter from 'vue-router';
import BootstrapVue from 'bootstrap-vue'
import 'bootstrap/dist/css/bootstrap.css'
import 'bootstrap-vue/dist/bootstrap-vue.css'

import SplashPage from "./components/SplashPage.vue";
import CompanyProfile from "./components/profiles/CompanyProfile.vue";
import SponsorshipPage from "./components/SponsorshipPage.vue";

Vue.use(VueRouter);
Vue.use(BootstrapVue)

const router = new VueRouter({
    mode: 'history',
    base: __dirname,
    routes: [
        {path: '/', component: SplashPage},
        {path: '/sponsorship', component: SponsorshipPage},
        {path: '/statistics', component: CompanyProfile}
    ]
})

```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Brett\Documents\UMSAE\umsae-sponsorship-ui> 
```

How 2 git (command line 😎)

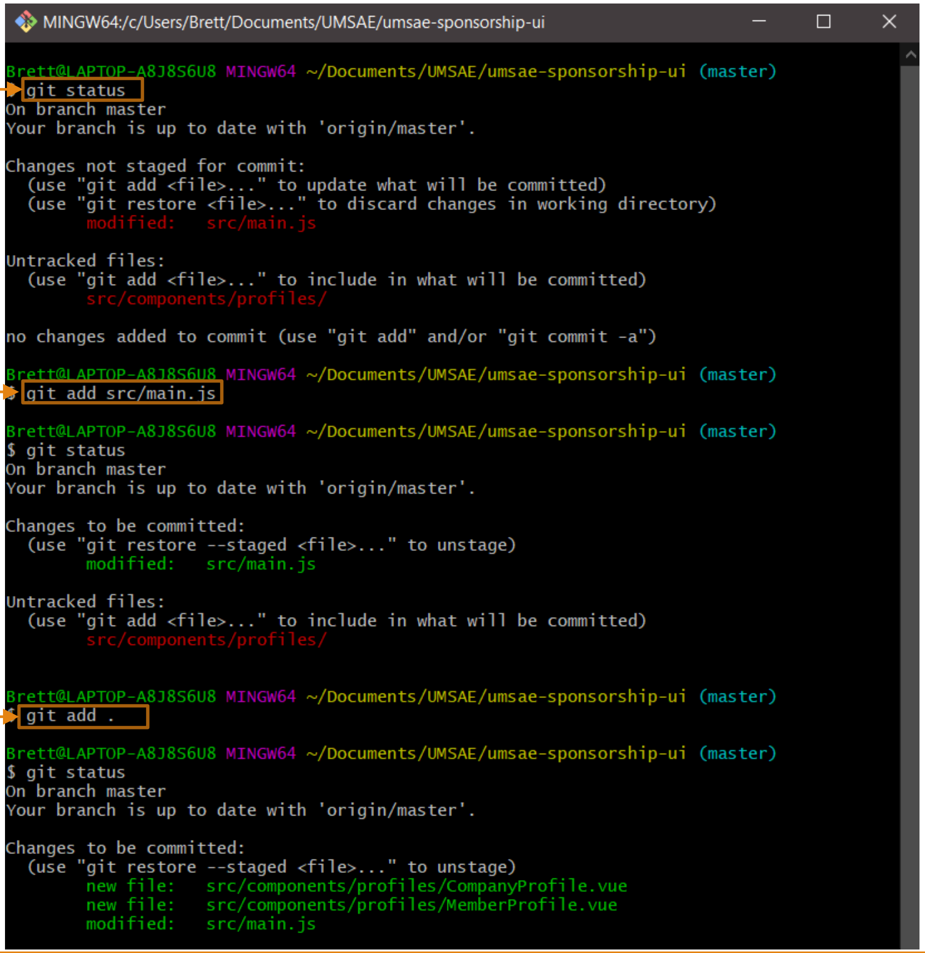
5. Stage your changes for commit
   - Before you can commit your changes, you have to tell git what changes you actually want to commit
   - This is so you don't start adding unwanted files to the remote repo that have been generated by your IDE

