

GIT

What is GIT ?

- Git is a version control system
- Git provides revision information
- Git provides development and performance statistics
- Git provides a means of keeping track of deviations/forks/branches
- **NOT A DEROGATORY TERM FOR AN UNDESIRABLE**

What is a version control system ?

- A piece of software that houses reference to previous iterations of work
- Why is this particularly useful with regard to software ?
- Why is tis particularly useful with regard to SCRUM development ?

Why GIT

- Git provides a peer versioning architecture.
- What does this mean ?
 - No central server
 - Effectively every version is a branch or a fork (not exactly)
 - No check in check out system

Advantages of peer-system

- No central server method utilised, no single point of contact
- No lockouts on development tasks
- Merges are achieved on device rather than using server resources
- Expense

Disadvantages of peer-system

- Elements can be edited offline
- Confusion as to where the master lies
- Some branching issues can occur

Version control features include

- Add
- Update
- Remove
- Request
- Fork

For this next bit...useful bash commands

- rm - remove (-rf force recursive)
- rmdir - remove directory
- cd - change directory
- ls - list contents
- mv – move
- cp - copy

Git Command Structure

- Git <<command>> <<argument>> <<switch>>

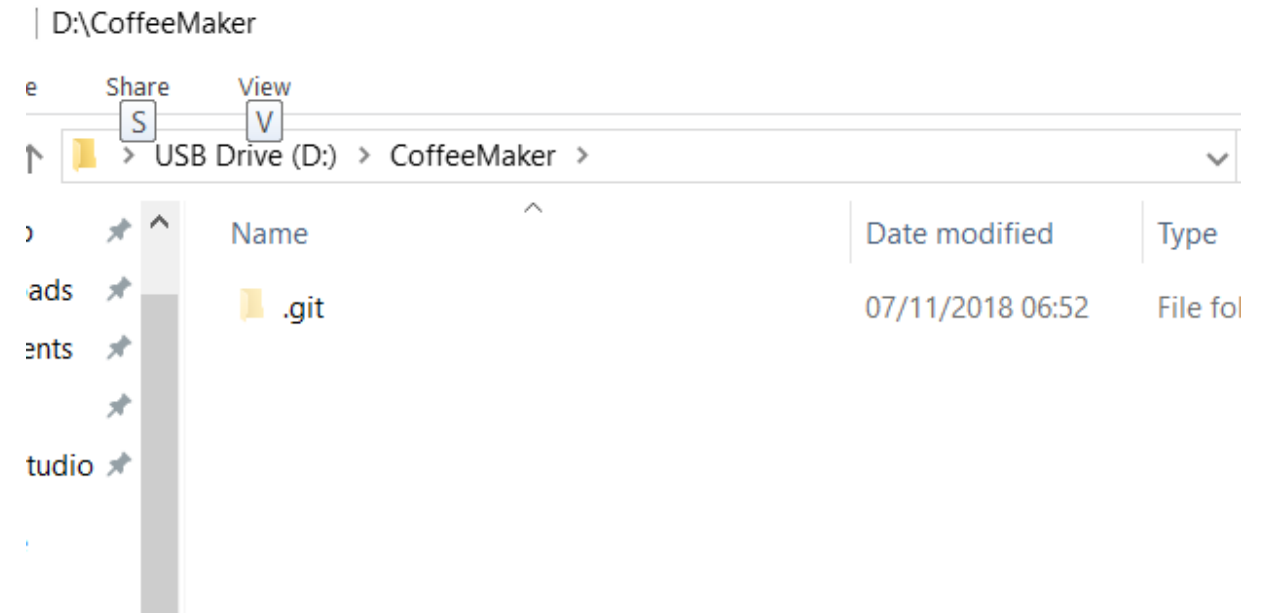
First there was

- In order to attribute versioning information to anything, a directory structure for such attribution needs to be created.
- How is this achieved ?
 - Initialisation

