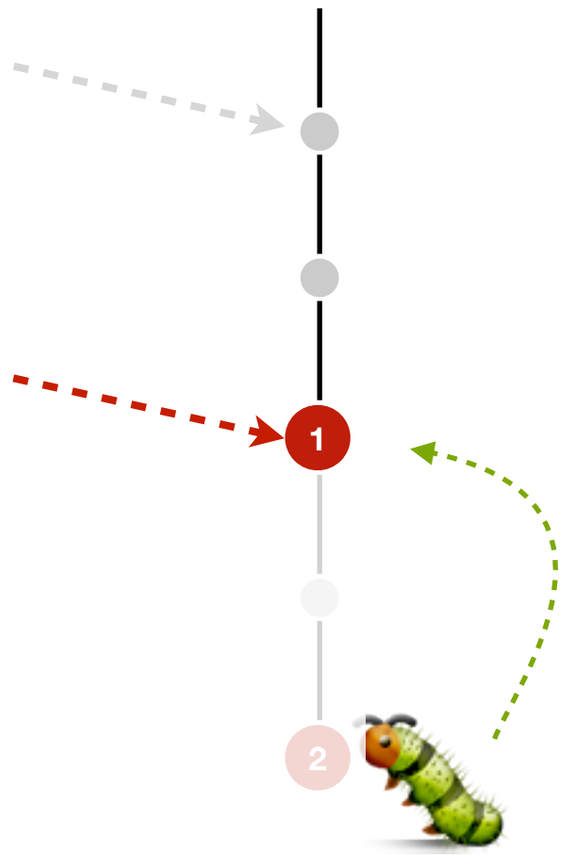
Distributed Version Control with **git**

