



Reproducible Workflow

Git

Michelle Donzallaz

3. What is git and how can I use it?

Slides and Material

You can find the slides and materials here: <https://github.com/jstbcs/ReproducibleWorkflowWorkshop>.

Materials heavily build on previous workshops by Julia Haaf

What I would like to show you about git

- How to use a terminal
- Git
 - What is it good for?
 - What is it?
 - What can it do?
- Set-up for your computer
 - GUI/terminal
 - R Studio & git
 - SSH
 - Set name & email address
- Your first repo
 - Github and GitLab
 - In R Studio
 - .gitignore
 - README
- Workflow
 - Add, Commit, Push
 - Diff
 - Merge, Branches, Tagging... (all the cool stuff)
 - What happens if something goes wrong? (And it will.)

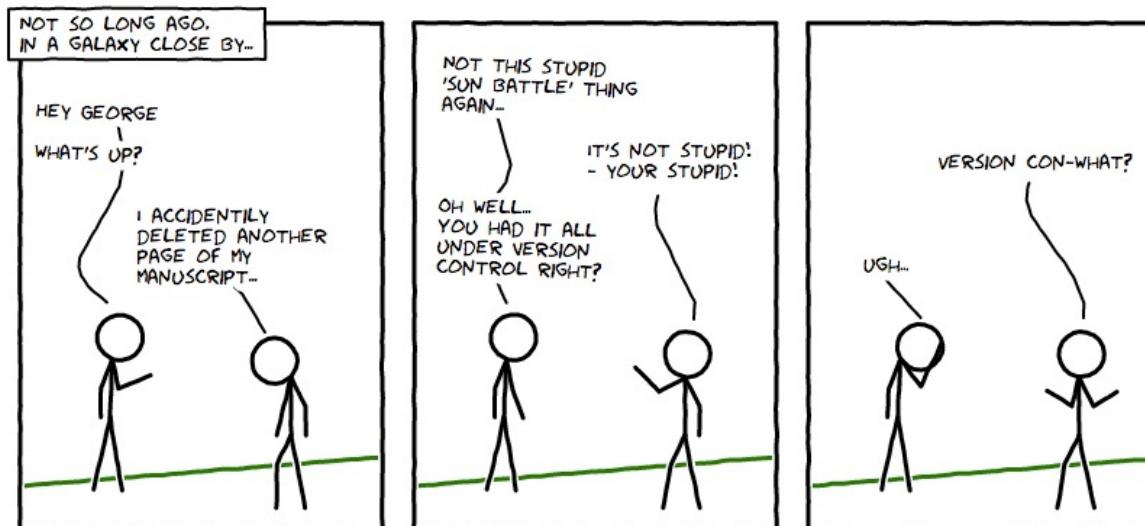
What we have time for

- *How to use a terminal*
- Git
 - **What is it good for?**
 - **What is it?**
 - What can it do?
- Set-up for your computer
 - *GUI/terminal*
 - **R Studio & git**
 - *SSH*
 - Set name & email address
- Your first repo
 - *Github and GitLab*
 - **In R Studio**
 - **gitignore**
 - **README**
- Workflow
 - **Add, Commit, Push**
 - Diff
 - Merge, Branches, Tagging... (all the cool stuff)
 - What happens if something goes wrong? (And it will.)

What is it good for?

Version control:

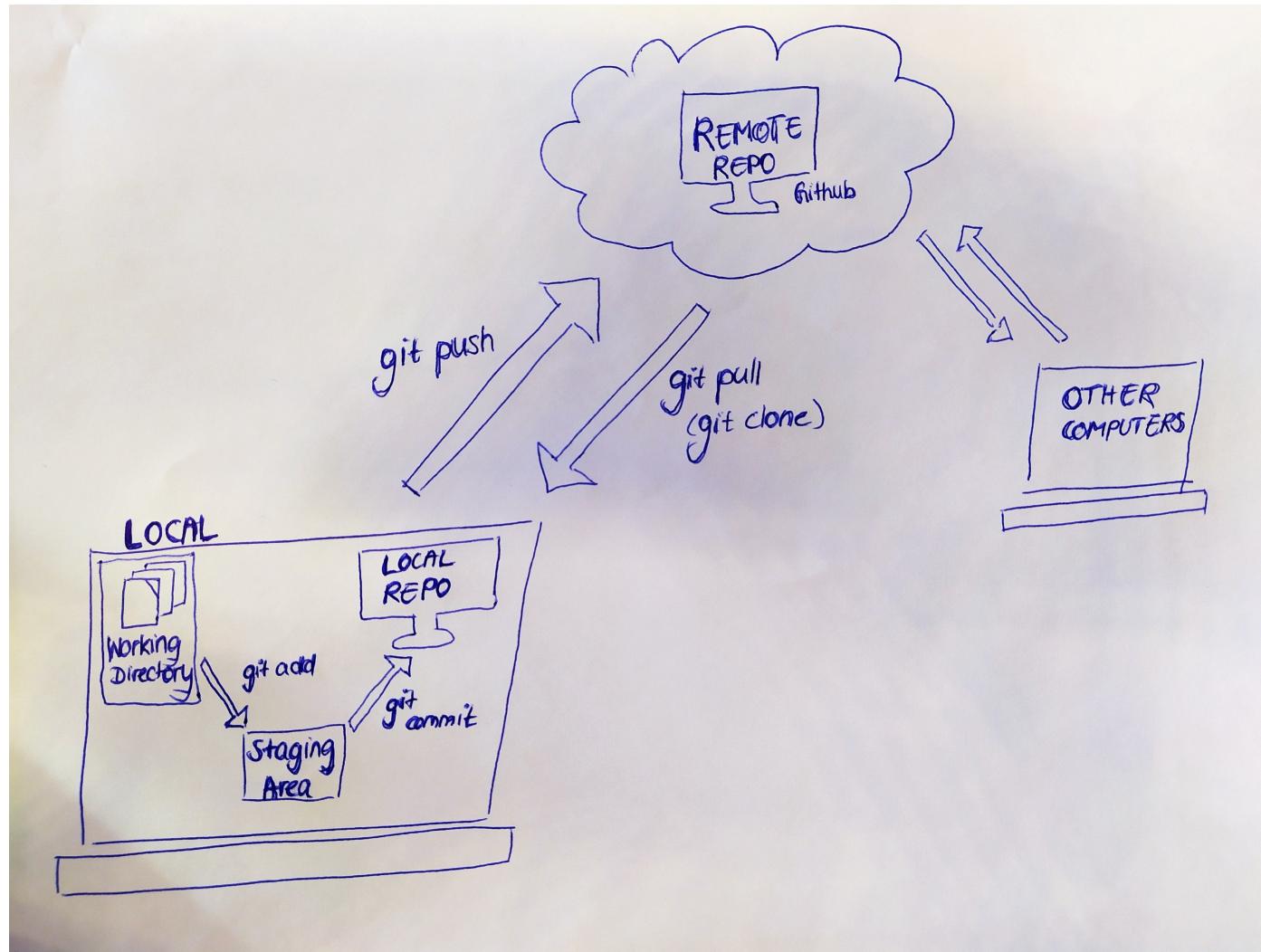
- Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.
- History of all changes (who, what, when).
- Helps to avoid mistakes (working on the wrong version, deleting, ...).
- Merging changes of multiple collaborators in one file.



