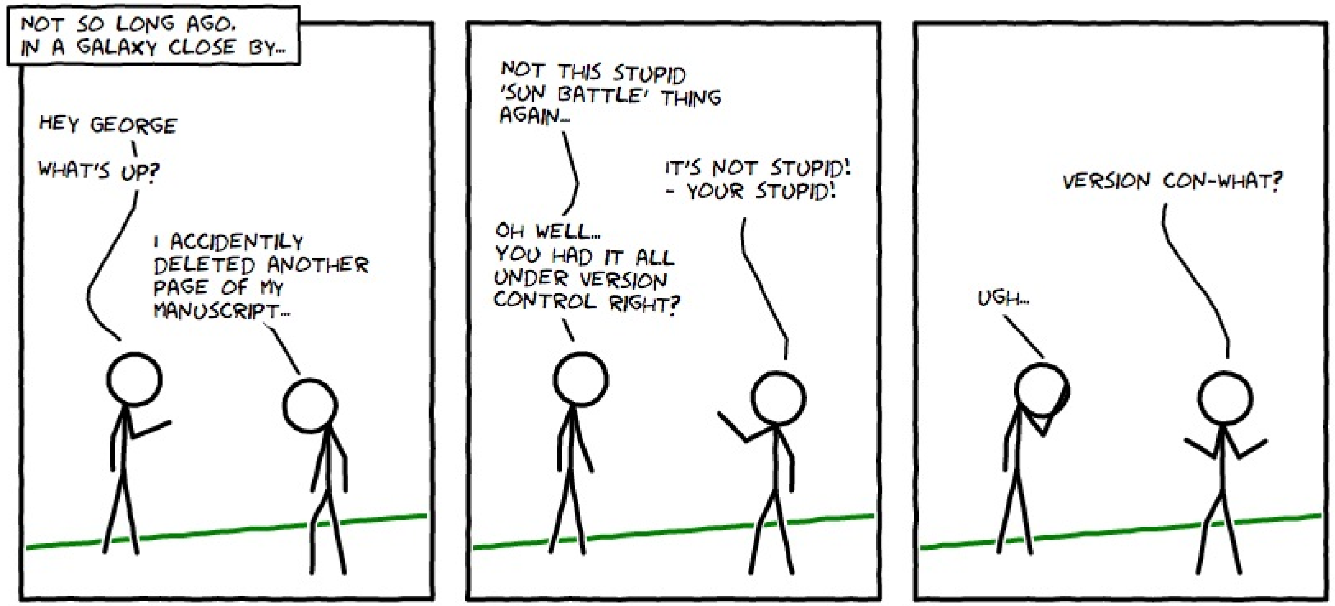
Version Control with git in Data Science Projects

Chair of Business Informatics, esp., Social Media and Data Science
Linus Hagemann

Structure

1. Why use version control?
2. How does git work?
3. How do I use git and GitHub in RStudio?
4. What's next?

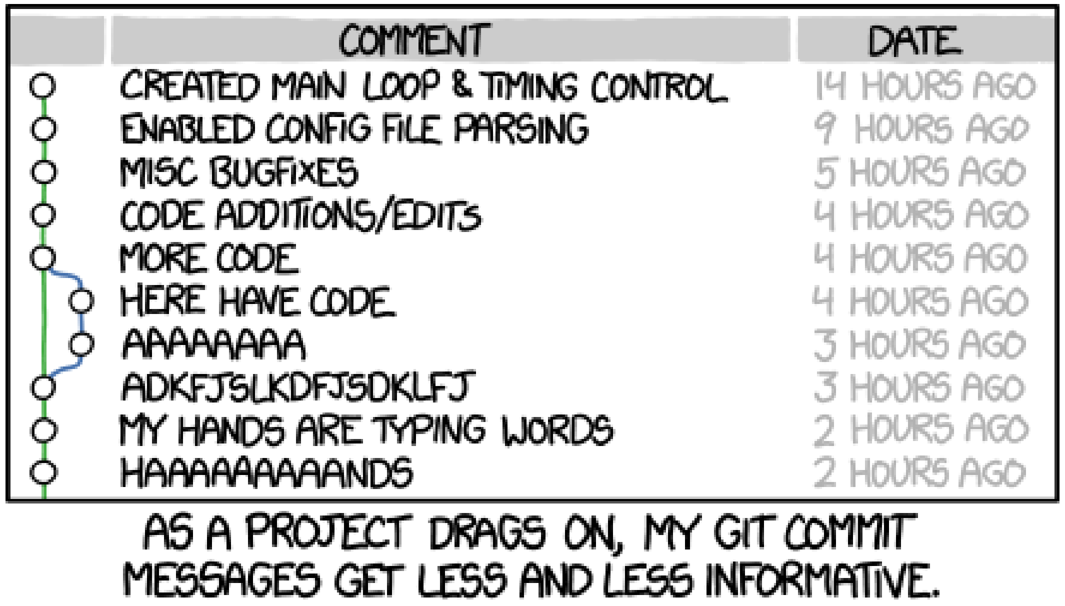
Why use Version Control? (1)


NOT SO LONG AGO. IN A GALAXY CLOSE BY...
HEY GEORGE
WHAT'S UP?
I ACCIDENTLY DELETED ANOTHER PAGE OF MY MANUSCRIPT...
NOT THIS STUPID 'SUN BATTLE' THING AGAIN...
OH WELL... YOU HAD IT ALL UNDER VERSION CONTROL RIGHT?
IT'S NOT STUPID! - YOUR STUPID!
UGH...
VERSION CON-WHAT?

smutch, CC BY-SA 3.0

That is the tool that allows to easily undo mistakes & affords worry-free explorative coding.

Why use Version Control? (2)



	COMMENT	DATE
○	CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
○	ENABLED CONFIG FILE PARSING	9 HOURS AGO
○	MISC BUGFIXES	5 HOURS AGO
○	CODE ADDITIONS/EDITS	4 HOURS AGO
○	MORE CODE	4 HOURS AGO
○	HERE HAVE CODE	4 HOURS AGO
○	AAAAAAAA	3 HOURS AGO
○	ADKFJSLKDFJSDKLFJ	3 HOURS AGO
○	MY HANDS ARE TYPING WORDS	2 HOURS AGO
○	HAAAAAAAANDS	2 HOURS AGO

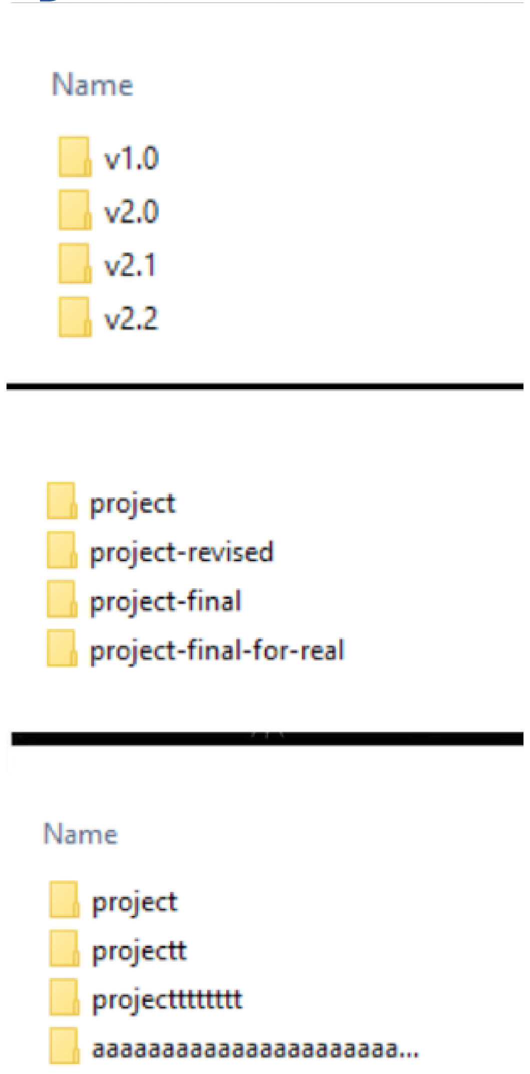
AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

xkcd, CC BY-NC 2.5

It adds some context to changes you made, making it easier for you and other people to make sense of them in the future.

...at least with some discipline.

