

# Intro to git and GitHub

*Daniel Anderson*  
Week 4, Class 2



Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

## Agenda

- Questions
- Intro to *git/GitHub*
- Lab

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

2 / 43

## Agenda

- Questions
- Intro to *git/GitHub*
- Lab

### *Learning objectives for today*

- Understand the basics of git and git vocabulary
- Be able to create repos, push projects, commit changes
- Begin collaborating on a project

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

2 / 43

What questions do you have?

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

3 / 43



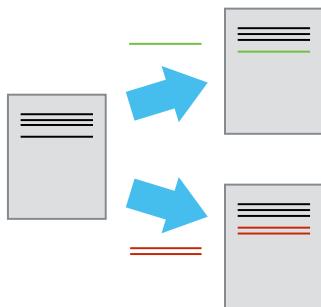
*From swcarpentry*



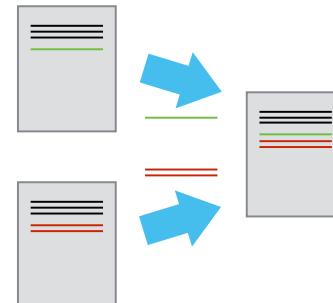
We can think of the changes as separate from the document

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

5 / 43



This means there are many possible versions of the same document



Unless there are conflicts, two changes from the same document can be merged together

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

7 / 43

# How?

That's what we'll talk about today!

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

8 / 43

# How?

That's what we'll talk about today!

- Get you to understand the structure of git
- Actual method of completing the tasks is up to you
  - GUI? Command line? RStudio?
- If you go with a GUI, I'd recommend [GitKraken](#)
  - In fact, I'd recommend everybody download/use it, even if for merge conflicts alone.

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

8 / 43

# How?

That's what we'll talk about today!

- Get you to understand the structure of git
- Actual method of completing the tasks is up to you
  - GUI? Command line? RStudio?

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

8 / 43

# Some basic terminology

- Version Control System
  - A tool to help us track changes. *git* is one such system (but there are others).

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

9 / 43

## Some basic terminology

- Version Control System
  - A tool to help us track changes. *git* is one such system (but there are others).
- Commit
  - Changes that have been made to the file(s)

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

9 / 43

## Some basic terminology

- Version Control System
  - A tool to help us track changes. *git* is one such system (but there are others).
- Commit
  - Changes that have been made to the file(s)
- Repository (repo)
  - The files, full commit history, and associated metadata for a project

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

9 / 43

## Some basic terminology

- Version Control System
  - A tool to help us track changes. *git* is one such system (but there are others).
- Commit
  - Changes that have been made to the file(s)
- Repository (repo)
  - The files, full commit history, and associated metadata for a project

### When working with *git*

- Each collaborator has the *entire* repo on their local machine
- There is also a cloud-based server hosting the repo (that's GitHub)
- The online version of the repo is called the *remote*

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

9 / 43

## Some key points

- Because *git* tracks the **entire** history of a project it is akin to unlimited "undo"
- *git* allows many people to work in parallel

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

10 / 43

# Creating a repository

- A couple of different options...

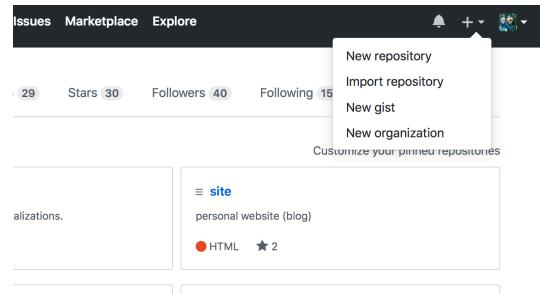
Slides available at: <http://www.datolorax.com/vita/ds/ds1-slides/w4p2/>

11 / 43

# GitHub

Go to GitHub

- Select drop down by your profile, then "New Repository"
- I'd suggest also adding the R .gitignore and a README



Slides available at: <http://www.datolorax.com/vita/ds/ds1-slides/w4p2/>

12 / 43

# Creating a repository

- A couple of different options...

*Rstudio*

Let's focus on the RStudio version first.

Slides available at: <http://www.datolorax.com/vita/ds/ds1-slides/w4p2/>

11 / 43

# Name it

## Create a new repository

A repository contains all the files for your project, including the revision history.

