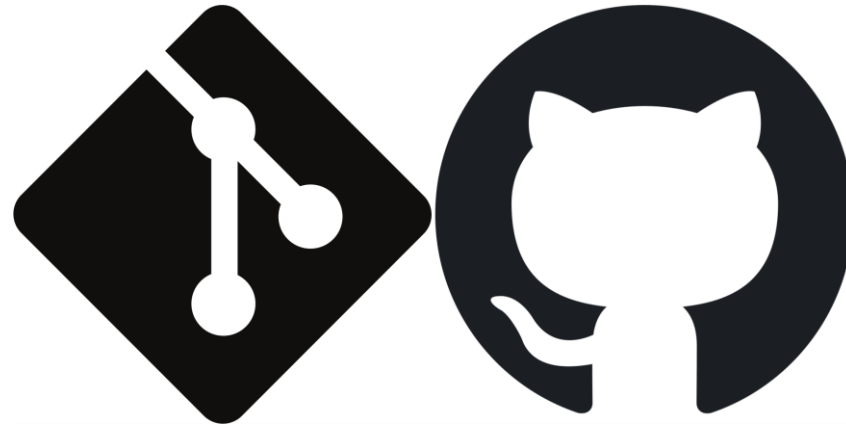


Welcome!

Some initial course info:

- You can find links to our draft schedule and all our course materials through our wiki.
 - https://github.com/CefasRepRes/Git_Training/wiki/Schedule
 - *We will take breaks during the day!*
- Feel free to interrupt and ask questions as we go.
- H&S...



Introduction to Git & GitHub

Jennifer Graham

Tiago Silva

David Ryder

Stephen Gregory

Joe Ribeiro

Session 1

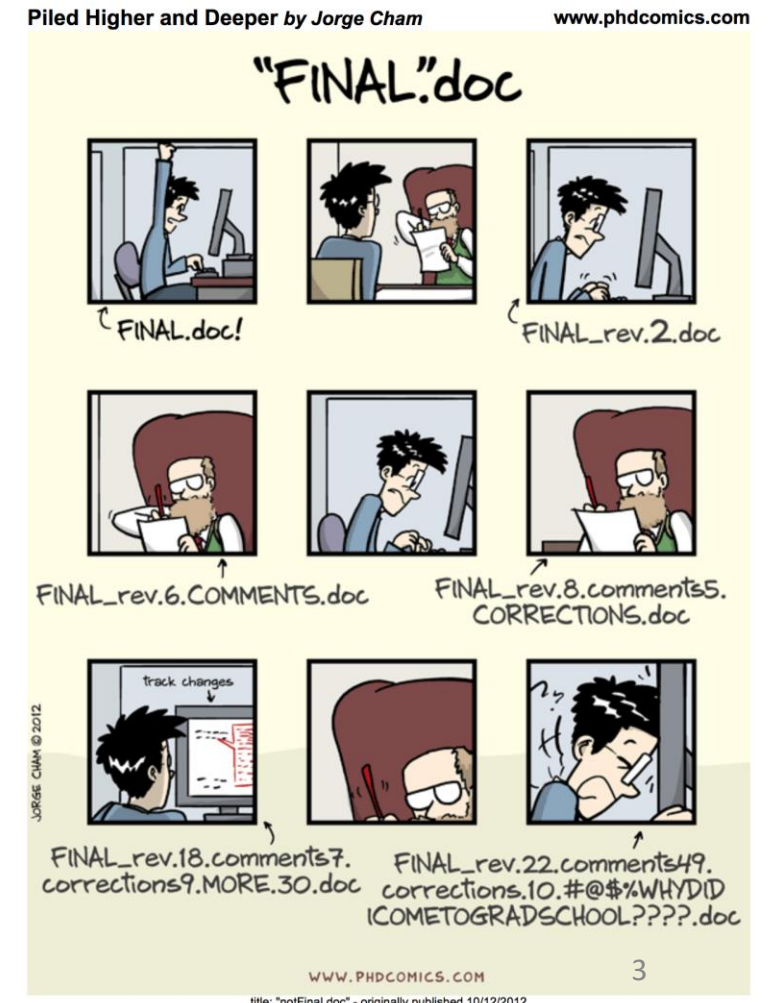
16th May 2025

Why use version control?

- How many people have directories that include files like this?

- `File_16042025.doc`, `File_30052025.doc`, ...
- `File_old.doc`, `File_new.doc`
- `File.doc.orig`
- `File.doc`, `File_test.doc`

- Version control allows you to:
 - Back up your code.
 - Keep track of changes.
 - Share your code with others.
 - Develop code with collaborators.



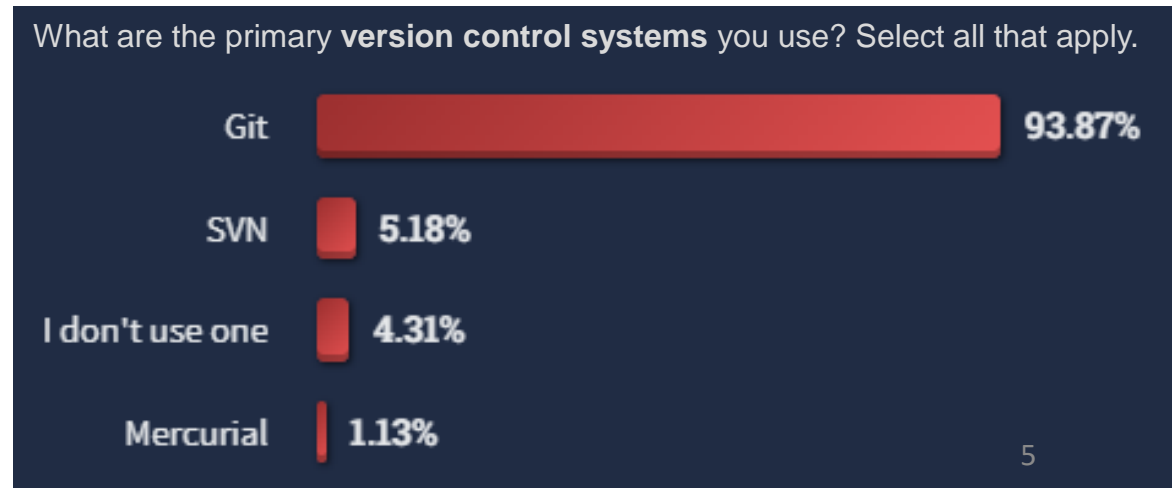
Changes are documented

- As code is backed up, can be confident in testing new ideas.
- Version control software ensures that changes are attributable to individuals.
 - If the code breaks, you know why!
- Code can be shared along with the documentation and file history.
 - Documentation can explain how and why the code should be used.
 - Can create fixed versions/releases, along with [DOIs](#).
 - Sharing now often required for publication.
- All the above provides “quality assurance” in the final product.

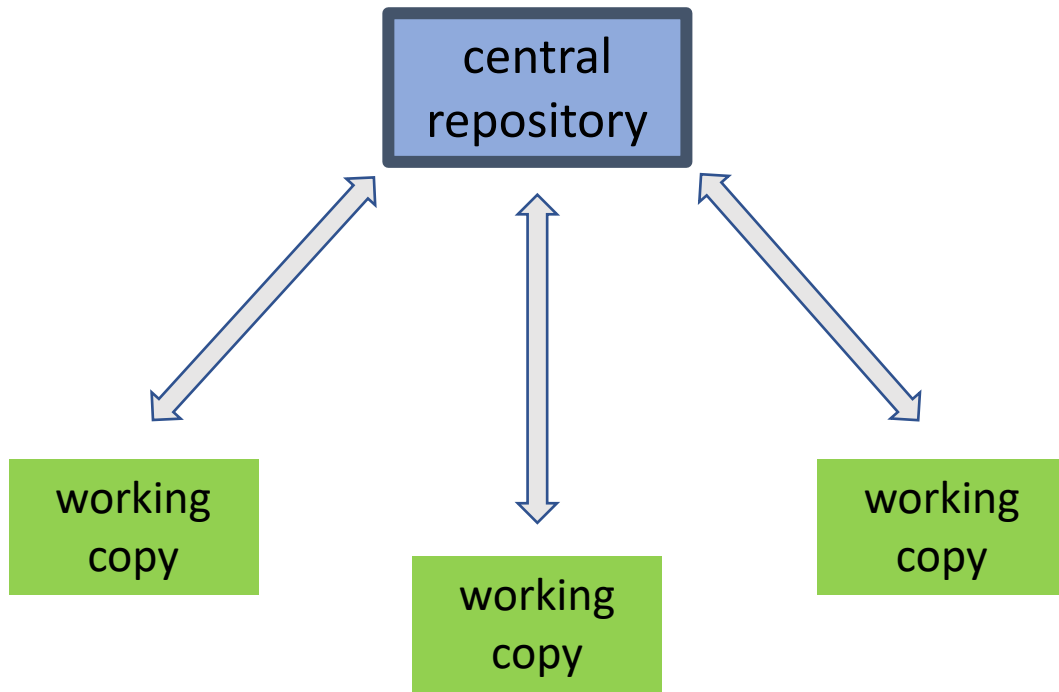
Version control software

- Many options out there...
 - https://en.wikipedia.org/wiki/List_of_version-control_software
 - All are *agnostic* in terms of the programming language
- Client-server model :: all users share a single code repository.
 - e.g., SVN (subversion)
 - Legacy choice, depending on when project started (many now switching)?
- Distributed model :: all users have their own local repositories.
Changes can be shared/merged as a separate step.

