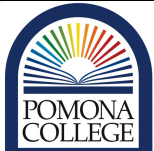


| | | | |
|---|---|------------------------------------|-----------------|
|  | Environmental Analysis Teaching and Research Laboratory | Date: 8/12/2016 | Number: 06 v.01 |
| | Standard Operating Procedure | Title: Rstudio Projects and Github | |
| | Approved By: TBD | Revision Date: September 7, 2016 | |

1. Scope and Application

1.1 R, RStudio, and Github combine as a resource for data analysis and display.

1.2 We explain how these resources can be set up to create collaborative projects using Github repositories.

1.3 We do not expect you to be an expert on how to use R or Rstudio. However, some experience with a computer language will be helpful.

2. Summary of Method

2.1 This SOP does this...

2.2 The use of Github is not intuitive. So, you will find many on-line resources to help.

Contents

| | | |
|-----------|--|----------|
| 1 | Scope and Application | 1 |
| 2 | Summary of Method | 1 |
| 3 | Acknowledgements | 3 |
| 4 | Definitions | 3 |
| 5 | Background | 3 |
| 6 | Interferences | 4 |
| 7 | Health and Safety | 4 |
| 8 | Personnel & Training Responsibilities | 4 |
| 9 | Required Materials and Apparati | 4 |
| 10 | Estimated Time | 5 |

| | |
|--|-----------|
| 11 Setting Up Accounts and Communications Keys | 5 |
| Creating a Connection between Github and RStudio | 5 |
| Pulling a Repository | 7 |
| Creating a New Repository | 7 |
| Creating a New Project | 7 |
| 12 Working on Projects | 7 |
| Rstudio Panels | 7 |
| Best Practices | 10 |
| 13 Troubleshooting | 11 |
| "Commit" problems | 12 |
| 'Merge' Errors | 13 |
| "Pull" is rejected | 13 |
| Dealing with non-fast-forward errors | 13 |
| "Push" fails | 13 |
| 14 Collaboration and Version Control | 13 |
| Workflow tracking | 13 |
| 15 References | 14 |

3. Acknowledgements

4. Definitions

- 4.1** RStudio is use interface for R. Although R by itself is an amazing example of crowd sourcing, where a wide range of staticians and programmers have created a free programming environments with a robust range of statistical packages, the RStudio interface provides a user with the tools to track and publish their analysis process in an effecient and transparent way.
- 4.2** Local Install versus Server — R and RStudio can be installed on a local computer/laptop from the CRAN download mirror sites. However, we also have access to the R and RStudio Server installed on the Pomona College mainframe, where you can access it via a web browser. Wow, this is conveient!
- 4.3** GitHub is a web-based Git repository hosting service.
- 4.4** Version Control is a method to track changes in software, and often in the context of collaborative projects. The final component of R and RStudio is its capacity to create projects (RStudio's terminology) and repositories (Github's terminalogy) that can be shared among collaborators. In particular, the collaboration allows for contributions to be tracked via version control tools. There are a number of ways that we can access these tools, but we'll try to limit the methods so keep the process relatively "simple".

5. Background

- 5.1** R is a powerful, open source program but combined with RStudio and Github the program becomes an archetype of a program that enables 1) collaboration, 2) transparency, and 3) accessibility.
- 5.2** However, becoming facile in using these program is like learning how to walk. We need to approach this process in descrete steps. Some will require weeks of mistakes to learn, others will quickly learn to run with the programs. How quickly you can feel comfortable with these programs will depend on many factors, but will be greatly improved by the time you invest!

Almost Universal Compatibility No downloading programs on personal computers.

Accessibility Accessible from any computer via webserver

Collaboration

Version Control

- 5.3** I recommend reading the following sites to better understand Git and Github

- Rstudio Youtube
- Github Youtube
- Github for Beginners
- Github Fw
- Understanding the GitHub Flow
- <http://blog.osteele.com/posts/2008/05/my-git-workflow/>

6. Interferences

- 6.1** R has an updated version about every six months. When performing advanced analyses, there are times that new versions will no longer run a code. Thus, older versions of R must be maintained. This is a unheard of issue for new users.
- 6.2** RStudio Server needs a functional network connection. If the network is down, then Rstudio Server is inaccessible. This can be a source of frustration.