Nick Hawes
n.a.hawes@cs.bham.ac.uk

MotorDemo.java

file saved

at important version



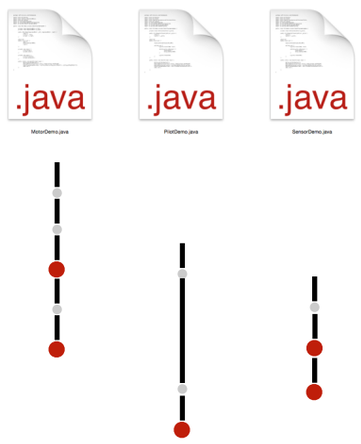
bug found so revert

MotorDemo.java

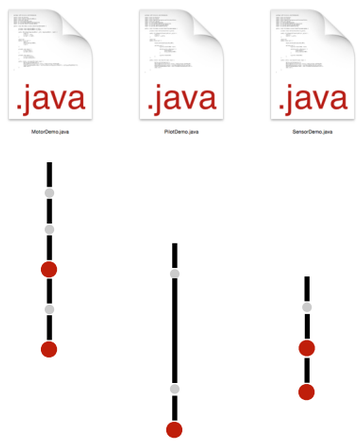
PilotDemo.java

SensorDemo.java

work at uni



backup



a **distributed** **version-control** system lets you

a) explicitly manage different **versions** of your files

