

Enhancing collaboration and reproducibility...

... using GitHub and distributed version control

Jay Brodeur | brodeujj@mcmaster.ca

Portage Webinar | 2020-10-06

McMaster University sits on the traditional Territories of the Mississauga and Haudenosaunee Nations, and within the lands protected by the “Dish With One Spoon” wampum agreement

(Indigenous Education Council, May 2016).

Learning objectives

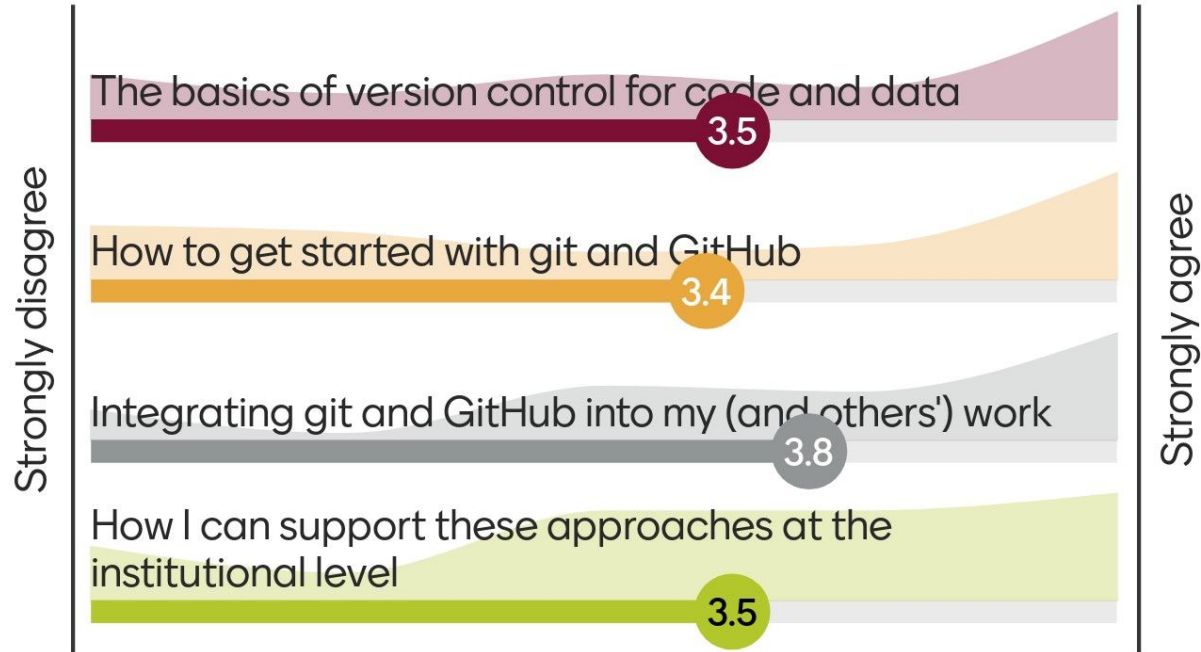
At the completion of this webinar, you should be able to:

- Explain the purpose and general function of version control systems
- Apply a variety of tools (git, GitHub, GitHub Desktop) to manage file versions within a ***repository***
- Apply best practices for efficiently managing and sharing repositories
- Describe how systems like GitHub can be used to support research collaboration and transparency
- Identify opportunities to implement these tools & practices to support research in your group or organization

But first ...

A few questions for you

Generally, I'm interested in learning about



My familiarity with version control with git & GitHub



This is all new to me!



I have heard of some of these



I know about them, but haven't used them before

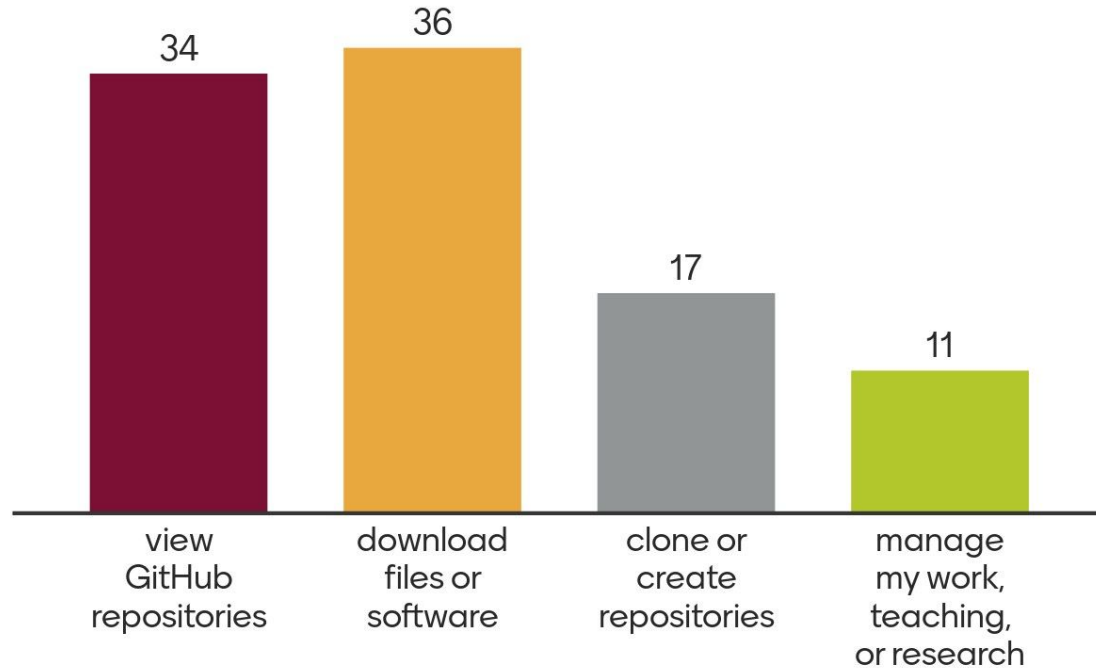


I'm familiar with them and I've used them

0

I should probably be giving this webinar!

In the past, I have you used GitHub to ...

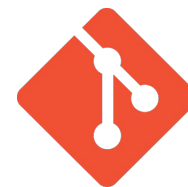




GitHub Pages



**Markup &
presentation**



**Local version
control**



GitHub

Repositories

**Administrative
tools**



Classroom



Education

Outline

Version control systems - types and value

Basic workflows in git and GitHub

Managing collaboration, access, & sharing

Sharing results: Markup and presentation

Administrative tools

Version control systems

(and why you might need them)

Local version control

my-research/

 readme.txt

 ↳ data/

 ↳ trial1results.csv

 ↳ scripts/

 ↳ t1analysis.py

Local version control (the hard way!)

my-research/

 readme.txt

 ↳ data/

 ↳ trial1results.csv

 ↳ scripts/

 ↳ t1analysis.py

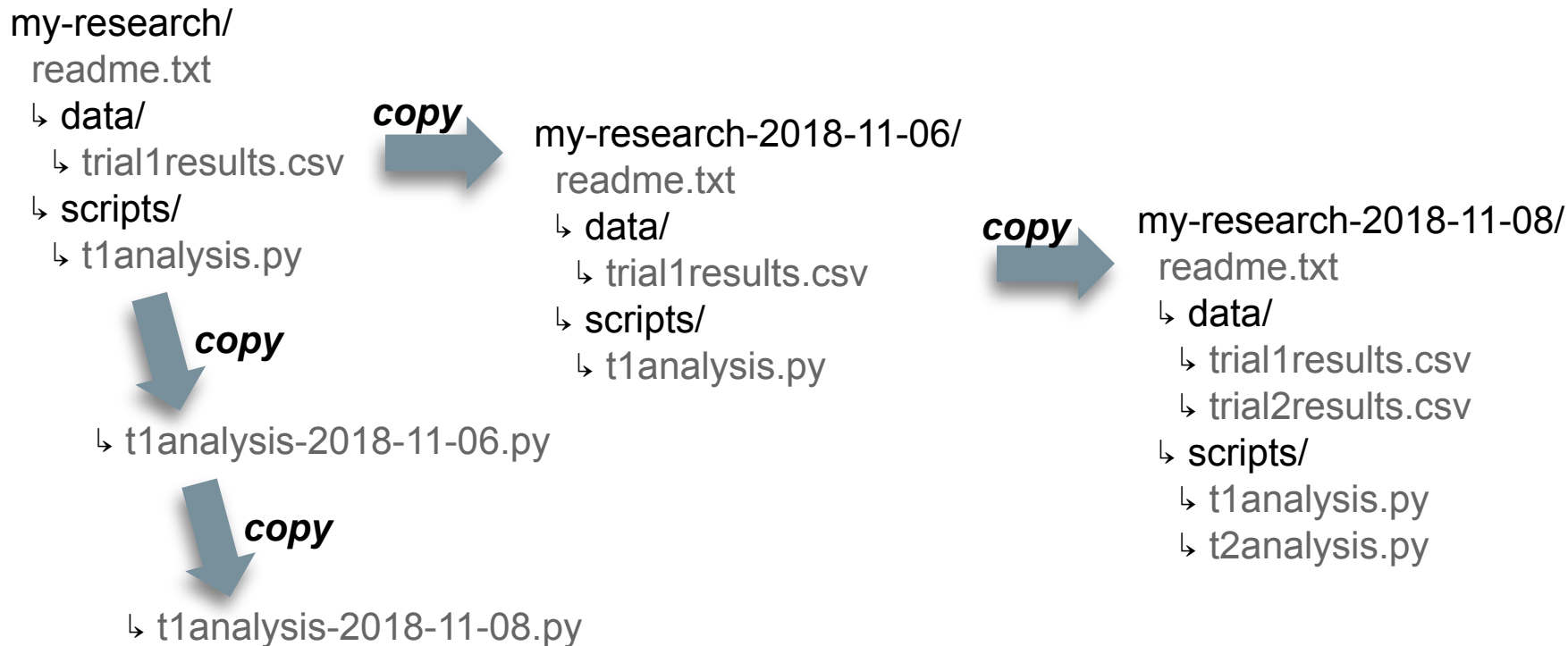


 ↳ t1analysis-2018-11-06.py



 ↳ t1analysis-2018-11-08.py

Local version control (the hard way!)