7. Health and Safety

- 7.1** Some risks include carpel tunnel syndrome.

Safety and Personnnel Protective Equipment

- 7.2** Good posture and work station

8. Personnel & Training Responsibilities

- 8.1** Researchers training is required before this the procedures in this method can be used...

9. Required Materials and Apparati

- 9.1** Laptop or destop computer
- 9.2** Access to Pomona College's SSO and server
- 9.3** Github account
- 9.4** Patience

10. Estimated Time

10.1 This set up procedure requires 30 minutes.

11. Setting Up Accounts and Communications Keys

11.1 Create Github Account: Go to Github.com and create an account. I suggest you use your Claremont e-mail address because this can come in handy later, should you want to create private repositories, which are free for college students! Your email address can be used to prove your student status.

11.2 Read the Readme file – THIS DOCUMENT WILL REPLACE THAT README FILE.

Creating a Connection between Github and RStudio

11.3 Open Rstudio – which can be either as a local installation or using an Rstudio Server via webbrowser.

11.4 To use Rstudio's version control, you must create a SSH Key that is used to open a secure connections between RStudio and Github.

11.5 To create a Key, follow the Rstudio menus Tools/Global Options/'Git/SVN' and select "Create RSA key".

11.6 You will have the option to create a passcode. I suggest you don't to make it easier for now. But when you have projects that you be worried about security, then it is a good idea.

11.7 The a crude image will be generated in a pop-up windows – that we don't care about as far as I can tell. So, close this window.

11.8 Next, select the 'View the public key' and copy the contents of the pop-up window using cntrl-c. Now we need to paste that key into Github.

11.9 Add SSH Key to Github

1. Go to Profile, in the upper right top of the Github page.
2. Select the menu item "Settings."
3. Using the left menu, select the "SSH and GPG keys" button.
4. Click on new SSH key, type in name of key (e.g. 'myRstudio SSH key')
5. Paste in the RSA key, from your clipboard, into the window below.

