Git and GitHub

What we'll cover today

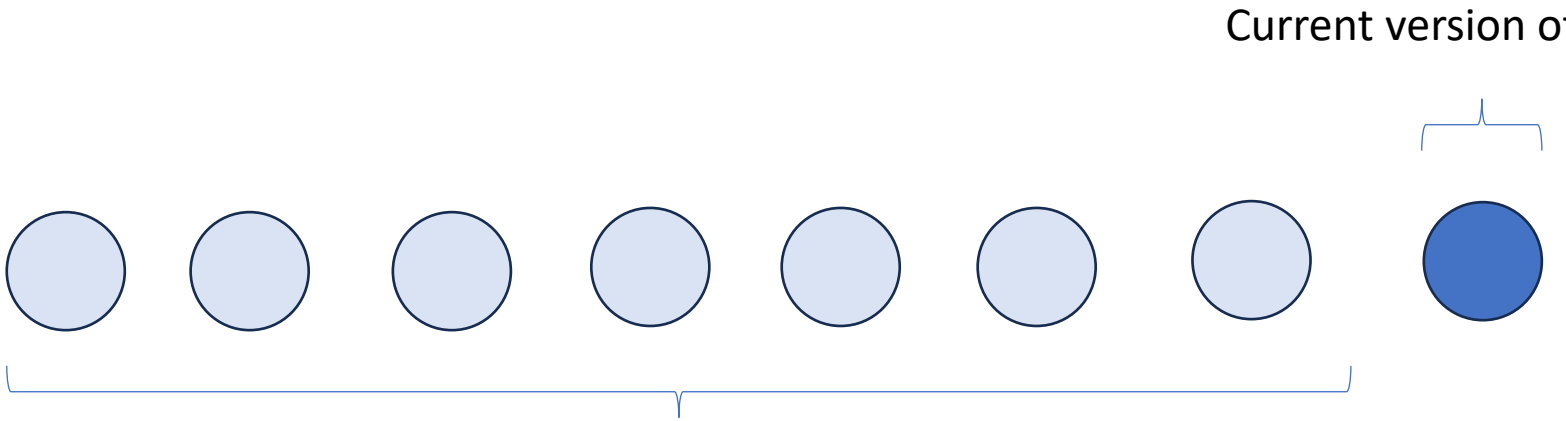
- Intro to Git and GitHub
- Deeper dive:
  - Git
  - Github
- Collaborating with Git and GitHub
- Forking around with branches
- Project management with GitHub
- Other cool stuff with GitHub

- Git and GitHub are two separate things
- Can be used separately and independently
- But most people use it together

Git

- Version control on your local computer
- Allows you to take snapshots of your code as it progresses/evolves
- Allows you to view the history of your code over time

Version control



GitHub

- GitHub is a web service that allows you to host Git
- You can see changes over time online
- Allows for collaboration
- GitHub is not the only option
  - Bitbucket
  - GitLab
  - Several others…

Why learn to use Git and GitHub

Why learn Git & GitHub

- Version control
  - Access at any time to previous versions
  - Peace of mind (can delete confidently)
- Backup
  - Dropbox, Google Drive etc for code

Repository on GitHub

RStudio project on local computer

Repository on GitHub

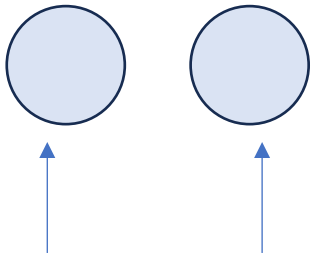
Synchronizes

RStudio project on local computer

Why learn Git & GitHub

- Version control
  - Access at any time to previous versions
  - Peace of mind (can delete confidently)
- Backup
  - Dropbox, Google Drive etc for code
- Collaboration
  - Supervisors
  - Colleagues/Clients
  - Using Dropbox/OneDrive and coding on the same file will create file conflicts

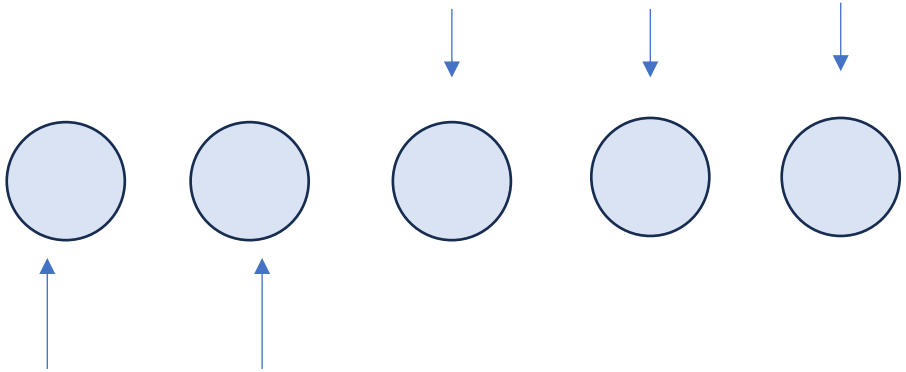
Collaboration



Haresh made changes

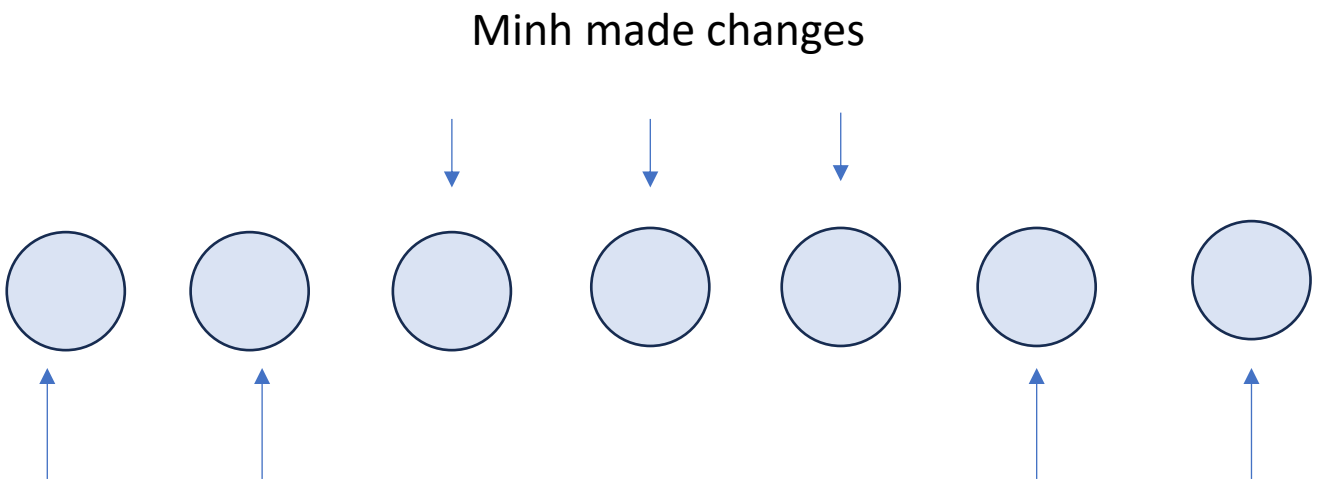
Collaboration

Minh made changes

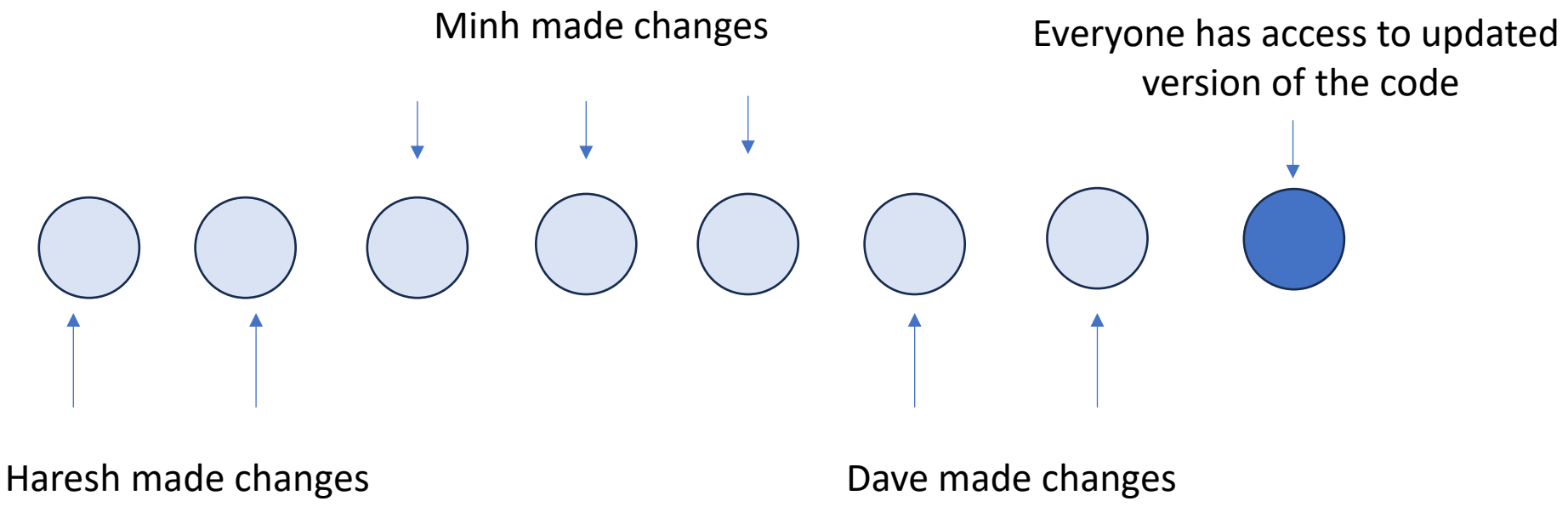


Haresh made changes

Collaboration



Collaboration



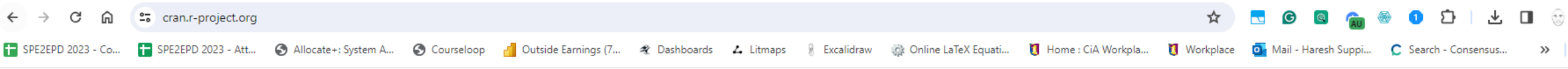
Before starting, we need to update everything

Update everything

- Most problems with Git and Github tend to be due to having older versions of
  - RStudio
  - R packages
  - R

```
version
```

```
> version
               _
platform       x86_64-w64-mingw32
arch           x86_64
os             mingw32
crt            ucrt
system         x86_64, mingw32
status
major          4
minor          2.1
year           2022
month          06
day            23
svn rev        82513
language       R
version.string R version 4.2.1 (2022-06-23 ucrt)
nickname       Funny-Looking Kid
>
```



CRAN
Mirrors
What's new?
Search
CRAN Team

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Task Views
Other

Documentation
Manuals
FAQs
Contributed

Donations
Donate

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux (Debian, Fedora/Redhat, Ubuntu)
- Download R for macOS
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2023-10-31, Eye Holes) R-4.3.2.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

Supporting CRAN

- CRAN operations, most importantly hosting, checking, distributing, and archiving of R add-on packages for various platforms, crucially rely on technical, emotional, and financial support by the R community.

  Please consider making financial contributions to the R Foundation for Statistical Computing.

Your turn

- Check versions of and update (if necessary)
  - R
  - Rstudio
  - Packages
    - Yes to install from sources that require compilation.

Installing Git

Installing Git

- Slight differences in OS steps
  - Demo on Windows
- Name of initial branch
  - ‘Master’ or ‘main’

git --fast-version-control

Search entire site...

About
Documentation
Reference
Book
Videos
External Links
Downloads
Community

This book is available in English.
Full translation available in
azərbaycan dili,
български език,
Deutsch,
Español,
Français,
Ελληνικά,
日本語,
한국어,
Nederlands,
Русский,
Slovenščina,
Tagalog,
Українська
简体中文,
Partial translations available in
Čeština,
Македонски,

Chapters ▾ 2nd Edition

1.5 Getting Started - Installing Git

Installing Git

Before you start using Git, you have to make it available on your computer. Even if it's already installed, it's probably a good idea to update to the latest version. You can either install it as a package or via another installer, or download the source code and compile it yourself.

Note	This book was written using Git version 2. Since Git is quite excellent at preserving backwards compatibility, any recent version should work just fine. Though most of the commands we use should work even in ancient versions of Git, some of them might not or might act slightly differently.

Installing on Linux

If you want to install the basic Git tools on Linux via a binary installer, you can generally do so through the package management tool that comes with your distribution. If you're on Fedora (or any closely-related RPM-based distribution, such as RHEL or CentOS), you can use `dnf`:

```
$ sudo dnf install git-all
```

If you're on a Debian-based distribution, such as Ubuntu, try `apt`:

```
$ sudo apt install git-all
```

For more options, there are instructions for installing on several different Unix distributions on the Git website, at https://git-scm.com/download/linux.

Installing on macOS

There are several ways to install Git on macOS. The easiest is probably to install the Xcode Command Line Tools. On Mavericks (10.9) or above you can do this simply by trying to run `git` from the Terminal the very first time.

```
$ git --version
```

If you don't have it installed already, it will prompt you to install it.

Čeština,
Македонски,
Polski,
Српски,
Ўзбекча,
繁體中文,
Translations started for
Беларуская,
فارسی,
Indonesian,
Italiano,
Bahasa Melayu,
Português (Brasil),
Português (Portugal),
Svenska,
Türkçe.

The source of this book is hosted on GitHub.
Patches, suggestions and comments are welcome.

$ git --version

If you don't have it installed already, it will prompt you to install it.