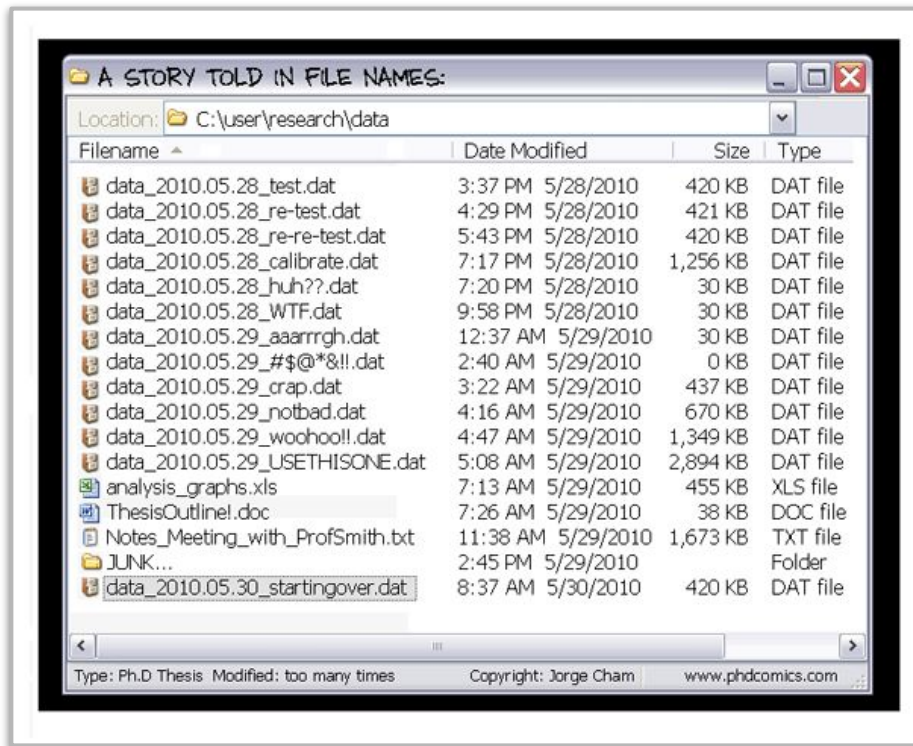


Local version control

You can track versions manually!

BUT, it's prone to errors:

- Writing to the wrong file/folder
- Overwritten files
- Misnamed (or poorly named) files
- “I just keep forgetting to do it”
- “Which old version is the correct one?”

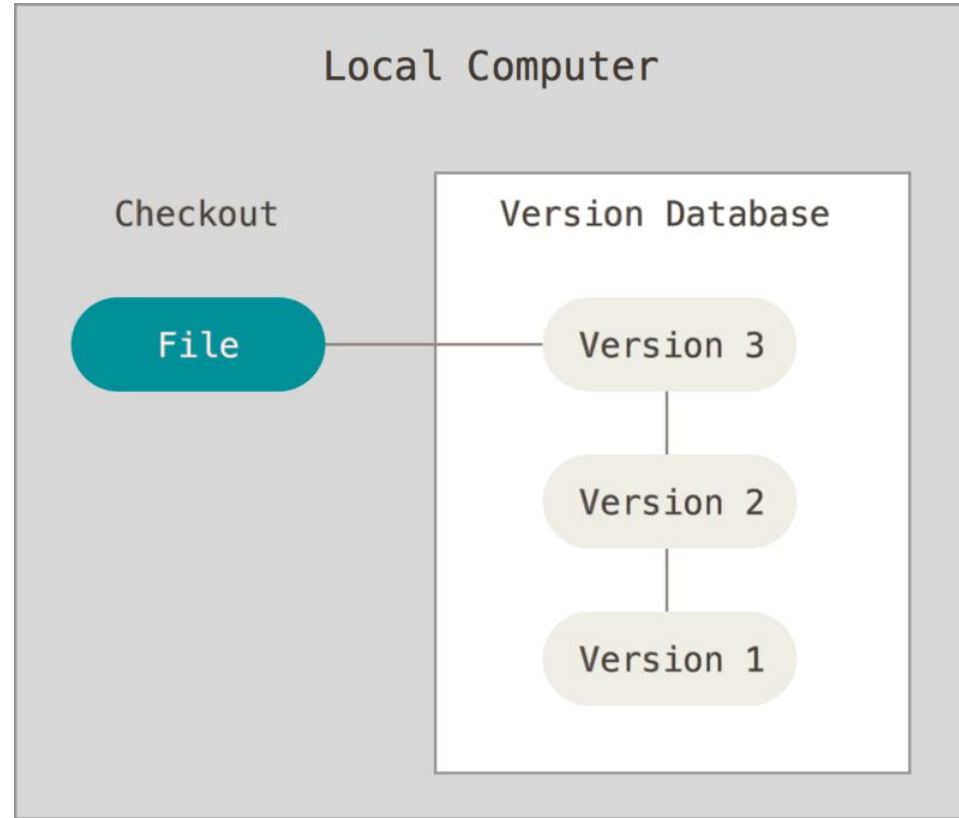


Local version control

Database system records changes to files and folders over time

Benefits: Can be mostly automated; consistent and dependable; traceability

Challenges: Not conducive to collaboration; local system failure could lead to data loss



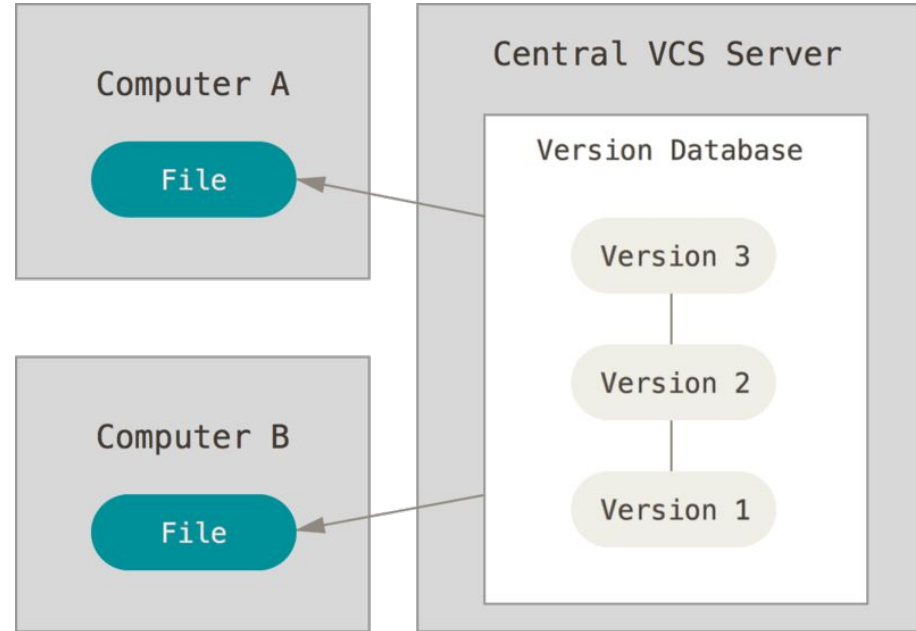
Centralized version control

Central (remote) database records changes from multiple local users

Users 'check out' a version they are working on.

Benefits: Allows for collaboration & granular permissions

Challenges: Can get 'locked out' or lose access altogether during outages



Distributed version control

Clients (users) **clone** the entire repository locally

Clients work locally; **push** changes to the server

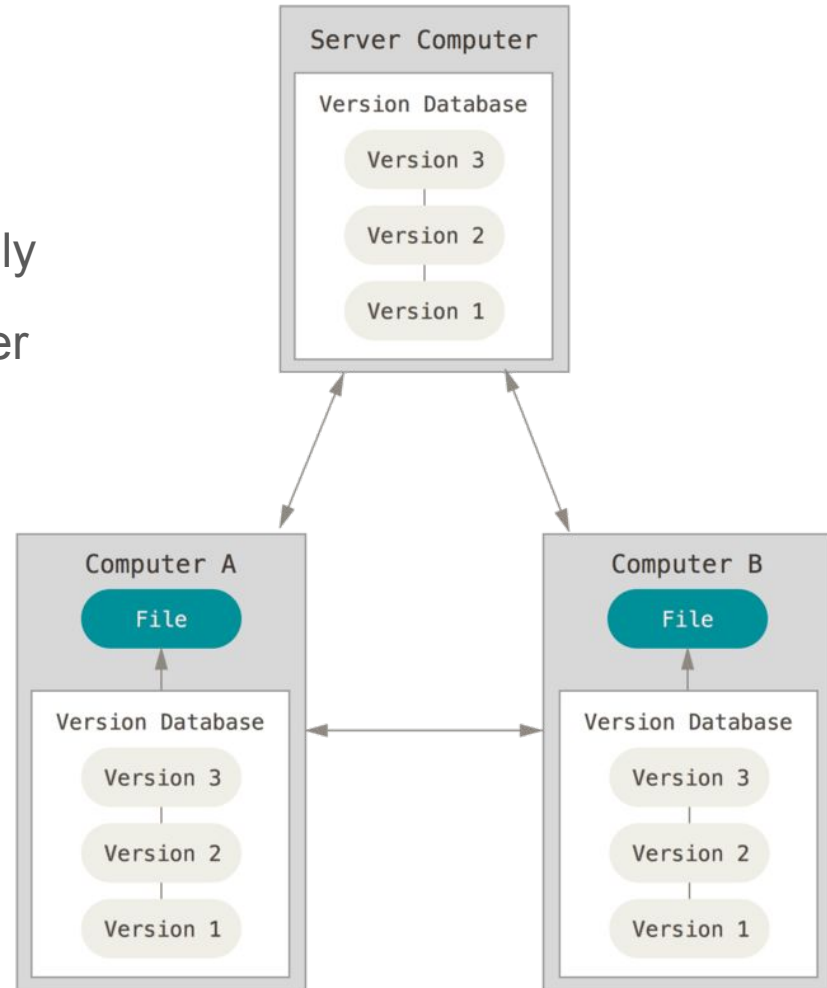
Changes managed and **merged** at the server

Clients **pull** new changes

Benefits:

- Collaboration & concurrent development
- Granular permission
- No single point of failure

Image credit: [Pro Git](#)



Why use distributed version control?

