

# Collaboration with Git

Patrick McCann

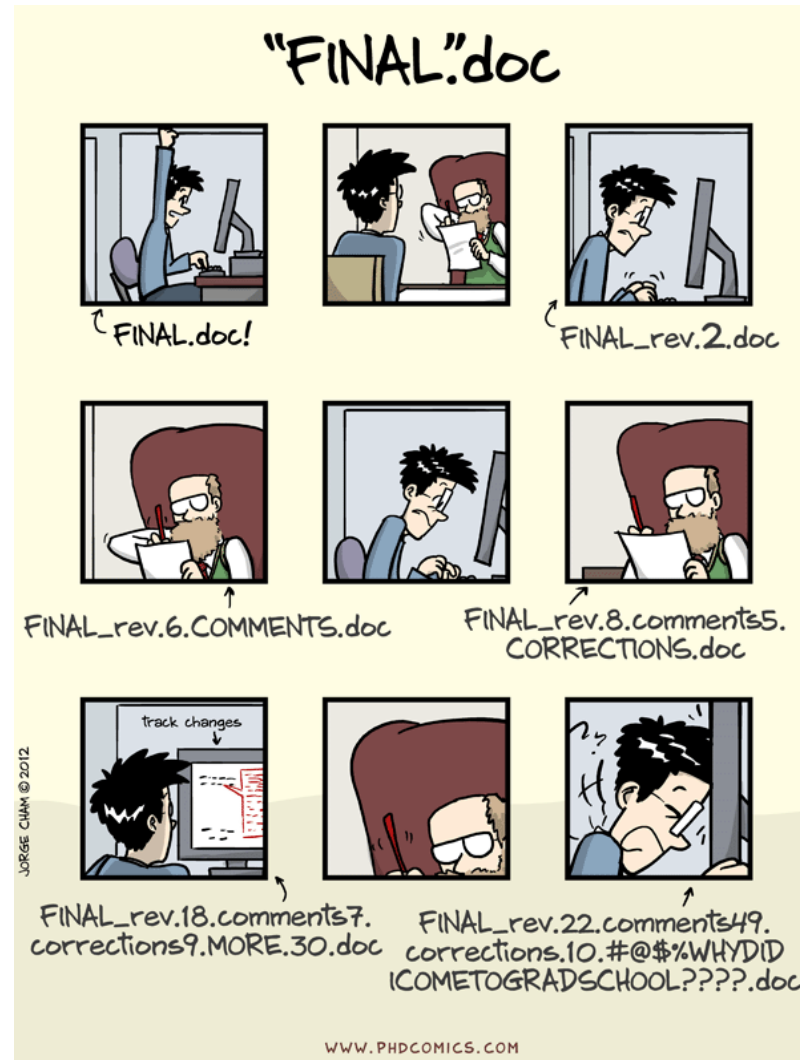
Research Computing, University of St Andrews

# Software Carpentry

Much of this presentation will draw on the [Software Carpentry](#) lesson [Version Control with Git](#).

That lesson goes into topics in more detail, with examples and exercises. There are regular Software Carpentry workshops at the University, open to all researchers.

# Why use Version Control?



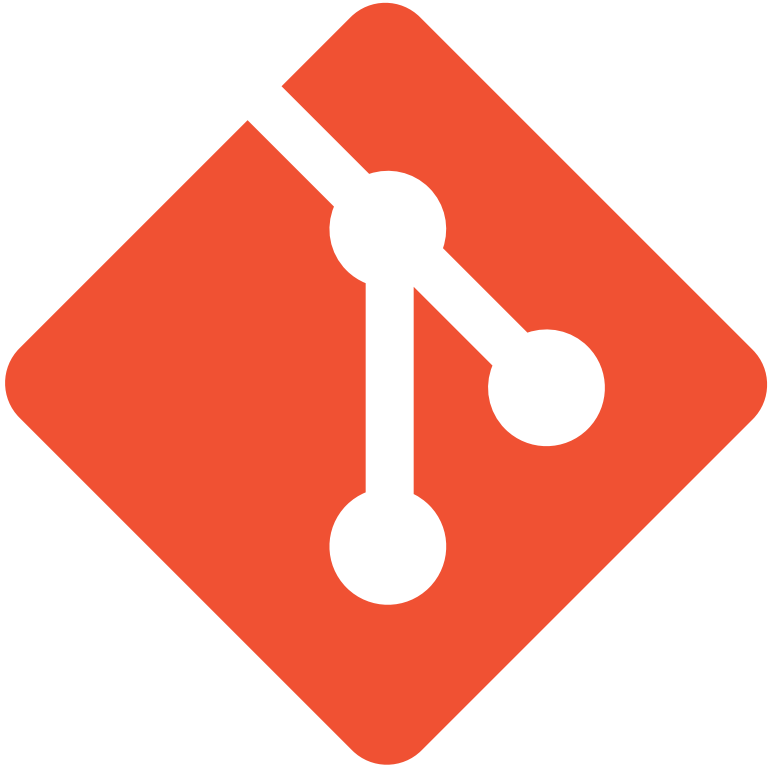
Version control systems record the details of changes made to a base version of a document or documents.

