

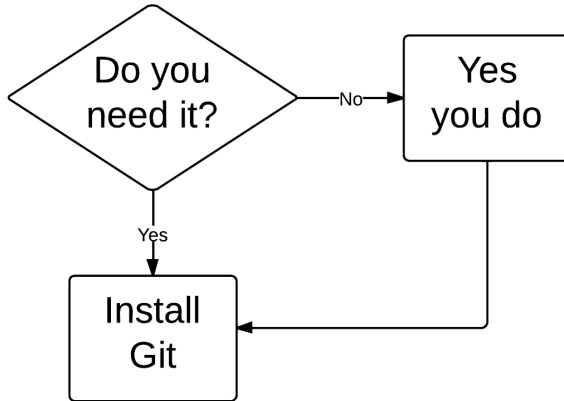
Reproducible research #2

git

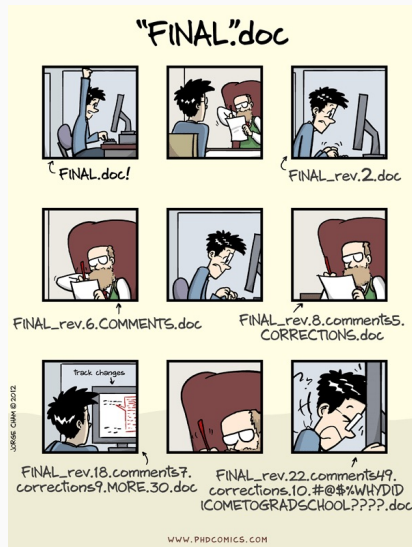
Git is a version control system.

It saves information about all changes you make on all files in a repository. This allows you to revert back to previous versions and search through the history for all files in the repository.

Version Control Flowchart



Source: <https://ali-dev.medium.com/useful-git-commands-to-get-things-done-56273e3c9746>



Source: PhDComics

Git is open source. You can download it for different operating systems here:

<https://git-scm.com/downloads>

You will need git on your computer to use git with RStudio and create local git repositories you can sync with GitHub repositories.

Sometimes, RStudio will automatically find git (once you've installed git) when you start RStudio.

However, in some cases, you may need to take some more steps to activate git in RStudio. To do this, go to "RStudio" -> "Preferences" -> "Git/SVN". Choose "Enable version control". If RStudio doesn't find your version of git in the "Git executable" box, browse for it.

Before you use git, you should configure it. For example, you should make sure it has your name and email address.

You can configure git with commands at the shell. For example, I would run the following code at a shell to configure git to have my proper user name and email:

```
git config --global user.name "Brooke Anderson"  
git config --global user.email "brooke.anderson@colostate.edu"
```


Initializing a git repository

You can initialize a git repository for a directory that is an R Project directory through R Studio.

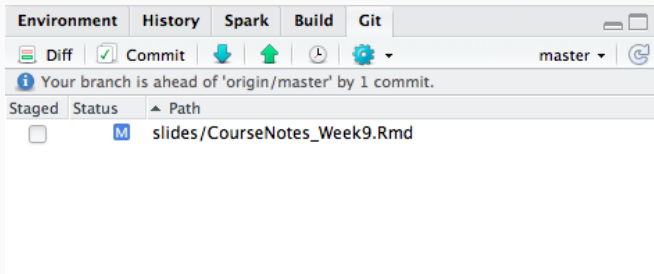
1. Open the Project.
2. Go to “Tools” -> “Version Control” -> “Project Setup”.
3. In the box for “Version control system”, choose “Git”.

Note: If you have just installed git, and have not restarted RStudio, you'll need to do that before RStudio will recognize git. If you do not see “Git” in the box for “Version control system”, it means either that you do not have git installed on your computer or that RStudio was unable to find it.

Initializing a git repository

Once you initialize the project as a git repository, you should have a “Git” window in one of your RStudio panes (top right pane by default).

As you make and save changes to files, they will show up in this window for you to commit. For example, this is what the Git window for our coursebook looks like when I have changes to the slides for week 9 that I need to commit:



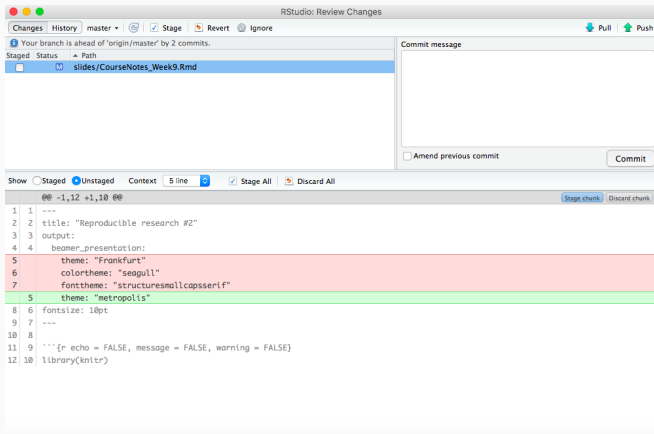
Committing

When you want git to record changes, you *commit* the files with the changes. Each time you commit, you have to include a short commit message with some information about the changes.

