Git and R Markdown

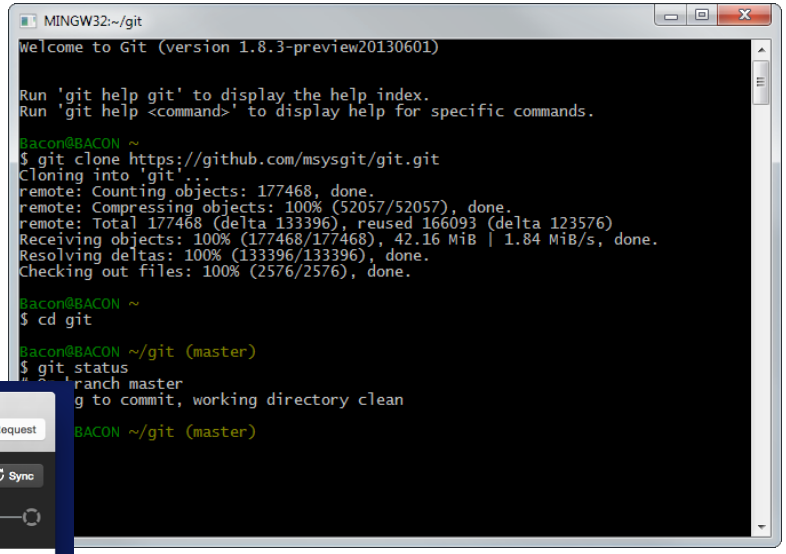
Why they are useful and how to get started

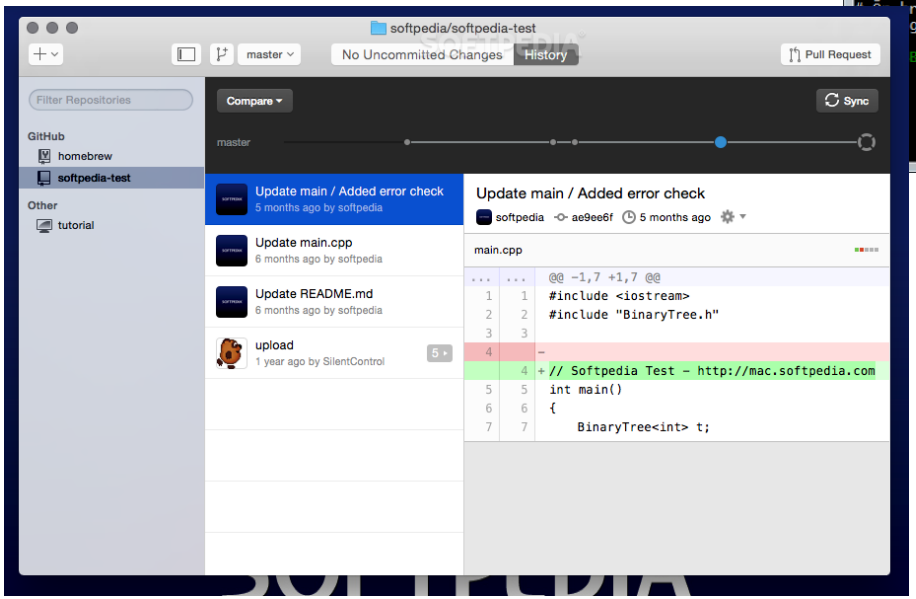
What is Git?

- Distributed version control
- Manages changes without overwriting them

What is Git?