As well as allowing you to track changes over time - and to move back and forward between versions - they allow collaborators to maintain differing versions and provide mechanisms for resolving those differences.

You get to decide what changes get grouped together in a *commit*, marking a new version.

A project's commit history and metadata make up a *repository*. Repositories can be kept in sync between different computers.

# Why use Git?



# git

Version control systems have been around since the 1980s - you may have heard of e.g. CVS or Subversion.

Modern systems like **Git** and Mercurial are *distributed*, so they don't need a central server to host repositories.

Git has become the de facto standard.



# Why use GitHub?



# GitHub

[GitHub.com](#) has become the most popular platform for hosting Git Repositories - especially public repositories for open-source software.

Others platforms include [BitBucket](#) and [GitLab.com](#).

The University has an instance of **GitLab** (VPN) available to researchers - this is not available to users outside the University.



GitLab

GitHub also provides some additional features which are particularly useful to academic researchers.

# Using Git



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










There are a number of ways to work with Git on your computer:

- The command-line interface referenced in the XKCD cartoon
- Terminal based user interfaces like [gitui](#) and [lazygit](#)
- Graphical user interfaces like [GitHub Desktop](#) and [SourceTree](#)
- As a feature (or plugin) of development tools and editors like RStudio.








Other user interfaces can be considered as wrappers around the command-line interface. They tend to hide some of the details of how Git works.

This presentation includes screenshots from GitHub Desktop alongside the equivalent commands.

# A Git Repository

-  my-project
  -  .git
  -  data
  -  src
  -  test
  -  .gitignore
  -  LICENSE.txt
  -  README.md
  -  run.sh



-  my-project
  -  .git
  -  data
  -  src
  -  test
  -  .gitignore
  -  LICENSE.txt
  -  README.md
  -  run.sh

**.git** is where Git stores the metadata about the project.

We (almost) never edit its contents directly.

Technically, **.git** is the *Git Repository*, but you will often see **my-project** described as such.

-  my-project
  -  .git
  -  data
  -  src
  -  test
  -  **.gitignore**
  -  LICENSE.txt
  -  README.md
  -  run.sh

Many Git-managed projects will have a **.gitignore** file, which allows us to exclude some files from version control.

# Initialising

```
$ cd /Users/paddy/Documents/GitHub  
$ mkdir my-project  
$ cd my-project  
$ git init
```

Create a New Repository

×

Name

my-project

Description

Local Path

/Users/paddy/Documents/GitHub

Choose...