Why use Version Control? (3)



Name

v1.0
v2.0
v2.1
v2.2

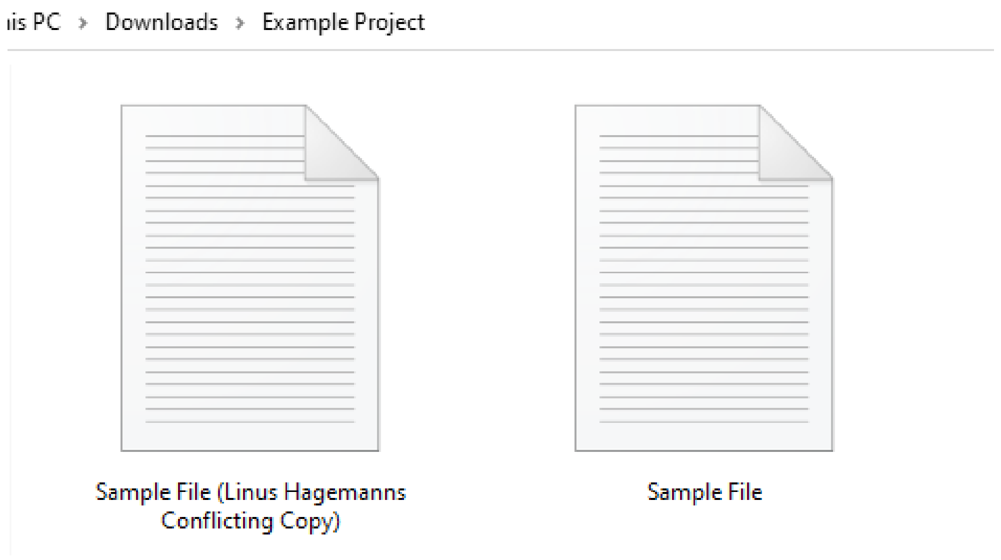
project
project-revised
project-final
project-final-for-real

Name

project
projectt
projecttttttttt
aaaaaaaaaaaaaaaaaaaaaa...

It structures your changes in a sensible way and saves us from ever-growing file naming-schemes.

Why use Version Control? (4)



ıis PC › Downloads › Example Project
Sample File (Linus Hagemanns Conflicting Copy)
Sample File

It allows to easily merge together new code that you and e.g., your project-partner wrote.

...at least most of the time.

Why git (and what is it)?

- git is the de-facto standard for version control systems
  - There is a ton of help available online
  - There are many tools and integrations that make working with git pleasant
  - The basics of git have become a valuable and required industry skill
- git is free, open-source software
  - Free as in “Freibier”, but also
  - Free as in freedom
- git is a distributed version control system
  - Enabling collaboration is a core design-goal of git

...and GitHub?

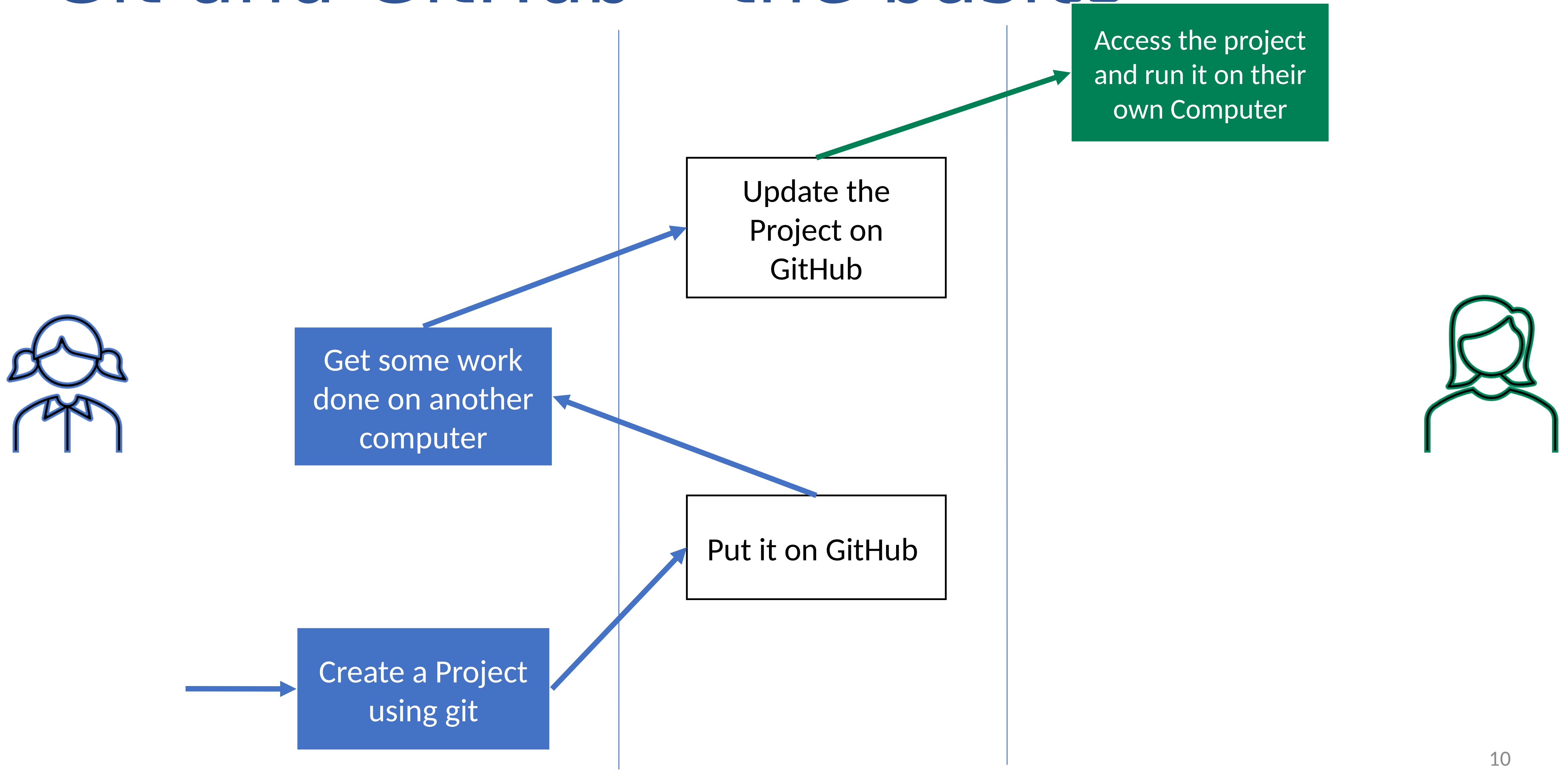
- GitHub is an online platform that
  - provides a multitude of features for collaboration using git
  - provides an ever-growing set of project management tools
  - makes open-source software discoverable in a single place
  - **stores projects managed by git**
- There are alternatives like GitLab or BitBucket, which have their benefits and disadvantages, all are fine.
- We use GitHub here since it provides the easiest interface and it is the most widespread platform.

A Few Practical Applications

Git for studies and work:

1. Manage your personal programming and data science projects, as well as academic writing.
2. Knowing git is the first step towards contributing to and using open-source projects.
   E.g., the code for many R packages can be found on GitHub
3. Use GitHub like your personal Data Science/Programming portfolio.
   A well-maintained profile with some projects and activity can provide a headstart when applying for jobs.

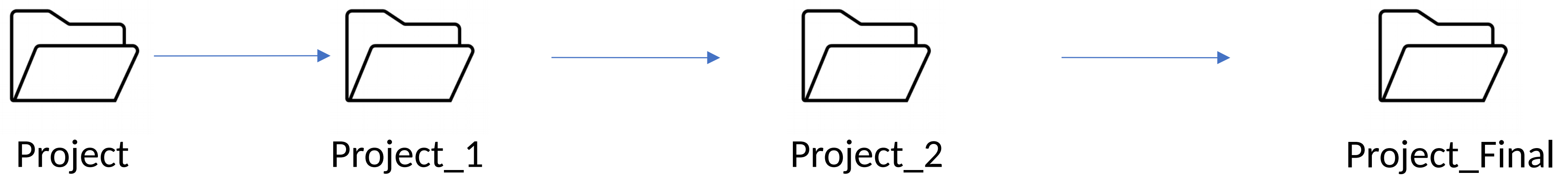
Git and GitHub – the basics


Access the project and run it on their own Computer
Update the Project on GitHub
Get some work done on another computer
Put it on GitHub
Create a Project using git