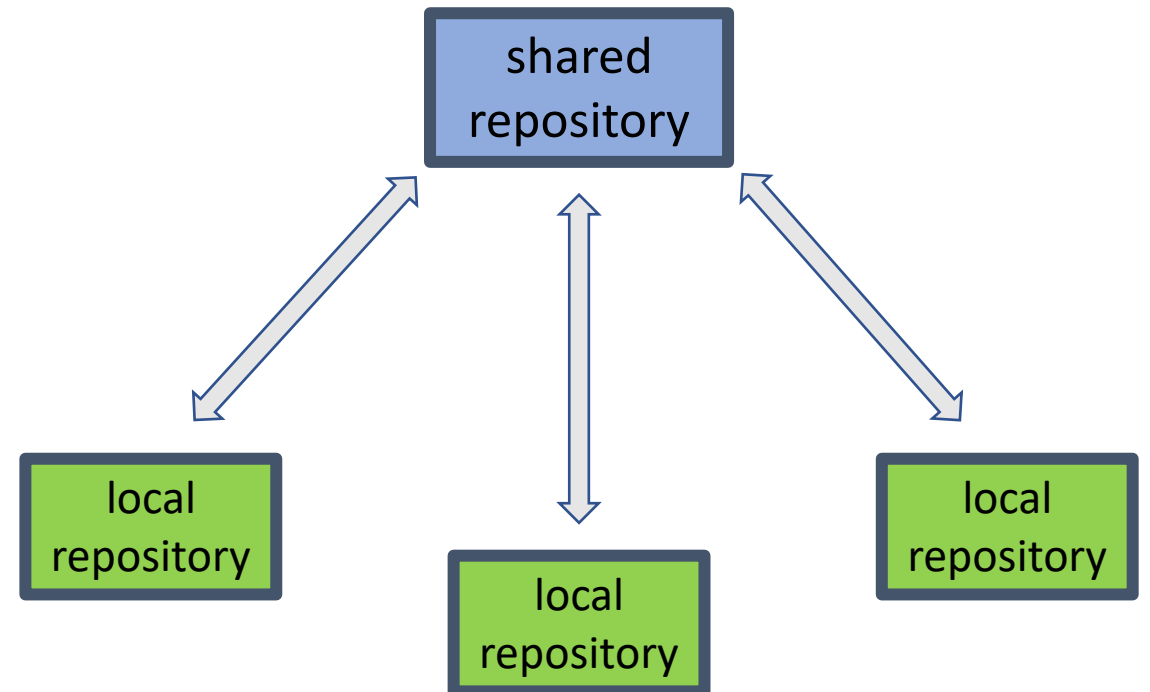
- e.g., **Git**
 - What we're using today!
 - Well supported, large user community, e.g.,
From Stack Overflow user survey (2022),
"No other technology is as widely used as Git."



Central vs Distributed workflow?



Centralised/client-server model (e.g., SVN)



Distributed model (e.g., Git)

So, what are Git & GitHub?

NB. These are two different things

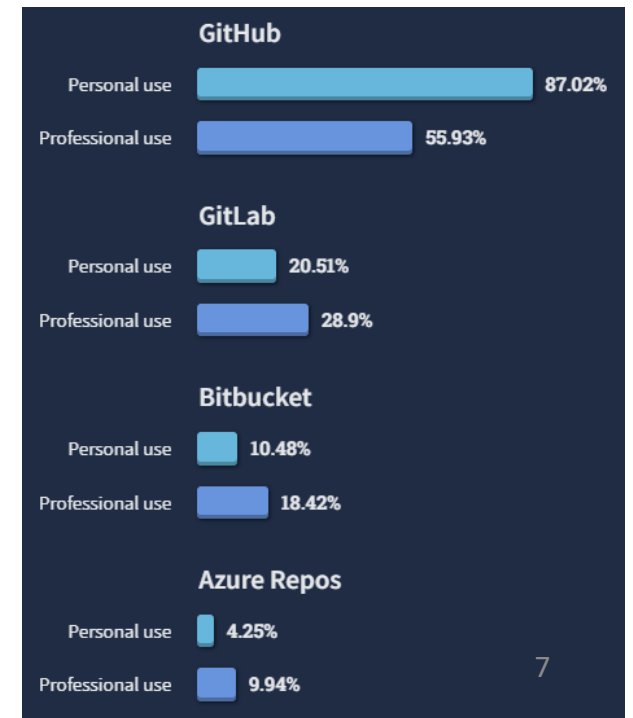
- Git = version control software.
- GitHub = web-based repository hosting service*.

i.e. Git is the software behind the GitHub web service.

You can use Git without GitHub. However, GitHub provides useful tools (especially for sharing your code).



* Other providers exist, but GitHub is the most popular, e.g.:



Useful resources

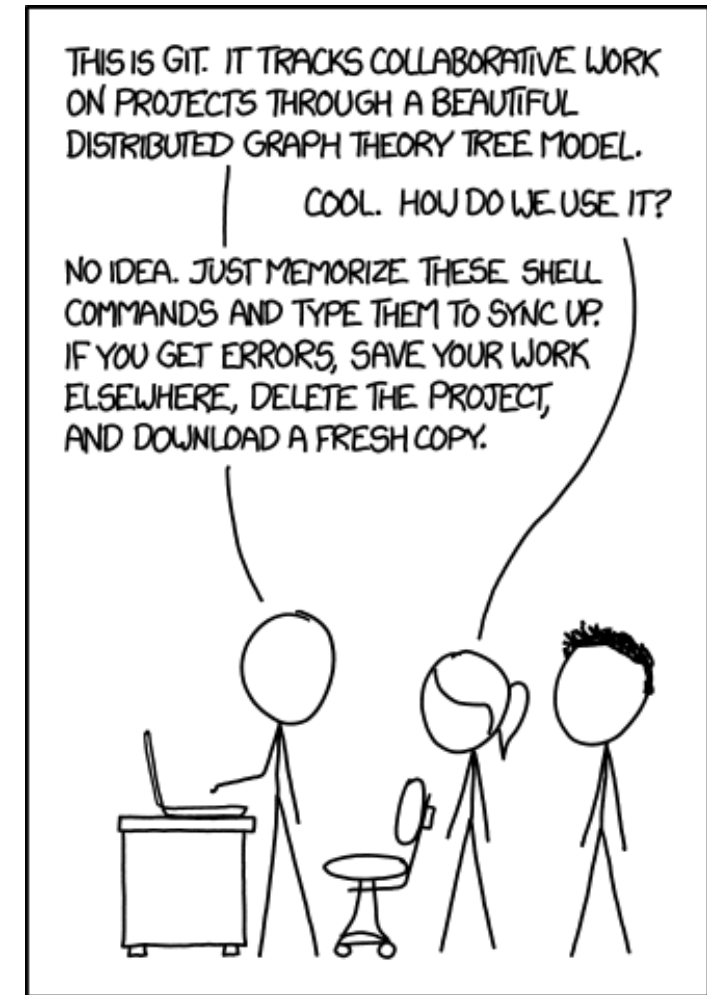
Git and GitHub are widely used, so there are lots of useful websites, training videos, online courses etc. e.g.,

- GitHub guides & help:
 - <https://guides.github.com/>
- Git guides:
 - <https://git-scm.com/doc>
- Git cheat sheet:
 - <https://education.github.com/git-cheat-sheet-education.pdf>
- E-books:
 - <https://www.git-tower.com/learn/git/ebook/>
- Webinars:
 - <https://www.youtube.com/watch?v=v3Y8c2KMay8>
 - <https://www.youtube.com/watch?v=ShH1g4l9A54>
- Internet searching!



Using Git

- Various options depending on preference:
 - **Command line e.g., Git for windows** (bash shell)
 - Many other command line interfaces available e.g., Windows 10 bash shell, HPC, Linux or Mac terminals.
 - GUI e.g., GitHub desktop
 - Rstudio or pycharm (some built-in functionality, similar to GUI)
 - *[NB. Admin rights not needed to install Git]*
- Whatever option you choose, all the above follow the same basic workflow.
 - Can choose whatever works for you e.g., are you more comfortable with a GUI or command line?
 - Will initially take some practice, but using git can (should) become a regular activity.



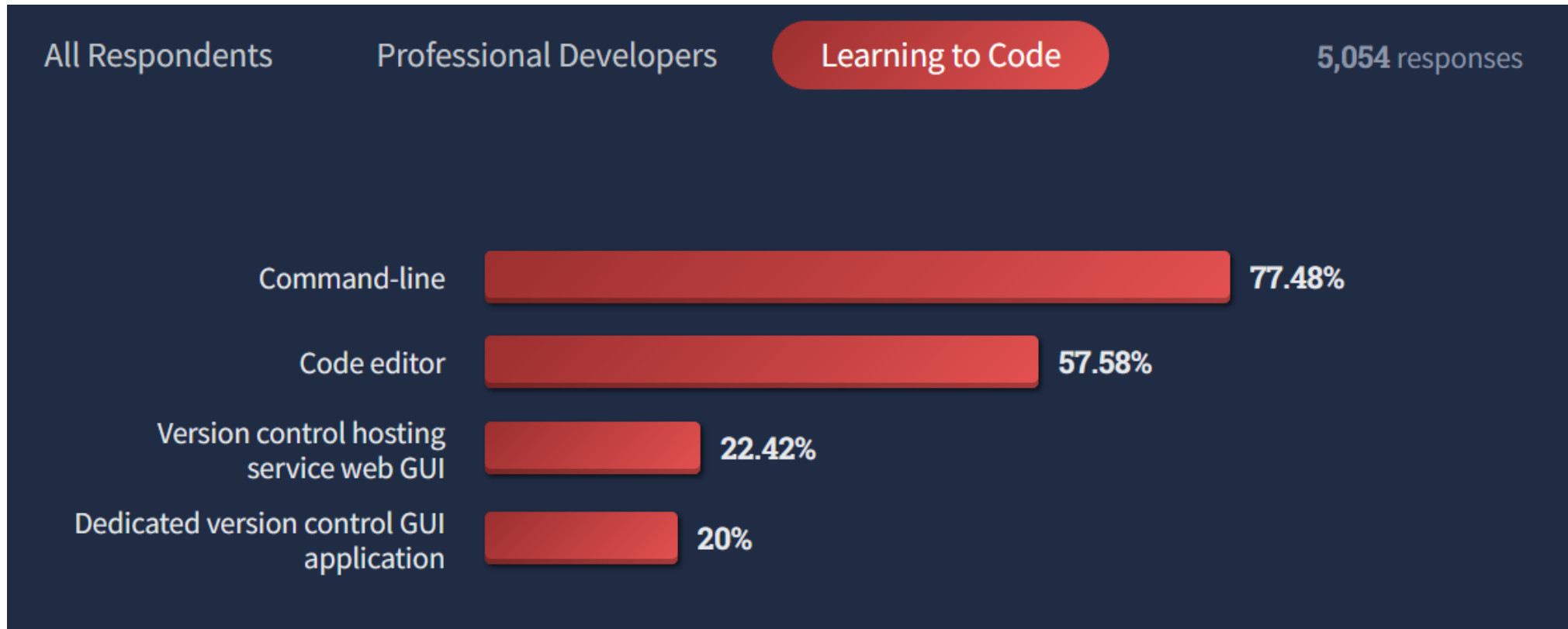
<https://xkcd.com/1597/>

Why are we using the command line?

- **Clearly shows the commands being used at each step in the workflow.**
 - Command name matches exactly what is used on the command line.
- Can apply the workflow to a chosen GUI at a later date (e.g., Rstudio, pycharm, or GitHub Desktop).
- Not all commands are available in every GUI, so always have the option of moving back to the command line.

Why are we using the command line?

Stack Overflow user survey (2022): *“The command line is the primary way developers interact with their version control system.”*



<https://survey.stackoverflow.co/2022/#technology-version-control>

Introduction to command line...

Basic Git workflow

- Create a local repository (code directory).
- Add/edit a file.
- Commit your changes (i.e. document changes).
- Then repeat!

Optional (but recommended):

- Push changes to remote repository on GitHub (back-up/sharing).

Creating a repository...

- I have a folder or directory, how do I start tracking my changes?