☐ Initialize this repository with a README

Git Ignore

None

▼

License

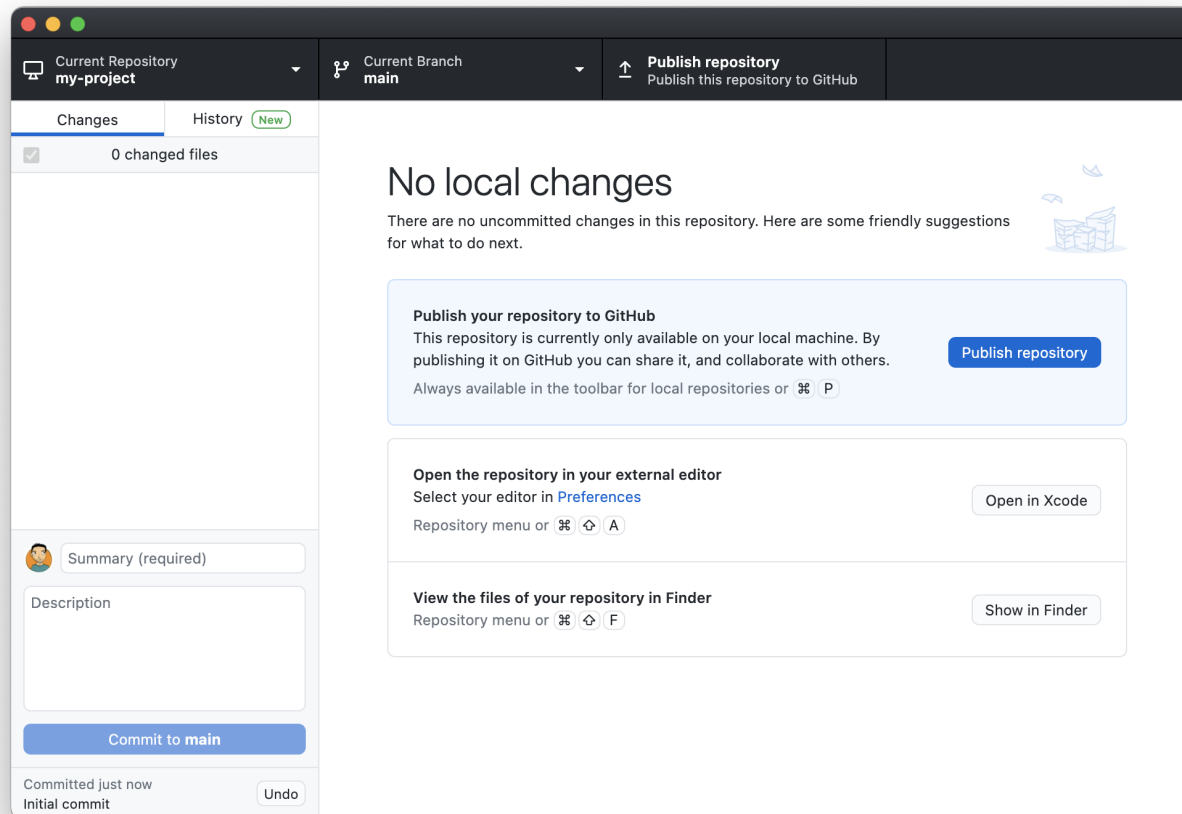
None

▼

Cancel

Create Repository

```
$ ls -a
.      ..      .git
$ git status
On branch main
nothing to commit, working tree clean
```



# Adding and Committing

We can add a file to our project folder using any text editor or, indeed, any piece of software which allows us to save files.

Here, we use a text editor to create a README file using markdown syntax and save it as `README.md`.

```
$ ls -a  
.  
..  
.git  
README.md
```

```
$ cat README.md
```

```
# My Project
```

This is an example project to illustrate the use of Git for collaboration in a research context.

Current Repository  
my-project

Current Branch  
main

↑

Publish repository

Publish this repository to GitHub

Changes 1

History New

README.md

⚙️

New

+

✓

1 changed file

✓

README.md

+

1

2

3

4

@@ -0,0 +1,4 @@

1 +# My Project

2 +

3 +This is an example project to illustrate the use of Git for

4 +collaboration in a research context.

Create README.md

Description

Commit to main

Committed 3 days ago

Initial commit

Undo

```
$ 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
$ git add README.md
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   README.md
```



```
$ git commit -m 'Adds basic README describing project'
```



basic README describing project

Description

**Commit to main**

# Viewing the Log

The screenshot displays the GitHub web interface for a repository named "my-project". The top navigation bar includes the repository name, the current branch "main", and a "Publish repository" button. Below this, the "History" tab is selected, showing a list of commits on the left and a diff view on the right.

**Commit History (Left Panel):**

- Adds bash script to run analysis** (Patrick McCann • 11m)
- Adds test script** (Patrick McCann • 11m)
- Adds analysis script** (Patrick McCann • 14m)
- Adds input data** (Patrick McCann • 15m)
- Adds MIT license** (Patrick McCann • 15m)
- Adds basic README describing proj...** (Patrick McCann • 3d)
- Initial commit** (Patrick McCann • 6d)

**Diff View (Right Panel):**

The diff view shows the changes for the commit "Adds bash script to run analysis" by Patrick McCann. The file "run.sh" is highlighted with a green background. The diff shows the following changes:

```
@@ -0,0 +1,4 @@
1 +#!/bin/bash
2 +
3 +cp data/input.csv data/output.csv
4 +python src/analysis.py
```

```
$ git log
commit c8bacfbdfd15e5f0924dde8461662ba8fdb676da
Author: Patrick McCann <pgm5@st-andrews.ac.uk>
Date:   Mon Nov 1 15:36:13 2021 +0000
```

Adds bash script to run analysis

```
commit 3fa305a9d4e76e9a4bb58f724a15c2f5ae032edc
Author: Patrick McCann <pgm5@st-andrews.ac.uk>
Date:   Mon Nov 1 15:35:39 2021 +0000
```

Adds test script

```
commit 382abfc258f398954ad897e52ff224a75188a7ca
Author: Patrick McCann <pgm5@st-andrews.ac.uk>
Date:   Mon Nov 1 15:32:32 2021 +0000
```

```
$ git diff HEAD~1 HEAD
diff --git a/run.sh b/run.sh
new file mode 100755
index 0000000..a84485c
--- /dev/null
+++ b/run.sh
@@ -0,0 +1,4 @@
+#!/bin/bash
+
+cp data/input.csv data/output.csv
+python src/analysis.py
```



	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
○	AAAAAAAAA	3 HOURS AGO
○	ADKFJSLKDFJSDKLFJ	3 HOURS AGO
○	MY HANDS ARE TYPING WORDS	2 HOURS AGO
○	HAAAAAAAAAANDS	2 HOURS AGO

