Science & Technology Facilities Council | UK Research and Innovation | NERC SCIENCE OF THE ENVIRONMENT

Introduction to Git and GitHub

Managing your code: quietly introducing *Git* - a friend for life

Thanks to all contributors:

Alison Pamment, Sam Pepler, Ag Stephens, Stephen Pascoe, Kevin Marsh, Anabelle Guillory, Graham Parton, Esther Conway, Eduardo Damasio Da Costa, Wendy Garland, Alan Iwi, Matt Pritchard and Tommy Godfrey.

Contents

- What is version control?
- What are Git & GitHub?
- Nice features of GitHub
- Basic

Foreword

Git is easy to use but will take a bit of practice to get comfortable. With that in mind:

- We will encourage you to commit your work at then end of every exercise.
- The basics will be enough for most use-cases
- We will provide you with a cheat sheet
- The internet is full of answers
- Give it a go. **You don't need to "get it" to "git it".**
- **The exercise following this presentation will make sure you are setup with git/github**

What is a version control system (VCS)?

- Version control software keeps track of your changes
- Allows you to revert back to a previous point
- Manages contributions from multiple people
- Creates freeze points which won't change
- Stores the full history of the things under version control including who did what, when?

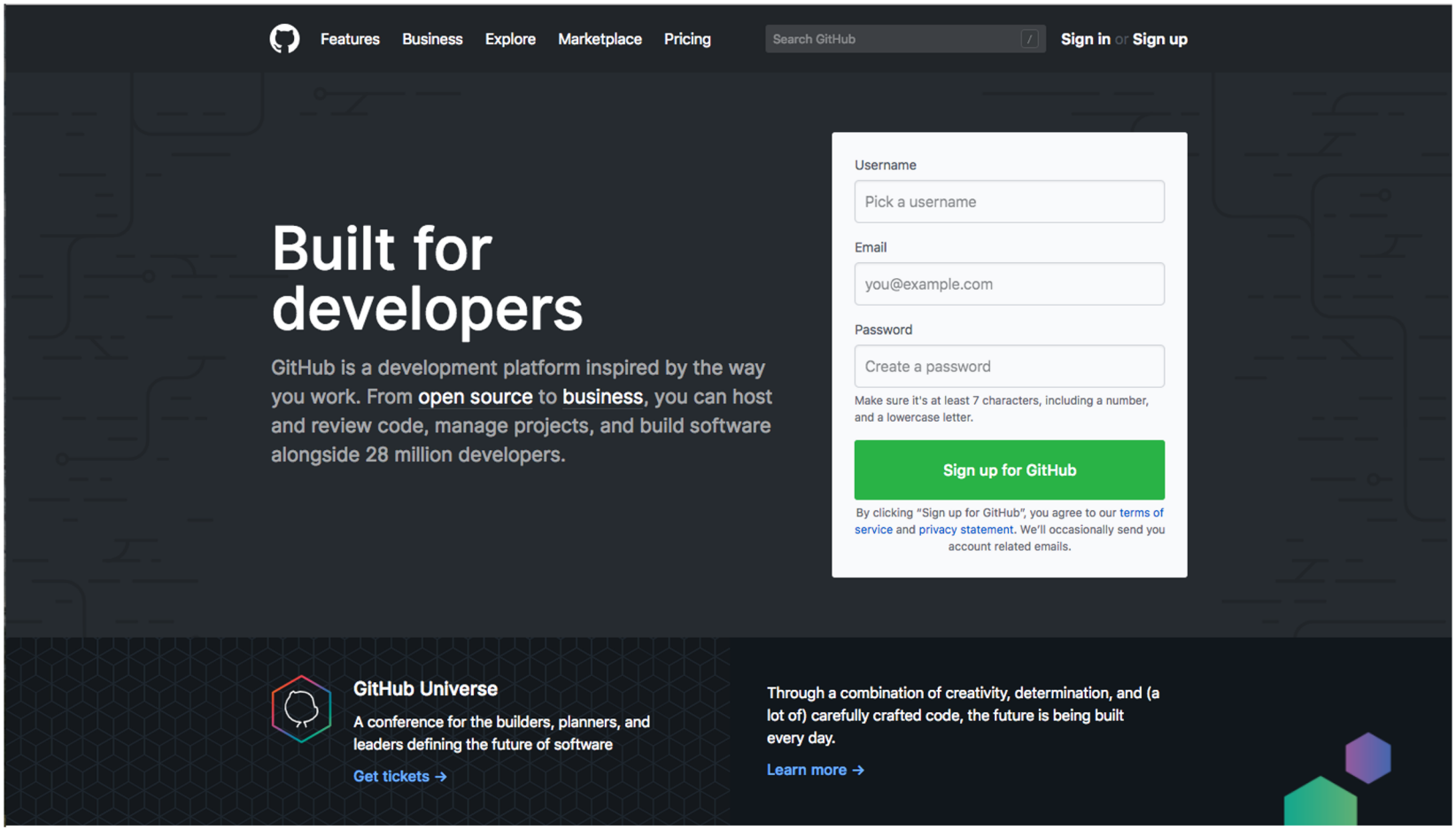
Why might you need VCS?

