Console Tools

Conda

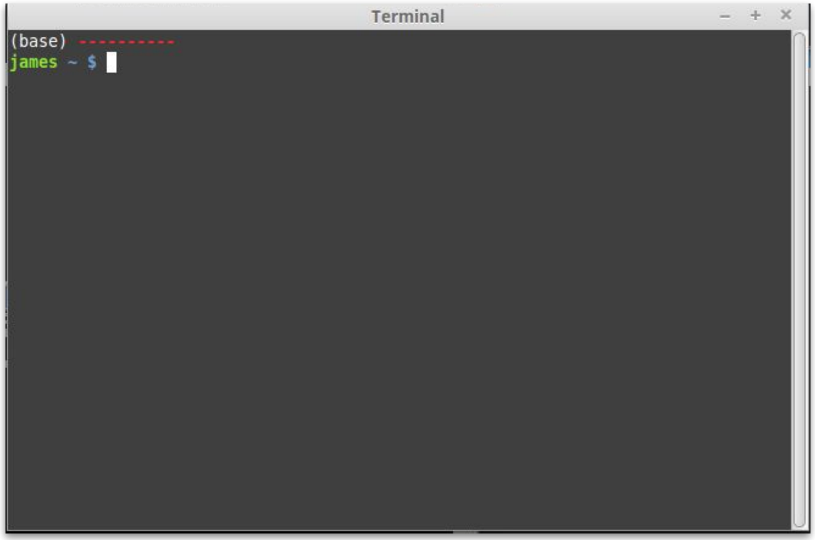
Cross-platform package management

https://conda.io/docs/user-guide

CAMBRIDGE SPARK

Let's jump straight in

- Open up a terminal!
- Windows users: open an **Anaconda Prompt**



```
(base) ----------
james ~ $ 
```

Practical conda

```
$  conda --version
$  conda update conda
$  conda create -n myenv3 python=3 jupyter
$  conda create -n myenv2 python=2 jupyter
$  conda info --envs
$  echo $PATH; cat ~/.bashrc
$  which -a python
$  conda activate myenv3
$  which -a python
$  conda install matplotlib numpy pandas
$  conda deactivate
```

- Don’t worry about `$PATH` or `.bashrc` stuff…
- …instead use anaconda prompt
- `which = where`
- `cat = type`

CAMBRIDGE SPARK

Why do we need this?

- Different applications require different configs:
  - One computer, many configs
  - Quickly switch between e.g. python2 and python3 projects!
- Cross-platform
- Consistent packages
- Future proof
- Release code - not a bundled binary
- Storage space for packages kept minimal

CAMBRIDGE SPARK

Jupyter

A multi-language Integrated Development Environment

https://jupyter-notebook.readthedocs.io/en/stable/

Quick start guide

CAMBRIDGE SPARK

Quickstart

```
$   cd /somewhere/you/wanna/make/notebooks
$   conda activate myenv3
$   jupyter notebook
$   ctrl + c
$   jupyter console  # exit()
$   jupyter qtconsole  # exit()
$   jupyter notebook
$   ipython
```

Again, do this in an Anaconda Console

Inside the jupyter notebook

```
In your browser navigate to the server.
Make a new notebook.
Hit esc then h to see shortcuts.
$  import matplotlib.pyplot as plt
$  import numpy as np
$  vec = np.arange(10)
$  vec  # IPython.display
$  plt.plot(vec)
$  %connect_info
$  %qtconsole
```

Review of jupyter notebook shortcuts

```
In edit mode:
1.  <shift> + <enter> - run cell and move to next
2.  <ctrl> + <enter> - run cell and stay here
3.  ?command - documentation on command
4.  np.ara<tab> - tab completion
In command mode:
1.  <h> - show help!
2.  <a> or <b> - create cell [a]bove or [b]elow
3.  <m> or <y> - make cell [m]arkdown or code[y]
4.  <d><d> - delete cell
```

CAMBRIDGE SPARK

Hands-on session

01-jupyter-skeleton.ipynb
5 mins

CAMBRIDGE SPARK

Why is this useful?!

- Server can be accessed remotely:
  - Different projects require different computational resources
  - Access requirements to data require controlled storage

- Plots *were* a pain when working with servers:
  - Make a file then scp that file to your local machine to view
  - X window forwarding...bleaugh
  - ...for quick analysis, plotting in a notebook is great!