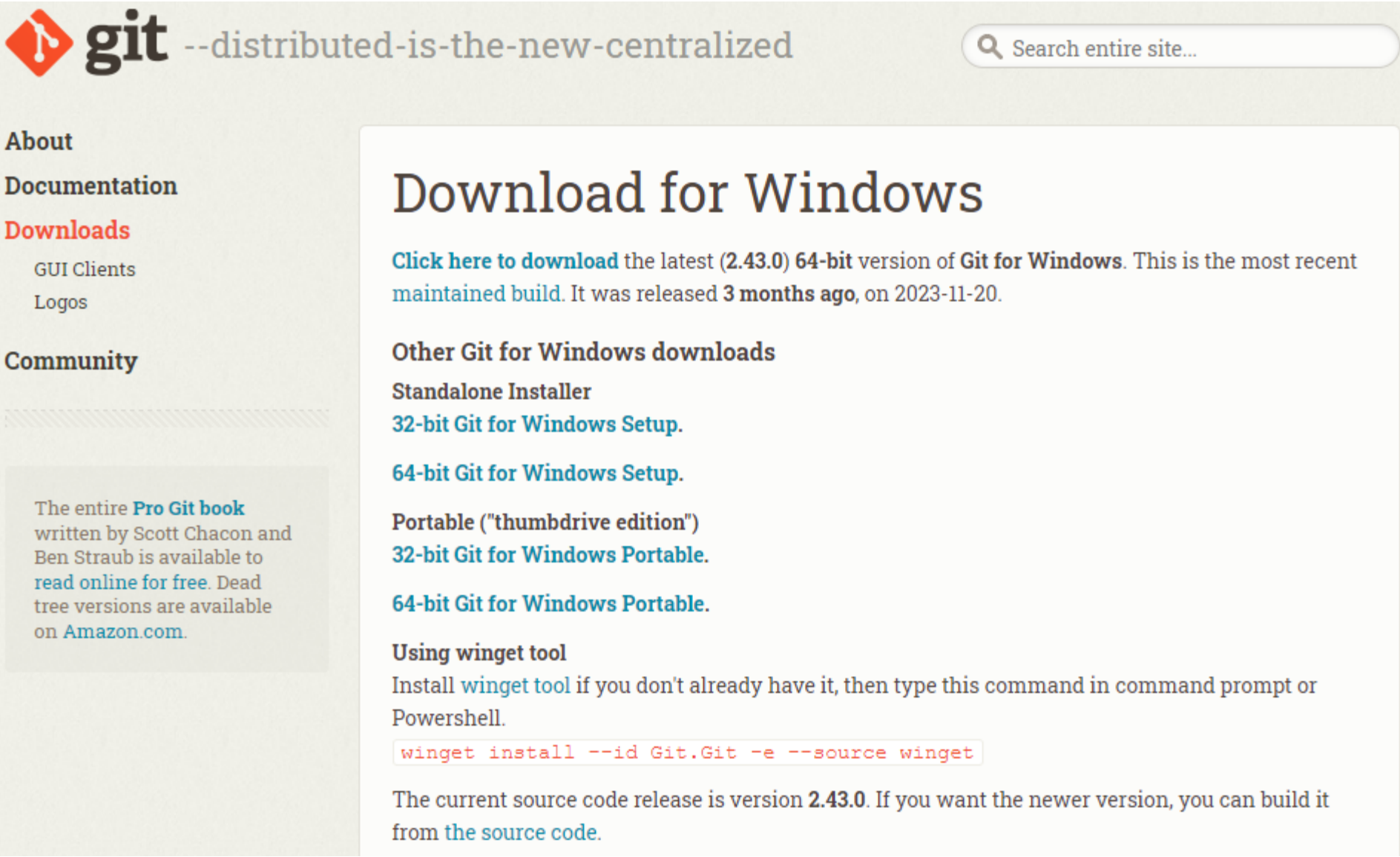
If you want a more up to date version, you can also install it via a binary installer. A macOS Git installer is maintained and available for download at the Git website, at https://git-scm.com/download/mac.

Figure 7. Git macOS installer

Installing on Windows

There are also a few ways to install Git on Windows. The most official build is available for download on the Git website. Just go to https://git-scm.com/download/win and the download will start automatically. Note that this is a project called Git for Windows, which is separate from Git itself; for more information on it, go to https://gitforwindows.org.

To get an automated installation you can use the Git Chocolatey package. Note that the Chocolatey package is community maintained.



Installing Git

- Slight differences in OS steps
  - Demo on Windows
- Name of initial branch
  - ‘Master’ or ‘main’
- Confirming you have Git installed properly
  - Go to RStudio terminal
  - ‘git --version’

Configuring Git

Configuring Git

- Need to tell Git who you are
  - usethis package

```
# Install the usethis package if it's not already installed
install.packages("usethis")

# Load the usethis library
library(usethis)

# Set user information for Git configuration
use_git_config(user.name = "Jane Doe", user.email = "yourgithub@email.com")
```

Your turn

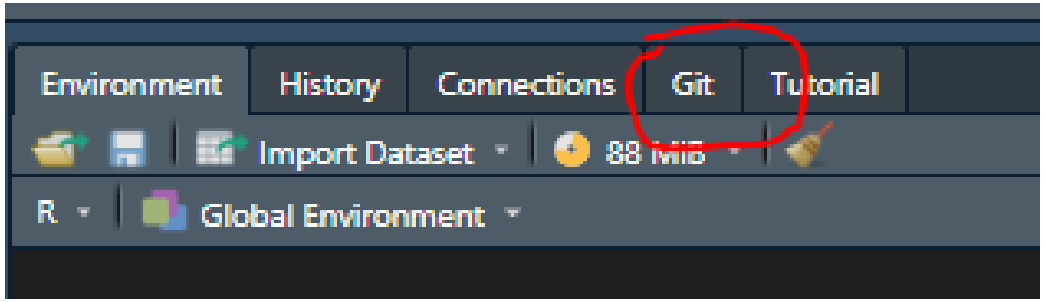
- Configure Git in Rstudio
  - You can either run `**git config --list**` in the Terminal to see if the username and email was saved in your gitconfig file or **usethis::git_sitrep()** in your Console

```
# Install the usethis package if it's not already installed
install.packages("usethis")

# Load the usethis library
library(usethis)

# Set user information for Git configuration
use_git_config(user.name = "Jane Doe", user.email = "yourgithub@email.com")
```

Creating a Local Git Repository

Creating a Local Git Repository

- New Project…
- New Project > New Directory
- Name the project (e.g. local-demo)
  - Ignore ‘Create git repository’ option for now
- Load usethis package in console
- Run use_git() function
- Select ‘Absolutely/Yes’ (or any manner of affirmative) to ‘*Is it ok to commit them?*’
  - Select yes to restart RStudio

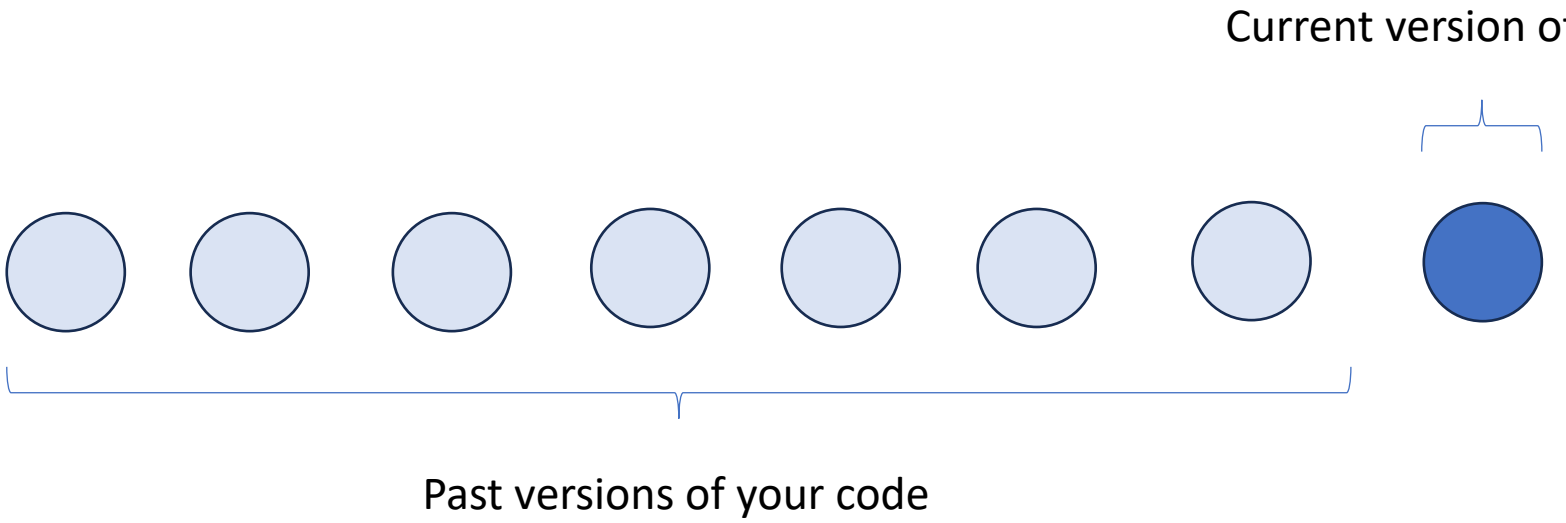


Your turn

- Configure new Rstudio project named local-demo
- Add a new Git repo to your project using the usethis package
- Ensure the *Git* tab appears on the top right panel

Commits

Commits are snapshots of your code

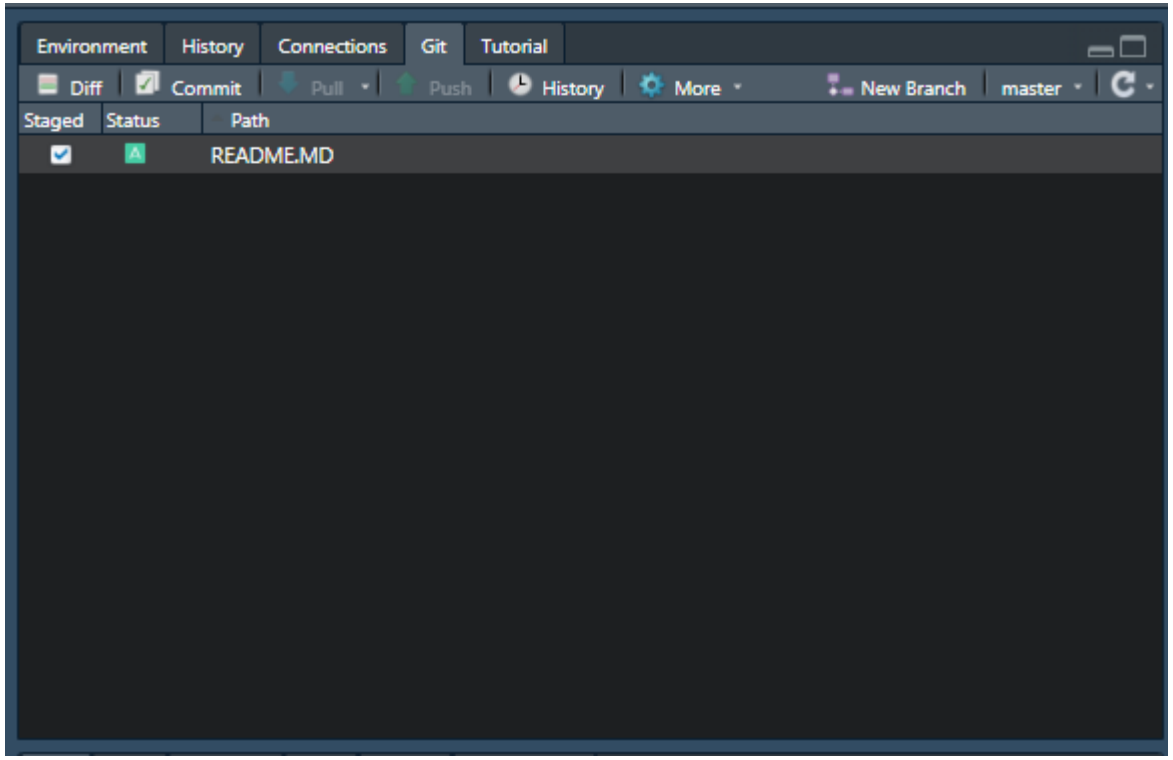


Commits have two main parts

- Part 1: Files you **stage** (i.e. want to include in that snapshot)
- Part 2: A **commit message** that describe what has changed since last commit

Staging files

- Which files do you want to include in this commit?
- We'll just include all for now



Commit messages

- What has changed since the last time we did a commit
  - A better way to describe your file rather than having file names such as
    - Final-V3-HS-DC-draft2.docx
- A good commit message tells you what changes were made (so take the effort to be concise yet descriptive)

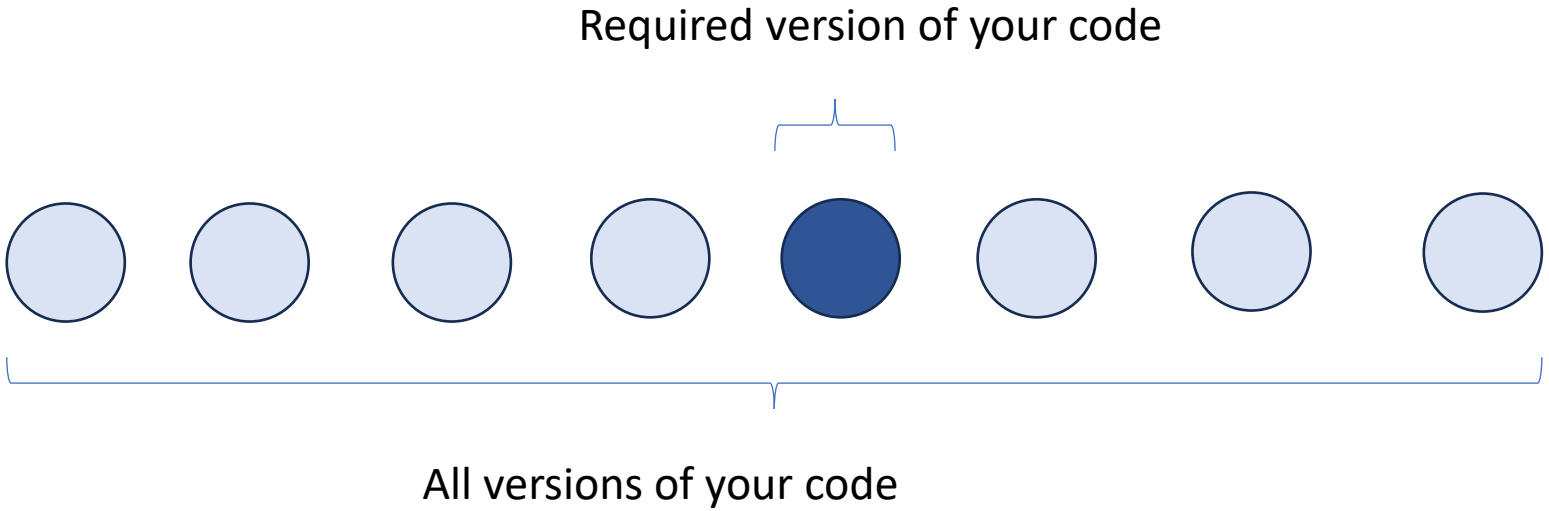
Your turn

- Create a new README.MD file in your project folder, add some text
- Go to the Git tab and hit commit
- Check the staged button next to your new README.MD file (and other possibly modified files)
- Add commit message
- Press *commit*