What is it?

- Git is essentially a program on your computer, you have to communicate with it.
- GitHub (or GitLab) is a Git repository hosting service.

What is it?



What can it do?

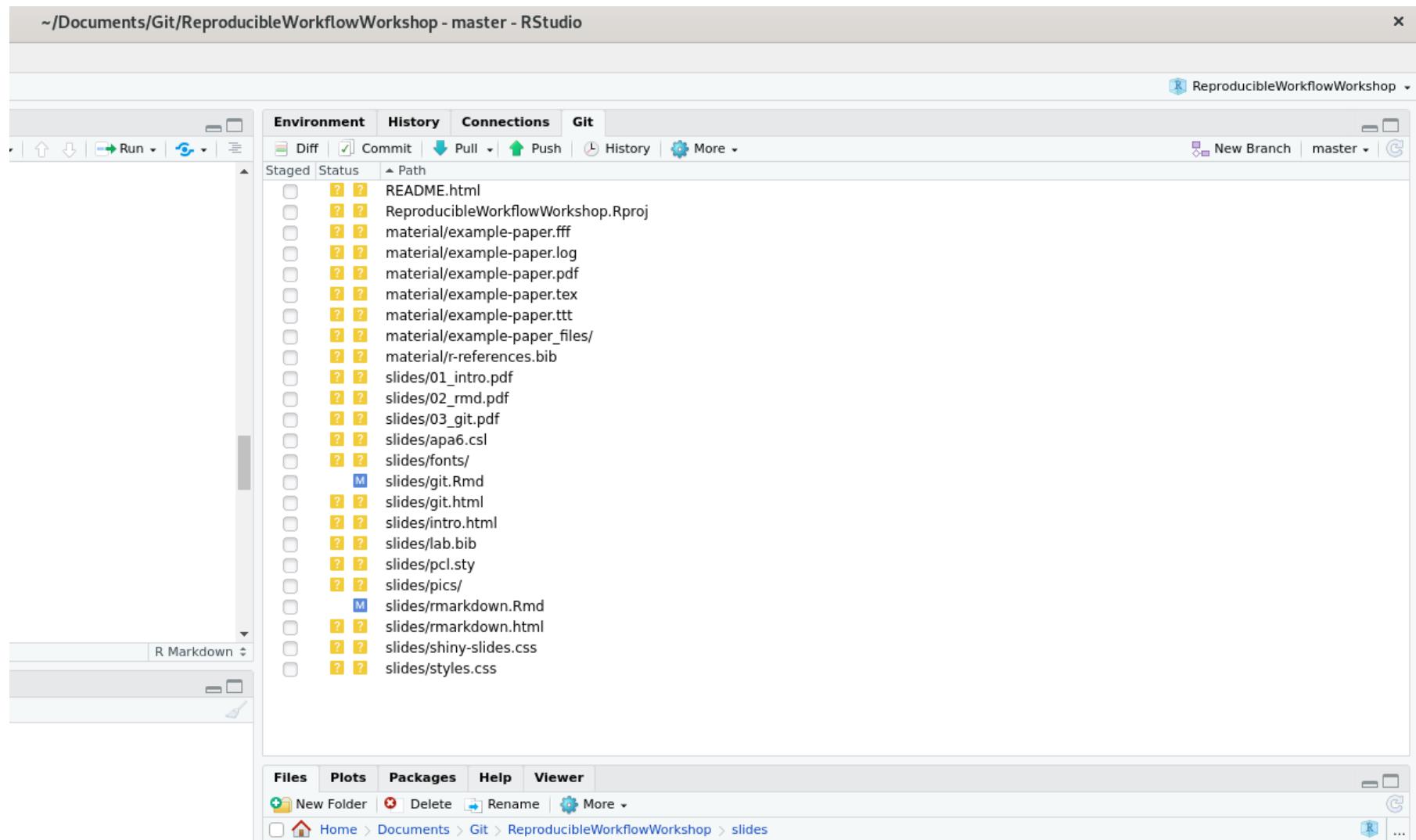
- A lot! Which is why I can only mention part of its functionality here.
- Working on one product in (large) teams.
- Working on things that can break.
- git can only integrate and show changes in text files.
- binary files (images, etc.) can be tracked and uploaded but changes cannot be shown.

