

**Softwire**

# **Softwire x OxWEST**

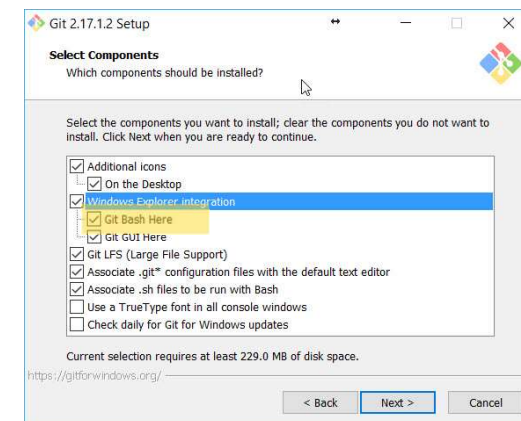
## Collaborating with Code

**Sarah Binney**  
**Ellen Skipper**  
**Aishah Omar**

# Before we start!

Important note

- **Please install Git**
  - <https://github.com/git-guides/install-git>
  - Pro tip: if you are on Windows and it offers you “Windows Explorer Integration > Git Bash Here”, say yes!



# Welcome!

Who are we?



- **Aishah Omar**
- Software Developer
- Studied Economics & German
- Did a bootcamp then started at Softwire in 2021
- Spent a lot of time playing pool in the past



- **Ellen Skipper**
- Technical Lead
- Studied Natural Sciences (Neurobiology + Psychology)
- At Softwire since 2020, except one year working in Tokyo
- Big fan of board games and karaoke

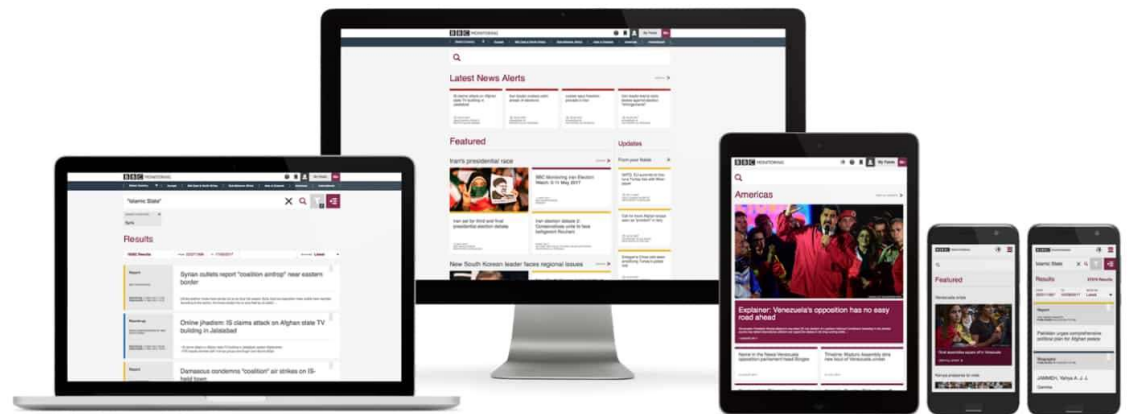


- **Sarah Binney**
- Technical Principal
- Did Physics at uni
- Been at Softwire since 2017
- Outside of work I like escape rooms and recently started playing drums in Softwire Band

# What is Softwire?

## About us

- We make software
- Founded in 2000 by Dan, Pete and Phil
- Grown steadily year on year, now 350+ perm employees
- Software development, product strategy and innovation, data engineering

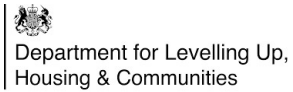


Softwire |

# Our projects

What we do

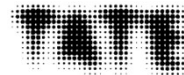
## Government & Public Sector



## Health and Care



## Leisure, Entertainment and Travel



## Other



Softwire |

# Our work

Languages and technologies we use



# Programs you'll need

And where to find them

- **Git**
  - <https://github.com/git-guides/install-git>
- **A text editor**
  - No formatting, just edit code
  - We recommend **Visual Studio Code**
  - Or use **Notepad** (Windows) or **Textedit** (Mac)
- **A terminal**
  - You don't need to install this either, it's already on your computer

