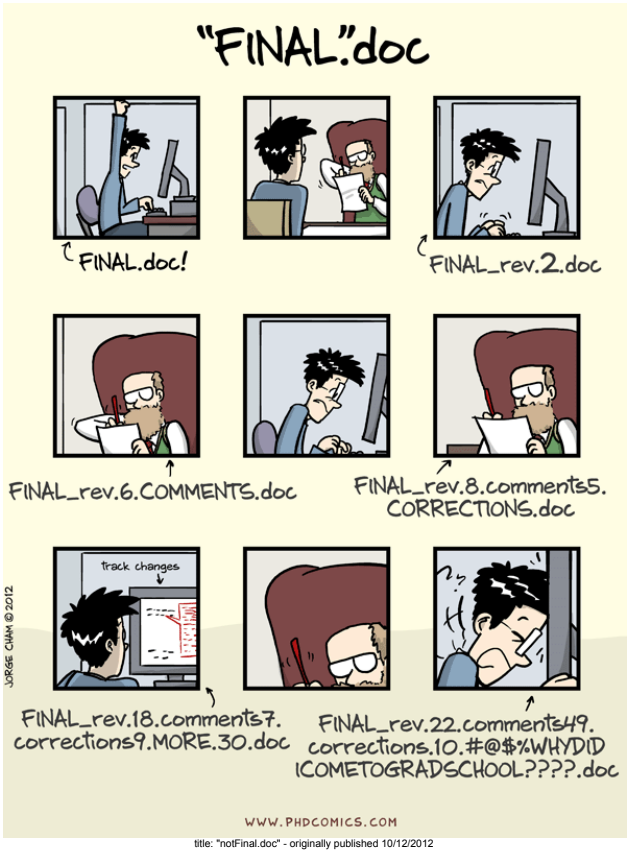
Notes on how to use Git

Sara Taheri

August 14, 2016

What is Git and Github and why is it usefull to use them for version controlling of our projects?



title: "notFinal.doc" - originally published 10/12/2012

situation 1

You have written a paper. You think it is finished and name it "final". After a while you make some changes to the paper and name it to "final_new". Then again you realize that you need to add a new experiment, so you edit the paper and name it to "final_really". You keep doing this for a while until it is really finalized.

Someone asks you a year later that if he can view your paper to learn from it. You open your folder and cannot remember which file was actually the final version of the paper![1]

situation 2

You are working on your project. The project is now in a good state. You have an idea and you want to implement your idea in the project. You add new codes and chang some parts of the project. At some point the project gets so messy with lots of errors that you don't know how to solve them and you wish if you could just go back to the previous state the project was in before you change it!

[1] situation 1 is from : http://r-bio.github.io/intro-git-rstudio/

You can always save a copy of the good state project somewhere in your computer and then make changes to the original version. But the project may take a year to be completed and you have to do this process many times and save different versions of it with different names.

situation 3

You and your teammate are working on the same project. You make some changes to the project. Your teammate also makes some changes. You will notify your teammate of the changes you made and she will notify you of the changes she made. Now both of you have to manually find the changes and add them to your codes. This is frustrating!

What is the solution to avoid these situations?

Git is a software that you install locally on your computer and you can handle **version control** of your files easily. Version control means to take a snapshot of your files everytime you edit them. Your files will be updated to the latest version while you always have access to previous versions whenever you need them. This solves the problems in situation 1 and 2.

When you start a Git project on your computer, you are going to store the entire history of the project locally. The storage of your project and its history is called a **repository**.

Every time you save your work, Git creates a **commit**. A commit is a snapshot of all your files at a point in time. If a file has not changed from one commit to the next, Git uses the previously stored file. Commits create links to other commits, forming a graph of your development history . You can revert your code to a previous commit, inspect how files changed from one commit to the next, and review information such as where and when changes were made.[2]

Github is a commercial website that lets you store your repository publicly for free. Once you commit something to repository, you can **push** them to Github. Now everything is updated locally and also on the server. This is useful because if something happens to your computer you will not lose your data.

Pushing the changes to Github has another benefit too. It lets you to work on a project with a team easily. Your teammate make some changes to the project. She commits the changes on her local repository. Then she pushes the changes to Github server. Now you want to work on the project. First you need to **pull** the new changes from Github and update your project to the latest changes and then start working on it. This solves the problem in situation 3.

Each time someone wants to work on the project has to pull first, then edit the file, commit his changes and push them to server.

[2]figure and the paragraph from *https://www.visualstudio.com/learn/what-is-git/*

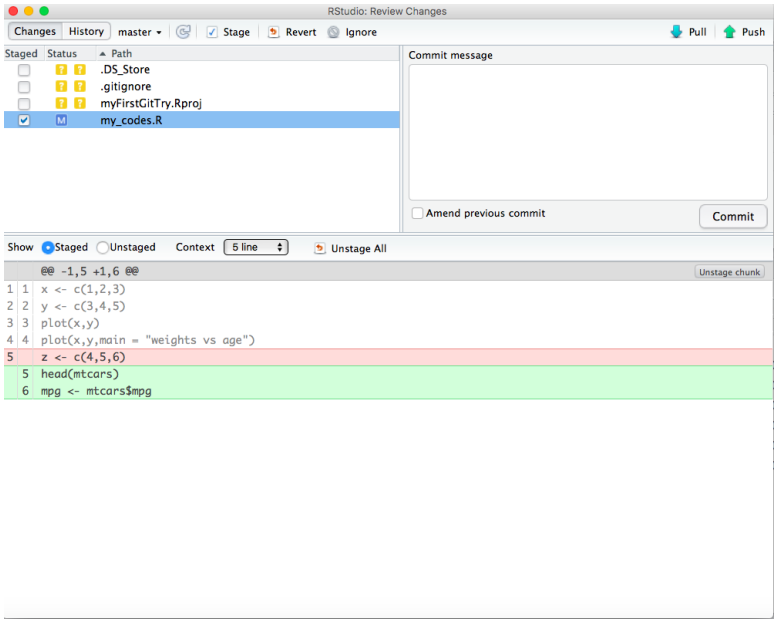
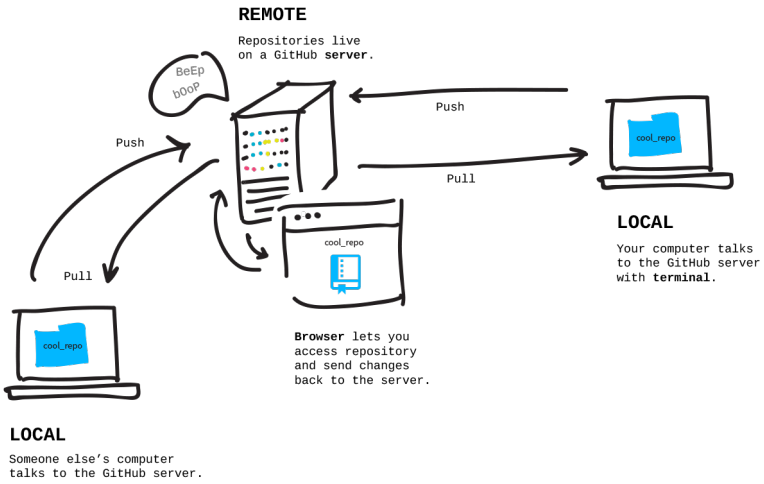


Figure 1: The red line is the line that was deleted and the green lines are the ones that are added to the file.



(From https://github.com/Rafase282/My-FreeCodeCamp-Code/wiki/Lesson-Save-your-Code-Revisions-Forever-with-Git)

Now let's see what we learned so far in practice.