Distributed version control software allows you (and your collaborators) to:


- Track, compare, and revert changes (more quickly and granularly)
- Enable and manage collaborative development
- Deal with challenges of scale (# files, # changes, # collaborators)
- Share materials (openly or controlled); allow collaboration and reuse
- Backup your work to an external repository

Use it to manage:

- **Software / code** — Openrefine: github.com/OpenRefine/OpenRefine
- **Datasets** — OpenIndexMaps: github.com/OpenIndexMaps
- **Documentation** — DCN data curation primers: github.com/DataCurationNetwork/data-primers
- **Books** — Git from the Bottom Up: github.com/jwiegley/git-from-the-bottom-up
- **Websites** — UBC Research Commons' intro to git: ubc-library-rc.github.io/intro-git/



```
MINGW64 ~/Local/Digital-Archive-Tools
brodeurj@LT-10146-L MINGW64 /c/Local/Digital-Archive-Tools (master)
$ git pull origin master
From https://github.com/mac1ibGIS/Digital-Archive-Tools
 * branch                master      -> FETCH_HEAD
Updating cda4736..2d39cba
Fast-forward
 BulkTools/DA_bulk_downloader.asv      | 107 ++++++
 BulkTools/DA_bulk_downloader.m        | 31 +-----
 BulkTools/DA_dc_to_csv.m              | 140 ++++++
 BulkTools/DA_georef_prep.m            | 64 ++++++
 BulkTools/DA_list_map_collections.asv | 122 +-----
 BulkTools/old_DA_bulk_downloader.m    | 107 ++++++
 BulkTools/omeka-import-instructions.md | 8 ++
 BulkTools/run_DA_bulk_downloader.m    | 47 ++++++
 BulkTools/run_DA_dc_to_csv.m          | 2 +
 9 files changed, 471 insertions(+), 157 deletions(-)
 create mode 100644 BulkTools/DA_bulk_downloader.asv
 create mode 100644 BulkTools/DA_bulk_downloader.m
 create mode 100644 BulkTools/DA_georef_prep.m
 create mode 100644 BulkTools/DA_list_map_collections.asv
 delete mode 100644 BulkTools/old_DA_bulk_downloader.m
 create mode 100644 BulkTools/omeka-import-instructions.md
 create mode 100644 BulkTools/run_DA_dc_to_csv.m
brodeurj@LT-10146-L MINGW64 /c/Local/Digital-Archive-Tools (master)
$ |
```


[Search or jump to...](#)
[Pulls](#)
[Issues](#)
[Marketplace](#)
[Explore](#)


[macLibGis / Digital-Archive-Tools](#)

[Unwatch](#)
[3](#)
[Star](#)
[1](#)
[Fork](#)
[1](#)

[Code](#)
[Issues](#)
[Pull requests](#)
[Actions](#)
[Projects](#)
[Wiki](#)
[Security](#)
[Insights](#)

[master](#)

[Go to file](#)
[Add file](#)
[Code](#)


Jasonbrodeur updated scripts and documentation

on Jun 16 188

BulkTools

updated scripts and documentation

4 months ago

Google_Analytics_API...

Updating readme.md

2 years ago

MonitoringTools

updated URL for checking

2 years ago

OldMacrepo

Adding script to create top-level collection rep...

2 years ago

OldTools

Updating documents.

3 years ago

georeferencing-tools

Update readme.md

17 months ago

Empty.csv

Creating Scraper_V2.py

4 years ago

Guide.png

Uploading latest files and updated documentati...

3 years ago

Macrepo_Lookup.csv

Updating readme.md

2 years ago

Macrepo_Lookup2018...

Uploading latest files and updated documentati...

3 years ago

Macrepo_Lookup2018...

Uploading latest files and updated documentati...

3 years ago

Macrepo_Lookup2018...

Uploading latest files and updated documentati...

3 years ago

Macrepo_Lookup2018...

Uploading latest files and updated documentati...

3 years ago

Macrepo_Lookup2018...

Uploading latest analytics reports.

2 years ago

Macrepo_Lookup2018...

Uploading latest analytics reports.

2 years ago

Macrepo_Lookup2018...

Uploading latest analytics reports.

2 years ago

Macrepo_Lookup2018...

Uploading latest analytics reports.

2 years ago

Macrepo_Lookup2018...

Uploading latest analytics reports.

2 years ago

Macrepo_Lookup2018...

Uploading latest analytics reports.

2 years ago

Tools to use with the McMaster Digital Archive

[Readme](#)


Releases


No releases published
[Create a new release](#)


Packages

No packages published
[Publish your first package](#)

Contributors 3


 cochoznik Noel Kristen C...


 Jasonbrodeur Jason Brod...


 Jordanaharoni

Languages

HTML 97.7%

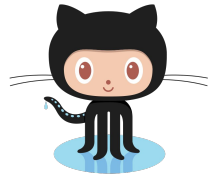
Python 1.8%

MATLAB 0.5%

CSS 0.0%

JavaScript 0.0%

Shell 0.0%



Basic workflows

in git and GitHub

1. *Initialize* or *clone* a *repository* (in git)

A **repository** (**repo**) is a set of files/directories managed with a VCS

Initialize git to create a new local repo in a selected directory
(with or without files)

```
$ cd C:/Local/my-repo
```

```
$ git init
```

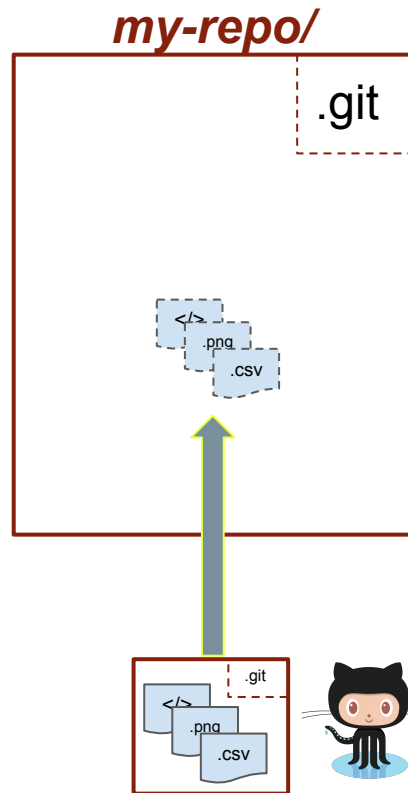
OR

Clone an existing repo (e.g from GitHub) to your local system

```
$ git clone https://github.com/username/my-repo.git
```

Tip: Create a readme.md and LICENSE file in the top directory

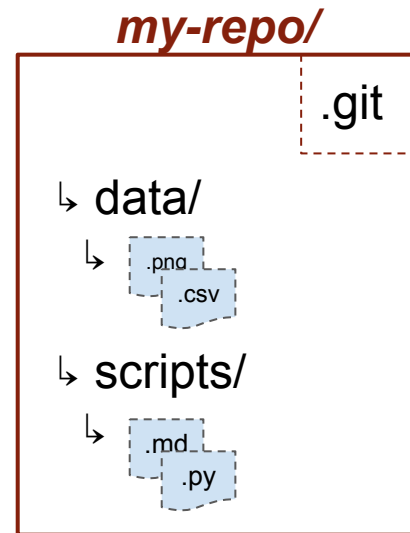
➤ These become your repo's readme file and license



2. Do your work

Create and edit files and folders

NOTE: Changes aren't tracked until you take a snapshot of them.