Commit history

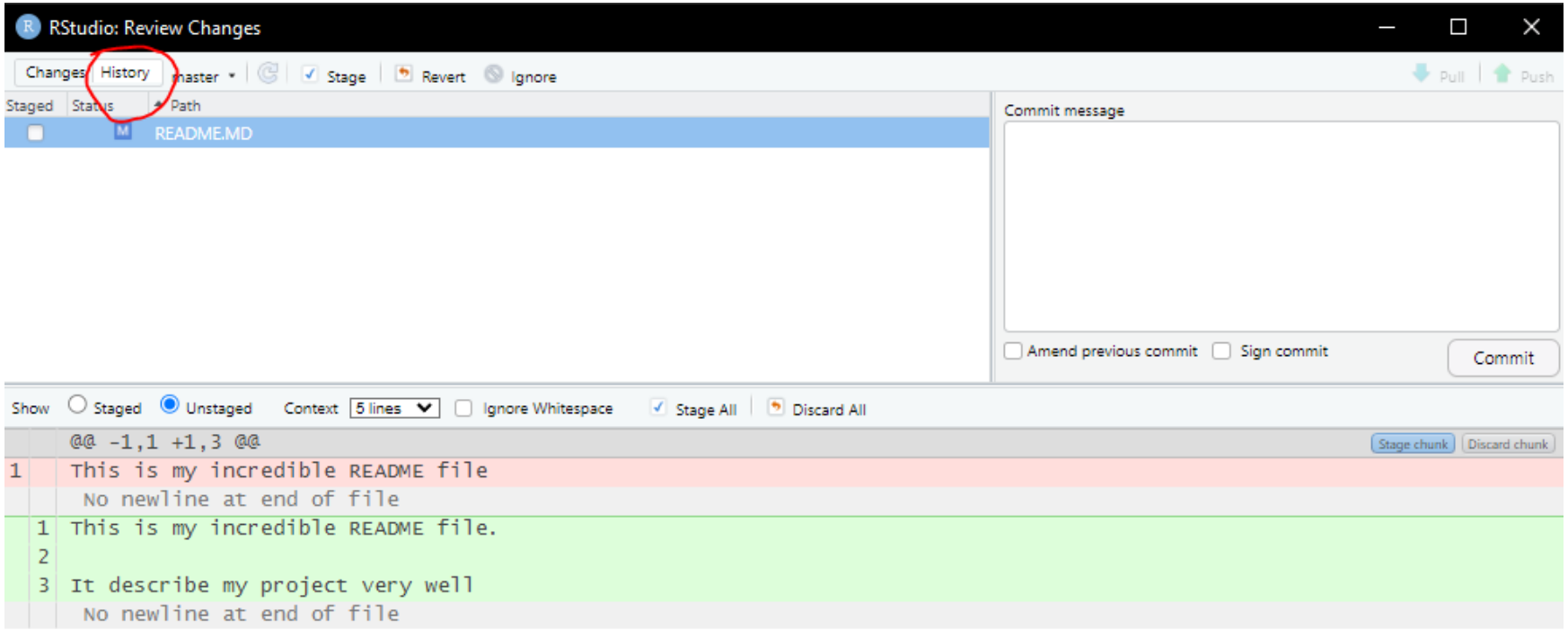
Commit history

- How can we see the history of our code…



Commit history

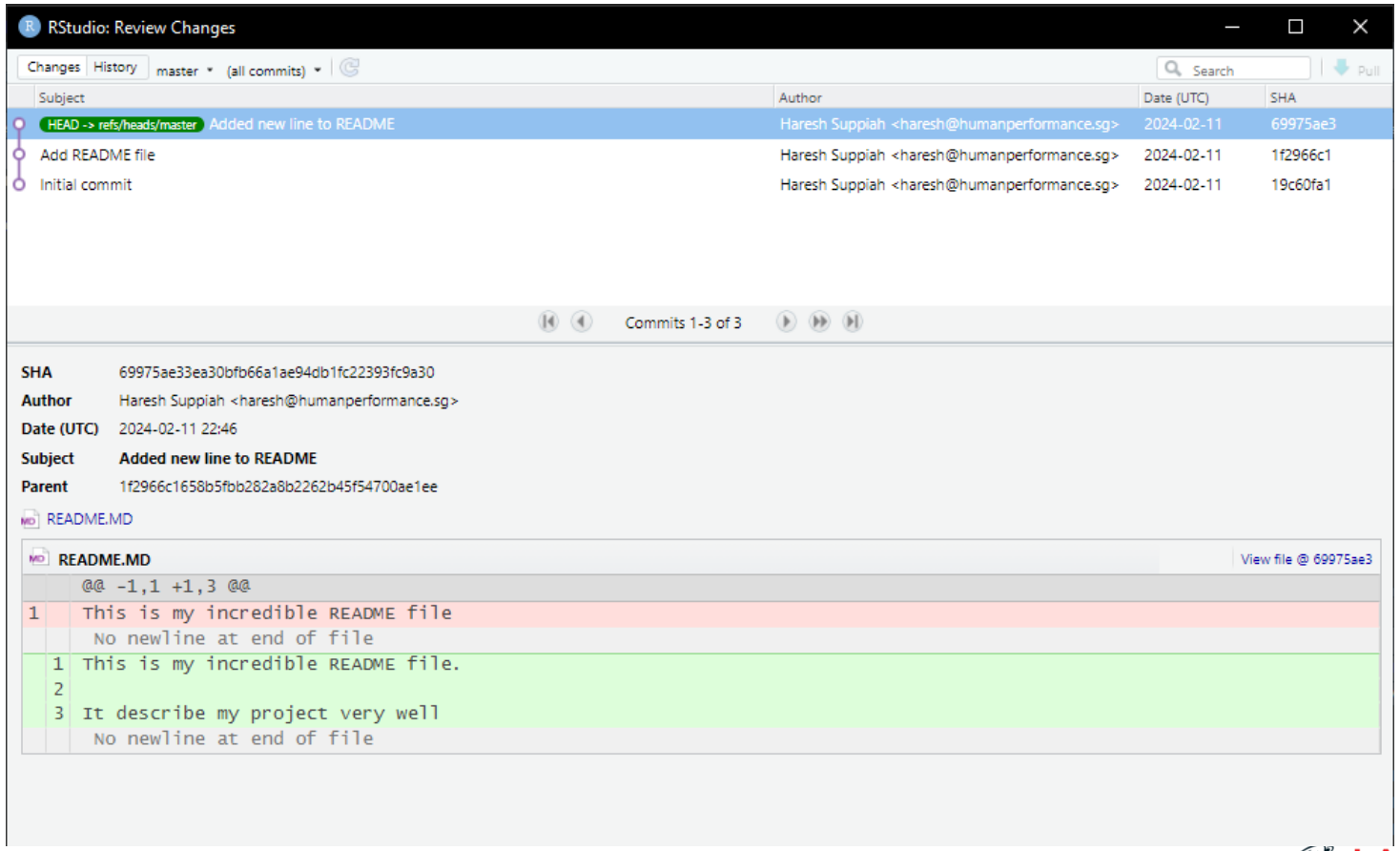
- How can we see the history of our code…



Commit history

- Changes over time

Commit history



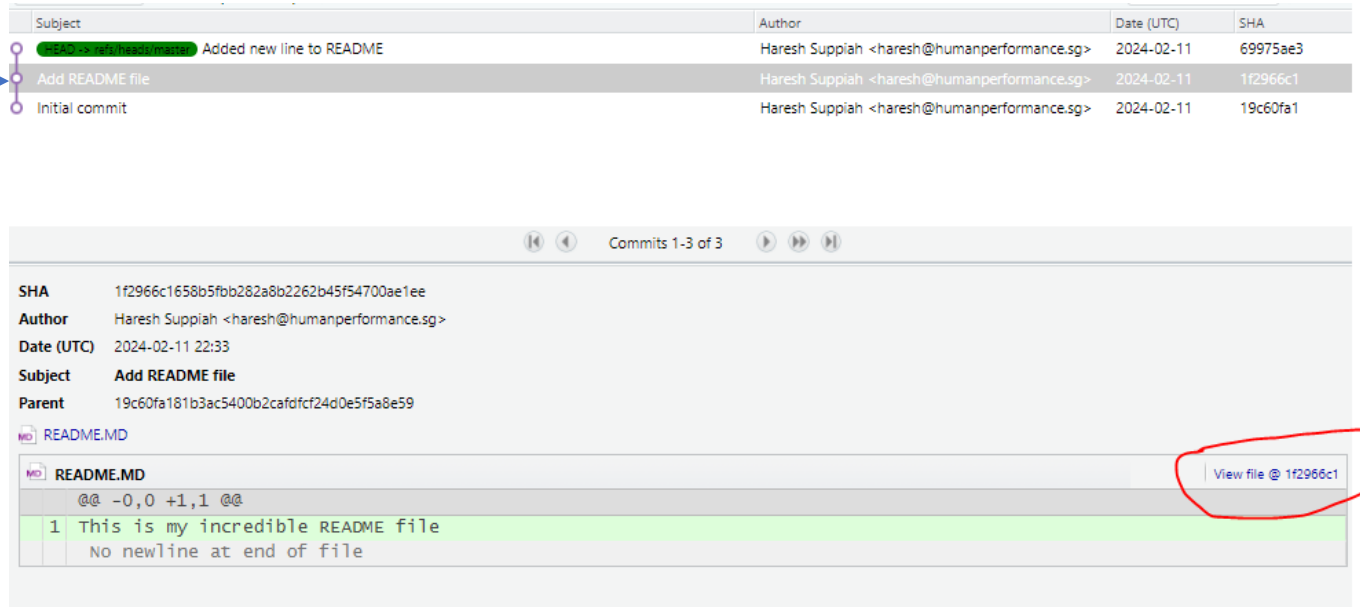
Deletions

New lines

Commit history

- How to see previous code/file

Select version



'View' file

Your turn

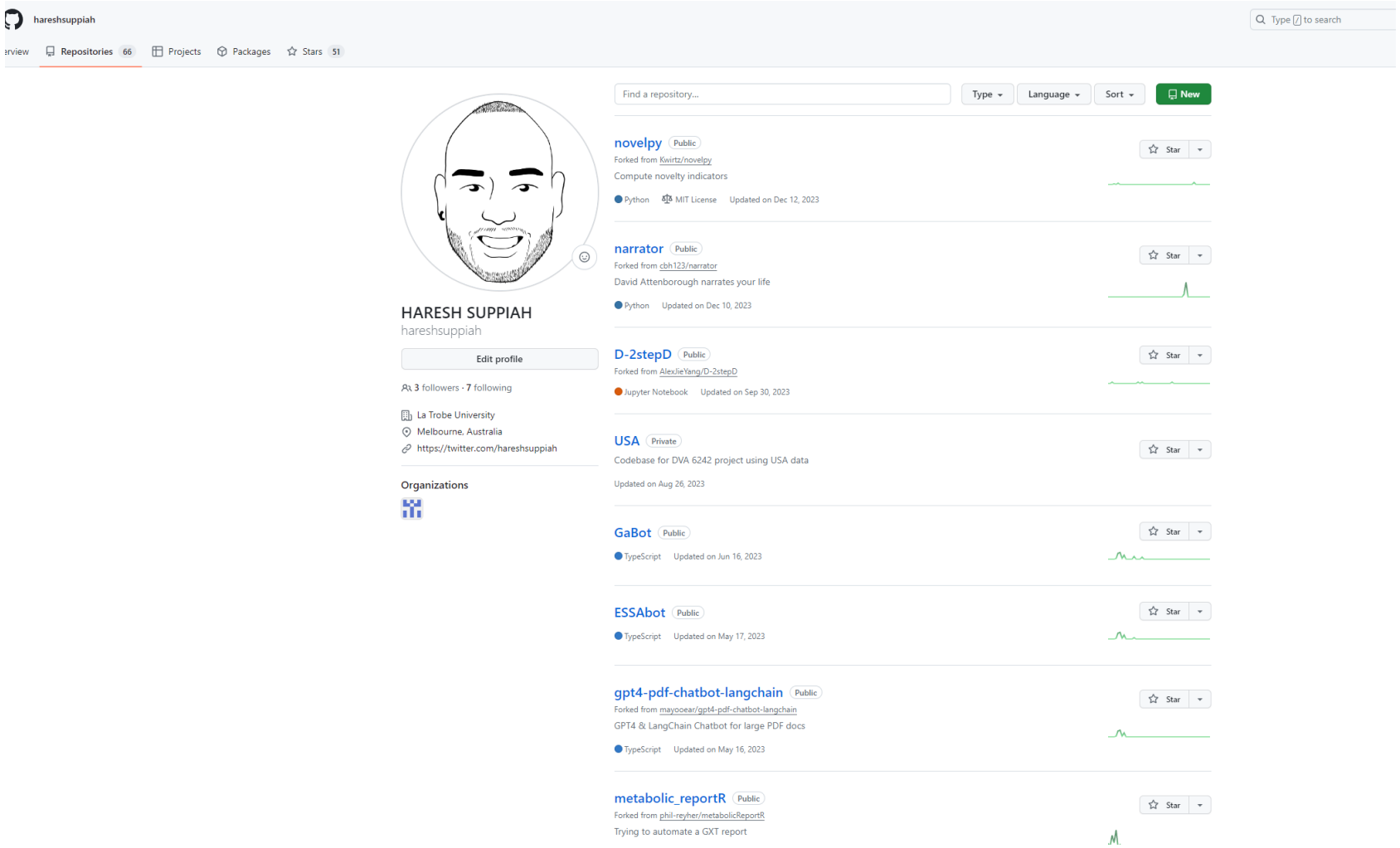
- Add some new text to your README.MD file
- Commit changes (ensure you stage and add a commit message)
- View history of your repo
- Ensure you understand the commit history (top section) and the diff (bottom section)
- View README.md at a previous commit by clicking ‘View file’