Git- Version control features

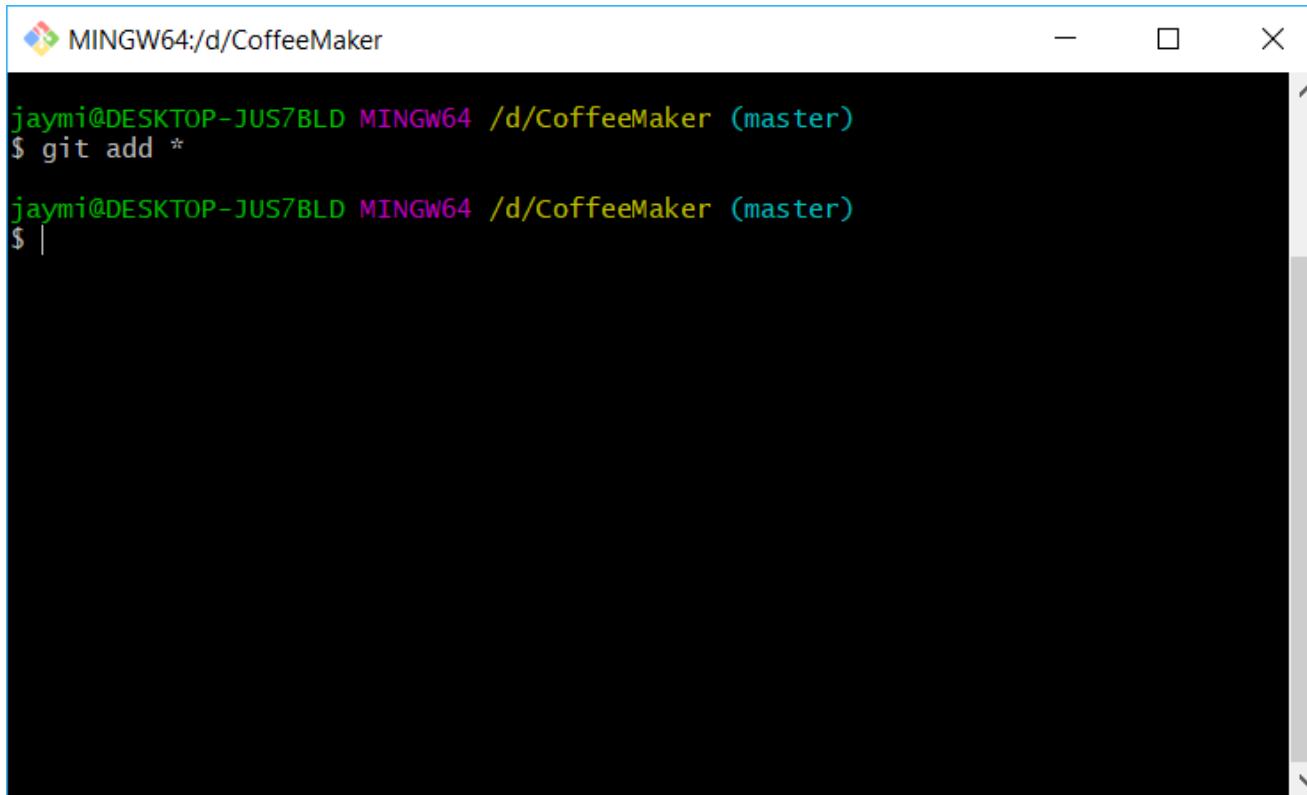
- Init – Initialises a repository in the specified directory

```
MINGW64:/d/CoffeeMaker
jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker
$ git init
Initialized empty Git repository in D:/CoffeeMaker/.git/
jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ |
```



Adding files to a project

- add – Adds files to the project

A screenshot of a terminal window titled 'MINGW64:/d/CoffeeMaker'. The window has a black background with green and yellow text. The prompt 'jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)' is shown twice. The first time, the command '\$ git add *' is entered. The second time, the prompt is shown with a cursor after '\$'.

```
MINGW64:/d/CoffeeMaker
jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ git add *
jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ |
```