b) **distribute** your files to different machines

c) collaboratively work on files

a **distributed** **version-control** system

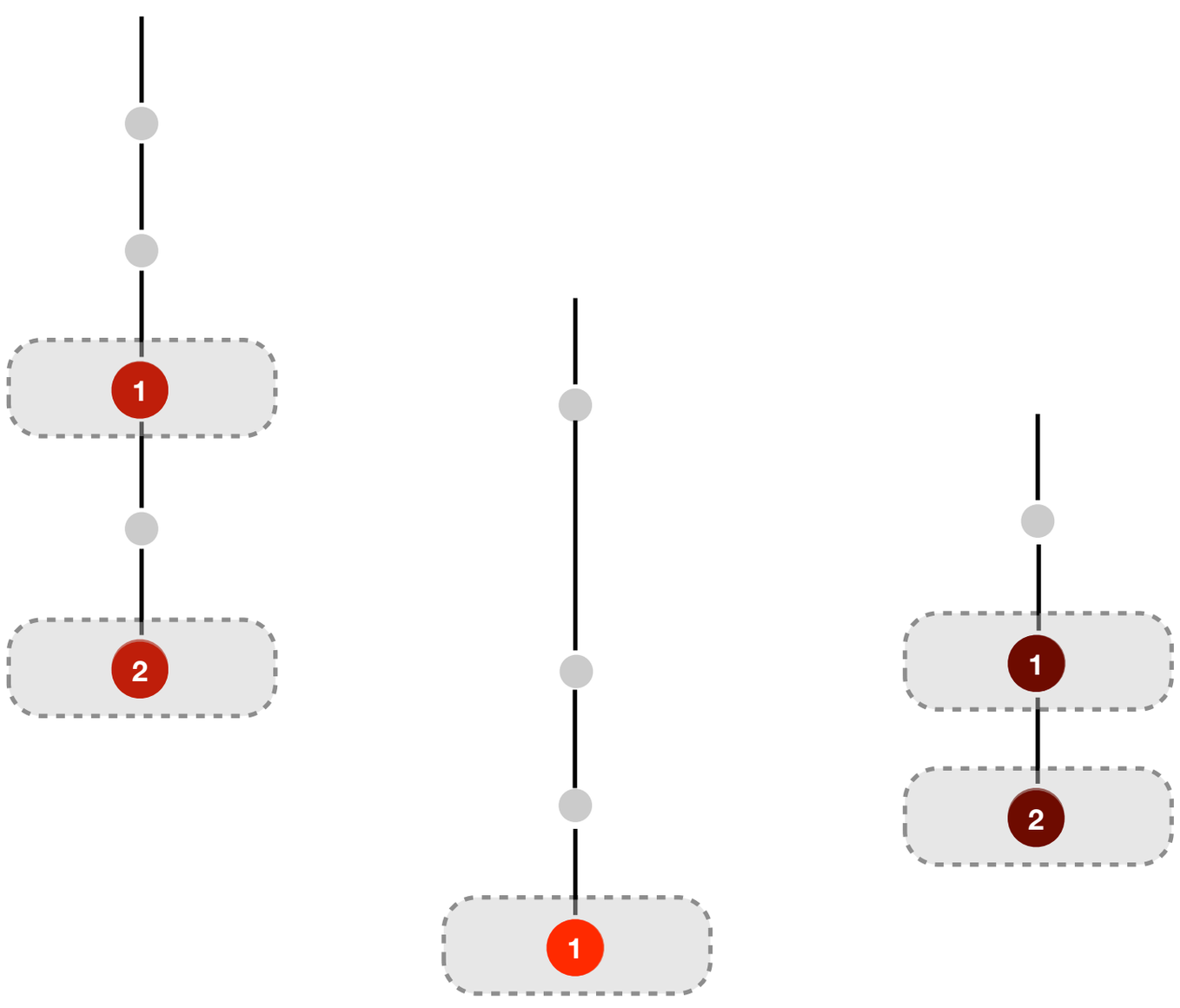
http://git-scm.com

MotorDemo.java

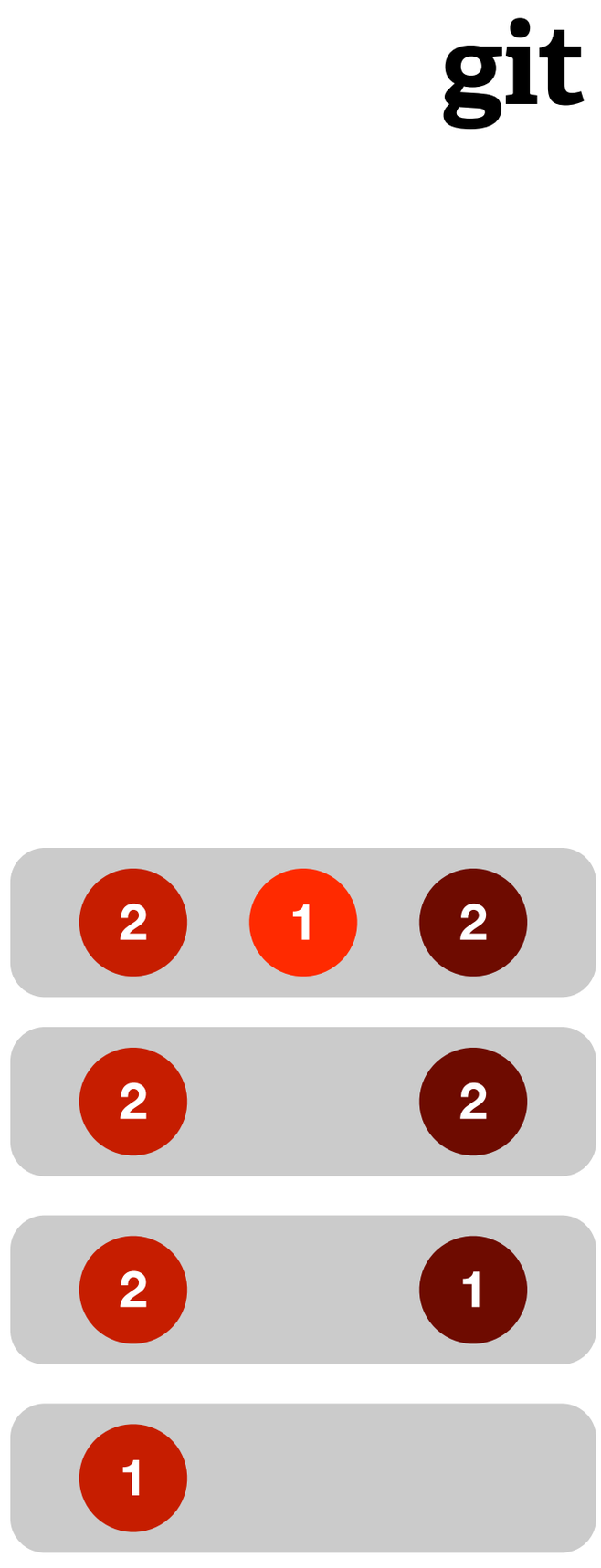
PilotDemo.java

SensorDemo.java

git



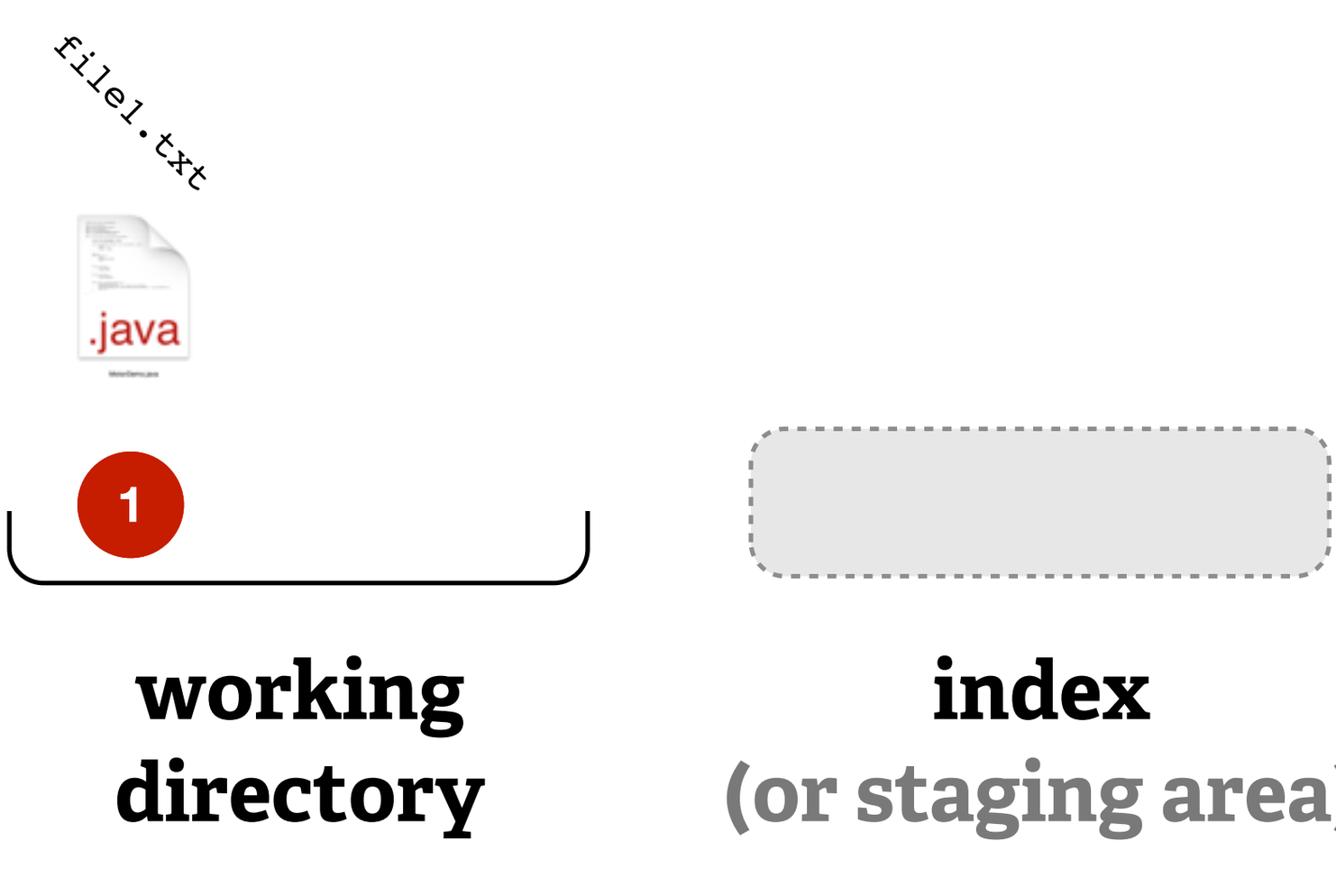
