Week 8 — Git On

Z620: Quantitative Biodiversity, Indiana University

March 6, 2015

OVERVIEW

A major goal of Quantitative Biodiversity is to provide you with tools to conduct reproducible science. One of these tools is Github. Github helps ensure the integrity of a research project by managing changes to data and code. The version-control features of Github are useful for an individual working on a "solo" project, but also greatly facilitate collaboration among researchers.

This semester we've been using an Enterprise version of Github (https://github.iu.edu/), which is restricted to people with an IU affilition. Although effective for teaching and development, there are a few downsides to using Enterprise versions of Github. First, you may lose access to data or code that is stored in an Enterprise respository upon graduating from IU. Second, your ability to collaborate with colleagues at other institutuions will be hindered if you rely on an Enterprise version of Github. Last, Enterprise versions of Github do not help us meet the requirements of many journals and funding agencies that our data and code be "open". In this handout, we present a workflow that will assist you with conducting reproducible science upon "graduation" from Quantitative Biodiversity.

After completing this exercise you will know how to:

1. Create a Github respository
2. Make a source-code file
3. Create and modify a README file
4. Properly license your work
5. Create a github.com account (not restricted to IU)
6. Migrate repositories from IU's Github to github.com

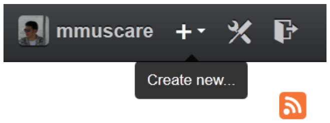
1) CREATING A GITHUB RESPOSITORY

If you adopt Github as a part of your scientific workflow, you are going to want to create a new repository when you start a new project. A repository holds all of your files, along with a history of the changes that have been made. This semester, you have been working with a repository that the instructors created for you (2015-QuantitativeBiodiversity). On the first day of class, you forked and cloned this respository to your local computer. After modifying the files on your local computer, you saved and commited the changes, which were then pushed to your origin (e.g., QB2015_Smith). The final step in the Github version control process is to create a pull request so that changes are merged into the upstream repository.

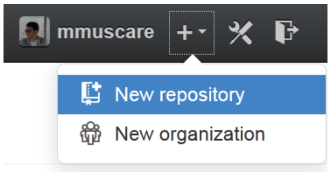
In the following section, we are going to go through the steps that are involved in creating your own repository:

1. Open a web browser and navigate to https://github.iu.edu/

2. At the top of page there is black banner. On the right-hand side there is **+** symbol. If you hover over this symbol, it will say "Create new. . . ".



