

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 workflow

Foreword

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

- We will encourage you to commit your work at the end of every exercise.
- The basics will be enough for most use-cases
- We will provide you with a cheatsheet
- 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?

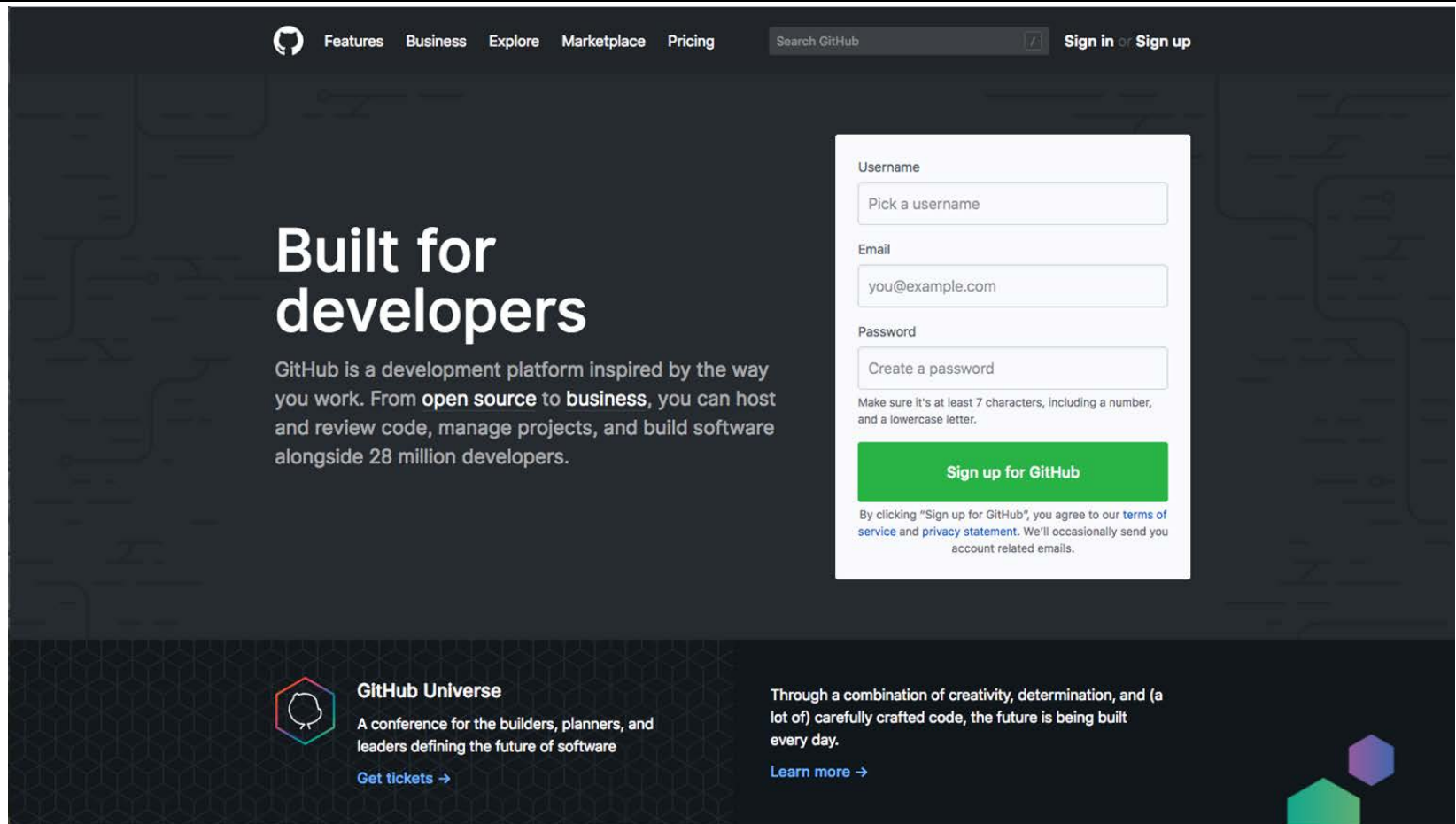
- Scientists are typically **required to publish data and code** (by their funders/institutions).
- Collaboration between scientists requires data-sharing; this implicitly relies upon **code-sharing**.
- There are **tools that make it easy** to record our changes, document our workflow and "fix" releases of our code at important steps along the way.
- Reduce errors and admin burden ("latest", "new2"...)
- 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



The screenshot shows the GitHub homepage with a dark background and a light-colored sign-up form. The navigation bar at the top includes links for Features, Business, Explore, Marketplace, and Pricing, along with a search bar and sign-in/sign-up options. The main heading is "Built for developers", followed by a description of GitHub as a development platform. The sign-up form includes fields for Username, Email, and Password, with a green "Sign up for GitHub" button. Below the button is a disclaimer about terms of service and privacy. At the bottom, there is a section for "GitHub Universe" conference tickets and a "Learn more" link.