Committing changes to working repository

- Commit- confirm the changes made within the local repo

```
MINGW64:d/CoffeeMaker
```

```
# Please enter the commit message for your changes. Lines starting  
# with '#' will be ignored, and an empty message aborts the commit.  
#  
# On branch master  
#  
# Initial commit  
#  
# Changes to be committed:  
#       new file:   Source.cpp  
#  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
  
D:/CoffeeMaker/.git/COMMIT_EDITMSG [unix] (07:05 07/11/2018)  
"D:/CoffeeMaker/.git/COMMIT_EDITMSG" [unix] 11L, 232C
```

```
MINGW64:/d/CoffeeMaker

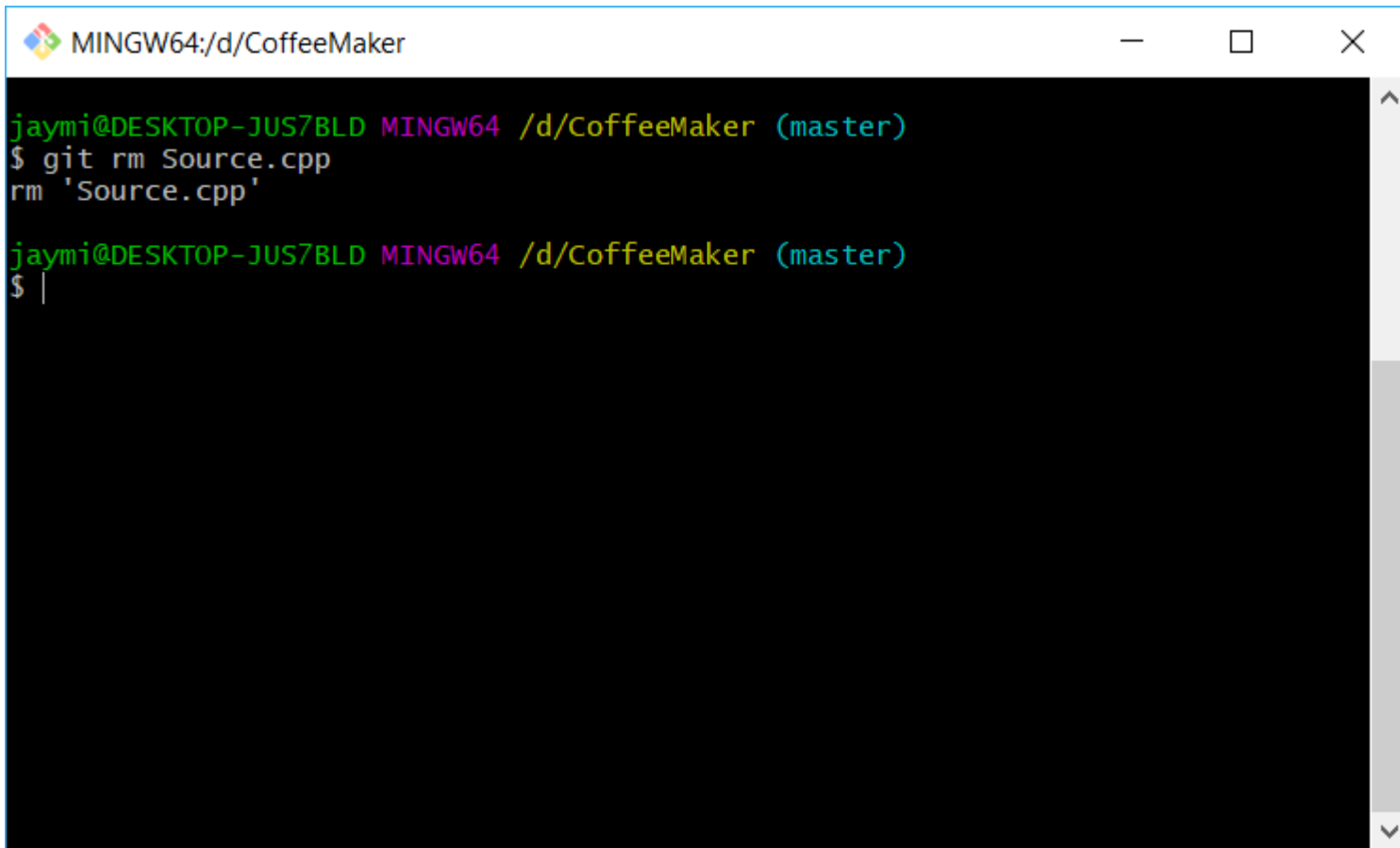
jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ git commit
Aborting commit due to empty commit message.

jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ git commit -m "Initial Commit"
[master (root-commit) fc5ac06] Initial Commit
1 file changed, 54 insertions(+)
create mode 100644 Source.cpp

jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ |
```

Removing files from a project

- `rm` – removes from local files not from the entire repository or history

A screenshot of a Windows terminal window titled "MINGW64:/d/CoffeeMaker". The terminal shows a user named "jaymi" at a desktop named "JUS7BLD" in a "MINGW64" environment, working on the "master" branch of a project in the directory "/d/CoffeeMaker". The user enters the command "git rm Source.cpp", which is followed by the shell command "rm 'Source.cpp'". The prompt then returns to the user, and the terminal shows the next prompt "\$ |" with a cursor.