# Today's aims

Other than "have fun"

- Learn Git
- Get started on Github
- Make and launch a website
  - <https://ejskipper.github.io/softwire-oxwest/>
- Understand how to collaborate with a team on a piece of software





- Send commands as a line of text
- Get responses back also as text
- Historically the only way to interact with a computer (before mice, windows, GUIs were invented)
- Still the only way to use certain tools
- Offers automation and scripting

```

Macintosh HD -- top -- 80x24

Processes: 210 total, 2 running, 9 stuck, 199 sleeping, 901 threads   23:30:03
load avg: 1.40, 1.75, 1.75   CPU usage: 4.15 user, 4.48 sys, 91.44% idle
SharedLibs: 1648K resident, 0B data, 0B linked.
MemRegions: 31278 total, 1892M resident, 117M private, 564M shared.
PhysMem: 5893K used (1191M wired), 10G unused.
VM: 523G vsize, 1026M framework vsize, (0/0) swaptos, (0/0) swaptos.
Networks: packets: 12105/8925K in, 11907/1964K out.
Disks: 80156/2205M read, 21235/425M written.

PID COMMAND           %CPU TIME          #TH  #WQ   #PORT MEM      PURG      CMPR  PGRP  PPID
592 screencaptur 0.0 00:00:02.7 5 55+ 1952K+ 20K+ 0B 262 262
590 mdworker 0.0 00:00:01.3 0 44 2032K 0B 0B 590 1
589 mdworker 0.0 00:00:01.3 0 44 1572K 0B 0B 589 1
588 top 1.7 00:00:01.1/0 0 22+ 2668K 0B 0B 588 1
584 bash 0.0 00:00:00.1 0 15 5 598K 0B 0B 584 1
583 login 0.0 00:00:01.3 1 28 1272K 0B 0B 583 482
574 auditd 0.0 00:00:00.2 0 25 560K 0B 0B 574 1
567 System Prefe 0.0 00:03:23.3 0 270 39M 8364K 0B 567 1
561 systemstatd 0.0 00:00:01.2 1 19 1648K 0B 0B 561 1
560 com.apple.We 0.0 00:01:11.9 0 229 25M 0B 0B 560 1
559 com.apple.We 0.0 00:00:07.15 3 224 151M 1716K 0B 559 1
555 bash 0.0 00:00:00.1 0 15 604K 0B 0B 555 554
554 login 0.0 00:00:01.3 1 28 1176K 0B 0B 554 482
550 bash 0.0 00:00:00.1 0 15 608K 0B 0B 550 548

```

[illegible]

# The command line

How to look like a hacker without really trying

## On Windows:

- Type “cmd” into the Start menu
- ...or open a folder, right click and select “Git Bash Here”

## On Mac:

- From Launchpad, type “Terminal”

## On Unix:

- If you’re on Unix, you already know how to find the command line ;)
- (On Linux it’s Ctrl + Alt + T)

```
echo "Hello world!"
```

# Terminal Fun

Pls try this at home

- Press Enter to run a command
- Spaces and punctuation generally matter
- Up arrow to get the previously run command back
- Tab to “autocomplete” (particularly file names)
- Ctrl+C to cancel a command
- You’re “in” a particular folder on your computer. Navigate around with `cd` (change directory)
- Don’t run random stuff off the internet unless you know what it does

