Week 1 Exercise: Git and GitHub

Z620: Quantitative Biodiversity, Indiana University

January 16, 2015

Git

Git is a software system designed to allow collaborators to simultaneously work on text documents, project data, and computing code for websites and for analyzing data and conducting scientific analyses. Git is a form of **version control**.

Version control

Version control allows people to examine, comment on, and revert back to any changes within the life of a document. In short, version control works when a person makes changes to a version of a project (code, data, or text) on their a *local* computer, and then *pushes* those changes to their personal *online* version. Having an online version of your project helps keep it safe from loss or damage. If working with collaborators, you can then submit a *pull request* for others to check and *pull* the changes into the group's project. One of the advantages of version control is that it allows multiple people in remote locations to collaborate on projects without irretrievably deleting or writing over others' work.

GitHub

GitHub is a web-based service for hosting projects that use the Git version control system. GitHub provides a graphical interface for viewing and managing a project's code, data, and text files, whether in collaboration with others or working individually.

–> Perhaps the conceptus of forking can be explained little more clearly

GitHub allows users to *fork* a personal version of a project. Nowadays (***???***), downloading new open-source software is often done by forking a GitHub repository. ***is this sentence necessary?*** Forking immediately creates a copy of a project on your personal GitHub page [(or other GitHub page that you have permissions ***necessary***)]. It is important to note, however, that forking does not create a copy of a project on your local machine. Therefor, after forking a project, you need to *clone* the repository onto your personal computer ***first time "repository" is used – should we define or at least not introduce parenthetically?*** Cloning a repository can be done via GitHub (the graphical web-interface) or by working with Git directly using **Linux** commands, which is what we will be doing in Quantitative Biodiversity this semester.

GitHub at Indiana University

IU has an Enterprise GitHub system that is a university managed version of GitHub available to IU faculty, staff, and students. So, IU basically has basically forked a restricted version of the GitHub project. IU's Enterprise version GitHub is only visible to members of the IU community.

Goals

- Install and Configure Git
- Create an GitHub account using IU's GitHub Enterprise
- Learn what repositories are and they will be used during Quantitative Biodiversity

Git Installation

If you do not have a current Git installation (already installed in computer lab), please do the following:

1. Naviate to git-scm.com/download/
2. Select the appropriate operating system
3. The download should start automatically
4. Open the installer and follow the onscreen directions

On Mac: You will need to make sure you have Xcode Command Line Tools installed.
On Windows: This process will install Git Bash (msysGit). During installation, you will be asekd to adjust your PATH environment. We recommend that you select the option to "Use Git from the Windows Command Prompt". This will give you the most flexibility with Git. In addition, we recommend that during installation you select "Use OpenSSH" for your secure shell client with GitBash.

During installation, you will be asked how to configure the line ending conversions **On Mac**: We recommend "Checkout as-is, commit Unix-style line endings" **On Windows**: We recommend "Checkout Windows-style, commit Unix-style line endings"

Git Test

Before we get started with Git, we first need to test our current installation to make sure there aren't any issues. The easiest way to do this is to look up the Git version currently installed. We will use terminal (GitBash on Windows) to accomplish this.

The first thing we need to do is find and start terminal. On the lab computers, you can find terminal in the **Utilities Folder**. On your personal computer: **Mac** you can search for terminal with spotlight [Cmd+Space]; **Windows** you can find GitBash in the Start Menu.

1. Find terminal (or GitBash) and open a new window
2. Type the following commands:

```
pwd
ls
git --version
```

Git User Setup

1. **Organization:** We recommend that you create a folder in your user directory (> cd ~) called '*GitHub*' to make this and future assignments easier to manage. (**Mac** Users: Do this from Terminal; **Windows** Users: Do this from GitBash)

```
cd ~
mkdir ./GitHub
cd ./GitHub
pwd
```

2. **User Configuration:** You will need to configure your local Git installation. We will do this by entering your name and email. We will also set two parameters: push.default and credential.helper

```
git config --global user.name "John Doe"
git config --global user.email johndoe@example.com
git config --global push.default simple
git config --global credential.helper store
```

3. The last thing you need to do is configure how Git handles line endings. Line endings are invisible characters that your operating system places at the end of each line in a document. On Unix machines (e.g. Mac), this is the linefeed character (LF). On Windows machines, this is the carriage-return (CR) and linefeed (LF) characters. This difference in line endings between Mac and Windows causes incompatabilities between the two systems. However, Git is enabled to handle the differences by silently converting line endings when repos are push to remote servers. We recommend that you configure this behavior in order to prevent any future issues when collaborating across computer platforms.

   On Mac

   ```
   git config --global core.autocrlf input
   ```

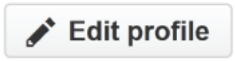
   On Windows

   ```
   git config --global core.autocrlf true
   ```

You are now ready to *Git !!!*

Create User with IU's Enterprise GitHub Service

1. Navigate to https://github.iu.edu
2. Sign in with your IU Username and Passphrase
3. On the top right of your screen, click on your Username
4. Click on the Edit Profile Icon to edit your profile

Fork and Clone a Repo

1. Navigate to and click on your student repository (repo) on https://github.iu.edu/2015-QuantitativeBiodiversity
2. Fork your repo by clicking on the Fork Icon in the top right of your screen

   You should now see the repo on your GitHub page.

3. Clone the repo onto your local machine using the command line (terminal or GitBash). Replace "User_Name" with your IU Username and "Repo" with your QB Repository.

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

The repo should have downloaded onto your local machine and the status should stay "all up to date". You should also see that the only thing in your repo is a file named README.md

4. Check and update remote repo. The following commands will at your upstream remote repository (located on the QB Course GitHub). Replace "Repo" with your QB Repository

   ```
   git remote -v
   git remote add upstream https://github.iu.edu/2015-QuantitativeBiodiversity/Repo
   git remote -v
   ```

   You can copy and paste the URL for your upstream repo from the GitHub website.

5. Open and edit the README.md file:

   This file is a Markdown file. Markdown is markup language for writing and editing text that can be easily converted to other formats (e.g. HTML, PDF, Word). During this semester, we will edit Markdown files using RStudio. On the lab computers, you can find RStudio in the **Analysis & Modeling Folder**. From RStudio, navigate to and open your README.md file. Edit the file as needed (we will demonstrate). Update your '*Student Name*' with your full name. Enter your email address. Write a short Bio about yourself (~ 1 paragraph). List three to five course expectations. Hint: View the Markdown guide to learn about formatting and making ordered lists (https://guides.github.com/features/mastering-markdown/). When you are done, save the close the document.

6. Now we need to add and commit our changes to git. However, before we add anything we want to make sure that we are

   ```
   git status
   git add ./README.md
   git commit -m "Updated README.md with student information"
   ```

7. Now push the changes to GitHub. Before we push our changes, we always want to check for (fetch) and merge in any changes others have made.

   ```
   git fetch upstream
   git merge upstream/master
   git push origin
   git status
   ```

   You should now see the repo, including your recent changes, on your GitHub page.

8. Navigate to your GitHub page to make sure that the file was uploaded correctly. If so, submit a Pull Request to submit your file to the course instructors.

   The course instructors can now merge and see your changes.

9. To get new assignments, you will pull (fetch & merge) your upstream repo. This will allow any updates your instructors have made to be merged with your local documents. In addition to pulling yoru upstream repo, you always want to push any updates to your origin.

   ```
   git status
   git fetch upstream
   git merge upstream/master
   git push origin
   git status
   ```

During this course, you will recieve and submit all assignments using these methods. In addition, you will use Git and GitHub to contribute to assignments in class and on your personal computers.