3. *Add* or *update* files (in git)

Add: tell git to begin keeping track of a file and its versions

```
$ git add README.md
```

```
$ git add --all
```

```
$ git add *.py
```

Update: tell git to take note of the changes that has been made to a file (staging)

```
$ git add -u
```

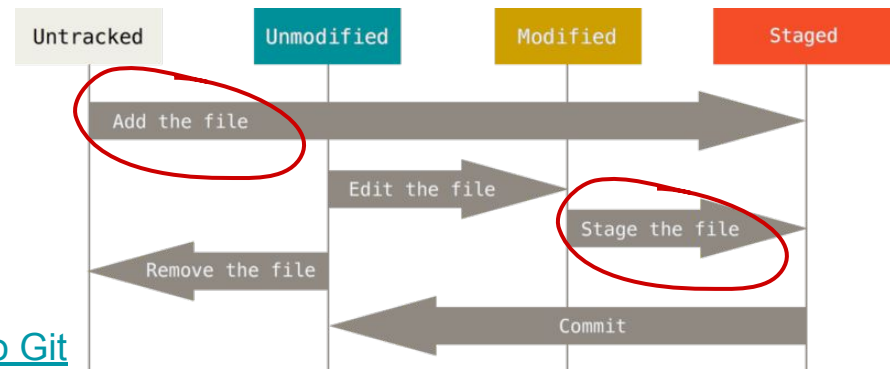
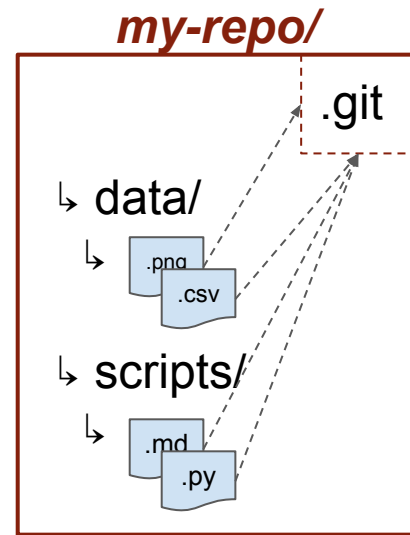


Image credit: [Pro Git](#)

4. **Commit** changes (in git)

Commit: tell git to take a snapshot of all staged (changed) files (while keeping old snapshots):

```
$ git commit -m "Title" -m "Description....."
```

- Add a short comment and a longer description

Add and **commit** at once with:

```
$ git commit -a -m "Title"
```

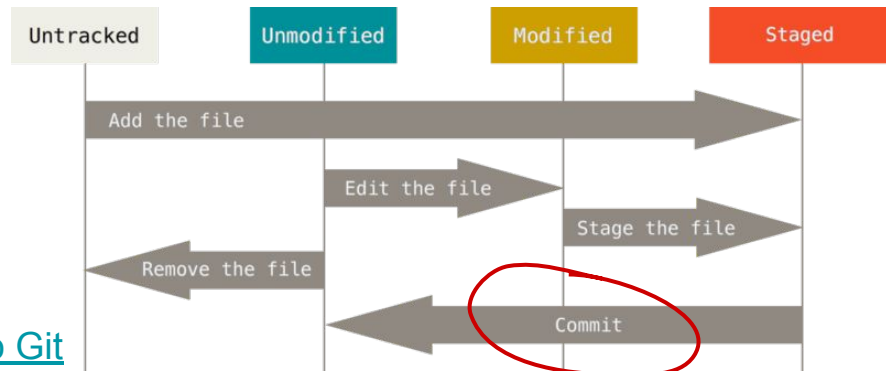
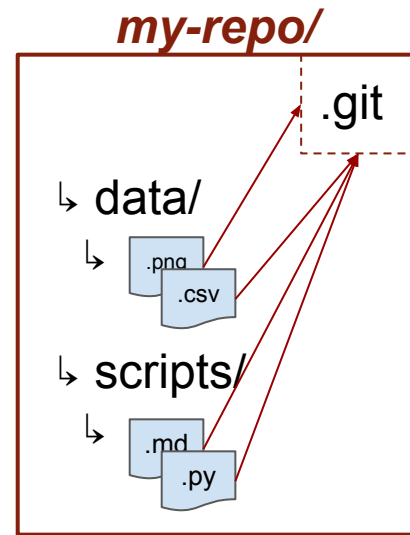


Image credit: [Pro Git](#)

OK

What Now?

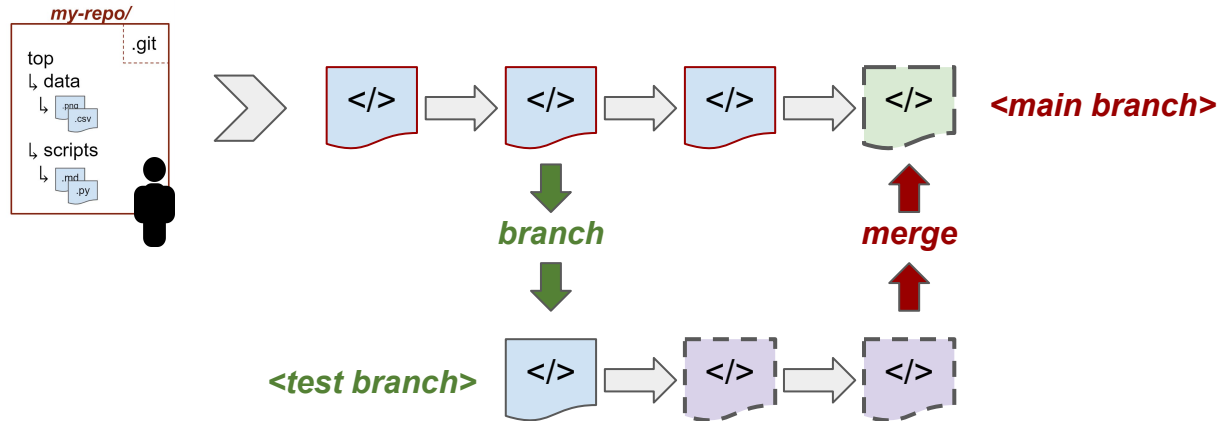
5a. Continue to work, add + commit

5b. Make a new **branch** to allow separate development

Clones the **main/master branch** (but tracked in the same git)

Allows separate development without breaking what's in place

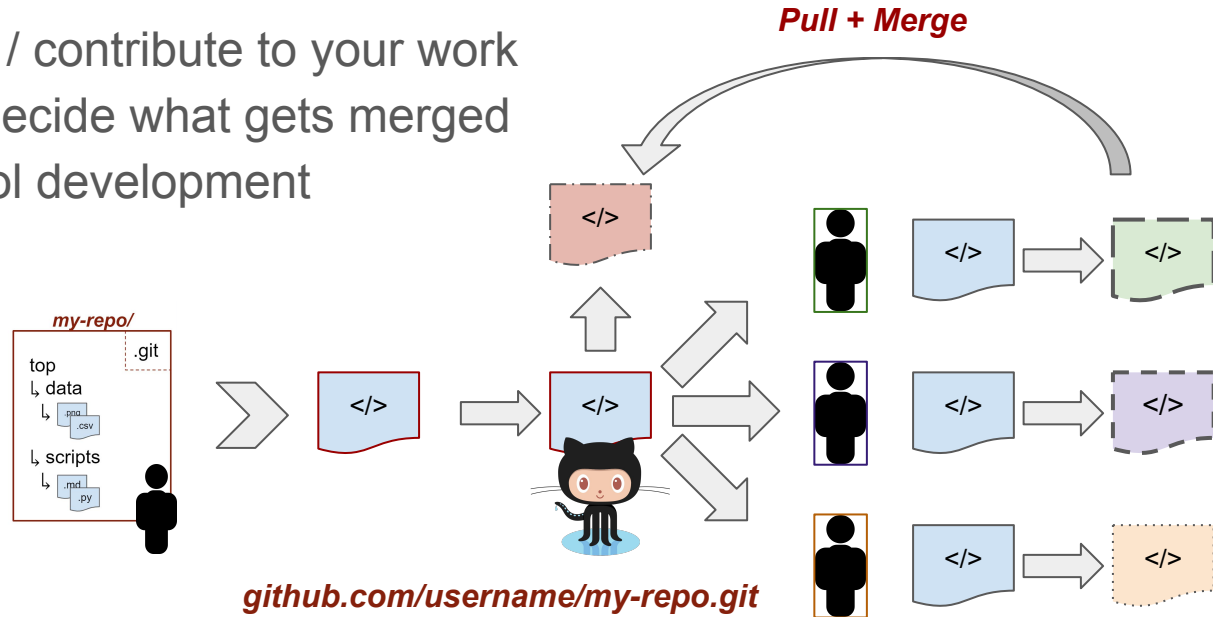
New branch can be later **merged** into the **main/master** one



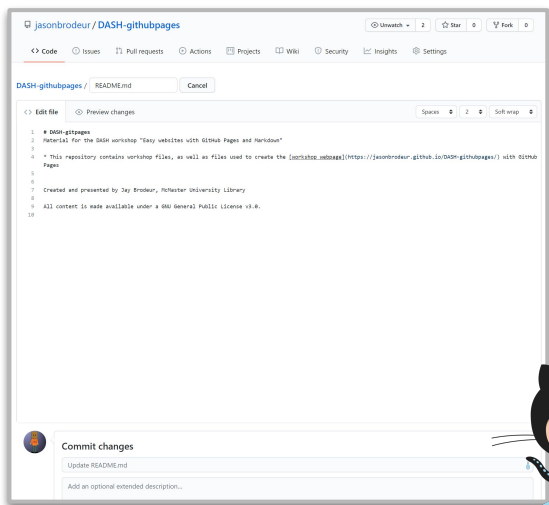
5c. **Push** changes to a remote repository (e.g. GitHub)

Push: send modified files (and git database and associated metadata) to a remote repository (e.g. GitHub, or another hosted repository)

- Disseminate & share
- Enable others to modify / contribute to your work
- Manage contributions; decide what gets merged
- Merge changes & control development



In GitHub



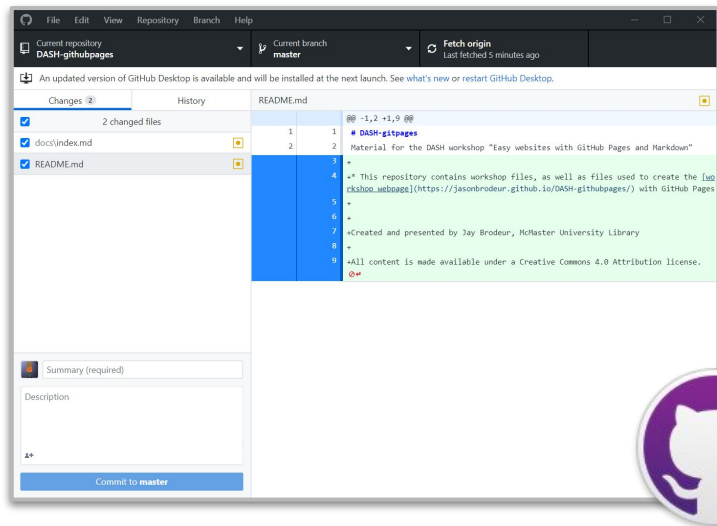
Use the web interface to:
create / clone repository (with readme)

↩ upload, edit etc.