   - "git add <file(s)>"
     - "git add *" will add all files & folders. 99% of the time this is what you should use

git status to see changes

Add main.js change

Add all changes (careful)

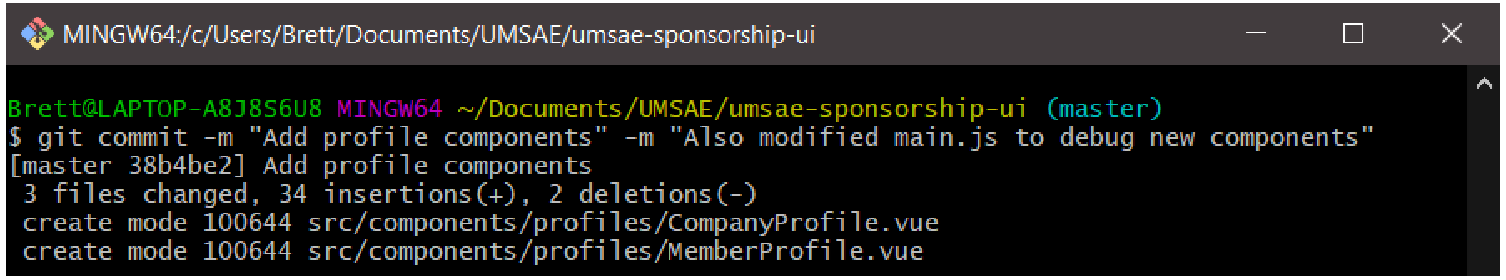


MINGW64:/c/Users/Brett/Documents/UMSAE/umsae-sponsorship-ui

```
Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/umsae-sponsorship-ui (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   src/main.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        src/components/profiles/

no changes added to commit (use "git add" and/or "git commit -a")

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/umsae-sponsorship-ui (master)
$ git add src/main.js

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/umsae-sponsorship-ui (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   src/main.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        src/components/profiles/


Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/umsae-sponsorship-ui (master)
$ git add .

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/umsae-sponsorship-ui (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   src/components/profiles/CompanyProfile.vue
        new file:   src/components/profiles/MemberProfile.vue
        modified:   src/main.js
```

How 2 git (command line 😎)

6. Commit your changes
   - Add a descriptive but concise summary
   - If you want to add more detail you can add an optional description



MINGW64:/c/Users/Brett/Documents/UMSAE/umsae-sponsorship-ui

```
Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/umsae-sponsorship-ui (master)
$ git commit -m "Add profile components" -m "Also modified main.js to debug new components"
[master 38b4be2] Add profile components
 3 files changed, 34 insertions(+), 2 deletions(-)
 create mode 100644 src/components/profiles/CompanyProfile.vue
 create mode 100644 src/components/profiles/MemberProfile.vue
```

THIS IS USUALLY THE PART WHERE YOU CHECK IF YOU ARE ON THE RIGHT BRANCH
You should probably not be committing to master/main, I'm only doing this to a new repo that hasn't been worked on by anyone else yet.

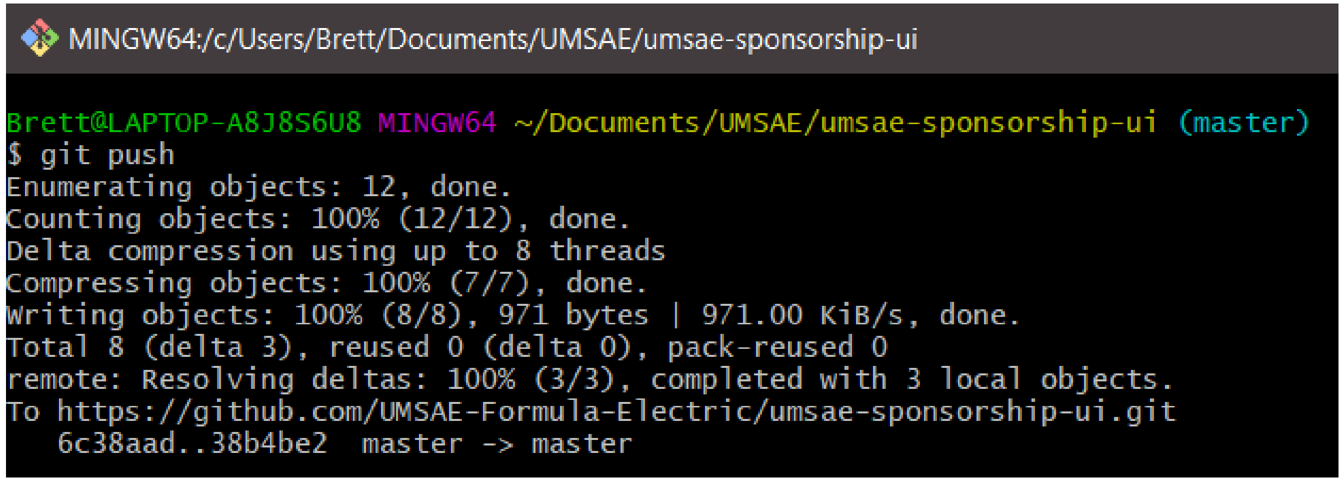
git commit –m "<description>" –m "<subtitle>"