Install and set up Git

Mac: Download Git from here

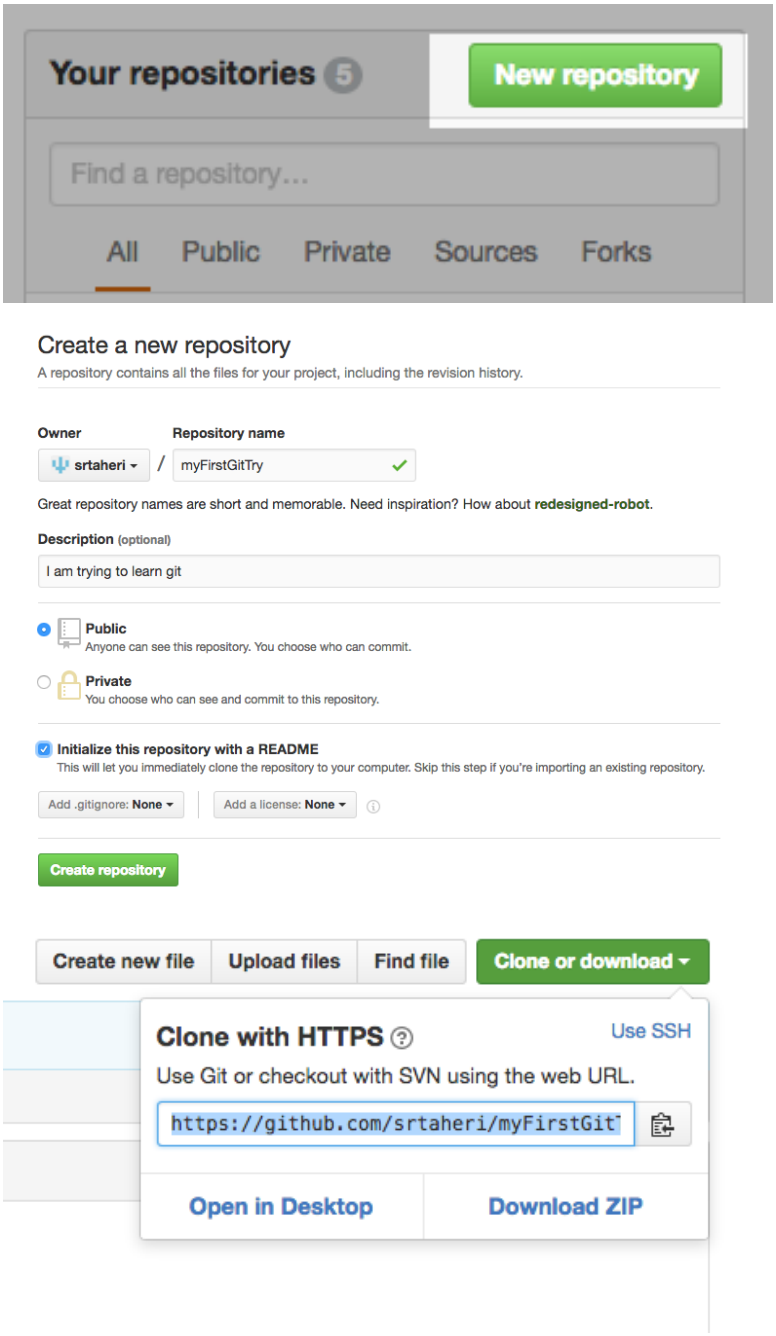
Windows: Download Git from here

Debian/Ubuntu: sudo apt-get install git-core.

Other Linux distros: Download Git from here.

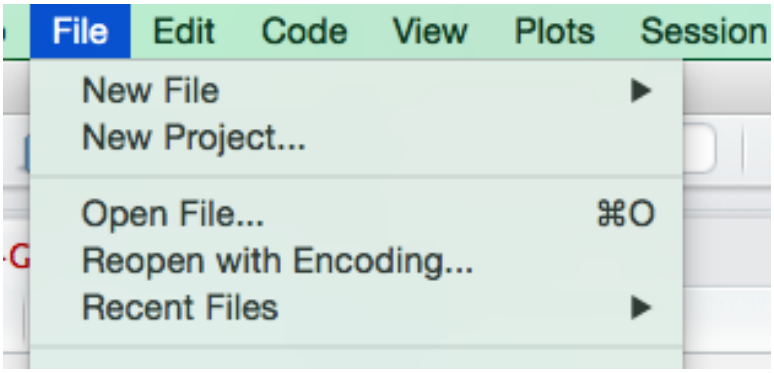
How to start using Github

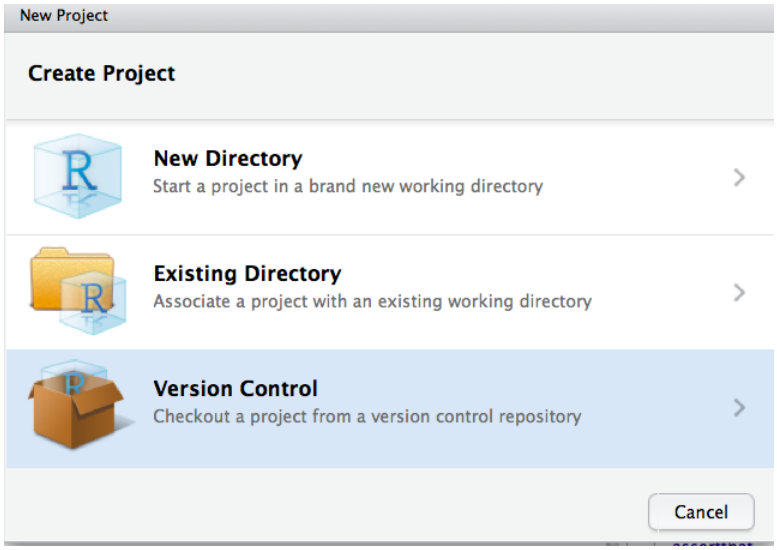
- Register for a free Github account : https://github.com
- Creat a new repository. Name it “myFirstGitTry”.
- Write a brief description (For example : “I am trying to learn git.”)
- Choose “public”
- Check the box “Initialize this repository with a README”
- Click “create repository”
- Click on “clone and download”
- Copy the “web URL”