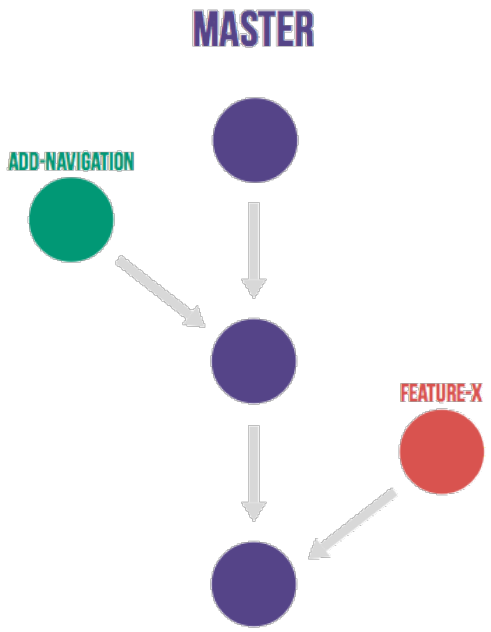
- Distributed version control
- Shell software and GUIs





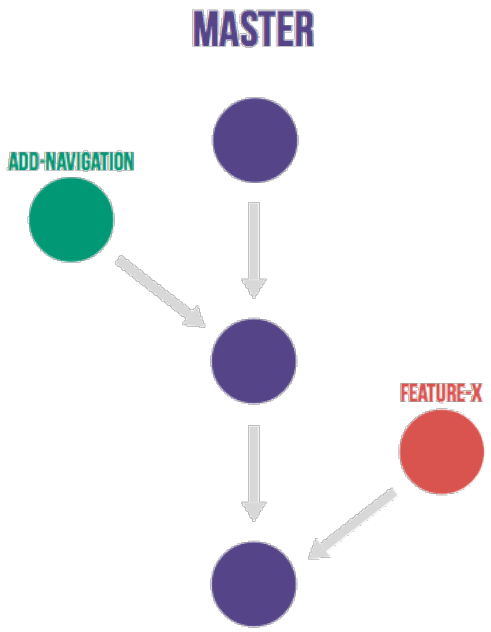
What is Git?

- Distributed version control
- Manages changes without overwriting them
- Local and remote copies



What is Git?

- Distributed version control
- Manages changes without overwriting them
- Local and remote copies
- GitHub: Collaborative platform and hosting service

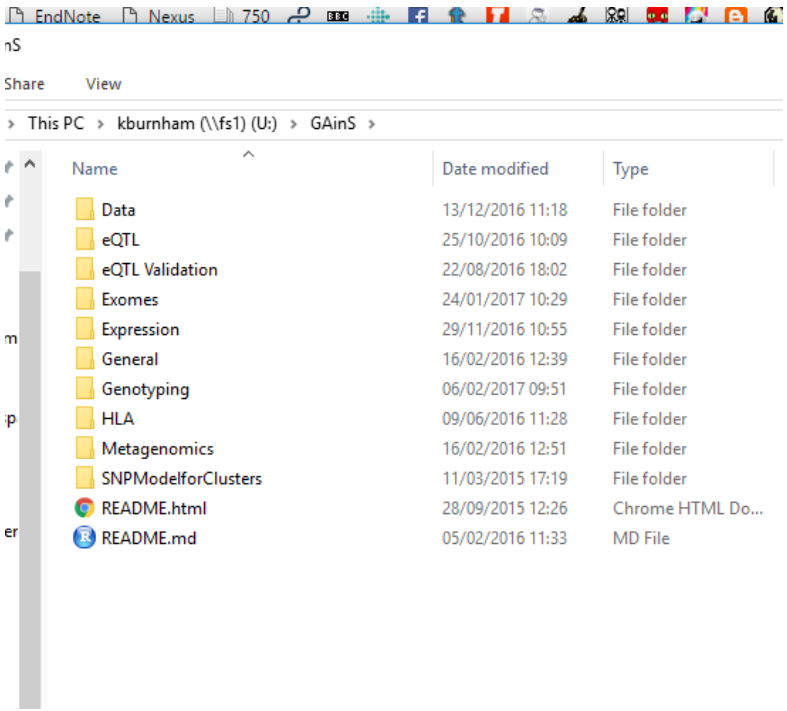


Why use Git?

- Code sharing and publishing, a programmer's Linked In?
- Or: Word track changes + Dropbox shared folders
- Not just for code: any type of file
- Not just for coders: GUIs available
- Manages and stores revisions: filing system for drafts
- Joint projects

How do you use Git?