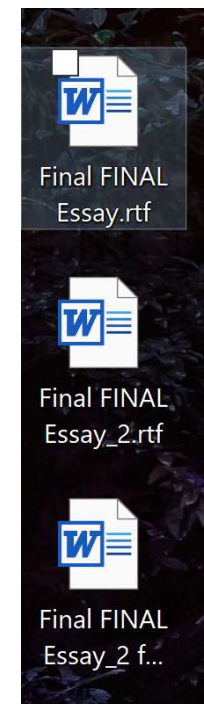
## Terminal Cheat Sheet

- `echo "Here is a message!"`
- `cd MyFolder/`
- `cd ..`
- `ls`
- `mkdir ANewFolder`

# What is Git?

Literally why are we here

- Git is a program for controlling versions of a project. It allows you to track changes you make to files over time.
- To make an analogy, it's like having a save button on your project. At any time, you can save what you've written and mark that point in time. This means you can review versions of the code or go back to a previous version.



# Let's git going

## Exercise

- **Create a new folder**
  - Make sure this is in a folder that is not backed up e.g. by OneDrive
  - Also, not inside another Git repository
  - Avoid spaces in the folder name
  - I keep mine in C:/Work/ProjectName
- **Create a new file in that folder**
  - Use VSCode for preference
- **Type something into the file**
- **Save the file as README.md**
- **Navigate to the folder using the command line**
- **Init** the Git repository
- **Add** files and **commit**

## Terminal Cheat Sheet

- `cd MyFolder/`
- `cd ..`
- `ls`
- `mkdir ANewFolder`
- `git init`
- `git add .`
- `git commit -m "Message here"`

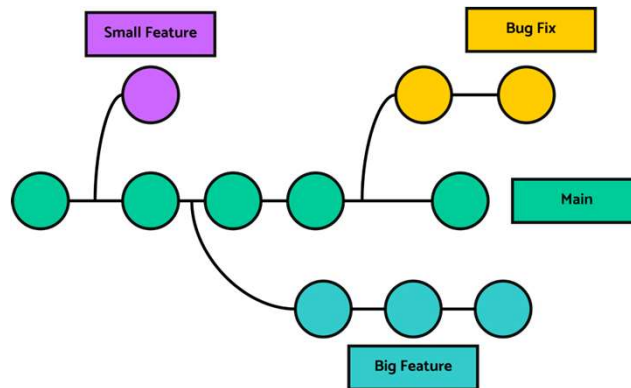
# Git techniques

- Make some more commits and watch your repository grow
- View history
  - `git log --oneline`
- Time travel
  - `git checkout <commit-hash>`

# Git techniques

Think of trees

- **Branching** is a way of keeping your changes isolated from the existing codebase
- **Merging a branch**
  - A 'merge' is just applying all the changes from one branch, onto another.
- **Deleting a branch**