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

\$ ls

hello_world.py

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

\$

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

\$ ls

hello_world.py

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

\$ git init

git init
:: turn current directory
into git repository.

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

\$ ls

hello_world.py

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

\$ git init

Initialized empty Git repository in C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/hello_world/.git/

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

```
$ ls  
hello_world.py
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

```
$ git init  
Initialized empty Git repository in C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/  
/hello_world/.git/
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

```
$ git status
```

git status

:: show which files have
been tracked or modified.

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

```
$ ls
hello_world.py
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

```
$ git init
Initialized empty Git repository in C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/hello_world/.git/
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

```
$ git status
On branch main
```

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)
hello_world.py

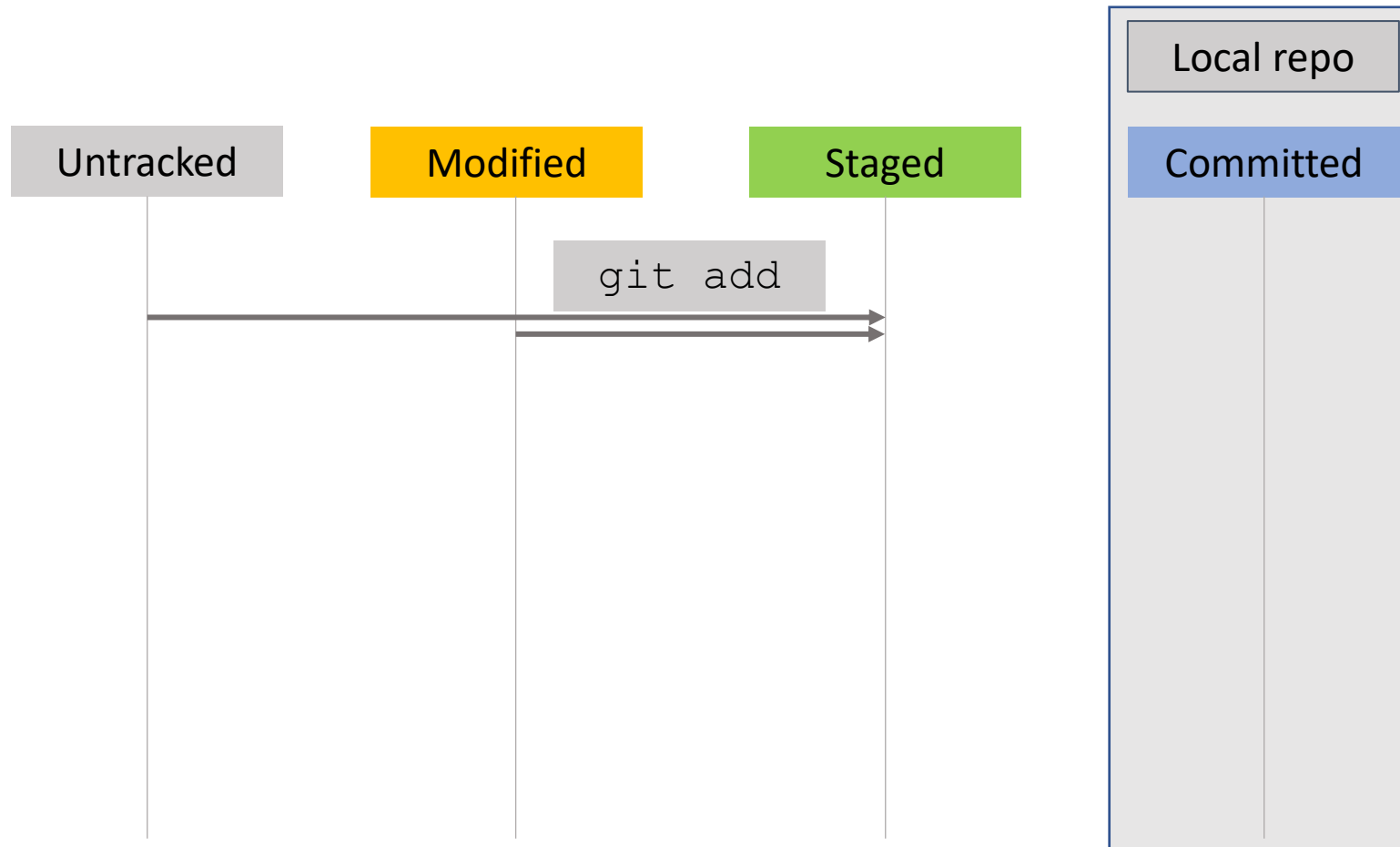
nothing added to commit but untracked files present (use "git add" to track)

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

```
$
```

The “Staging area”

- Git doesn't have to track all files in a repository.
- There are three “states” of tracked files in a Git repository.
 - Working directory :: current version of files, containing any modifications.
 - Staging area :: snapshot of files/modifications that you plan to commit.
Note: the Staging area is also often called the index.
 - Repository/Git directory :: contains the commit history for all files in the repository.
- You can check the state of each file at any point using “`git status`”



JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

```
$ ls
hello_world.py
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

```
$ git init
Initialized empty Git repository in C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/hello_world/.git/
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

```
$ git status
On branch main
```

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)
hello_world.py

nothing added to commit but untracked files present (use "git add" to track)

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

```
$ git add hello_world.py
```

```
git add <file>
:: stage the file for
commit i.e. track changes.
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

```
$ ls
hello_world.py
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world

```
$ git init
Initialized empty Git repository in C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/hello_world/.git/
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

```
$ git status
On branch main
```

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

hello_world.py

nothing added to commit but untracked files present (use "git add" to track)

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

```
$ git add hello_world.py
warning: LF will be replaced by CRLF in hello_world.py.
The file will have its original line endings in your working directory
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

```
$
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git add hello_world.py

warning: LF will be replaced by CRLF in hello_world.py.

The file will have its original line endings in your working directory

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git status

On branch main

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: hello_world.py

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$


```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git add hello_world.py
```

```
warning: LF will be replaced by CRLF in hello_world.py.
```

```
The file will have its original line endings in your working directory
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git status
```

```
On branch main
```

```
No commits yet
```

```
Changes to be committed:
```

```
(use "git rm --cached <file>..." to unstage)
```

```
new file:   hello_world.py
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git commit
```

git commit

:: confirm changes with
message/explanation.



*COMMIT_EDITMSG - Notepad



File Edit Format View Help

Adding first file to repository

```
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch main
#
# Initial commit
#
# Changes to be committed:
#   new file:   hello_world.py
#
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git add hello_world.py

warning: LF will be replaced by CRLF in hello_world.py.

The file will have its original line endings in your working directory

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git status

On branch main

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: hello_world.py

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git commit

hint: waiting for your editor to close the file... unix2dos: converting file C:/Users/JG10/OneDrive - AS/GitHub/Repos/hello_world/.git/COMMIT_EDITMSG to DOS format...

dos2unix: converting file C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/hello_world/.git/COMMIT_EDITMSG to Unix format...

[main (root-commit) a59cb3f] Adding first file to repository

1 file changed, 9 insertions(+)

create mode 100644 hello_world.py

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git add hello_world.py

warning: LF will be replaced by CRLF in hello_world.py.

The file will have its original line endings in your working directory

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git status

On branch main

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: hello_world.py

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git commit

hint: waiting for your editor to close the file... unix2dos: converting file C:/Users/JG10/OneDrive - AS/GitHub/Repos/hello_world/.git/COMMIT_EDITMSG to DOS format...

dos2unix: converting file C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/hello_world/.git/COMMIT_EDITMSG to Unix format...

[main (root-commit) a59cb3f] Adding first file to repository

1 file changed, 9 insertions(+)

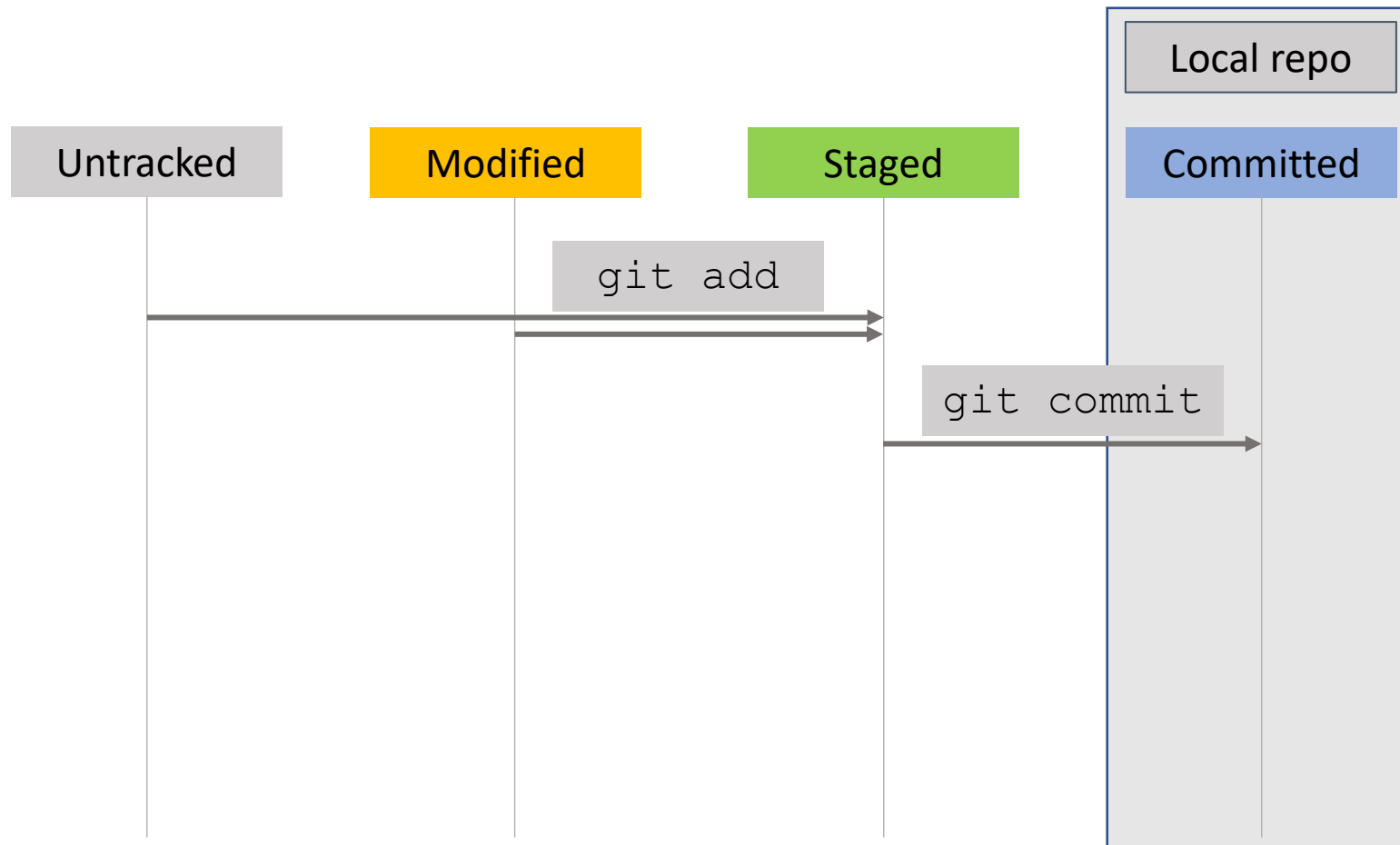
create mode 100644 hello_world.py

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git status

On branch main

nothing to commit, working tree clean



Committing changes

- Commit your change.
 - You will be prompted for a commit message.
 - Make a concise, meaningful description of the changes and why they were made.
 - Best practice is to stage only associated modifications in each commit.
- How often should you commit?
 - More often than you think - whenever you make a change that works?
 - The more often you commit, the easier it is to document (with short, meaningful messages), or roll-back if needed.
 - Avoid committing untested or unfinished modifications!