Owner  /

Great repository names are short and memorable. Need inspiration? How about [super-sniffle](#)

Description (optional)

**Public** Anyone 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 let you immediately clone the repository to your computer. Skip this step if you're importing an

Slides available at: <http://www.datolorax.com/vita/ds/ds1-slides/w4p2/>

13 / 43

# Copy path

2 Wiki Insights

in R <http://ggplot2.tidyverse.org>

Push this button to copy the path

24 releases 157 contributors GPL-2.0

Create new file Upload files Find file Clone or download

Clone with HTTPS Use SSH

https://github.com/tidyverse/ggplot2

Open in Desktop Download ZIP

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

14 / 43

## Create a new RStudio Project

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

15 / 43

New Project

Back Create Project from Version Control

Git  
Clone a project from a Git repository

SVN  
Checkout a project from a Subversion repository

Cancel

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

16 / 43

RStudio File Edit Code View Plots Session Build Debug Profile Tools Window Help

Console Terminal R Markdown

.../w4p2.Rmd

label: unnamed-chunk-1  
List of 2  
\$ eval : logi FALSE  
\$ engine: chr "bash"  
|.....  
ordinary text without  
label: unnamed-chunk-2  
List of 2  
\$ eval : logi FALSE  
\$ engine: chr "bash"  
|.....  
ordinary text without

~/usr/local/bin/pandoc --  
re�iseasciI\_idendifie  
cation none -V "mathjax"  
L\_AM\_CHTML" -V "title-s  
tion-divs --template /L  
an/rmarkdowm/templates/  
intro.html --mathjax /var/  
tr17674475708d9.htm  
T/Rtmp0R0D0/xaringan1  
9x4565c4w0000gn/T/Rtmp0R0D0/xaringan17674607cf09e.js --variable title-slide=true --vari  
able math=true  
output file: w4p2.knit.md

Output created: w4p2.html

New Project

Back Clone Git Repository

Repository URL:  
https://github.com/datalorax/esvis.git

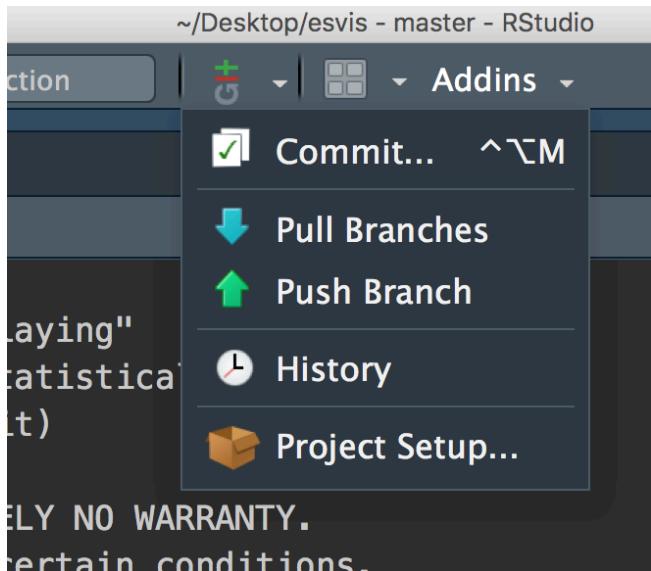
Project directory name:  
esvis

Create project as subdirectory of:  
~/Desktop

Open in new session Create Project Cancel

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

17 / 43



Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

18 / 43

## [demo]

Let's look at the git plugin

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

19 / 43

## Tracking

Notice the hidden `.git` folder

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

20 / 43

## Tracking

Notice the hidden `.git` folder

On mac, show hidden files with

`cmd + shift + .`

(Note, I have hidden files shown all the time)

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

20 / 43

# Tracking

Notice the hidden `.git` folder

On mac, show hidden files with

`cmd + shift + .`

(Note, I have hidden files shown all the time)

- As long as that folder exists, the **entire** folder will be tracked.
  - **Do not create repos inside of repos**
  - If you want a folder to show up, there needs to be something in it (e.g., a `README.md`)

# Ignoring Files

- When we initialized the repo, we started it with a `.gitignore` file
- The `.gitignore` file tells the repo not to track certain files
  - e.g., proprietary data

# Ignoring Files

- When we initialized the repo, we started it with a `.gitignore` file
- The `.gitignore` file tells the repo not to track certain files
  - e.g., proprietary data

Probably not super important for now, but if you need to ignore a file in your repo for whatever reason, just put the name of the file in the `.gitignore`

# Adding a file

- Let's create an R Markdown file and put it in our repo.
- Open terminal, type `git status`. What do you see?
- Use the RStudio `git` plugin. What do you see?

# Staging

- When you add files to a repo you are *staging* them for tracking.

[demo gitkraken & RStudio staging]

# Staging

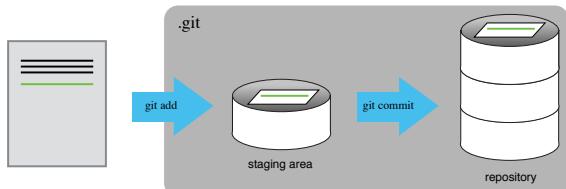
- When you add files to a repo you are *staging* them for tracking.