- Scientists are typically **required to publish data and code** (by their funders/institutions).
- Collaboration between scientists requires data-sharing; this implicitly relies on **code-sharing**.
- There are **tools that make it easy** to record our changes, document our workflow and "fix" releases of our code at important steps along the way.
- Reduce errors and admin burden (latest, new1, new2 etc…)
- Allows you test ideas with confidence, you can always go back.

So, working on the premise that we accept that we need to know about, and use, version control…

We will use Git and GitHub

Introducing GitHub



https://github.com/

What is GitHub?

"A **web-based** Git repository **hosting service**"

GitHub allows you to:

- Share your repositories with others
- Access other user's repositories
- Store remote copies as a backup of your local repositories
- Add bug tracking, feature requests, wikis, ...

GitHub is **free** for most use cases

Git vs GitHub

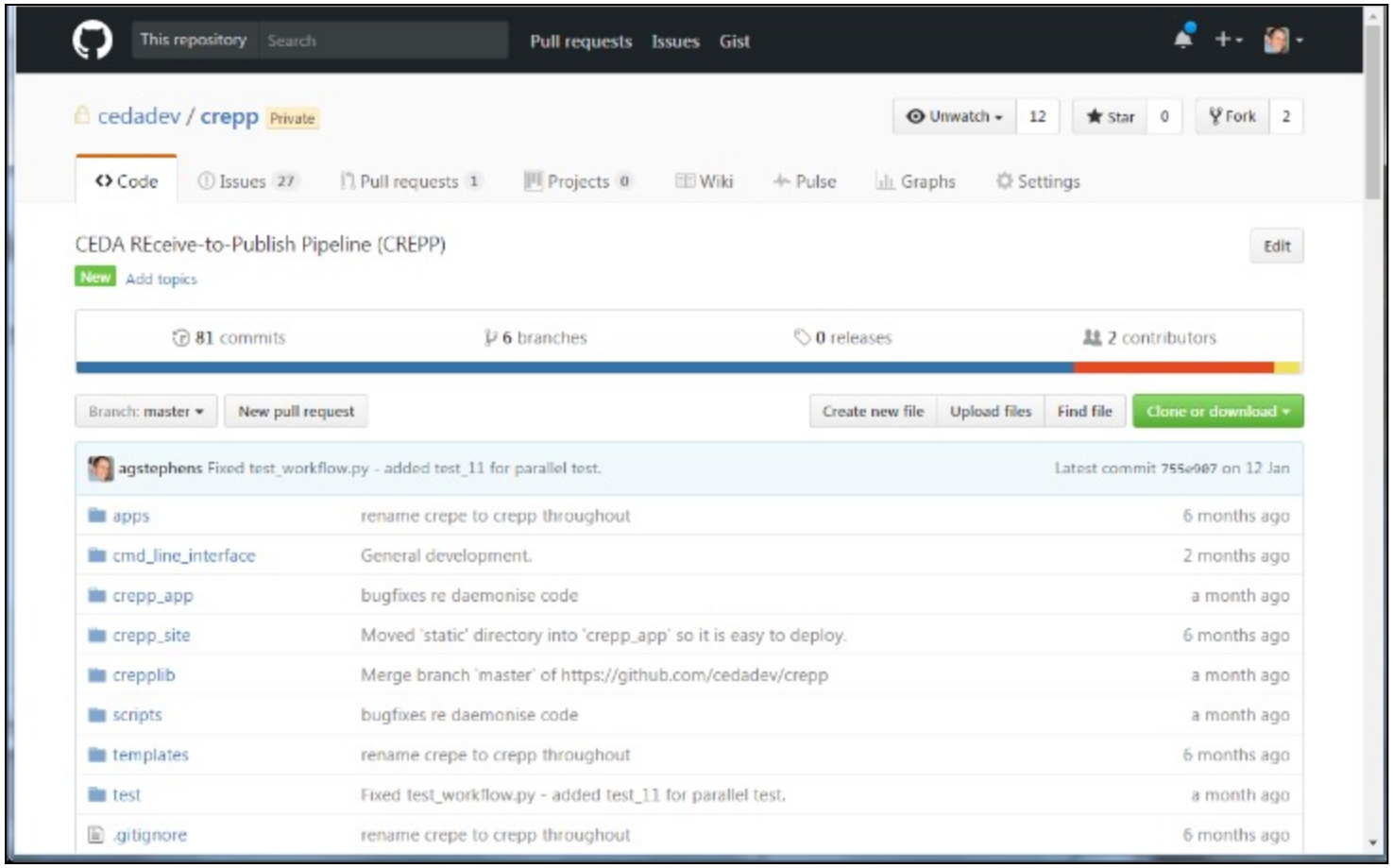
Git is a revision control system, a tool to manage your source code history.