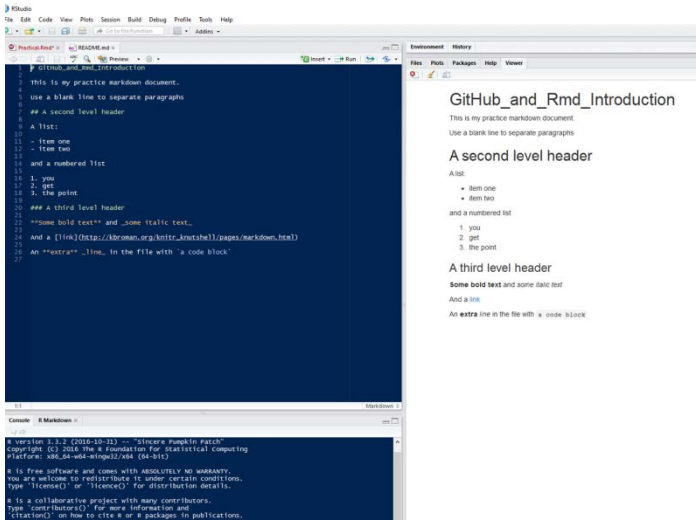
- Usual folder structure on your computer



How do you use Git?

- Usual folder structure on your computer

- Do some work (the hardest part)

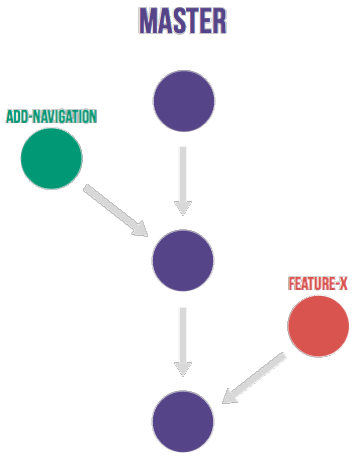


How do you use Git?

- Usual folder structure on your computer
- Do some work
- Save your changes locally; Git notices and tracks them
- Confirm that you want these changes to be remembered
- Synchronise with a central copy of your folder on GitHub

How do you use Git?

- Download from the central copy to another computer (this could be you, or someone else)
- Do some work
- Synchronise (*push*) the changes to GitHub
- *Pull* them down to the first computer



Some terms you will come across

- **Repository** - a directory
- **Remote** - the online copy (also called the **origin**)
- **Clone** - download a whole repository
- **Add** - changes that you **add** are waiting **(staged)** to be stored
- **Commit** - store a snapshot
- **Push** - upload changes
- **Pull** - download changes and merge all together

Some terms you might come across

- **Revert** – reverse previous changes (but still store them)
- **Branch** - a parallel copy of the repository
- **Master** – the main/default branch
- **Fetch** - download changes
- **Checkout** – switch branches or discard changes
- **Merge -** combine branches
- **Fork** – duplicate a repository

What is R Markdown?

- Lightweight markup for interactive documents
- Instruction tags e.g. HTML, LaTeX
  - Word: WYSIWYG
- Readable markup + conversion tool
- Include code (R, Python, SQL) in your text
- Reproducible dynamic documents

Why use Markdown?

- Word + figures = Yuck.
- Generate nice HTML/PDFs/Word docs
- Good integration with GitHub
- REALLY useful for R(/python/bash/SQL) users: integrate code and generate summary docs all in one
- Reproducible and dynamic analysis
- (Parameters, slideshows, notebooks, websites…)

Why use Markdown?

```
<html>
<body>
<h1>Title</h1>
</head>
</html>
```

```
# Title
```

```
\documentclass[a4paper,12pt]
{report}
\textheight=23.5cm
\textwidth=15cm

\section{Title}
```

How do you use R Markdown?

- Open an R Markdown file in R Studio
- *Type as normal*
- Add code in chunks
- Click “Knit” to generate the document
- Open the file created in the same directory