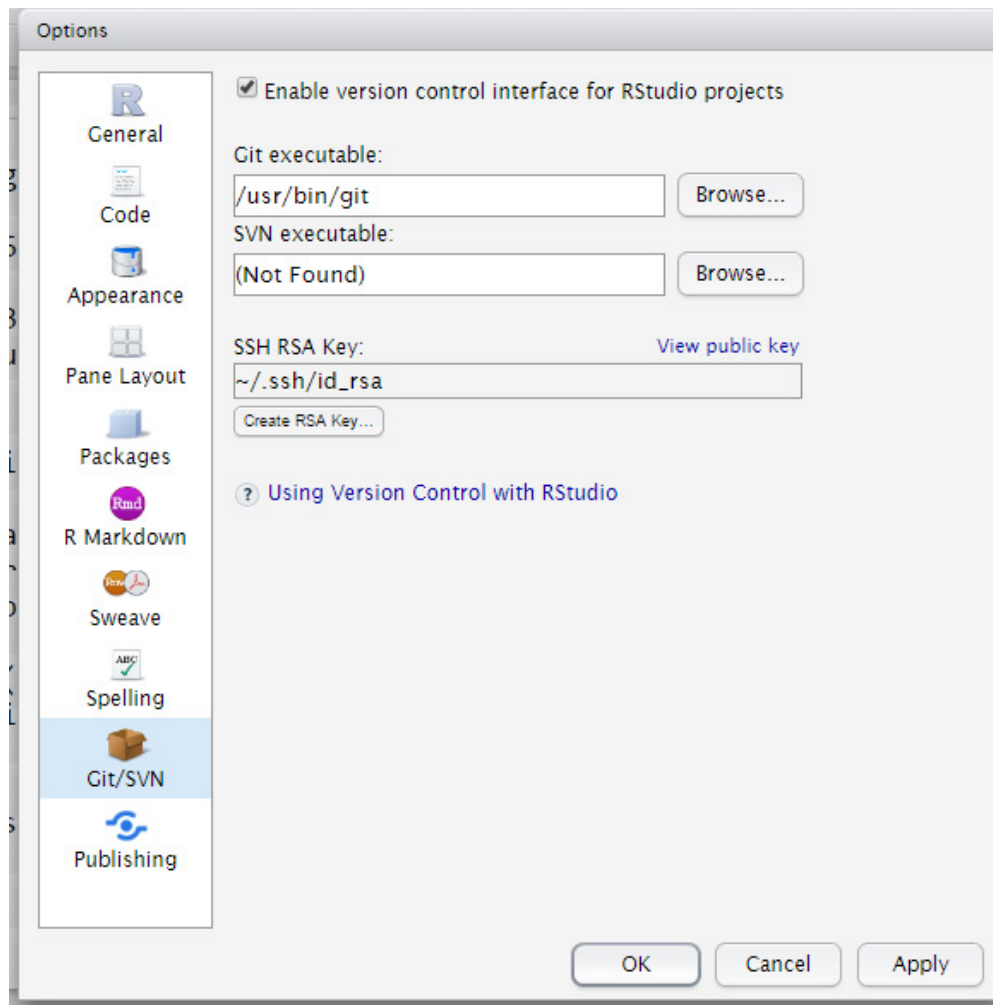


Figure 1: After navigating through the Rstudio menu, select 'Create an RSA Key'.

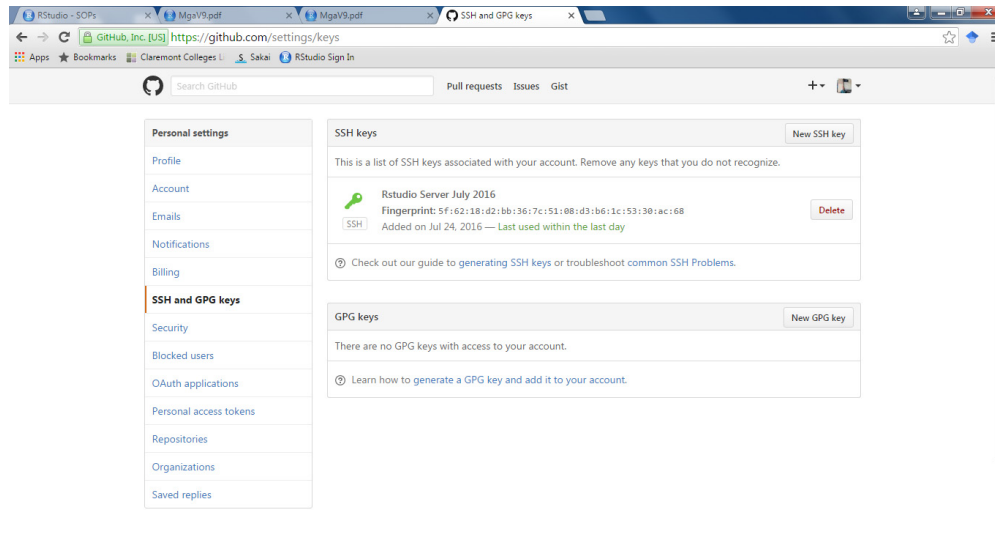


Figure 2: Creating the connection between Rstudio and Github requires a ‘key’. From this spot, you can create a new key and then name it based on the topic that you are working in.

6. Hit green “add SSH key” button.

11.10 Now Github and R can communicate.

11.11 To obtain access to repository in Rstudio, you will need to “clone” the site as a new project in Rstudio. To accomplish this...

Pulling a Repository

11.12 The easiest place to begin, is to ‘Pull’ a repository. But first you need to create a project in Rstudio. I suggest you the following steps as you begin:

Use the following Beginners_Repository,
<https://github.com/marclos/beginnersluck>