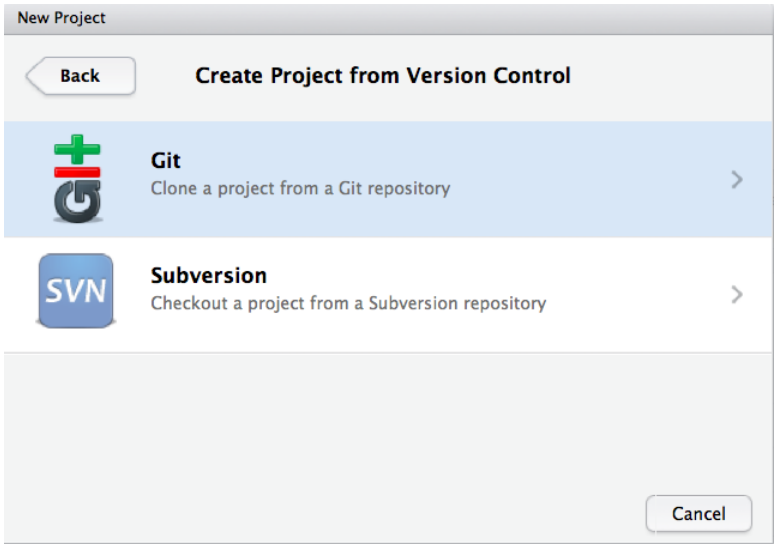


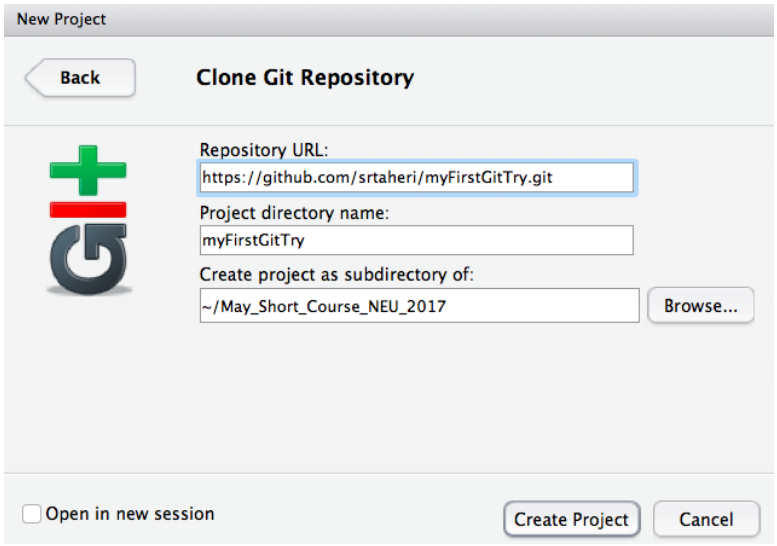
Cloning the repository through R Studio

- Go to RStudio
- File -> new project
- Click on “version control”"
- Click on “Git”(Clone a project from a repository)
- Paste the web URL in “Repository URL”.
- Browse the “Create as a subdirectory of” and create a new folder called “May_Short_Course_NEU_2017”
- Choose “open in new session”
- Click on “create project”



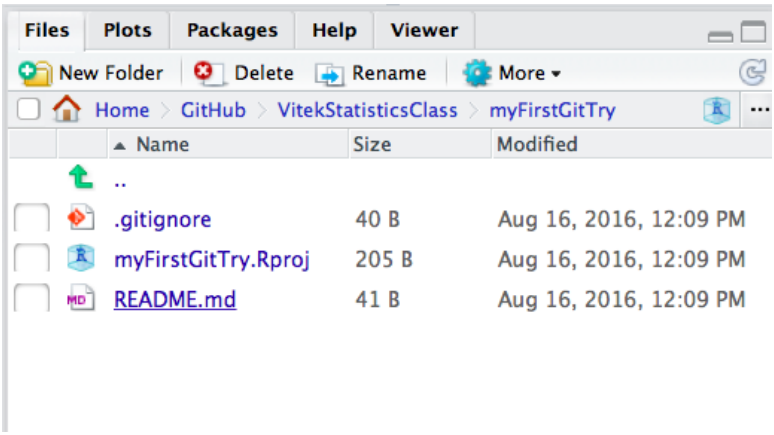


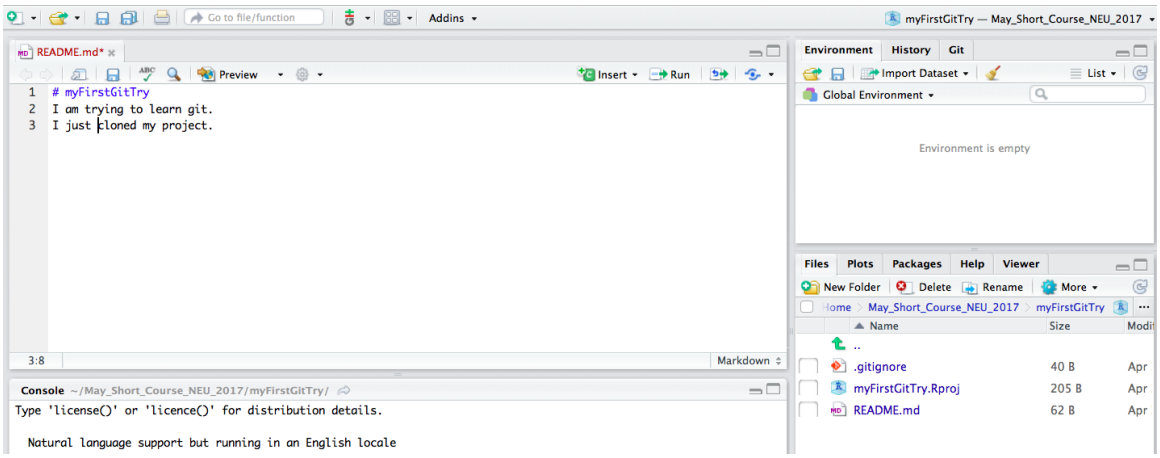


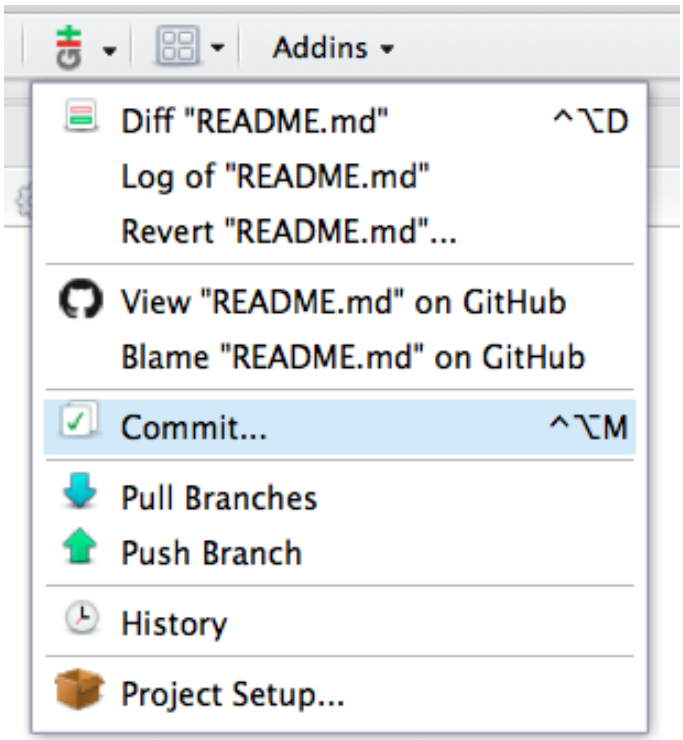


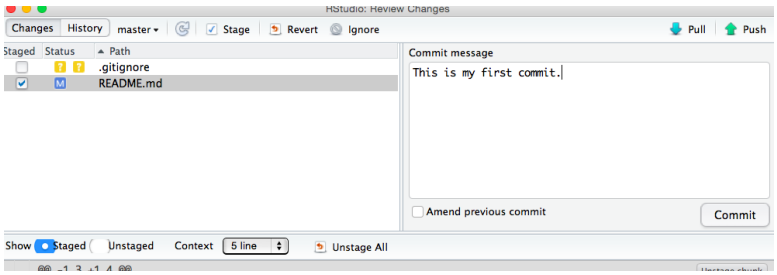
Commit a change and push it to Github server