- `git branch <branch-name>`
- `git checkout <branch-name>`

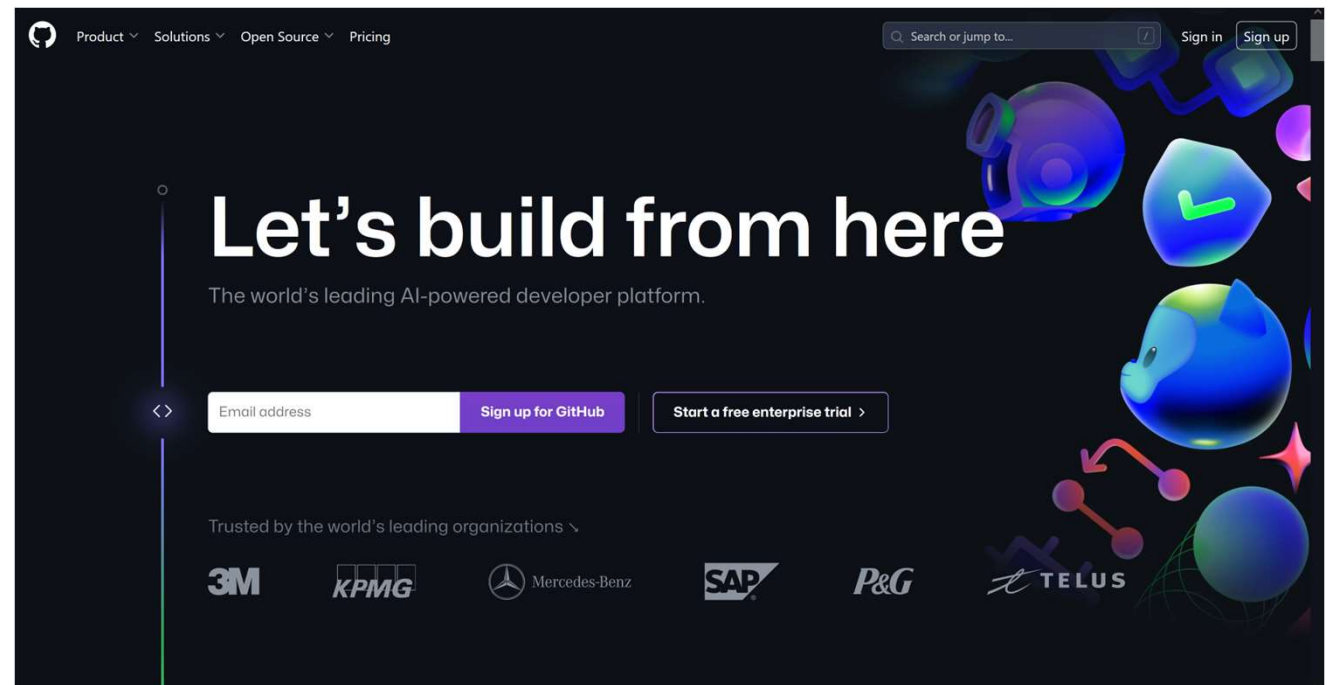
OR `git checkout -b <branch-name>` as a shortcut

- `git branch` (lists all branches)
- `git merge <other-branch-name>`
- `git branch -d <branch-name>`

# What is Github?

Not the same things as Git

- Somewhere you can host your Git repositories online for free
- Overwhelmingly Git host of choice for open source projects
  - Others include: Gitlab, Bitbucket (useful if you don't like Microsoft)
- Also creators of Github Copilot (AI coding tool)
- Watch out! Your code is **public** and **open source** by default

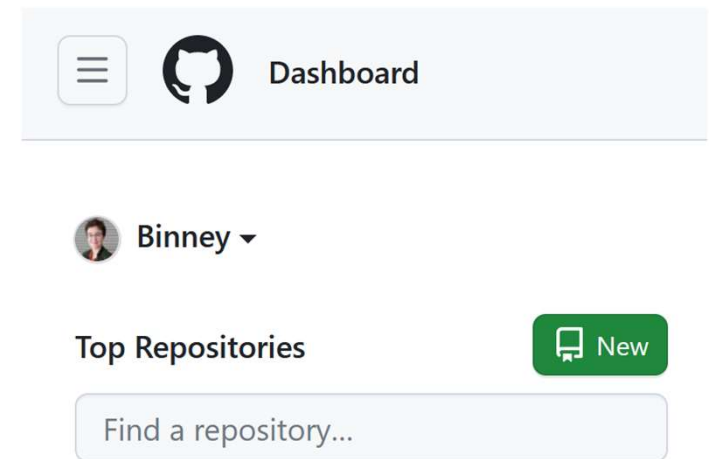




# Let's Github going

Start using Github

- Create an account, or log in if you already have one
- Create a new repository
- Follow the instructions onscreen to **clone** into it in the folder you made earlier
- **Push** your commit to Github
- Refresh and voila\*!