working directory



repository

```
$ git add file1.txt
```

this command creates a *copy* of `file1.txt` in the **index**



```
$ git commit
```

this command adds the contents of the **index** as a new **commit**

your local **repository**

file1.txt

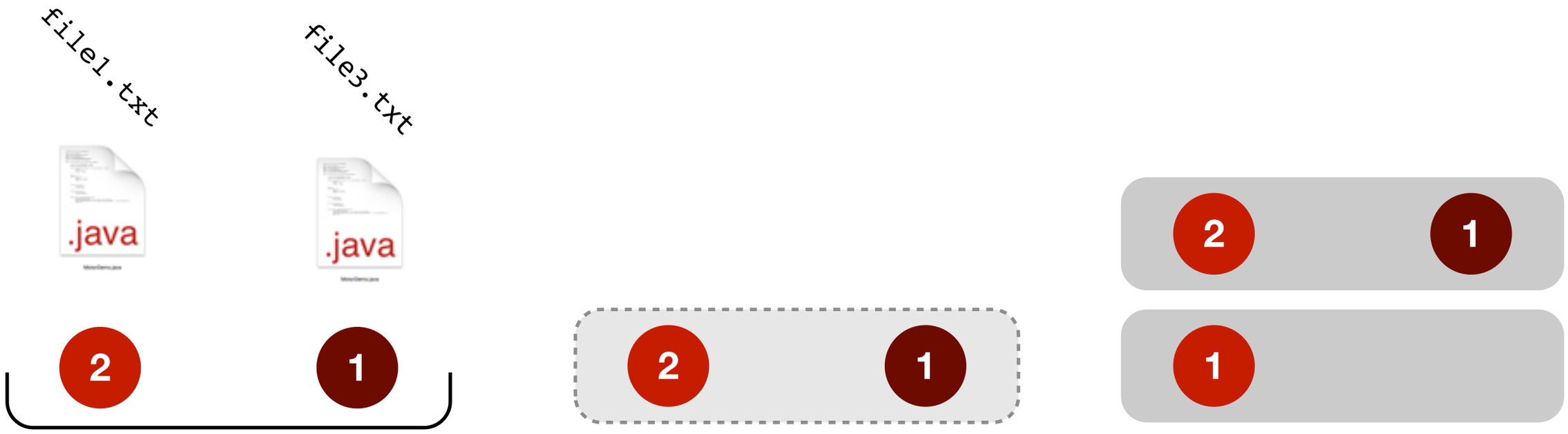
working directory

index
(or staging area)