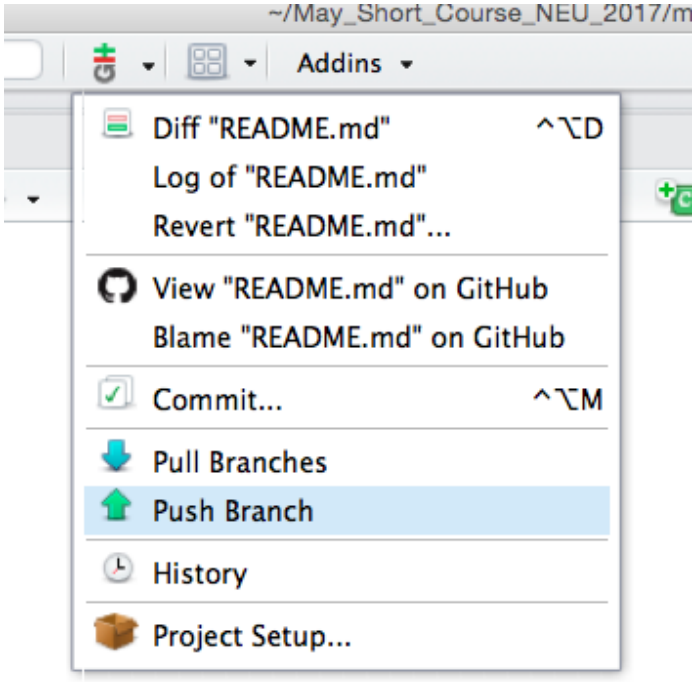
- Click on "README.md"
- Write another line of message (For example : "I cloned my project.")
- Save it (command+s in mac or Ctrl+s in windows)
- Click on the small GIT icon and choose "Commit"
- Check the icon "README" on left.
- Write a short message of what you have done so far. Click on "commit". Close the window.
- Again click on the small GIT icon and choose "Push Branch". Then close the window that appears.
- Go to the github website https://github.com, sign in and choose "myFirstGitTry" repository from bottom right.
- Your first commit is there!
- Click on README.md. You can see that your changes are successfully pushed to Github server.

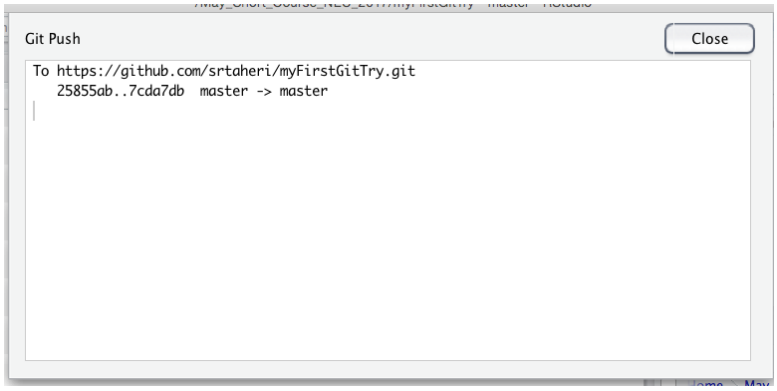

Files
Plots
Packages
Help
Viewer
New Folder
Delete
Rename
More
Home
GitHub
VitekStatisticsClass
myFirstGitTry
Name
Size
Modified
..
.gitignore
40 B
Aug 16, 2016, 12:09 PM
myFirstGitTry.Rproj
205 B
Aug 16, 2016, 12:09 PM
README.md
41 B
Aug 16, 2016, 12:09 PM

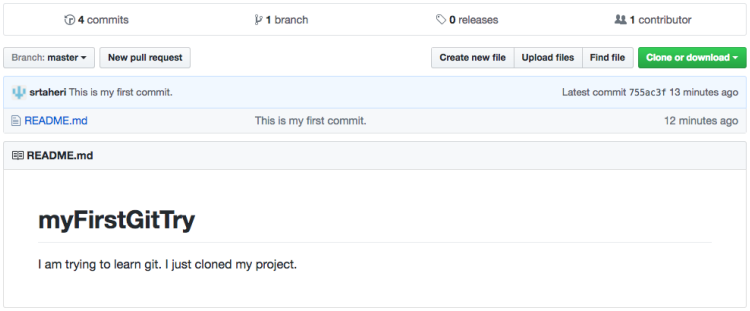

README.md*
Preview
Insert
Run
1 # myFirstGitTry
2 I am trying to learn git.
3 I just cloned my project.
3:8
Markdown
Console ~/May_Short_Course_NEU_2017/myFirstGitTry/
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
Environment
History
Git
Import Dataset
List
Global Environment
Environment is empty
Files
Plots
Packages
Help
Viewer
New Folder
Delete
Rename
More
May_Short_Course_NEU_2017
myFirstGitTry
Name
Size
..
.gitignore
40 B
myFirstGitTry.Rproj
205 B
README.md
62 B

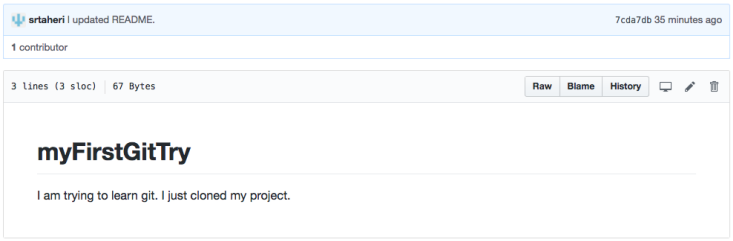

Addins
Diff "README.md"
^⌥D
Log of "README.md"
Revert "README.md"...
View "README.md" on GitHub
Blame "README.md" on GitHub
Commit...
^⌥M
Pull Branches
Push Branch
History
Project Setup...


RStudio: Review Changes
Changes
History
master
Stage
Revert
Ignore
Pull
Push
Staged
Status
Path
.gitignore
README.md
Commit message
This is my first commit.
Amend previous commit
Commit
Show
Staged
Unstaged
Context
5 line
Unstage All


~/May_Short_Course_NEU_2017/m
Addins
Diff "README.md" ^⌥D
Log of "README.md"
Revert "README.md"...
View "README.md" on GitHub
Blame "README.md" on GitHub
Commit... ^⌥M
Pull Branches
Push Branch
History
Project Setup...

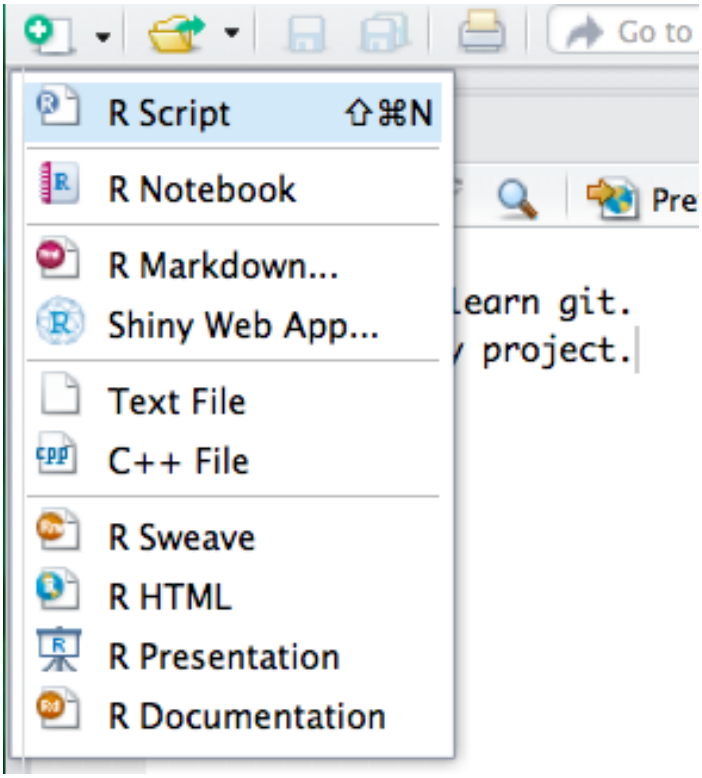

Git Push
Close
To https://github.com/srtaheri/myFirstGitTry.git
25855ab..7cda7db master -> master