GitHub is a hosting service for Git repositories.

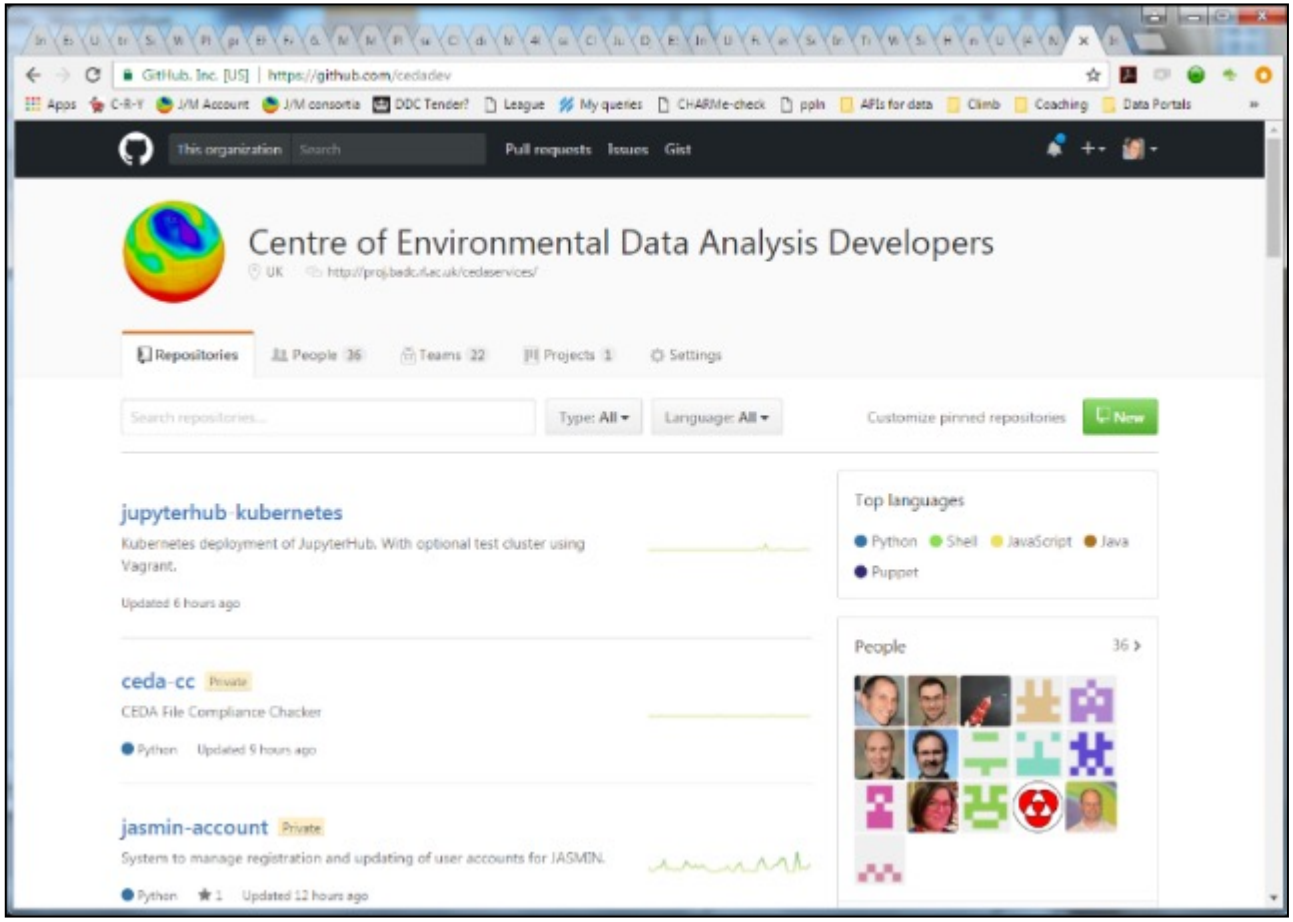
They are not the same thing. **Git** is the **tool**, **GitHub** is a **web service**.

You **do not** need GitHub to use Git but GitHub adds useful functionality.