repository

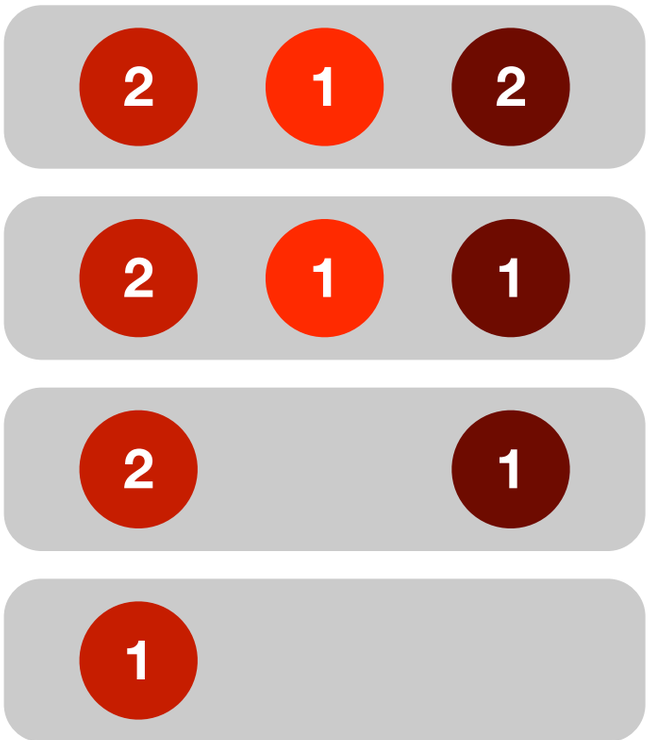
```
$ git add file1.txt

$ git add file3.txt
```

```
$ git commit
```


file1.txt
file3.txt
.java
.java
2
1
2
1
2
1
1
working
directory
index
(or staging area)
repository

working directory

index
(or staging area)



repository

```
$ git init <repo>
```

creates an empty git **repository** <repo>

```
$ git status
```

shows the status of the **working directory**

```
$ git log
```

shows the history of **commits**

```
$ git commit -a
```

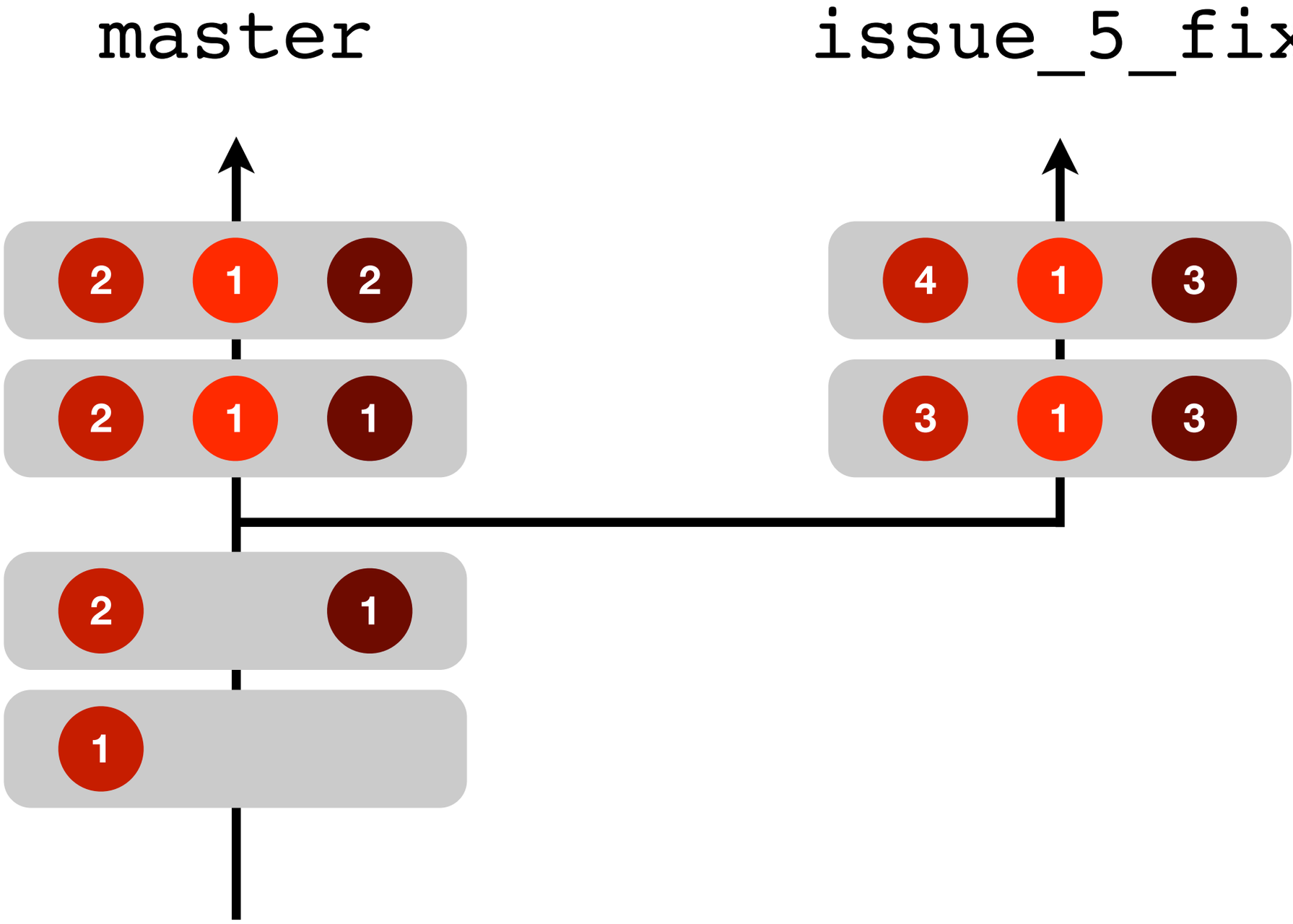
adds and commits all known, changed `files`

```
$ git commit --amend
```