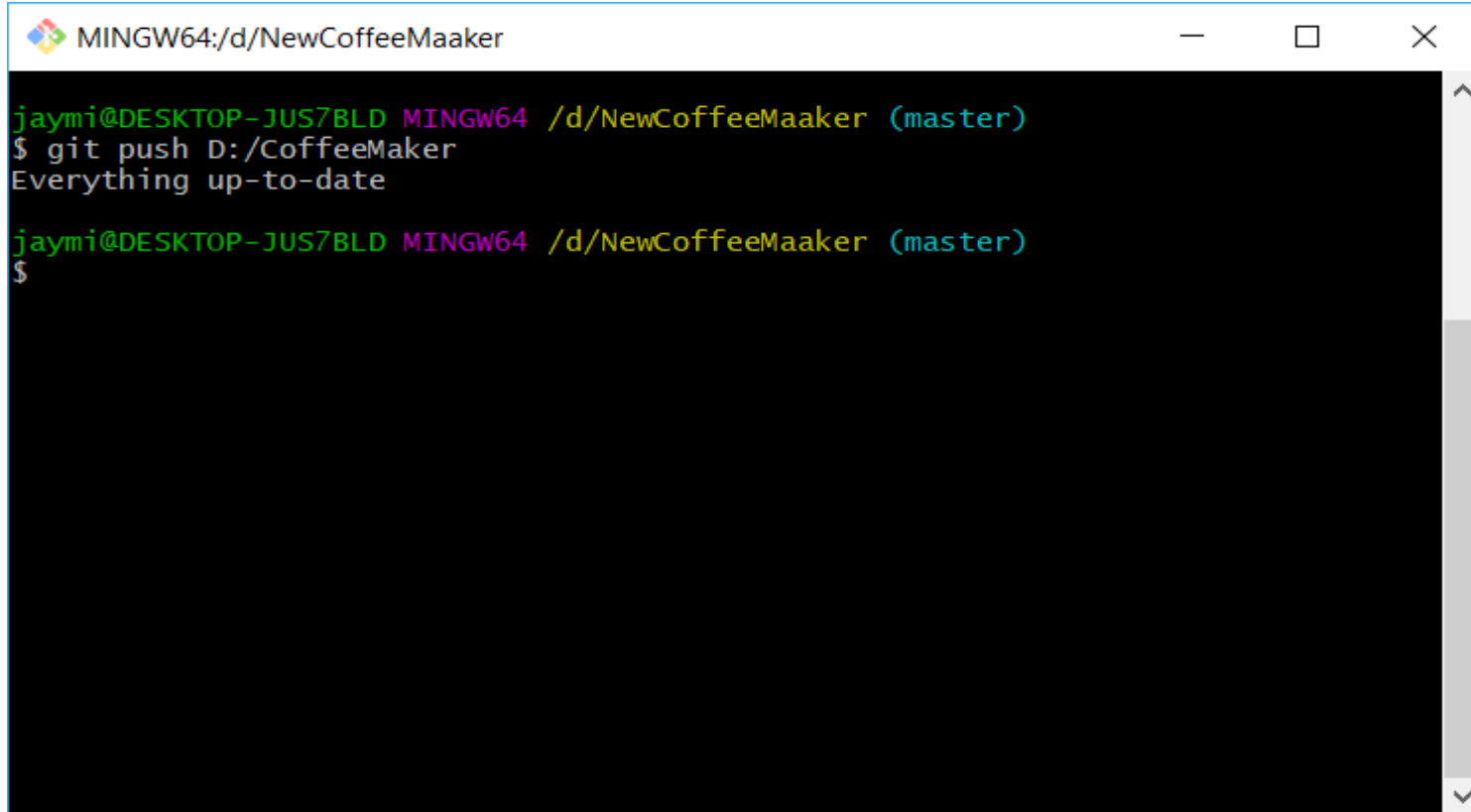
```
MINGW64:/d/CoffeeMaker

jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ git rm Source.cpp
rm 'Source.cpp'

jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ |
```