| | COMMENT | DATE |
|---|------------------------------------|--------------|
| ○ | CREATED MAIN LOOP & TIMING CONTROL | 14 HOURS AGO |
| ○ | ENABLED CONFIG FILE PARSING | 9 HOURS AGO |
| ○ | MISC BUGFIXES | 5 HOURS AGO |
| ○ | CODE ADDITIONS/EDITS | 4 HOURS AGO |
| ○ | MORE CODE | 4 HOURS AGO |
| ○ | HERE HAVE CODE | 4 HOURS AGO |
| ○ | AAAAAAA | 3 HOURS AGO |
| ○ | ADKFJ\$LKDFJSDKLFJ | 3 HOURS AGO |
| ○ | MY HANDS ARE TYPING WORDS | 2 HOURS AGO |
| ○ | HAAAAAAAAAANDS | 2 HOURS AGO |

AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

<https://xkcd.com/1296/>

```
git init           :: turn current directory into git repository.  
git add <file>     :: stage the file for commit i.e., track changes.  
git commit         :: confirm changes with message/explanation.  
  
git status         :: show which files have been tracked or modified.
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ ls
```

```
README.md  hello_world.py
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git status
```

```
On branch main
```

```
Untracked files:
```

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

```
README.md
```

```
nothing added to commit but untracked files present (use "git add" to track)
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git add README.md
```

```
warning: LF will be replaced by CRLF in README.md.
```

```
The file will have its original line endings in your working directory
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git commit
```

```
hint: Waiting for your editor to close the file... unix2dos: converting file C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/hello_world/.git/COMMIT_EDITMSG to DOS format...
```

```
dos2unix: converting file C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/hello_world/.git/COMMIT_EDITMSG to Unix format...
```

```
[main a5f1734] Adding README file
```

```
1 file changed, 1 insertion(+)
```

```
create mode 100644 README.md
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git status
```

```
On branch main
```

```
nothing to commit, working tree clean
```

Create a new file,
add and commit ...


```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ ls
```

```
README.md  hello_world.py
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ vi hello_world.py
```

Edit files and
make changes ...

```
@author JGraham
created 11/06/21
'''
```

```
hello = 'Hello world'
print(hello)
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ ls
```

```
README.md  hello_world.py
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ vi hello_world.py
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git diff hello_world.py
```

```
git diff <file>
:: show difference between
current file and last
commit.
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ ls
```

```
README.md  hello_world.py
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ vi hello_world.py
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git diff hello_world.py
```

```
warning: LF will be replaced by CRLF in hello_world.py.
```

```
The file will have its original line endings in your working directory
```

```
diff --git a/hello_world.py b/hello_world.py
```

```
index 0f33ce4..e779e75 100644
```

```
--- a/hello_world.py
```

```
+++ b/hello_world.py
```

```
@@ -5,5 +5,5 @@ My first code.
```

```
    created 11/06/21
```

```
'''
```

```
-hello = 'Hello wold'
```

```
+hello = 'Hello world'
```

```
print(hello)
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$
```

```
git diff <file>
:: show difference between
current file and last
commit.
```

```
-hello = 'Hello wold'  
+hello = 'Hello world'  
print(hello)
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ vi README.md

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git diff README.md

warning: LF will be replaced by CRLF in README.md.

The file will have its original line endings in your working directory

diff --git a/README.md b/README.md

index eb6d976..0d08e3c 100644

--- a/README.md

+++ b/README.md

@@ -1,1 @@

-This is my first repository.

+This is Jenny's first repository.

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$

```
-hello = 'Hello wold'  
+hello = 'Hello world'  
print(hello)
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)  
$ vi README.md
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)  
$ git diff README.md  
warning: LF will be replaced by CRLF in README.md.  
The file will have its original line endings in your working directory  
diff --git a/README.md b/README.md  
index eb6d976..0d08e3c 100644  
--- a/README.md  
+++ b/README.md  
@@ -1,1 @@  
-This is my first repository.  
+This is Jenny's first repository.
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)  
$ git status  
On branch main  
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:   README.md  
        modified:   hello_world.py
```

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

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)  
$ |
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git add hello_world.py
warning: LF will be replaced by CRLF in hello_world.py.
The file will have its original line endings in your working directory
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git add hello_world.py
warning: LF will be replaced by CRLF in hello_world.py.
The file will have its original line endings in your working directory
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   hello_world.py

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:   README.md
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$
```

Don't have to add
all files to commit.

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git add hello_world.py

warning: LF will be replaced by CRLF in hello_world.py.

The file will have its original line endings in your working directory

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git status

On branch main

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: hello_world.py

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: README.md

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git commit

hint: Waiting for your editor to close the file... unix2dos: converting file C:/Users/JG10/OneDrive - CEFAS/

Hub/Repos/hello_world/.git/COMMIT_EDITMSG to DOS format...

dos2unix: converting file C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/hello_world/.git/COMMIT_EDITMSG to Unix

format...

[main 832863d] Fixing typo

1 file changed, 1 insertion(+), 1 deletion(-)

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git status
On branch main
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:   README.md

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

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git status
On branch main
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:   README.md
```

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

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git log
```

```
git log <file>
:: show commit history
[for file]
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git status
On branch main
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:   README.md
```

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

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git log
commit 832863dd3215b30c3bb7d09eb6ffff229dbff113 (HEAD -> main)
Author: jenniferagraham <jennifer.graham@cefas.co.uk>
Date:   Fri Jun 11 15:47:28 2021 +0100
```

Fixing typo

```
commit a5f173410ae2855cddc0999d58c0e3dfca54a3cf
Author: jenniferagraham <jennifer.graham@cefas.co.uk>
Date:   Fri Jun 11 15:41:13 2021 +0100
```

Adding README file

```
commit a59cb3fc93e0b2efbe607c79340c13549a6cba6e
Author: jenniferagraham <jennifer.graham@cefas.co.uk>
Date:   Fri Jun 11 15:37:05 2021 +0100
```

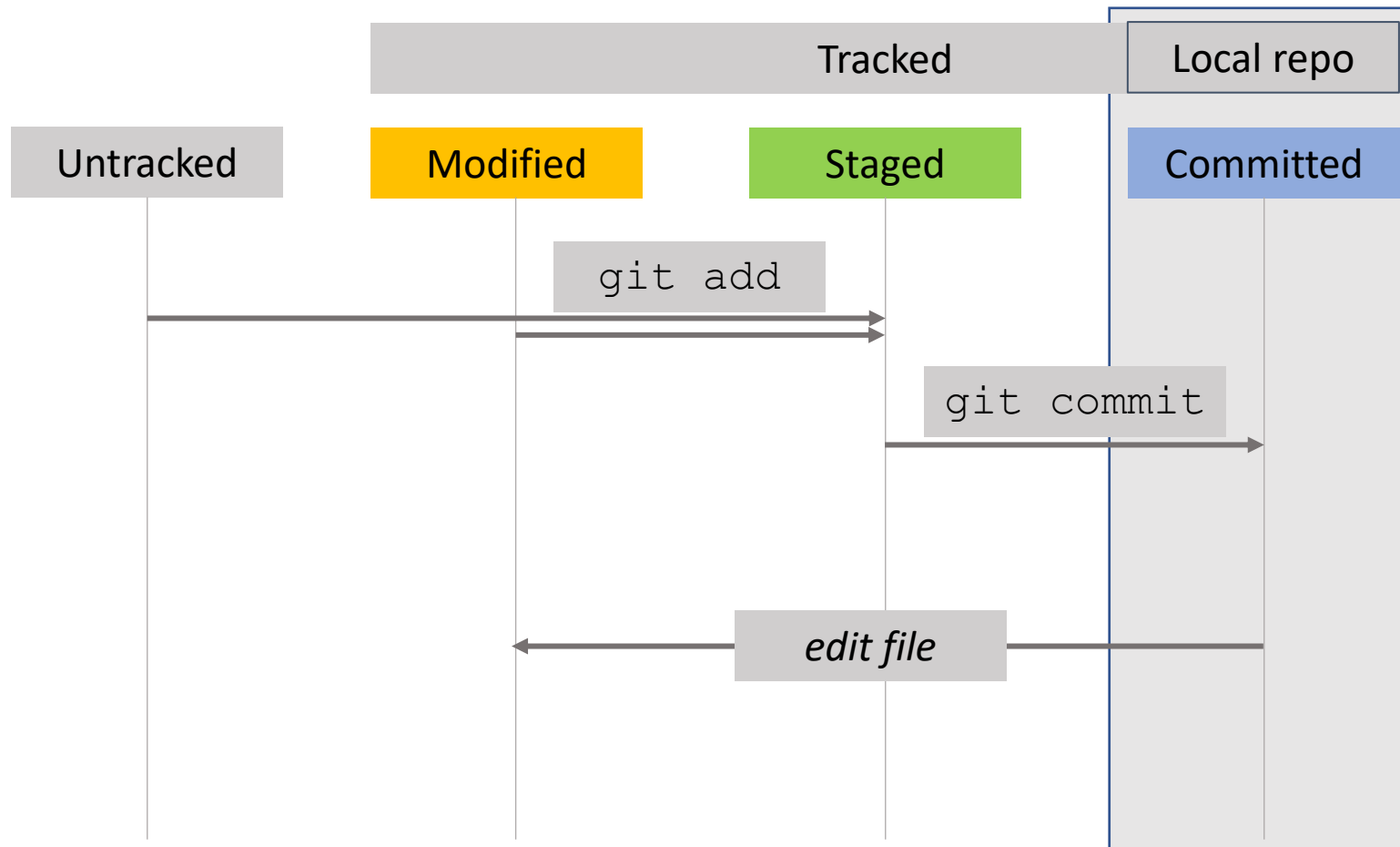
Adding first file to repository

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ |
```

`git log <file>`
:: show commit history
[for file]

```
git init                :: turn current directory into git repository.
git add <file>          :: stage the file for commit i.e., track changes.
git commit              :: confirm changes with message/explanation.

git status              :: show which files have been tracked or modified.
git diff <file>         :: show difference between current file and last commit.
git log <file>          :: show commit history [for file]
```



Aside: repository etiquette - I



- **Do not try to track all your files within a single repository!**
 - *Do not turn your OneDrive or Home into a repository!*
- **Give repositories a meaningful name.**
 - *Think before you create and name your repositories.*
 - Choosing meaningful and consistent names will help both yourself and others.
 - Consider how many repositories could be called “Data_Processing” or “python”?
 - Think, what data is being processed and why? What project are you working on?
 - Consider adding your surname as a prefix or suffix
 - Could distinguish between personal “working directories” vs. collaborative projects?
 - e.g. `jgraham_python_plotting` ?
 - Discuss with others to determine what scripts need to be included as collaborative, central repositories (avoid duplication).