Creating a New Repository

Creating a New Project

12. Working on Projects

Rstudio Panels

File Structure and Rnw/Tex files

Console and Compile PDF windows

Git Panel

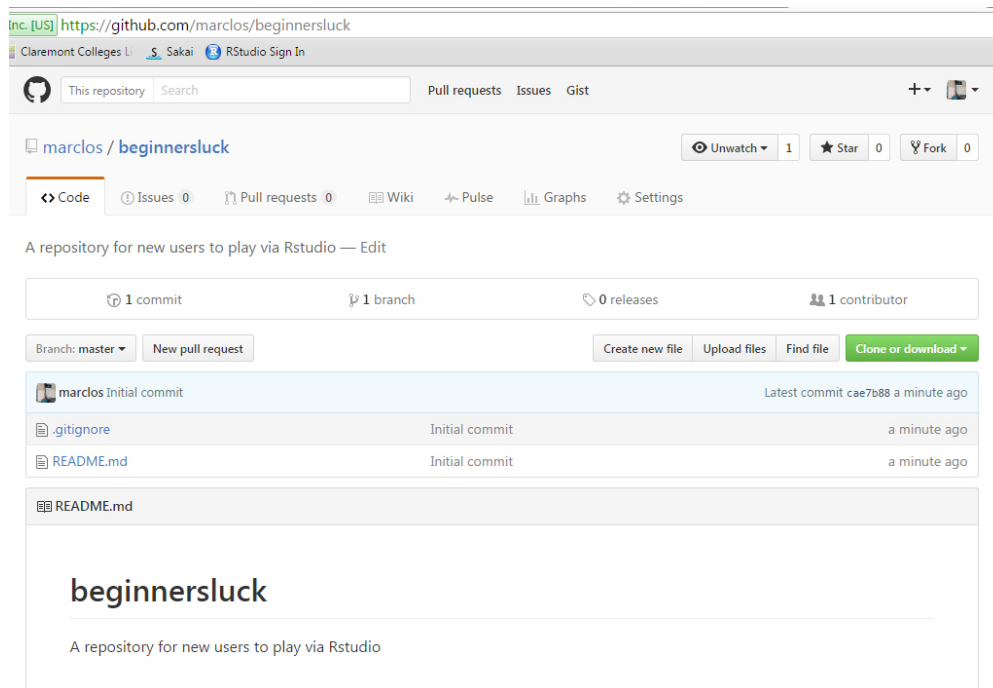


Figure 3: Step 1 should look like this...

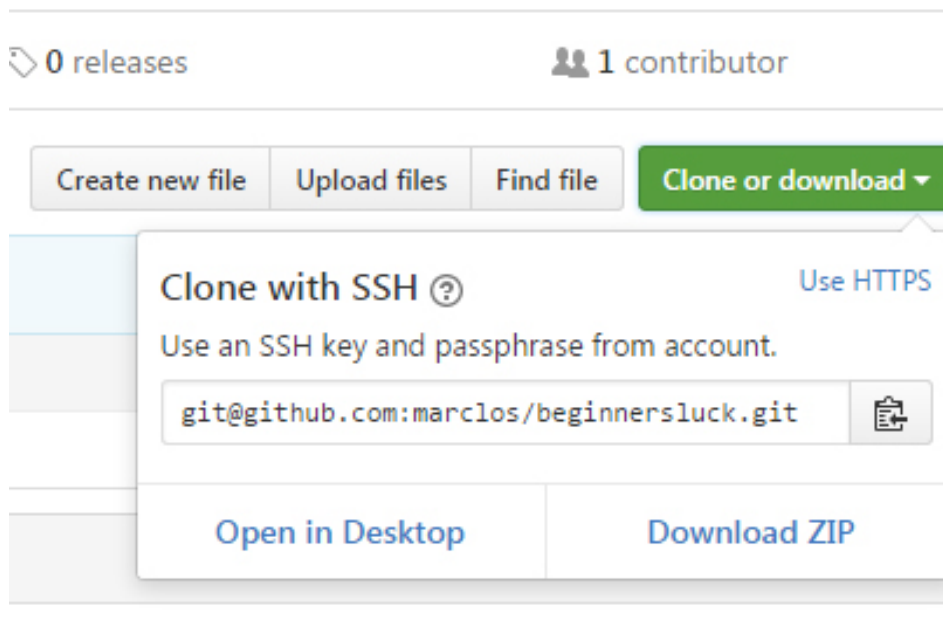


Figure 4: Step 1 should look like this...