Set-up for your computer

Using git

- Git does not have a user interface.
- You can either use the terminal, or install an additional interface.
- Github has its own GUI. Some people like it.
- We will use Rstudio (and the terminal) as user interface.

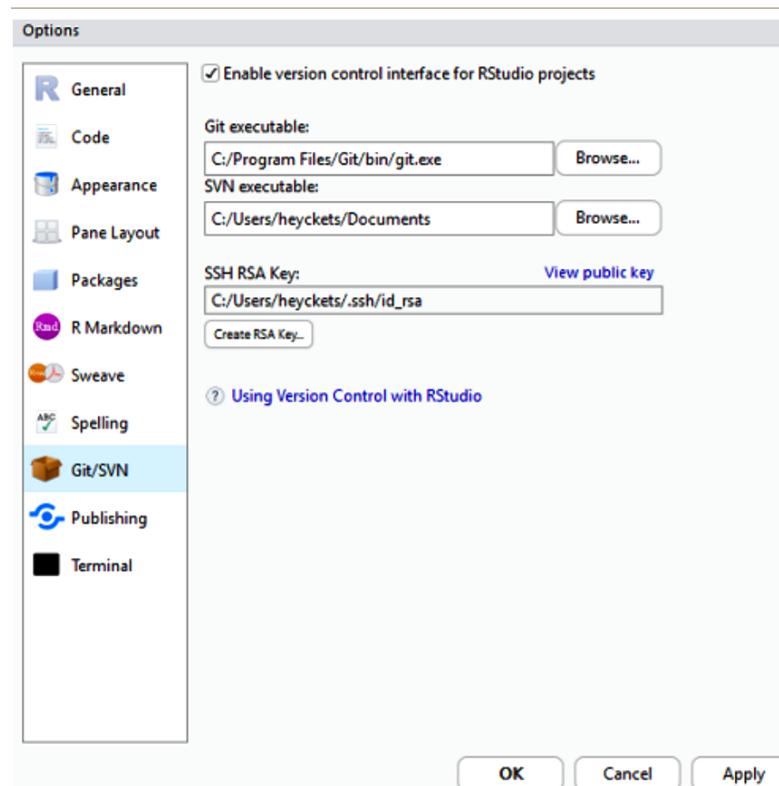
R Studio & git



R Studio & git

Tools ► Global Options ► Git/SVN.

Make sure the first box is ticked and “git.exe” (Windows) “usr/bin/git” (MacOS) is included in the first box.



Set name & email address

- Open the Terminal in R Studio.
- Set an email address and user name for git.

Set name & email address

- Open the Terminal in R Studio.
- Set an email address and user name for git.

```
git config --global user.email "myemail@email.com"  
git config --global user.name "My commit name"
```

Set name & email address

- In the R console

```
## set your user name and email:
usethis::use_git_config(user.name = "My commit name", user.email = "myemail@email.com")
```

First Repository!

Github

The screenshot shows the GitHub dashboard for the user 'jstbcs'. The left sidebar lists repositories such as 'Jeffrey-s-Platitude', 'snarc-indiff', 'ReproducibleWorkflowWorkshop', 'CreativeOrNot', 'PerceptionAndC... /rm-emotion', 'crsh/papaja', and 'vandenman/A-Cautionary-Note-on-Esti...'. It also shows sections for 'Your teams' and 'Explore repositories'. The main area features a prominent 'Learn Git and GitHub without any code!' guide with 'Read the guide' and 'Start a project' buttons. Below this, two recent pushes are listed: one from 'mariusbarth' to 'methexp/rawdata' and another to 'methexp/g-sync-test'.

Repositories

- jstbcs/Jeffrey-s-Platitude
- jstbcs/snarc-indiff
- jstbcs/ReproducibleWorkflowWorkshop
- jstbcs/CreativeOrNot
- PerceptionAndC... /rm-emotion
- crsh/papaja
- vandenman/A-Cautionary-Note-on-Esti...

Show more

Your teams

- Find a team...
- jasp-stats/internal-testers
- jasp-stats/issue-collaborators
- PerceptionAndCognitionLab/lab
- methexp/owners

Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Read the guide Start a project

mariusbarth pushed to methexp/rawdata 4 hours ago

1 commit to master
6f19793 automatic commit

mariusbarth pushed to methexp/g-sync-test 16 hours ago

4 commits to master

Securing software together

Welcome to the new dashboard. Get closer to the stuff you care about most.