GitHub Repositories

GitHub Repositories

- AKA ‘repos’
- One repo for each project

GitHub Repositories



Previous Article Next Article

A Strategy to Inform Athlete Sleep Support From Questionnaire Data and Its Application in an Elite Athlete Cohort

in International Journal of Sports Physiology and Performance

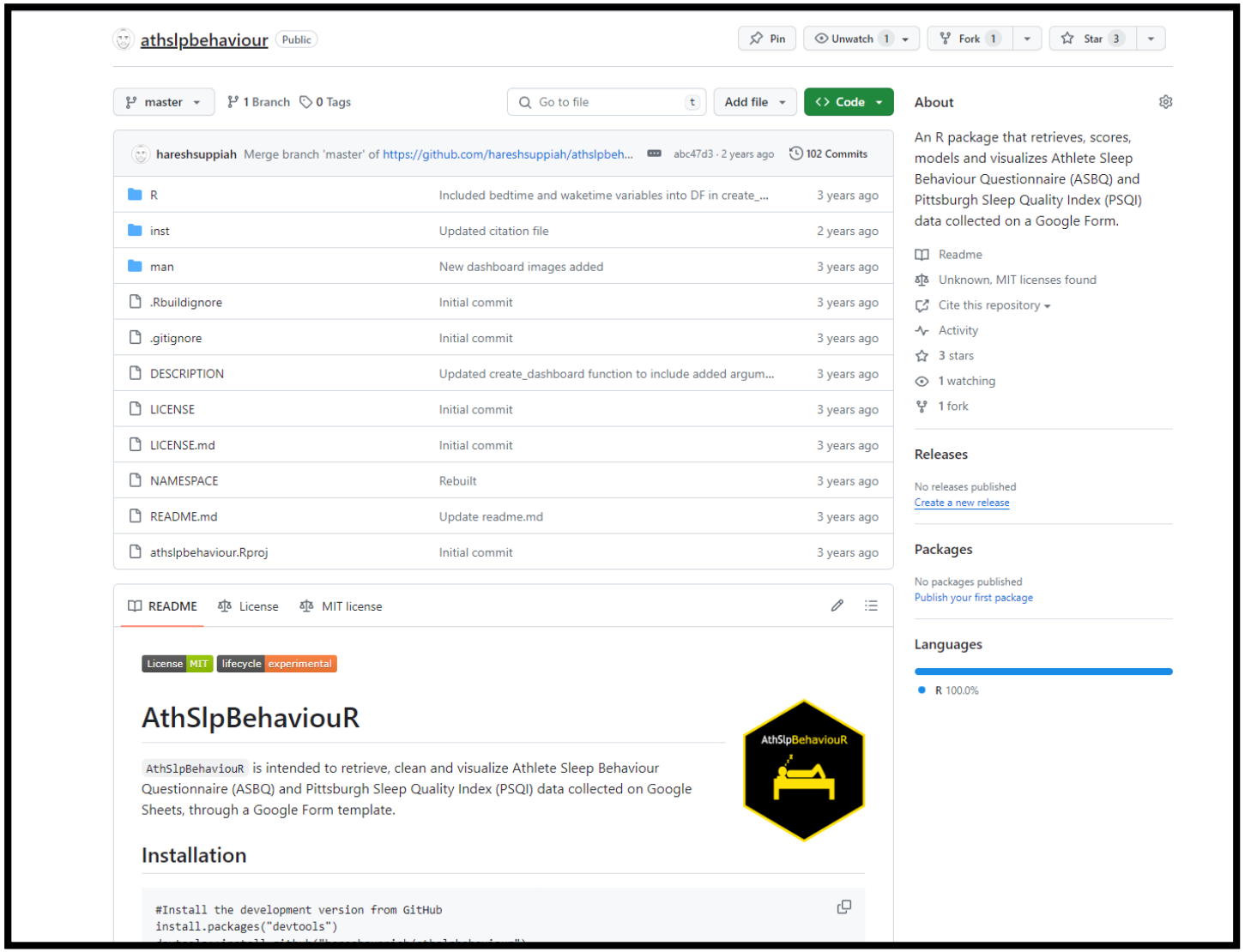
Click name to view affiliation

Haresh T. Suppiah *, Paul B. Gastin, and Matthew W. Driller

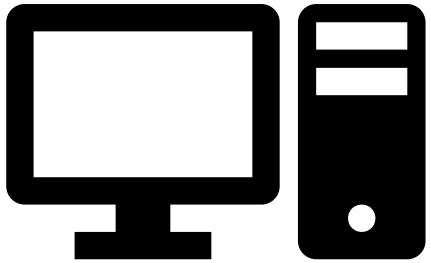
DOI: https://doi.org/10.1123/ijspp.2021-0561
Keywords: PSQI; ASBQ; clustering; R package; machine learning
First Published Online: 20 Jul 2022

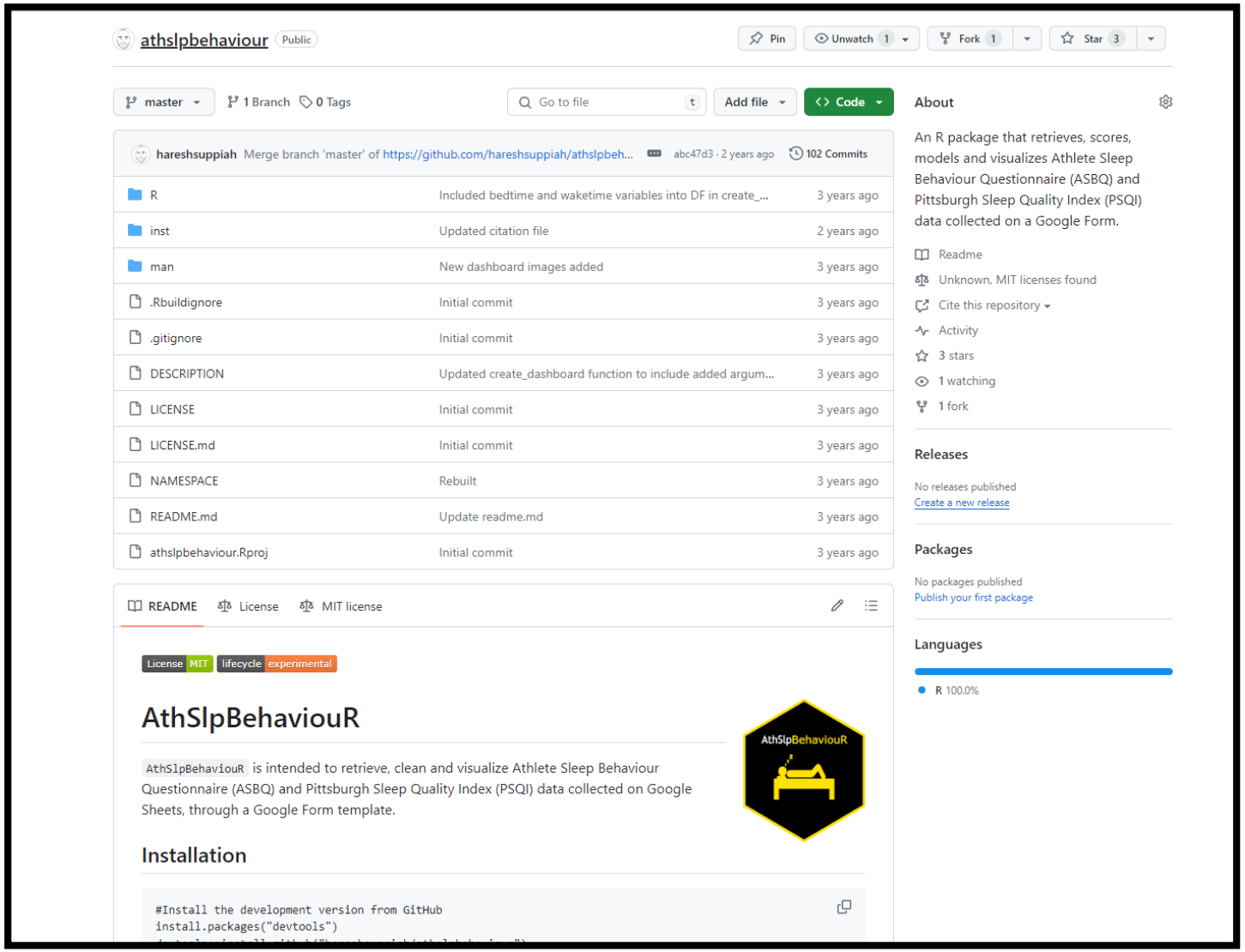
In Print: Volume 17: Issue 10 **Page Range:** 1532–1536