How 2 git (command line 😎)

7. Push your new commits to the remote repo
   - The commit that you just made is only stored on your local repo, you need to “push” your repo up to the GitHub servers

   “git push”

   - If git is being stupid and tells you to use “git push --set-upstream origin <branch name>” just use that. I don’t know why that happens

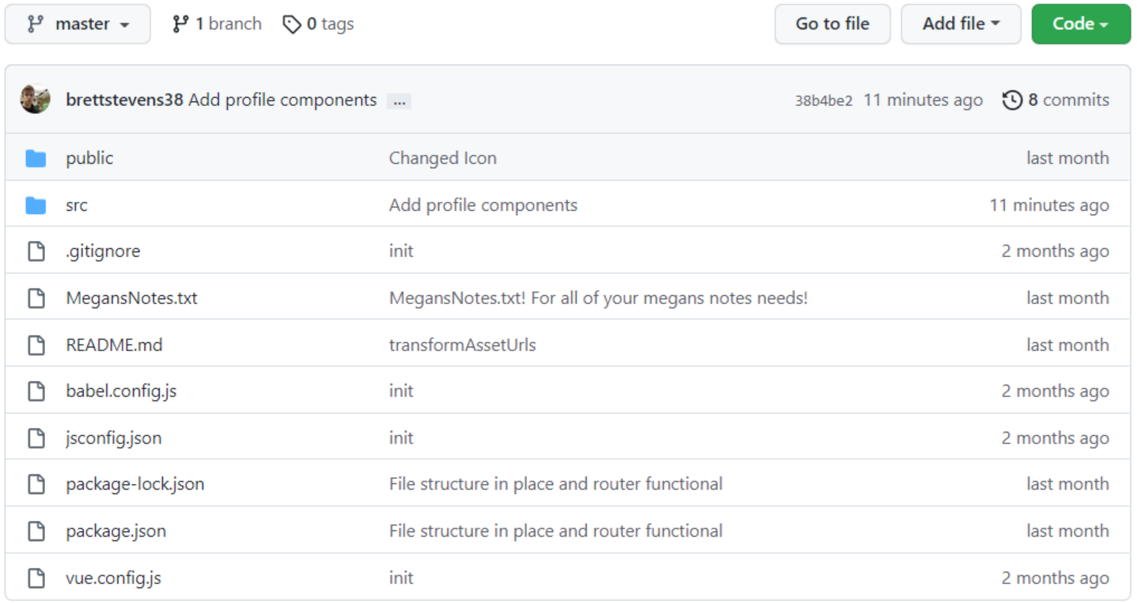


MINGW64:/c/Users/Brett/Documents/UMSAE/umsae-sponsorship-ui

```
Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/umsae-sponsorship-ui (master)
$ git push
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 8 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 971 bytes | 971.00 KiB/s, done.
Total 8 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/UMSAE-Formula-Electric/umsae-sponsorship-ui.git
   6c38aad..38b4be2  master -> master
```