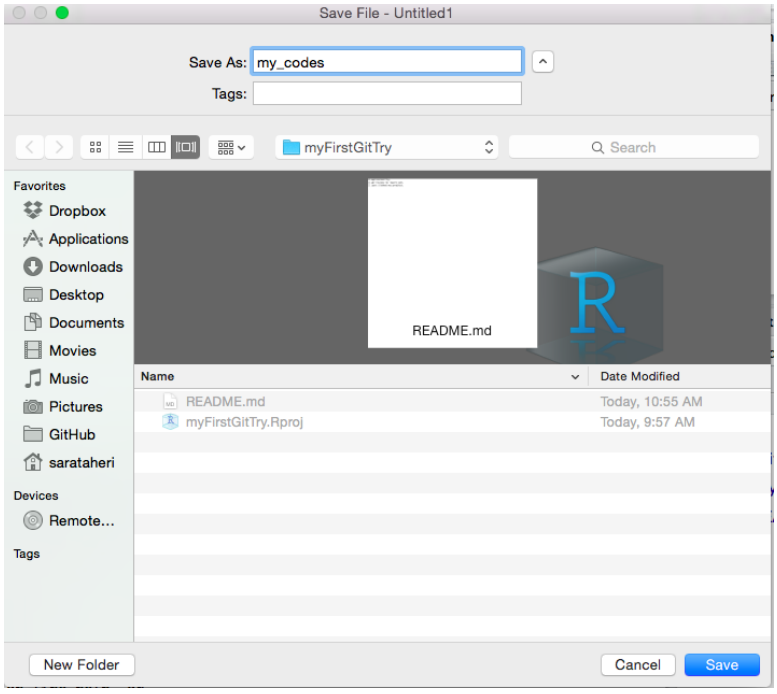


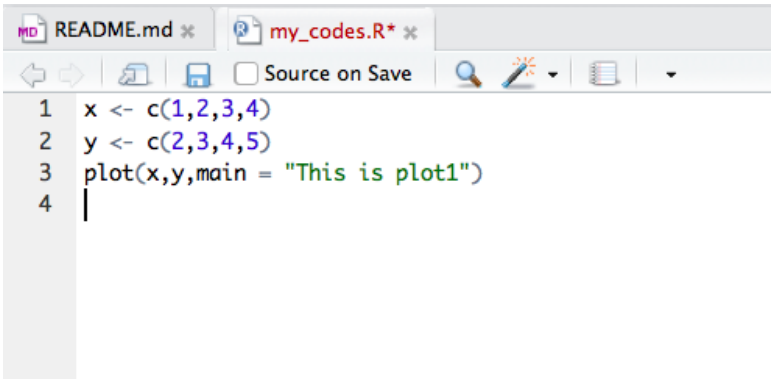


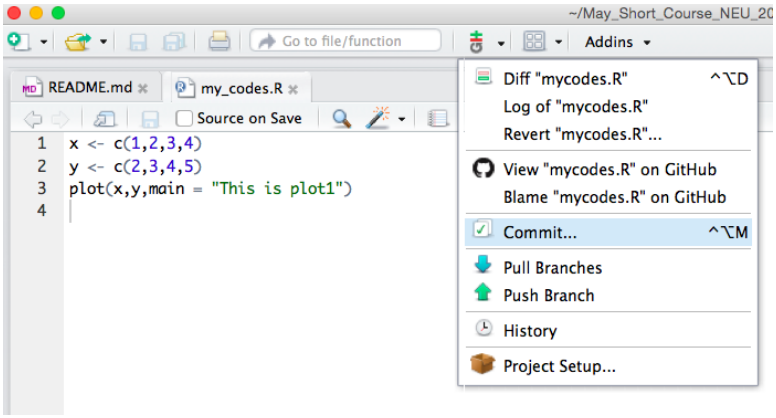
Creat a new R script

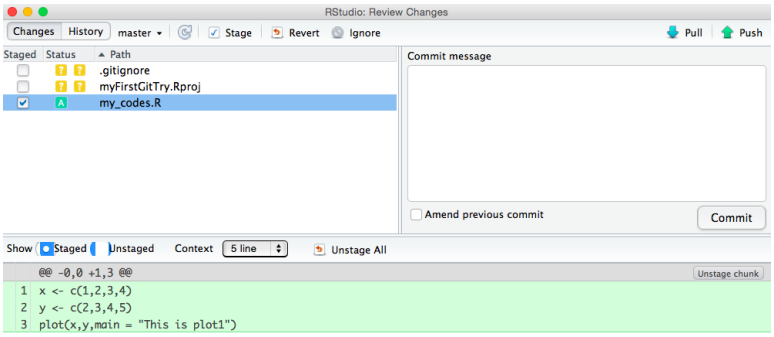
- Creat a new R script and save it with the name "my_codes" in the "myFirstGitTry" folder.
- Try writting some simple code. Save them (command+s in mac or Ctrl+s in windows).
- Click on the small GIT icon and choose "Commit"
- Check the icon "my_codes" on left.
- On the right, write some comments. For example say,"I added plot1 to the project."
- Click on the *commit* button.
- Close the 2 windows related to committing.
- Click on the small GIT icon and choose "Push Branch". Then close the window that appears.
- Go to the github website https://github.com, and choose "myFirstGitTry" repository from bottom right.
- Your second commit ("I added plot1 to the project") is there!
- Click on "my_codes.R". You can see that your changes are successfully pushed to Github server.
- Try adding more lines to your code and then practice to committ your changes and push them to github server. Do this several times.

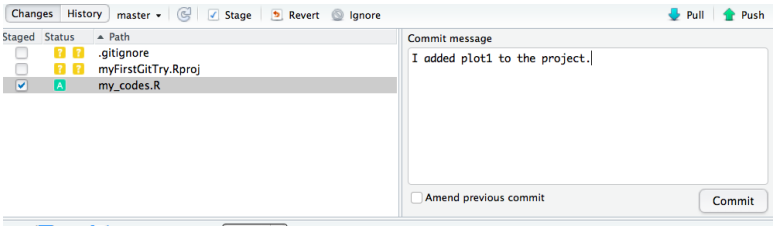

Go to
R Script ⇧⌘N
R Notebook
R Markdown...
Shiny Web App...
Text File
C++ File
R Sweave
R HTML
R Presentation
R Documentation
Pre
earn git.
project.