- That quick analysis output is saved:
  - You can insert markdown comments throughout
  - Plot or print data
  - Even save sound samples

CAMBRIDGE SPARK

Git

Version control

https://git-scm.com/doc

- Getting-Started-Git-Basics
- https://git-scm.com/videos

CAMBRIDGE SPARK

Motivation for version control

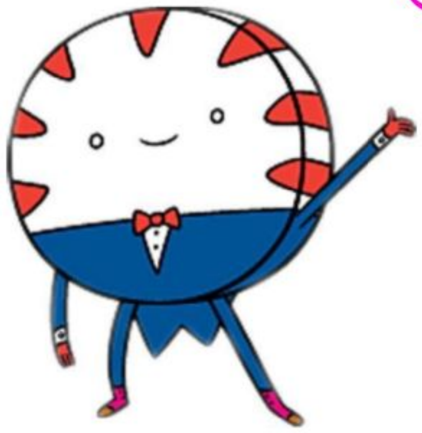
- You’re **always** working in a team:
  - Multiple people working
  - Future self will not remember why past self made that change!
- Save well and save often:
  - Reduce anxiety:
  - Keep a clean top level…
  - ...but retain entire history!
- Tell a story
- Work on one change at a time
- Side benefit: easy code sharing

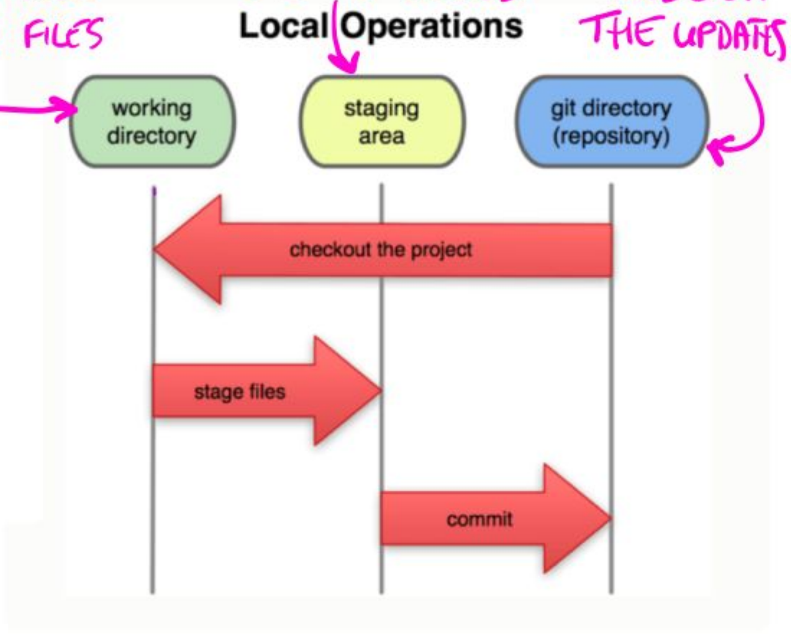
CAMBRIDGE SPARK

Quick configuration

```
$  git --version
$  git config --list
$  git config --global user.name “csparkGenius”
$  git config --global user.email “csg@cambridgespark.com”
$  git config --global core.editor vim
     › https://help.github.com/articles/associating-text-editors-with-git/
$  git config --list
```