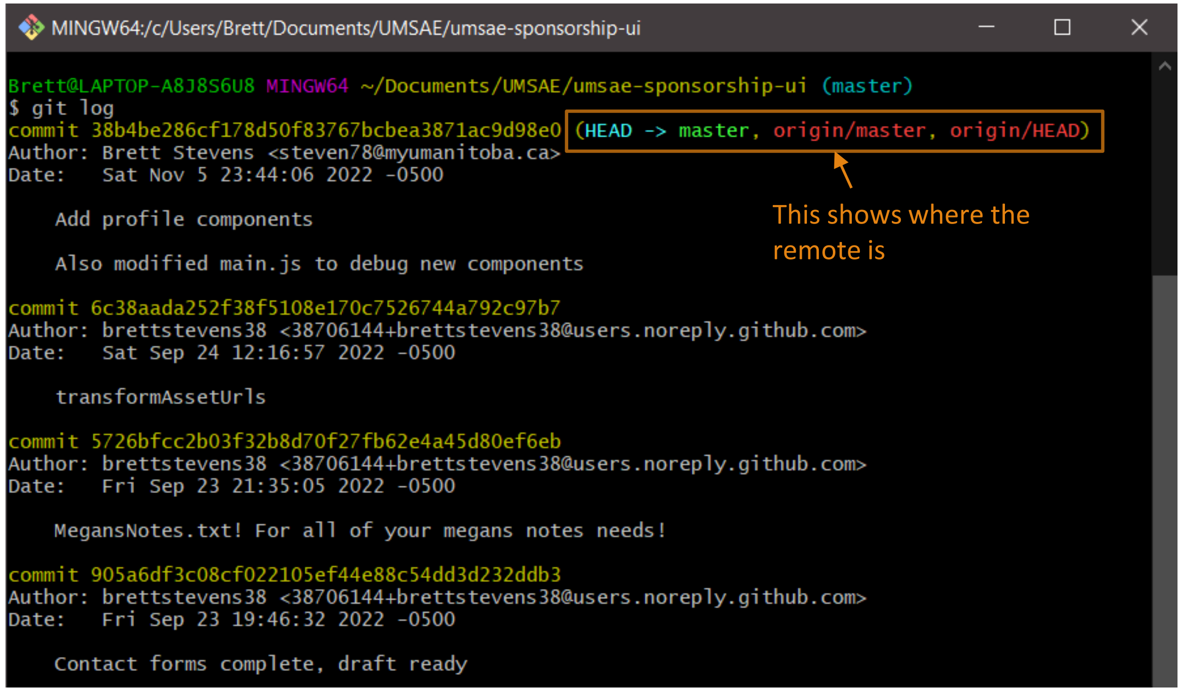
How 2 git (command line 😎)

8. Confirm your changes were pushed successfully
   - Use either "git log" or go to GitHub



brettstevens38 Add profile components ... 38b4be2 11 minutes ago 8 commits

public	Changed Icon	last month
src	Add profile components	11 minutes ago
.gitignore	init	2 months ago
MegansNotes.txt	MegansNotes.txt! For all of your megans notes needs!	last month
README.md	transformAssetUrls	last month
babel.config.js	init	2 months ago
jsconfig.json	init	2 months ago
package-lock.json	File structure in place and router functional	last month
package.json	File structure in place and router functional	last month
vue.config.js	init	2 months ago



MINGW64:/c/Users/Brett/Documents/UMSAE/umsae-sponsorship-ui

```
Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/umsae-sponsorship-ui (master)
$ git log
commit 38b4be286cf178d50f83767bcbea3871ac9d98e0 (HEAD -> master, origin/master, origin/HEAD)
Author: Brett Stevens <steven78@myumanitoba.ca>
Date:   Sat Nov 5 23:44:06 2022 -0500

    Add profile components

    Also modified main.js to debug new components

commit 6c38aada252f38f5108e170c7526744a792c97b7
Author: brettstevens38 <38706144+brettstevens38@users.noreply.github.com>
Date:   Sat Sep 24 12:16:57 2022 -0500

    transformAssetUrls

commit 5726bfcc2b03f32b8d70f27fb62e4a45d80ef6eb
Author: brettstevens38 <38706144+brettstevens38@users.noreply.github.com>
Date:   Fri Sep 23 21:35:05 2022 -0500

    MegansNotes.txt! For all of your megans notes needs!

commit 905a6df3c08cf022105ef44e88c54dd3d232ddb3
Author: brettstevens38 <38706144+brettstevens38@users.noreply.github.com>
Date:   Fri Sep 23 19:46:32 2022 -0500

    Contact forms complete, draft ready
```