Note: Further guidance for Cefas users can be found [here](#).

Aside: repository etiquette - II

- **Git & GitHub are primarily for code/methods, NOT for data storage.**
 - Large data files waste space in the repository, and changes to can't be tracked anyway!
 - However, paths/descriptions of data required should be included, either in the code or associated documentation (e.g., README files).
 - Example data files could be useful when sharing/releasing code.
 - Also, for *small* text or csv data files, you may still find git & GitHub useful for version control.

Try it yourself... [15 min]

1. Create your own repository locally:
 - Create a directory, e.g., named “My_First_Repo”.
 - Use “git init” to turn it into a repository.
2. Follow the basic workflow to add a simple txt file, make some edits, and then commit changes as you go...

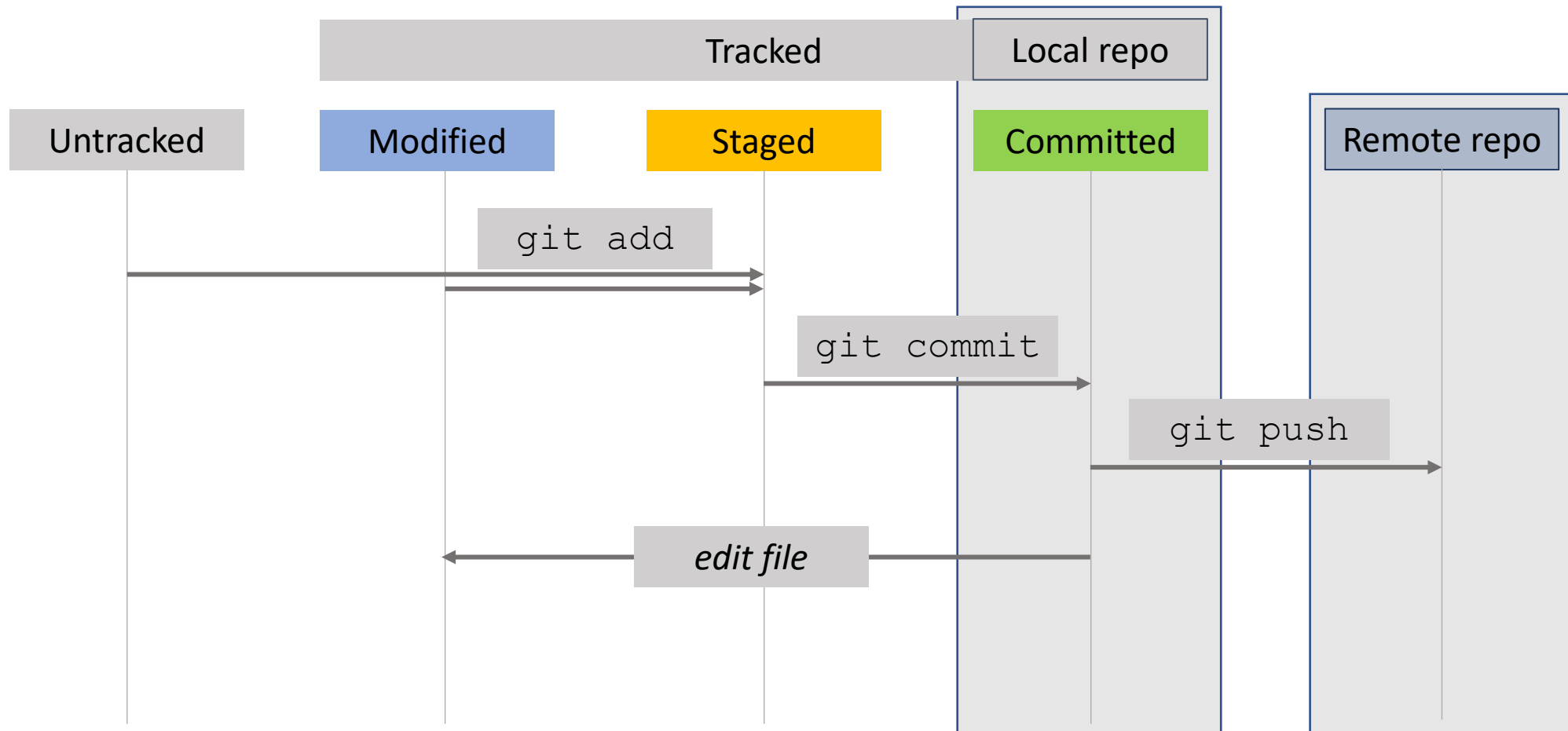
```
git init          :: turn current directory into git repository.
git add <file>    :: stage the file for commit i.e. track changes.
git commit        :: confirm changes with message/explanation.

git status        :: show which files have been tracked or modified.
git diff <file>   :: show difference between current file and last commit.
git log <file>    :: show commit history [for file]
```