# Git and authentication

Keeping your code secure

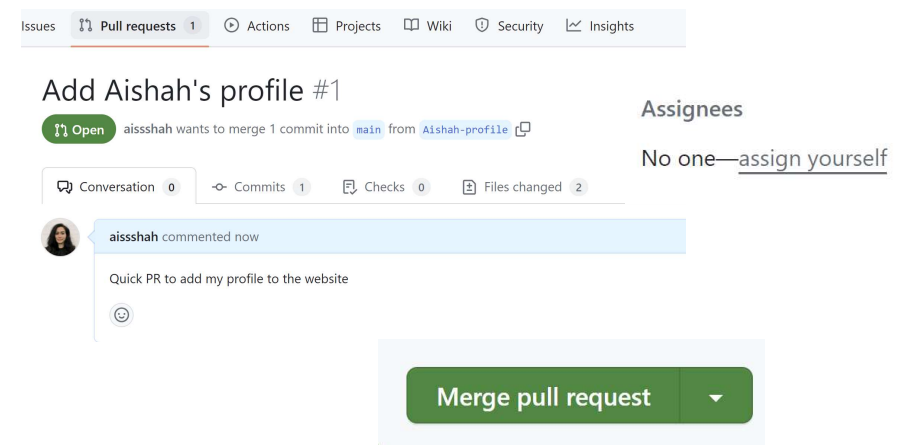
- **\*Except... 2 main ways to log in on the command line**
  - HTTPS opens a login dialog on your computer
  - SSH involves more one-time setup but is then seamless
- **SSH (Secure Shell) based on public key cryptography**
  - Upload your public key to Github
  - Keep your private key secret
- `ssh-keygen`
- Press enter to all defaults
- `more ~/.ssh/id_rsa.pub`
- Copy the result into Github -> Settings -> SSH and GPG keys -> New SSH key
- Give your key a title (e.g. "Dell work laptop")
- `git push`
- Actually voila

**Fun Fact: README.md is a special name that Github takes and uses as your "homepage" when viewing the code**

# Pull Requests

Also known as Merge Requests

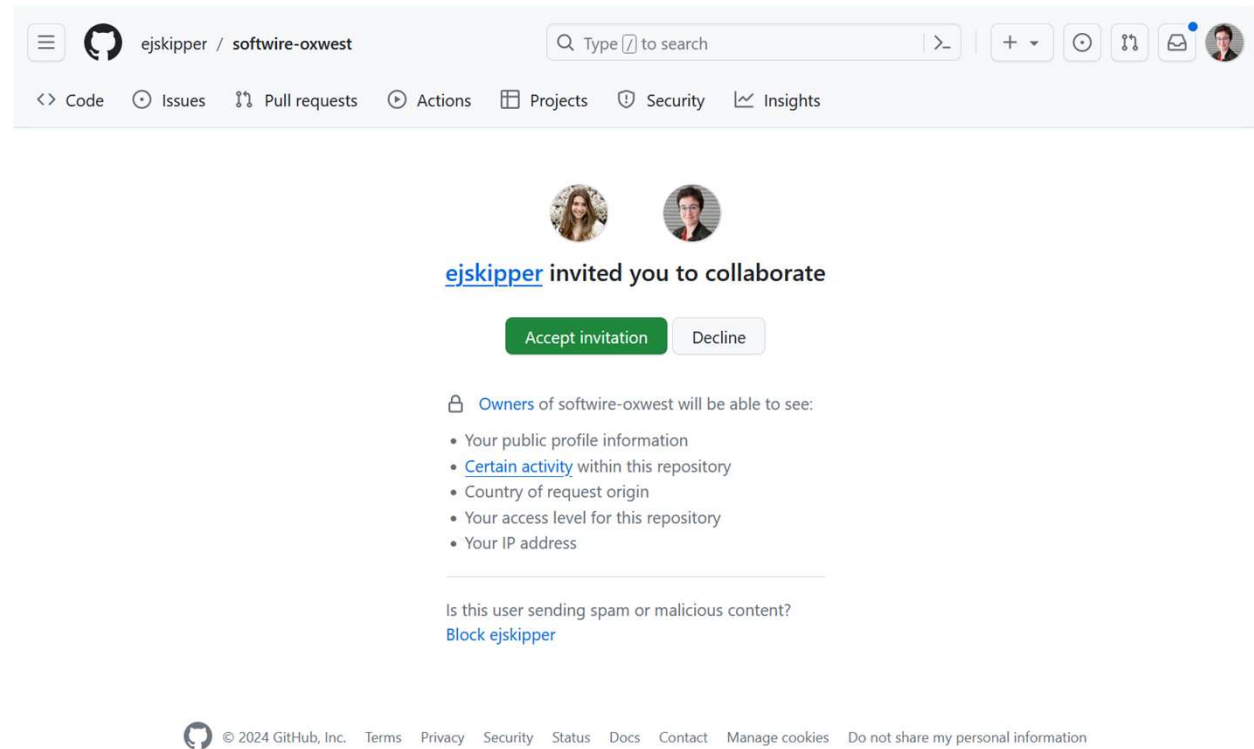
- “I would like to merge my code please”
  - Create a branch, make a change to the code, commit it and push it
  - Open a Pull Request in the Github UI, assign yourself, double-check, then merge it and delete the branch
  - Pull your main branch to see changes locally
- `git checkout -b branch-name`
  - `git add .`
  - `git commit -m “Commit message”`
  - `git push`