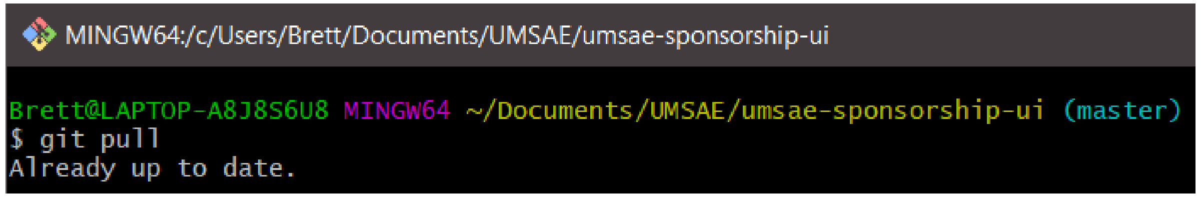
And that's it!

How 2 git (command line 😎)

What if you step away for a day or two and a teammate makes some changes one of the branches you want to work on?
(The whole purpose of Git)

- Call "git pull"

This will update your local repo to line up with the remote repo (Assuming you don't have any local changes).



```
MINGW64:/c/Users/Brett/Documents/UMSAE/umsae-sponsorship-ui

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/umsae-sponsorship-ui (master)
$ git pull
Already up to date.
```

There weren't any new changes

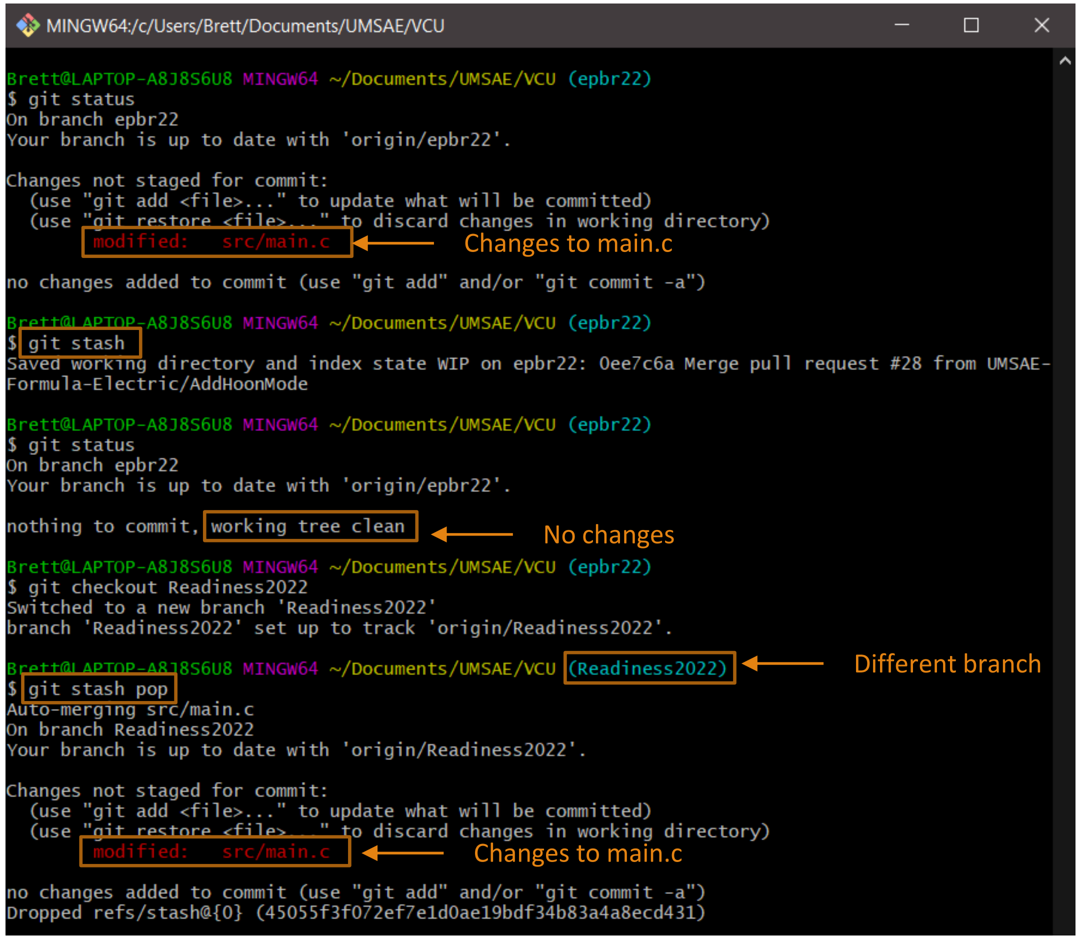
How 2 git (command line 😎)