Features Business Explore Marketplace Pricing Search GitHub Sign in or Sign up

Built for developers

GitHub is a development platform inspired by the way you work. From **open source** to **business**, you can host and review code, manage projects, and build software alongside 28 million developers.

Username
Pick a username


Email
you@example.com

Password
Create a password

Make sure it's at least 7 characters, including a number, and a lowercase letter.

[Sign up for GitHub](#)

By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy statement](#). We'll occasionally send you account related emails.

 **GitHub Universe**
A conference for the builders, planners, and leaders defining the future of software
[Get tickets →](#)

Through a combination of creativity, determination, and (a lot of) carefully crafted code, the future is being built every day.
[Learn more →](#)

<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)

The screenshot displays the GitHub interface for the repository **cedadev / crepp**, which is marked as **Private**. The repository has 12 watchers, 0 stars, and 2 forks. The main navigation bar includes links for **Code**, **Issues** (27), **Pull requests** (1), **Projects** (0), **Wiki**, **Pulse**, **Graphs**, and **Settings**.

The repository description is **CEDA REceive-to-Publish Pipeline (CREPP)**, with an **Edit** button. Below this, a progress bar shows the repository's activity: 81 commits, 6 branches, 0 releases, and 2 contributors.

Navigation options include **Branch: master**, **New pull request**, **Create new file**, **Upload files**, **Find file**, and **Clone or download**.

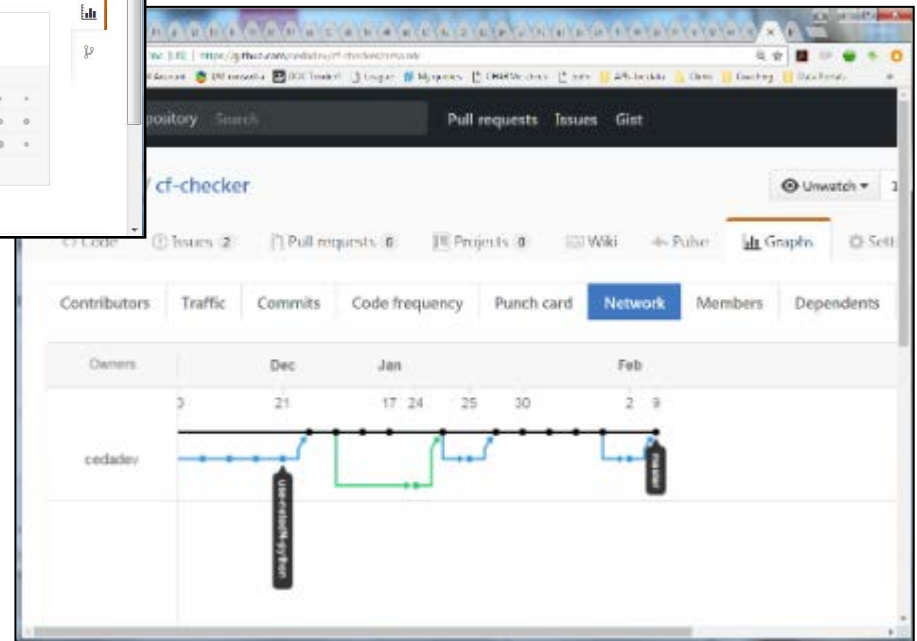
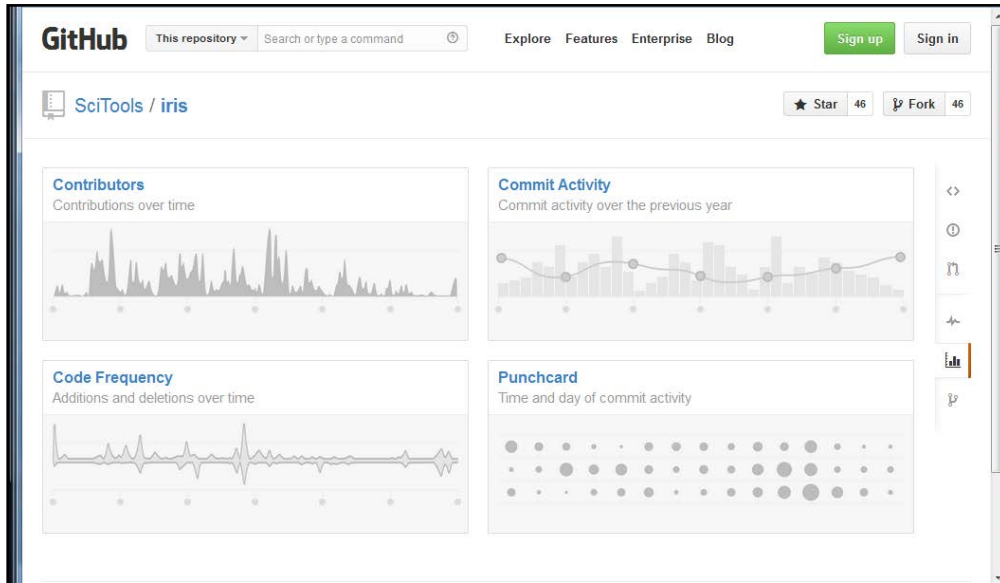
The file list shows the following items and their commit history:

File/Folder	Commit Message	Time Ago
apps	rename crepe to crepp throughout	6 months ago
cmd_line_interface	General development.	2 months ago
crepp_app	bugfixes re daemonise code	a month ago
crepp_site	Moved 'static' directory into 'crepp_app' so it is easy to deploy.	6 months ago
creplib	Merge branch 'master' of https://github.com/cedadev/crepp	a month ago
scripts	bugfixes re daemonise code	a month ago
templates	rename crepe to crepp throughout	6 months ago
test	Fixed test_workflow.py - added test_11 for parallel test.	a month ago
.gitignore	rename crepe to crepp throughout	6 months ago

GitHub: organisations

The screenshot shows the GitHub organization page for the 'Centre of Environmental Data Analysis Developers'. The page header includes the GitHub logo, the organization name, location (UK), and website URL. Below the header, there are tabs for 'Repositories', 'People' (36), 'Teams' (22), 'Projects' (1), and 'Settings'. A search bar and filters for 'Type' and 'Language' are present. The main content area displays a list of repositories, including 'jupyterhub-kubernetes', 'ceda-cc', and 'jasmin-account'. Each repository entry shows its name, description, and update status. On the right side, there are sections for 'Top languages' (Python, Shell, JavaScript, Java, Puppet) and 'People' (36), which includes a grid of member avatars.

GitHub: collaboration (branch/fork)




GitHub: Issue tracking

The screenshot displays the GitHub interface for the repository `cedadev / ceda_moles_django`. The repository is private and has 11 watchers, 0 stars, and 0 forks. The 'Issues' tab is selected, showing 104 open issues. A search filter 'is:issue is:open' is applied. The issues list includes:

- Can we embed schema.org tags into MOLES returned content to improve indexing by search engines** (#147, opened 5 days ago by philipkershaw)
- Fix database connection problem in parallel connections** (#145, opened 27 days ago by agstephens)
- Update MOLES PyDAP link from `dap.ceda.ac.uk` to `data.ceda.ac.uk`** (#143, opened on 14 Nov 2016 by gap736uk). Labels: Quick Item, urgent, User View.
- Add cci-tagger to dependencies in MOLES deploy on ingest1** (#142, opened on 8 Nov 2016 by agstephens)
- Create tests: Validation of DRSDataset properties** (#141, opened on 7 Nov 2016 by agstephens)
- Export: templating issues to resolve** (#140, opened on 3 Nov 2016 by gap736uk). Labels: Export, high priority. Progress: 0 of 4. DCS checks: reco...
- Adapt Obs Col and Obs templates to display selected GEMET theme(s) for the record** (high priority, User View).