Updating central repository

- Push – passes all local changes to what is considered the main path

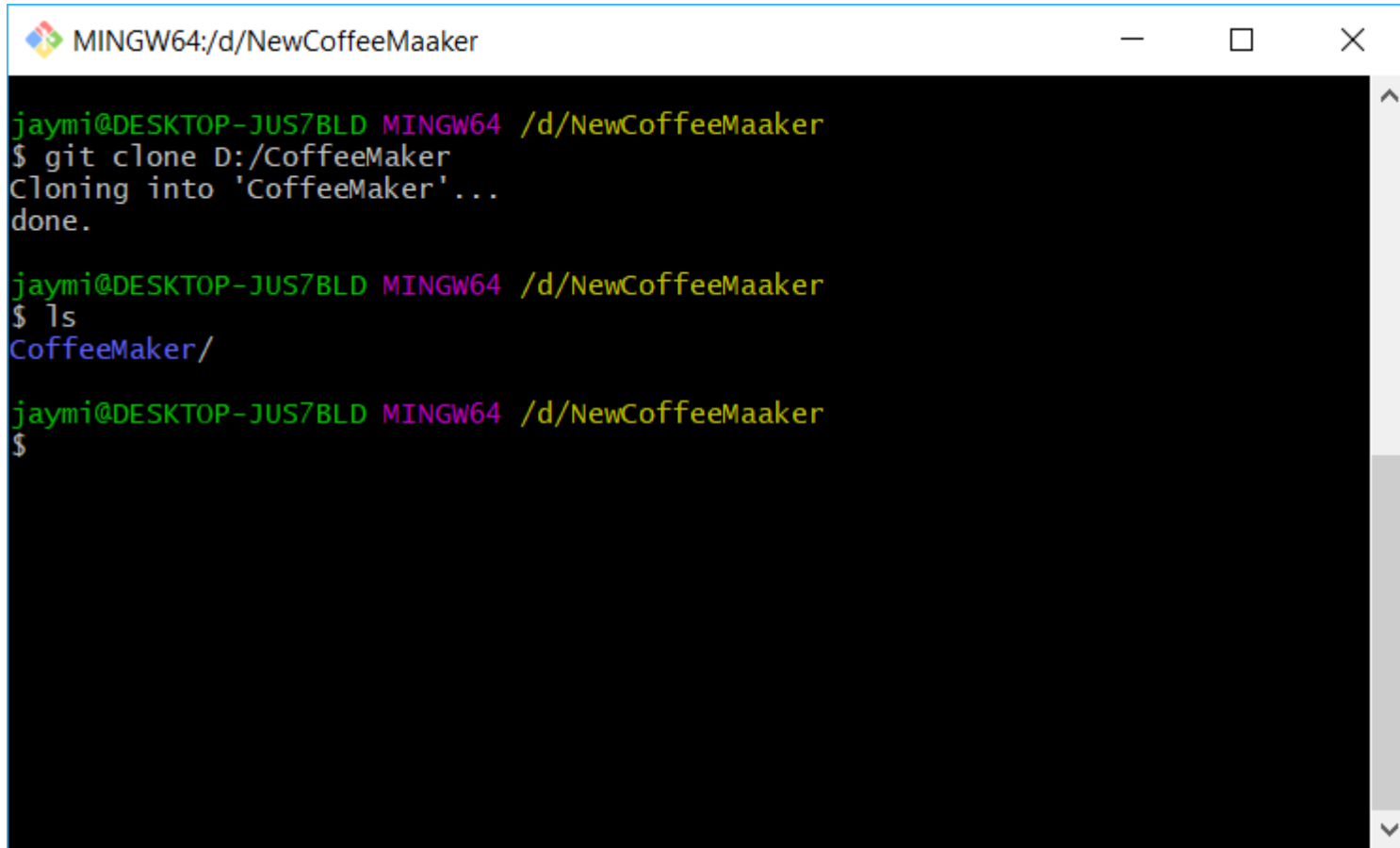
A screenshot of a Windows terminal window titled 'MINGW64:/d/NewCoffeeMaaker'. The window has standard Windows window controls (minimize, maximize, close) in the top right. The terminal shows a user named 'jaymi' at a desktop named 'JUS7BLD' in a 'MINGW64' environment, located in the directory '/d/NewCoffeeMaaker' on the 'master' branch. The user enters the command '\$ git push D:/CoffeeMaker', and the terminal responds with 'Everything up-to-date'. The prompt '\$' appears again on the next line.

```
MINGW64:/d/NewCoffeeMaaker

jaymi@DESKTOP-JUS7BLD MINGW64 /d/NewCoffeeMaaker (master)
$ git push D:/CoffeeMaker
Everything up-to-date

jaymi@DESKTOP-JUS7BLD MINGW64 /d/NewCoffeeMaaker (master)
$
```