↩ add + commit

↩ branch + merge

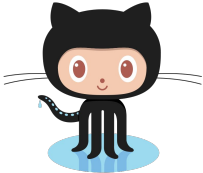
In GitHub Desktop



GitHub Desktop is a desktop application for local version control and interaction with GitHub using a GUI.

Managing collaboration, access, & sharing with GitHub

In GitHub



Use the web interface to:

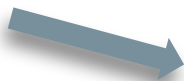
↳ create / clone a repository

- Create name and description
- Set visibility
- Add README, .gitignore, license

↩ upload, edit etc.

↩ add + commit

↩ branch + merge



Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Repository template

Start your repository with a template repository's contents.

No template ▾

Owner *

 jasonbrodeur ▾

Repository name *

research-repo ✓

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

Description (optional)

Project repository for our research work

☒  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

☒ **Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▾

☒ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

License: GNU General Public ... ▾

This License

Filter licenses...

None

Apache License 2.0

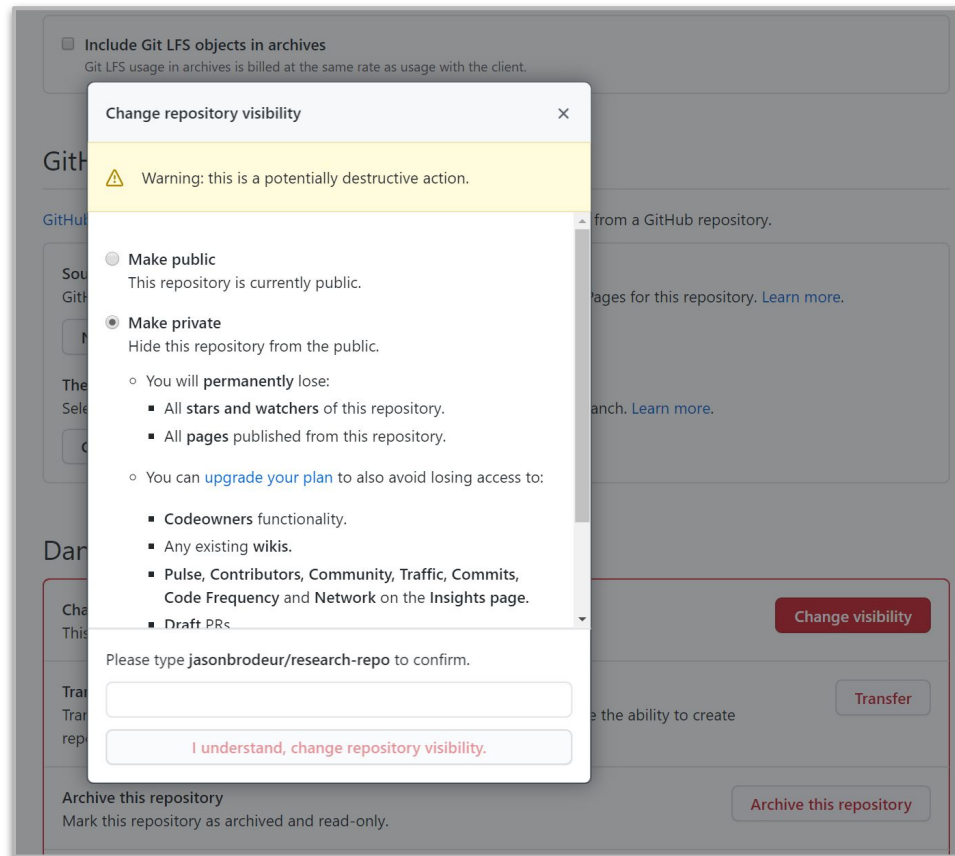
✓ GNU General Public License v3.0

Managing repository visibility

Set at creation or anytime in >Settings

Options:

- Public to everyone
- Private to collaborators / teams
 - Added private features with upgrade to paid account / organization



Adding a license

Created in the LICENSE file in the top-level of the repository

Built-in license selector (or add your own)

The screenshot shows the GitHub interface for adding a license to a project. The repository is 'jasonbrodeur / research-repo'. The navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The main heading is 'Add a license to your project'.

On the left, there is a list of licenses to choose from:

- Apache License 2.0
- GNU General Public License v3.0** (selected)
- MIT License
- BSD 2-Clause "Simplified" License
- BSD 3-Clause "New" or "Revised" License
- Boost Software License 1.0
- Creative Commons Zero v1.0 Universal
- Eclipse Public License 2.0
- GNU Affero General Public License v3.0
- GNU General Public License v2.0
- GNU Lesser General Public License v2.1
- Mozilla Public License 2.0
- The Unlicense

The right side of the interface displays the details for the selected 'GNU General Public License v3.0':

Permissions of this strong copyleft license are conditioned on making available complete source code of licensed works and modifications, which include larger works using a licensed work, under the same license. Copyright and license notices must be preserved. Contributors provide an express grant of patent rights.

Permissions	Limitations	Conditions
<ul style="list-style-type: none">✓ Commercial use✓ Modification✓ Distribution✓ Patent use✓ Private use	<ul style="list-style-type: none">✗ Liability✗ Warranty	<ul style="list-style-type: none">① License and copyright notice① State changes① Disclose source① Same license

This is not legal advice. [Learn more about repository licenses.](#)

The license text is displayed in a light blue box:

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <https://fsf.org/> Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

The 'Preamble' section is highlighted in a light blue box:

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

On the far right, there is a green button labeled 'Review and submit' and a note: 'You'll have a chance to review before committing a LICENSE file to a new branch or the root of your project.'

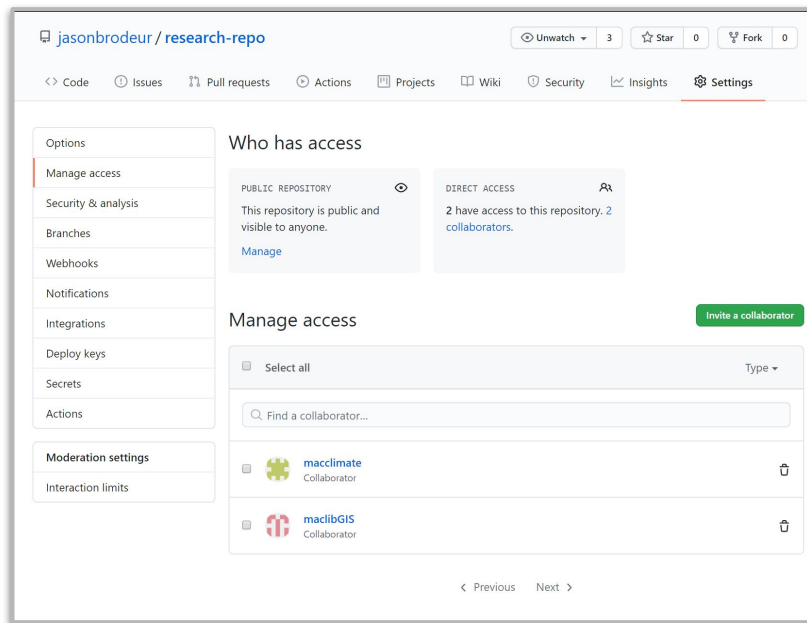
Managing collaborators on a personal repository

Collaborators can be added to both public and private repositories

For ***personal repositories*** (owned by a user), collaborators have only one set of privileges

- Can push, pull (read), and fork
- Manage pull requests, wikis, releases, etc.

More granular permissions are available for repositories owned by ***organizations***

