GitHub from RStudio

1. Create a GitHub repo (using the GitHub website)
   1. Click on the picture that represents your profile (in the top right corner of GitHub)
   2. Click “Your organizations”
   3. Click “LaryLab”
   4. From the tabs along the top, select “Repositories”
   5. Click the big green button “New repository”
   6. Give the repository a (informative) name
   7. Add a description if you like
   8. Select “Public”
   9. Check “Add a README file”
   10. Click the big green button at the bottom “Create repository”
   11. Write something in the README file – just a short line about what is contained in this repo (normal markdown rules apply)
   12. Now when you click on this repo, you’ll see a big green button on the right side that says “<> Code” that you can click
   13. The first option under “Clone” is for HTTPS – this is what we want – click the button over on the right side that looks like two overlapping sheets of paper to copy the URL.
2. Create a personal access token (classic)
   1. Click on the picture that represents your profile (in the top right corner of GitHub)
   2. Click “Settings”
   3. On the left side, click “Developer settings”
   4. On the left side, click “Personal access tokens”
   5. Click “Tokens (classic)”
   6. Click the button that says “Generate new token”
   7. From the dropdown menu, click “Generate new token (classic)”
   8. In “Note” give the token a name (that makes sense)
   9. For “Expiration” select “Custom”, then enter a date a year from now (you’ll have to make a new one then, but it helps to keep things more secure)
   10. Under “Select scopes” select: repo, write:packages, delete:packages, and project
   11. You will be shown the new personal access token – make sure you save this somewhere safe because you won’t be able to access it again!
3. Start a new project on RStudio
   1. File > New Project > Version Control > Git
   2. It will ask you for the repository URL – this is the address you copied in step 1.
   3. It will ask for a Project directory name – it’s easiest to make this the same name as the repo
   4. It will ask where you want to create the project – pay attention to this and make sure it’s where you want it to go
   5. A popup window will ask for your GitHub username – enter it
   6. Another popup window will ask for your password – **this is not your github password, nor is it your Discovery password** – it is looking for the public access token we made earlier, so copy and paste that into the box
   7. Now (hopefully?) you will have a little greed and red “Git” that is sideways at the top of the screen under the toolbar, and a “Git” tab over to the right on the toolbar with the “Environment” tab
   8. Let’s edit something to see how it works – we’ll create a script file
      1. Open a new script file and save it as something – doesn’t matter what
      2. Type a line in there and save it
   9. Click on EITHER of the “Git” tabs – if the colorful one, click the “Commit” option, if the tab on the right side, just click that tab
   10. There will be boxes on the side of the file names, select all of these and click “Commit”
   11. If it hasn’t already, a new window will pop up that is a more extended version of the same commands, and there is a space on the right side to write a commit message – write one (doesn’t have to be long) and click “Commit”
   12. You *might* get a message (in a new popup window) that says that it doesn’t know who you are (this has not been consistent for me), and in that case go back to the terminal and enter two commands: `git config --global user.email "jennifer.l.spillane@gmail.com"` and `git config --global user.name "Jennifer Spillane"`, then try to commit again
   13. If the files disappear from the window, it’s a good sign, and we can click the “Push” button to send these changes to GitHub
   14. Go back to GitHub and refresh the repo page and you should see the files there!

Some alternate paths for a similar destination:

1. Create a project from an existing directory
   1. File > New Project > Existing Directory
   2. Type in or select the working directory you want to make into a project
   3. Tools > Project Options > Git/SVN
   4. Version control system: Git
   5. Window will pop up asking if you want to initialize a new git repository for this project > Yes
   6. You will need to restart RStudio so that the change will take effect > Yes
2. Create a new project in RStudio and link it to a new repo in GitHub
   1. File > New Project > New Directory
   2. Click “New Project”
   3. Give it a name, check/select the location, and check “Create a git repository” (I also check “Use renv with this project”)
      1. You can tell that this worked because there should be two new files in the “files” section of RStudio: project\_name.Rproj and .gitignore (probably also a .Rhistory file but you can ignore this)
   4. Click on the “Git” tab – this could be the green and red sideways “Git” under the toolbar at the top (in this case click it and then click “Commit”), or it could be the “Git” tab over on the side where you have other tabs for “Environment” and “History” (in this case just click this tab)
   5. Both files should be listed with boxes next to them, check both boxes
   6. Click “Commit”
   7. There should be a window that pops up and you can enter a commit message – make it something informative about what changes you made
   8. Another window will pop up telling you what just happened, click “Close”
   9. Go to GitHub and create a new repository, just like above but with these changes:
      1. Name it the same thing that you names the RProject that you made
      2. **DO NOT** check the “Initialize this repository with a README”
      3. Create the repo
   10. There should be different sections with lines of code in them, we are interested in the code under “…or push an existing repository from the command line” – copy the first line of this code (should be three lines)
   11. Go back to RStudio and click on the “Terminal” tab and paste the command there and execute it. Do this for all three lines of code. (It’s probably okay to do this for all three at once, but I haven’t tested this, and doing them one at a time gives us time to address any errors)
   12. Now if you go back to GitHub (and refresh the page) you should see the .Rproj and .gitignore files that have been pushed to GitHub from the local repo
   13. Let’s add a README file to the repository. We can add it in either place, but let’s add it in GitHub so that we can pull it into our local repo from RStudio. There should be a green “Add a README” button to click in your repo on GitHub
   14. Write something super short here (normal markdown syntax applies) and then click the “Commit new file” button

If you need to merge repositories from old GitHub accounts: <https://docs.github.com/en/repositories/creating-and-managing-repositories/transferring-a-repository>

If you want a rundown of the benefits of keeping things in R projects: <https://r4ds.had.co.nz/workflow-projects.html#workflow-projects>

If you want to use renv to save the settings you use for a project and share them with other people: <https://rstudio.github.io/renv/articles/renv.html#getting-started>

If you want a general overview of git/GitHub and some tutorials with well explained processes: <https://happygitwithr.com/>