Getting latest version in a repository



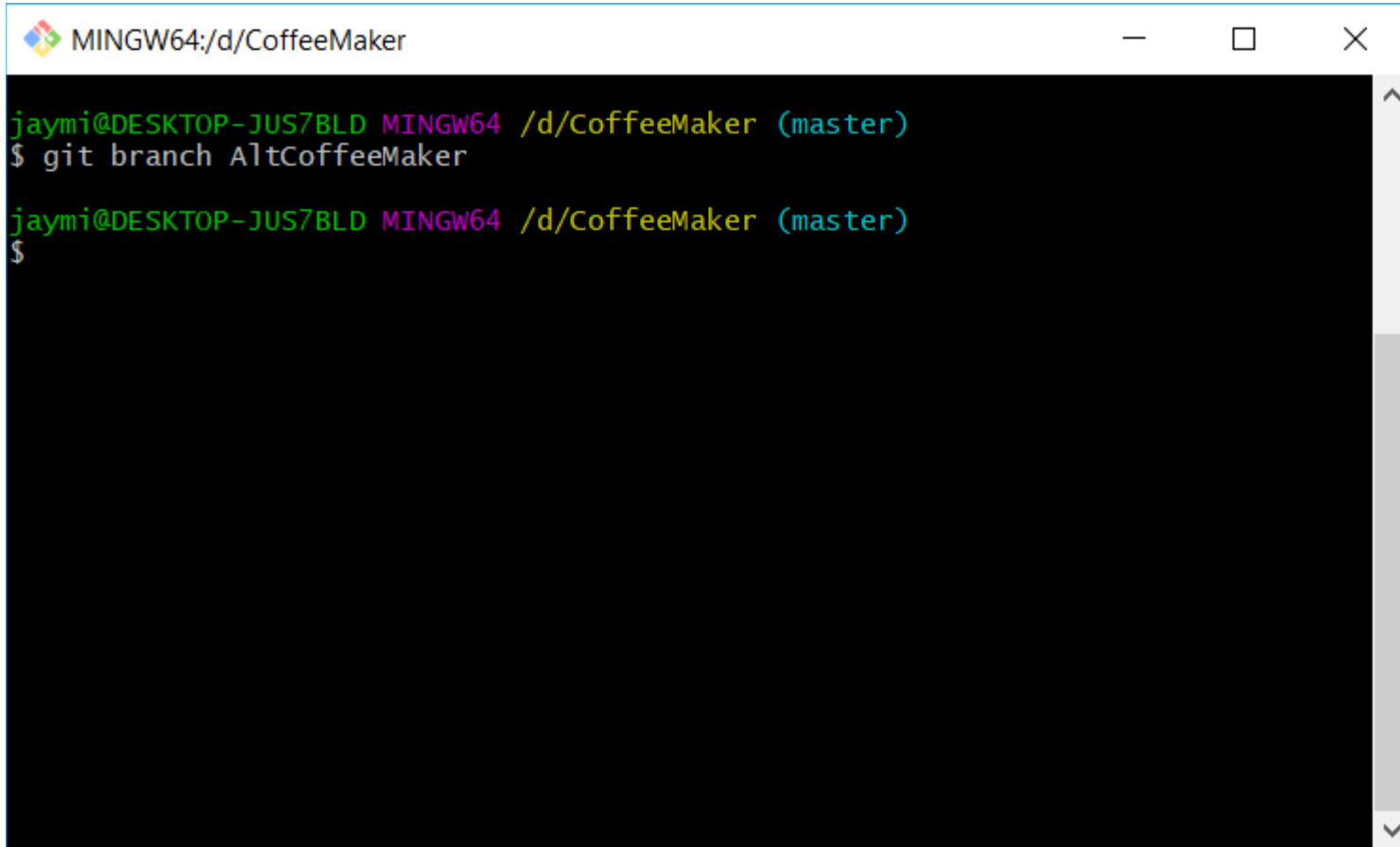
```
MINGW64:/d/NewCoffeeMaaker
jaymi@DESKTOP-JUS7BLD MINGW64 /d/NewCoffeeMaaker
$ git clone D:/CoffeeMaker
Cloning into 'CoffeeMaker'...
done.

jaymi@DESKTOP-JUS7BLD MINGW64 /d/NewCoffeeMaaker
$ ls
CoffeeMaker/

jaymi@DESKTOP-JUS7BLD MINGW64 /d/NewCoffeeMaaker
$
```

Branching

- Branch <<name>> - creates a branch with the defined name

A screenshot of a Windows terminal window titled "MINGW64:/d/CoffeeMaker". The window has standard Windows window controls (minimize, maximize, close) in the top right corner. The terminal shows a user named "jaymi" at a desktop named "DESKTOP-JUS7BLD" in a "MINGW64" environment, located in the directory "/d/CoffeeMaker" on the "master" branch. The user enters the command "git branch AltCoffeeMaker". The prompt returns, and the user enters another dollar sign "\$" on the next line.

```
MINGW64:/d/CoffeeMaker
jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ git branch AltCoffeeMaker
jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$
```