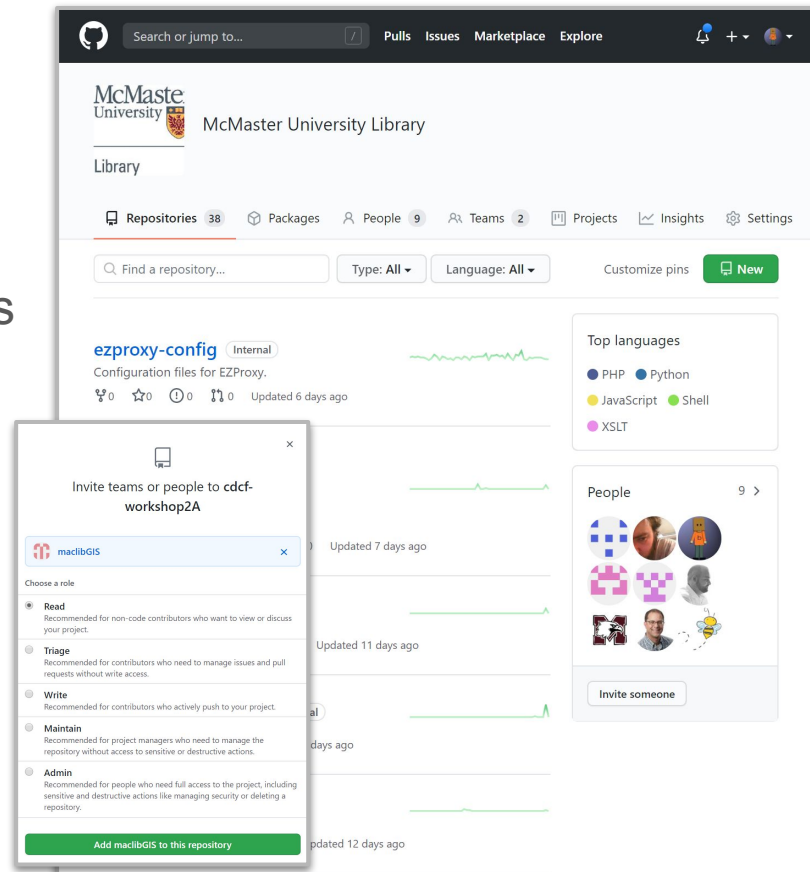


Managing access with *organizations* and *teams*

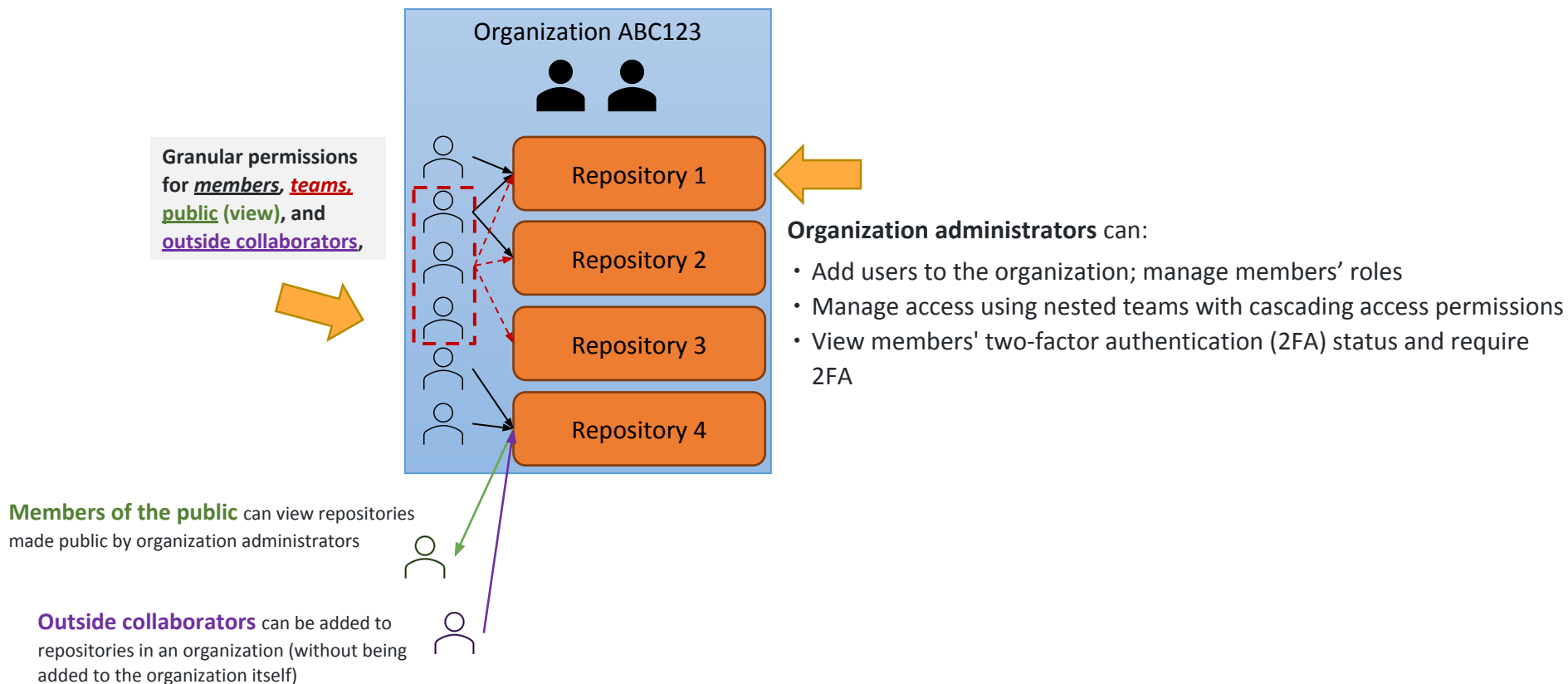
Organizations are shared accounts for projects

Benefits:

- Shared ownership of (unlimited) repositories
- Top-level management of repositories
- Unlimited membership
- Range of roles and permissions
- Nested *teams* with cascading access
- Two-factor authentication
- Are free to create



Managing access with *organizations* and *teams*

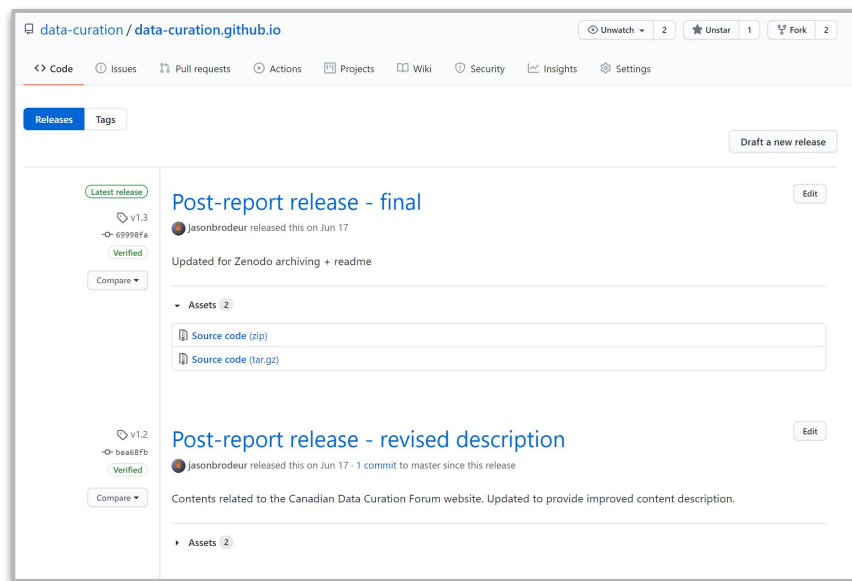


Packaging and ***releasing*** repositories

A release is a tagged snapshot of a repository for deploying a discrete version to broader audiences.

Releases are used when:

- Deploying software packages
- Packaging as supplemental materials (e.g. supporting a publication)
- Archiving in data / code repositories



An Example - Archiving in Zenodo

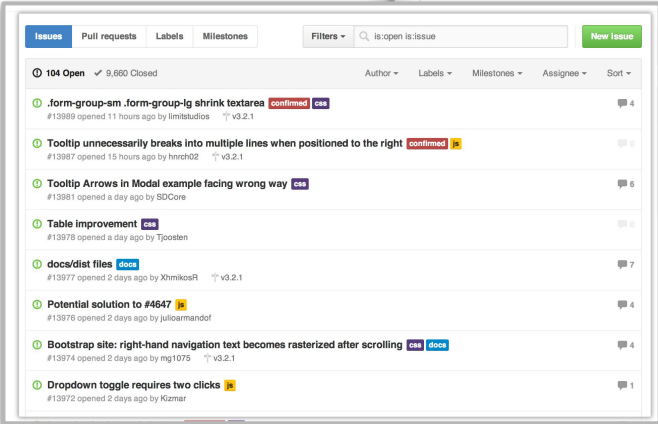
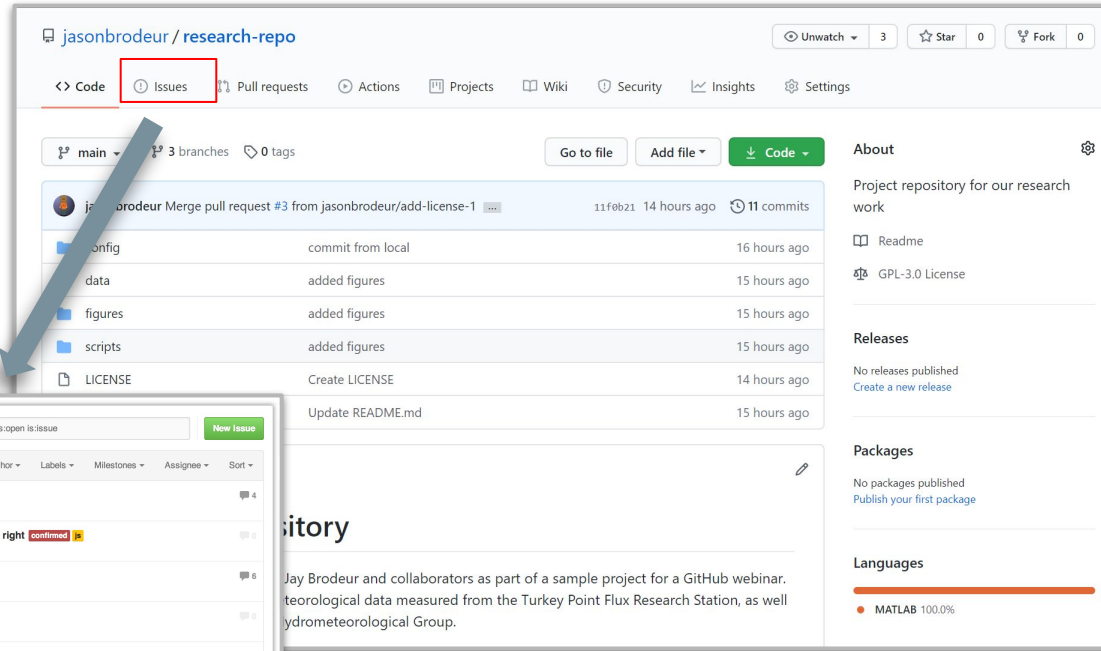
The screenshot shows the Zenodo interface for the repository 'Yahoo Knowledge Graph COVID-19 Datasets'. The page header includes the Zenodo logo, a search bar, and links for 'Upload' and 'Communities'. The repository title is 'Yahoo Knowledge Graph COVID-19 Datasets', created on August 12, 2020, by Ashley Wolf, Asaf Ary, and Hossein Firooz. It has 102 views and 5 downloads. The description states it is an archived fork of the Yahoo Knowledge Graph COVID-19 Datasets public repo, created August 8th, 2020, for archiving and citing purposes. The original repo URL is <https://github.com/yahoo/covid-19-data>. A link to the data release description is provided: <https://yahoodevelopers.tumblr.com/post/616566076523839488/yahoo-knowledge-graph-announces-covid-19-dataset>. The file list shows a folder 'covid-19-data-8-08-2020.zip' (42.6 MB) and a folder 'rexdouglass-covid-19-data-fb5f7eb' containing various files like CREDITS.md, LICENSE.txt, README.md, code-of-conduct.md, contributing.md, contributors.md, and a 'data' folder with regional data files. The DOI is [10.5281/zenodo.3981432](https://doi.org/10.5281/zenodo.3981432). The repository is indexed in GitHub and OpenAIRE.

