



Reproducible Workflow

Git

Julia Haaf

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

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.)

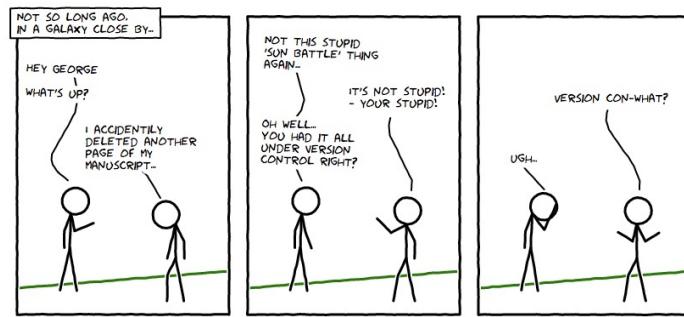
What we have time for

- *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.)

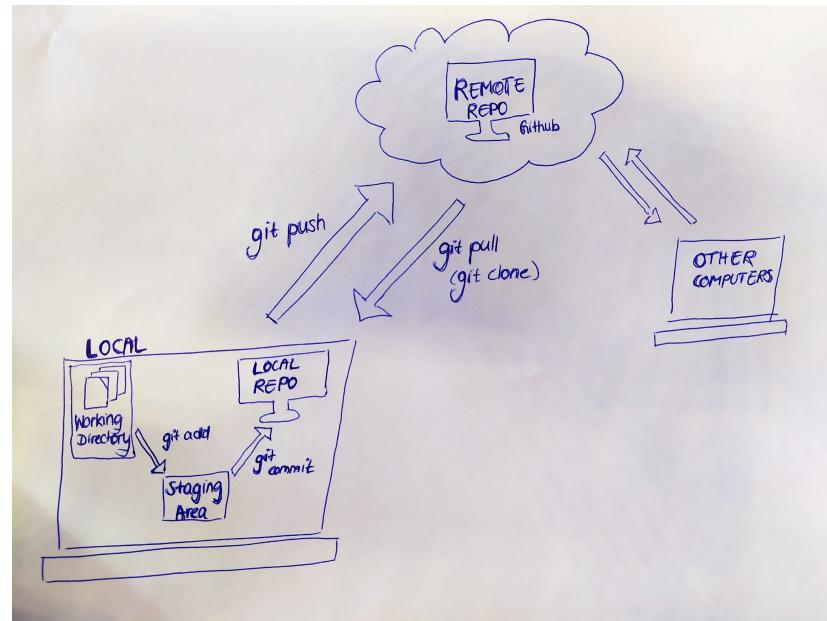
What is it good for?

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?