Getting information

- Status – dumps information relating to the repository

```
MINGW64:/d/CoffeeMaker

jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ git status
On branch master
nothing to commit, working tree clean

jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ |
```

```
MINGW64:/d/CoffeeMaker

jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$ git log
commit 9ceeff7d859f0ef1a07d33e0e700bc34d1fcc9d4 (HEAD -> master, AltCoffeeMaker)
Author: Jaymie Phillips
Date:   Wed Nov 7 07:12:05 2018 +0000

    Full Code

commit ddb781e8969a88ac1656a73034f4a7202ddba7cb
Author: Jaymie Phillips
Date:   Wed Nov 7 07:09:52 2018 +0000

    Update Commit

commit fc5ac06db164aeca2178a67858b6698d8d291822
Author: Jaymie Phillips
Date:   Wed Nov 7 07:07:10 2018 +0000

    Initial Commit

jaymi@DESKTOP-JUS7BLD MINGW64 /d/CoffeeMaker (master)
$
```

So how else can we control the information

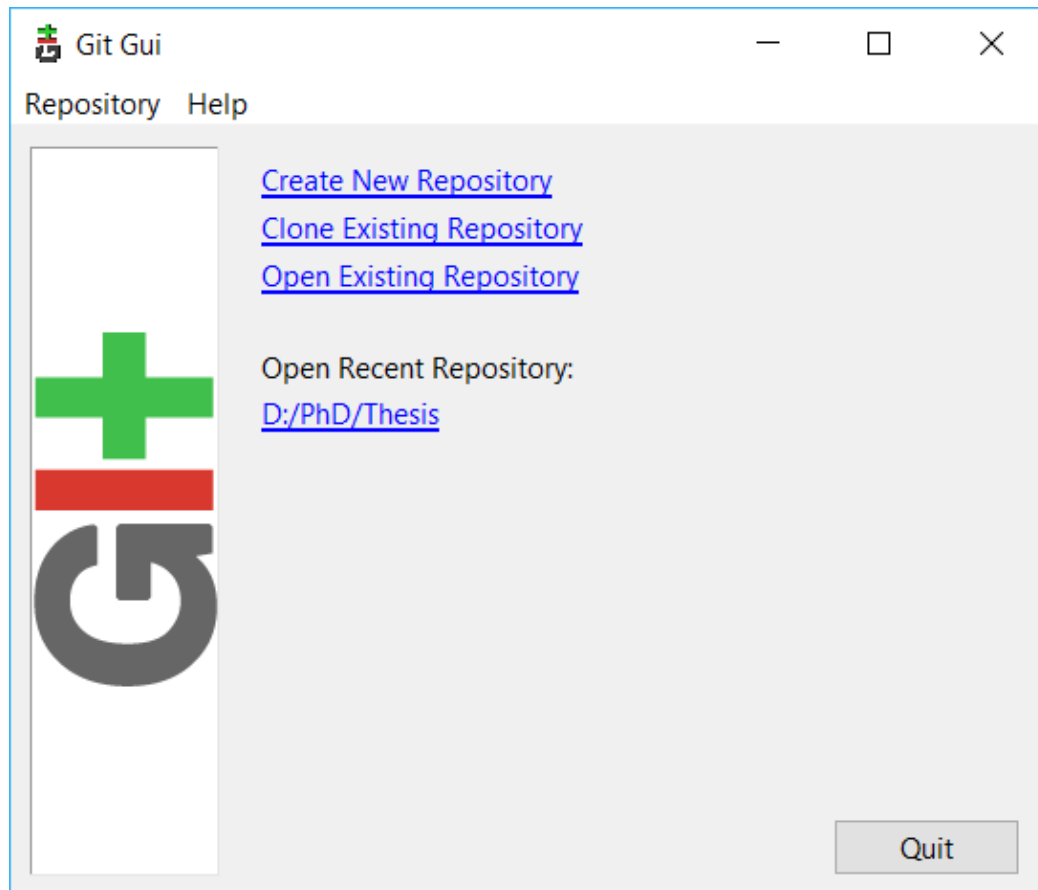
- Use of BASH scripts
- The use of GIT scripts
- The use of gitignore lists

Typical gitignore file

```
$ git status
[...]
# Untracked files:
[...]
#       Documentation/foo.html
#       Documentation/gitignore.html
#       file.o
#       lib.a
#       src/internal.o
[...]
$ cat .git/info/exclude
# ignore objects and archives, anywhere in the tree.
*.[oa]
$ cat Documentation/.gitignore
# ignore generated html files,
*.html
# except foo.html which is maintained by hand
!foo.html
$ git status
[...]
# Untracked files:
[...]
#       Documentation/foo.html
[...]
```

Are there other ways to monitor or control repositories ?