CAMBRIDGE SPARK





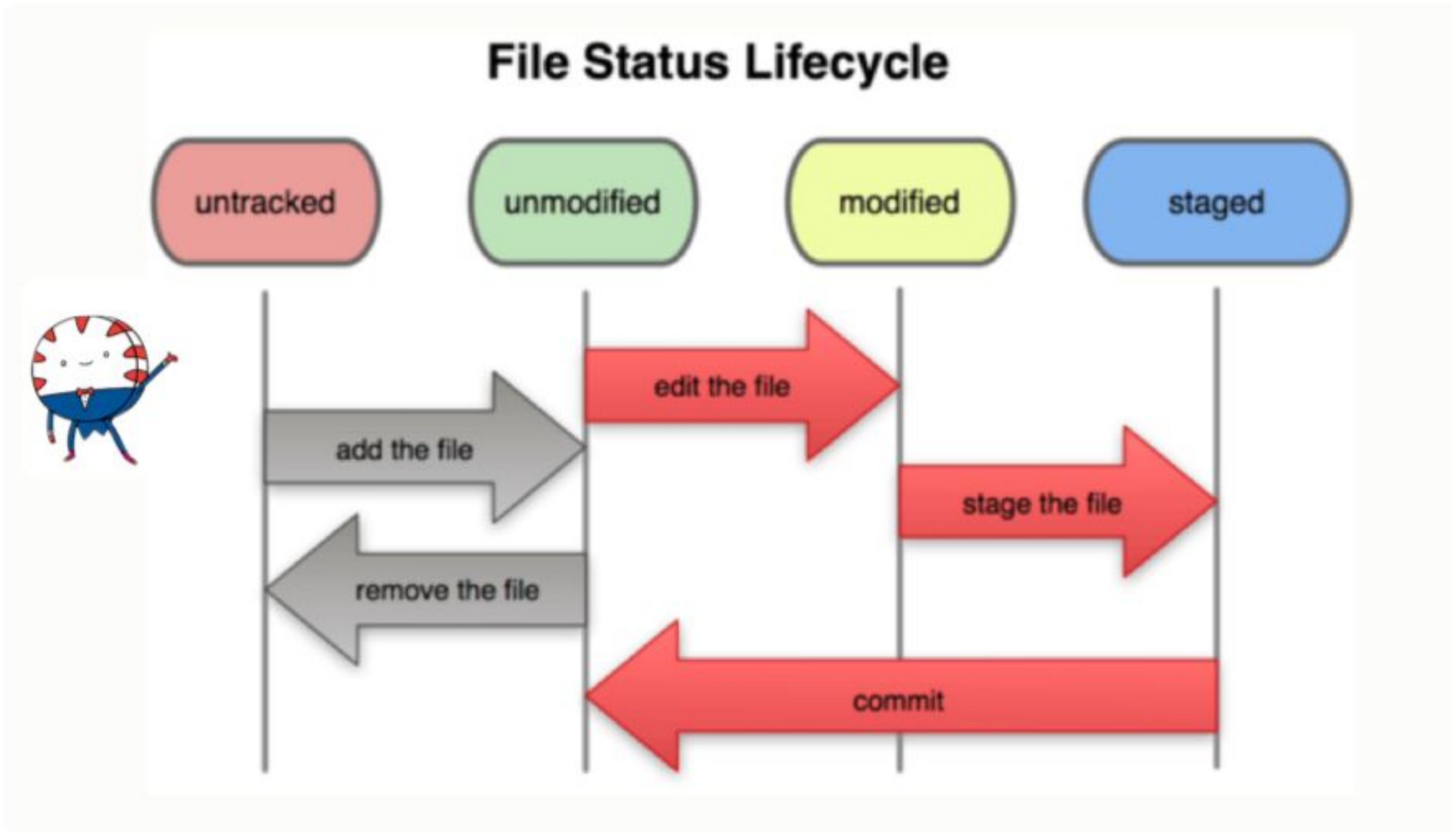
I'M MAKING NEW EDITIONS TO FILES

MAKING A STORY OF THE CHANGES

PUBLISH THE UPDATES

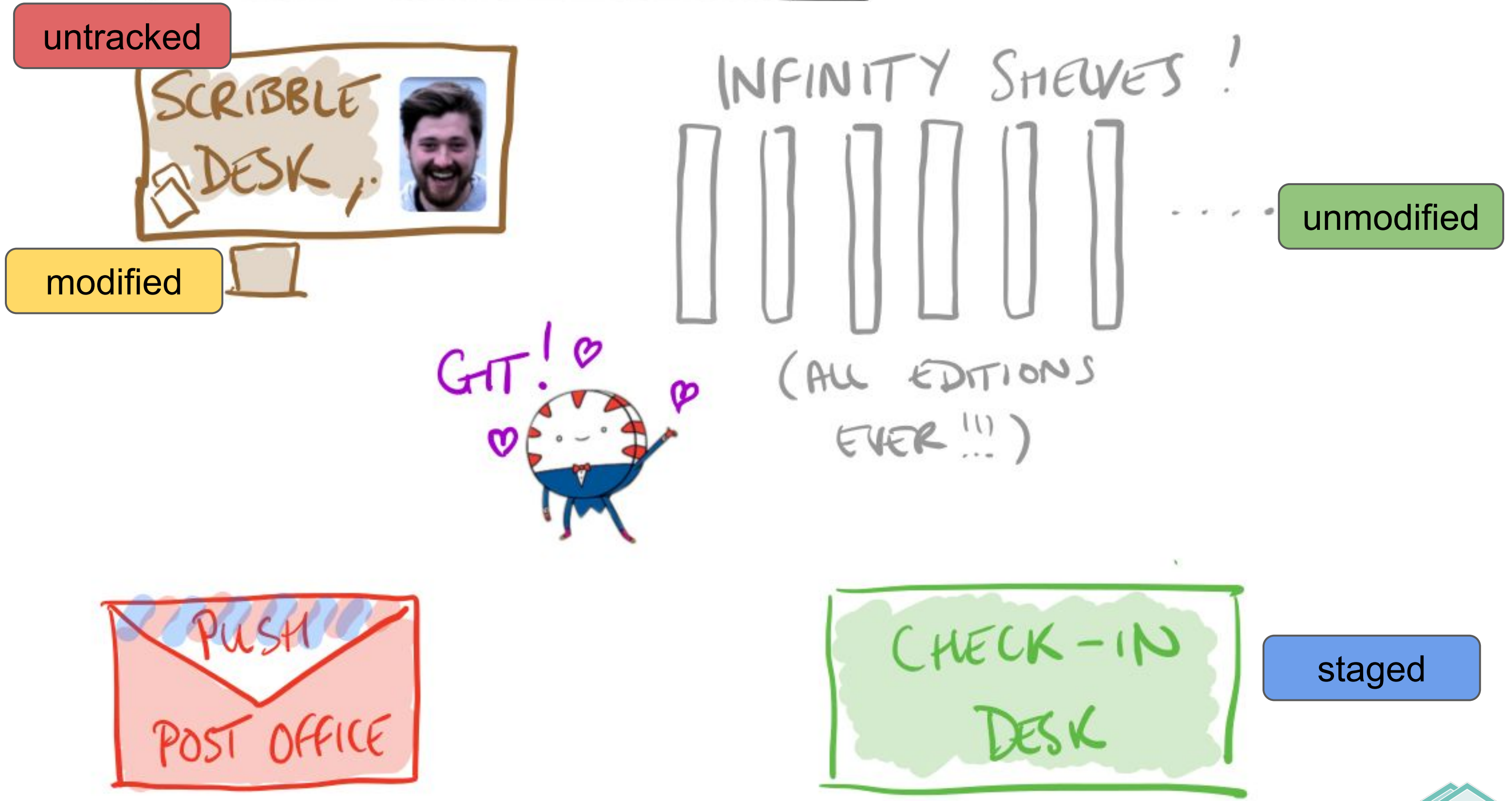
MY RESEARCH BUTLER!

CAMBRIDGE SPARK



CAMBRIDGE SPARK

LIBRARY ANALOGY



CAMBRIDGE SPARK

A first repository...

```
$  cd /somewhere/to/make/a/repo
$  mkdir myfirstrepo
$  cd myfirstrepo
$  git status
$  git init
$  git status
     >  On branch master
        No commits yet
        nothing to commit (create/copy files and use
        "git add" to track)
```

CAMBRIDGE SPARK

...A first repository (adding a file)

```
$  touch file.txt
$  git status
    >  ...
       Untracked files: ...
           file.txt
$  git add file.txt
$  git status
    >  ...
       Changes to be committed: ...
           new file:   file.txt
```