Ashley Wolf, Asaf Ary, & Hossein Firooz. (2020, August 12).
Yahoo Knowledge Graph COVID-19 Datasets (Version
8-08-2020). Zenodo. <http://doi.org/10.5281/zenodo.3981432>

The screenshot shows the GitHub interface for the repository 'yahoo/covid-19-data'. The repository has 27 watchers, 74 stars, and 10 forks. It is on the 'master' branch with 1 pull request and 0 tags. The file list shows a folder 'data' (Update data, 10 minutes ago) and several files: CREDITS.md (start, 6 months ago), LICENSE.txt (start, 6 months ago), README.md (adding region codes for country, state and county, 3 months ago), code-of-conduct.md (start, 6 months ago), contributing.md (fix a small typo, 4 months ago), contributors.md (Update contributors.md, 5 months ago), and data-sources.md (start, 6 months ago). The README.md file is expanded, showing the title 'Yahoo Knowledge Graph COVID-19 Datasets' and links to the Zenodo repository and a chat. The repository is related to the 'Coronavirus Disease Research Community - COVID-19' and 'Zenodo' communities. The license is 'Other (Open)'. The repository is also linked to the 'sd-buildbot Update data' repository.

Main repo: <https://github.com/yahoo/covid-19-data>
Release (forked):
<https://github.com/rexdouglass/covid-19-data/tree/8-08-2020>

Communication & collaboration tools - Pull requests



Communication & collaboration tools - Pull requests

The image displays two screenshots of a GitHub repository interface. The top screenshot shows the repository page for 'jasonbrodeur / research-repo'. The 'Pull requests' tab is highlighted with a red box, and a large blue arrow points from it to a detailed view of a pull request in the bottom screenshot.

Repository Overview (Top Screenshot):

- Repository: jasonbrodeur / research-repo
- Unwatch: 3, Star: 0, Fork: 0
- Navigation: Code, Issues, Pull requests (highlighted), Actions, Projects, Wiki, Security, Insights, Settings
- Branches: main (3 branches), 0 tags
- Buttons: Go to file, Add file, Code
- About: Project repository for our research work, Readme, GPL-3.0 License, Releases, Packages, Languages

Pull Request Details (Bottom Screenshot):

Label issues and pull requests for new contributors

Now, GitHub will help potential first-time contributors discover issues labeled with **good first issue**

Filters: ispr is closed

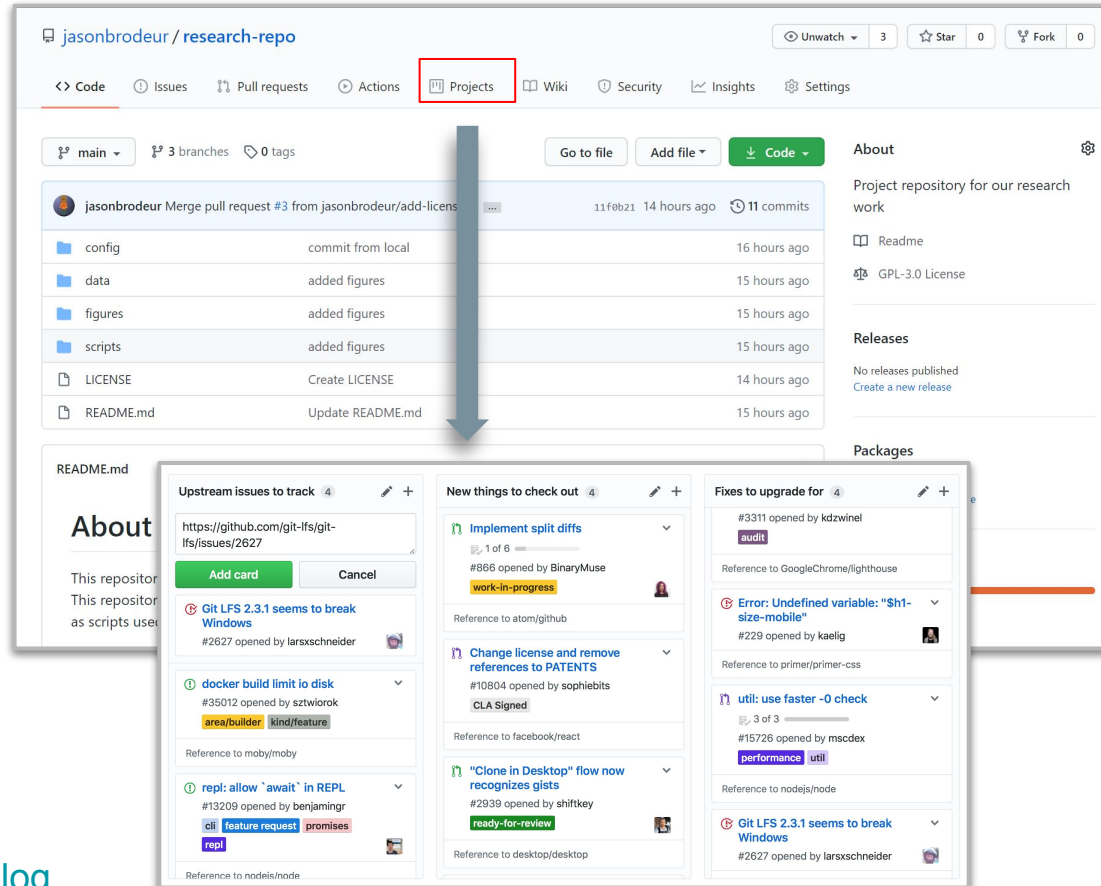
Clear current search query, filters, and sorts

0 Open, 3 Closed

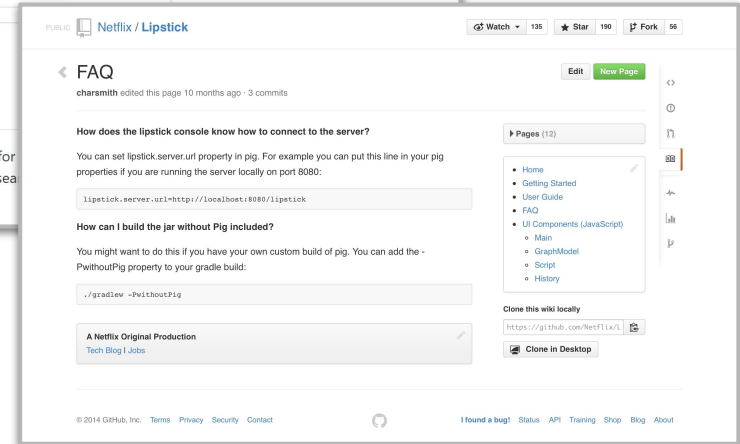
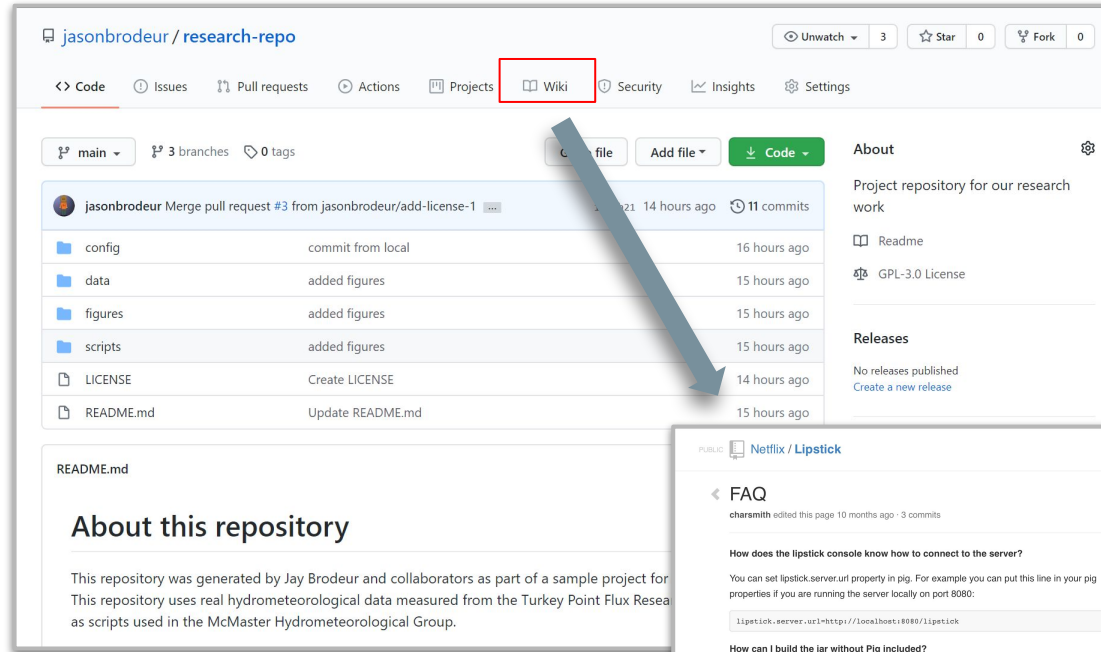
Issue	Author	Label	Projects	Milestones	Reviews	Assignee	Sort
Create LICENSE #3 by jasonbrodeur was merged 14 hours ago	jasonbrodeur	good first issue					
shortened data file #2 by jasonbrodeur was merged 15 hours ago	jasonbrodeur	good first issue					
commit from local #1 by jasonbrodeur was merged 16 hours ago	jasonbrodeur	good first issue					