- Yes... through graphical user interfaces





Bitbucket



Your work



Repositories



Projects



Pull requests



Issues



Snippets



Your work



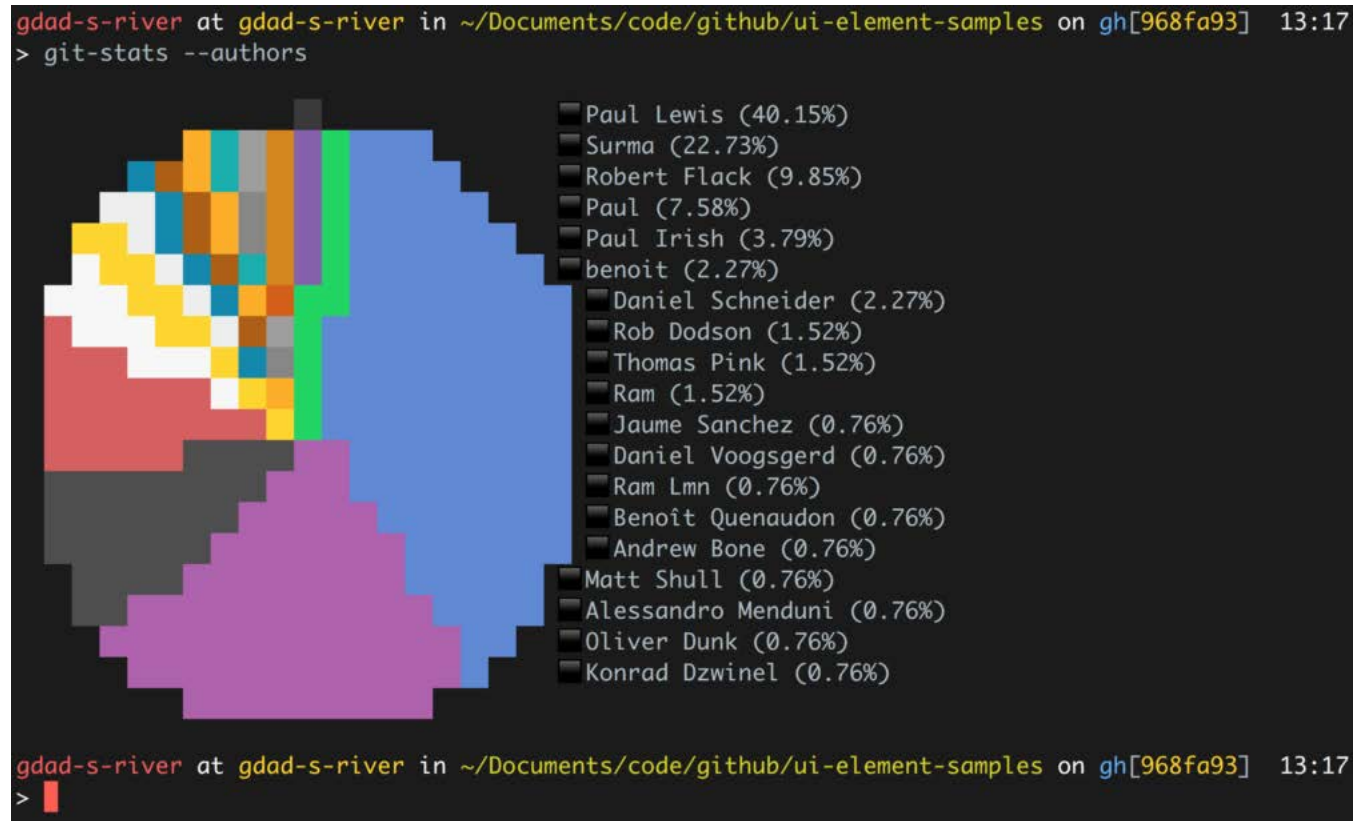
Here's where your work shines through

Set up a repository to get going with your code. After that, you'll find your relevant repositories and work right here.

[Create repository](#)[Import repository](#)

Other ways to view detailed information

- gitstats



Usefulness

- Can store changes
- Create personal versions of software projects
- Try new ideas
- Restore from an old version