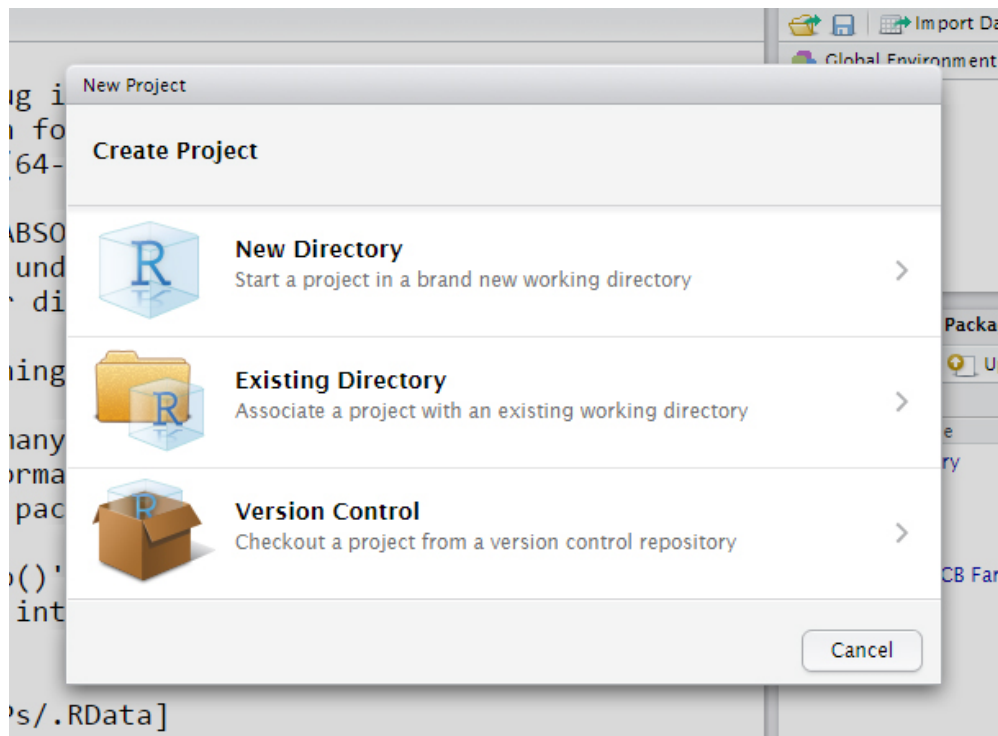


Figure 5: Step 1 should look like this...

