Git and GitHub with R

Shannon B. Hagerty

7/10/2019

Pre-Session Set Up

This week we're going to be talking about reproducibility & version control, and learning how to use git and directly send and receive files from GitHub to Rstudio. To get ready for the session you need to create a GitHub Account and to download git to your computer.

1. If you do not have an account already sign up for GitHub online **Note: you don't need to specify organization right now.**
2. Download Git (see step 3 if you think you already have it): you can We're going to follow the instructions given by the Software Carpentries, they also provide a video of how to do this install (you can ignore the part about installing a text editor called nano)
   a. For Windows: Look at the section called Bash and use those install directions for Windows. Pay close attention to the instructions as at some points you need to deviate from the default install settings (sometimes your install guide will have additional questions beyond what is mentioned in The Carpentry instructions you can leave these at default and hit the button to move to the next step). Also Ignore setting the home environment.
   b. For Macs: Follow the Mac directions under the Git set up
3. Confirm git is installed and ready to work with Rstudio: go to Rstudio (it needs to be restarted if it was open during the git install) and in the bottom left pane where the console is, there is also a tab that says terminal. Click that and after the dollar type the word git then hit enter. If you get a big long chunk of text about common git commands, you're all set. If you get a response saying command not found, send me an email.

Git and GitHub with R

What is Git? Git is a version control system, used to track changes for files on your computer. It is very popular of keeping track of development of code files over time, and allowing you to restore previous versions of the file if needed.

What is GitHub? GitHub is a website that hosts your Git files. There are other options to host your Git files, but GitHub is most widely used.

Why should you use version control with Git and GitHub?

- Inevitably you're going to encounter a situation where you have a script working and later come back to it make a few *small* changes and break it or run it and get unexplainable differences. You could just save each version separately on your computer but that can get so messy (**Personal anecdote** I have several files on my computer with names like StatsTest.R, StatsTest-working.R, StatsTest-reallyworking.R, StatsTest-reallyworking5-2.R, I still have not found the working version of the script. I wish I had used version control when I was writing code)
- It's also possible you may want to share your code publicly, this may be part of your review process or may be something you make available for other researchers to use in their own work or to reproduce yours. GitHub is one of the main places people publicly share their code. It also allows you to keep your code in private repositories if you choose.
- Git/GitHub is also a really useful tool to collaborate with others.

Introduce yourself to Git

```
install.packages('usethis')
use_git_config(user.name= "YourGitHubUsername", user.email="YourEmail@email.com")
```

Get a copy of the BrandeisLibrarySummeR repo onto your own computer

Repository: Is basically a folder that has a collection of your git-tracked files *Cloning:* You take a GitHub repository and make a copy of it on your local drive.
- To Clone a Repository using Rstudio -> File -> New Project -> Version Control -> Git

```
- We're going to clone the SummeR of R Repo:
**url:** https://github.com/sbhagerty/BrandeisLibrarySummeR.git
**Project directory name:** BrandeisLibrarySummeR
**Create project as subdirectory of:** Use the Browse button to tell Rstudio where you want the folder
```

Pull: When you want to update your local copy of the repo with the one that is on GitHub (i.e. We have added the new files for next session and you want to bring them onto your computer), you would hhhfirst open up the R project BrandeisLibrarySummeR and on the top right console where environment tab is change the select tab to git.Then hit the down arrow this pulls info from GitHub down to you. It's good practice to always pull first when you go to work on a repo that is not yours alone.

Create your own repo on GitHub and clone to your local drive

1. Login to GitHub, hit the + icon next to the avatar icon in the top right corner. Under the drop down menu select 'New repository'
2. Name your repository. You can leave it public or change it to private. You do not need to initialize a ReadMe, or change any of the other defaults.
3. Open a new R project with Git.
   - File -> New Project -> Version Control -> Git
   - Add the url from the page that popped up after you created your repo on GitHub, check the box at the bottom left that will open a new R session (this lets you continue to have this session open with these instructions)

Create a file on your local drive and send it to GitHub

4. In the new R session (it will now have a little bubble with the repo name to distinguish it), open a new R file, Rmd or R. Write anything you want inside it and save it. It will by default save into the R project that is connected to the GitHub Repo (which is what you want, R projects create a folder on your computer that should hold all the files necessary for a project altogether). You should now see this file in the files window on the bottom right console.
5. Go to the Git tab at the top right console. Hit the Diff button on the left side of the top right console. This opens a panel that will show you the difference between your local files and the repo. In our case when we look at every file all of the content inside the file is green. This is because the repo we cloned from GitHub was empty initially, Rstudio has automatically populated the .gitignore for us since then, we created the .Rproj file when we created the R project, and we have our new file.
6. *Add / Stage* is the first step in version controlling your files. In the diff panel we can 'stage' the files (i.e. get ready to commit the changes we made into our version control system then send to GitHub), just by checking the box next to the file.
7. After you have checked the boxes for each file and staged the files, you can now commit the changes to the local copy of the repo. You do this by adding a commit message in the top right box and hitting commit. Commit messages are meant to be messages to yourself that help you keep track of why you

made the changes. You want these to be as meaningful as possible. After you hit the commit button a window will pop up reminding you of what you changed and after you close that you should have a clear diff window open.

8. Your files are now being tracked on your local repo (the copy of the repo on your computer). They are NOT on GitHub. To send the files on GitHub, we *push* them. We do this by hitting the up arrow in the top right Rstudio panel (when the Git tab is selected). You'll need to enter in your GitHub credentials.
9. Check GitHub to confirm you can now see your files.
10. Add more to your file and repeat the process of *Add/ Stage -> Commit ->Push*, then hit clock icon in the Git panel in Rstudio and checkout how your versions/changes are being tracked.

Want to learn more? Check out these resources:

1. Happy Git with R
2. Software Carpentry Git Lesson