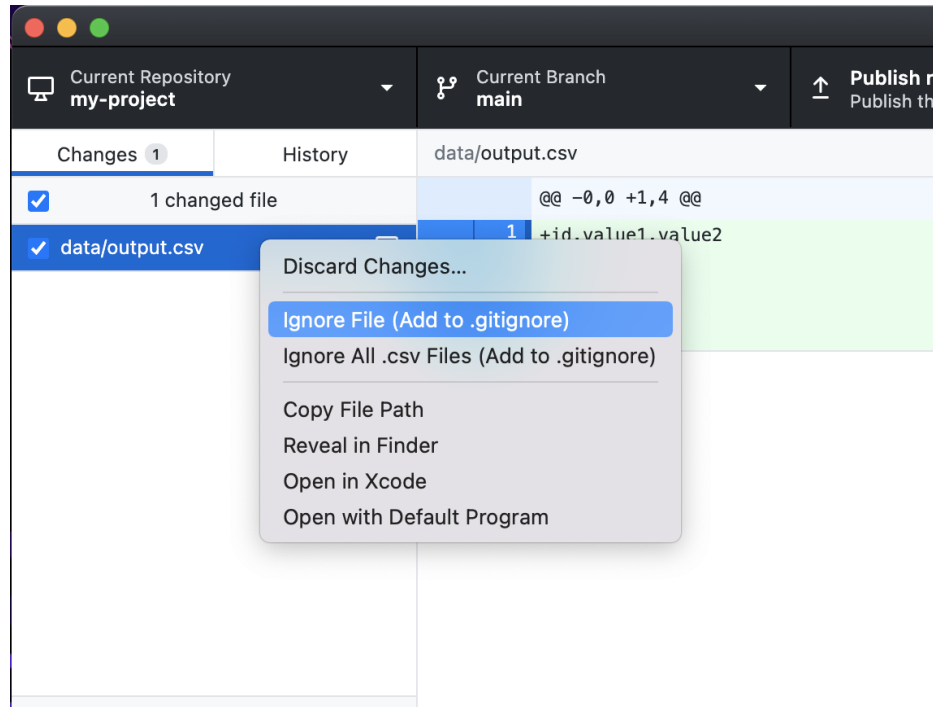
AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

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

# Ignoring Things





There are situations where we have a file or files in our repository which we don't want to place under version control e.g.

- The results of analysis.
- Anything containing a password or other sensitive data.
- Files created by your tools which aren't really part of the project.



```
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    data/output.csv

nothing added to commit but untracked files present (use "git add" to track)
$ git ignore data/output.csv
Adding pattern(s) to: .gitignore
... adding 'data/output.csv'
```

 Current Repository <b>my-project</b>		 Current Branch <b>main</b>		 <b>Publish</b> Publish th
Changes <b>1</b>	History	.gitignore		
<input checked="" type="checkbox"/> 1 changed file			@@ -0,0 +1 @@	
<input checked="" type="checkbox"/> .gitignore 		1	+data/output.csv	

```
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitignore

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



# Remotes

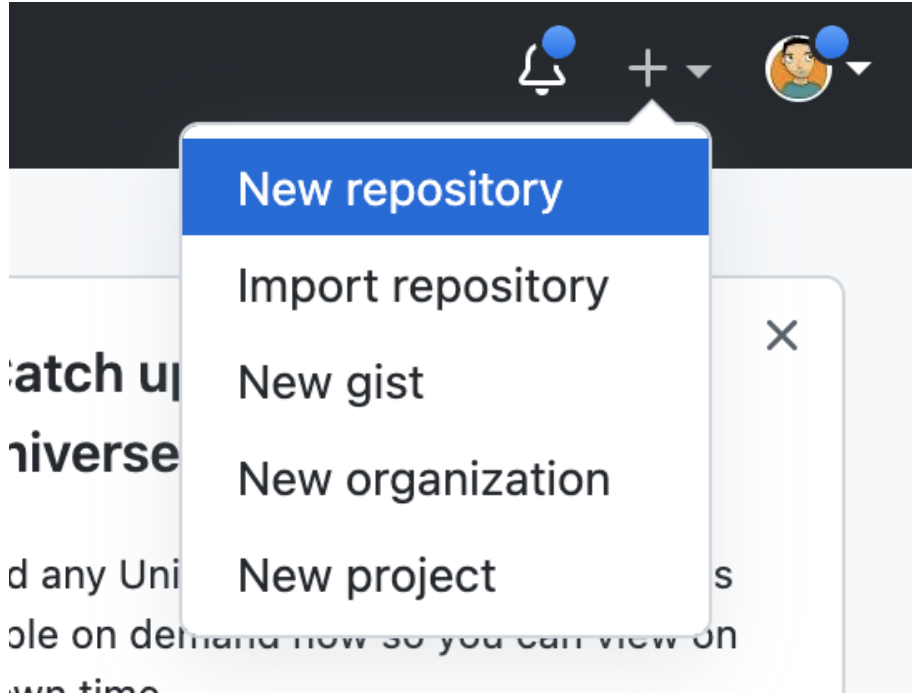
So far, everything has been on our own computer. This is useful, but Git really comes into its own when collaborating with others.

To do that, we need to keep our projects in sync between multiple computers (also useful if you work from multiple computers yourself).

You can, in principle, have your Git on your computer communicate directly with your colleagues' to keep things in sync, but it's generally easier to sync with another location to which you all have access.

That other location could be a desktop computer in your office or it could be GitHub (or GitLab) - as far as Git is concerned there isn't really a difference.

# Creating a repo on GitHub.com



## Create a new repository

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

### Repository template

Start your repository with a template repository's contents.

No template ▾

Owner \*

 pgmccann ▾

Repository name \*

/ my-project ✓

Great repository names are short and memorable. Need inspiration? How about [cautious-pancake?](#)

Description (optional)

Demonstration project for talk on Collaboration using Git

☒  **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:

Skip this step if you're importing an existing repository.

☐ **Add a README file**

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

☐ **Add .gitignore**


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

☐ **Choose a license**

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

Create repository

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

 Set up in Desktop

or

HTTPS

SSH

git@github.com:pgmccann/my-project.git



Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

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