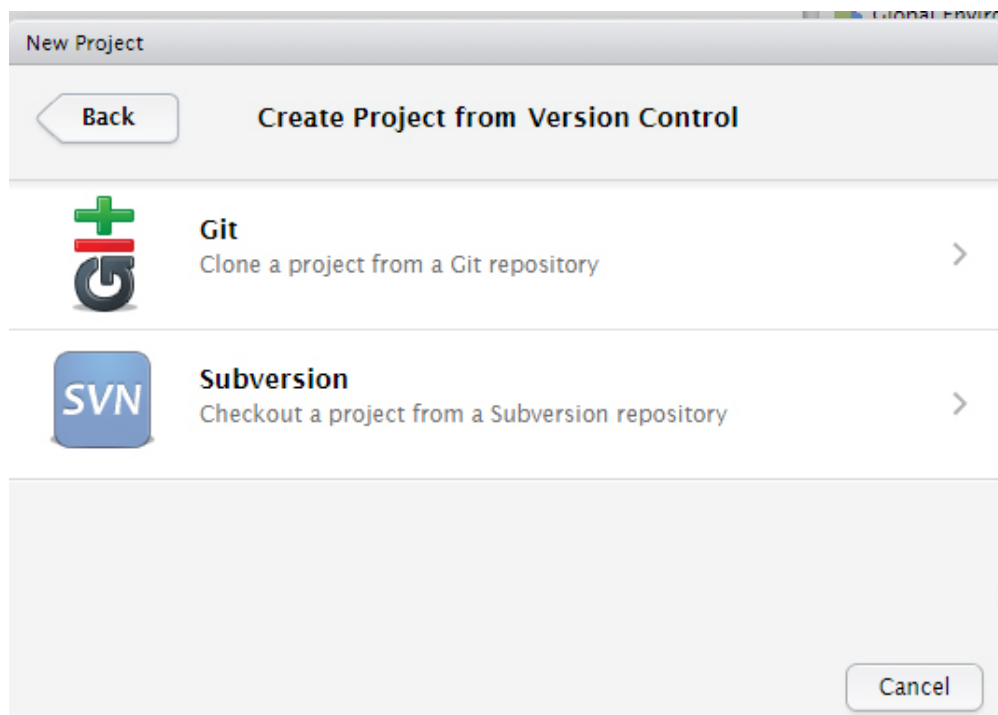


Figure 6: Step 1 should look like this...

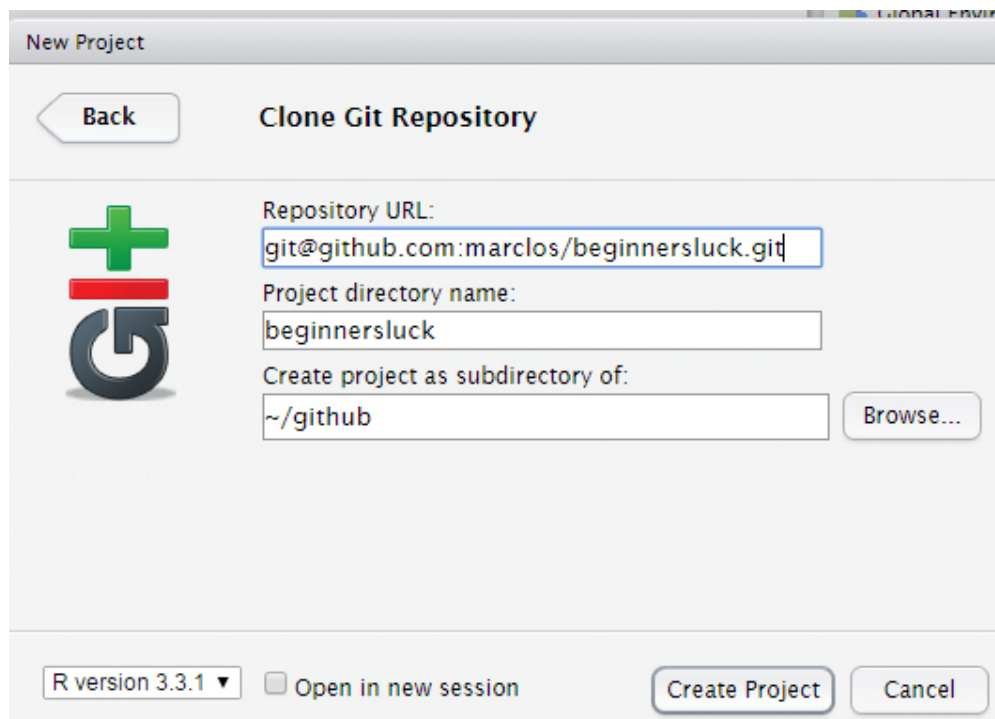


Figure 7: Step 1 should look like this...

Best Practices

Pull When you open RStudio, the first thing you should do is "Pull" from the repository to ensure your files are up-to-date. When you "Pull", you will get one of three results:

Already up-to-date. This means that your files have not been updated on the Github site since you last pulled the files. If you suspect someone has worked on the files, but are not getting those changes, it means that your collaborator has failed to "Push" these changes onto the repository. If this is the case, you might need to go to the troubleshooting question to address this problem.

Successful Updates If your files are successfully updated from the repository, you will see:

summary of updates

master 6038765 Started to explain set up procedures

5 files changed, 434 insertions(+), 8 deletions(-)