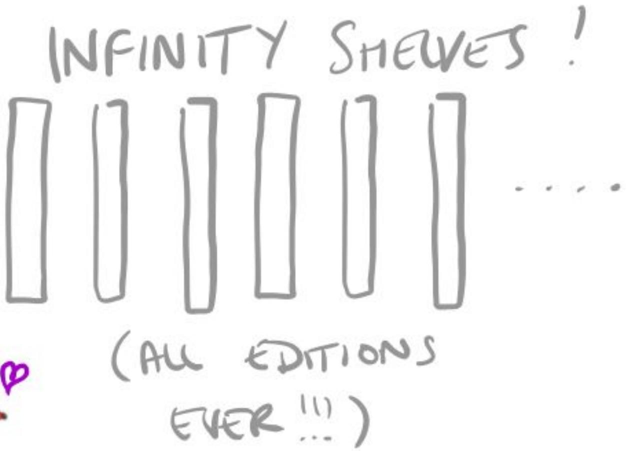
```
touch file.txt =
echo.>file.txt
```

CAMBRIDGE SPARK

LIBRARY ANALOGY

untracked

SCRIBBLE
DESK



PUSH
POST OFFICE

CHECK-IN
DESK

staged

CAMBRIDGE SPARK

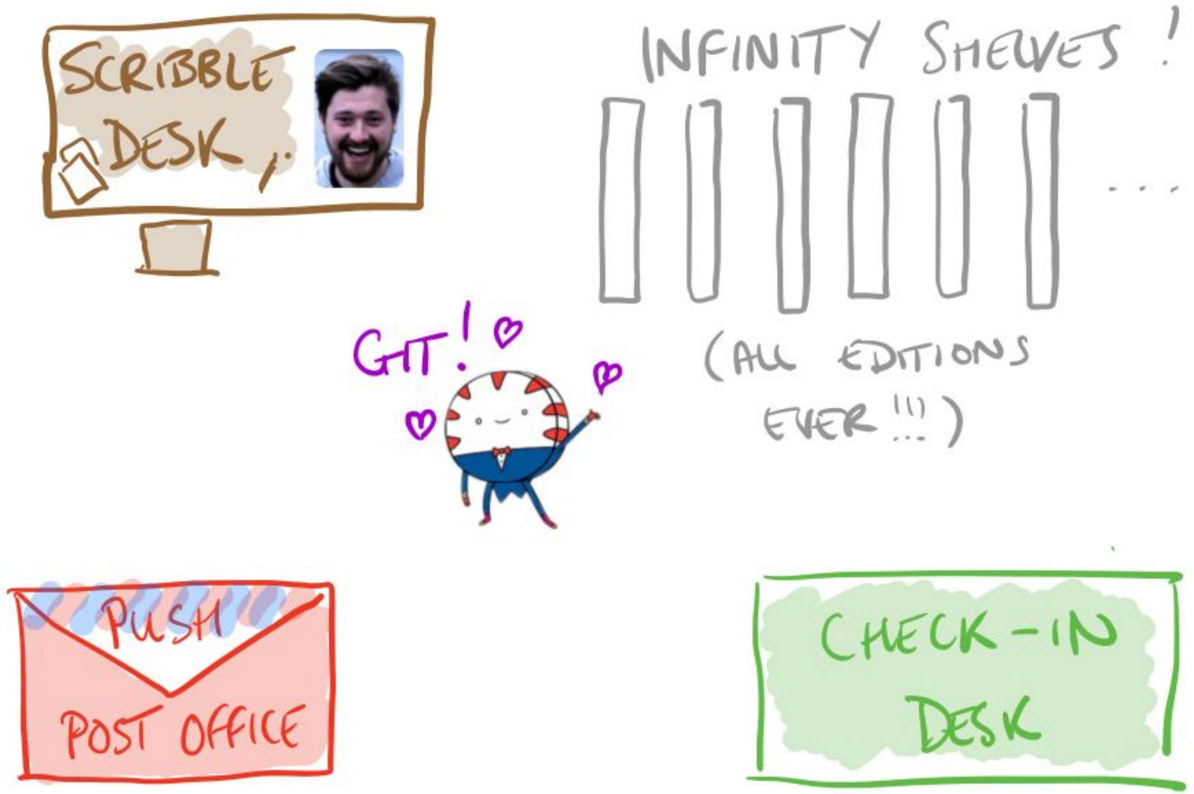
...A first repository (committing a file)

```
$  git commit -m ‘Initial commit’
$  git status
    >  On branch master
       nothing to commit, working tree clean
$  git log
    >  commit 819513fae9... (HEAD -> master)
       Author: csparkGenius <csg@cambridgespark.com>
       Date:   Thu Jan 01 00:00:00 1970 +0000

           Initial commit
```

