GitHub_Tutorial

Jonas Frankel-Bricker

10/30/20

Contents

Background

GitHub (github.com) is an online platform that provides hosting for software development and version control. Version control allows researchers and coders to work collaboratively and simultaneously when writing and revising code, promoting transparency and providing a forum for discourse and collaboration. The Quantitative Conservation Laboratory at Boise State University revises, stores, and disseminates code via this platform, and it is imperative that researchers affiliated with this lab become familiar and proficient with the methodology used to work in GitHub. This tutorial covers basic methods to conduct fundamental tasks in GitHub. For more information on the full utility of using this platform, please see other resources such as those available at https://guides.github.com/

Objectives

The following tutorial is presented as an introduction to using GitHub in combination with RStudio. It describes the basic protocols used to set up a GitHub account, create an online repository, connect an R project, push and pull information between R and GitHub, and manage and organize online repositories.

Outline

1. Set up a GitHub account
2. Create an online repository to store code and data
3. Construct an example R Project
4. Link the R Project with the online repository
5. Transfer information between platforms
6. Manage the online repository

Install "Git" on your computer

If you are working on a Mac, your computer likely already has the Git program installed in the Path "/usr/local/git" or "/usr/bin/git". However, Windows-based systems do not come with Git. To install Git follow the instructions for your operating system provided at the URL: https://git-scm.com/book/en/v2/Getting-Started-Installing-Git For Windows users, open the downloaded install file and accept all of the default options. **Keep track of the Path where git.exe will be installed. For most Windows users, this will be "C:/Program Files/Git/bin/git.exe".**

Set up a GitHub account

Prior to working through this tutorial, you'll need to sign up for a GitHub account.

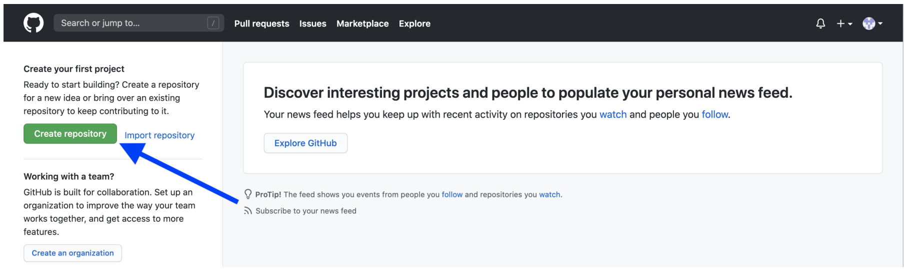
1. Navigate to GitHub's webpage (https://github.com/)
2. Click the green "Sign up for GitHub" button
3. Create a professional Username, password, and connect your BSU email with the account
4. Verify your account and click the blue "Create Account" button

Create a GitHub repository

Repositories are a core foundation of GitHub's infrastructure. They serve as online directories that contain information that can be shared between researchers affiliated with a project. All lab members will maintain their own private repositories to store functioning code. Repositories are often composed of multiple "branches", which can serve as different "versions" of the repository. The "main" branch represents the most up-to-date version of code. You will spend most of your time exchanging information between your computer and the "main" branch. As you write code on your local computer, it is imperative that you frequently save ("commit" and "push") the code to your online repository. **Do not store code on GitHub that is broken or not functioning as intended.** Saving working code in the online repository provides an invaluable safety net if your code gets broken on your local computer. If this happens, you can simply reload the previous version stored on GitHub. In this tutorial, you will be introduced to these concepts and components through the creation of a simple repository and transfer of information between this repository and an example R Project.

First, navigate to your GitHub's homepage and create an online repository following these steps:

1. Click the green "Create repository" button on the left of the screen if this is your first time creating a repository. If you already have access to other repositories, the green button will be labeled "New" instead.



Create your first project

Ready to start building? Create a repository for a new idea or bring over an existing repository to keep contributing to it.

Working with a team?

GitHub is built for collaboration. Set up an organization to improve the way your team works together, and get access to more features.

Discover interesting projects and people to populate your personal news feed.

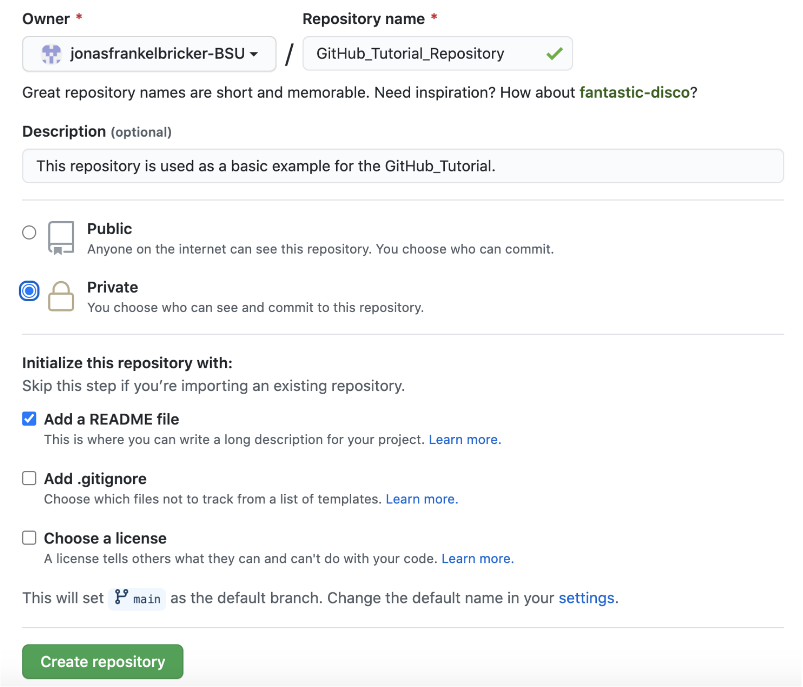
Your news feed helps you keep up with recent activity on repositories you watch and people you follow.