create mode 100644 06_Rstudio_Github/Rstudio-and-Github-concordance.tex

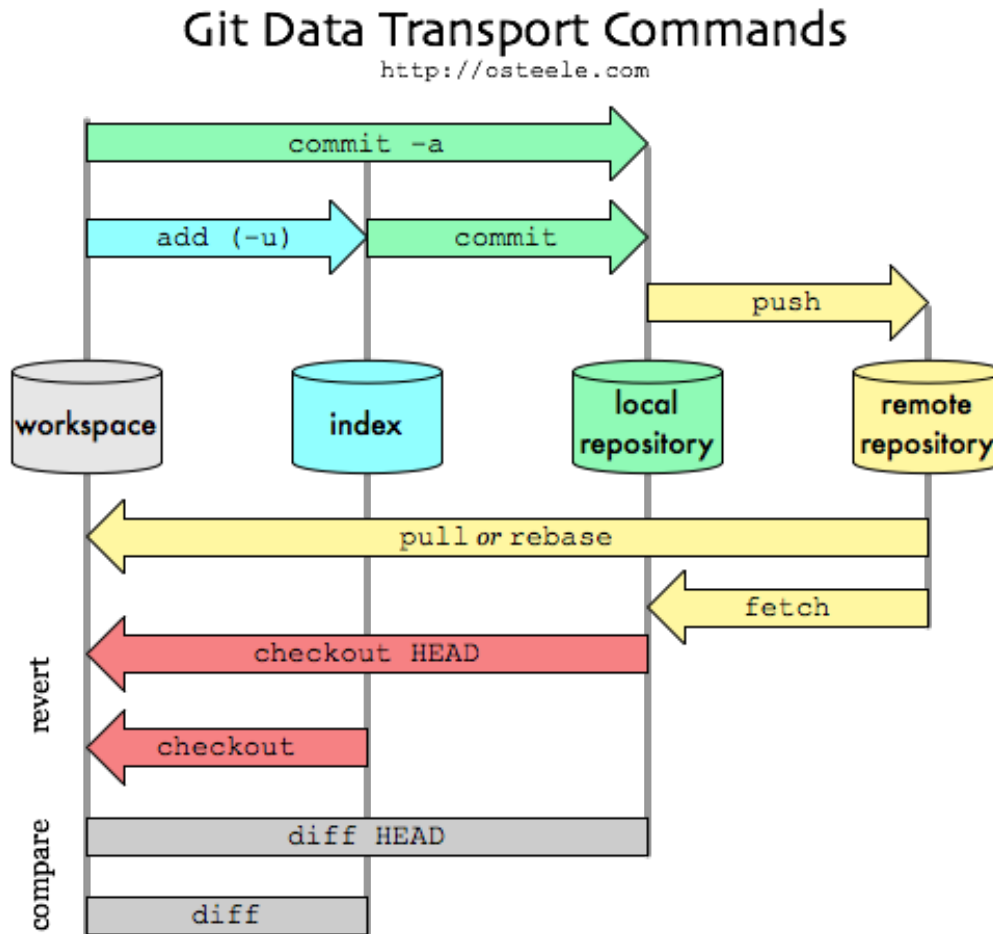
create mode 100644 06_Rstudio_Github/Rstudio-and-Github.log

create mode 100644 06_Rstudio_Github/Rstudio-and-Github.pdf

create mode 100644 06_Rstudio_Github/Rstudio-and-Github.tex

Unsuccessful "Pull"

Commit



Push As you might guess, when you "Push" you can also have several outcomes:

Everything up-to-date is certainly simple.

Successful "Push" "To git@github.com:marclos/SOPs.git
e2efe6f..0b18250 master -> master"