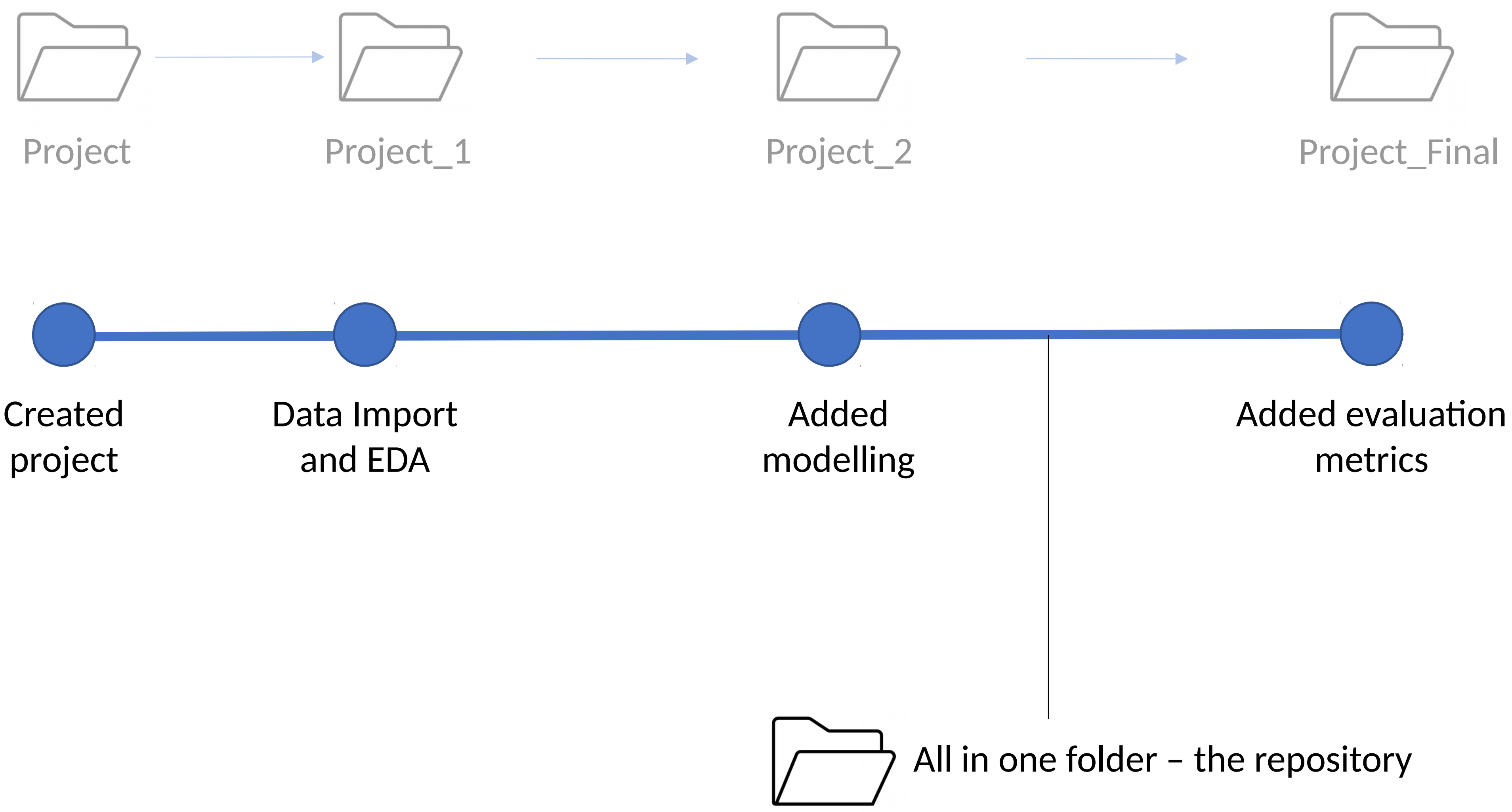
Working with git / git Terminology

- Projects are called **repositories.**
- git will **track** files, in order to observe changes in them.
- When you tell it to, git will take all files that are **staged**
- and create a new **commit** with their changes.
  - commits are basically collections of file copies with their content at a given time
  - staged files therefore are files of which changes should be considered for the next commit
- Since a commit contains the tracked files as they were when the commit was made, we can always **checkout** a commit from the past.
- Commits are associated with a **commit-message**.

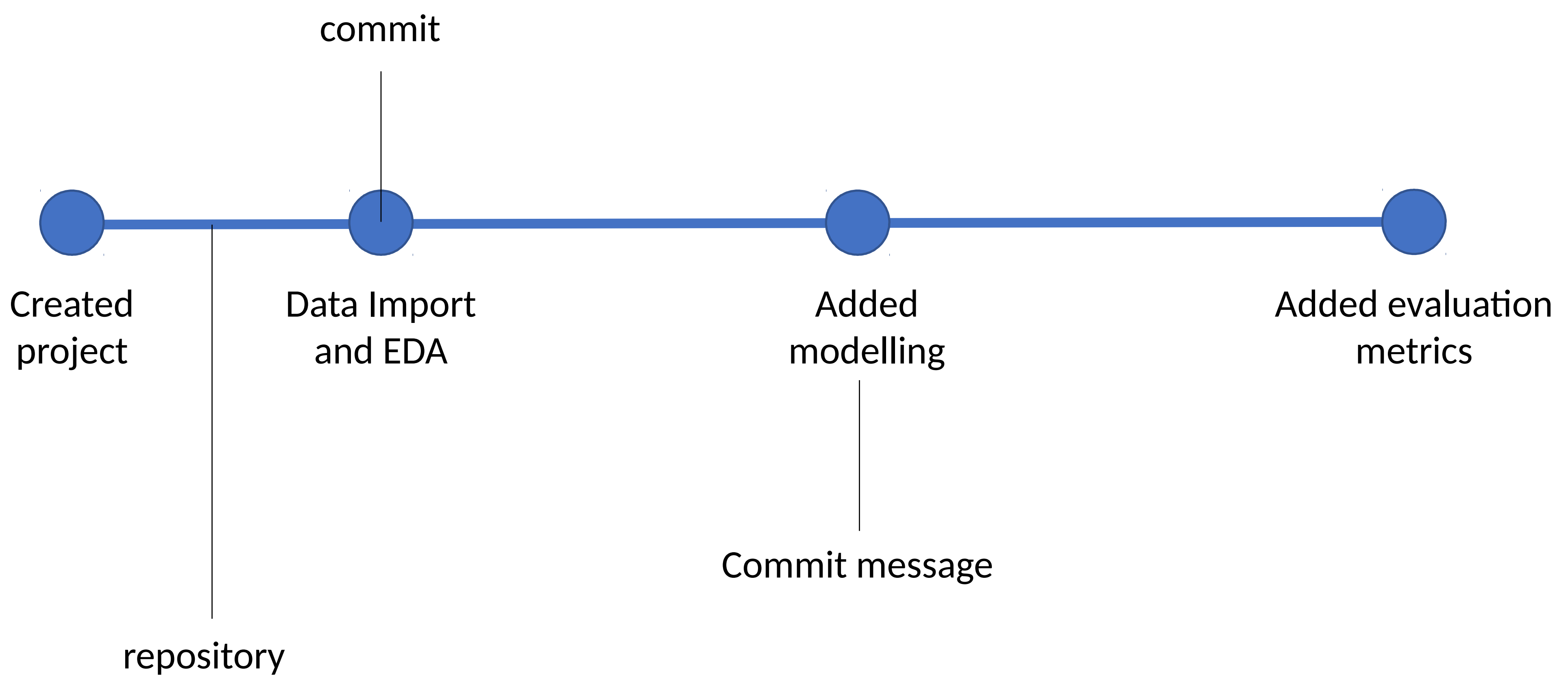
Working with git on a FoDS Project (1)



Working with git on a FoDS Project (2)