Abstract Author Notes Supplementary Materials



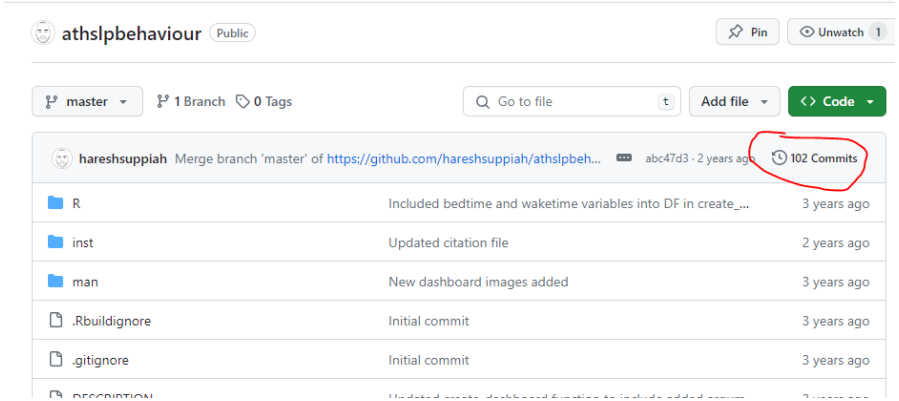
Synced locally

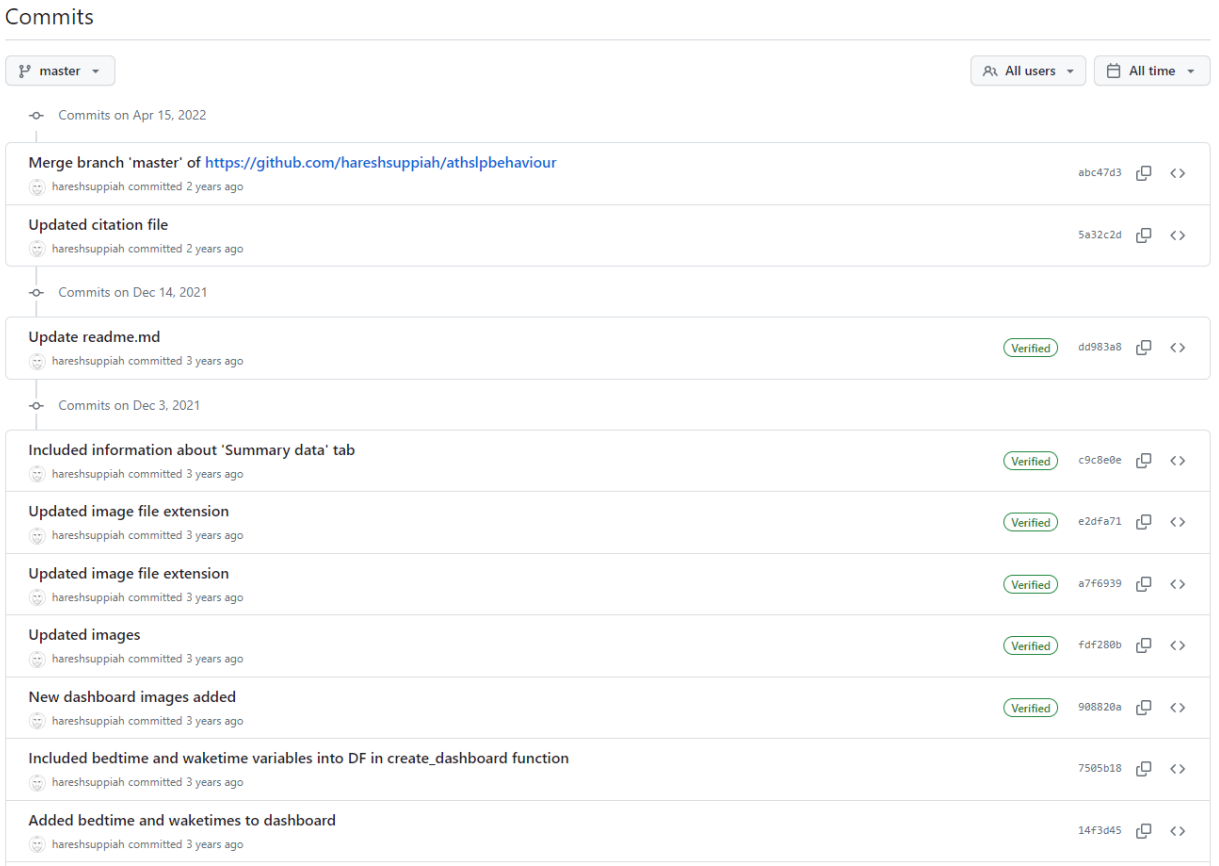




GitHub Repositories

- AKA ‘repos’
- One repo for each project
- Can see commit history

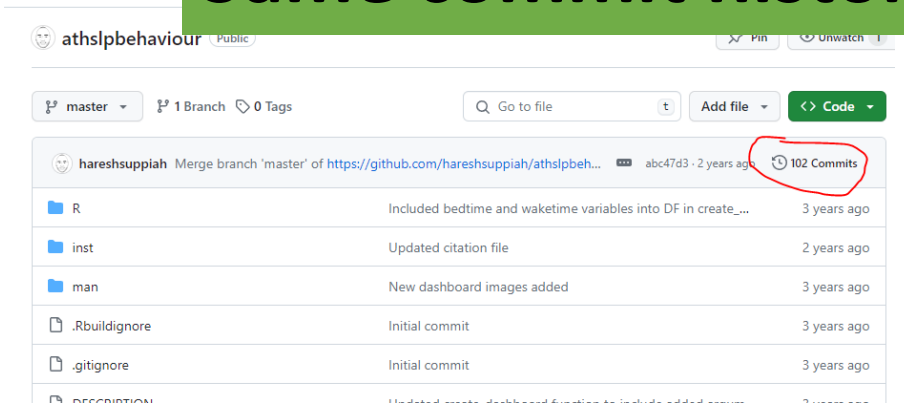


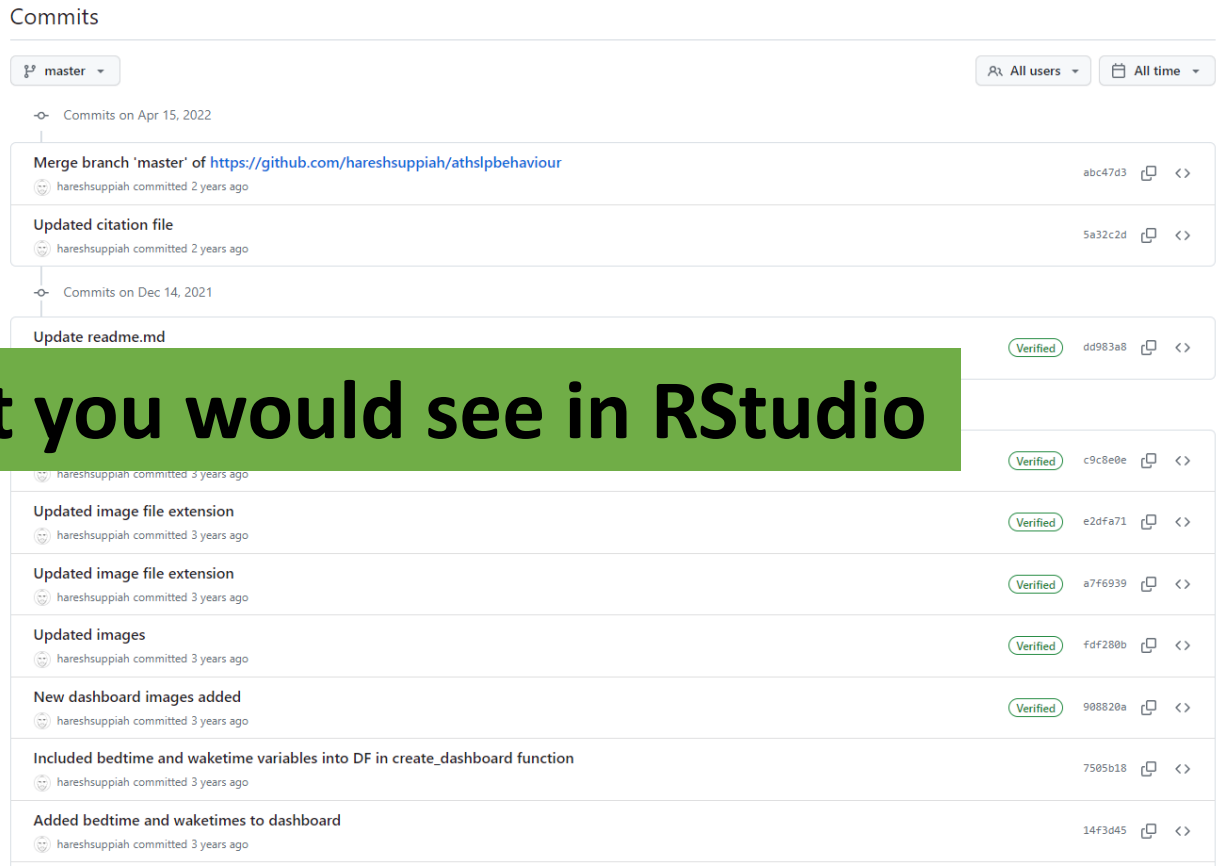


GitHub Repositories

- AKA ‘repos’
- One repo for each project
- Can see commit history

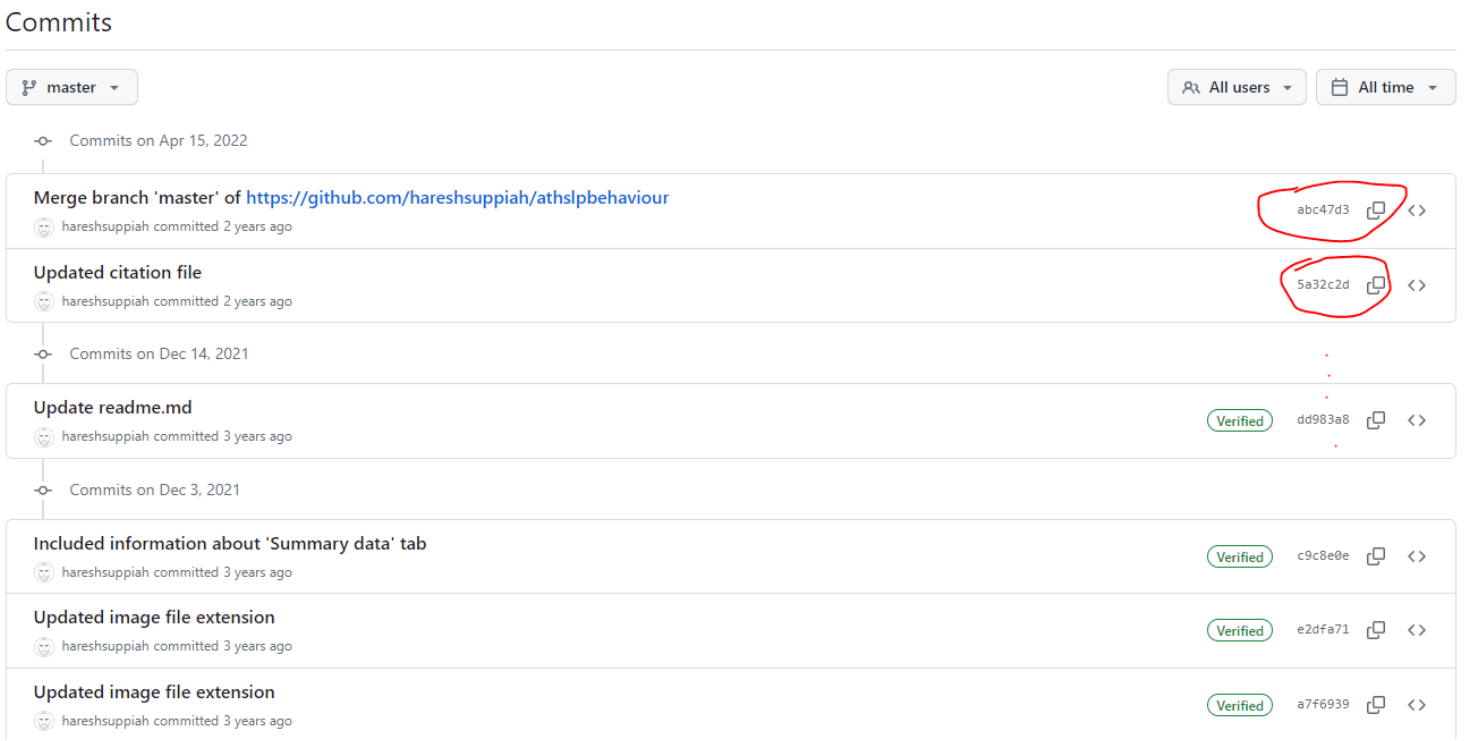
Same commit history that you would see in RStudio





GitHub Repositories

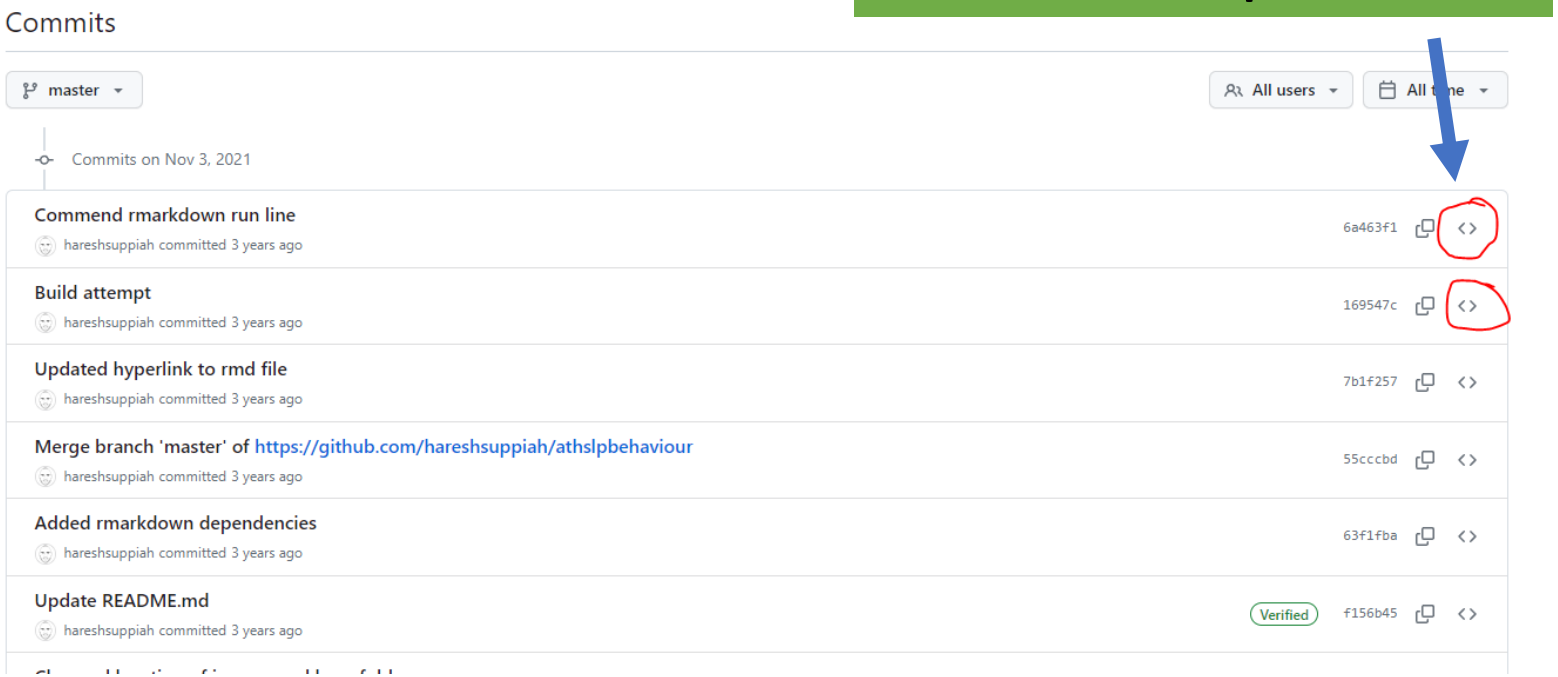
- AKA ‘repos’
- One repo for each project
- Can see commit history
- View ‘diffs’ in GitHub



GitHub Repositories

- AKA ‘repos’
- One repo for each project
- Can see commit history
- View ‘diffs’ in GitHub
- View past versions of code

See code at that point in time



Connecting RStudio and GitHub

Repository on GitHub

Synchronizes

RStudio project on local computer

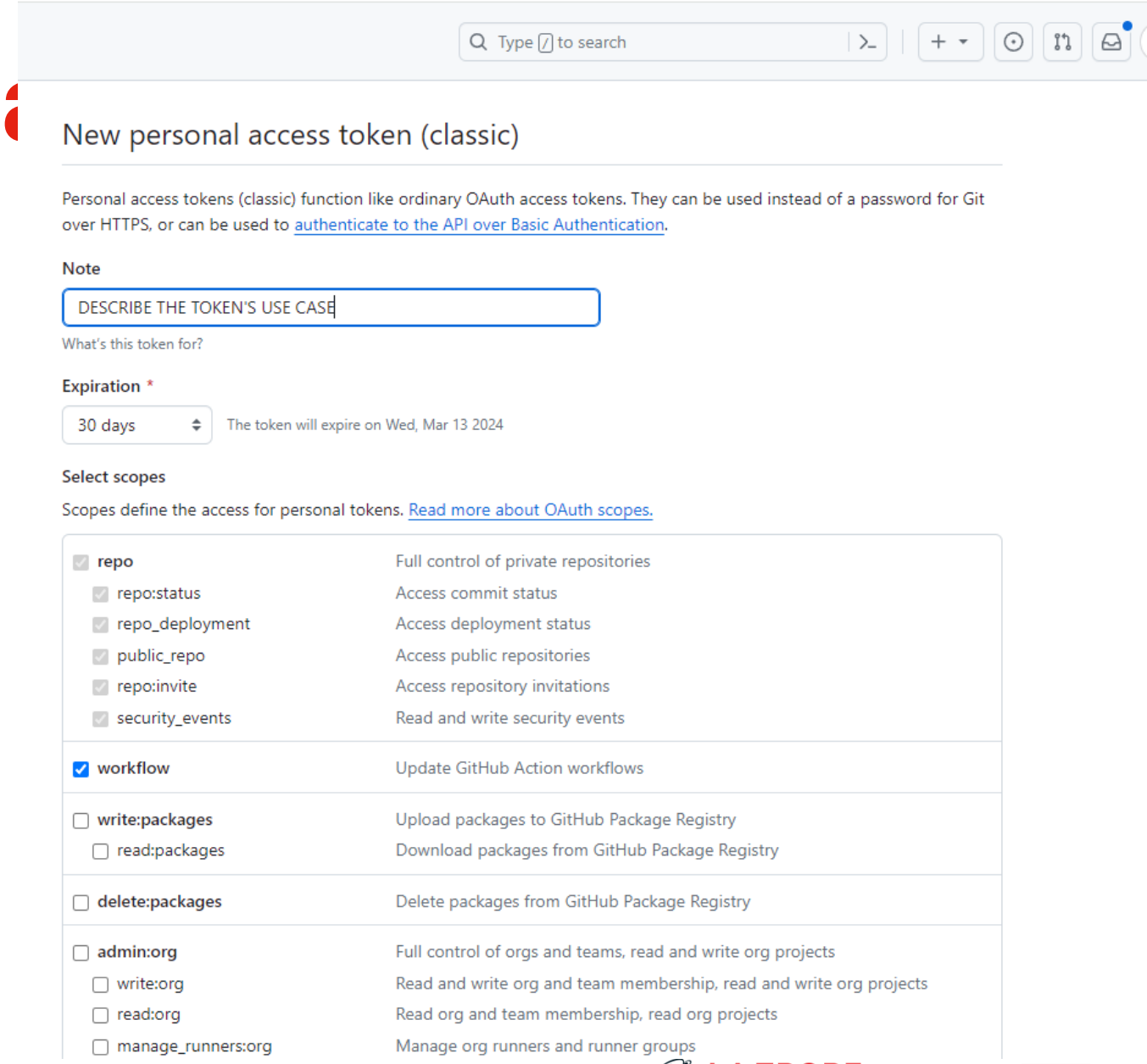
Connecting Rstudio and GitHub

- Create GitHub account
- Create Personal Access Token (PAT)
  - Authentication

```
library(usethis)
create_github_token()
```