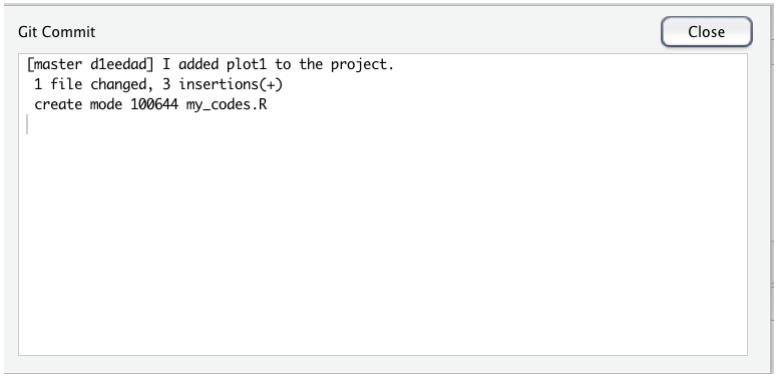

Save File - Untitled1
Save As: my_codes
Tags:
myFirstGitTry
Search
Favorites
Dropbox
Applications
Downloads
Desktop
Documents
Movies
Music
Pictures
GitHub
sarataheri
Devices
Remote...
Tags
README.md
Name
Date Modified
README.md
Today, 10:55 AM
myFirstGitTry.Rproj
Today, 9:57 AM
New Folder
Cancel
Save

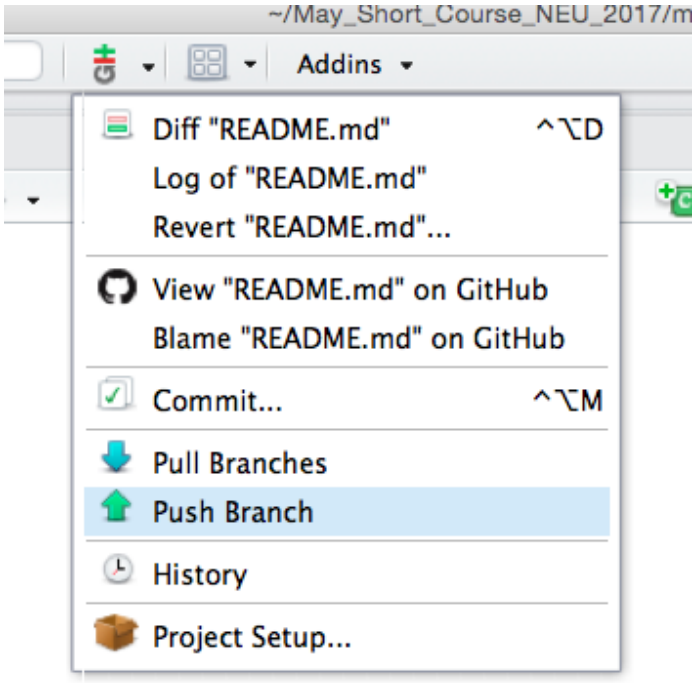

README.md
my_codes.R*
Source on Save
1 x <- c(1,2,3,4)
2 y <- c(2,3,4,5)
3 plot(x,y,main = "This is plot1")
4

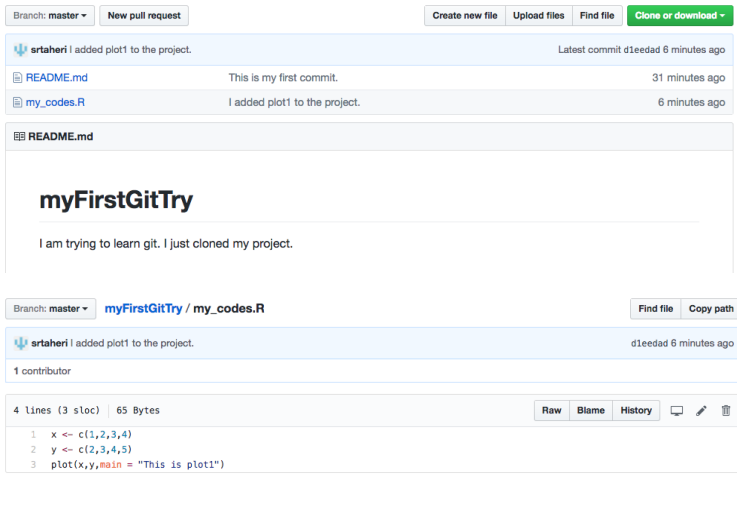

~/May_Short_Course_NEU_20
Go to file/function
Addins
README.md
my_codes.R
Source on Save
1 x <- c(1,2,3,4)
2 y <- c(2,3,4,5)
3 plot(x,y,main = "This is plot1")
4
Diff "mycodes.R" ^⌥D
Log of "mycodes.R"
Revert "mycodes.R"...
View "mycodes.R" on GitHub
Blame "mycodes.R" on GitHub
Commit... ^⌥M
Pull Branches
Push Branch
History
Project Setup...


RStudio: Review Changes
Changes
History
master
Stage
Revert
Ignore
Pull
Push
Staged
Status
Path
.gitignore
myFirstGitTry.Rproj
my_codes.R
Commit message
Amend previous commit
Commit
Show
Staged
Unstaged
Context
5 line
Unstage All
@@ -0,0 +1,3 @@
Unstage chunk
1 x <- c(1,2,3,4)
2 y <- c(2,3,4,5)
3 plot(x,y,main = "This is plot1")


Changes
History
master
Stage
Revert
Ignore
Pull
Push
Staged
Status
Path
.gitignore
myFirstGitTry.Rproj
my_codes.R
Commit message
I added plot1 to the project.
Amend previous commit
Commit


Git Commit
Close
[master d1eedad] I added plot1 to the project.
1 file changed, 3 insertions(+)
create mode 100644 my_codes.R


~/May_Short_Course_NEU_2017/m
Addins
Diff "README.md" ^⌥D
Log of "README.md"
Revert "README.md"...
View "README.md" on GitHub
Blame "README.md" on GitHub
Commit... ^⌥M
Pull Branches
Push Branch
History
Project Setup...


Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
srtaheri I added plot1 to the project.
Latest commit d1eedad 6 minutes ago
README.md
This is my first commit.
31 minutes ago
my_codes.R
I added plot1 to the project.
6 minutes ago
README.md
myFirstGitTry
I am trying to learn git. I just cloned my project.
Branch: master
myFirstGitTry / my_codes.R
Find file
Copy path
srtaheri I added plot1 to the project.
d1eedad 6 minutes ago
1 contributor
4 lines (3 sloc) 65 Bytes
Raw
Blame
History
1 x <- c(1,2,3,4)
2 y <- c(2,3,4,5)
3 plot(x,y,main = "This is plot1")

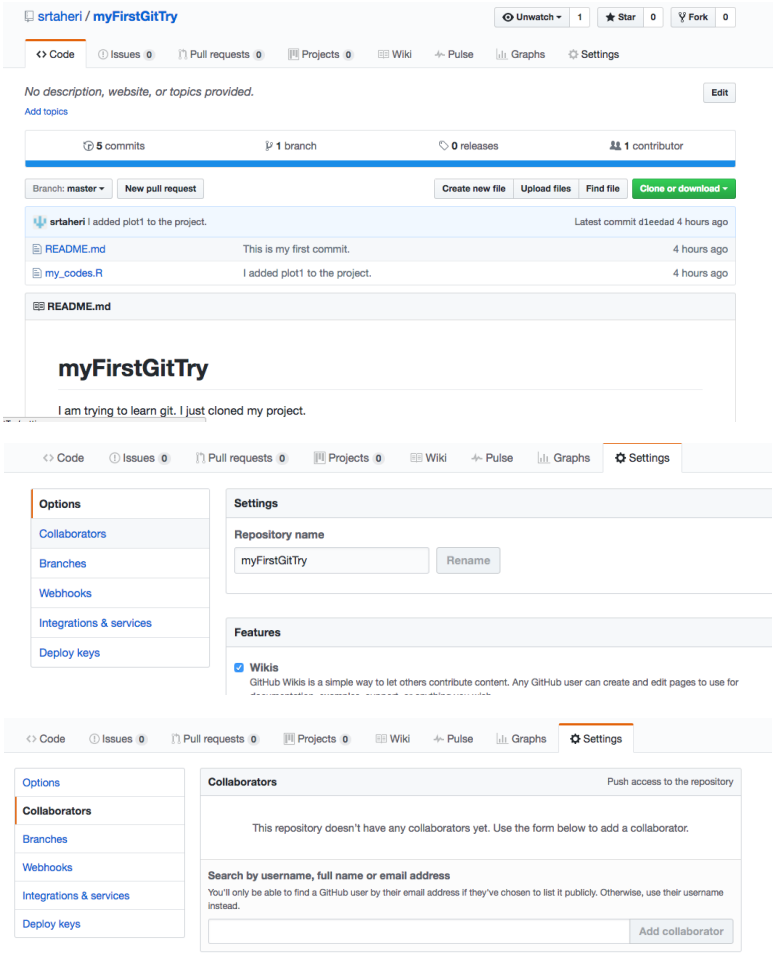
remember!

always try to commit regularly and don't forget to push!

read this and this articles for a more description about working with Git through RStudio.