Syncing/publishing changes

- Nothing will be visible on GitHub until you “push” your commits.
- You don’t need to “push” every commit (they are still recorded) but remember to do so regularly.
 - Make sure changes are visible for others to see & use.
 - Back-up your code.





```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git status
On branch main
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:   README.md

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

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git push
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git status

On branch main

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: README.md

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

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git push

fatal: No configured push destination.

Either specify the URL from the command-line or configure a remote repository using

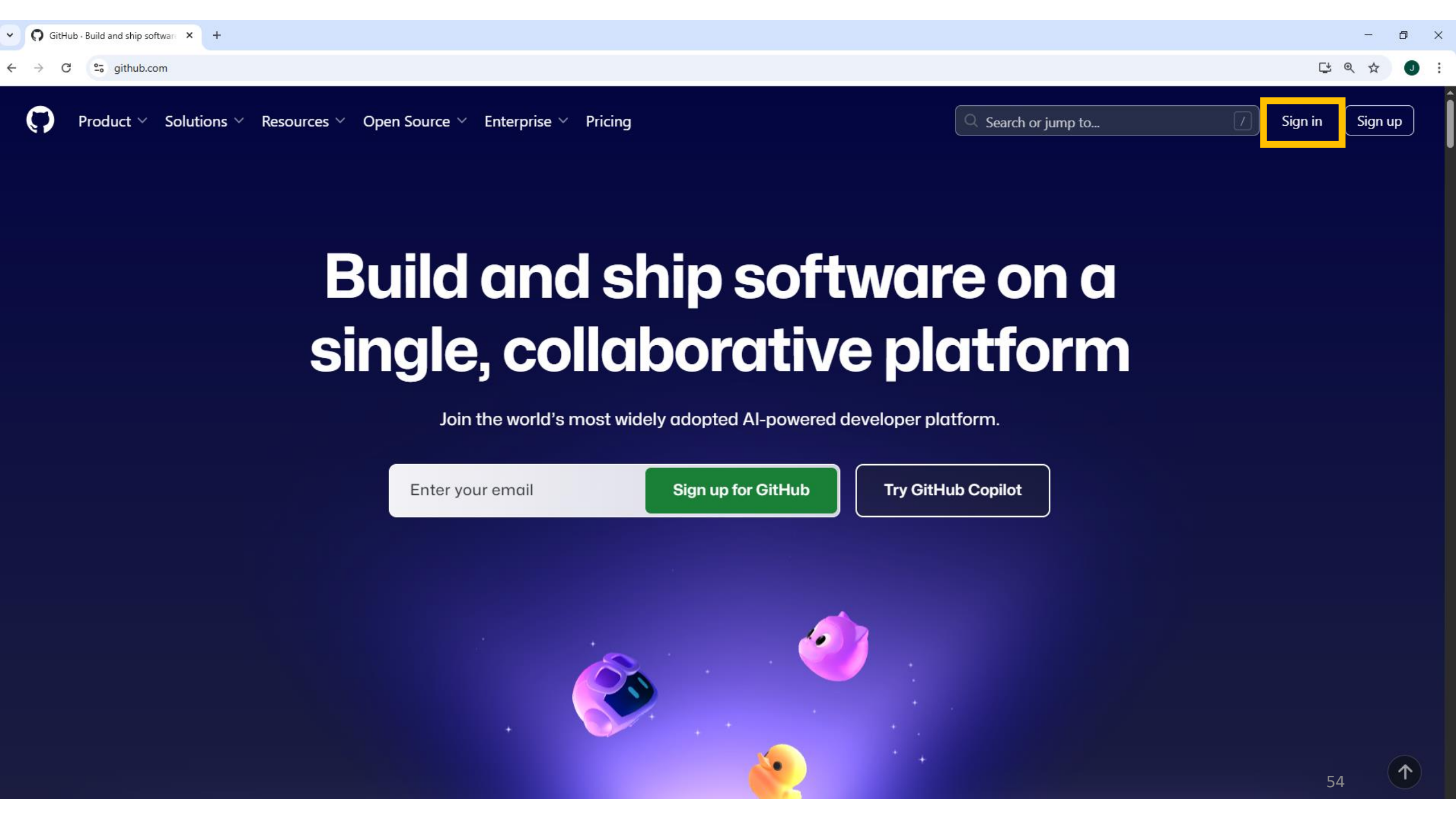
git remote add <name> <url>

and then push using the remote name

git push <name>

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$



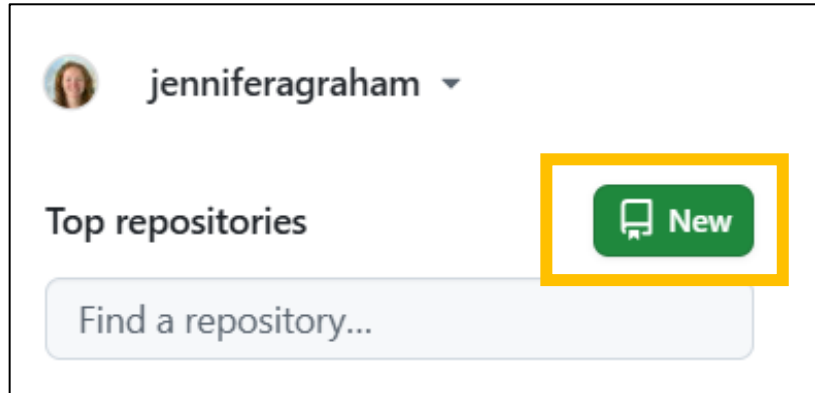
Build and ship software on a single, collaborative platform

Join the world's most widely adopted AI-powered developer platform.

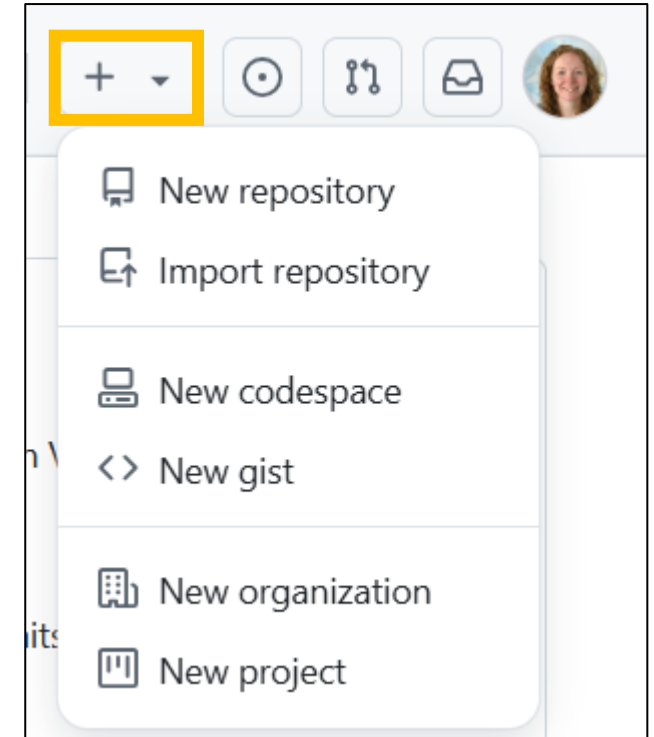
Enter your email

Sign up for GitHub

Try GitHub Copilot



Two options to create your new repository.





🔍 Type / to search



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner *

Repository name *



jenniferagraham



/

Filter...



jenniferagraham



CefasRepRes



ueapy

memorable. Need inspiration? How about [symmetrical-goggles](#) ?



Private

You choose who can see and commit to this repository.

Initialize this repository with:



Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore



🔍 Type / to search

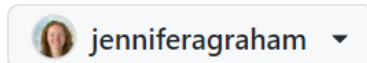


Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner *



Repository name *

hello_world

✓ hello_world is available.

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

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:



Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)



🔍 Type / to search



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner *



Repository name *

/ hello_world

✓ hello_world is available.

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

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.



Initialize this repository with:

☐

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Description (optional)

- ☐  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☒  **Private**
You choose who can see and commit to this repository.

DO NOT TICK
OR ADD
anything in this
section

Initialize this repository with:

- ☐ Add a README file
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None ▾



A license tells others what they can and can't do with your code. [Learn more about licenses.](#)



 You are creating a private repository in your personal account.

Create repository

Description (optional)

- ☐  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☒  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

- ☐ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None ▼

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

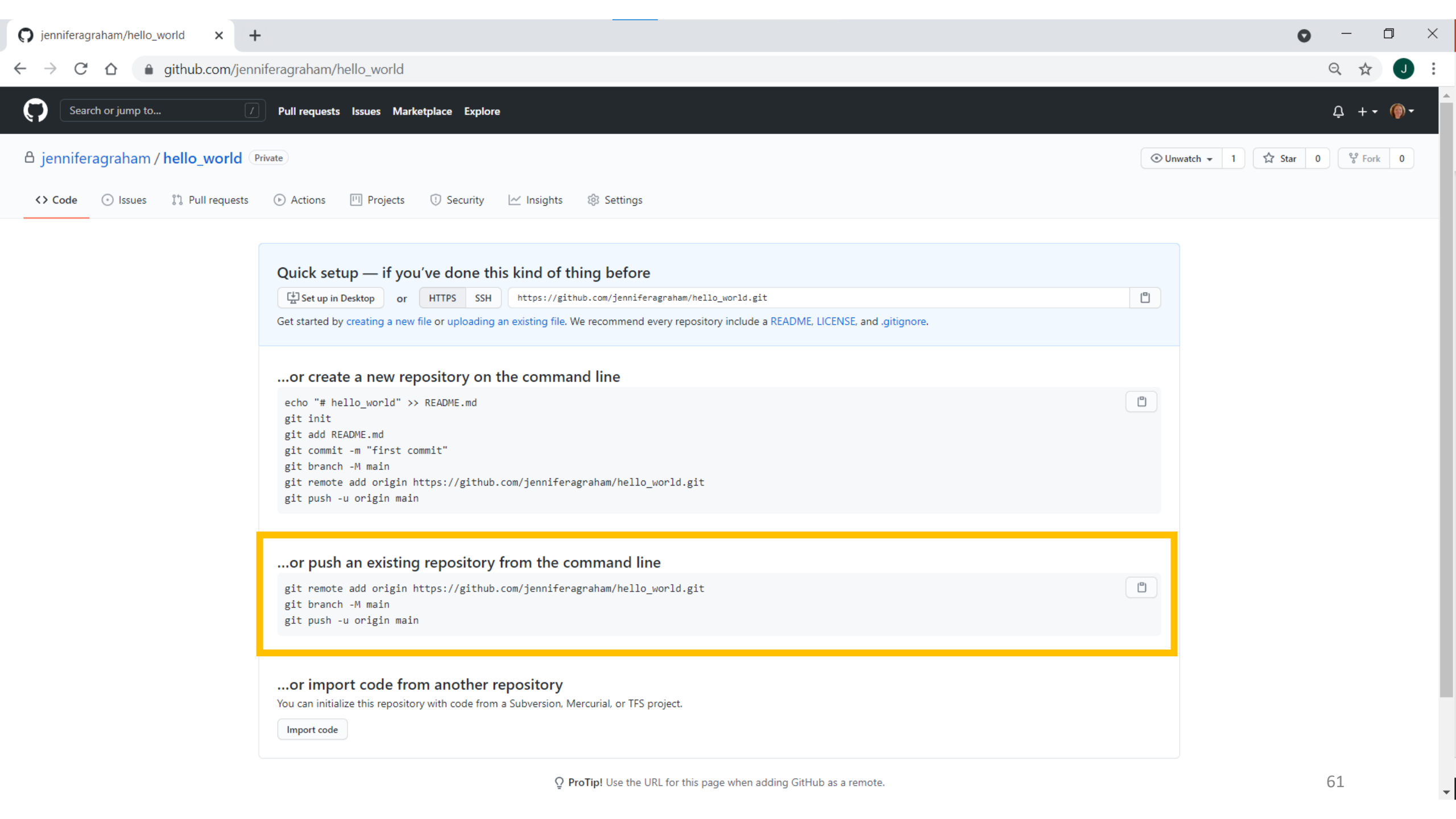
Choose a license

License: None ▼

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

 You are creating a private repository in your personal account.

Create repository



JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git push

fatal: No configured push destination.

Either specify the URL from the command-line or configure a remote repository using

```
git remote add <name> <url>
```

and then push using the remote name

```
git push <name>
```

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git remote add origin https://github.com/jenniferagraham/hello_world.git

git push -u origin main

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git branch -M main

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git push -u origin main

Enumerating objects: 9, done.

Counting objects: 100% (9/9), done.

Delta compression using up to 4 threads

Compressing objects: 100% (7/7), done.

Writing objects: 100% (9/9), 892 bytes | 297.00 KiB/s, done.

Total 9 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (1/1), done.

To https://github.com/jenniferagraham/hello_world.git

* [new branch] main -> main

Branch 'main' set up to track remote branch 'main' from 'origin'.

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$


Use commands shown
on GitHub to set
up link to remote
repository.

jenniferagraham/hello_world: My

+

github.com/jenniferagraham/hello_world

☆J





Search or jump to...

Pull requests

Issues

Marketplace

Explore

+

jenniferagraham / hello_world

Private

Unwatch

1

Star

0

Fork

0

<> Code

Issues

Pull requests

Actions

Projects

Security

Insights

Settings

main


1 branch

0 tags

Go to file

Add file

Code

 jenniferagraham

Fixing typo

832863d 24 minutes ago 3 commits

README.md

Adding README file

31 minutes ago

hello_world.py

Fixing typo

24 minutes ago

README.md

This is my first repository.

About

My first repository

Readme

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages

Python 100.0%

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git status

On branch main

Your branch is up to date with 'origin/main'.

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: README.md

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

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$ git diff README.md

warning: LF will be replaced by CRLF in README.md.

The file will have its original line endings in your working directory

diff --git a/README.md b/README.md

index eb6d976..0d08e3c 100644

--- a/README.md

+++ b/README.md

@@ -1,1 @@

-This is my first repository.

+This is Jenny's first repository.

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)

\$


```
--- a/README.md
```

```
+++ b/README.md
```

```
@@ -1 +1 @@
```

```
-This is my first repository.
```

```
+This is Jenny's first repository.
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git add README.md
```

```
warning: LF will be replaced by CRLF in README.md.
```

```
The file will have its original line endings in your working directory
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git commit
```

```
hint: Waiting for your editor to close the file... unix2dos: converting file C:/Users/JG10/OneDrive - CEFAS/
```

```
Hub/Repos/hello_world/.git/COMMIT_EDITMSG to DOS format...
```

```
dos2unix: converting file C:/Users/JG10/OneDrive - CEFAS/GitHub/Repos/hello_world/.git/COMMIT_EDITMSG to Uni
```

```
format...
```

```
[main 5fa034d] Specify Jenny's repository
```

```
1 file changed, 1 insertion(+), 1 deletion(-)
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$ git push
```

```
Enumerating objects: 5, done.
```

```
Counting objects: 100% (5/5), done.
```

```
Delta compression using up to 4 threads
```

```
Compressing objects: 100% (2/2), done.
```

```
Writing objects: 100% (3/3), 333 bytes | 333.00 KiB/s, done.
```


```
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
```

```
To https://github.com/jenniferagraham/hello_world.git
```

```
832863d..5fa034d main -> main
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
```

```
$
```

 jenniferagraham / hello_world

Private

👁 Unwatch

1

☆ Star

0

🍴 Fork

0

<> Code

⌚ Issues

🔗 Pull requests

▶ Actions

📁 Projects

🛡 Security

📊 Insights

⚙ Settings

🔗 main


🔗 1 branch

🏷 0 tags

Go to file


Add file

↓ Code

 jenniferagraham Specify Jenny's repository


5fa034d 2 minutes ago

🕒 4 commits

 README.md

Specify Jenny's repository


2 minutes ago

 hello_world.py

Fixing typo

27 minutes ago

README.md



This is Jenny's first repository.

About

⚙

My first repository

📖 Readme

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

● Python 100.0%