What if you’re just about to commit changes and you realize you’re on the wrong branch?

(You get one pass)

- Call “git stash”
- Checkout on the desired branch
- Call “git stash pop”

This will store your changes and remove them from that branch, then you can just “drop” them back on the branch you meant to work on.



```
Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/VCU (epbr22)
$ git status
On branch epbr22
Your branch is up to date with 'origin/epbr22'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   src/main.c

no changes added to commit (use "git add" and/or "git commit -a")

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/VCU (epbr22)
$ git stash
Saved working directory and index state WIP on epbr22: 0ee7c6a Merge pull request #28 from UMSAE-
Formula-Electric/AddHoonMode

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/VCU (epbr22)
$ git status
On branch epbr22
Your branch is up to date with 'origin/epbr22'.

nothing to commit, working tree clean

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/VCU (epbr22)
$ git checkout Readiness2022
Switched to a new branch 'Readiness2022'
branch 'Readiness2022' set up to track 'origin/Readiness2022'.

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/VCU (Readiness2022)
$ git stash pop
Auto-merging src/main.c
On branch Readiness2022
Your branch is up to date with 'origin/Readiness2022'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   src/main.c

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (45055f3f072ef7e1d0ae19bdf34b83a4a8ecd431)
```

How 2 git (command line 😎)