ProTip! The feed shows you events from people you follow and repositories you watch.

Subscribe to your news feed

2. In the next window, provide preliminary information for the repository.

- Name the repository "GitHub_Tutorial_Repository"
- Provide a brief description of the repository
- Set the repository to "Private"
- Click the box "Add a README file"
- Click the green "Create repository" button at the bottom of the screen



Owner * / Repository name *

jonasfrankelbricker-BSU ▾ / GitHub_Tutorial_Repository ✓

Great repository names are short and memorable. Need inspiration? How about fantastic-disco?

Description (optional)

This repository is used as a basic example for the GitHub_Tutorial.

○ 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.

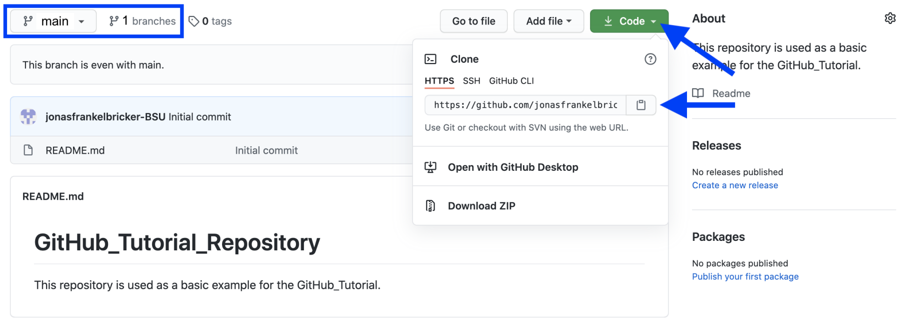
□ Add .gitignore
Choose which files not to track from a list of templates. Learn more.

□ Choose a license
A license tells others what they can and can't do with your code. Learn more.

This will set `main` as the default branch. Change the default name in your settings.

A new repository has now been created. Researchers with permission can now access the components of the repository. Notice how the dropdown menu in the top left of the page says "main". This indicates that you are viewing the "main" branch of the repository. Currently, this is the only branch associated with the repository, and "README.md" is the only file in the repository. For this example, "README.md" contains the the description of the repository you wrote when creating the repository.

4. Click the green dropdown menu "Code" and press the clipboard icon to copy the repository's URL (hereafter, "Repository_URL")



This branch is even with main.

jonasfrankelbricker-BSU Initial commit

README.md Initial commit

README.md

GitHub_Tutorial_Repository

This repository is used as a basic example for the GitHub_Tutorial.

Clone

HTTPS SSH GitHub CLI

https://github.com/jonasfrankelbric

Use Git or checkout with SVN using the web URL.

About

This repository is used as a basic example for the GitHub_Tutorial.

Readme

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

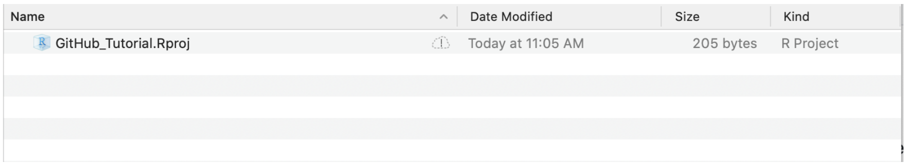
This is the unique URL associated with the repository. You will use this URL to connect your subsequent R Project to this online repository.

Connect GitHub to RStudio and link repository with R Project

This section focuses on the basic methods used to connect online repositories with R Projects. The following steps describe how to make a new project and connect it with the online repository you just created.

1. Create a new R Project

- Open RStudio
- Click the “File” tab at the top left of the screen
- Select “New Project” from the dropdown menu
- Click “New Directory”
- Click “New Project”
- Name the project “GitHub_Tutorial”
- Click the “Browse” button and select the appropriate PATH where the project will be located
- Click “Create Project”



Name	Date Modified	Size	Kind
GitHub_Tutorial.Rproj	Today at 11:05 AM	205 bytes	R Project

Your new project’s directory will be a folder located in the PATH you designated. Currently, this directory only contains a .Rproj file.