You can make commits from a shell. However, in this course we'll just make commits from the RStudio environment. Each time you commit, you are saving a record of the change **locally** (i.e., on your computer). To send this change to other remote repositories, you'll need to link with **remote** repositories and then **push** and **pull** your commits.

Committing

To make a commit from RStudio, click on the “Commit” button in the Git window. That will open a separate commit window that looks like this:



Committing

In this window, to commit changes:

1. Click on the files you want to commit to select them.
2. If you'd like, you can use the bottom part of the window to look through the changes you are committing in each file.
3. Write a message in the "Commit message" box. Keep the message to one line in this box if you can. If you need to explain more, write a short one-line message, skip a line, and then write a longer explanation. (Think of this like an email, where the first line is the subject line and then the rest is the body of the email.) Write the message in the imperative voice ("Add Rmarkdown file", "Change code for cleaning data").
4. Click on the "Commit" button on the right.

Once you commit changes to files, they will disappear from the Git window until you make and save more changes in them.

Committing

	COMMENT	DATE
○	CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
○	ENABLED CONFIG FILE PARSING	9 HOURS AGO
○	MISC BUGFIXES	5 HOURS AGO
○	CODE ADDITIONS/EDITS	4 HOURS AGO
○	MORE CODE	4 HOURS AGO
○	HERE HAVE CODE	4 HOURS AGO
○	AAAAAAAAA	3 HOURS AGO
○	ADKFJSLKDFJSDKLFJ	3 HOURS AGO
○	MY HANDS ARE TYPING WORDS	2 HOURS AGO
○	HAAAAAAAAAANDS	2 HOURS AGO

AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

Source: xkcd

Browsing history

On the top left of the Commit window, you can toggle to “History”. This window allows you to explore the history of commits for the repository.

The screenshot shows the RStudio 'Review Changes' window. At the top, there's a tab labeled 'History' with a dropdown menu showing 'master' and '(all commits)'. Below this is a table of commit history. The table has columns for Subject, Author, Date, and SHA. The commit 'origin/master origin/HEAD Make a small change to in-course exercise' is selected, highlighted in blue. Below the table, there's a section for the selected commit, showing the file '09-reproduciblesearch2.Rmd' and a diff view. The diff view shows changes to the file, with line numbers on the left and the diff content on the right. The diff content includes instructions for setting up an R Project, downloading FARS data, and linking the project with a GitHub repository.

Subject	Author	Date	SHA
data-rw: reproduciblesearch2.Rmd Move some figure files	Brooke Anderson <brooke.anderson@colo	2017-10-16	20e32e94
Add in some figures for slides	Brooke Anderson <brooke.anderson@colo	2017-10-16	58f43de6
origin/master origin/HEAD Make a small change to in-course exercise	Brooke Anderson <brooke.anderson@colo	2017-10-16	2293e8f8
Try adding coursebook material for week 9	Brooke Anderson <brooke.anderson@colo	2017-10-16	e4201f2
Some final changes	Brooke Anderson <brooke.anderson@colo	2017-10-11	5f633f71
Try adding back in choroplethr code	Brooke Anderson <brooke.anderson@colo	2017-10-11	d8cf0aaa
A bit more work on in-course exercise	Brooke Anderson <brooke.anderson@colo	2017-10-11	5460160b

Commits 1-100 of 584

09-reproduciblesearch2.Rmd

09-reproduciblesearch2.Rmd View file @ 2293e8f8

```
@@ -399,7 +399,7 @@ In this part of the group exercise, you will set up an R Project to use for the
399 399 - The "data-rw" directory will ultimately have your raw data as well as some R scripts with code for cleaning up the raw
data. The homework requires you to pull FARS data from several years. Create a subdirectory in "data-rw" that will just
have that data. In the "Files" pane in RStudio, navigate into the "data-rw" subdirectory. Use the "New Folder" button to
create a new subdirectory within the "data-rw" subdirectory. Name it "yearly_person_data".
400 400 - Download FARS data from the years 1999 to 2010. From each year, pull out the "person" file. Save these yearly "person"
files in the "yearly_person_data" subdirectory you created. As a file name, use "person_" and then the year. For example, if
you are saving this file for 1999 in the form of a csv, you would name the file "person_1999.csv".
401 401 - The "writing" subdirectory will have your R Markdown file and its output. Create a new R Markdown file ("File" -> "New
File" -> "R Markdown") and save it to this subdirectory. You can change the name and date for the file if you'd like. Delete
all the text that comes as a default. Write a piece of code that lists the files you saved in "data-rw/yearly_person_data".
Remember that the working directory for an R Markdown file is the directory in which it's saved, so you'll need to use a
relative pathname that goes up one directory ("..") and then goes into "data-rw" and "yearly_person_data".
402 - If you have time, go to the [FARS documentation](https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812316) and
find out more about which variables are included in this data set and which values they can have.
402 - If you have time, go to the FARS documentation that you found in an earlier in-course exercise.
403 403 ### Linking your project with a GitHub repository
404 404
405 405
```