[demo gitkraken & RStudio staging]

Command line version is `git add <file>`

# Commits

After staging, you *commit* changes to the file



# Push

After you've committed the changes you want, push them to the remote

## Push

After you've committed the changes you want, push them to the remote



Wait what's a remote again?

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

25 / 43

## Push

After you've committed the changes you want, push them to the remote



Wait what's a remote again?

The cloud-based repo

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

25 / 43

## Refresh the repo

datalorax / myrepo

Code Issues Pull requests Projects Wiki Insights Settings

No description, website, or topics provided.

Manage topics

1 commit 1 branch 0 releases 1 contributor

Branch: master New pull request

Create new file Upload files Find file Clone or download

datalorax initial commit Latest commit b947f39 35 seconds ago

example.Rmd initial commit 35 seconds ago

Add a README

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

26 / 43

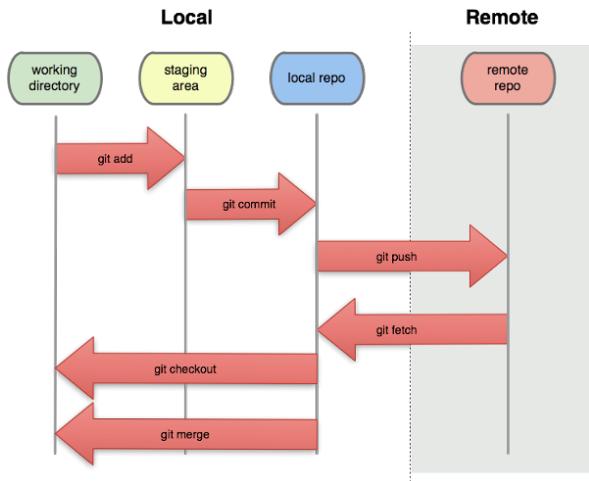
## gitkraken approach



Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

27 / 43

## An overall view



Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

28 / 43

## Revisiting some vocabulary

- stage
- commit
- push
- pull
- clone

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

29 / 43

## Revisiting some vocabulary

- stage
- commit
- push
- pull
- clone

At least half the difficulty with *git* is all the weird words

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

29 / 43

### *Simple git through command line*

90% of your workflow requires just a few commands.

- **git clone <path>**
  - Used once - gets the remote repo on your local.

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

30 / 43

## *Simple git through command line*

90% of your workflow requires just a few commands.

- `git clone <path>`
  - Used once - gets the remote repo on your local.
- `git pull`
  - First command each time - syncs your local repo with the remote

Slides available at:<http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

30 / 43

## *Simple git through command line*

90% of your workflow requires just a few commands.

- `git clone <path>`
  - Used once - gets the remote repo on your local.
- `git pull`
  - First command each time - syncs your local repo with the remote
- `git add`
  - Stages files in your local repo to be tracked

Slides available at:<http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

30 / 43

## *Simple git through command line*

90% of your workflow requires just a few commands.

- `git clone <path>`
  - Used once - gets the remote repo on your local.
- `git pull`
  - First command each time - syncs your local repo with the remote
- `git add`
  - Stages files in your local repo to be tracked
- `git commit <file> -m "my commit message"`
  - Commits changes to files in the repository, along with a commit message

Slides available at:<http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

30 / 43

## *Simple git through command line*

90% of your workflow requires just a few commands.

- `git clone <path>`
  - Used once - gets the remote repo on your local.
- `git pull`
  - First command each time - syncs your local repo with the remote
- `git add`
  - Stages files in your local repo to be tracked
- `git commit <file> -m "my commit message"`
  - Commits changes to files in the repository, along with a commit message
- `git push`
  - Pushes the changes you've made from your local repo to the remote

Slides available at:<http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

30 / 43

## Simple git through command line

90% of your workflow requires just a few commands.

- `git clone <path>`
  - Used once - gets the remote repo on your local.
- `git pull`
  - First command each time - syncs your local repo with the remote
- `git add`
  - Stages files in your local repo to be tracked
- `git commit <file> -m "my commit message"`
  - Commits changes to files in the repository, along with a commit message
- `git push`
  - Pushes the changes you've made from your local repo to the remote
- `git status`
  - Gives you an update on your local repo (what files still need to be added, what changes have been made since last commit, etc.)