```
echo "# my-project" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:pgmccann/my-project.git
git push -u origin main
```



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

```
git remote add origin git@github.com:pgmccann/my-project.git
git branch -M main
git push -u origin main
```

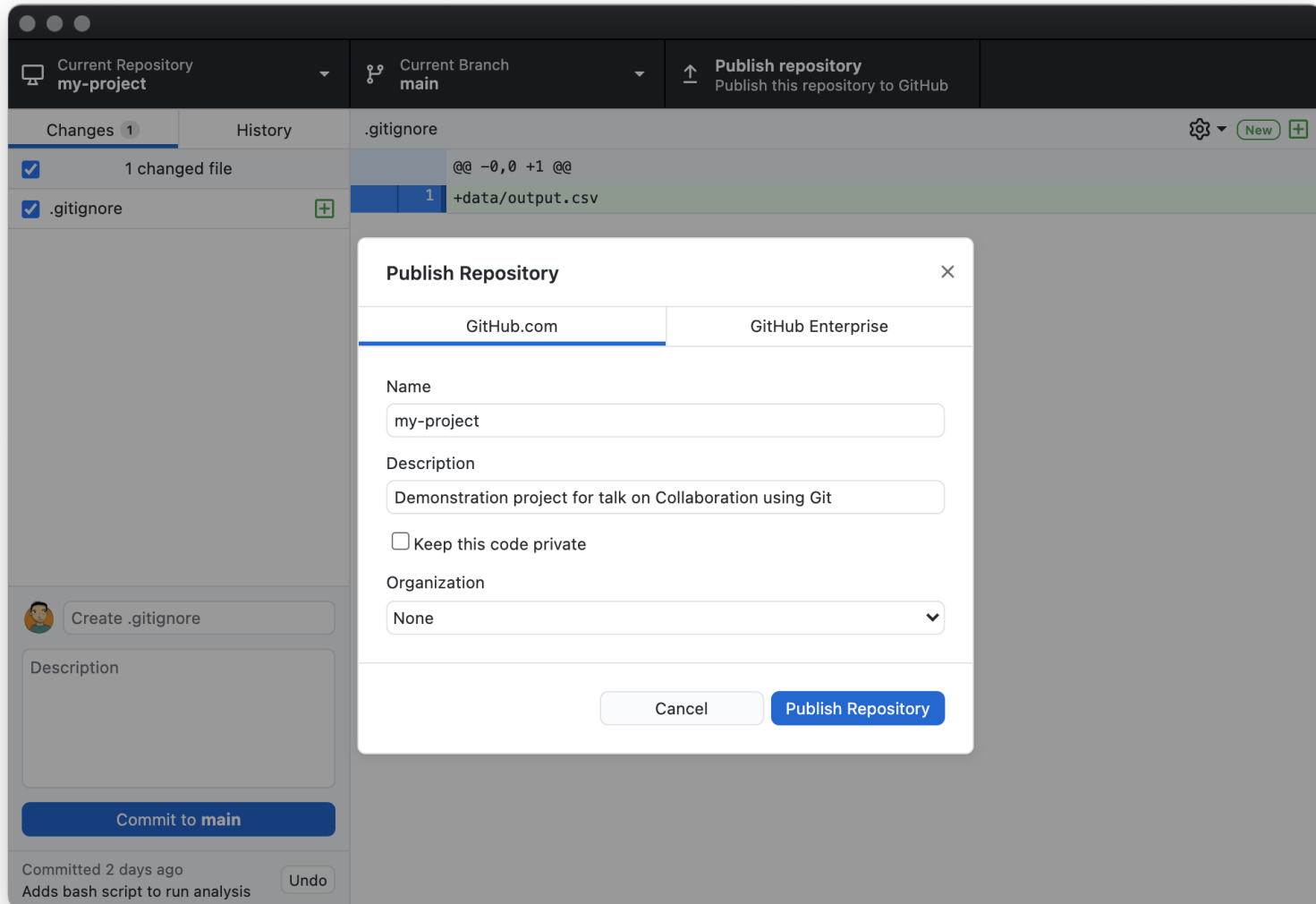


### ...or import code from another repository

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

Import code

# Publishing a repo from GitHub Desktop



# The repo on GitHub

The screenshot shows the GitHub interface for a repository named 'my-project' by user 'pgmccann'. The repository is public and has 1 pull request, 0 stars, and 0 forks. The 'Code' tab is selected, showing a list of files and folders. The 'About' section on the right provides a description of the project as a demonstration for a talk on Git collaboration. The 'Releases' and 'Packages' sections indicate that no releases or packages have been published yet.

Search or jump to... Pulls Issues Marketplace Explore