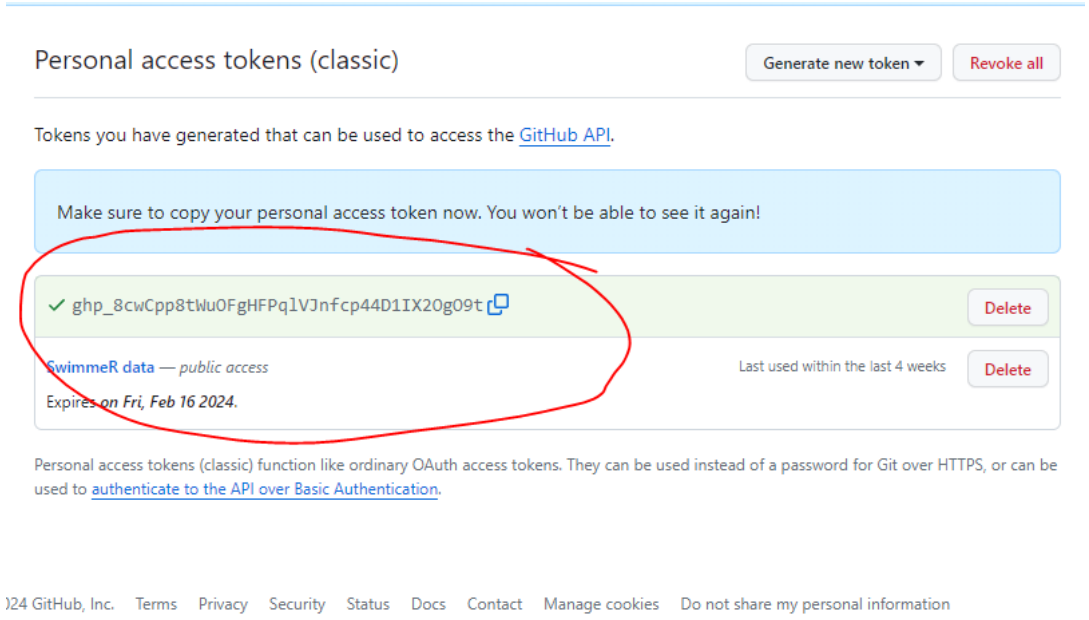
Connecting Rstudio a

- Create GitHub account
- Create Personal Access Token (PAT)
  - Authentication
  - Name it whatever you want
  - Set an expiration if you want
  - Default setting for all options
    - create_github_token() function sets it up for us automatically



Connecting Rstudio and GitHub

- Create GitHub account
- Create Personal Access Token (PAT)
  - Authentication
  - Name it whatever you want
  - Set an expiration if you want
  - Default setting for all options
    - create_github_token() function sets it up for us automatically
  - *Do not share token info*
  - Copy token info
    - *You won't get to see it again*



Connecting Rstudio and GitHub

- Use PAT to connect RStudio and GH
- Use *gitcreds* package

```
install.packages("gitcreds") # If not installed
library(gitcreds)
gitcreds_set()
```

Paste your PAT into console

Your turn

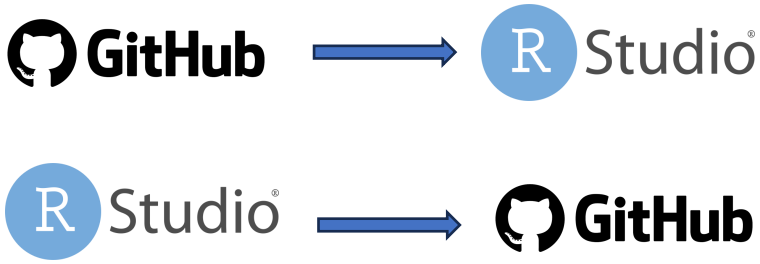
- Create GitHub account
- Create Personal Access Token
- Register PAT using the gitcreds package. To check if it works, run”\
  - library("usethis")
  - git_sitrep()

```
install.packages("gitcreds") # If not installed
library(gitcreds)
gitcreds_set()
```

Pushing an RStudio Project to a GitHub Repo

Pushing an RStudio Project to a GitHub Repo

- Two options to keep RStudio (local) project in sync with GitHub Repo
  - Create GitHub repo (on GH website) then sync with Rstudio
  - Create project in Rstudio, initialize on Git, push to GitHub



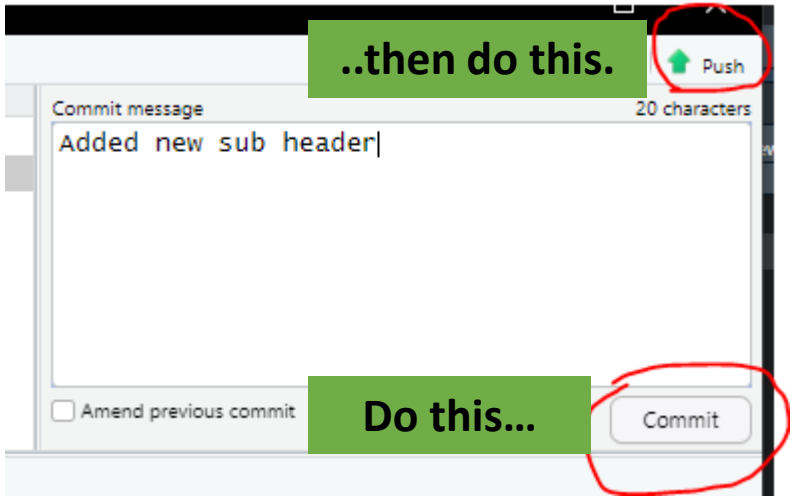
Method 1: RStudio → GitHub

- Run the use_github() function in the project console

```
use_github()
```

Pushing edits to GitHub

- After making edits
  - Save
  - Stage
  - Commit
  - Commit message
  - **'Push' button will automatically update on GH repo [NEW STEP]**



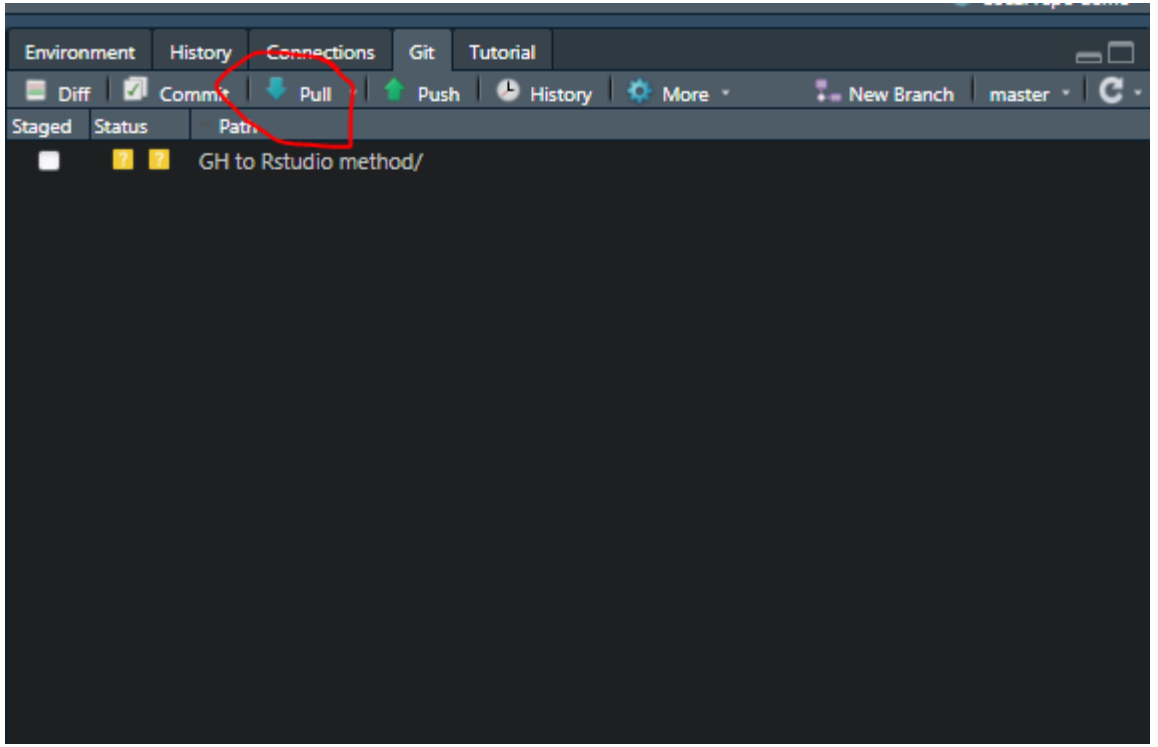
Your turn

- Using the project you created previously
- Push your project to GitHub using the use_github() function (Method 1)

Pulling from a GitHub Repo

Pulling from a GitHub Repo

- Assuming there are edits made on your GitHub project Repo
  - Maybe another collaborator has made edits



Your turn

- Make some edits on GH repo
- Pull changes to local RStudio instance

Keeping Rstudio and GitHub in Sync

Keeping Rstudio and GitHub in Sync

1. Work/Code in Rstudio
2. Any time you make significant changes (whenever you would usually save), make a **commit**
3. **Push** your work to GitHub

Your turn

- Make edits to either of your github projects (in Rstudio)
- Stage this and add a commit message
- Commit
- Push to GitHub
- Check that edits are made in GitHub repo

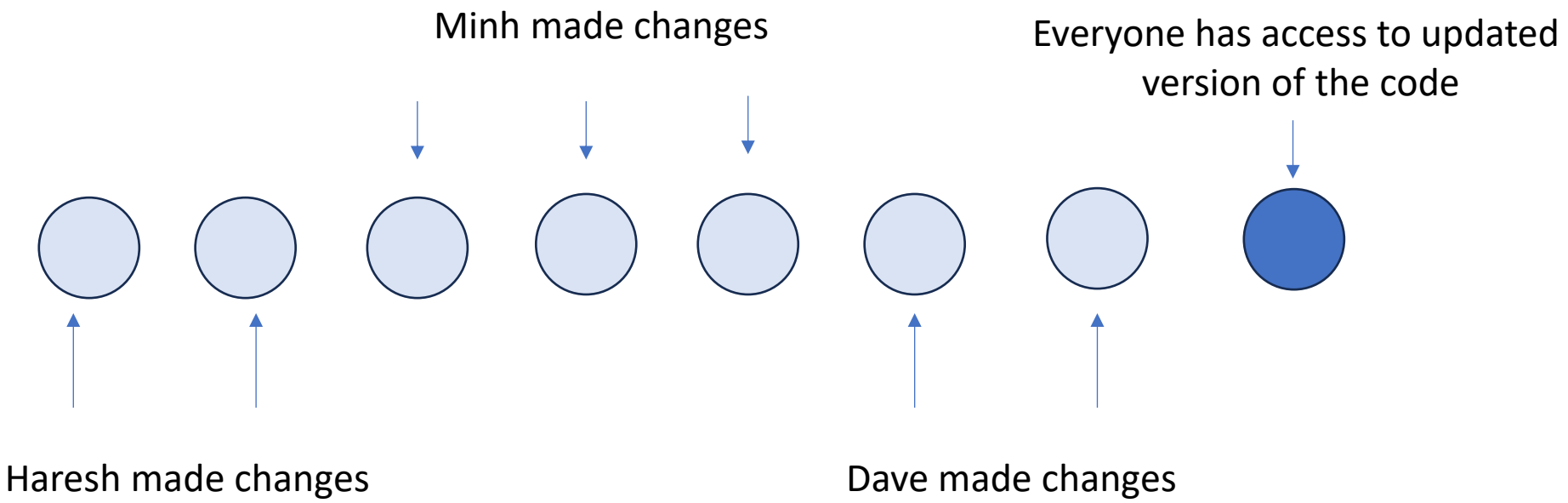
- ✓Set up Git
- ✓Set up GitHub
- ✓Created repo (both methods)
- ✓Pushed to GitHub
- ✓Pulled from GitHub
- ✓Adopted new workflow

Things to note about using this new workflow

Things to note about using this new workflow

1. Collaboration
   - Multiple people can technically work on the same code and avoid conflicts (compared to other cloud-sharing services (e.g. Dropbox, OneDrive, etc.)