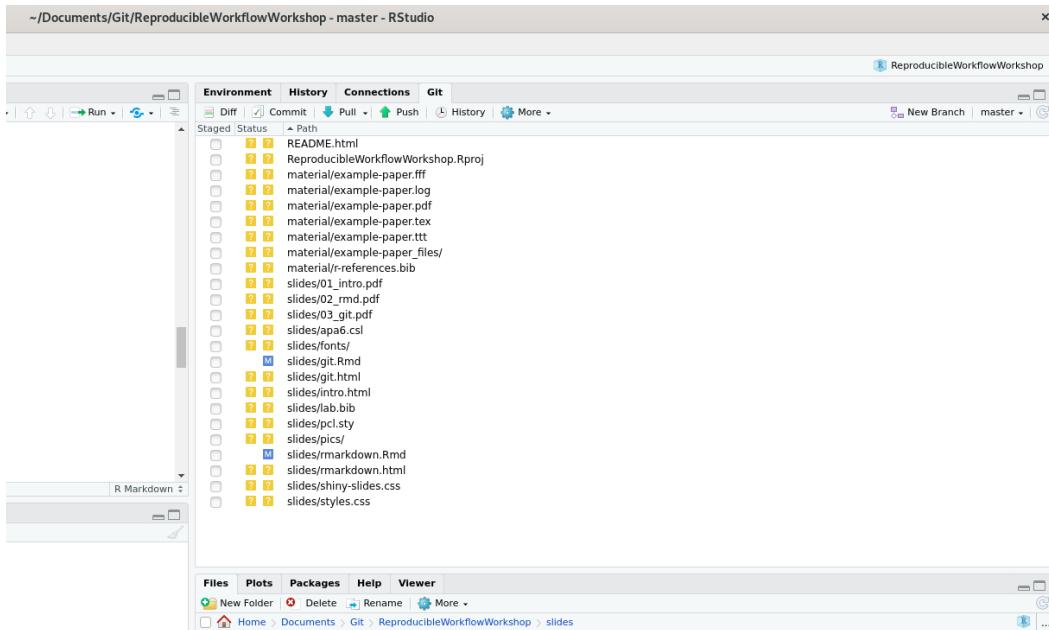


What can it do?

- A lot! Which is why we only need a part of its functionality.
- 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.
- Track changes for MS Word is improving.

Set-up for your computer

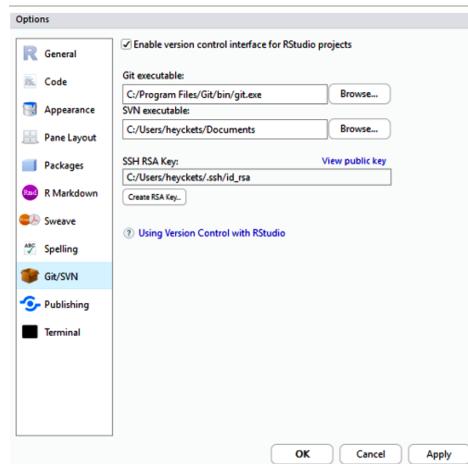
R Studio & git



R Studio & git

Tools ► Global Options ► Git/SVN.