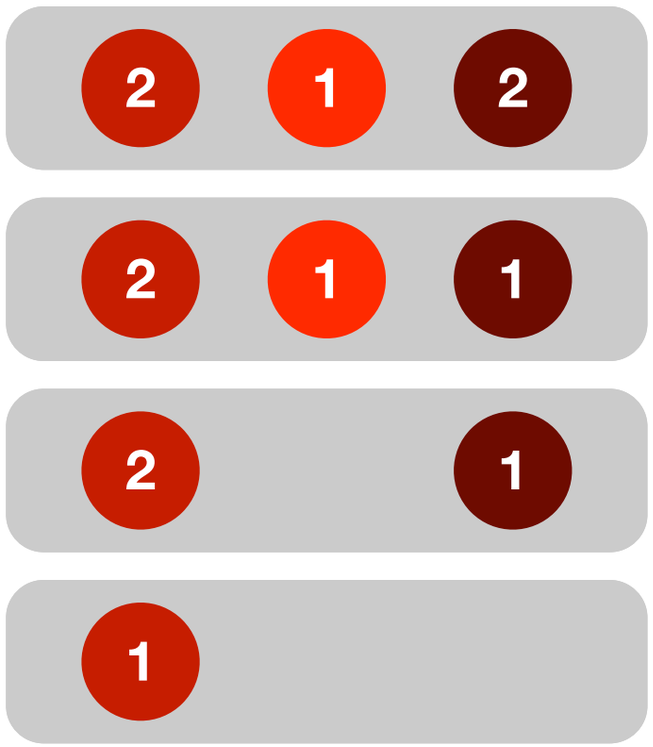
alter the last **commit**

(this should be done *before* `push`ing to a remote **repository**)