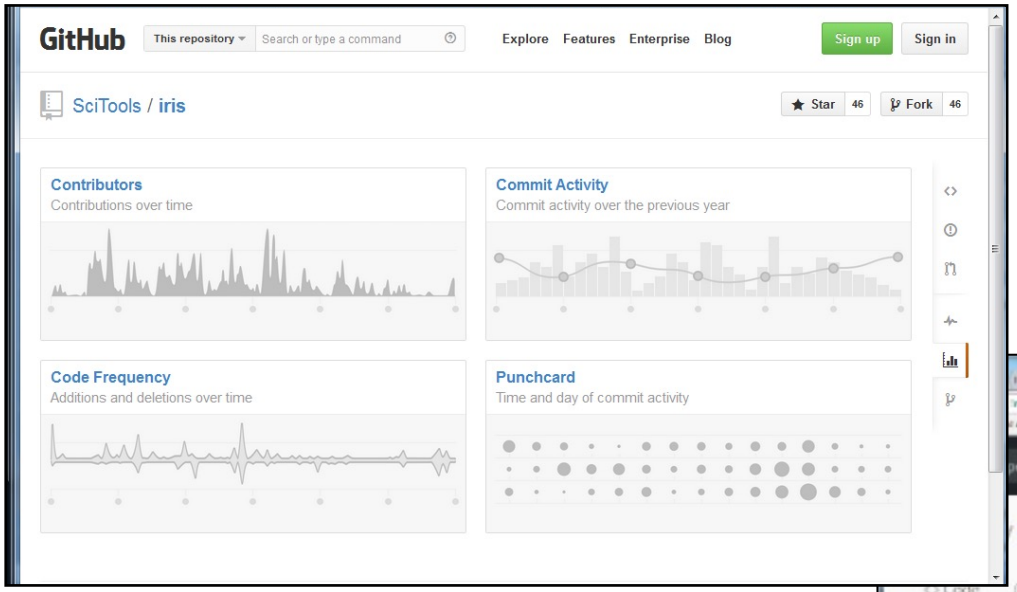
GitHub: repositories (public *or* **private**)