# Collaborating on Github

Hackathon participants, take note

- **Nominate one person in your team to create your team repository**
- **Repo owner please create and commit a file called `index.html`**
  - For now it can just say "Hello world!" and nothing else
- **That person should add everybody else**
- **Everybody clone into the shared repo**
  - Make sure you don't clone it inside of an existing Git repository!



# Top Etiquette Tips

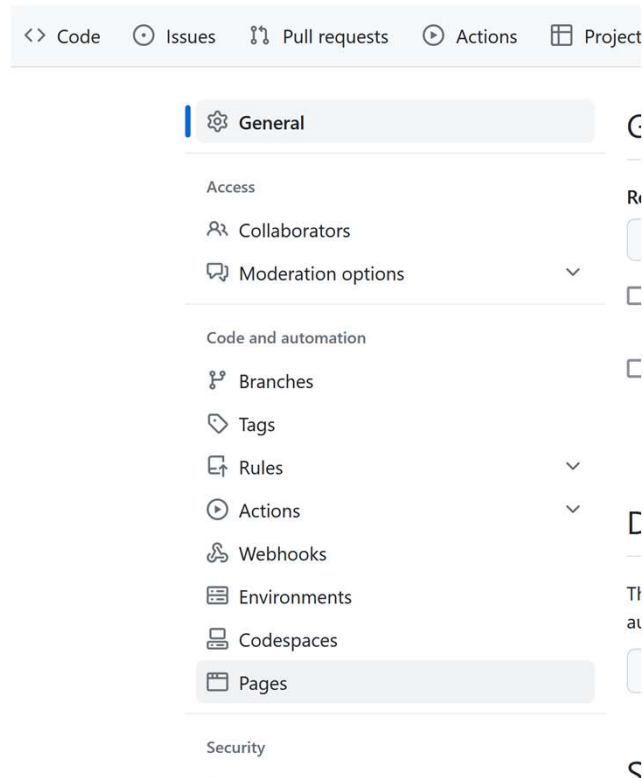
More stuff I wish I'd known when I started using Git

- **Don't commit directly to main!**
- **Always start a branch for your latest feature. The reason for this will become clear...**
- **Commit early and often**
- **Don't change shared history**

# Github Pages

Repository owner will need to do this

- **Free GitHub offering**
- **Host a live, free, public version of your website**
- **Share the link with anyone and they can visit websites that you've created!**
  - index.html is a magic name that gets served at the root URL (i.e. "/")