branches



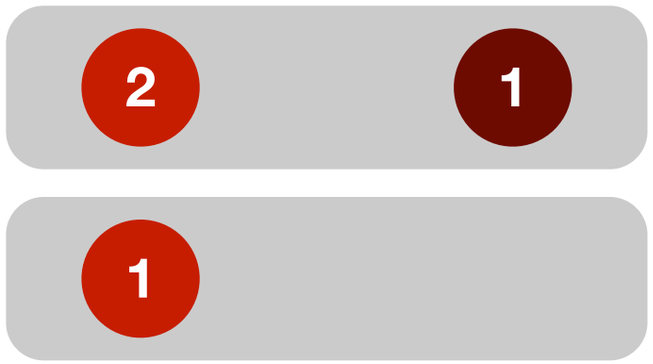
```
$ git push
```



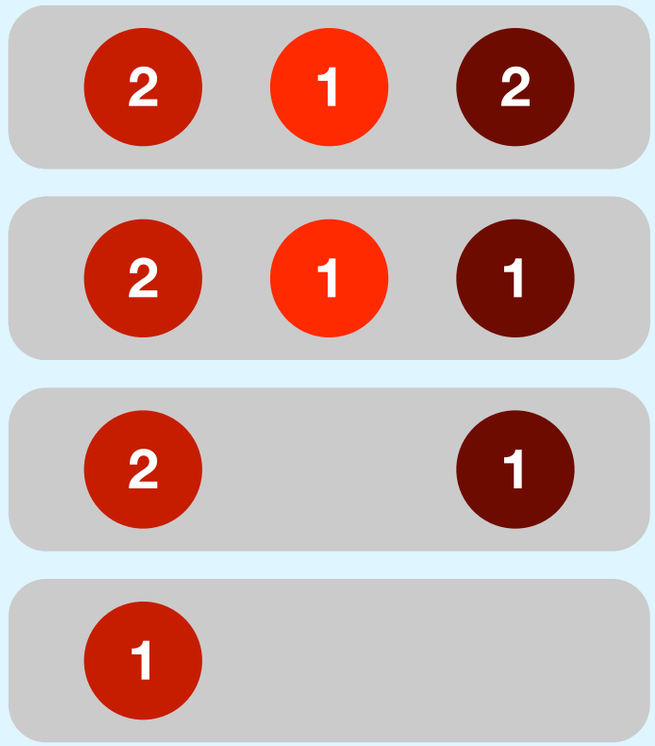
local **repository**

remote **repository**

```
$ git pull
```



local **repository**



remote **repository**

GitHub

Atlassian Bitbucket

pull

push

local **repository**

remote **repository**

remote **repository**

pull

push

local **repository**

push

pull

local **repository**

repository

pull
push

pull
push

pull
push

pull
push

pull

push

repository

repository

upstream **repository**

pull

push

local **repository**

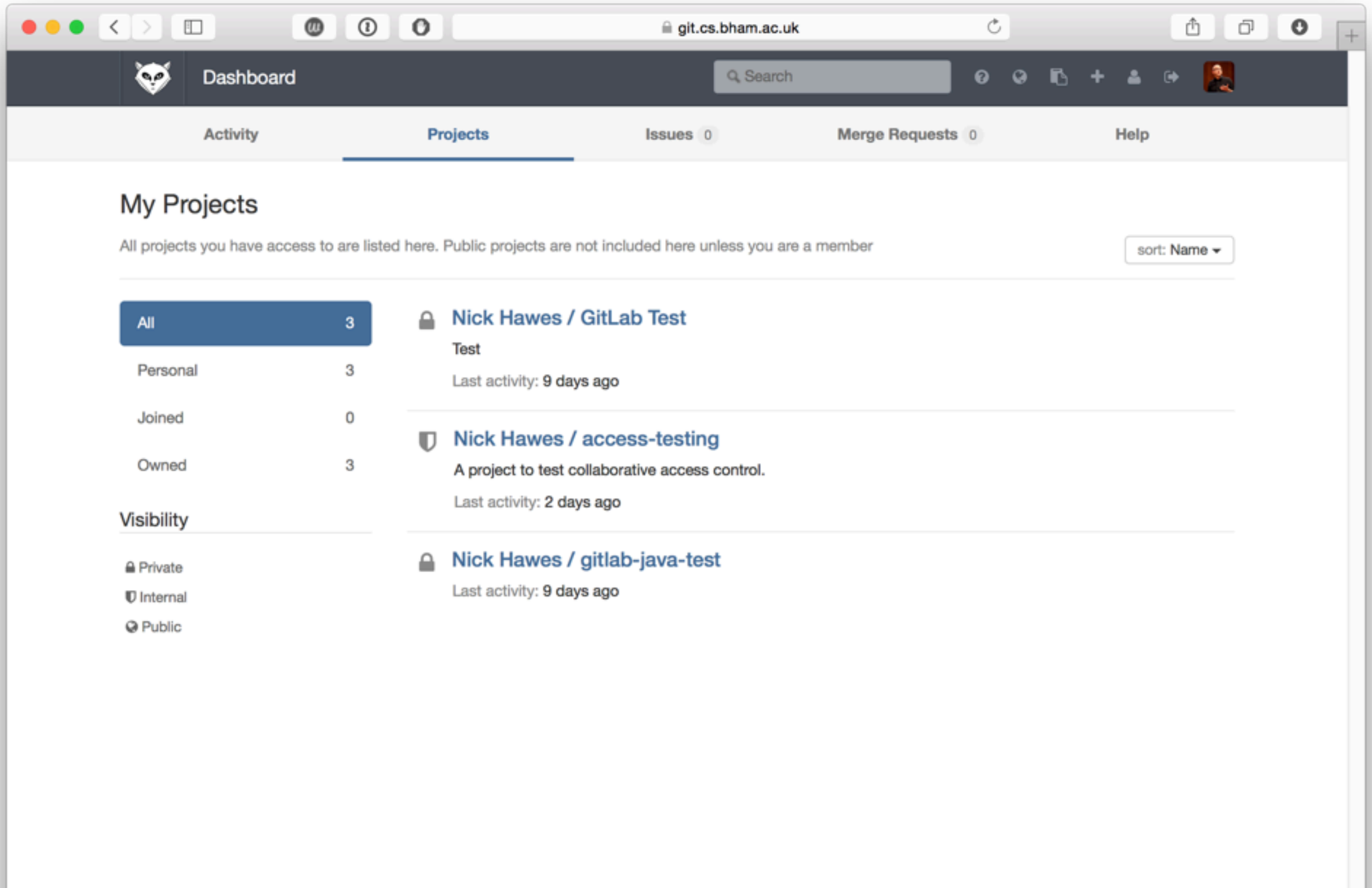
push

pull

local **repository**

GitLab

https://git.cs.bham.ac.uk/



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

adds the remote git **repository** `<url>` which can be referred to with **`<name>`**

```
$ git push -u <name> <ref>
```

update **branch** `<ref>` in remote **repository** `<name>`

(**`-u`** sets this default for future **`push`**es and **`pull`**s)

```
$ git clone <url>
```

creates a local copy of the remote **repository** found at `<url>`

upstream **repository**

pull

push

local **repository**

push

pull

local **repository**