1 comment

Communication & collaboration tools - Project Boards



Communication & collaboration tools - Wikis



Markup and presentation

GitHub as the medium

Example: GitHub repository as a preprint

The screenshot displays the GitHub interface for the repository 'paezha / covid19-environmental-correlates'. The repository has 1 watch, 4 stars, and 3 forks. The main content area shows a list of files and folders, including 'Environmental-Correlates-of-COVID-19...', 'README_cache/gfm', 'README_files/figure-gfm', 'Visualization-Contagion', 'covid19env', 'DS_Store', '.Rbuildignore', '.gitignore', 'COVID-19 Municipalities v0.Rmd', 'COVID-19-Municipalities-v0.html', 'README.Rmd', 'README.md', 'References.bib', 'covid19-environmental-correlates.Rpr...', 'covid19env_0.1.0.tar.gz', and 'spsur_1.0.1.3.tar.gz'. The README.md file is selected, showing the title 'A spatio-temporal analysis of the environmental correlates of COVID-19 incidence in Spain' and the author 'Antonio Paez (McMaster University)'. The right sidebar contains the 'About' section with the repository description, a DOI link 'doi.org/10.1111/gean.12241', a list of keywords, a 'Readme' section, 'Releases' (No releases published), 'Packages' (No packages published), 'Contributors' (2), and a 'Languages' section showing HTML (97.6%), TeX (2.1%), and R (0.3%).

Repository: paezha / covid19-environmental-correlates

Files and Folders:

- Environmental-Correlates-of-COVID-19... RELEASE. Updated preprint with doi information 4 months ago
- README_cache/gfm Updated README.Rmd after acceptance by Geographical Analysis 5 months ago
- README_files/figure-gfm Updated README.Rmd after acceptance by Geographical Analysis 5 months ago
- Visualization-Contagion Merge. 5 months ago
- covid19env Merge. 5 months ago
- DS_Store no changes 5 months ago
- .Rbuildignore Migrated files from Fernando's repo 6 months ago
- .gitignore Migrated files from Fernando's repo 6 months ago
- COVID-19 Municipalities v0.Rmd rmd municipalities 5 months ago
- COVID-19-Municipalities-v0.html rmd municipalities 5 months ago
- README.Rmd RELEASE. Updated preprint with names of authors and added doi inform... 4 months ago
- README.md RELEASE. Updated preprint with names of authors and added doi inform... 4 months ago
- References.bib Added acknowledgments to README.Rmd 4 months ago
- covid19-environmental-correlates.Rpr... Migrated files from Fernando's repo 6 months ago
- covid19env_0.1.0.tar.gz First round of revisions after invitation to revise and resubmit. 5 months ago
- spsur_1.0.1.3.tar.gz Migrated files from Fernando's repo 6 months ago

README.md

A spatio-temporal analysis of the environmental correlates of COVID-19 incidence in Spain

Antonio Paez (McMaster University)

About

Paez, A., López, F., Menezes, T., Cavalcanti, R., Pitta M.G.R., 2020. A Spatio-Temporal Analysis of the Environmental Correlates of COVID-19 Incidence in Spain, Geographical Analysis (Early View)

doi.org/10.1111/gean.12241

Keywords: covid-19, spatio-temporal-analysis, temperature, humidity, population-density, older-adults, incidence, mass-transit-systems, spain

Releases

No releases published

Packages

No packages published

Contributors 2

paezha Antonio Paez

tatianedemenezes

Languages

HTML 97.6% TeX 2.1% R 0.3%

Paez, A., López, F., Menezes, T., Cavalcanti, R., Pitta M.G.R., 2020. A Spatio- Temporal Analysis of the Environmental Correlates of COVID- 19 Incidence in Spain, Geographical Analysis

<https://github.com/paezha/covid19-environmental-correlates>

Markdown in GitHub



- A very lightweight markup language used by GitHub (and Reddit, and Trello)
- Improves formatting while leaving the plain document readable.
- Mostly just regular text with a few non-alphabetic characters thrown in

```
1  # This is a heading
2  I am making these words bold
3
4  Here is a list
5  * Of things
6  * And stuff
```

Markdown



This is a heading

I am making **these words** bold

Here is a list

- Of things
- And stuff

Rendered text

Learn more: <https://guides.github.com/features/mastering-markdown/>

GitHub Pages

“**GitHub Pages** is a static site hosting service that takes HTML, CSS, and JavaScript files straight from a repository on GitHub, optionally runs the files through a build process, and publishes a website.”

GitHub pages also allows you to create webpages from markdown files, using a built-in software called [Jekyll](#).



GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Source

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. [Learn more.](#)

🔗 Branch: main ▾

📁 / (root) ▾

Save

Theme Chooser

Select a theme to publish your site with a Jekyll theme using the gh-pages branch. [Learn more.](#)

Choose a theme

Administrative Tools

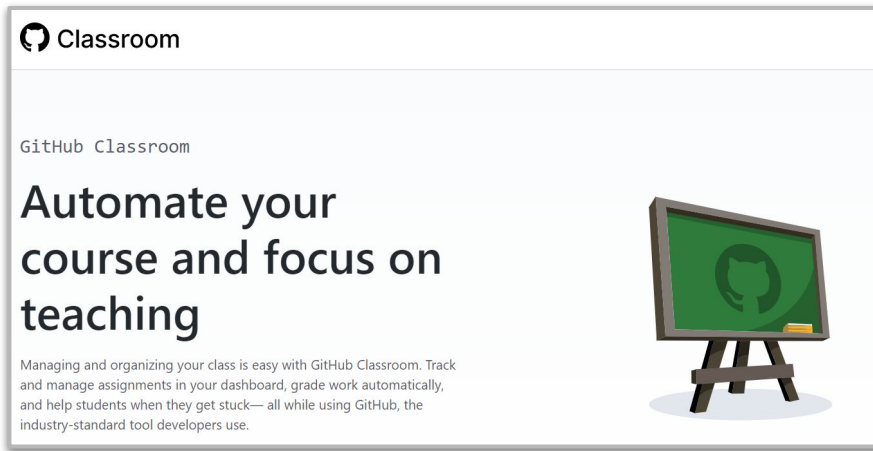
To support research & teaching

GitHub Classroom



Tools to use GitHub for course management

- Manage students in an organization
- Create assignment repositories from templates
- Granular access management of submitted materials
- Automated management & grading



GitHub Campus Program



Provides Institutional-level access to **GitHub Enterprise Cloud**, which:

- Helps institutions manage collaboration and access (including SAML single sign on and 2FA)
- Allows unlimited organizations
- Access to **GitHub Enterprise Support**
- Offers premium features (such as continuous integration)
- Provides administrators a single point of visibility and management.

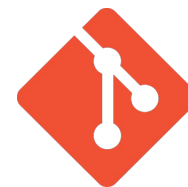
education.github.com/schools



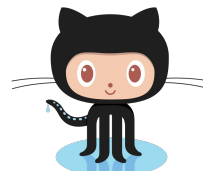
GitHub Pages



**Markup &
presentation**



**Local version
control**



GitHub

Repositories

**Administrative
tools**



Classroom



Education

And again ...
One more question

I would be interested in learning more about...

How to cherry-pick select elements only from pull requests.

GitHub classroom

Actual implementation and training across a real research group for collaboration.

git on its own

command line vs. desktop github

Needed to be hands on

version control with GitHub

how to start using it ? for someone working in the domain of info literacy how do I use it?

moving repositories to an organization

Best practices for sharing/pulling/forking/pushing/etc.

More examples of team collaboration and the logistics eg. merge requests, plus best practices?

maybe how linking content to dissertations thesis or master

more depth each area to support actual implementation to support my research work for HQP

version control and developing software

Sample practical examples of pull, merge, branch, etc.

Data-driven model

Hands on would be great!

github

I would be interested in learning more about...

A practical look at collaboration

Hands on would be good - specific topics - markdown, or gitpages etc.

Integration with Jira and pull requests

"Case studies" - How academic research groups are currently using GitHub, and in particular what did GitHub replace for them? e.g. "we were using dropbox and just hack and bashing", etc. Also how the wiki feature works!

Seconding more depth in each area. Mini-tutorials!

upvoting case studies suggestion!

Hand on using github

Learn more

The Git Pro book: <https://git-scm.com/book/en/v2>

Introduction to GitHub: <https://lab.github.com/githubtraining/introduction-to-github>

GitHub Guides: <https://guides.github.com/>

UBC Library Research Commons - Intro to git and GitHub: <https://ubc-library-rc.github.io/intro-git/>

Getting started with GitHub Pages: <https://guides.github.com/features/pages/>

GitHub Classroom: <https://classroom.github.com/>

GitHub Campus Program: <https://education.github.com/schools>