3. Click on the **+** symbol and select "New Repository"

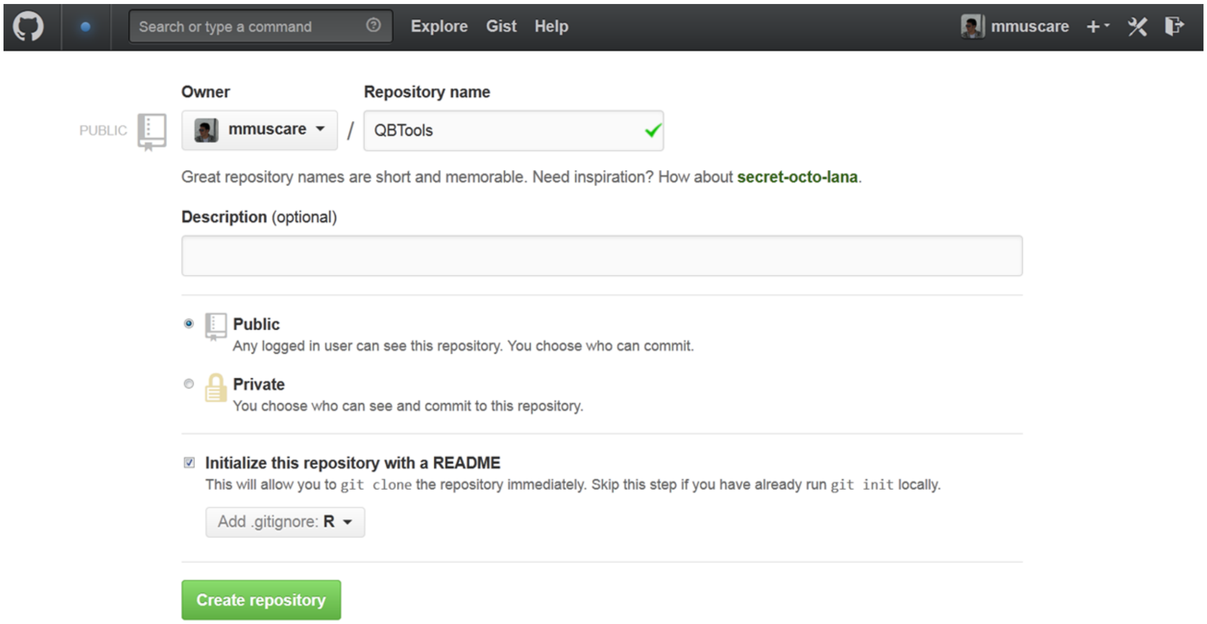


4. Give your repository a name. For this exericse, name your repository “QBTools”.

5. Decide if you want the repository to be accessible to all IU affiliates (Public) or not (Private)

6. Click the button that says “Initialize this repository with a README”

7. Optional: you can click on the “Add .gitignore” button and select R. This allows you to identify certain files (e.g., .Rhistory) that will be ignored by git when it looks for modified files to commit.



Owner: mmuscare / Repository name: QBTools ✓

Great repository names are short and memorable. Need inspiration? How about **secret-octo-lana**.

Description (optional)

● **Public**
Any logged in user can see this repository. You choose who can commit.

○ **Private**
You choose who can see and commit to this repository.

☑ **Initialize this repository with a README**
This will allow you to `git clone` the repository immediately. Skip this step if you have already run `git init` locally.

Add .gitignore: **R**

8. Click on the green “Create repository” button

9. Now, we are going to clone the newly created repository to your local machine. Open a terminal window and type the following:

~~~
```sh
cd ~/GitHub
git clone https://github.iu.edu/User_Name/QBTools
cd ./QBTools
git status
```
~~~

10. Congratulations on successfully creating your first repo! Before we move on, it is good practice to check your remotes. You should have just one remote: origin. Open a terminal window, `cd` to the correct directory, and type the code below to confirm.

~~~
```sh
git remote -v
```
~~~

2) CREATE A SOURCE-CODE FILE

Over the past eight weeks, we have created a number of functions in R. Some of them took a long time to write. In this part of the handout, we are going to compile some of these user-defined functions into an R source-code file. As you will recall, the benefit of a source file is that it contains “vetted” code, which can be used across multiple projects. Once created, you will be able to load your source-file into a new project and use all of the functions that you created in Quantitative Biodiversity. Moreovoer, you can add new functions to this source-code file and effectively create a library of commonly used functions. The instructions below will guide you through the steps needed to create an R source-code file:

A. Create a Blank R File

Go to your menu bar and choose: File > New File > R Script. Alternatively, you can use the following shortcut key: Ctrl + Shift + N. Note: you want this to be a .R file, not a .Rmd file. Save this blank file as `QBTools.R` in your QBTools respository directory..

B. Add a File Header

The header of your source-code file should provide necessary information that will allow others to use your script.

Here is an example header:

```
################################################################################
#                                                                              #
# Quantitative Biodiversity Functions Source Code                              #
#                                                                              #
# Written by: Student X                                                        #
#                                                                              #
# Last update: 2015/03/06                                                      #
#                                                                              #
# Notes:                                                                       #
#                                                                              #
################################################################################
```

There are a few things to note about the header.
First, the comment character (#) prevents R from interpreting the information in the header. Second, the header is formatted for stylistic purposes. In our example, the header is 80 characters wide. As a rule of thumb, this is an appropriate width for both your header and code. We use the comment character (#) to box-off the top and sides of the header.