- **Settings -> Pages**
- **Source: Deploy from a branch**
- **Branch: main**
  - Note! Github Pages will ignore all other branches
- **Wait 5 mins**
- **username.github.io/repository-name**
  - E.g. <https://ejskipper.github.io/software-oxwest/>

# Merge conflicts

AKA how to do a hackathon without losing your mind

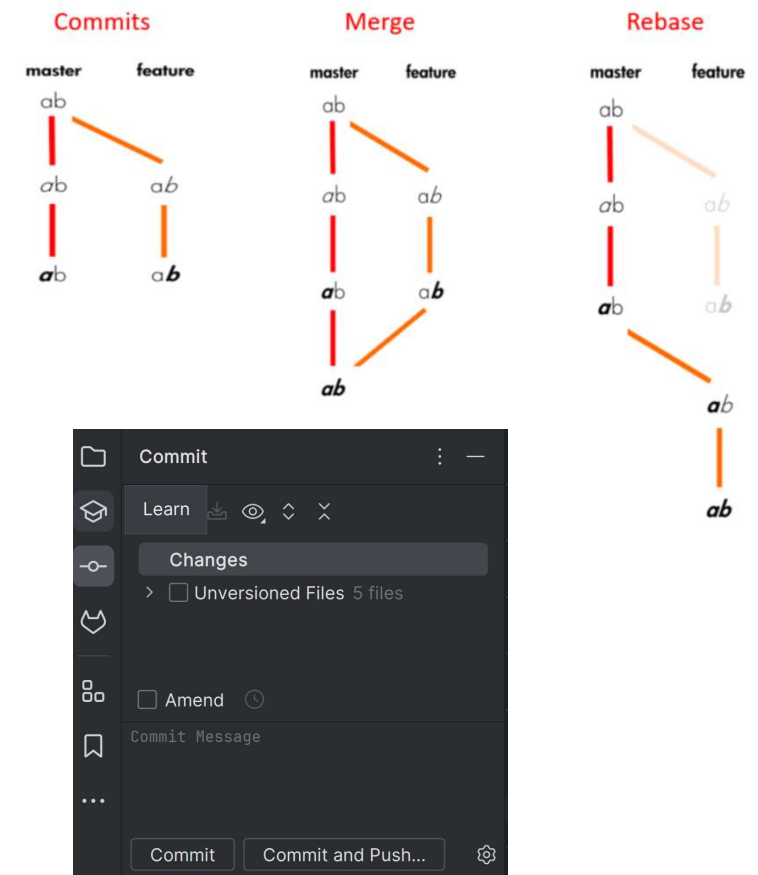
- If the same line is changed by multiple people, git will show a **merge conflict**
- Don't panic
- Conflicts are highlighted with HEAD, main and symbols
  - to resolve, delete the changes which aren't relevant and the extra lines
- Avoid them by working on different areas in parallel
  - e.g. separate files

```
<body>
  <<<<<<< HEAD
    <h1 class="page-heading">GIT WORKFLOW WORKSHOP</h1>
    =====
    <h1 class="some-heading">GIT WORKFLOW WORKSHOP</h1>
  >>>>>> main
</body>
```

*Tip: Visual Studio Code has great tools for resolving merge conflicts*

# Advanced Git

- Rebasing vs merging
- Use `git log --graph` to see logs as a graph
- `git diff` to compare files or branches
- Add aliases (shortcuts) for common commands
  - e.g. `gc` for `git checkout`
- Other Git clients
  - e.g. Github Desktop, GitKraken or in-built within your IDE