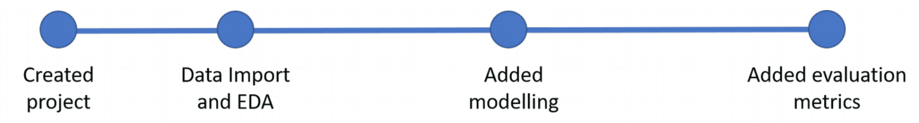


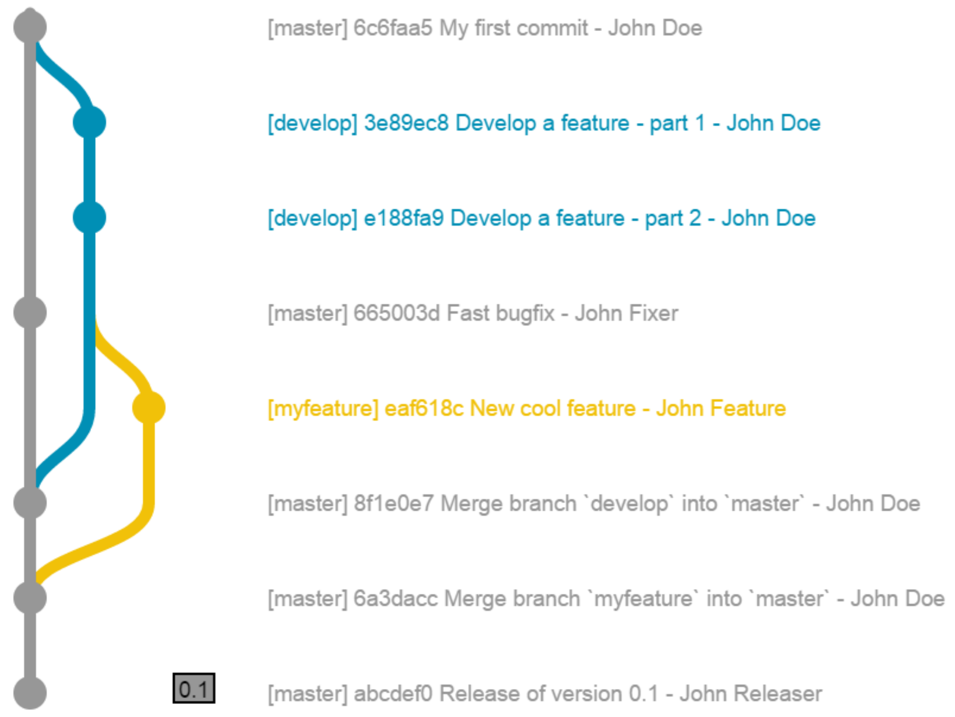
git Terminology Revisited



Excursus: git internals

- The internal data structure of git looks pretty similar
  - A Graph of commits
- This is also what allows us to “jump around in time” and merge the changes of multiple people that occurred on the same files
  - Commits are clearly ordered, each one having parents and children





```
[master] 6c6faa5 My first commit - John Doe
[develop] 3e89ec8 Develop a feature - part 1 - John Doe
[develop] e188fa9 Develop a feature - part 2 - John Doe
[master] 665003d Fast bugfix - John Fixer
[myfeature] eaf618c New cool feature - John Feature
[master] 8f1e0e7 Merge branch `develop` into `master` - John Doe
[master] 6a3dacc Merge branch `myfeature` into `master` - John Doe
[master] abcdef0 Release of version 0.1 - John Releaser
```

fracz, CC BY-SA 4.0

Diving In – A git(Hub) Workflow With RStudio

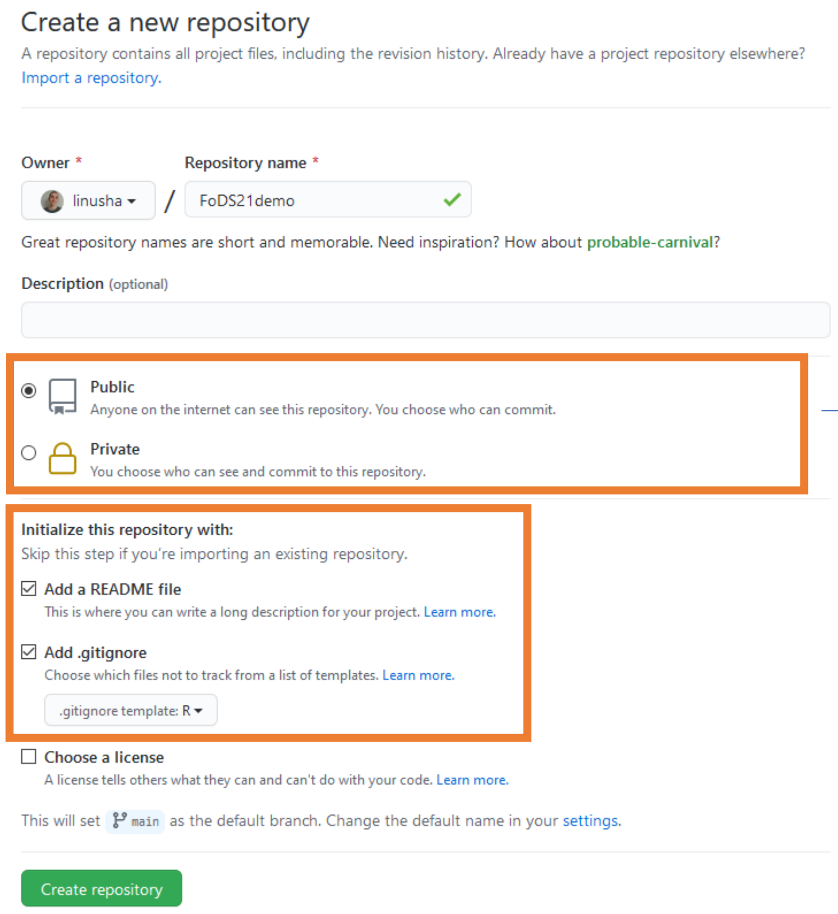
Prerequisites *(see the Setup slides provided on Moodle)*:

- A GitHub Account
- RStudio and git installed

- We will cover one possible workflow to use git and GitHub in RStudio.
- If you have used git before and are comfortable with using a command line, feel free to continue using git in the preferred way. There are many possible ways of use. The proposed way is our recommendation for starters.
- **The following slides are intended as a step-by-step guide, that can be used for later reference.**

Creating a New Repository

Login to your GitHub Account and go to **github.com/new**



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * linusha / Repository name * FoDS21demo

Great repository names are short and memorable. Need inspiration? How about probable-carnival?

Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

- [x] Add a README file
This is where you can write a long description for your project. Learn more.

- [x] Add .gitignore
Choose which files not to track from a list of templates. Learn more.

.gitignore template: R

- [ ] Choose a license
A license tells others what they can and can't do with your code. Learn more.

This will set `main` as the default branch. Change the default name in your settings.

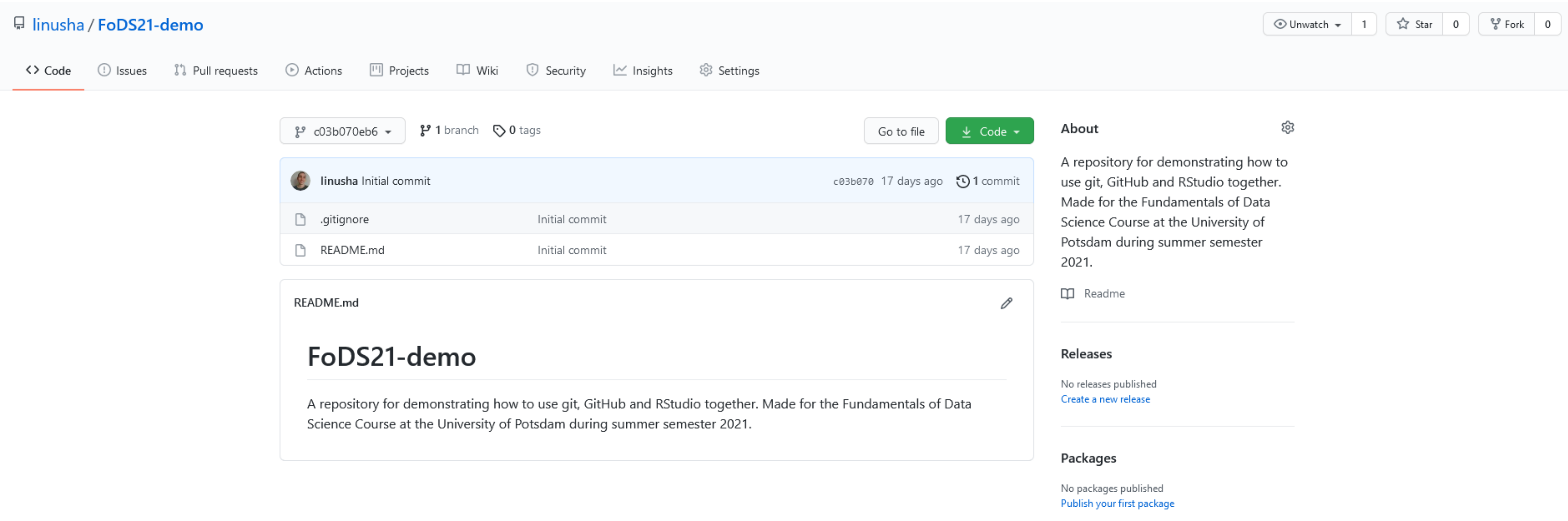
Create repository

For the demo, I propose to use a private Repositoryю

Note: either choice is fine, just remember that "public" means that everyone can see your project at all times.

