How to Git with RStudio

Part I

Why Git?

Git is a **version control system**.

Its original purpose was to help groups of developers work collaboratively on big software projects.

Git manages the evolution of a set of files – called a **repository** – in a sane, highly structured way.

In addition to using it for source code, we use it to manage the motley collection of files that make up typical data analytical projects, which often consist of data, figures, reports, and, yes, source code.

Why GitHub?

Hosting services like GitHub, Bitbucket, and GitLab provide a home for your Git-based projects on the internet.

Think of it as DropBox but much, much better.

The remote host acts as a distribution channel or clearinghouse for your Git-managed project.

It allows other people to see your stuff, sync up with you, and perhaps even make changes.

What is the Payoff?

Exposure

If someone needs to see your work or if you want them to try out your code, they can easily get it from GitHub.

Exposure

If they use Git, they can clone or fork your repository.

Exposure

If they don't use Git, they can still browse your project on GitHub like a normal website and even grab everything by downloading a zip archive.

Collaboration

If you care about someone else's project, such as an R package you use heavily, you can track its development on GitHub.

Collaboration

You can fork it to keep your own copy.

Collaboration

You can modify your fork to add features or fix bugs and send them back to the owner as a proposed change.

Register a GitHub Account

Register an account with GitHub.

Username Advice

Avoid the use of upper vs. lower case to separate words.

Username Advice

A better strategy for word separation is to use a hyphen.

Install R and RStudio

Install a pre-compiled binary of R for your OS.

Install RStudio Desktop.

Install Git

You need Git, so you can use it at the command line and so RStudio can call it.

Install Git for Windows.

When asked about *Adjusting your PATH environment*, make sure to select

Git from the command line and also from 3rd-party software.

Introduce yourself to Git

Click the **Git** menu in the Windows menu and select **Git Bash**.

```
git config --global user.name 'John Doe'
git config --global user.email 'john.doe@example.com'
git config --global --list
```

Configure the Git Editor

```
git config --global core.editor "emacs"
```

Install a Git Client

GitKraken is a free, powerful Git(Hub) client.

GitHub offers a free Git(Hub) client, GitHub Desktop, for Windows.

Personal access token for HTTPS

When we interact with a remote Git server, such as GitHub, we have to include credentials in the request.

Git can communicate with a remote server using one of two protocols, HTTPS or SSH, and the different protocols use different credentials.

With HTTPS, we will use a **personal access token (PAT)**.

Go to **https://github.com/settings/tokens**

and click **Generate token**.

Copy the generated PAT to your clipboard.

Or leave that browser window open and available, so you can come back to copy the PAT.

Provide this PAT next time a Git operation asks for your password.

You should be able to work with GitHub now, i.e. push and pull.

Connect to GitHub

On your profile page, click on **Repositories**,

then click the big green **New** button.

Repository Template: **No template**

Repository Name: **firstrepo**

Description:

"This is a test repository for my Git/GitHub setup."

Public.

Initialize this repository with:

Add a README file.

Click the green button that says

Create repository.

Now click the green button that says

<> **Code**.

Copy a clone URL to your clipboard.

Clone the repo to your local computer

Go to the shell.

Take charge of what directory you're in.

`pwd` displays the working directory.

`cd` is the command to change directory.

```
cd ~/Desktop/temp
```

Clone `firstrepo` from GitHub to your computer.

```
git clone
  https://github.com/gabriel-msun-nanotox/firstrepo.git
```

```
Cloning into 'firstrepo'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0),
  pack-reused 0
Receiving objects: 100% (3/3), done.
```

Make this new repo your working directory, list its files, display the README, and get some information on its connection to GitHub.

```
cd myrepo
ls
head README.md
git remote show origin
```

```
* remote origin
  Fetch URL: https://github.com/gabriel-msun-nanotox/firstrepo.git
  Push  URL: https://github.com/gabriel-msun-nanotox/firstrepo.git
  HEAD branch: main
  Remote branch:
    main tracked
  Local branch configured for 'git pull':
    main merges with remote main
  Local ref configured for 'git push':
    main pushes to main (up to date)
```

Make a local change, commit, and push

Add a line to README and verify that Git notices the change.

```
echo "A new line written with my local computer."
  >> README.md
```

```
git status
```

```
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```

Stage (“add”) and commit this change and push to your remote repo on GitHub.

```
git add README.md
git commit -m "A commit from my local computer"
git push
```

```
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 358 bytes | 358.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/gabriel-msun-nanotox/firstrepo.git
   e9e4d22..fa34122  main -> main
```

Go back to the browser.

Refresh.

You should see the new “A new line written with my local computer.” in the README.

If you click on **commits**, you should see one with the message

"A commit from my local computer."

Clean up

Local

You can delete the local repo any way you like.

It's just a regular directory on your computer.

Local

```
cd ..
rm -rf myrepo/
```

GitHub

In the browser, go to your repo's landing page on GitHub.

GitHub

Click on **Settings**.

GitHub

Scroll down,

click on **delete repository**,

and do as it asks.

Next Lesson

Next Lesson

Connect RStudio to Git and GitHub

Learn More

Reference

Happy Git and GitHub for the useR

by Jennifer Bryan