GitHub: history and change

```
6  cedamoles_app/admin_tools/integrity/routine_checks.py

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

41 41
42 42     class ResultChecks(ChecksBase):
43 43
44 -     def check_internalPath(self):
44 +     def check_dataPath(self):
45 45         found = Counter()
46 46         for result in Result.objects.all():
47 -         path = result.internalPath
47 +         path = result.dataPath
48 48         found.update([path])
49 49
50 50         dupes = [(path, count) for (path, count) in found.items() if count > 1]
@@ -53,7 +53,7 @@ def check_internalPath(self):

53 53         for path, count in dupes:
54 54             print path, count
```



GitHub: wikis

The screenshot shows a GitHub repository page for 'cedadev / ceda-fbs'. The repository has 13 pulls, 1 star, and 0 forks. The 'Wiki' tab is selected, showing a page titled 'Elasticsearch Filters' edited by 'jrainnie' on 19 Sep 2016. The page content includes a section for 'Boolean filter' and a JSON snippet for a 'bool' filter. A sidebar on the right lists other wiki pages: Home, Elasticsearch Filters, Elasticsearch Mapping, Example queries, and Example queries dumped from Sense interface. At the bottom, there is a 'Clone this wiki locally' button with the URL 'https://github.com/cedadev/'.

cedadev / ceda-fbs

Unwatch 13 Star 1 Fork 0

Code Issues 1 Pull requests 0 Projects 0 Wiki Pulse Graphs Settings

Elasticsearch Filters

jrainnie edited this page on 19 Sep 2016 · 1 revision

Elasticsearch Filters

Boolean filter

The "bool" filter is used to combine other filters, and is composed of three sections;

```
{
  "bool" : {
    "must" : [],
    "should" : [],
    "must_not" : [],
  }
}
```

- "must" clause is equivalent of && (Logical AND) operator
- "should" clause is equivalent of || (Logical OR) operator

Pages 5

- Home
- Elasticsearch Filters
- Elasticsearch Mapping
- Example queries
- Example queries dumped from Sense interface

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/cedadev/>

GitHub does lots of funky things, but...

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

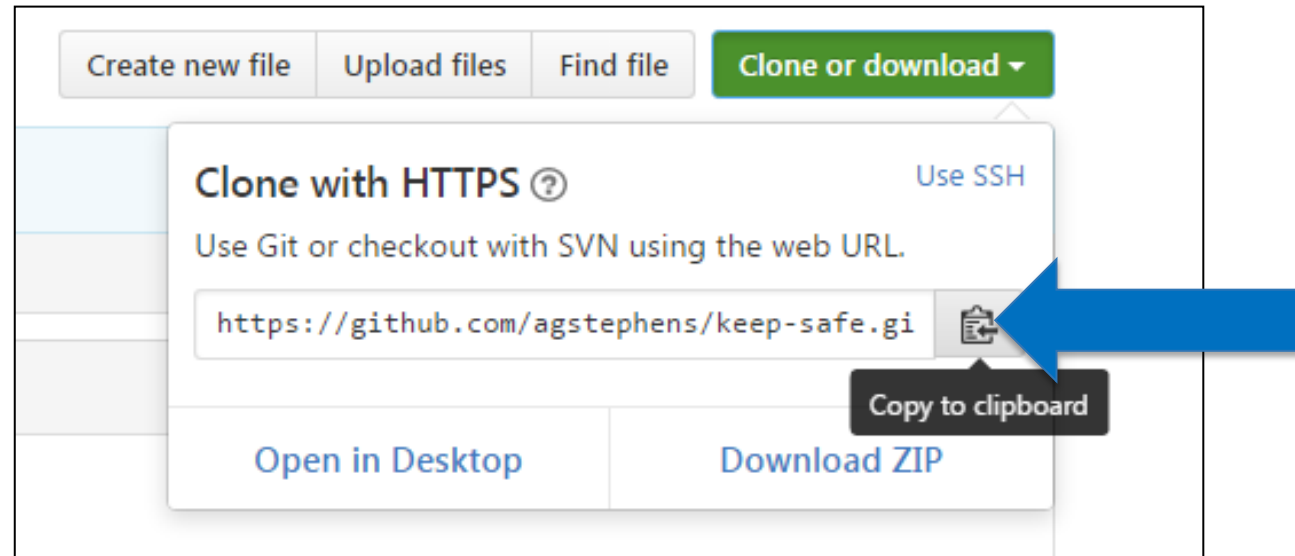
Where to start?

There are three different start points when using Git:

- Clone an existing repository **from GitHub**
- Create a new, empty repo and clone it **from GitHub**
- Turn an existing **local directory** into a Git repo; it can contain files or be empty

Where to start 1: Clone Existing Repo

This makes a copy of a repository locally.