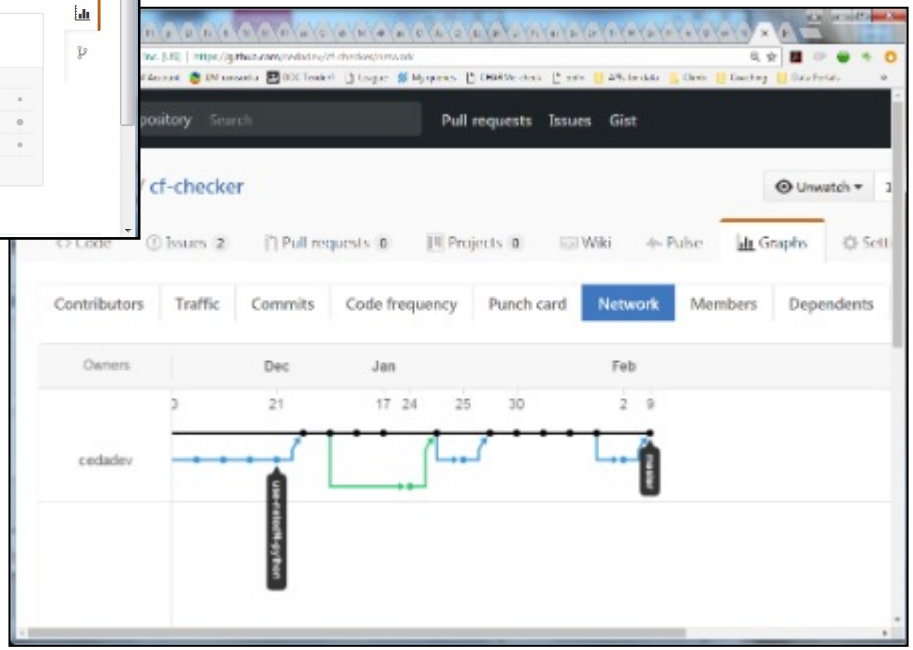


GitHub: organisations

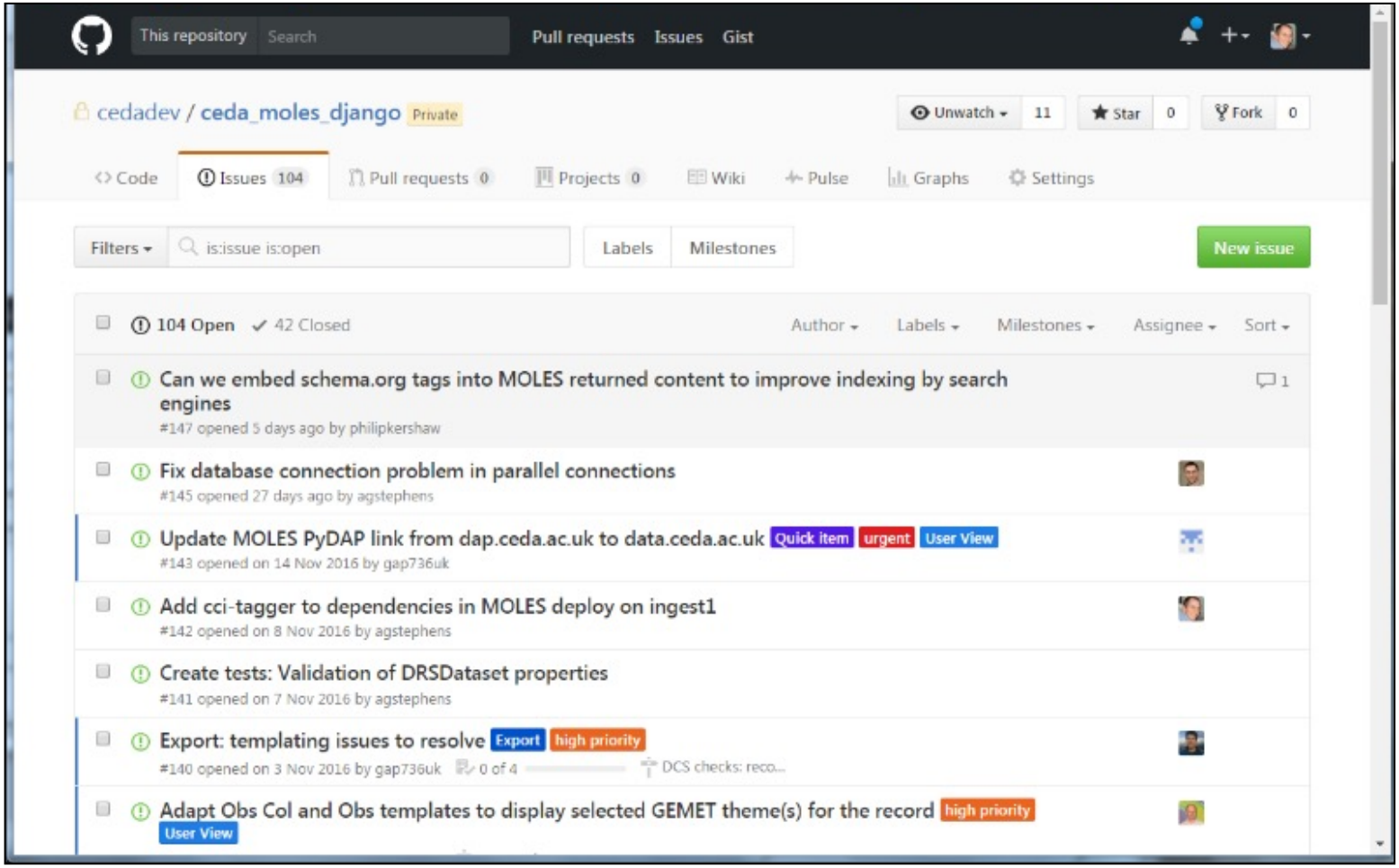


GitHub: collaboration (branch/fork)