when **`push`**ing and **`pull`**ing git tries to automatically **merge** `files` which have changed in both **repositories**

if this is not possible you must fix **conflicts** manually

edit each **conflicted** `file` to produce the contents you require then **`git add`** it

a **conflicted** `file`

local reference
(**`HEAD`** is roughly an alias
for your current branch)

```
<<<<<<< HEAD
Here is some nicer text. And some more.
And yes more.
=======
Here is some boring text. And some more.
And yes more.
>>>>>>>
dc29cc29732dbc0c0bcb07f958a176d0fcc4133a
```

conflicted parts
are marked by
angled brackets

remote **commit**
reference

a **conflicted** file

```
$ git pull
```

eclipse supports git via the `Team` menu, using a plugin called **EGit**

the interface isn't particularly great

the recommended way for you to use git is via the **command line**

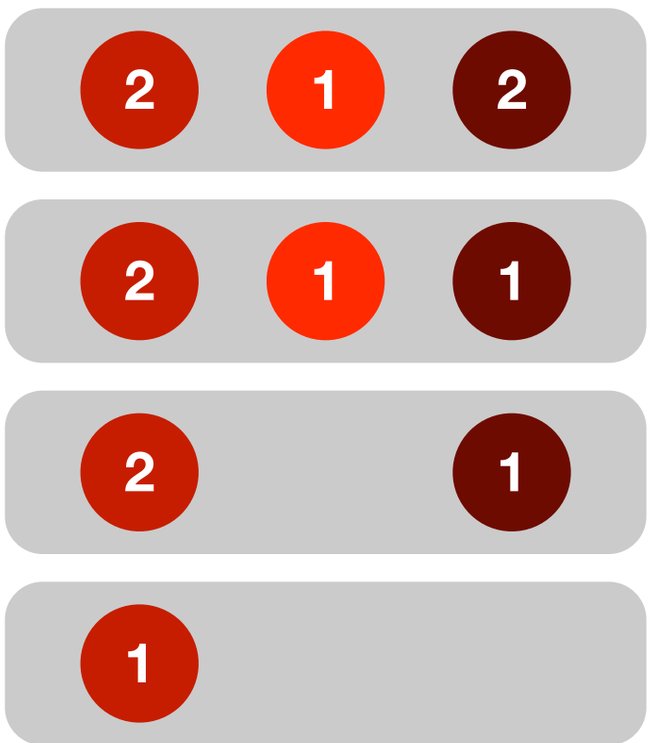
practice makes a huge difference with **git**

most employers will expect you to use version control *fluently*

so start using it for **all your code** now

working directory

index
(or staging area)



repository

https://canvas.bham.ac.uk/courses/10065/assignments/26165

Git Exercise: available now, due Jan 23 at 11:55pm