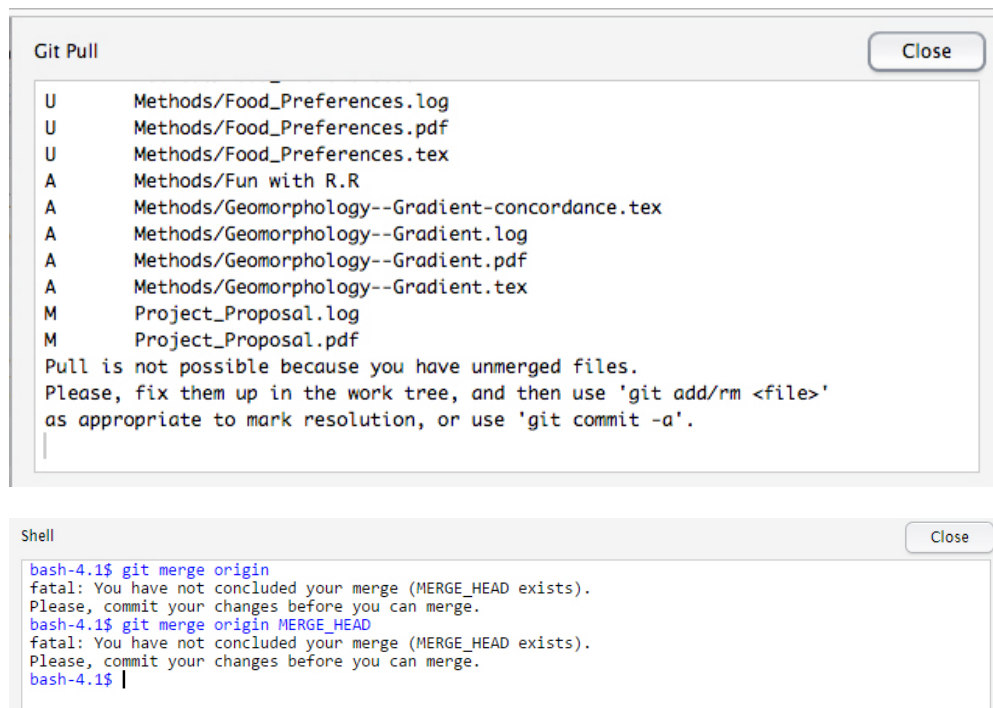
Unsuccessful "Push"

Basically git commit "records changes to the repository" while git push "updates remote refs along with associated objects". So the first one is used in connection with your local repository, while the latter one is used to interact with a remote repository.

Here is a nice picture from Oliver Steele, that explains the git model and the commands:

13. Troubleshooting

13.1 At first when you're a newcomer to working on projects within Rstudio connected to a GitHub repository, you may forget the "best practices" of pulling, committing and pushing described above. You might have already gone through all the steps involved in setting up your GitHub account, linking your workspace in Rstudio with



the correct project, and begun work on a specific file, but if you forget to update your workspace each time you come back to modify a file (especially in the case of coming back the next day to a file in Rstudio and forgetting to pull changes other collaborators may have made), you will run into problems committing, and pushing your changes. In this most common case, it's not likely that you'll notice any problems until you try to commit your changes. Thus, we'll begin this section on trouble shooting problems with Github and Rstudio at the committ level.

13.2 Alternatively, if you and others edit the same section, Github is going to need some “human” decision making to negotiate these changes. In Github jargon, this requires one to 'merge' changes.

13.3 (un)Fortunately, students are very good at discovering ways that the pull/commit/push process can be disrupted. As each case comes to my attention, I ask for a screen shot and a description, so we can trouble shoot each problem when they present themselves.

”Commit” problems

13.4 In the scenario described above, you will get an error code when trying to committ your new changes to a file.

'Merge' Errors**"Pull" is rejected****13.5** Merging changes... how??**Dealing with non-fast-forward errors**

13.6 Sometimes, Git can't make your change to a remote repository without losing commits. When this happens, your push is refused. If another person has pushed to the same branch as you, Git won't be able to push your changes.

13.7 You can fix this by fetching and merging the changes made on the remote branch with the changes that you have made locally:

- `$ git fetch origin`
- `# Fetches updates made to an online repository`
- `$ git merge origin YOUR_BRANCH_NAME`
- `# Merges updates made online with your local work`

13.8 Or, you can simply use `git pull` to perform both commands at once:

"Push" fails

13.9 Below are several potential remedies:

merge asdfasdf

Deleting a Project in R Studio If you are willing to sacrifice the changes you made or have mailed them to a collaborator to deal with you can delete the entire project in Rstudio. To accomplish this delete all files in directory, clear workspace and console, don't save, then go to session tab: Terminate R, and hopefully that will do the trick. Then commence with starting a new project.

14. Collaboration and Version Control**Workflow tracking**

14.1 Collaborators can create ways that each one is responsible for certain activities..

14.2 Branching...

15. References

- 15.1** APHA, AWWA. WEF. (2012) Standard Methods for examination of water and wastewater. 22nd American Public Health Association (Eds.). Washington. 1360 pp. (2014).