pgmccann / my-project Public Unwatch 1 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights

main Go to file Add file Code About

pgmccann Adds bash script to run analysis 2 days ago 7

data	Adds input data	2 days ago
src	Adds analysis script	2 days ago
test	Adds test script	2 days ago
.gitattributes	Initial commit	8 days ago
LICENSE.txt	Adds MIT license	2 days ago
README.md	Adds basic README describing project	5 days ago
run.sh	Adds bash script to run analysis	2 days ago

README.md

## My Project

This is an example project to illustrate the use of Git for collaboration in a research context.

Demonstration project for talk on Collaboration using Git

Readme MIT License

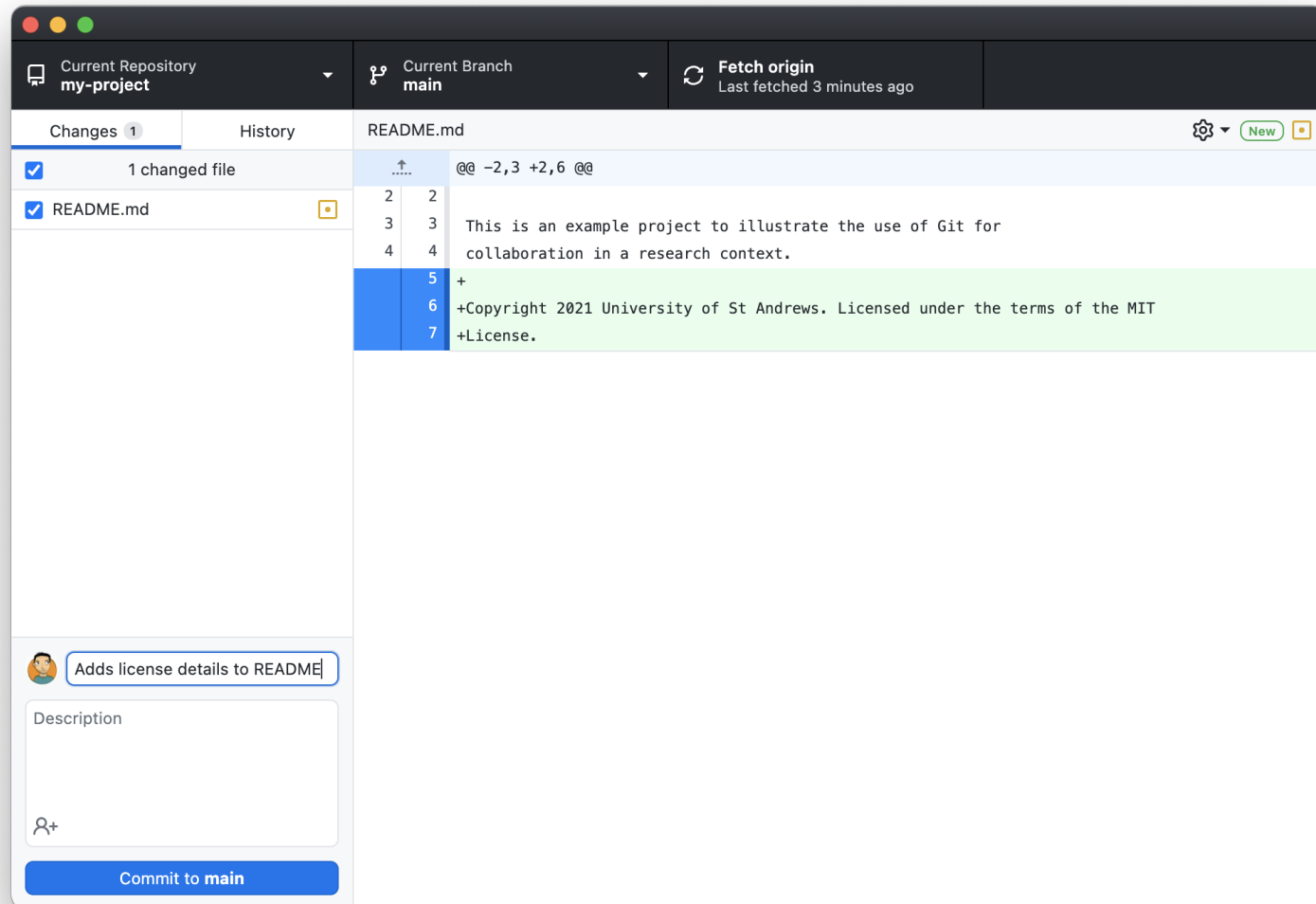
Releases

No releases published  
[Create a new release](#)

Packages

No packages published  
[Publish your first package](#)

# Pushing



```
$ 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")
$ git diff
diff --git a/README.md b/README.md
index e0a3173..b9fd878 100644
--- a/README.md
+++ b/README.md
@@ -2,3 +2,6 @@
```



# After committing...