CAMBRIDGE SPARK

LIBRARY ANALOGY



unmodified

staged

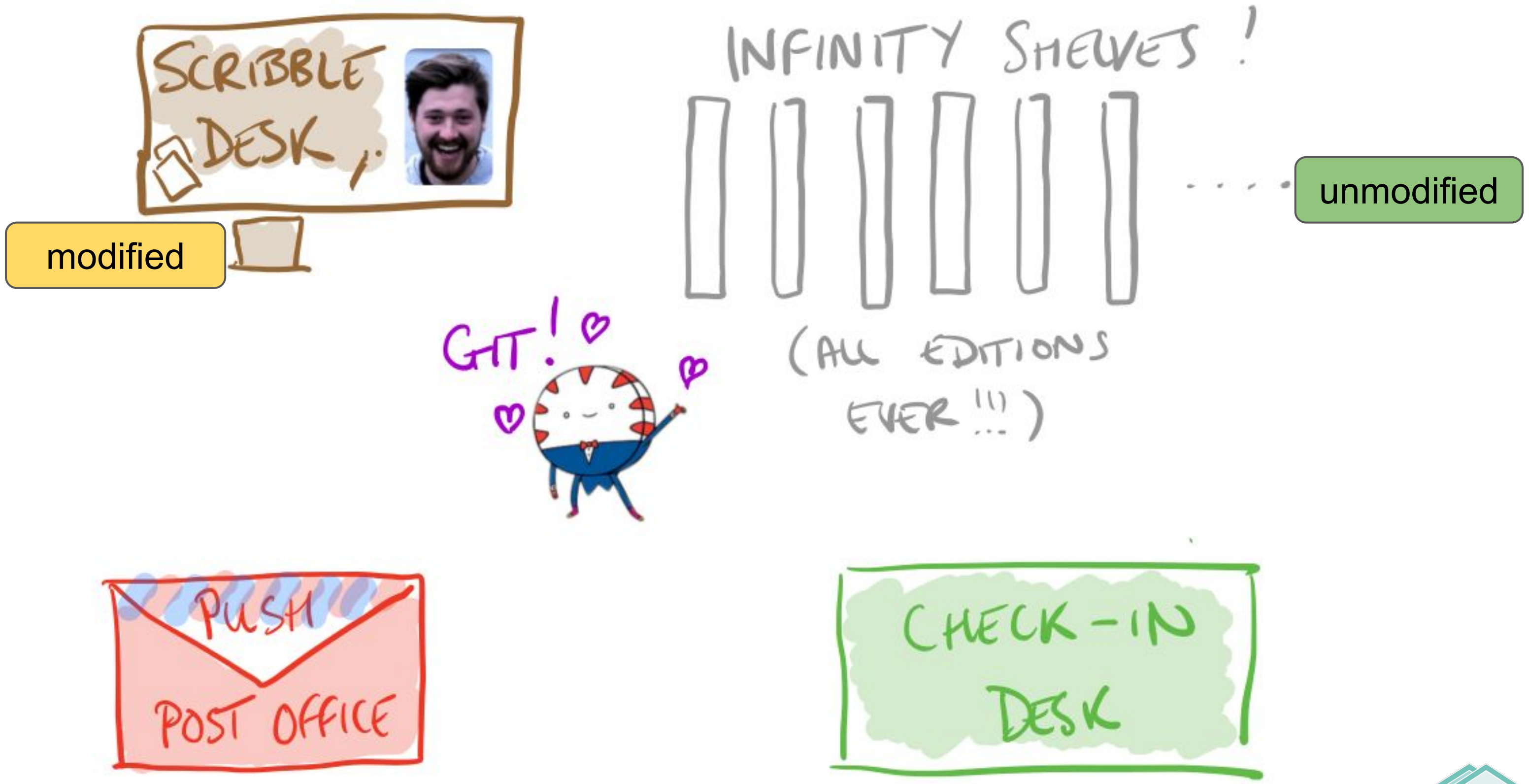
CAMBRIDGE SPARK

...A first repository (modifying a file)

```
$   echo ‘such changes’ > file.txt
$   git status
      >   ...Changes not staged for commit:...
                modified:    file.txt
$   git add file.txt
$   git status
      >   ...Changes to be committed:...
                modified:    file.txt
$   git commit -m “Changes added”
$   git log
      >   ...[log with 2 entries]...
```

CAMBRIDGE SPARK

LIBRARY ANALOGY



CAMBRIDGE SPARK

Remote repositories: GitLab, Github, Bitbucket, ...

PRACTICAL SESSION! **You** with a **partner**:

1. Everybody create a public gitlab repository called myfirstrepo
2. Link **your local repository** to a gitlab repository
   a. Follow 'Existing Git repository' instructions @ bottom of https://gitlab.com/<yourname>/myfirstrepo
3. Clone your **partner's gitlab repository**
   a. `git clone https://gitlab.com/<theirname>/myfirstrepo.git partnerfirstrepo`
   b. Be sure to either rename their repo (as above does to `partnerfirstrepo`) or just clone in diff folder
4. Make a change to your **partner's repository** `git add, git commit`
5. Push your changes to **their repository** `git push`
6. Pull the changes made to **your repository** `git pull`

CAMBRIDGE SPARK

Useful git commands

```
$  git status
$  git config --list
$  git add
$  git commit
$  git commit file.txt  # adds and commits a modified file
$  git commit -a  # adds and commits all modified files!
$  git clone
$  git remote -v  # shows where you’re pulling and pushing to
$  git pull / git push
$  git remote add origin ...
```

CAMBRIDGE SPARK