C. Nuts and Bolts of the Source-Code File

i. Require commands

In order for function in a source-code file to properly work, we need to make sure that the contributed packages are installed. By default, the following code will try to load a specificied package. If that package is not found, R will install the package.

```
require("vegan")||install.packages("vegan");require("vegan")
```

You can cusotmize this code for any package that you want loaded for our source file.

ii. Functions

We are now going to populate our source-code file with functions that we have written this semester. Go through your assignments and find the functions that you want to save. Copy those functions into the new source-code file. Add comments about each function before the code. This annotation will help you and others understand how the functions operate. Here is a hypothetical example:

```
# My_Function: returns estimate of beta biodiversity
# Inputs: x, y, z

My_Function <- function(x, y, z){
  ...
}
```

When you are done copying functions over, save your `QBTools.R` file to your new repository. In a terminal window, add and commit the file using git.

```
git status
git add ./QBTools.R
git commit -m "Initial commit"
git status
```

3) CREATE A README FILE

You may recall at the beginning of the course that we had you edit a README file, which gave us some basic information about you. README files are commonly used and contain important documentation about the files (code, data, etc.) that are contained in a directory. It's good practice to inclucde a README file for all of your Github respositories. A basic README file should contain the following:

1. General information on the content of a repository
2. The version of programs and software that are needed to run the code in a repository
3. Contact information for the owner of the repository (e.g., name and email)
4. A description of any potential bugs

When you initialized your new repository, you created a blank README.md file. Take a moment to modify this README file to include the information needed for your QBTools repository. You can open this file in RStudio to make changes. We have provided an example README.md file in the GitOn directory of your QB_2015 repo; you view this file as an example.

Before moving on, save the modified `README.md` file in your new repository. In a terminal window, add and commit the file using git:

```
git status
git add ./README.md
git commit -m "Initial commit"
git status
```

4) LICENSING YOUR CODE

(this should come after "MIGRATING" because you can't automatically license code on IU's GitHub. If you figure out how to add a license, it won't mean much because IU probably owns whatever you put on their Enterprise, i.e., IU rights supercede your license)

When setting up public repositories on GitHub, you will want to license your work because it will be in the public domain. Licenses are what allow you to establish the right to waive responsibilities and liabilities, to attract copyright, and to legally establish intellectual primacy. Licensing allows us to be visible, transparent, and fearless professionals; protected by putting our ideas and work into the public domain while maintaining rights and restrictions to (re)use and (re)distribution.

GitHub offers a broad selection of licenses that range from short and tidy (MIT) to the thorough but lengthy (GPL V3); see http://choosealicense.com/. 9 Simple ways to make it easier to (re)use your data. There's lots of goodies of in it, the repo which is found here: https://github.com/weecology/data-sharing-paper.

5) CREATE A GITHUB.COM ACCOUNT

In the overview section of this document, we described some of the advantges that are afforded by using github.com. If you don't already have a github.com account, let's create one now. Open up a new browser and navigate to https://github.com/. After supplying a username, your email, and a password, click the green button that says "Sign up for GitHub". You will then be asked to fill out some additional information. *Step 1: Set up a personal account* – You may be asked to add personal information again (username and password). *Step 2: Choose your plan* – Here, you will most likely want to select the free plan ($0/month). Last, click the green "Finish sign up" button at the bottom of the page. You should then be taken to the main page of your github.com account. The interface should be identical to that of IU's Enterprise Github.

6) CREATE NEW REPOSITORY IN GITHUB.COM

–> KJL AND MEM WILL WORK OUT DETAILS

A) CREATE NEW REPO IN GITHUB.COM ("QBTOOLS")
B) INITIALIZE WITH NOTHING
C) ADD REMOTE
D) THIS WILL BRING IN SOURCE CODE, README, AND LICENSE FILE.
E) STUDENTS CAN REPEAT FOR ASSIGNMENTS IF THEY WANT

Many scientists who seriously engage in reproducible science elect to make their respositories public. If for whatever reason you prefer not to do this, you have a few options:
1) Don't have to migrate your files
2) Choose to make your repository private, at which time github.com is likely to ask for your credit card information (it generally costs money to maintain private repositories)
3) Make your respository public for now, and then request private repositories from github.com (see here: https://education.github.com/guide/private_repos)
4) Make your respository public for now, and the delete it (or github.com account) after class

6) MIGRATING TO GITHUB.COM

notes, add: 1. how to setup 2. create a repo…GitHub will ask whether you want to add a README.md. 3. add a license…should the licensing section be moved here, as a subsection? 4. add source file