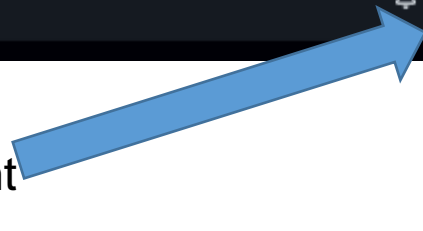


```
$ git clone  
https://github.com/agstephens/keep-safe
```

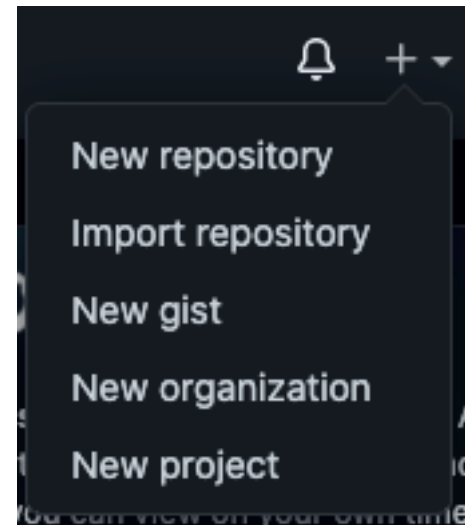
Where to start 2: Create a repository on GitHub



Click "+" in top right



Click "New repository"



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

Where to start 3: start a new repository from an existing local directory

```
$ ls
x      y      z

$ git init
Initialized empty Git repository in
/Users/sjp23/play/york_workshop_shell/test-package/.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:

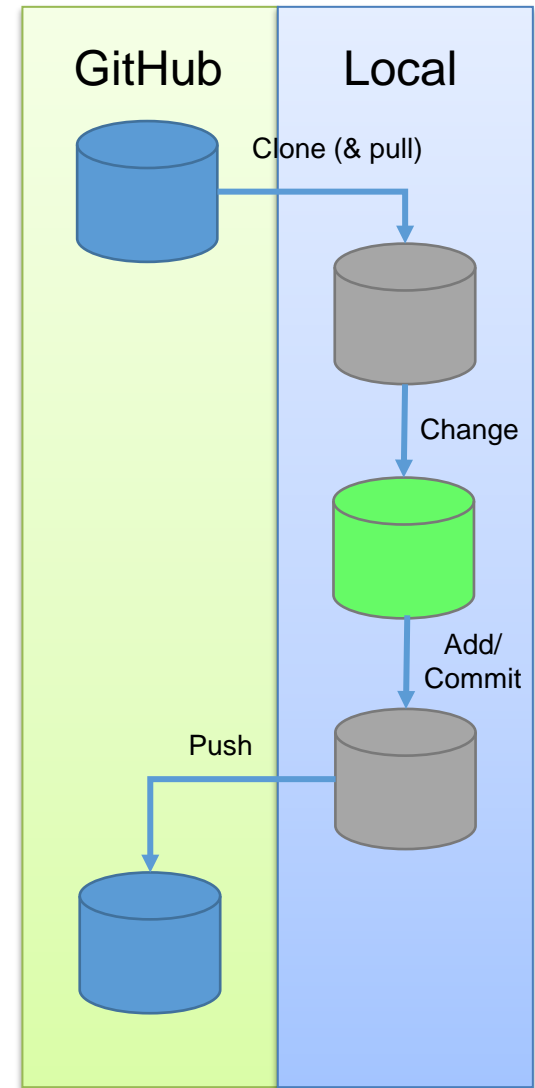
```
$ git commit -m "added hello"
```

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...


Branch: master ▾ New pull request

Create new file Upload files Find file Clone or download ▾

This branch is 2 commits behind ncasuk:master. [Pull request](#) [Compare](#)

 **spepler** added shell example data Latest commit 72b7c75 7 days ago

 [shell_example_data/pain](#) added shell example data 7 days ago


 [README.md](#) Initial commit 10 days ago


After...


Branch: master ▾ New pull request


Create new file Upload files Find file Clone or download ▾

This branch is 1 commit ahead, 2 commits behind ncasuk:master. [Pull request](#) [Compare](#)

 **spepler** added hello Latest commit fdd3c9e 22 seconds ago

 [shell_example_data/pain](#) added shell example data 7 days ago

 [README.md](#) Initial commit 10 days ago

 [hello.txt](#) added hello 22 seconds ago

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.
- We encourage you to create and update your own GitHub repository with files from exercises throughout the course.