Working in a team and using Git

- Go to github website, log in, and choose your project "myFirstGitTry"
- Click on "Setting".
- In the Options column at the left, click on "Collaborators".
- It may ask you to confirm your password to continue adding a collaborator.
- Write down the github username (usually email address) of your collaborator to add him/her to this project.
- Click on "Add collaborator".
- An invitation will be emailed to your collaborator. Ask him/her to check his/her inbox and accept the invitation.



Assume you and your collaborator are working on your class project. You make some changes in the project, commit the changes and "push" them to github server. Your collaborator wants to start working on the project. The first thing he/she should do is to "pull" from server. By clicking on "pull branches", he will recieve all the changes that has been made so far and his files will be updated to the latest changes. Then he can work on the project and edit it. At each step, when he/she has done an important change, he/she should "commit" the change and then "push" to server. Next time you want to start working on your project, first

you should "pull", to get the latest updates and then start making your own changes. After you are done, you should "commit", and "push" to server. This process should happen at all the time that you and your collaborators are working on the project.

When Merge conflicts happen

Merge conflicts may occur when your collaborator changes a line of the file and then commited and pushed the changes to Github and then you change exactly the same line of the file and try to commit and push your changes to Github. In this case you will receive an error. It may also occur if a file is deleted that another person is attempting to edit. The person who receives conflict error is responsible to resolve it.

Solving conflicts

We will create a scenario in which a conflict will happen and we will solve it together.

- On Github website https://github.com, choose your repository (myFirstGitTry), then click on "my_codes.R". Click on the icon that looks like a pen to edit your code. Add a line of code (like z <- c(3,4,5,6)).
- Go to the bottom of the page. Write your comments and then click on "commit changes" button.
- In R Studio, also edit "my_codes.R" and write a line of different code exactly in the same line number that you wrote a message in "my_codes.R" on github website.
- Commit your changes.
- Try pushing the change you made in "my_codes.R" in R.

You will receive an error! That's because you made changes in exactly two same places of "my_codes.R". Now git cannot understand which edit is correct and will be confused!

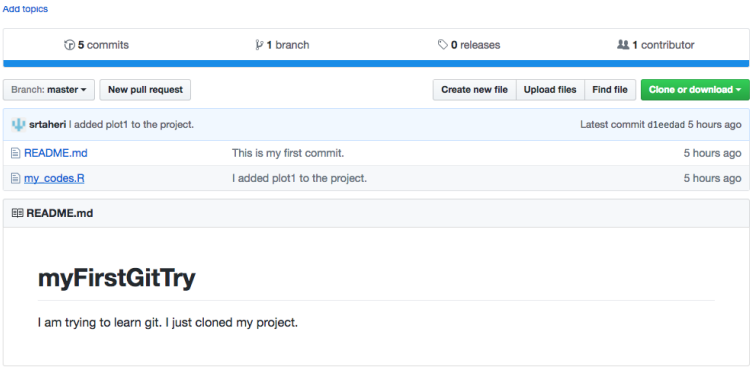
- Now try pulling.

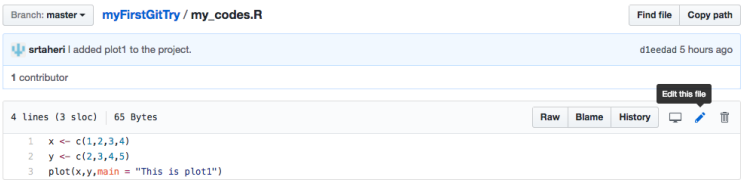
Again an error!

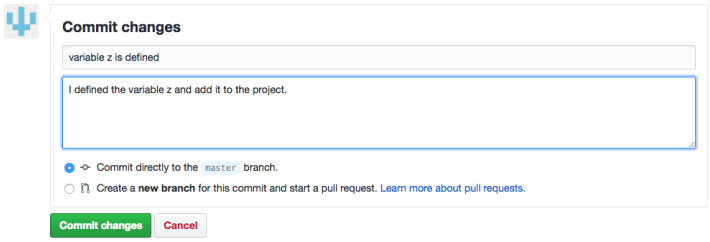
- You will see a sign like this in your my_codes.R:

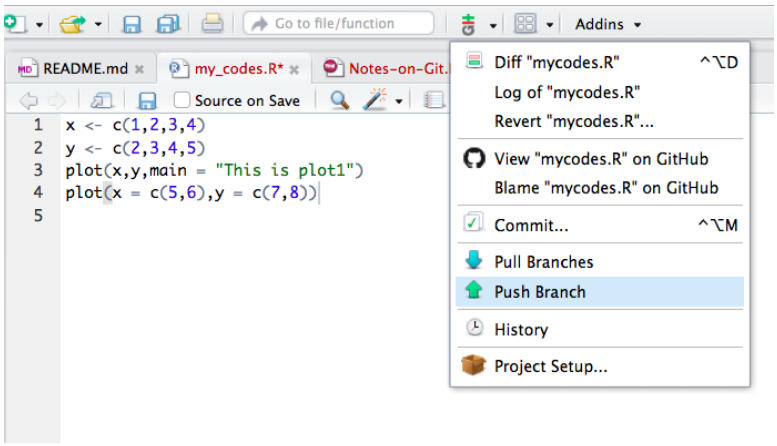
```
<<<<<<< HEAD

some codes

=======

some codes

>>>>>>> 6b501e627cb2927056a04c05d80df9c9ea64f9ae
```

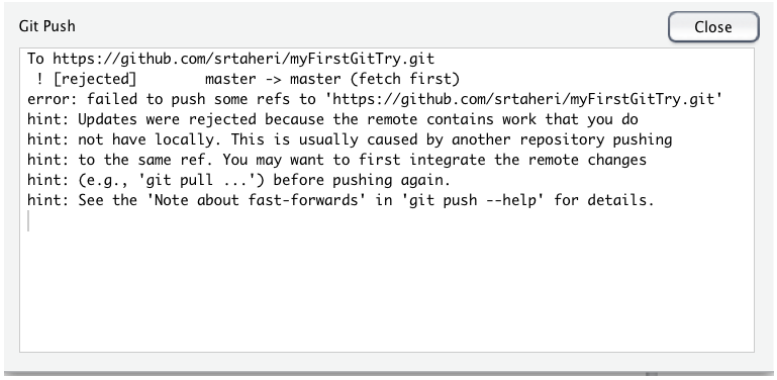
lines between these signs are the source of conflict. You can decide whether to keep the both lines or to delete one of them. Then you should delete all those signs and commit that you have solved the conflicts and then push it to server.