Select "Add a README file"
Select the "Add .gitignore"

Creating a new Repository



linusha Initial commit c03b070 17 days ago 1 commit

.gitignore	Initial commit	17 days ago
README.md	Initial commit	17 days ago

README.md

FoDS21-demo

A repository for demonstrating how to use git, GitHub and RStudio together. Made for the Fundamentals of Data Science Course at the University of Potsdam during summer semester 2021.

About

A repository for demonstrating how to use git, GitHub and RStudio together. Made for the Fundamentals of Data Science Course at the University of Potsdam during summer semester 2021.

Readme

Releases

No releases published
Create a new release

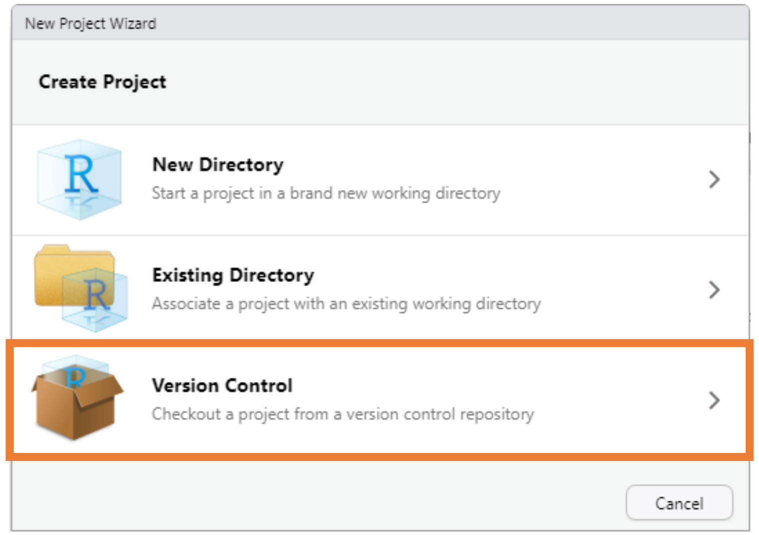
Packages

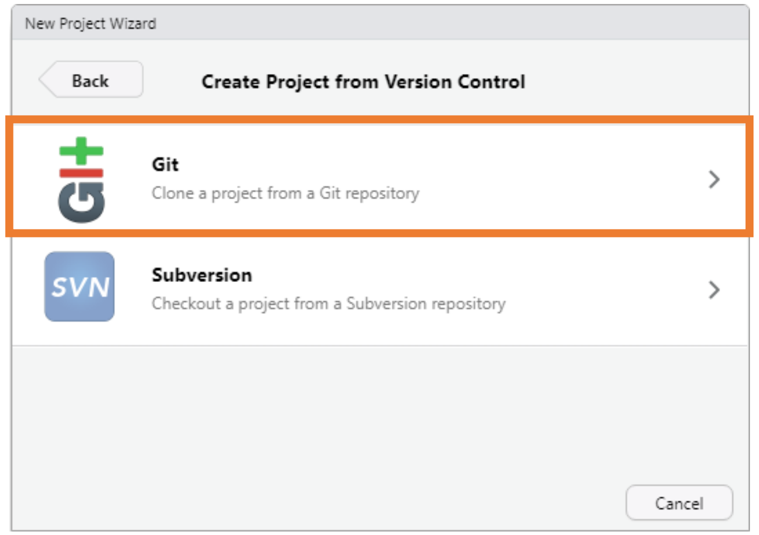
No packages published
Publish your first package

- Keep the URL of the resulting site at hand
- Open RStudio

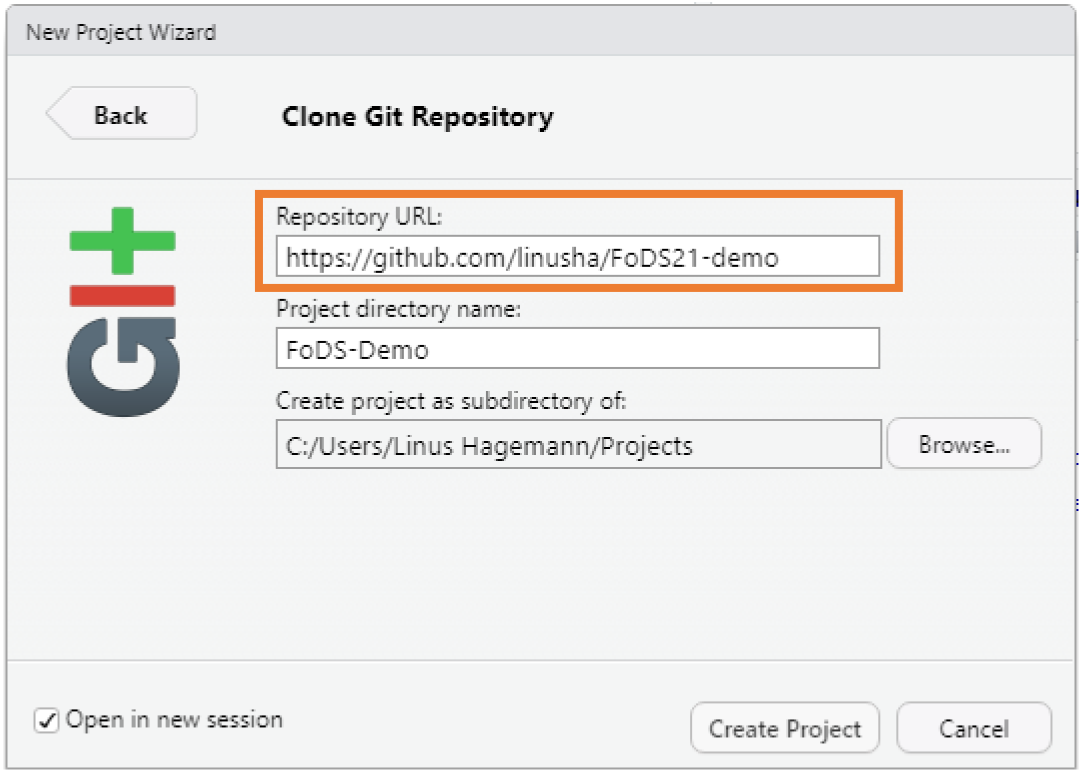
Creating an RStudio Project from Version Control (1)

Choose File -> New Project in RStudio


New Project Wizard
Create Project
New Directory
Start a project in a brand new working directory
Existing Directory
Associate a project with an existing working directory
Version Control
Checkout a project from a version control repository
Cancel


New Project Wizard
Back
Create Project from Version Control
Git
Clone a project from a Git repository
Subversion
Checkout a project from a Subversion repository
Cancel

Creating a RStudio Project from Version Control (2)