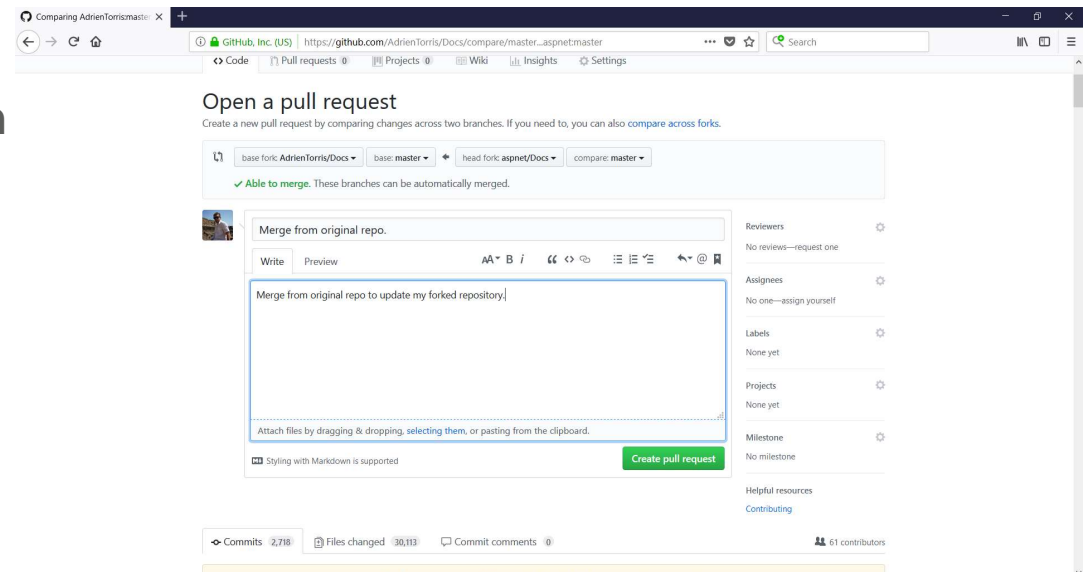
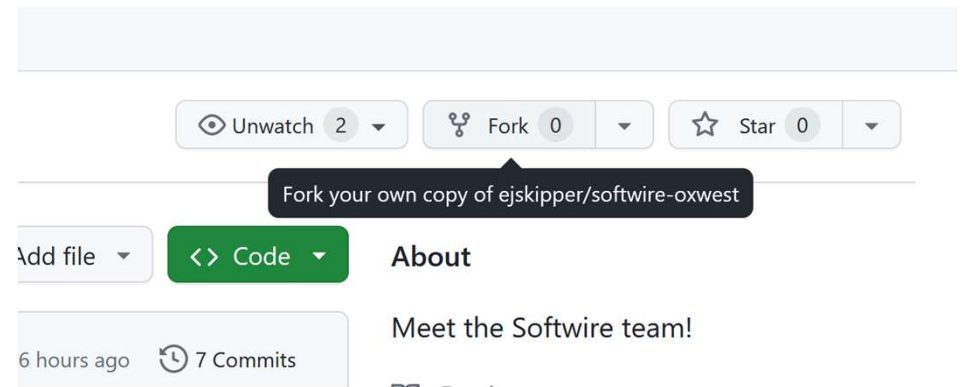




# Fork





Collaborating without permission

- Fork = make a complete copy of a repository
- The new version is yours to do with as you see fit
- Then create a Pull Request between forks
- Useful for open source contributing



# Today's aims

Other than “have fun”

- Learn Git 
- Get started on Github 
- Make and launch a website 
- Understand how to collaborate with a team on a piece of software 

# Where to go next!

Some ideas

- **Grow out your team Github Pages website!**
  - Practice HTML, CSS and JavaScript
  - Add a README.md
  - Add more web pages and links between them
- **Practice your Git skills!**
  - <https://github.com/cameronmccormack/ts-git-training>
- **Submit a Pull Request to an open source project!**
  - Especially documentation!
- **Go to a hackathon!**
- **Start a project with friends!**

**Softwire**

**Thank you!**  
Aishah, Ellen and Sarah

29<sup>th</sup> February 2024 🐸

[www.softwire.com/careers](http://www.softwire.com/careers)