GitHub: Issue tracking



GitHub: history and change

6 cedamoles_app/admin_tools/integrity/routine_checks.py

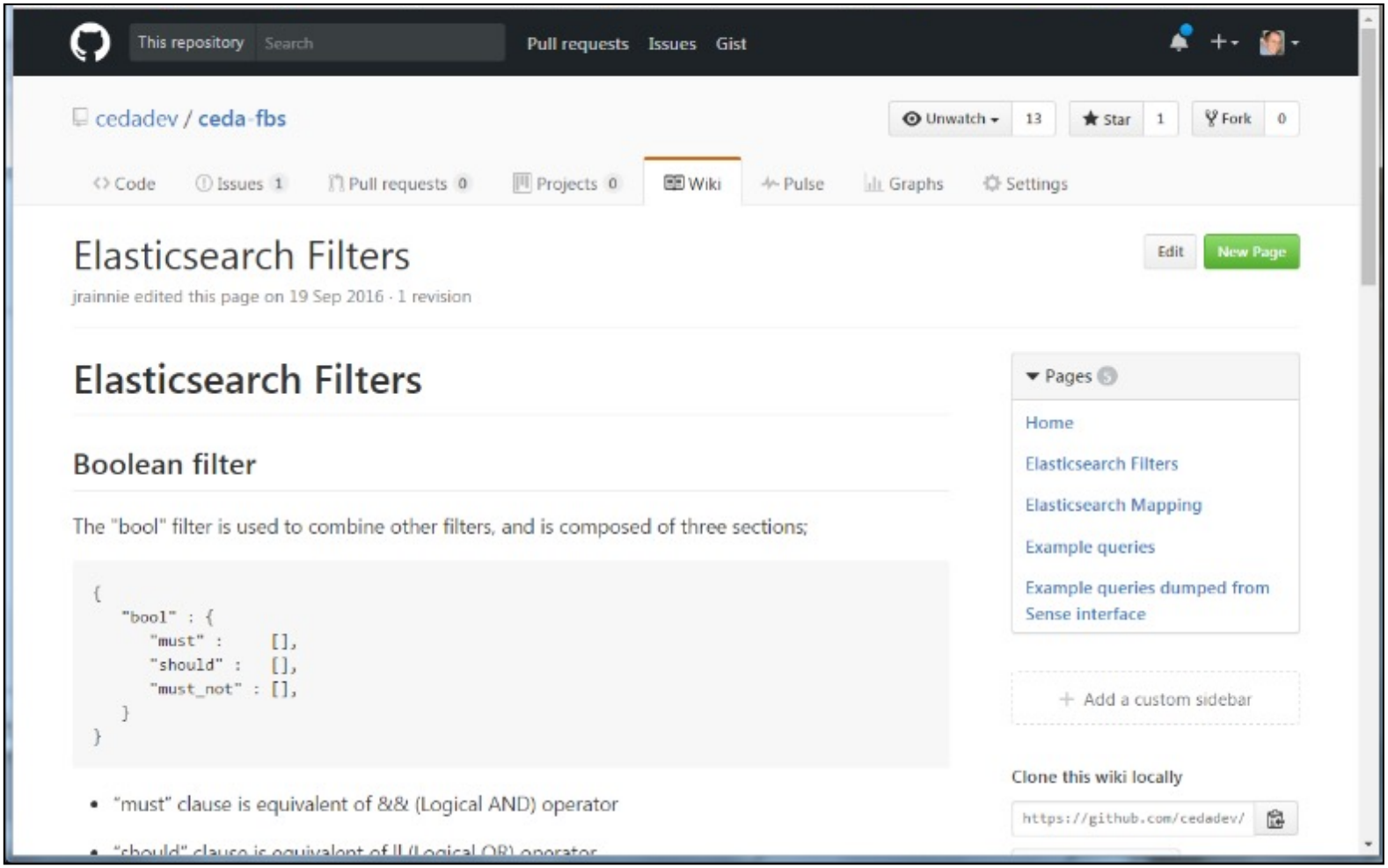
```
@@ -41,10 +41,10 @@ def run_checks(self):

 class ResultChecks(ChecksBase):

-    def check_internalPath(self):
+    def check_dataPath(self):
         found = Counter()
         for result in Result.objects.all():
-            path = result.internalPath
+            path = result.dataPath
             found.update([path])

         dupes = [(path, count) for (path, count) in found.items() if count > 1]
@@ -53,7 +53,7 @@ def check_internalPath(self):
         for path, count in dupes:
             print path, count
```

GitHub: wikis



GitHub does lots of funky things, but…

- On this course we are going only using it as a remote repository.
- We are going to concentrate on simply using git.

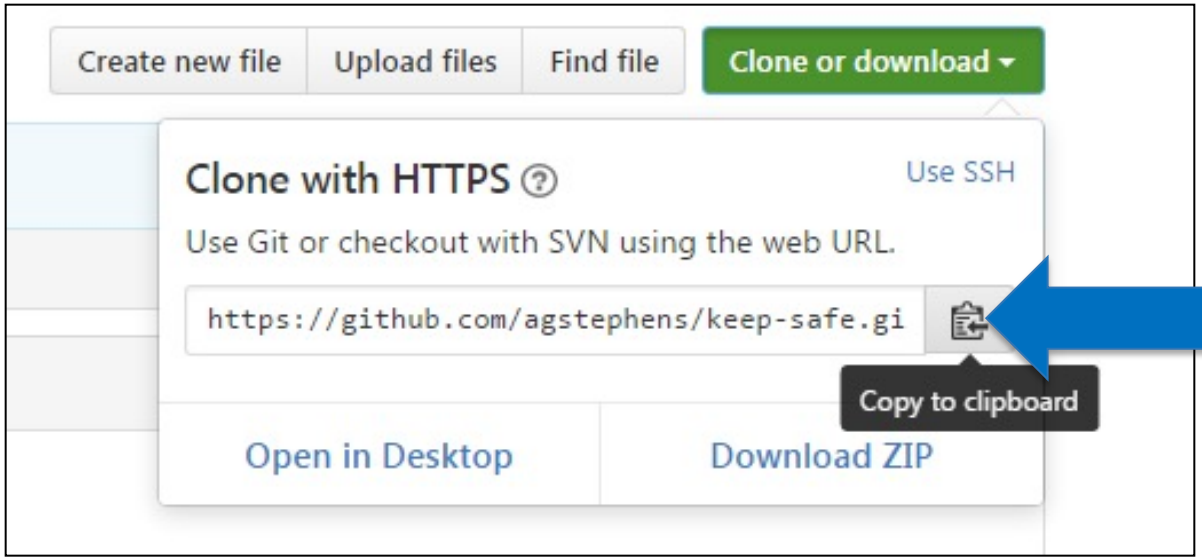
Where to start?