Repository URL:
https://github.com/linusha/FoDS21-demo

Project directory name:
FoDS-Demo

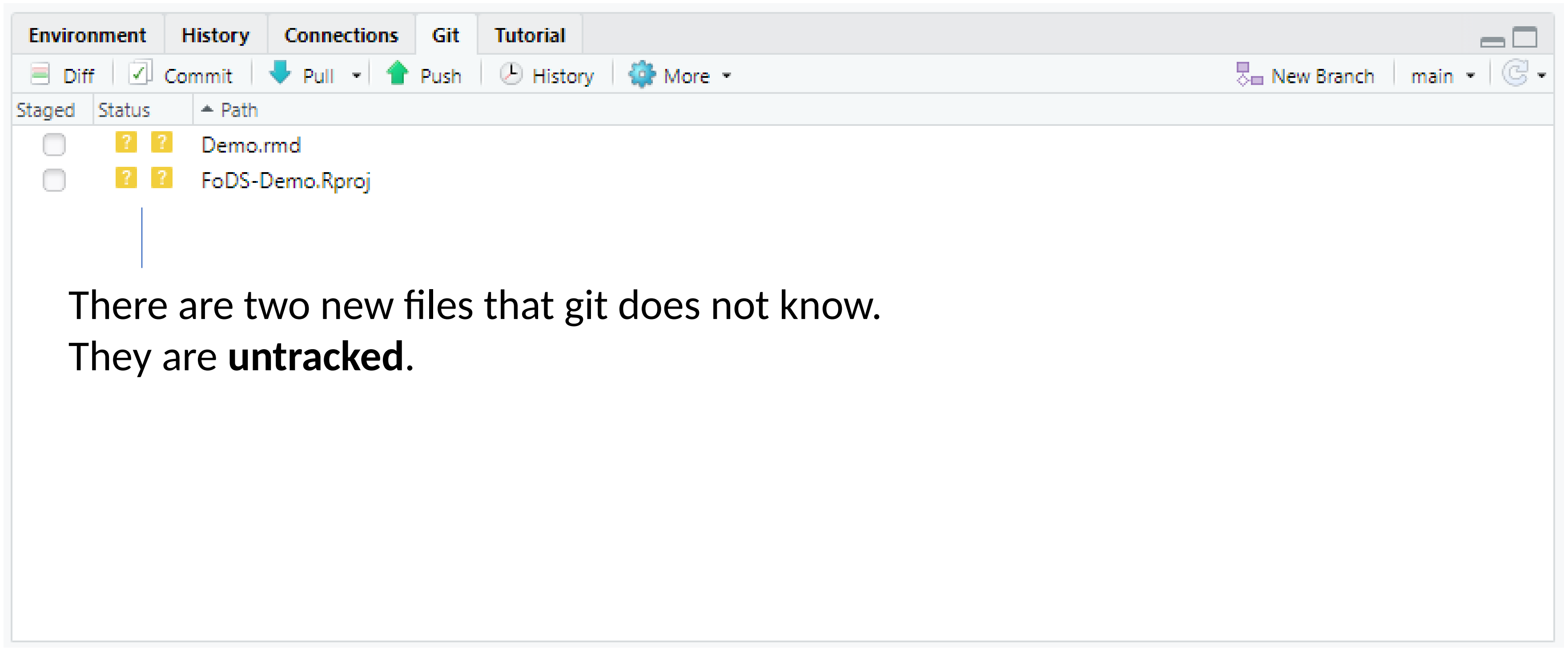
Create project as subdirectory of:
C:/Users/Linus Hagemann/Projects

✓ Open in new session

- Fill in the URL of the repository that you created before.
- Save the new project where you decide.
- Choose **Create Project**
    - A popup with a status bar appears for a short time. You do not need to react. Afterwards your Project opens.
    - If you chose a private repository, you might be prompted for your GitHub credentials.

Making Your First Commit (1)

1. Create a new RMarkdown file in your project and save it
2. Take a look at the Git pane in RStudio:



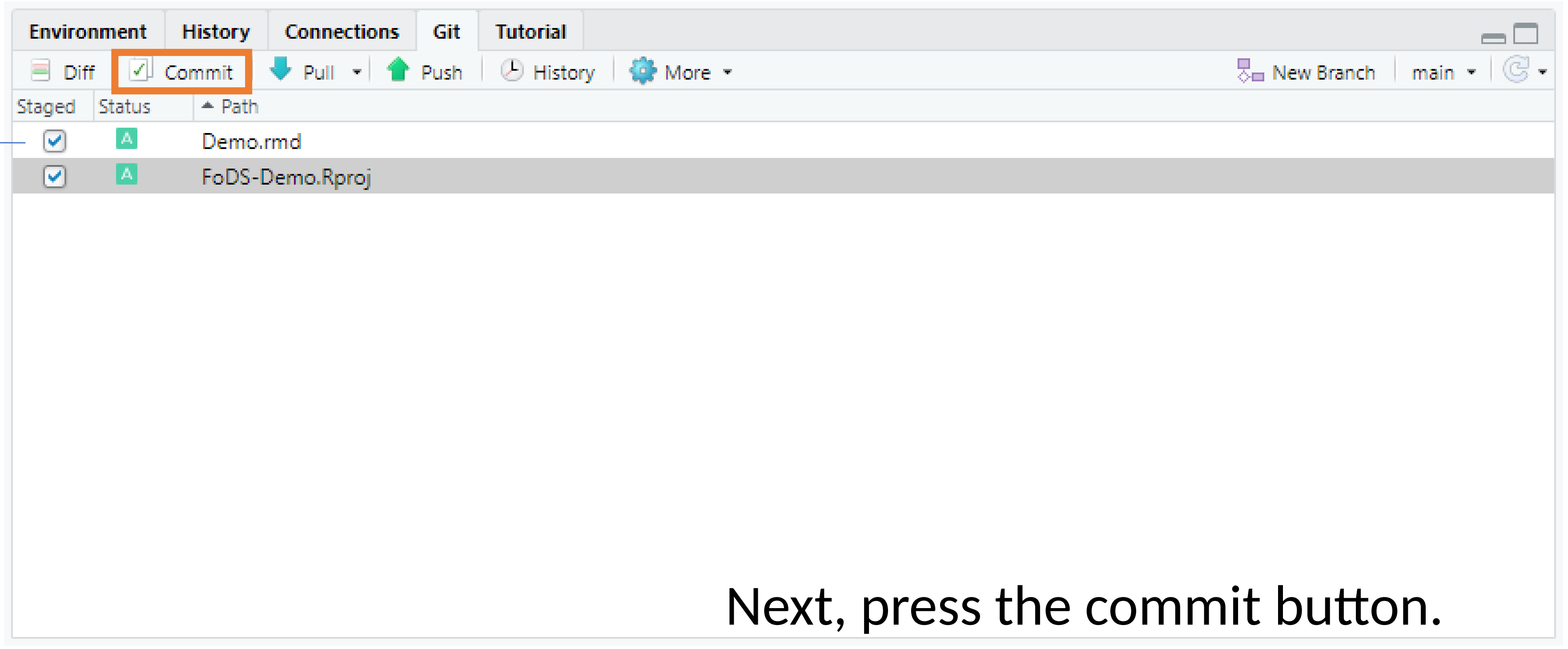
Staged	Status	Path
☐	? ?	Demo.rmd
☐	? ?	FoDS-Demo.Rproj

There are two new files that git does not know.
They are **untracked**.

Making Your First Commit (2)

Stage them by ticking the box.

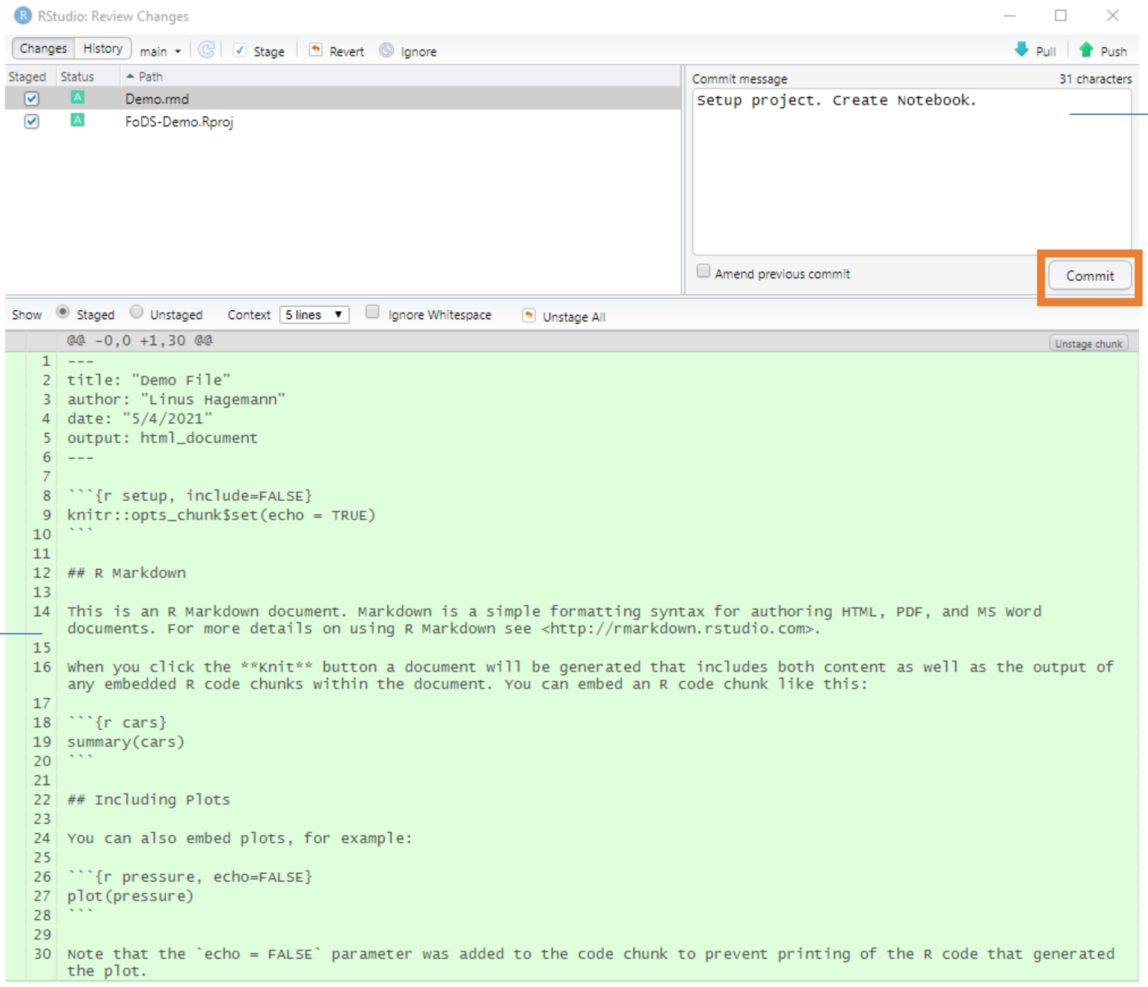
This means our next commit will take the checked files into account.