Make sure the first box is ticked and the “git.exe” (Windows) 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"
```

My first repo

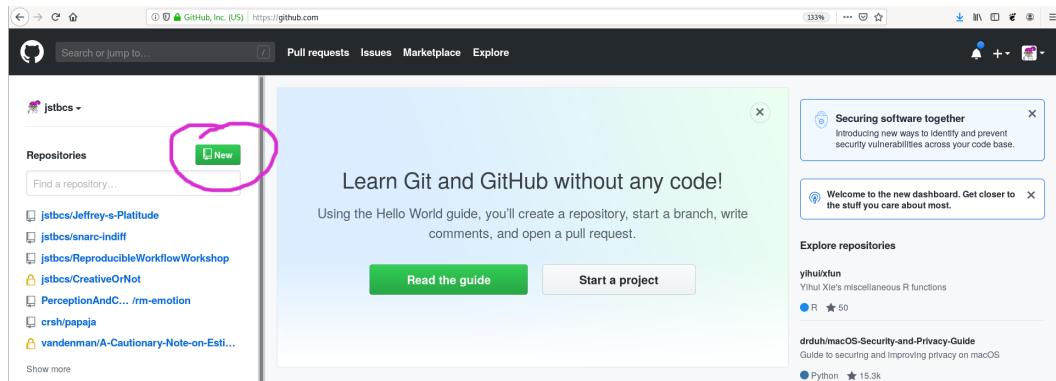
Github

The screenshot shows the GitHub homepage with the new dashboard interface. On the left, there's a sidebar with 'Repositories' (including 'jstbcs/Jeffrey-s-Platitude', 'jstbcs/snarc-indiff', 'jstbcs/ReproducibleWorkflowWorkshop', 'jstbcs/CreativeOrNot', 'PerceptionAndC...', 'crsh/papaja', and 'vandenman/A-Cautionary-Note-on-Esli...'), 'Your teams' (including 'jasp-stats/internal-testers', 'jasp-stats/issue-collaborators', 'PerceptionAndCognitionLab/lab', and 'methexp/owners'), and a 'Search or jump to...' bar. The main area features a 'Learn Git and GitHub without any code!' section with 'Read the guide' and 'Start a project' buttons. It also displays recent activity: 'mariusbarth pushed to methexp/rawdata 4 hours ago' (1 commit to master) and 'mariusbarth pushed to methexp/g-sync-test 16 hours ago' (4 commits to master). A sidebar on the right shows 'Explore repositories' with projects like 'yihui/xfun', 'drduh/macOS-Security-and-Privacy-Guide', and 'wch/extrafont'. Top navigation includes 'Pull requests', 'Issues', 'Marketplace', and 'Explore'.

14/44

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 /

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.

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

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 / 

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.

| 



Github

Settings

The screenshot shows the 'Settings' page for a GitHub repository named 'myfirstrepo'. The repository is private, as indicated by the 'Private' link in the top right. The main navigation bar includes links for Code, Issues (0), Pull requests (0), Projects (0), Security, Insights, and Settings (which is the active tab). On the left, a sidebar menu lists Options, Collaborators, Branches, Webhooks, Notifications, Integrations & services, and Deploy keys. The 'Options' item is currently selected. The main content area is titled 'Settings' and contains a 'Repository name' field with 'myfirstrepo' entered, and a 'Rename' button. Below this is a 'Template repository' section with a checkbox labeled 'Template repository' and a descriptive text explaining its purpose. A 'Social preview' section is also present at the bottom.

18/44

Github

Settings