```
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 429 bytes | 429.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object
To https://github.com/pgmccann/my-project.git
   2bf80ab..3ec775a  main -> main
```



README.md

Adds license details to README

34 minutes ago



run.sh

Adds bash script to run analysis

2 days ago

README.md



# My Project

---

This is an example project to illustrate the use of Git for collaboration in a research context.

Copyright 2021 University of St Andrews. Licensed under the terms of the MIT License.

# Pulling

Let's add a License **badge** using GitHub's editing interface.



my-project /

README.md

in main

Cancel changes

<> Edit file

Preview

Spaces

2

Soft wrap

1 # My Project

2

3 This is an example project to illustrate the use of Git for  
4 collaboration in a research context.

5

6 Copyright 2021 University of St Andrews. Licensed under the terms of the MIT  
7 License.

8

my-project /

README.md

in main

Cancel changes

<> Edit file

Preview

Spaces

2

Soft wrap

```
1  \[!\[License: MIT\]\(https://img.shields.io/badge/License-MIT-yellow.svg\)\]\(https://opensource.org/licenses/MIT\)
2
3  # My Project
4
5  This is an example project to illustrate the use of Git for
6  collaboration in a research context.
7
8  Copyright 2021 University of St Andrews. Licensed under the terms of the MIT
9  License.
10
```

## Commit changes

Adds License badge to README

Add an optional extended description...



pgm5@st-andrews.ac.uk



Choose which email address to associate with this commit

- ☒ Commit directly to the `main` branch.
- ☐ Create a **new branch** for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes

Cancel



README.md

Adds License badge to README

now



run.sh

Adds bash script to run analysis

2 days ago

README.md



License MIT

# My Project

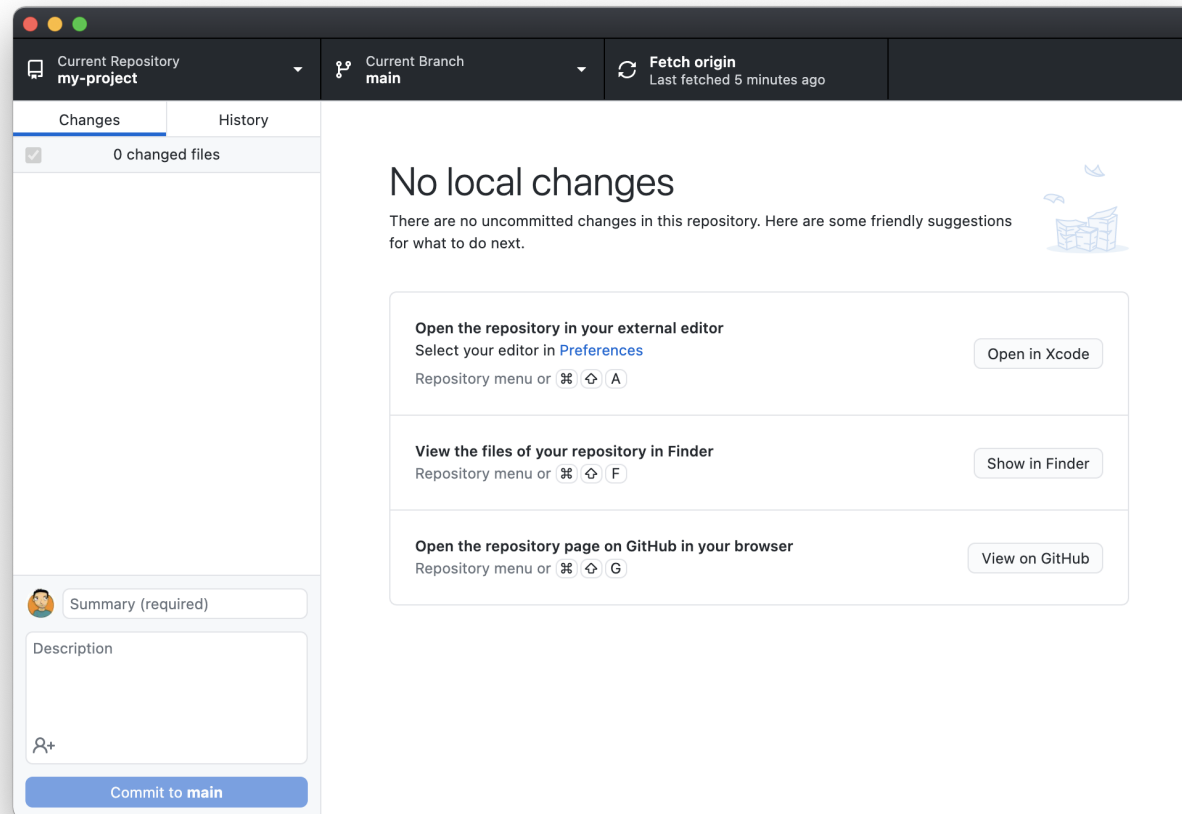
---

This is an example project to illustrate the use of Git for collaboration in a research context.

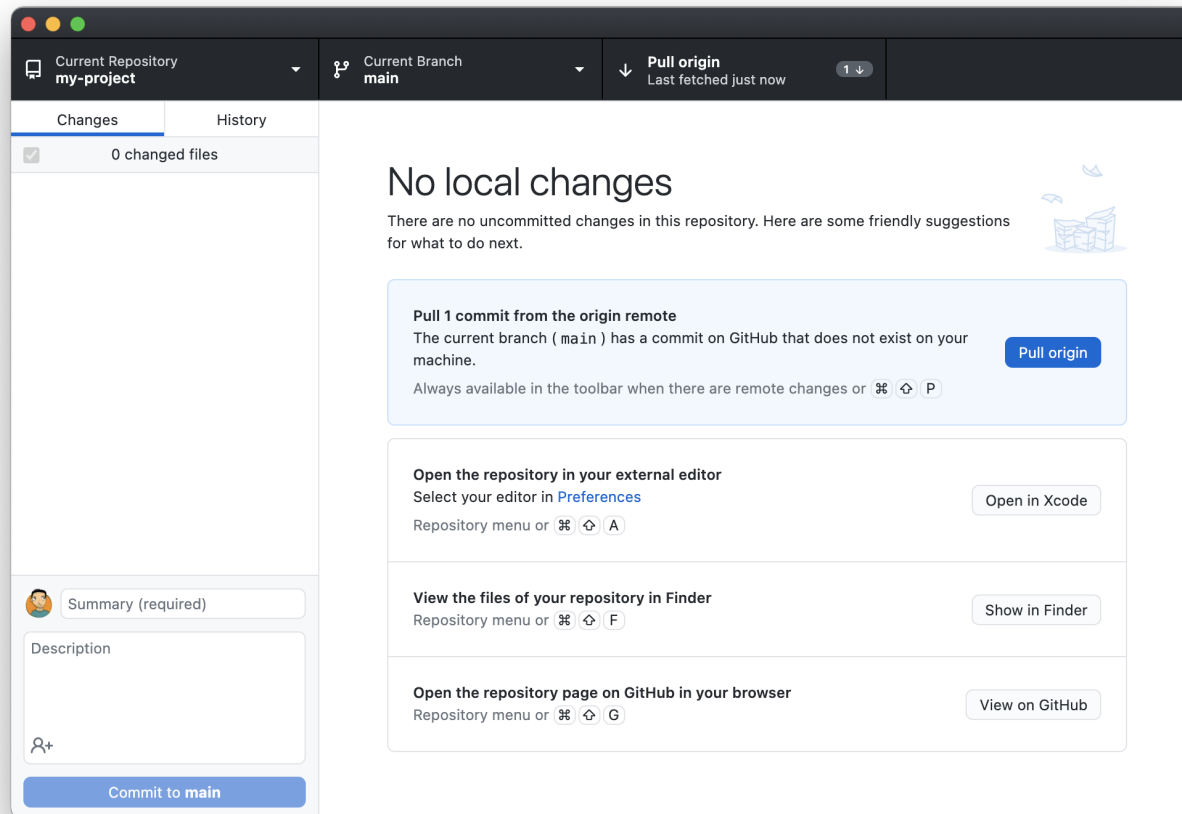
Copyright 2021 University of St Andrews. Licensed under the terms of the MIT License.