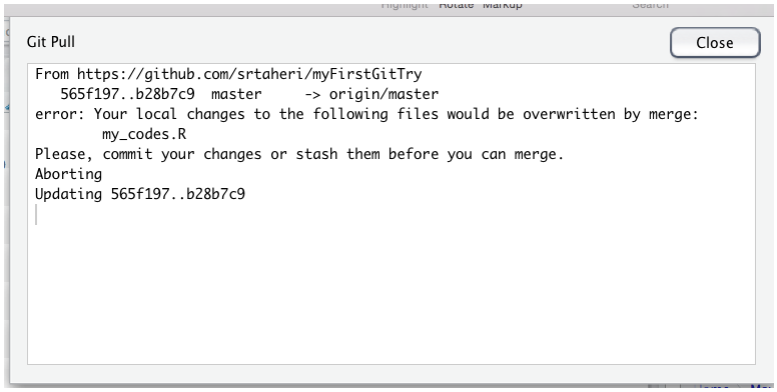

Add topics
5 commits
1 branch
0 releases
1 contributor
Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
srtaheri I added plot1 to the project.
Latest commit d1eedad 5 hours ago
README.md
This is my first commit.
5 hours ago
my_codes.R
I added plot1 to the project.
5 hours ago
README.md
myFirstGitTry
I am trying to learn git. I just cloned my project.

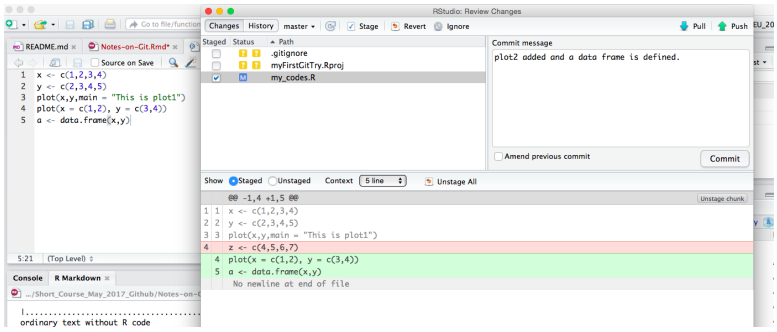

Branch: master
myFirstGitTry / my_codes.R
Find file
Copy path
srtaheri I added plot1 to the project.
d1eedad 5 hours ago
1 contributor
Edit this file
4 lines (3 sloc)
65 Bytes
Raw
Blame
History
x <- c(1,2,3,4)
y <- c(2,3,4,5)
plot(x,y,main = "This is plot1")

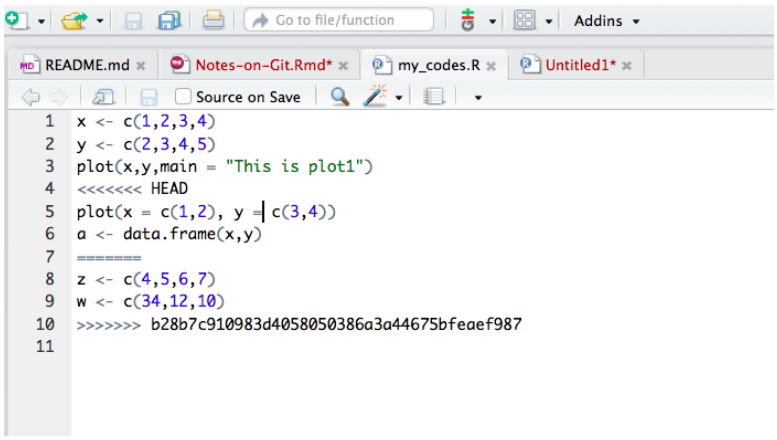

Commit changes
variable z is defined
I defined the variable z and add it to the project.
Commit directly to the master branch.
Create a new branch for this commit and start a pull request. Learn more about pull requests.
Commit changes
Cancel

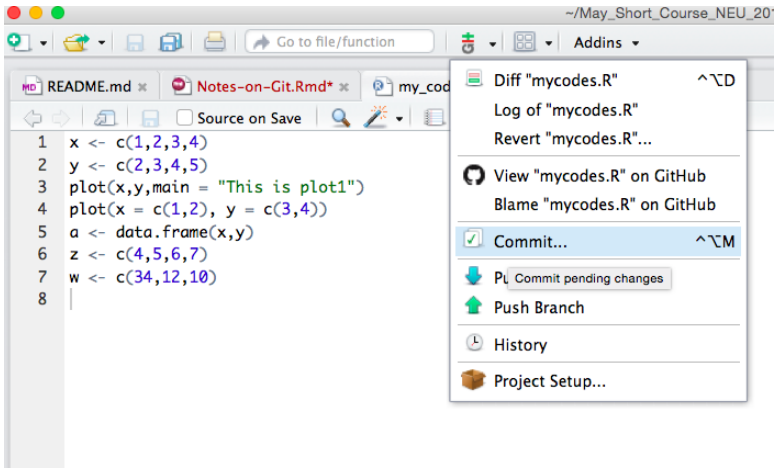

Go to file/function
Addins
README.md
my_codes.R*
Notes-on-Git.
Source on Save
1 x <- c(1,2,3,4)
2 y <- c(2,3,4,5)
3 plot(x,y,main = "This is plot1")
4 plot(x = c(5,6),y = c(7,8))
5
Diff "mycodes.R" ^⌥D
Log of "mycodes.R"
Revert "mycodes.R"...
View "mycodes.R" on GitHub
Blame "mycodes.R" on GitHub
Commit... ^⌥M
Pull Branches
Push Branch
History
Project Setup...

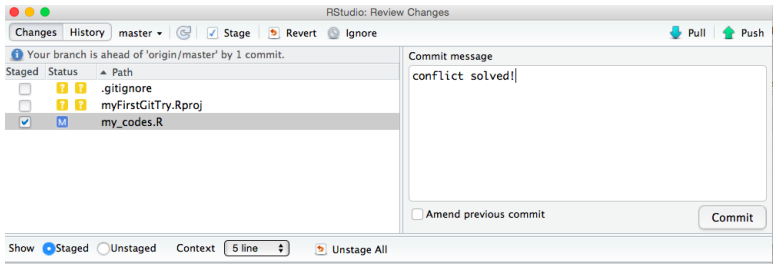

Git Push
Close
To https://github.com/srtaheri/myFirstGitTry.git
! [rejected] master -> master (fetch first)
error: failed to push some refs to 'https://github.com/srtaheri/myFirstGitTry.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.


Git Pull
Close
From https://github.com/srtaheri/myFirstGitTry
565f197..b28b7c9 master -> origin/master
error: Your local changes to the following files would be overwritten by merge:
my_codes.R
Please, commit your changes or stash them before you can merge.
Aborting
Updating 565f197..b28b7c9


RStudio: Review Changes
Changes
History
Commit message
plot2 added and a data frame is defined.
Commit
README.md
Notes-on-Git.Rmd*
my_codes.R
Source on Save


Go to file/function
Addins
README.md
Notes-on-Git.Rmd*
my_codes.R
Untitled1*
Source on Save
x <- c(1,2,3,4)
y <- c(2,3,4,5)
plot(x,y,main = "This is plot1")
<<<<<<< HEAD
plot(x = c(1,2), y = c(3,4))
a <- data.frame(x,y)
=======
z <- c(4,5,6,7)
w <- c(34,12,10)
>>>>>>> b28b7c910983d4058050386a3a44675bfeaef987





The other way of working with git is by using **github desktop**. It has a beautiful and friendly environment. you can download it here.

References

- https://www2.stat.duke.edu/courses/Fall15/sta112.01/post/slides/deck2.html#22
- https://jahya.net/blog/git-vs-github/
- http://r-bio.github.io/intro-git-rstudio/