The screenshot shows the GitHub repository settings page for 'myfirstrepo'. The top navigation bar includes links for 'Code', 'Issues 0', 'Pull requests 0', 'Projects 0', 'Security', 'Insights', and 'Settings'. The 'Settings' tab is active, indicated by an orange underline. On the left, a sidebar menu lists 'Options', 'Collaborators' (which is selected and highlighted in orange), 'Branches', 'Webhooks', 'Notifications', 'Integrations & services', and 'Deploy keys'. The main content area is titled 'Collaborators' and contains a sub-header 'Push access to the repository'. It states, 'This repository doesn't have any collaborators yet. Use the form below to add a collaborator.' Below this is a search bar labeled 'Search by username, full name or email address' with the placeholder 'You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.' A search result for 'alexandra sarafoglu' is shown, with a small profile picture, the name 'ASarafoglu', and the full name 'Alexandra Sarafoglu'. To the right of the search result is a blue button labeled 'Add collaborator'.

19/44

Github

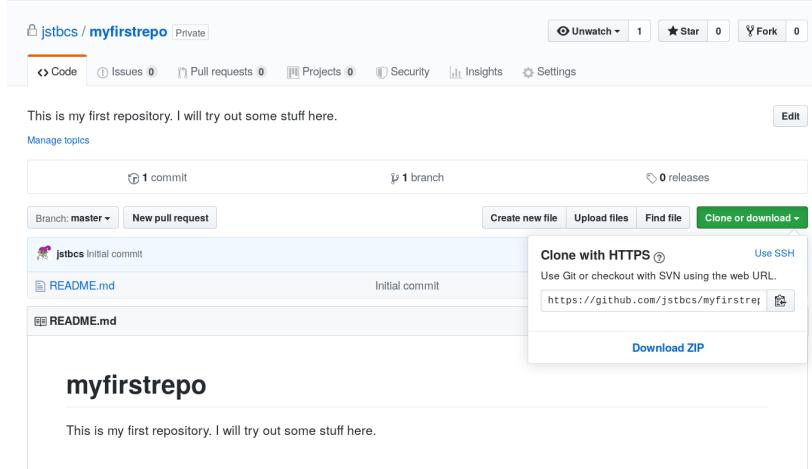
Settings

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

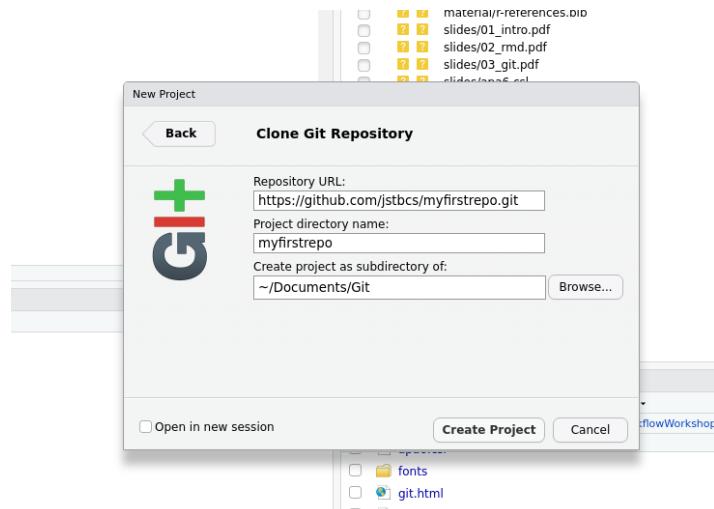
Clone It!



21/44

In R Studio

File > New Project > Version Control > Git



In R Studio

- You will have to type in your user name and password for github.
- 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.



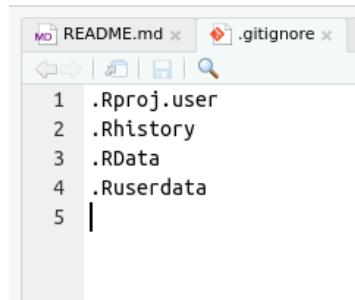
Task IV

- Make a new repository.
- Clone it using R Studio to make a local repository.



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`



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

README

The screenshot shows a window titled "README.md" with a dark header bar. The main content area has a light gray background. At the top, there is a title section:

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

Below the title, there is a short paragraph of text:

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.

Underneath the text, there are two lines of smaller text:

Licence: cc-by-nc 4.0
Author: Julia Haaf

Following this, there is a section header:

Abstract

Finally, there is a large block of descriptive text:

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!

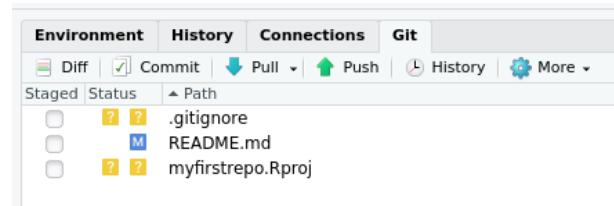
Task V

- Write a (short) README file for your test repository.
- Use Markdown formatting.

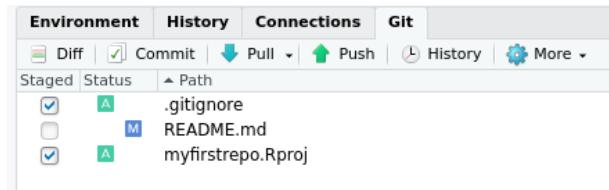


Git Workflow

Do some work



Git Add



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

git can do autocomplete for file names!

Git Commit



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

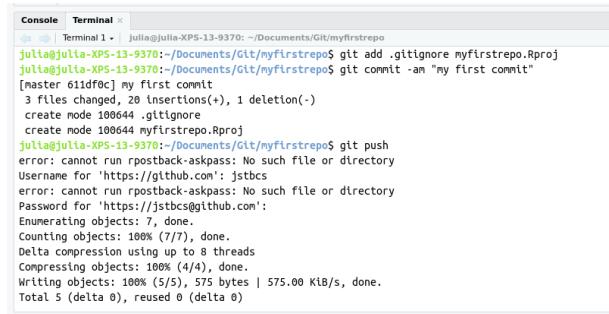
Commits always have a commit message.

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 following command history:

```
Console Terminal 1 - julia@julia-XPS-13-9370: ~/Documents/Git/myfirstrepo
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!

Task VI

- Add your unstages files.
- Commit the changes.
- Push to the remote repository.



Git Pull

Before you start working on the project the next time:

```
git pull
```

Pull, work some more, repeat.

What changed since the last commit?

git diff

The screenshot shows the RStudio: Review Changes interface. The left pane displays a tree view of files under the 'Changes' tab, with 'slides/git.Rmd' selected. The right pane shows the 'Commit message' field and a 'Commit' button. Below these are buttons for 'Pull' and 'Push'. The main area contains the git diff output:

```
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.  
113 113 - Working on one product in (large) teams.  
114 114 - Working on things that can break.  
115 115 - git can only integrate and show changes in text files.
```

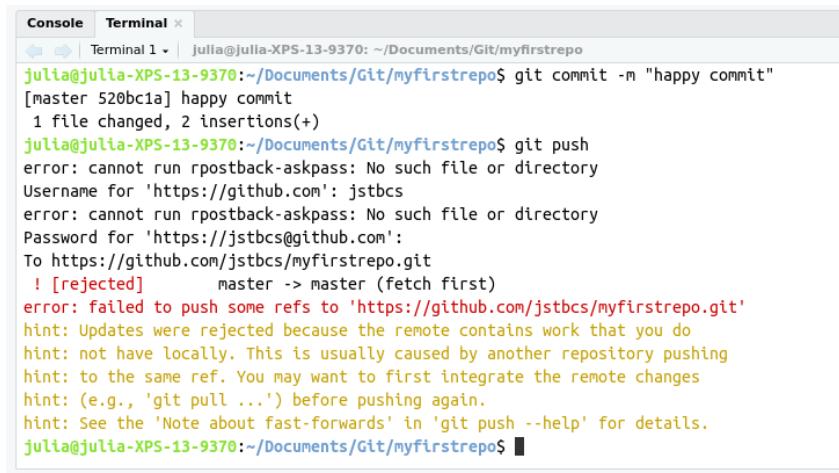
What happens if something goes wrong? (And it will.)



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

- Remember: You cannot break things.
- Most likely you have a merge conflict.

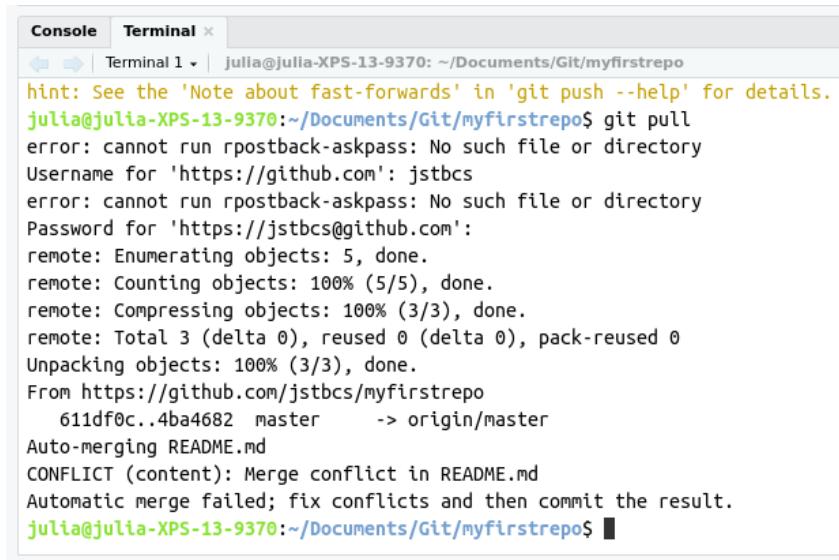
Push Conflict



The screenshot shows a terminal window with two tabs: "Console" and "Terminal". The "Terminal" tab is active, showing the following command-line session:

```
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$
```

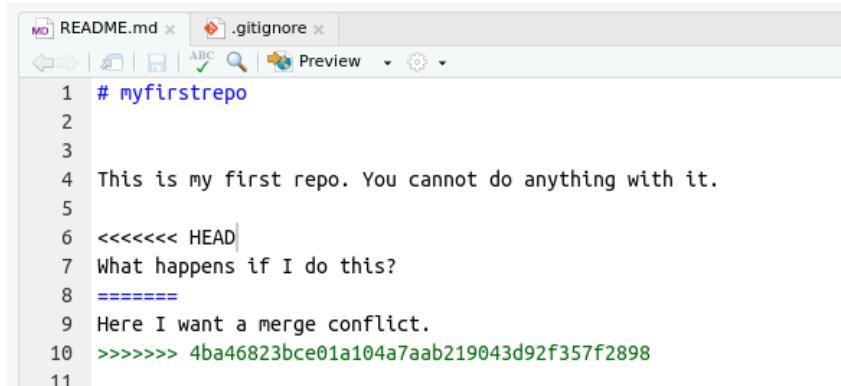
Merge Conflict



The screenshot shows a terminal window titled "Terminal 1" with the command "git pull" executed. The output indicates a merge conflict in the "README.md" file.

```
Console Terminal ×
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 window with two tabs: 'README.md' and '.gitignore'. The 'README.md' tab is active and displays 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 'What happens if I do this?' is preceded by '<<<<< HEAD' and followed by '>>>> 4ba46823bce01a104a7aab219043d92f357f2898', indicating a merge conflict.

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

Summary

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

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.