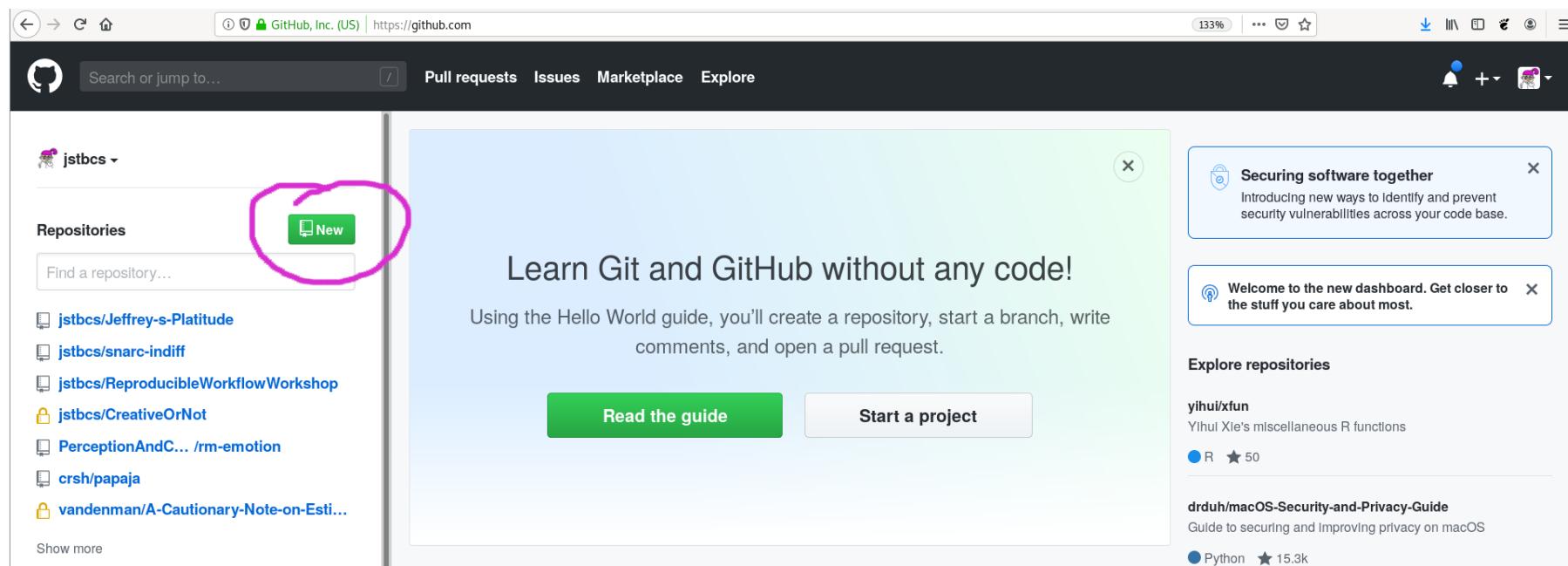
Explore repositories

- yihui/xfun
- drduh/macOS-Security-and-Privacy-Guide
- wch/extrafont

Explore more →

Github

New Repository



Github

New Repository

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner **Repository name ***

 **jstbcs** /

Great repository names are short and memorable. Need inspiration? How about [psychic-fortnight](#)?

Description (optional)

 **Public**
Anyone can see this repository. You choose who can commit.

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

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

Initialize this repository with a README
This will let you immediately clone the repository to your computer.

| 

Github

New Repository

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner **Repository name ***

 **jstbcs** / myfirstrepo 

Great repository names are short and memorable. Need inspiration? How about **psychic-fortnight?**

Description (optional)

This is my first repository. I will try out some stuff here.

 **Public**
Anyone can see this repository. You choose who can commit.

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

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

Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Add .gitignore: **None** | Add a license: **None** 

Create repository 

Github

Settings

The screenshot shows the GitHub repository settings page for 'myfirstrepo'. The repository is private, as indicated by the 'Private' badge. The top navigation bar includes links for Code, Issues (0), Pull requests (0), Projects (0), Security, Insights, and Settings. The Settings tab is currently selected. On the left, a sidebar titled 'Options' lists Collaborators, Branches, Webhooks, Notifications, Integrations & services, and Deploy keys. The main content area is titled 'Settings' and contains a 'Repository name' field with 'myfirstrepo' and a 'Rename' button. Below this is a section for 'Template repository' with a checkbox and explanatory text. A 'Social preview' section is also present.

jstbcs / myfirstrepo Private

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Security Insights Settings

Options

Collaborators

Branches

Webhooks

Notifications

Integrations & services

Deploy keys

Repository name

myfirstrepo Rename

Template repository
Template repositories let users generate new repositories with the same directory structure and files. Indicate if `jstbcs/myfirstrepo` can be used as a template for creating other repositories.

Social preview

Github

Settings

The screenshot shows the GitHub repository settings page for 'jstbcs / myfirstrepo'. The repository is private, with 1 watch, 0 stars, and 0 forks. The 'Settings' tab is selected. On the left, a sidebar lists options: Collaborators (selected), Branches, Webhooks, Notifications, Integrations & services, and Deploy keys. The main content area is titled 'Collaborators' and includes a note: 'This repository doesn't have any collaborators yet. Use the form below to add a collaborator.' A search bar contains the text 'alexandra sarafo...'. Below the search bar, a result for 'ASarafoglou Alexandra Sarafoglou' is shown, with an 'Add collaborator' button next to it.

Github

Settings

Danger Zone

Make this repository public

Make this repository visible to anyone.

[Make public](#)

Transfer ownership

Transfer this repository to another user or to an organization where you have the ability to create repositories.

[Transfer](#)

Archive this repository

Mark this repository as archived and read-only.

[Archive this repository](#)

Delete this repository

Once you delete a repository, there is no going back. Please be certain.

[Delete this repository](#)

Github

Clone It!

The screenshot shows a GitHub repository page for 'jstbcs / myfirstrepo'. The repository is private, has 1 commit, 1 branch, and 0 releases. It features a README.md file with the text 'This is my first repository. I will try out some stuff here.' A context menu is open over the repository name, displaying options to 'Clone with HTTPS' or 'Use SSH', and a link to 'https://github.com/jstbcs/myfirstrepo'. There is also a 'Download ZIP' option.