```
git init          :: turn current directory into git repository.
git add <file>    :: stage the file for commit i.e., track changes.
git commit        :: confirm changes with message/explanation.
git push          :: publish/back-up changes on GitHub (remote server).

git status        :: show which files have been tracked or modified.
git diff <file>   :: show difference between current file and last commit.
git log <file>    :: show commit history [for file]
```

Try this yourself... [15 min]

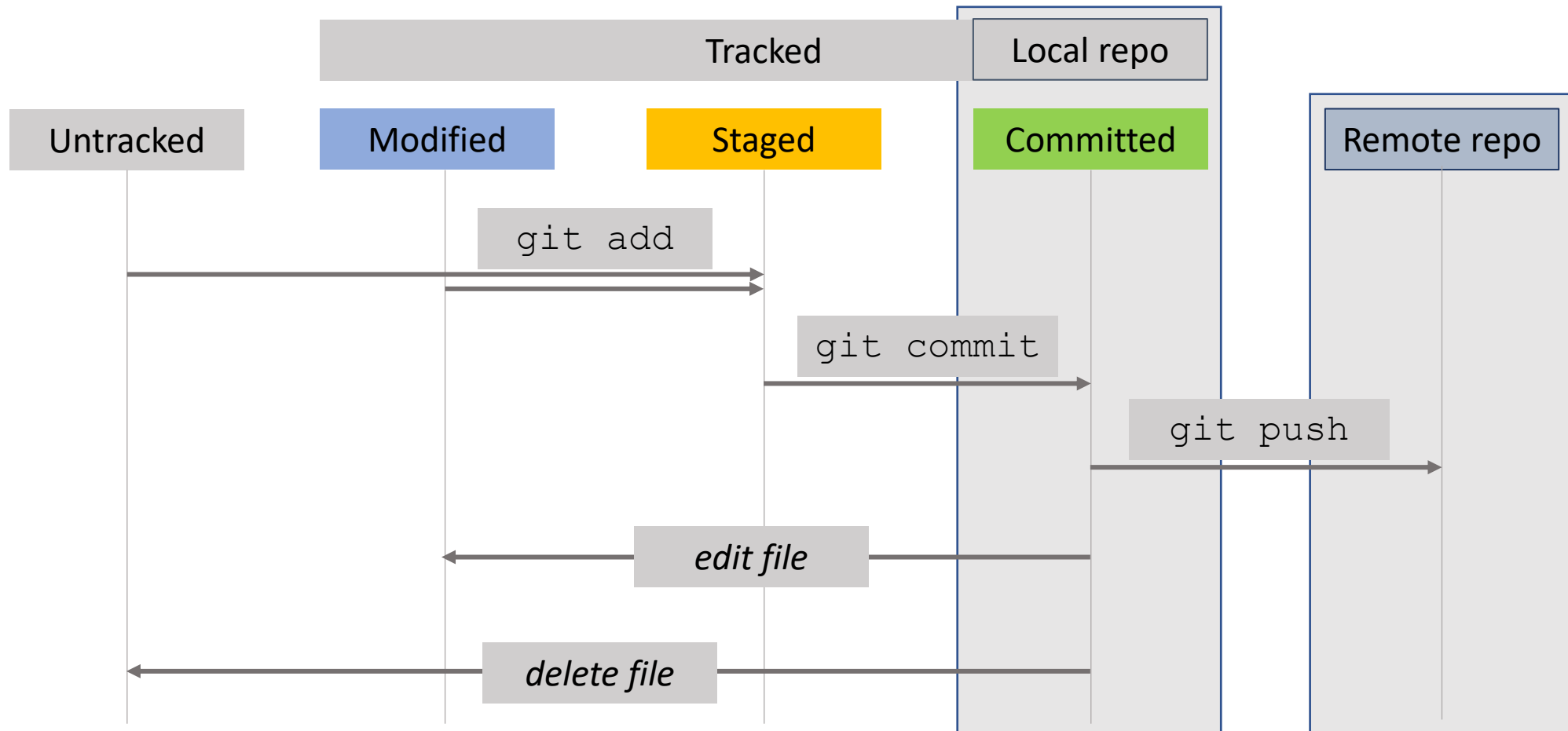
- Push your repository to GitHub, and view changes.
 - Create an **empty** repository on GitHub (with the name of your local repository).
 - Use the commands that GitHub provides to push your local repository up to the GitHub site.

```
git init          :: turn current directory into git repository.
git add <file>    :: stage the file for commit i.e., track changes.
git commit        :: confirm changes with message/explanation.
git push          :: publish/back-up changes on GitHub (remote server).

git status        :: show which files have been tracked or modified.
git diff <file>   :: show difference between current file and last commit.
git log <file>    :: show commit history [for file]
```

Further useful commands...

```
git rm <file> :: delete file and remove from repository.  
git rm --cached <file> :: delete file from repository, but keep  
local copy.  
  
git mv <file> <new_name> :: rename file, and keep tracking.  
  
git remote -v :: check url for tracked remote repository  
  
git <command> --help :: open help documentation for any command!
```



Summary: Terminology

- Repository: Folder containing collection of files to be stored/tracked (any sort of text/code).
- Local repository: Repository being tracked on your local machine (i.e. desktop, hpc).
- Remote repository: a separate copy of your repository stored in a remote location e.g. on GitHub.

- Add: Tells git what files to track / stage files for commit.
- Commit: Saves a snapshot of files, with explanation for any changes.
- Push: Updates from your local machine synced to the remote location.

- Branch: Repositories can be branched into parallel copies, to safely test large developments to the code.
 - The default/central branch is named the “main” (or “master”) branch.
 - This central branch should always contain a working version of the code.
- Head: Latest commit made, tip of the branch in your local repository (current working revision)

Aside: main vs master

- GitHub now uses “main” as the default central branch name (rather than master)
 - In 2020, other software companies announced similar changes.
- If you’re working with older software, or creating repositories locally with an older version of git, you may find that “master” is still the default branch name.
 - Your default will depend on your local git settings.
 - You may remember this being an option while installing git on your PC?
- There are a few different ways you can update this...

How to change the “central” branch name?

If you have just setup a new repository on GitHub, but don't want to rename anything locally:

```
git remote add origin <GitHub_Path>
git push -u origin master
```

If you have just setup a new repository on GitHub, and want to rename your current local branch to main:

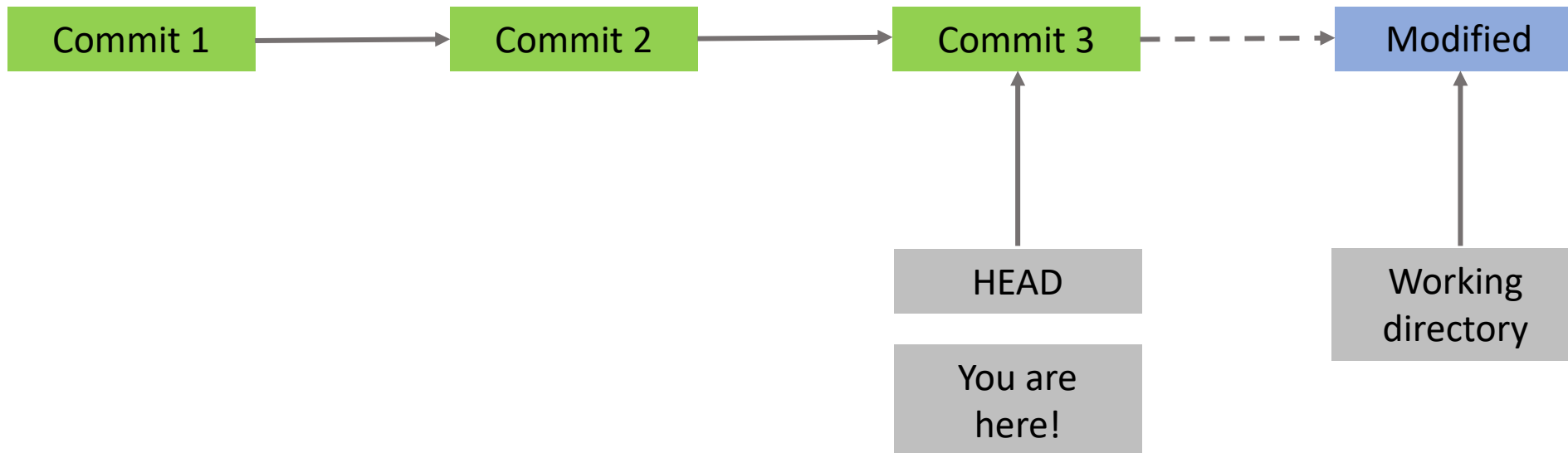
```
git remote add origin <GitHub_Path>
# Rename the current branch to main
git branch -M main
git push -u origin main
```

If you rename an existing repository on GitHub, and want to update your local repository

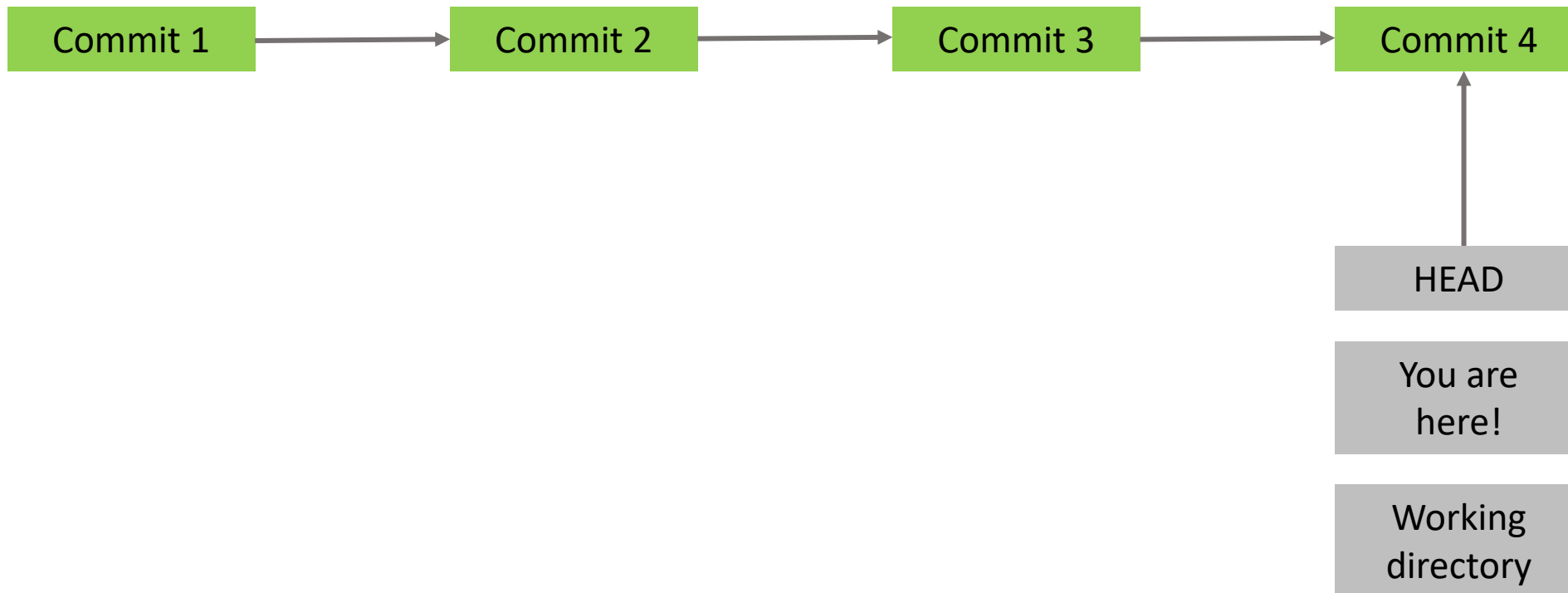
```
# Rename the master branch to main
git branch -m master main
git fetch origin
git branch -u origin/main main
git remote set-head origin -a
```

NB. GitHub will give tips for the above when creating a new repository.

Where is my HEAD?



Where is my HEAD?



More useful tips...

Contents of .git folder

- When a repository is created, this creates a .git folder in the directory.
- .git contains e.g.:

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ ls -a
./  ../  .git/  README.md  hello_world.py

JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ ls .git
COMMIT_EDITMSG  HEAD  config  description  hooks/  index  info/  logs/  objects/  refs/
```

