R projects and version control using GitHub

Krista Kraskura

Contents

Today we will:

1. talk about 'why GitHub'?
2. establish communication between RStudio and GitHub
3. setup a workflow working with R projects and Github
    - from existing local directory
    - from Github account that are linked with Github.
    - add README for each repo

Why GitHub?

- version control
- sharing data and code
- tractability and reproducibility
- enables collaboration
- it's free to host unlimited public repositories and private repositories (but with free plan private repos allows up to three external collaborators)

Setup pre-requisites:

1. GitHub account (github.com)
2. Installed git on the computer
3. Upgrade your R or RStudio

Then we get to create a workflow.

Do we have git installed?

Newer computers come with git pre-installed. To check if git is installed, go to RStudio, find Terminal (likely next to the Console), open it and type in:

```
# in Terminal:

which git
## /usr/bin/git
```

If git is not installed, the messages will look different, and command `git` will not be found.

No git? No Problem! Let's follow online guide to install it.

NOTE: When installing git for **Windows**:

On prompt: "Adjusting your PATH environment" –> select "Git from the command line and also from 3rd-party software"

Local location: `C:/Program Files/Git/bin/git.exe`

NOTE: When installing git for **Macs**:

Use terminal commands to install it. Install the Xcode line tools with git.

```
# option 1:
git --version
git config

# alternative
xcode-select --install
```

Do we have the latest version of git?

```
# check
git --version
# git version 2.43.0

# if needed: update (for Wondows)
git update git-for-windows

# for macs
```

Do we have the latest version of R?

Let's check:

```
# in RStudio Console:

R.version.string
#> [1] "R version 4.3.2 (2023-10-31)"
```

Staying up to date with RStudio and R updates can save a lot of headache in the long run.

Making connections

Make your git yours (done ONCE)

```r
# in RStudio Console:

# install package if needed. need to do this only once
# install.packages("usethis")
library(usethis)

use_git_config(user.name = "Firstnames Lastname",
               user.email = "emailaddress")
```

Make RStudio and Github talk

We can do this using personal access tokens (PAT)

```r
# in RStudio Console:

usethis::create_github_token()
```

We should be in a pop-up website window (on GitHub).

Click "Generate token". Copy PAT and save it (leave the site open while working, in case if something gets lost).

Now:

```r
# in RStudio Console:

gitcreds::gitcreds_set()

# ? Enter password or token: ghp_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# -> Adding new credentials...
# -> Removing credentials from cache...
# -> Done.
```

p.s. The token can also be generated by visiting https://github.com/settings/tokens. Click "Generate token". Select: "repo", "user", and "workflow".

Success?!

Creating workflow: in RStudio

First.

- Open RStudio
- Click File
- Click New Project
- Click Version Control
- Click Git
- Paste the remote repo URL and enter TAB

- Click Create Project

Then.

- Create a new file or make changes in README.md
- Find the **Git tab**
- Click Commit
- In the Review changes view, check the staged box for all files.
- Add a **commit message**
- Click **Commit**.
- Click the **Pull** button to incorporate any remote changes.
- Click the **Push** button to push your changes to the remote repository.
- Go on Github and check out the changes

Creating workflow using Terminal

First, make repository (repo) on GitHub

- Repository Template: 'no template'
- Repo Name: *something short and meaningful*
- Enter a description for your repository
- Visibility: 'Public'
- Select Initialize this repository with a README.
- Click Add .ignore and select R.
- Click Create repository.

Then, clone it to your computer locally and connect with RStudio.

- On GitHub, find the 'Code' tab.
- Click Clone
- Select HTTPS (assuming its right)
- Copy the link

```
# In Terminal:

# to see the current wd (important, keep repos tidy together)
pwd

# to change wd
cd [ENTER HERE]

# clone the repo
# git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY.git
git clone https://github.com/kraskura/newrepo.git

# change my wd to my new repo; assuming my repo is called
cd newrepo

# get info about the RStudio and GitHub connection
git remote show origin
```

```
# make a change in README
# ...

# check changes
git status

# add changes to local repo memory
git add --all
git add README.md

# commit added changes! -m initiated messages, use them
git commit -m "commit messages help me keep track"

# finally push the changes to the remote repo
git push
```

Asked for password? for authentication? Update everything as prompted. **the password is your PAT**

No Git pane? maybe its not a git repo. Let's check.

```
# In Terminal
git status

# fatal: not a git repository (or any of the parent directories): .git
```

Some tips and final notes from experience:

- if possible, avoid pushing very large files to remote repo, can use `.gitignore.` Consider using platforms like dryad to make your data open source, publicly accessible.
- create using GitHub a habit
  - every time we work on repo, try to push all recently made changes
  - practice throughout the semester
- there is lots of info out there, rely on community, use google
- note: we only covered the essentials here, there is so much more to using github and git
  - For example, interested in making a website? Github will host one for every user!

Resources

- **Happy Git and GitHub the useR** an amazing online resource that inspired the content presented herein. It is easy to follow. Some particularly relevant and helpful sections:
  - installing git
  - Some important troubleshooting, git disapears from RStudio? dont know where git is?