```
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```







```
$ git pull
Updating 3ec775a..658258a
Fast-forward
 README.md | 2 ++
 1 file changed, 2 insertions(+)
```

Current Repository  
my-project

Current Branch  
main

Fetch origin  
Last fetched 8 minutes ago

Changes

History

No Branches to Compare

Adds License badge to README  
Paddy McCann • 17m

Adds license details to README  
Patrick McCann • 1h

Ignores the output data  
Patrick McCann • 1h

Adds bash script to run analysis  
Patrick McCann • 1d

Adds test script  
Patrick McCann • 1d

Adds analysis script  
Patrick McCann • 1d

Adds input data  
Patrick McCann • 1d

Adds MIT license  
Patrick McCann • 1d

Adds basic README describing proj...  
Patrick McCann • 4d

Initial commit  
Patrick McCann • Oct 26, 2021

Adds License badge to README

Paddy McCann 658258a 1 changed file +2 -0 New

README.md

@@ -1,3 +1,5 @@

1 +[![License: MIT](https://img.shields.io/badge/License-MIT

2 -yellow.svg)](https://opensource.org/licenses/MIT)

2 +

1 3 # My Project

2 4

3 5 This is an example project to illustrate the use of Git f

or

# Git in RStudio

Collaboration

# Branching

# Forking

# Automation

Open Science



# Licensing

Citation

Zenodo and Pure