jstbcs / **myfirstrepo** Private

Code Issues 0 Pull requests 0 Projects 0 Security Insights Settings

This is my first repository. I will try out some stuff here. Edit

Manage topics

1 commit 1 branch 0 releases

Branch: master New pull request Create new file Upload files Find file Clone or download

jstbcs Initial commit README.md Initial commit README.md

myfirstrepo

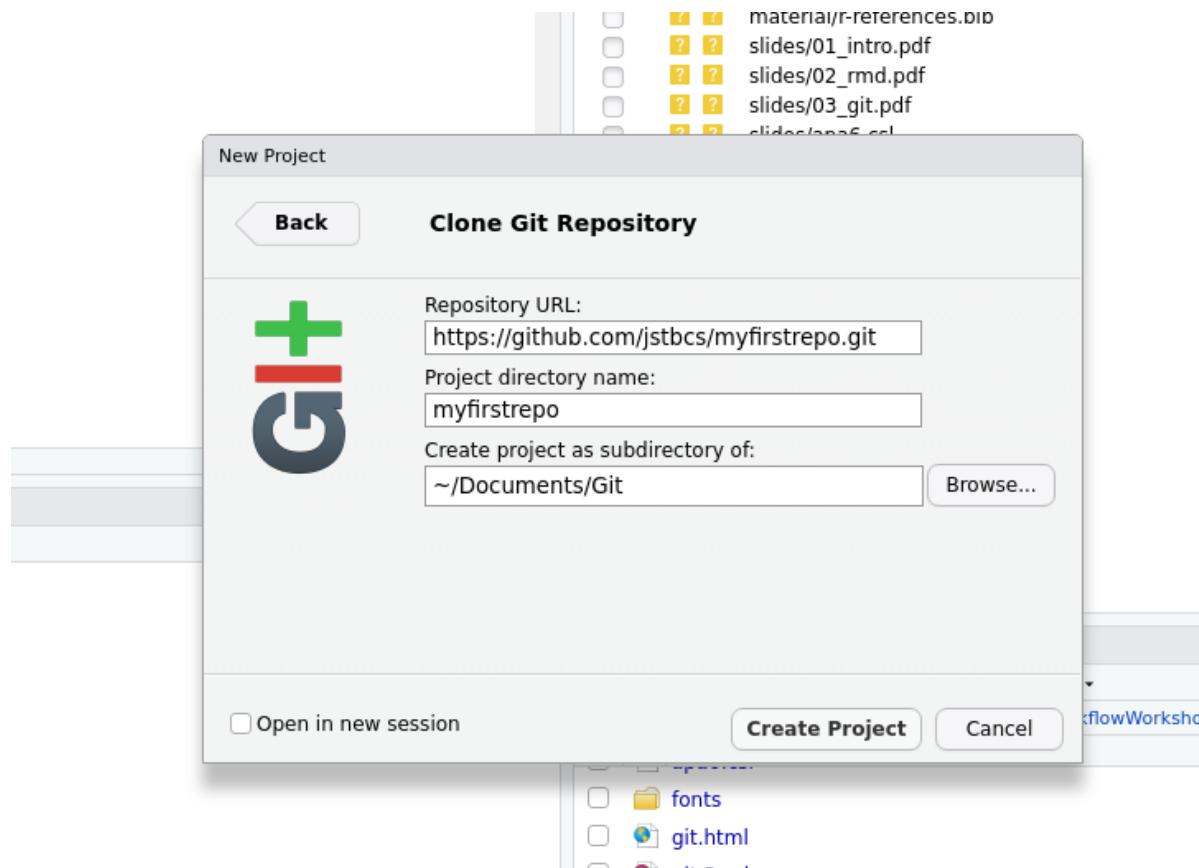
This is my first repository. I will try out some stuff here.

Clone with HTTPS Use SSH
Use Git or checkout with SVN using the web URL.
<https://github.com/jstbcs/myfirstrepo>

Download ZIP

In R Studio

File ➤ New Project ➤ Version Control ➤ Git



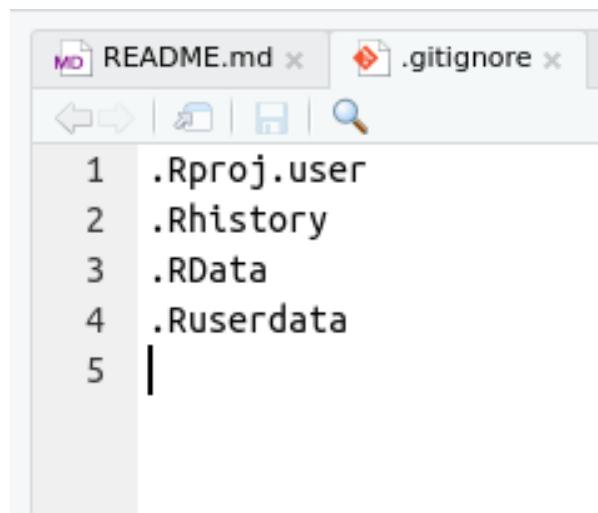
In R Studio

- Initializes a local git repository with an R project (opening a clean R Studio session when opening).
- You can see the README file from github.
- Adds a `.gitignore` file.



gitignore

- Specifies intentionally untracked files to ignore.
- Each line in a `gitignore` file specifies a pattern.
- R Studio pre-specifies some useful patterns.
- For R Markdown: Cache files! `.tiff`, `.eps`, `.rdb`, `.rdx`
- Pro tip: `?usethis::git_vaccinate()`



```
1 .Rproj.user
2 .Rhistory
3 .RData
4 .Ruserdata
5 |
```

README



The screenshot shows a code editor window with a light gray background. At the top left, there is a small icon followed by the text "README.md". At the top right, there is a small pencil icon. The main content area contains the following text:

Git and Rmarkdown: Two Useful Tools for a Fully Reproducible Workflow

This repository contains materials and slides for a workshop organized by the Open Science Community Amsterdam. It is meant as a short (two hour) teaser for different tools that can be used for a fully reproducible workflow in science.