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

30 / 43

## Collaborating

Need to get proper permissions. Two methods:

- Add users to a repo: Settings => Collaborators => add user w/ username

A screenshot of a GitHub repository settings page. At the top, there are buttons for Unwatch, Star, Fork, and Insights. Below that is a navigation bar with Code, Issues, Pull requests, Projects, Wiki, and Insights. The main area has tabs for Options, Collaborators, Branches, Webhooks, Integrations & services, Deploy keys, Moderation, and Interaction limits. The Collaborators tab is selected and highlighted with a pink arrow. To its left is another pink arrow pointing to the 'Collaborators' link in the navigation bar. The right side of the page shows a message: 'This repository doesn't have any collaborators yet. Use the form below to add a collaborator.' There is a search bar labeled 'Search by username, full name or email address' and a button labeled 'Add collaborator'.

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

31 / 43

## Collaborating

Second method:

- Create an organization

A screenshot of a GitHub user profile. At the top, there is a dark header with a bell icon, a plus sign, and a profile picture. Below it, the user's name is followed by a 'Following' button with the number '15'. A pink arrow points to a dropdown menu that is open. The menu contains options: 'New repository', 'Import repository', 'New gist', and 'New organization'. The 'New organization' option is highlighted with a pink box and a pink arrow pointing to it. Below the menu, there is a section titled 'Customize your pinned repositories'.

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

32 / 43

## Cloning

Means you're downloading the repo

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

33 / 43

# Cloning

Means you're downloading the repo

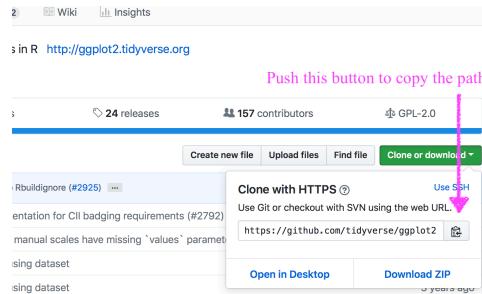
- Can also start a new RStudio project

# Cloning

Means you're downloading the repo

- Can also start a new RStudio project

```
git clone <path>
```



Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

33 / 43

# Merge conflicts

- Remember to always `git pull` first
- Let's create a merge conflict!

(Note the GUI from *gitkraken* is really nice for resolving them)

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

34 / 43

# Merge conflict activity

1. Get a partner
2. Partner1 add Partner2 as a collaborator
3. Partner2 clone the repo
4. Partner1 make changes to the file in the repo, commit, and push
5. Partner2 make different changes to the same parts of the file. Commit the changes. Try to push. What happens? Why?
6. Open up the files - do you see some weird stuff that's been added?

*Wait for next steps*

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

35 / 43

## Merge conflicts w/gitkraken

GitKraken Tutorial: Merge Conflict Tool



Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

36 / 43

## Some takeaways

Basic workflow is

- Create a repo
- Add collaborators
- Add files
- Commit changes
- Push changes to remote

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

37 / 43

## Advice on avoiding merge conflicts

- Always `git pull` (or the GUI equivalent) before you start working
- Try to stay in communication and don't work on the **exact** same thing at the same time
- Use a GUI to help you manage what merge conflicts you do run into

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

38 / 43

## Remember, today is an intro

We'll talk more about other `git` features later

- branching
- forking
- pull requests
- filing issues
- etc.

Slides available at <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

39 / 43

# Lab

- For lab today, just submit a link to the repo you created
- If we still have time, work with your group to get a shared repo
  - Discuss organization
  - Maybe get your outline uploaded?
- Chat with me before you leave if you're still confused (note, I expect you to be at least a little confused)

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

40 / 43

# Additional info

- Initializing a local repo through the command line

```
mkdir newrepo  
cd newrepo  
git init
```

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

41 / 43

# Additional info

- Initializing a local repo through the command line

```
mkdir newrepo  
cd newrepo  
git init
```

You should get a message similar to **Initialized empty Git repository in /Users/Daniel/newrepo/.git/**

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

41 / 43

# Name it

Doesn't have to be the same name as the folder on your local, but probably a good idea.

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: datalorax / Repository name: myrepo

Great repository names are short and memorable. Need inspiration? How about [super-sniffle](#)?

Description (optional):

Public: Anyone 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 let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None | Add a license: None | [?](#)

[Create repository](#)

Slides available at: <http://www.datalorax.com/vita/ds/ds1-slides/w4p2/>

42 / 43

# Connect it

Quick setup — if you've done this kind of thing before

Set up in Desktop or  HTTPS  SSH <https://github.com/datalorax/myrepo.git>

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# myrepo" >> README.md  
git init  
git add README.md  
git commit -m "first commit"  
git remote add origin https://github.com/datalorax/myrepo.git  
git push -u origin master
```

Which do you want?  
Copy/paste the code in your terminal

...or push an existing repository from the command line

```
git remote add origin https://github.com/datalorax/myrepo.git  
git push -u origin master
```