Collaboration

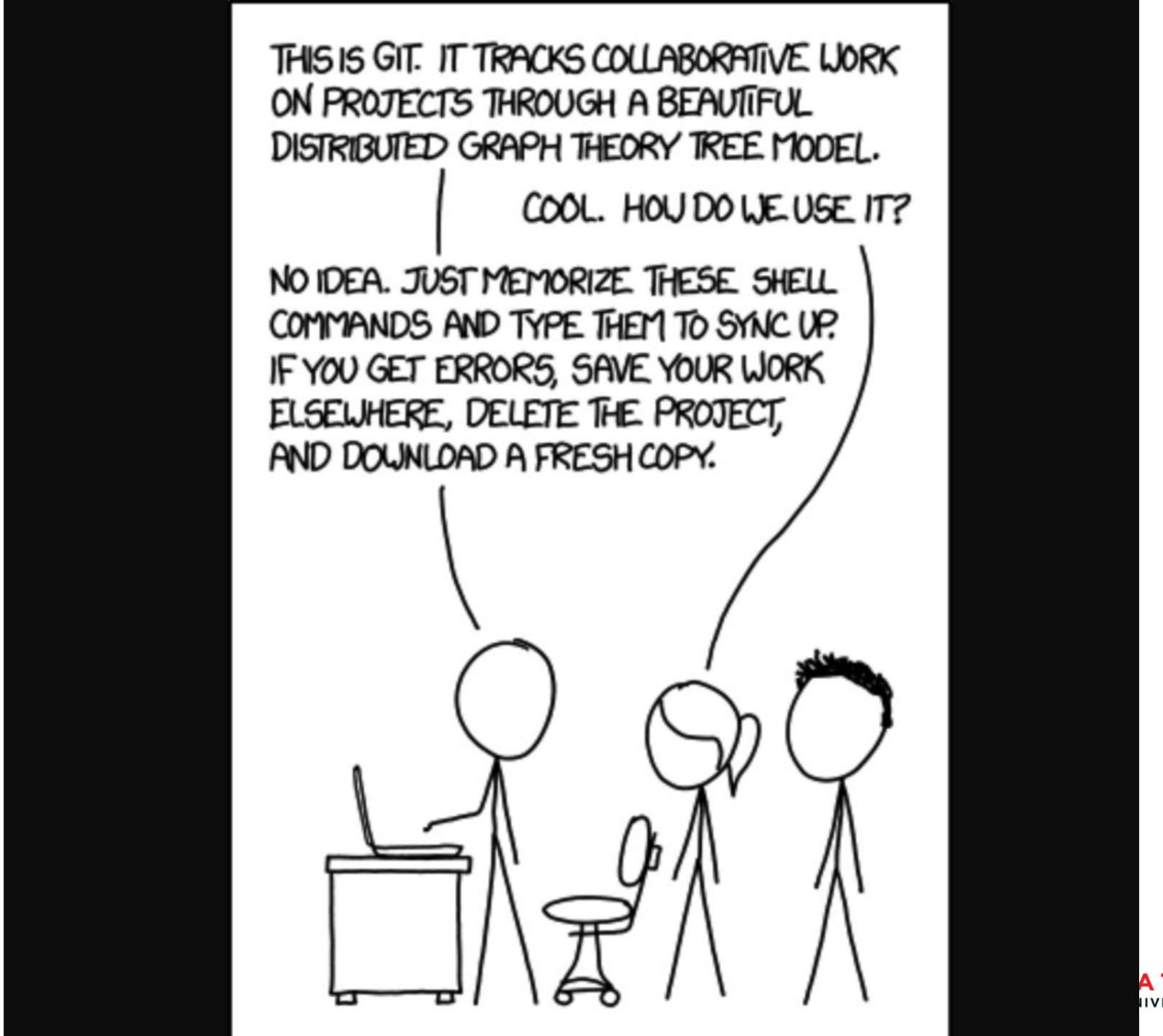


Why we're using this new process

1. Collaboration
   - Multiple people can technically work on the same code and avoid conflicts (compared to other cloud-sharing services (e.g. Dropbox, OneDrive, etc.)
2. Things can go wrong with working with GitHub
   1. Can't pull from GitHub
   2. Merge conflicts
   3. Can't Push to GitHub

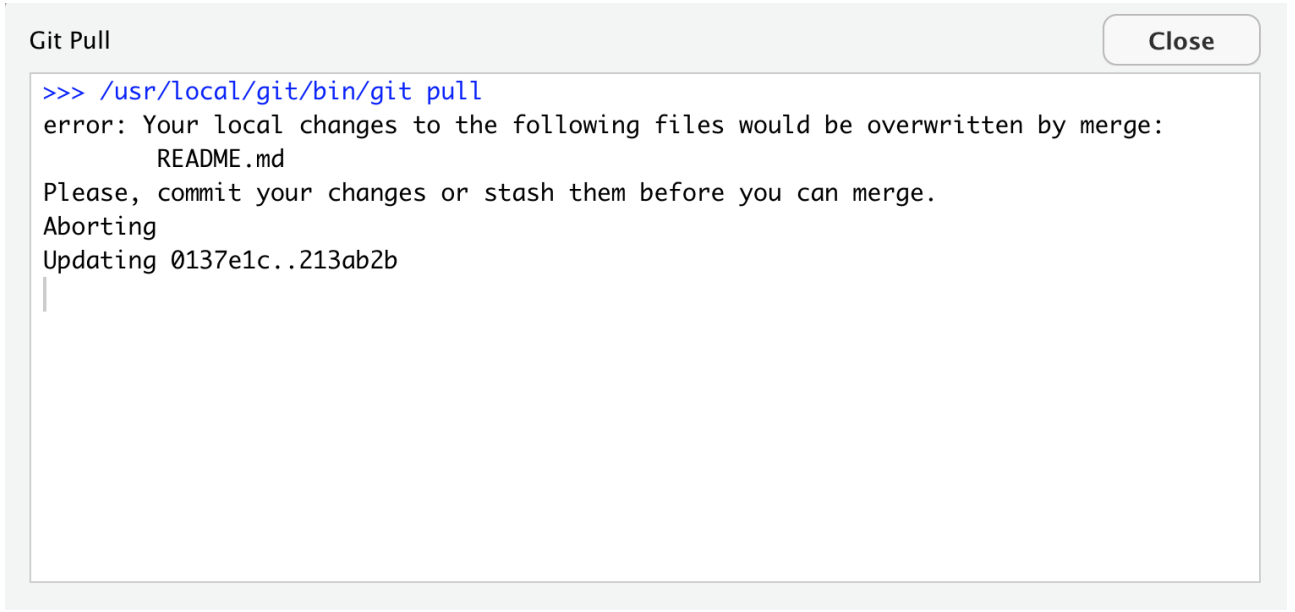
Why we're using this new process

1. Collaboration
   - Multiple people can technically work on the same code and avoid conflicts (compared to other cloud-sharing services (e.g. Dropbox, OneDrive, etc.)
2. Things can go wrong with working with GitHub
   1. Can't pull from GitHub
   2. Merge conflicts
   3. Can't Push to GitHub



Can’t pull from GitHub

1. Changes made locally that has not been committed
2. Git will not allow code to be pulled from GH until you commit
3. Generally solved by committing on your local machine before pulling*
   - *Tends to work when anyone else working on the code is not editing the same line of code as you

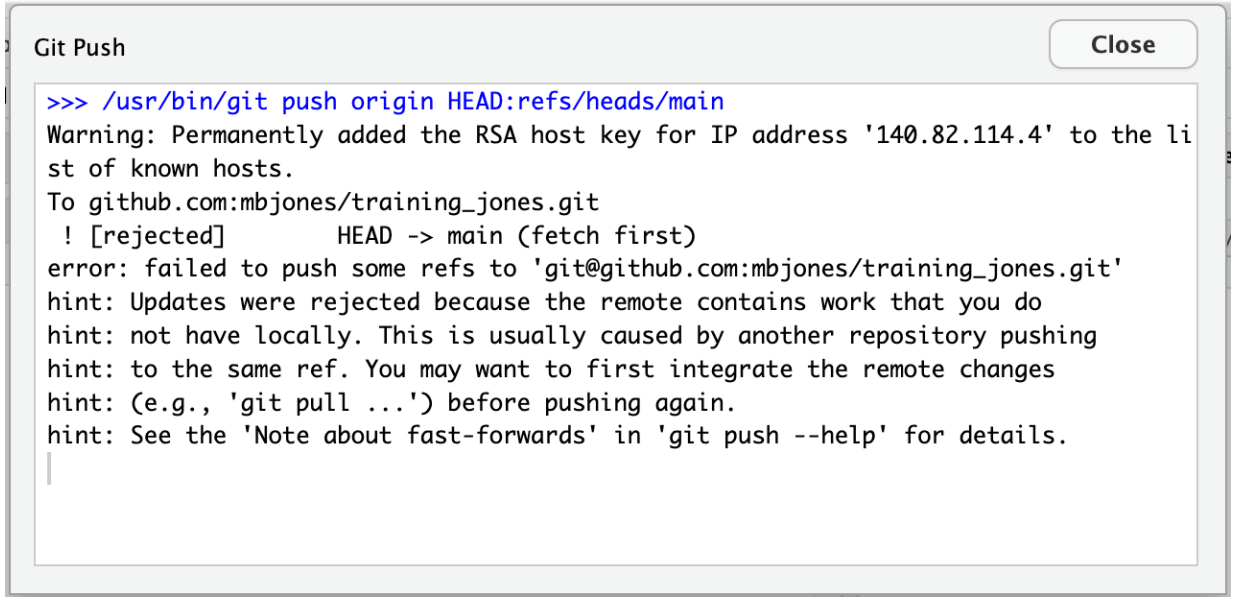


Merge conflicts

- Happens when working on the same line of code as someone else
- You are working on both GH and Rstudio versions of your project and you do not keep the versions updated
- Generally this is how you deal with it:
  - Decide which pieces of local/remote code you want to keep
  - Remove code you don't want to keep
  - Remove any conflict markers and divider
  - Commit code
  - Push code

Can’t push to GitHub

1. Usually there are changes in code on GitHub that you have not incorporated into local code
2. Have to bring those changes into the project
3. Dealing with it:
   - Pull code from GH
   - Deal with merge conflicts (if any)
   - Push

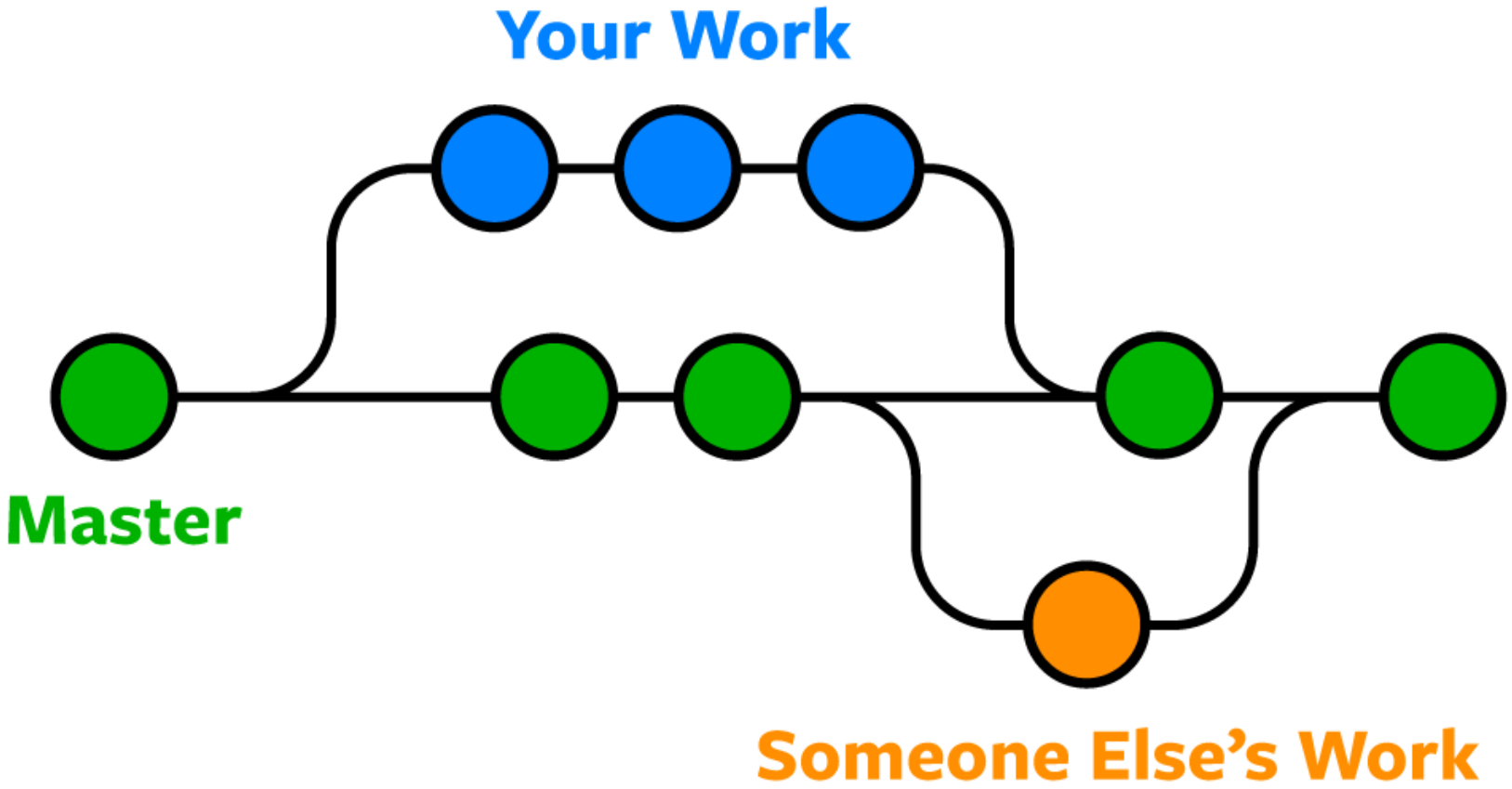


Trying to avoid issues (some good practices)