Staged	Status	Path
✓	A	Demo.rmd
✓	A	FoDS-Demo.Rproj

Next, press the commit button.

Committing



Staged	Status	Path
✓	A	Demo.rmd
✓	A	FoDS-Demo.Rproj

Commit message: `Setup project. Create Notebook.`

~~~
@@ -0,0 +1,30 @@
---
title: "Demo File"
author: "Linus Hagemann"
date: "5/4/2021"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
~~~

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word
documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of
any embedded R code chunks within the document. You can embed an R code chunk like this:

```{r cars}
summary(cars)
```

Including Plots

You can also embed plots, for example:

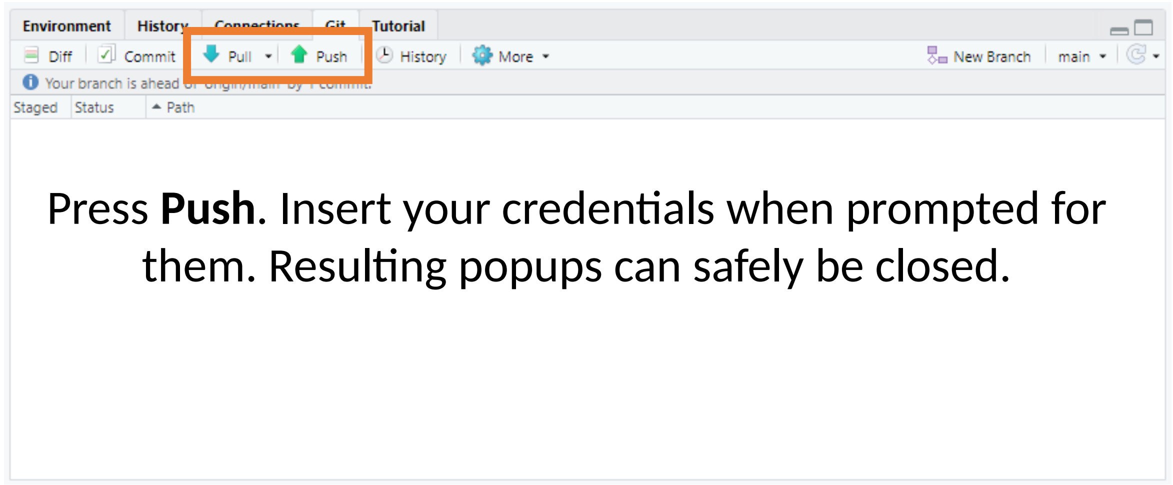
```{r pressure, echo=FALSE}
plot(pressure)
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated
the plot.
```

The **commit message**.

Make it a concise summary of the changes you made.

You can see the "**diff**" of all changes that will be committed. The "diff" shows which changes occurred since the last commit.