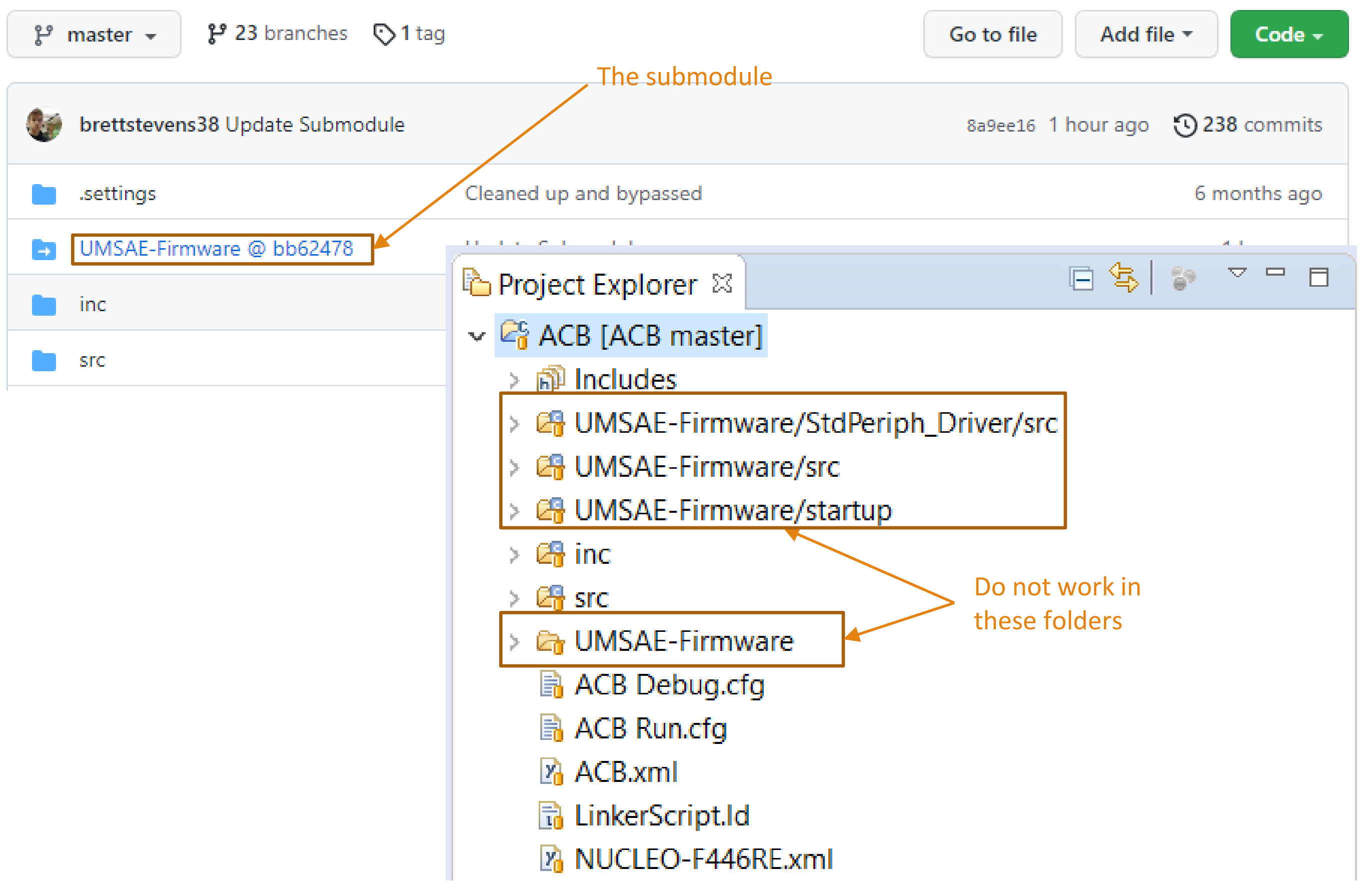
How 2 Submodule

ACB and VCU both contain submodules called UMSAE-Firmware

DO NOT WORK ON BOTH AT THE SAME TIME

Think of UMSAE-Firmware as a library that is being developed by a separate software company