1. Always pull from GH before pushing
2. Commit regularly
3. Use branches
   - I'll cover this briefly

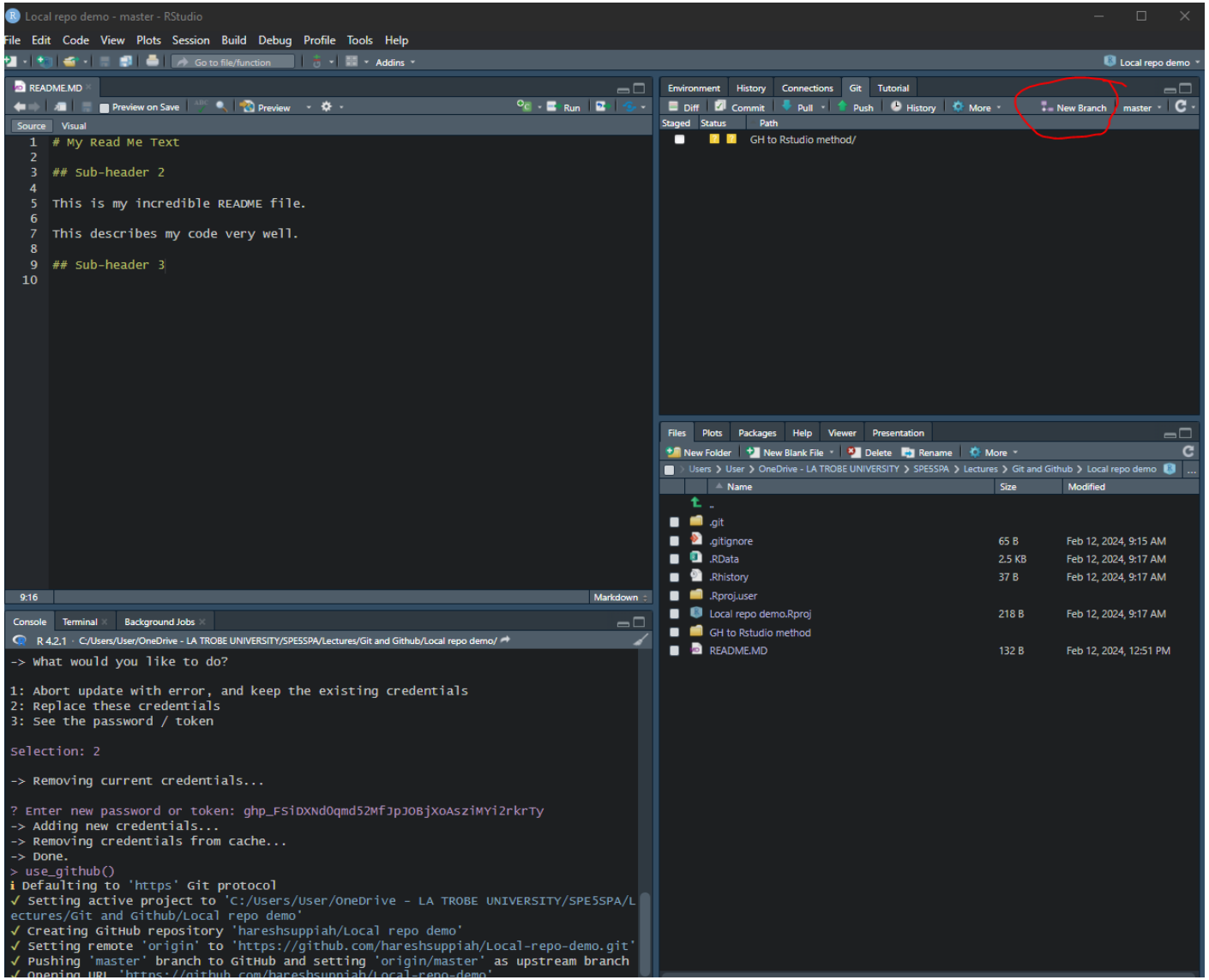
Branches

Branches



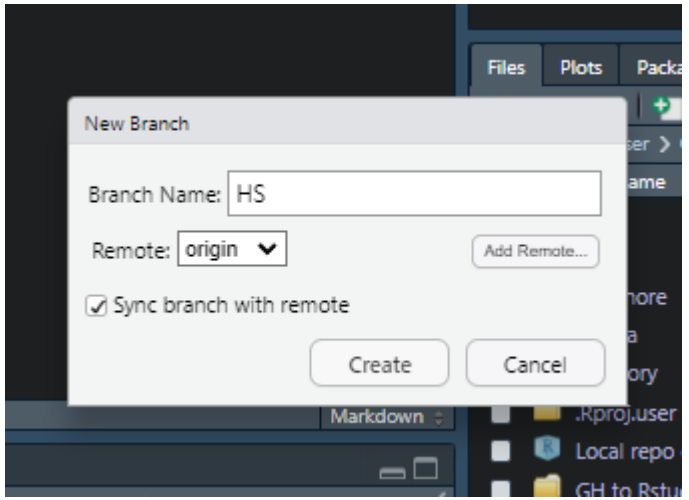
https://www.nobledesktop.com/learn/git/git-branches

How to create branches



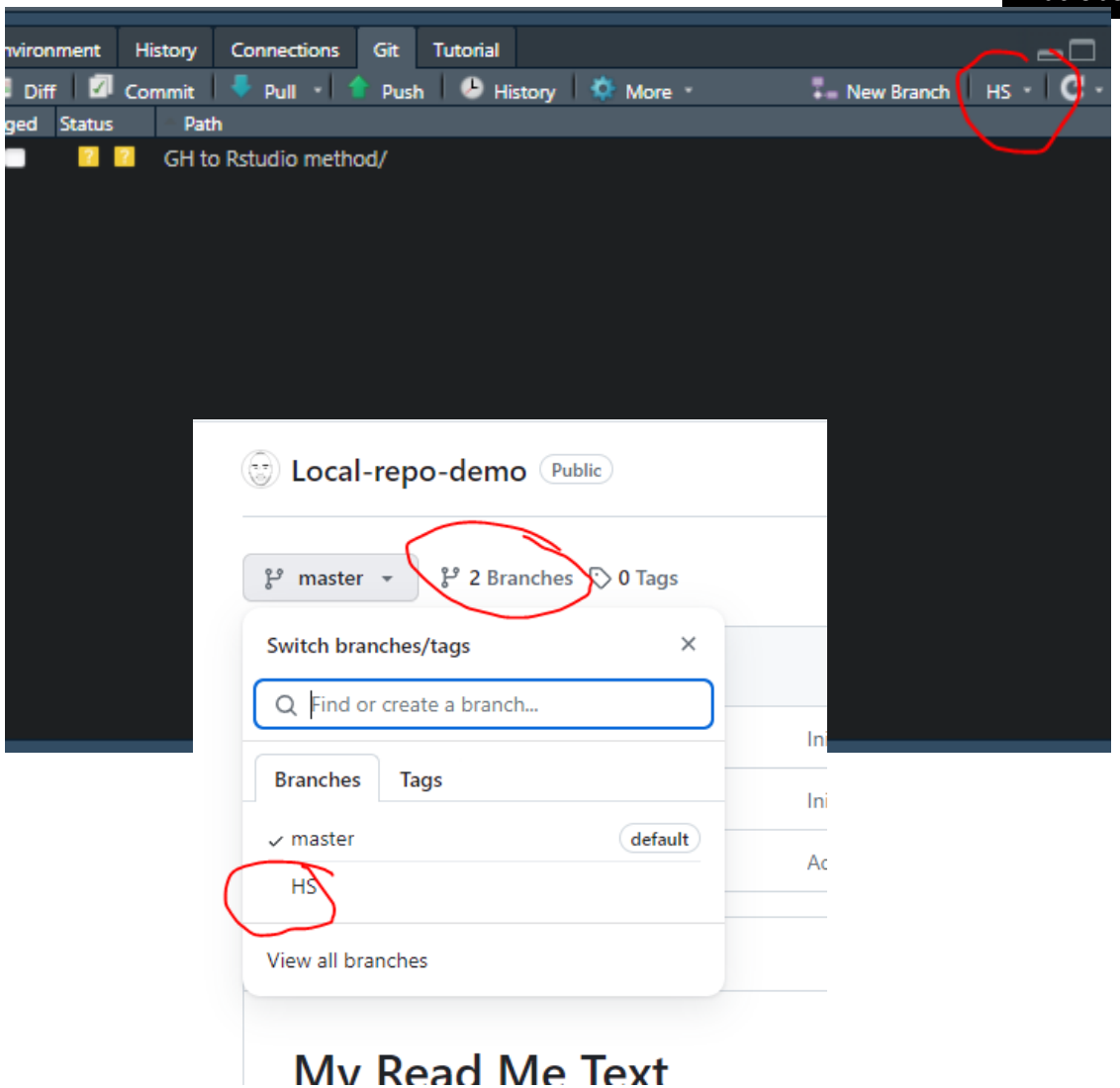
How to create branches

- Ensure ‘Sync branch with remote is checked’



How to create branches

- Only influences the branch reflected on local instance
- You can make edits and toggle between branches to see differences

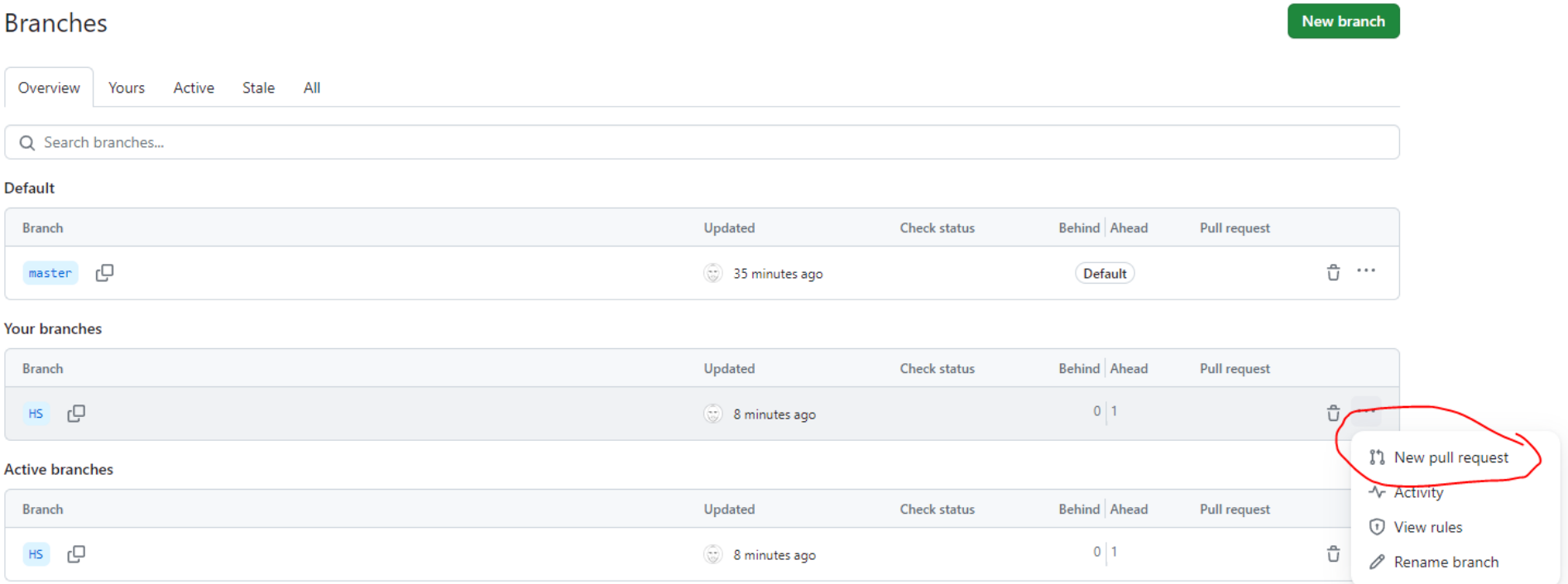


Your turn

- Create a new branch
- Edit README.md file, commit changes, push to GH
- Go to repo on GH and see if you can see changes
- Toggle between branches on local instance to see differences

Using pull requests to merge a branch into main

- Eventually, you might want to make a sub-branch a main/master branch
- Cannot be done in Rstudio
- Think of ‘pull’ request here as actually a merge request (in fact, GitLab calls it such)
- Go to GH and the branch you want to make the main branch



Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks or learn more about diff comparisons.

base: master ← compare: HS ✓ Able to merge. These branches can be automatically merged.

Add a title

Sub header for HS branch

Add a description

Write | Preview

Main code edited.

Markdown is supported | Paste, drop, or click to add files

Create pull request

Remember, contributions to this repository should follow our GitHub Community Guidelines.

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

Development
Use Closing keywords in the description to automatically close issues

Helpful resources
GitHub Community Guidelines

Sub header for HS branch #1

Open hareshsuppiah wants to merge 1 commit into `master` from `HS`

Conversation 0 | Commits 1 | Checks 0 | Files changed 1

hareshsuppiah commented now — Owner

Main code edited.

Sub header for HS branch 0fa43b2

Add more commits by pushing to the `HS` branch on **hareshsuppiah/Local-repo-demo**.

Require approval from specific reviewers before merging

Rulesets ensure specific people approve pull requests before they're merged.

Add rule

This branch has no conflicts with the base branch

Merging can be performed automatically.

Merge pull request

You can also open this in GitHub Desktop or view command line instructions.

Revie
No re
Still i
Assig
No o
Label
None
Proje
None
Miles
No m
Deve
Succe

Sub header for HS branch #1

Merged hareshsuppiah merged 1 commit into `master` from `HS` now

Conversation 0 | Commits 1 | Checks 0 | Files changed 1

hareshsuppiah commented now — Owner

Main code edited.

Sub header for HS branch `0fa43b2`

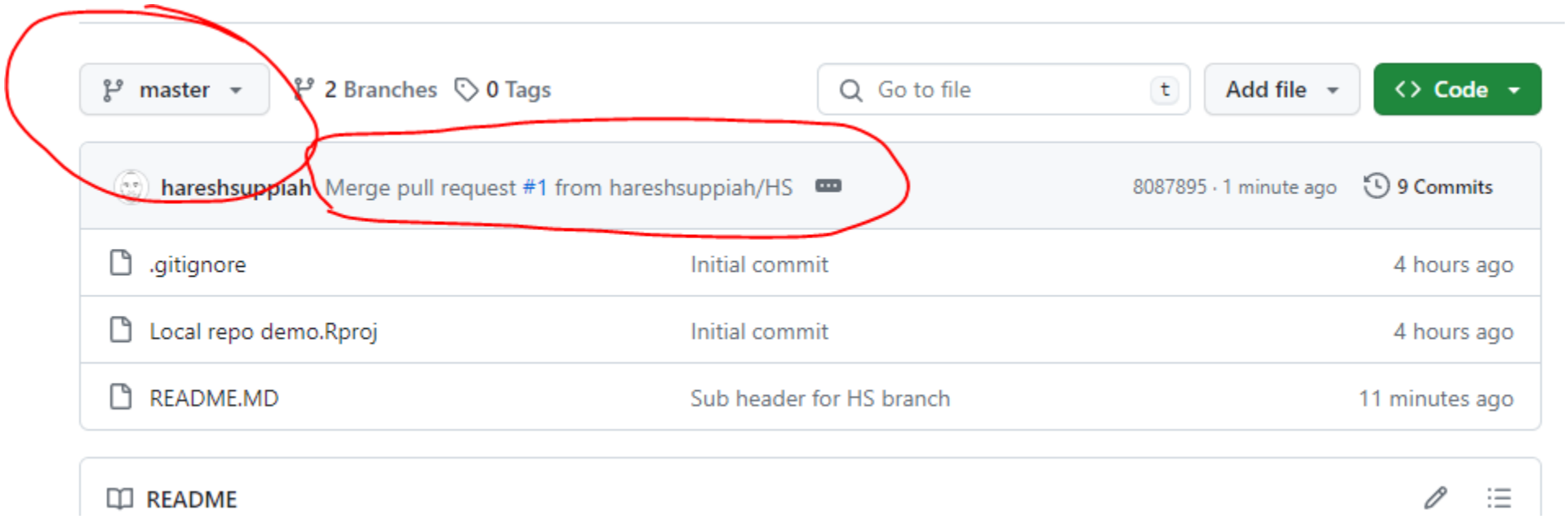
hareshsuppiah merged commit **8087895** into `master` now — Revert

Pull request successfully merged and closed

You're all set—the `HS` branch can be safely deleted.

Delete branch

Add a comment



Your turn

- Submit a pull request on GitHub
- Merge it with master branch