Addition are green.
Deletions are red.
```

## Syncing with GitHub (1)

- Congratulations, you just made your first commit! ^^

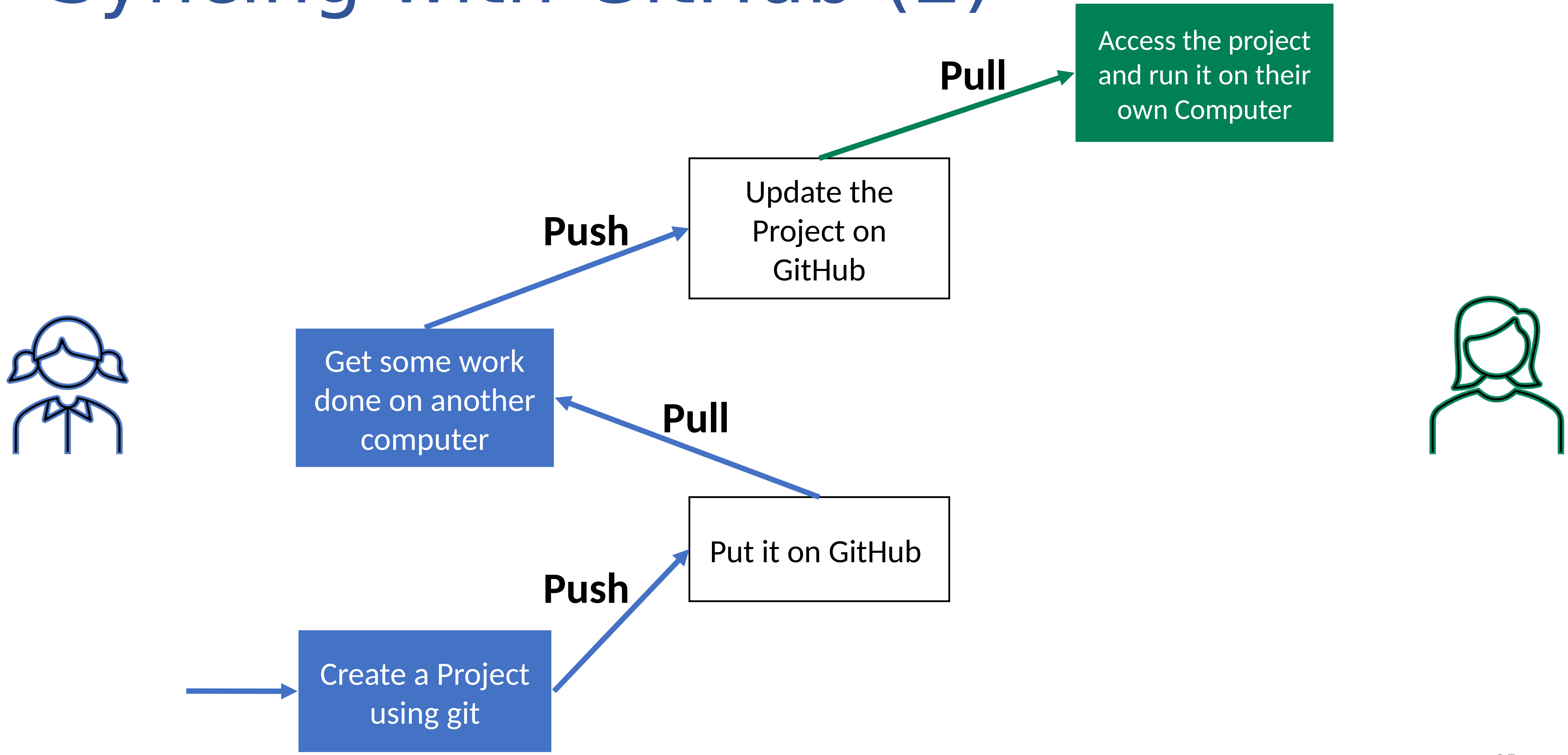
- For syncing with GitHub, there are two possible scenarios:
  - You want to put changes you made to GitHub -> **Push** them there
  - You want to get changes that are already on GitHub on your local computer -> **Pull** them



Your branch is ahead of 'origin/main' by 1 commit.

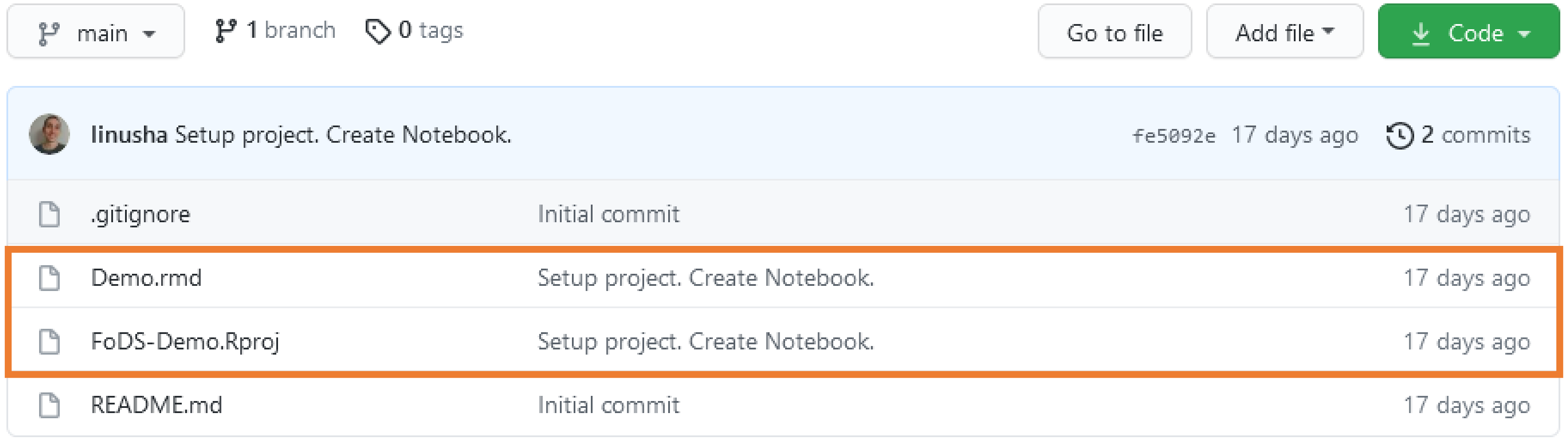
Press **Push**. Insert your credentials when prompted for them. Resulting popups can safely be closed.

## Syncing with GitHub (2)


Access the project and run it on their own Computer
Pull
Update the Project on GitHub
Push
Get some work done on another computer
Pull
Put it on GitHub
Push
Create a Project using git

## Syncing with GitHub (3)

- Go back to the repository site on GitHub.
- You will see the changes we just made there.



linusha Setup project. Create Notebook. fe5092e 17 days ago 2 commits

| | | |
|---|---|---|
| .gitignore | Initial commit | 17 days ago |
| Demo.rmd | Setup project. Create Notebook. | 17 days ago |
| FoDS-Demo.Rproj | Setup project. Create Notebook. | 17 days ago |
| README.md | Initial commit | 17 days ago |

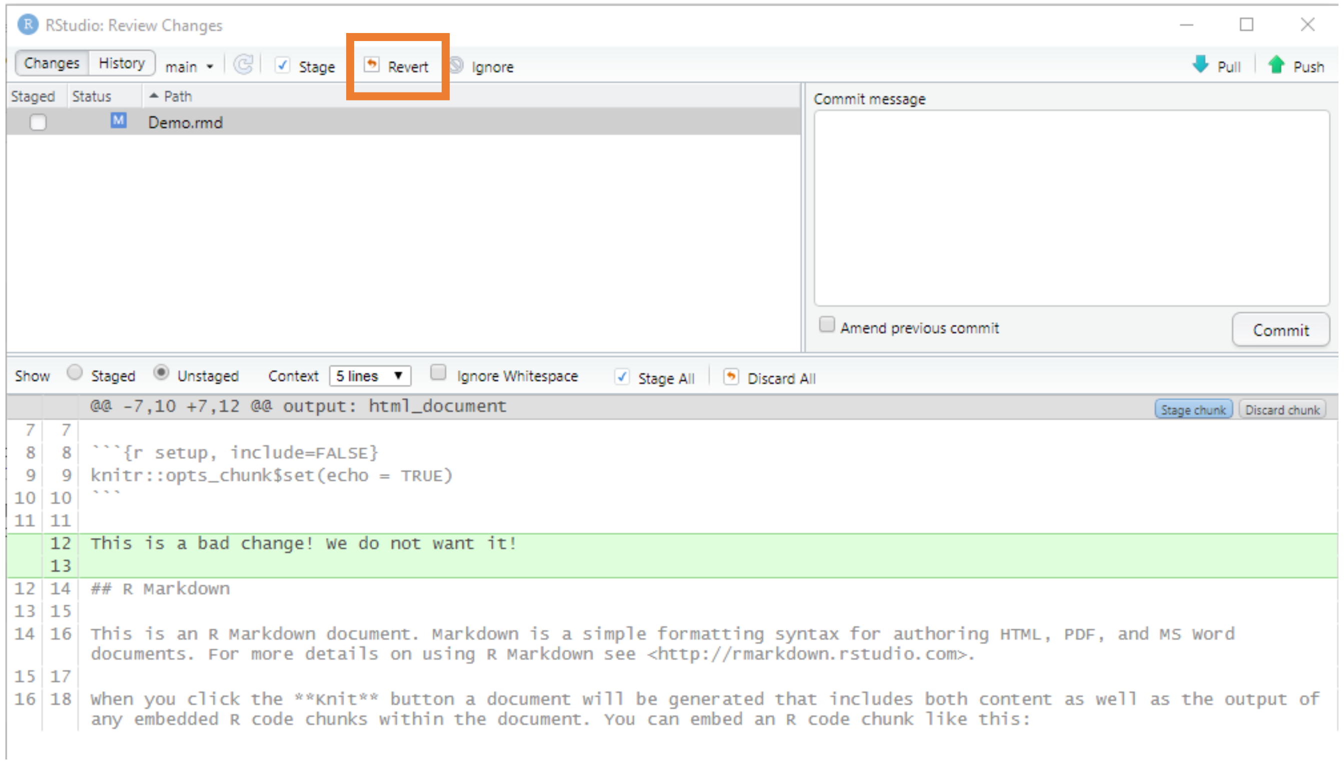
## The Workflow

- Make some changes
- When you want to safely store your progress after atomic changes (e.g., a chapter of your report, a changed function, a task of your homework,...) make a commit
  - Put effort in your commit messages, as they make it easier to find changes later
- Push your work to GitHub regularly
- If you want to work on an existing project e.g., from another computer, you can follow the steps beforehand expect the repository creation

## Collaborating

- Instructions on how to add collaborators to your repository on GitHub can be found here
- The overall process stays the same
  - If you were added as a collaborator by your project partner, you can follow the above steps, minus the creation of the repository.
- To get changes your collaborators pushed, perform a **pull** operation
- Important:
  - Before you push your changes, **always perform a pull** if someone else might have pushed changes before you
  - This will minimize the risk of **merge conflicts** happening
  - Some pointers on those conflicts can be found on later slides, we cannot talk about them in depth here.

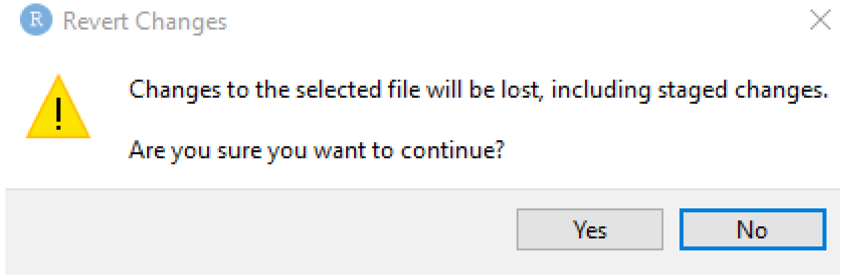
## Reverting Changes



Demo.rmd

~~~
@@ -7,10 +7,12 @@ output: html_document

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

This is a bad change! We do not want it!

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word
documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of
any embedded R code chunks within the document. You can embed an R code chunk like this:
~~~

Will **discard** all changes in the selected file. The file will then be in the same state as it was when you did your last commit.
Hence, commit regularly!



Changes to the selected file will be lost, including staged changes.

Are you sure you want to continue?

What’s Next?

- We could only discuss a small set of features compared to everything that git offers.
- However, applying and practising these skills …
  - should make it easier to grasp more advanced features.
  - provide extra value to your projects for studies and work.
- We encourage you to use these skills when working on your FoDS projects!
  - Questions? Problems? -> **Moodle Forum**

- Remember the memo:

CHANGE - COMMIT - PULL - PUSH

Additional Resources

- Comprehensive information about using git with GitHub and RStudio:
  https://happygitwithr.com/
- A Cheatsheet about the Git integration of RStudio

- The complete Git Pro book online
- dangitgit.com

Questions?

Thank You!