Licence: cc by-sa-nc 4.0

Author: Julia Haaf

Abstract

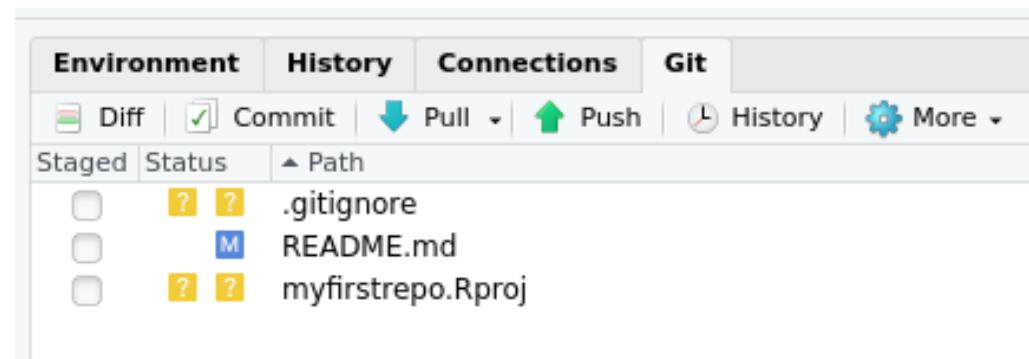
In the debate around the reproducibility crisis, most attention is paid to instances when researchers either trick themselves or others. Recently, however, some of the discussion has also shifted to scientific and reporting mistakes sneaking into the literature without anyone noticing. Whether these are mistakes in reporting statistics (Nuijten et al., 2016) or mistakes in the data collection process (Rouder et al., 2019), they prevent computational and empirical reproducibility of results in the literature. Rouder and colleagues (2019) proposed standardization and computer automation to minimize mistakes in the literature. In this workshop I will give an introduction to two tools we used for automation in the Perception and Cognition Lab at the University of Missouri. Git is a version control system that can be used to publish data or to collaboratively work

README

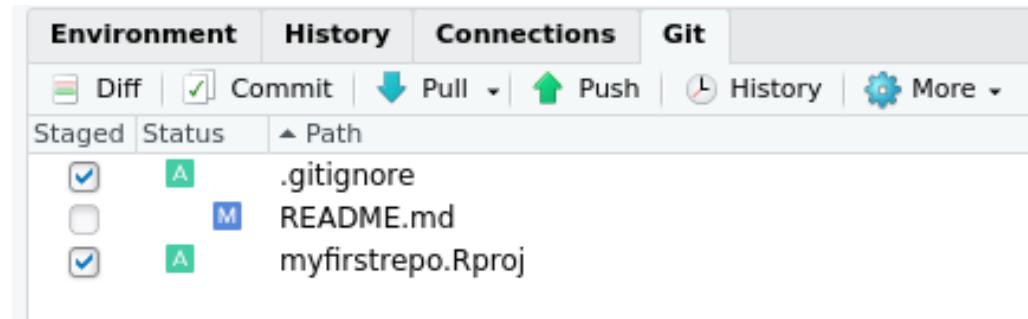
- Tell other people (and yourself in a year) why your project is useful, what they can do with your project, and how they can use it.
- On github default README files are Markdown files!

Git Workflow

Do some work



Git Add



```
git add .gitignore myfirstrepo.Rproj
```

- git can do auto-complete for file names!
- Note that many user interfaces combine git add and git commit (next step).

Git Commit



```
git add .gitignore myfirstrepo.Rproj  
git commit -am "My first commit"
```

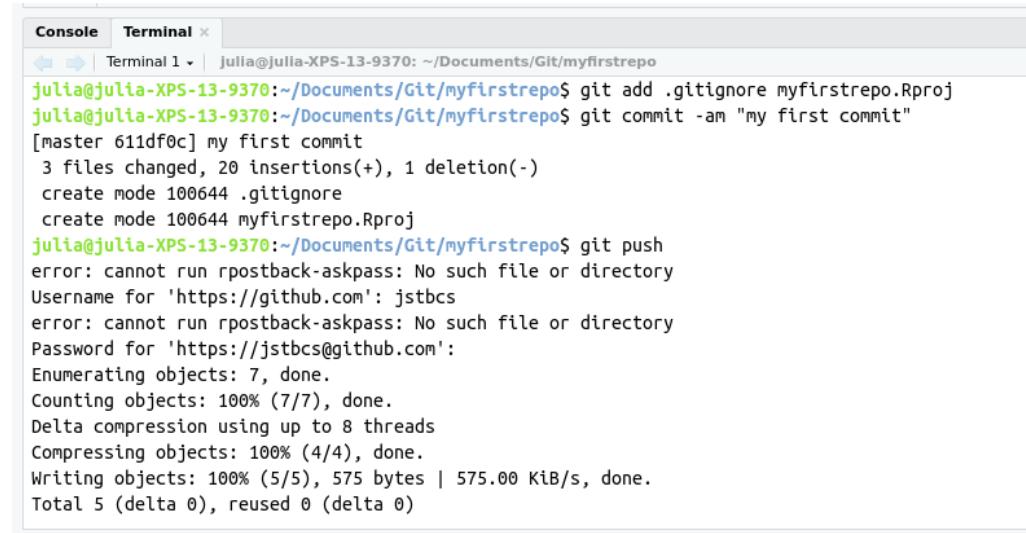
Commits always have a commit message.