There are three start points when using git:

1. Clone an existing repository **from GitHub**
2. Create a new, empty repo and clone it **from GitHub**
3. Turn existing **local directory** into a git repo, can either contain files or be empty

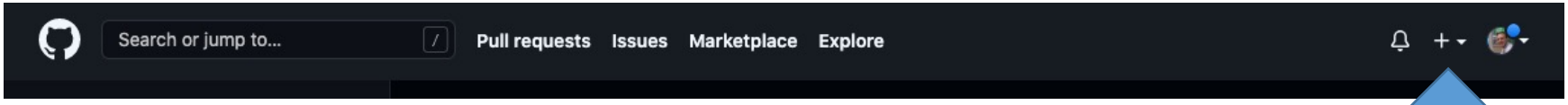
Where to start 1: Clone Existing Repo

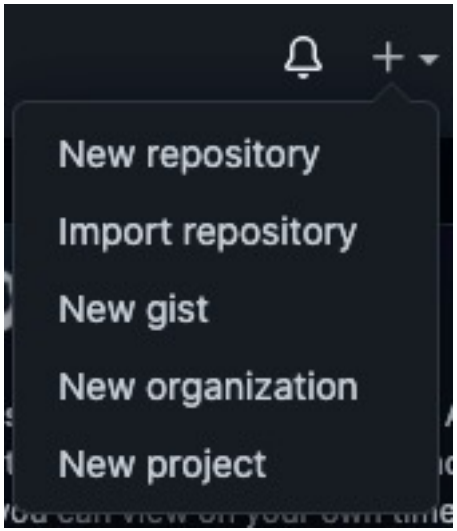
This makes a copy of a repository locally.




```
$ git clone
  https://github.com/agstephens/keep-safe
```

Where to start 2: Create a repository on GitHub





Click New repository

Click + in top right

You can then clone in the same way as Where to start 1

Where to start 3: start a new repository from existing files

```
$ ls
x       y       z
$ git init
Initialized empty Git repository in
/Users/sjp23/play/york_workshop_shell/test-pakage/.git/
$ git add .
$ git commit -m'Initial commit from existing files'
[master (root-commit) 71ecfcf] Initial commit from
existing files
 3 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 x
 create mode 100644 y
 create mode 100644 z
```