You only want to have the most up-to-date version of that library in your code



brettstevens38 Update Submodule 8a9ee16 1 hour ago 238 commits

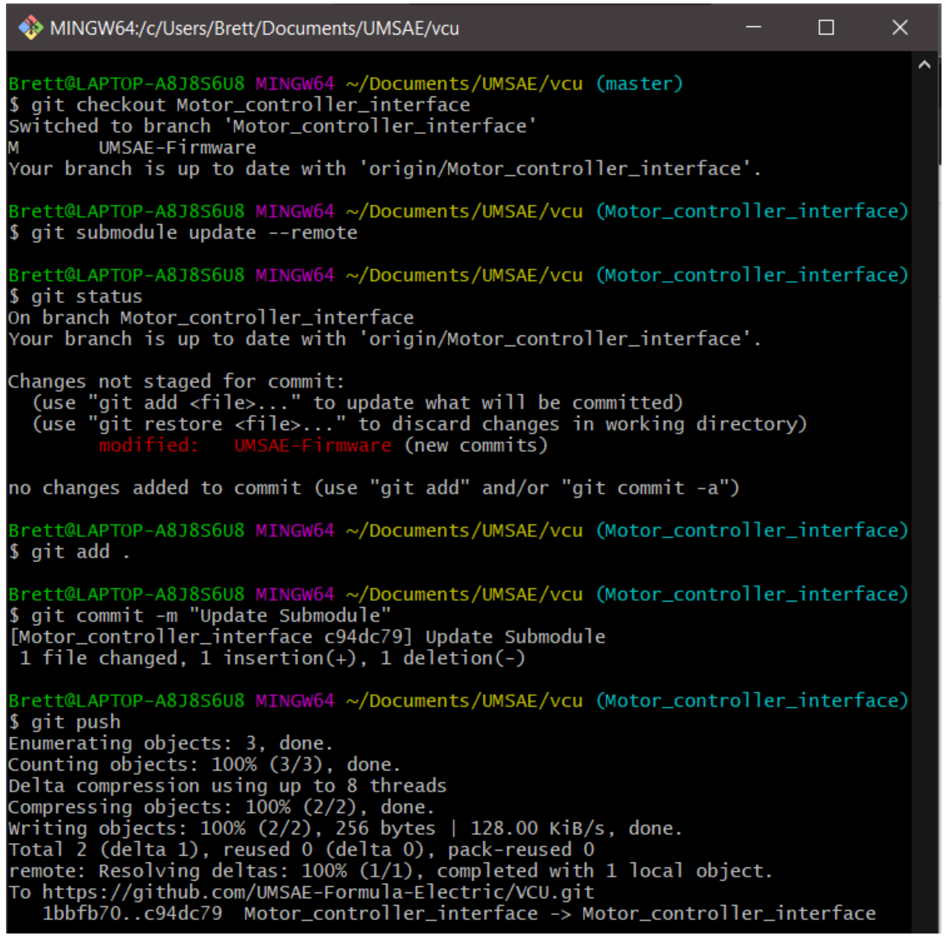
.settings	Cleaned up and bypassed	6 months ago
UMSAE-Firmware @ bb62478		
inc		
src		

```
ACB [ACB master]
  Includes
  UMSAE-Firmware/StdPeriph_Driver/src
  UMSAE-Firmware/src
  UMSAE-Firmware/startup
  inc
  src
  UMSAE-Firmware
  ACB Debug.cfg
  ACB Run.cfg
  ACB.xml
  LinkerScript.ld
  NUCLEO-F446RE.xml
```

How 2 git (command line 😎)

Steps:

1. Clone UMSAE-Firmware separately
2. Checkout desired branch
3. Make changes and commit
4. Have changes reviewed and merged into master
5. Go back to ACU/VCB and write “git submodule update --remote”
6. Stage, commit, and push to your branch



```
Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/vcu (master)
$ git checkout Motor_controller_interface
Switched to branch 'Motor_controller_interface'
M       UMSAE-Firmware
Your branch is up to date with 'origin/Motor_controller_interface'.

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/vcu (Motor_controller_interface)
$ git submodule update --remote

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/vcu (Motor_controller_interface)
$ git status
On branch Motor_controller_interface
Your branch is up to date with 'origin/Motor_controller_interface'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   UMSAE-Firmware (new commits)

no changes added to commit (use "git add" and/or "git commit -a")

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/vcu (Motor_controller_interface)
$ git add .

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/vcu (Motor_controller_interface)
$ git commit -m "Update Submodule"
[Motor_controller_interface c94dc79] Update Submodule
 1 file changed, 1 insertion(+), 1 deletion(-)

Brett@LAPTOP-A8J8S6U8 MINGW64 ~/Documents/UMSAE/vcu (Motor_controller_interface)
$ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 256 bytes | 128.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/UMSAE-Formula-Electric/VCU.git
   1bbfb70..c94dc79  Motor_controller_interface -> Motor_controller_interface
```

Review

1. navigate to directory
2. git clone / git pull (from server)
3. git checkout (to branch
4. make changes
5. git add (stage changes)
6. git commit
7. git push (to server)
8. confirm changes

Cheat Sheet

git clone <URL>
git clone --recurse-submodules <URL>
git branch --all
git checkout <branch name>
git checkout –b <new branch name>
git status
git log
git show
git add <file(s)>
git commit –m “<summary>” –m “<description>”

git push
git push --set-upstream origin <branch name>
git remote –v
git pull
git fetch
git stash
git stash pop
git init
git merge <branch to merge into current branch>

git remote add origin
git remote origin add
git add remote origin
git add origin remote
JAKE-CLARK.TUMBLR

Questions?