- What are these files?
 - They contain everything that git needs to know about the repository!
 - Whenever you run a command, it will use and/or edit files here.

.gitignore



- .gitignore files tell git what to ignore!
- Can ignore any file, directory, or file extensions...
 - e.g. executables, log files, data, figures... (any secrets)
 - Standard examples are provided on GitHub.
- Simple example file:

| Name | Status | Date modified | Type |
|----------------|--------|------------------|---------------|
| .git | ✓ | 02/07/2019 14:55 | File folder |
| .gitattributes | ✓ | 27/06/2019 11:32 | Text Document |
| .gitignore | ✓ | 27/06/2019 11:46 | Text Document |
| DATA.nc | ✓ | 26/11/2018 16:36 | NC File |
| Figure.PNG | ✓ | 26/01/2018 15:54 | PNG File |
| hello_world.py | ✓ | 25/06/2019 14:07 | Python File |
| README.md | ✓ | 27/06/2019 12:11 | MD File |

```
C:\Users\JG10\OneDrive - CEFAS\GitHub\Repos\jag_hello_desktop\.gitignore
File Edit Search View Encoding Language Settings Macro Run Plugins
1 *.nc
2 *PNG
3
```

git config

- Check global username and email settings: `git config --global -l`

```
JG10@1H959K3 MINGW64 ~  
$ git config --global -l  
core.editor=notepad  
user.name=Jennifer Graham  
user.email=jennifer.graham@cefas.gov.uk
```

- Change default editor?

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world  
$ git config --get core.editor  
vi  
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world  
$ git config --global core.editor notepad  
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world  
$ git config --get core.editor  
notepad
```


git config

- Check local repository settings e.g. path to remote repository

```
git config --local -l
```

```
JG10@G6W1YF2 MINGW64 ~/OneDrive/GitHub/Repos/hello_world (main)
$ git config --local -l
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.ignorecase=true
remote.origin.url=https://github.com/jenniferagraham/hello_world.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
branch.main.remote=origin
branch.main.merge=refs/heads/main
```

GitHub settings

- Global settings
 - Your account details, notifications, organisation links, etc.
- Repository settings
 - Access, features, visibility, etc.

General

github.com/jenniferagraham/hello_world/settings

github.com / jenniferagraham / hello_world

Type to search

CodeIssuesPull requestsActionsProjectsSecurityInsightsSettings

General

Access

Collaborators

Code and automation

Branches

Tags

Rules

Actions

Webhooks

Codespaces

Pages

Security

Code security and analysis

Deploy keys

Secrets and variables

Integrations

GitHub Apps

Email notifications

General

Repository name

hello_world

Rename

☐ Template repository

Template repositories let users generate new repositories with the same directory structure and files. [Learn more about template repositories.](#)

☐ Require contributors to sign off on web-based commits

Enabling this setting will require contributors to sign off on commits made through GitHub's web interface. Signing off is a way for contributors to affirm that their commit complies with the repository's terms, commonly the [Developer Certificate of Origin \(DCO\)](#). [Learn more about signing off on commits.](#)

Default branch

The default branch is considered the "base" branch in your repository, against which all pull requests and code commits are automatically made, unless you specify a different branch.

main

Features

☐ Wikis

Wikis host documentation for your repository.

Upgrade or make this repository public to enable Wikis

GitHub Wikis is a simple way to let others contribute content. Any GitHub user can create and edit pages to use for documentation, examples, support, or anything you wish.

Upgrade

[Learn more about wikis](#)

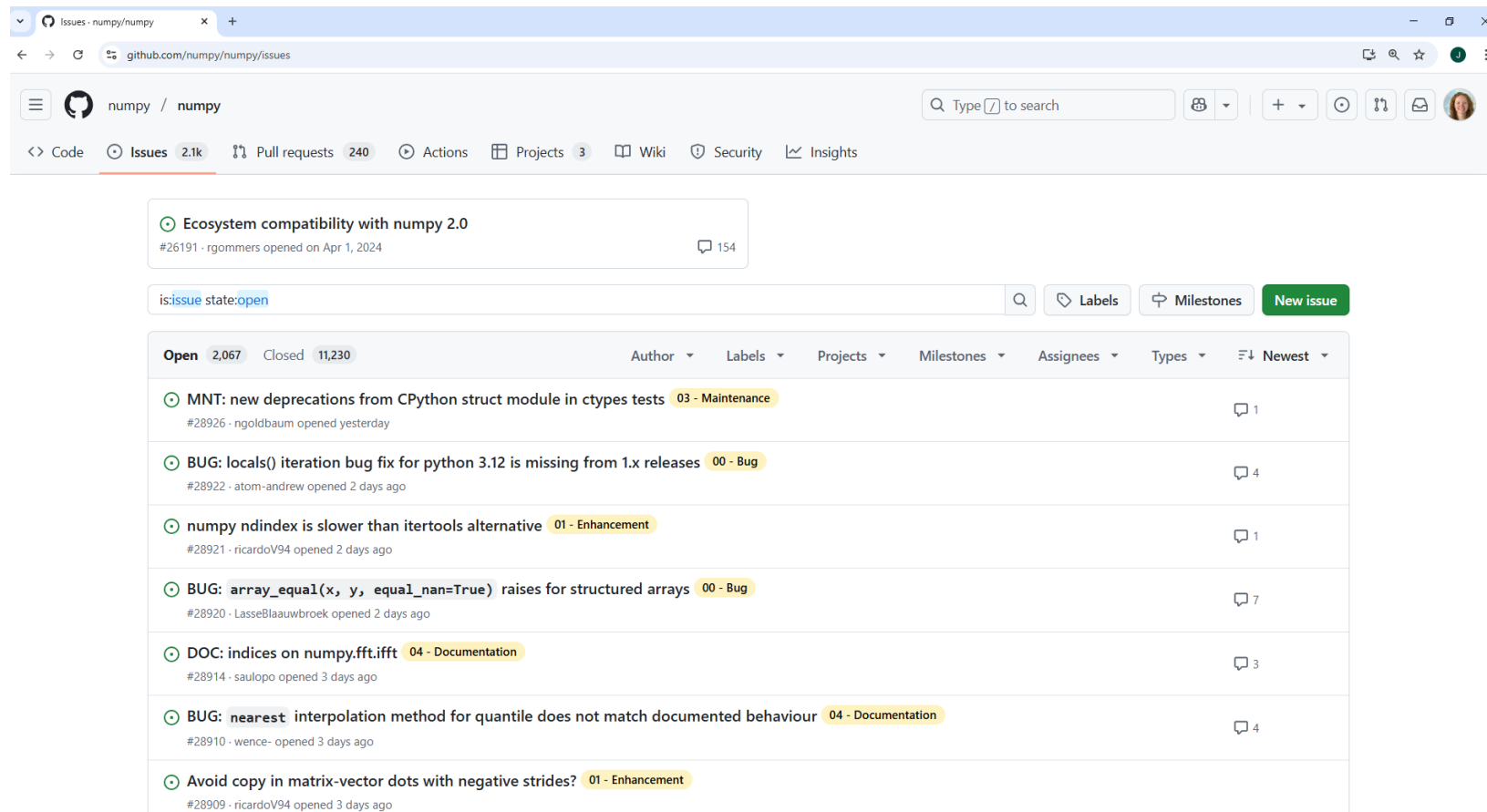
84

Adding documentation

- README
 - Automatically the front page of your repository.
 - [Markdown](#) formatting can be used
- Wiki
 - Broader information on use of repository.
 - Only available within public repositories or organisations.
 - e.g. https://github.com/CefasRepRes/Git_Training/wiki/
- Issues
 - Flagging development stages or known problems.
- Projects
 - Organise work/issues on repositories?
- Some repository tracking info only available within organisations or paid accounts
 - More info here: <https://github.com/pricing>

Issues

- Useful to flag development stages or known problems/bugs.
- Can be referred to elsewhere e.g. in commit messages.



The screenshot shows the GitHub Issues page for the `numpy/numpy` repository. The browser address bar shows `github.com/numpy/numpy/issues`. The repository navigation bar includes links for Code, Issues (2.1k), Pull requests (240), Actions, Projects (3), Wiki, Security, and Insights. A search bar is present with the text "Type / to search".

The main content area displays a list of issues. The first issue is "Ecosystem compatibility with numpy 2.0" (#26191) by rgommers, opened on Apr 1, 2024, with 154 comments. Below this is a filter bar with the text "is:issue state:open" and buttons for "Labels", "Milestones", and "New issue".

The issue list is sorted by "Newest" and shows the following issues:

| Issue Title | Category | Comments |
|--|--------------------|----------|
| MNT: new deprecations from CPython struct module in ctypes tests | 03 - Maintenance | 1 |
| BUG: locals() iteration bug fix for python 3.12 is missing from 1.x releases | 00 - Bug | 4 |
| numpy ndindex is slower than itertools alternative | 01 - Enhancement | 1 |
| BUG: array_equal(x, y, equal_nan=True) raises for structured arrays | 00 - Bug | 7 |
| DOC: indices on numpy.fft.iff | 04 - Documentation | 3 |
| BUG: nearest interpolation method for quantile does not match documented behaviour | 04 - Documentation | 4 |
| Avoid copy in matrix-vector dots with negative strides? | 01 - Enhancement | |