The basic workflow: Adding a file

1. Enter the repository directory:

```
$ cd ncas-isc
```

2. Create a new file:

```
$ echo "hello world" > hello.txt
```

3. Tell Git about the file:

```
$ git add hello.txt
```

4. Commit the file to the **local** Git repository:

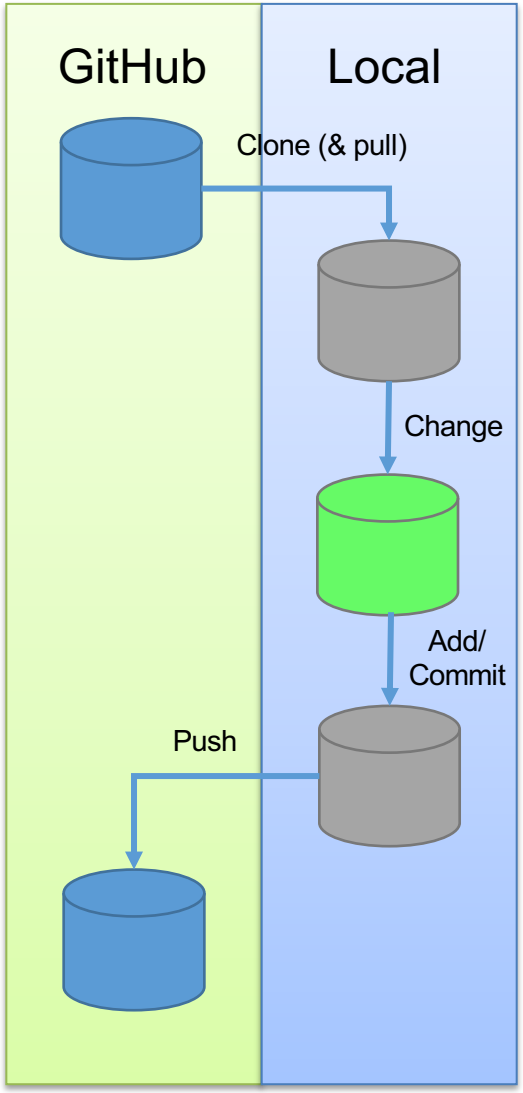
```
$ git commit -m "added hello"
```

5. Push any updates to the **remote** GitHub repo:

```
$ git push
```

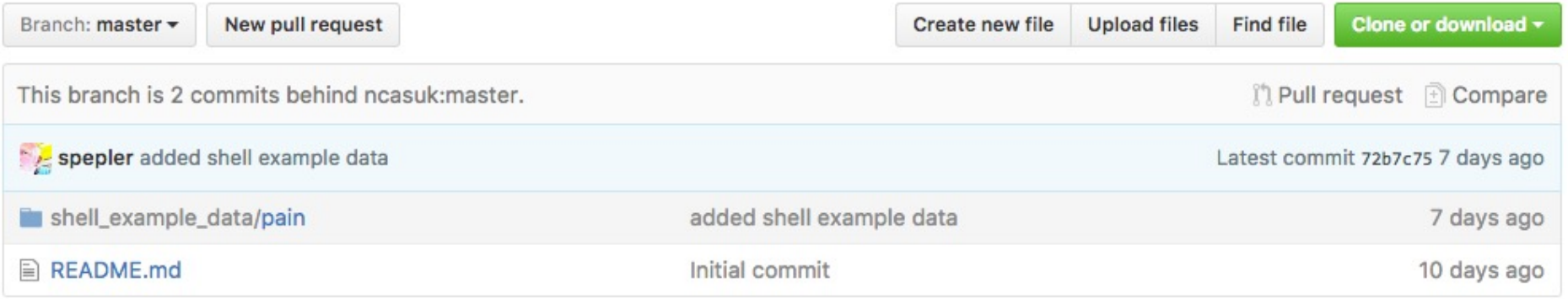
So, what just happened?

- We *cloned* the remote repository to our file system.
  - Now there are two identical copies of one repo.
- We *created* a new text file.
- We *added* and *committed* that new file to the local version of the repo.
- We used *push* to update the remote repo.

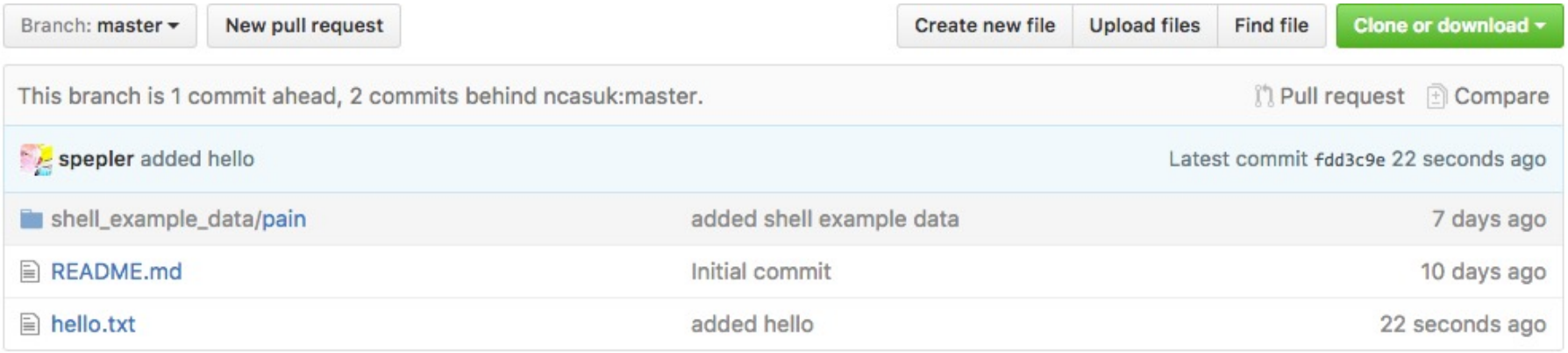


Let's look on GitHub

Before…



After…



The Plan: Use git / GitHub all week

- This stuff is hard to learn - we know that from experience.
- A presentation is quickly forgotten.
- So, we propose that you use Git/GitHub for every exercise.
- You are going to create and update your own Github repository with files from exercises throughout the course.