Commit message

	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
○	AAAAAAA	3 HOURS AGO
○	ADKFJSLKDFJSOKLFJ	3 HOURS AGO
○	MY HANDS ARE TYPING WORDS	2 HOURS AGO
○	HAAAAAAAAANDS	2 HOURS AGO

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

Git Push



The screenshot shows a terminal window with the title bar "Console Terminal x". The terminal is running on a Windows system (XPS-13-9370). The command history and output are as follows:

```
julia@julia-XPS-13-9370:~/Documents/Git/myfirstrepo$ git add .gitignore myfirstrepo.Rproj
julia@julia-XPS-13-9370:~/Documents/Git/myfirstrepo$ git commit -am "my first commit"
[master 611df0c] my first commit
 3 files changed, 20 insertions(+), 1 deletion(-)
  create mode 100644 .gitignore
  create mode 100644 myfirstrepo.Rproj
julia@julia-XPS-13-9370:~/Documents/Git/myfirstrepo$ git push
error: cannot run rpostback-askpass: No such file or directory
Username for 'https://github.com': jstbcs
error: cannot run rpostback-askpass: No such file or directory
Password for 'https://jstbcs@github.com':
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 575 bytes | 575.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0)
```

```
git add .gitignore myfirstrepo.Rproj
git commit -am "My first commit"
git push
```

Congrats! You have done it! Now local and remote repositories are up to date!

Git Pull

Before you start working on the project the next time (either in terminal or within R Studio):

```
git pull
```

Pull, work some more, repeat.

What changed since the last commit?

git diff

RStudio: Review Changes

Changes History master Stage Revert Ignore Pull Push

Commit message

Amend previous commit Commit

Show Staged Unstaged Context 5 line Ignore Whitespace Stage All Discard All

```
91 91 **Version control:**  
92 92  
93 93 >- Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.  
94 94 >- History of all changes (who, what, when).  
95 95 >- Helps to avoid mistakes (working on the wrong version, deleting, ...).  
96 96 >- Merging changes of multiple collaborators in one file  
97 97  
98 98   
99 99  
100 100 ## What is it?  
101 101  
102 102   
103 103  
104 104 ## What can it do?  
105 105  
106 106 >- A lot! Which is why we only need a part of its functionality.  
107 107 >- Working on one product in (large) teams.  
108 108 >- Working on things that can break.  
109 109 >- git can only integrate and show changes in text files.  
110 110 >- binary files (images, etc.) can be tracked and uploaded but changes cannot be shown.  
111 111 >- Track changes for MS Word is improving  
112 112 >- A lot! Which is why we only need a part of its functionality.
```

What happens if something goes wrong? (And it will.)



What happens if something goes wrong? (And it will.)

First things first: You cannot break things.

What happens if something goes wrong? (And it will.)

Read the output!

```
remote:  Target folder: /home/julia/pages
remote:  Tag name      : release_20200831-1430
remote:
remote: =====
remote:
To ntzwrk.net:pages.git
  1662d5e..d1c3203  master -> master
julia@julia-XPS-13-9370:~/Documents/Git/pages$ git pult
git: 'pult' is not a git command. See 'git --help'.

The most similar command is
  pull
julia@julia-XPS-13-9370:~/Documents/Git/pages$
```

What

Push Conflict

```
Console Terminal x
Terminal 1 | julia@julia-XPS-13-9370: ~/Documents/Git/myfirstrepo
julia@julia-XPS-13-9370:~/Documents/Git/myfirstrepo$ git commit -m "happy commit"
[master 520bc1a] happy commit
 1 file changed, 2 insertions(+)
julia@julia-XPS-13-9370:~/Documents/Git/myfirstrepo$ git push
error: cannot run rpostback-askpass: No such file or directory
Username for 'https://github.com': jstbcs
error: cannot run rpostback-askpass: No such file or directory
Password for 'https://jstbcs@github.com':
To https://github.com/jstbcs/myfirstrepo.git
 ! [rejected]      master -> master (fetch first)
error: failed to push some refs to 'https://github.com/jstbcs/myfirstrepo.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
julia@julia-XPS-13-9370:~/Documents/Git/myfirstrepo$ █
```

Merge Conflict

```
Console Terminal x
Terminal 1 | julia@julia-XPS-13-9370: ~/Documents/Git/myfirstrepo
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
julia@julia-XPS-13-9370:~/Documents/Git/myfirstrepo$ git pull
error: cannot run rpostback-askpass: No such file or directory
Username for 'https://github.com': jstbcs
error: cannot run rpostback-askpass: No such file or directory
Password for 'https://jstbcs@github.com':
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/jstbcs/myfirstrepo
  611df0c..4ba4682  master      -> origin/master
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
julia@julia-XPS-13-9370:~/Documents/Git/myfirstrepo$ █
```

Merge Conflict



The screenshot shows a code editor interface with two tabs: 'README.md' and '.gitignore'. The 'README.md' tab is active, displaying the following content:

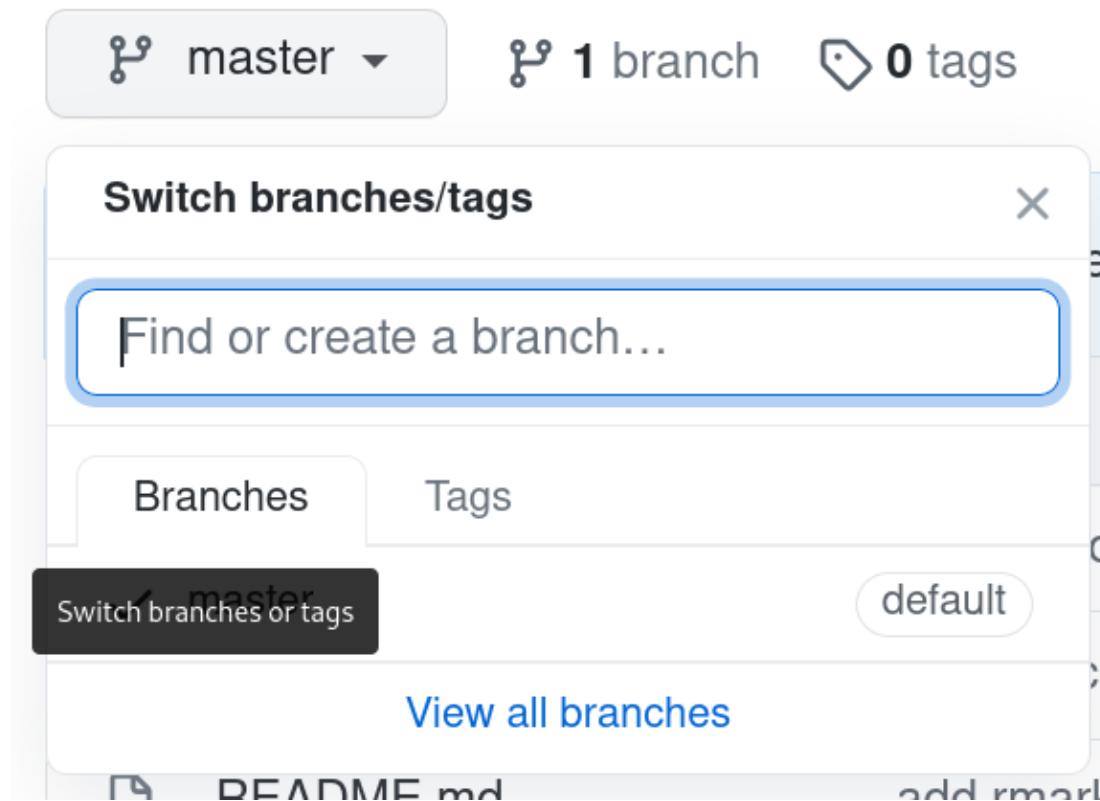
```
1 # myfirstrepo
2
3
4 This is my first repo. You cannot do anything with it.
5
6 <<<<< HEAD
7 What happens if I do this?
8 =====
9 Here I want a merge conflict.
10 >>>> 4ba46823bce01a104a7aab219043d92f357f2898
11
```

The text is color-coded: '# myfirstrepo' is blue, 'This is my first repo.' is black, 'What happens if I do this?' is black, 'Here I want a merge conflict.' is black, and the commit hash '4ba46823bce01a104a7aab219043d92f357f2898' is green. The merge conflict markers ('<<<<< HEAD' and '>>>>') are also color-coded.

- Resolve the conflict (Choose which changes to keep).
- Commit, and Push.

How to avoid merge conflicts

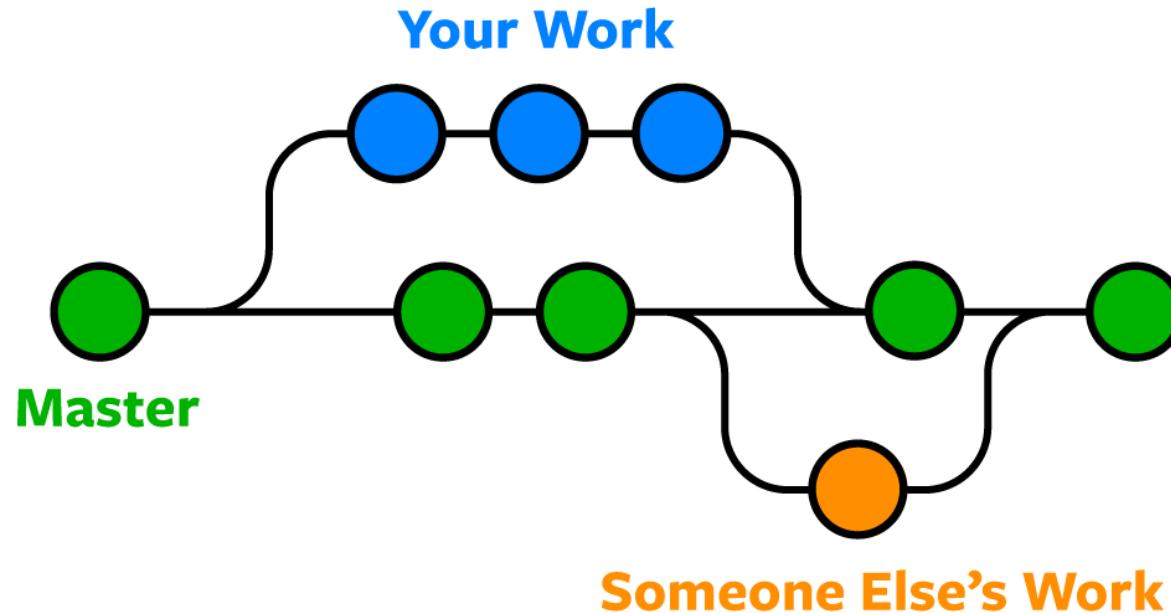
Or: Do not mess with the master



Do not mess with the master

- The master branch is *deployable*: It contains the final product code.
- This is not where people work.
- Instead you use *branches* for development and then *merge* the code into the master branch once you are done.

Do not mess with the master



Do not mess with the master

I don't really have time to get into this

- Here is the best [tutorial](#) I could find on branches: <https://thenewstack.io/dont-mess-with-the-master-working-with-branches-in-git-and-github/>.
- For working in a professional environment: A tutorial on pull requests:
<https://www.atlassian.com/git/tutorials/making-a-pull-request>

Summary

- Add, commit, push, pull.
- Use it!
- git documentation and error tracking are great!

Your turn!

- Complete the in-class assignment.



References

Additional info (in the [material folder](#)):

Vuorre, M., & Curley, J. P. (2018). Curating Research Assets: A Tutorial on the Git Version Control System. *Advances in Methods and Practices in Psychological Science*, 1(2), 219–236.