2. Set up version control with Git

- Click on the “Tools” tab at the top of screen
- Select “Global Options” from the dropdown menu
- Select the “Git/SVN” tab at the left side of the window and click the “Enable version control for RStudio projects” box. *You may need to provide the Path to the “Git executable”. If this is the case, click the “Browse” button and navigate to the Path where git.exe is located on your computer.*
- Click the “Apply” button (follow the prompt to restart the R session)

3. Link GitHub with the R Project

- Click on the “Tools” tab at the top of screen
- Select “Project Options” from the dropdown menu
- Select the “Git/SVN” tab at the left side of the window and click “Git” from the “Version control system:” dropdown menu
- Click apply at the bottom of the window (follow the prompts to restart the R session)

4. Link the repository’s URL with the R Project

- Click on the “Tools” tab at the top of screen
- Select “Shell”
  - This will open a separate shell terminal window
- Type the following code in the shell terminal window and press enter:

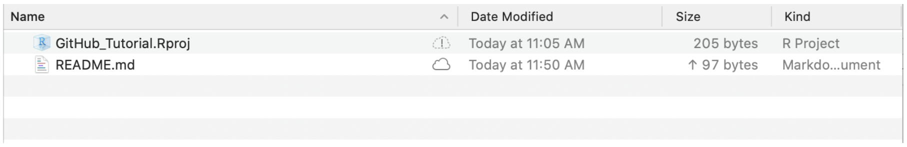
```
git remote add origin Repository_URL
```

For this example, “Repository_URL” is the unique URL associated with the online repository you copied earlier. You can paste this in the shell terminal window. “origin” now represents the online repository.

5. Pull the “main” branch from the repository

- Type the following code in the shell terminal window and press enter:

```
git pull origin main
```

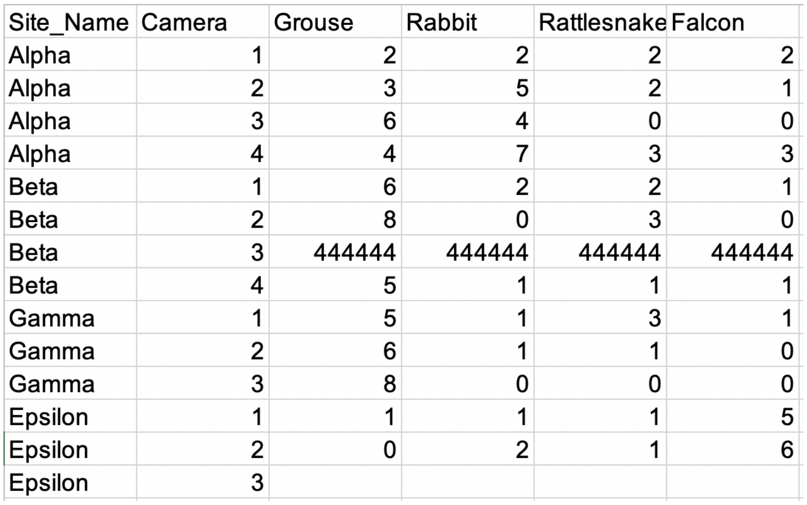


Name	Date Modified	Size	Kind
GitHub_Tutorial.Rproj	Today at 11:05 AM	205 bytes	R Project
README.md	Today at 11:50 AM	↑ 97 bytes	Markdo...ument

This step "pulls" the contents located in the "main" branch of your online repository and places the information in your project's directory. Note how the R Project's directory now contains the "README.md" file originally located in the online repository.

Working in RStudio

Now that the R Project and online repository are linked, changes to the R Project can be "pushed" back to the online repository. For this tutorial, you will process a very basic example data set, write several lines of code in an R script, "commit" these changes, and "push" the "commit" to a new branch of the online repository. The experimental data you will use is a comma-separated values file named "CountMatrix.csv".

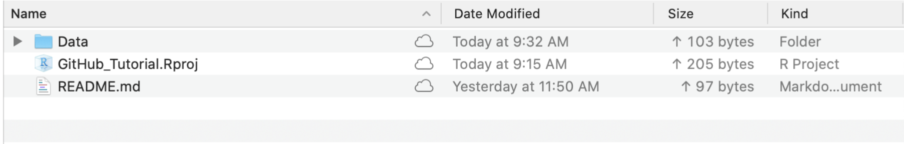


Site_Name	Camera	Grouse	Rabbit	Rattlesnake	Falcon
Alpha	1	2	2	2	2
Alpha	2	3	5	2	1
Alpha	3	6	4	0	0
Alpha	4	4	7	3	3
Beta	1	6	2	2	1
Beta	2	8	0	3	0
Beta	3	444444	444444	444444	444444
Beta	4	5	1	1	1
Gamma	1	5	1	3	1
Gamma	2	6	1	1	0
Gamma	3	8	0	0	0
Epsilon	1	1	1	1	5
Epsilon	2	0	2	1	6
Epsilon	3				

These data represent counts of different animal species (Columns) at cameras located at 4 different geographic sites (Rows). Each cell "value" represents the number of each species counted at each site.

1. Add the experimental data to the R project

- Right-click in the R Project directory and create a new folder named "Data"
- Click and drag the experimental data .csv file into the "Data" folder



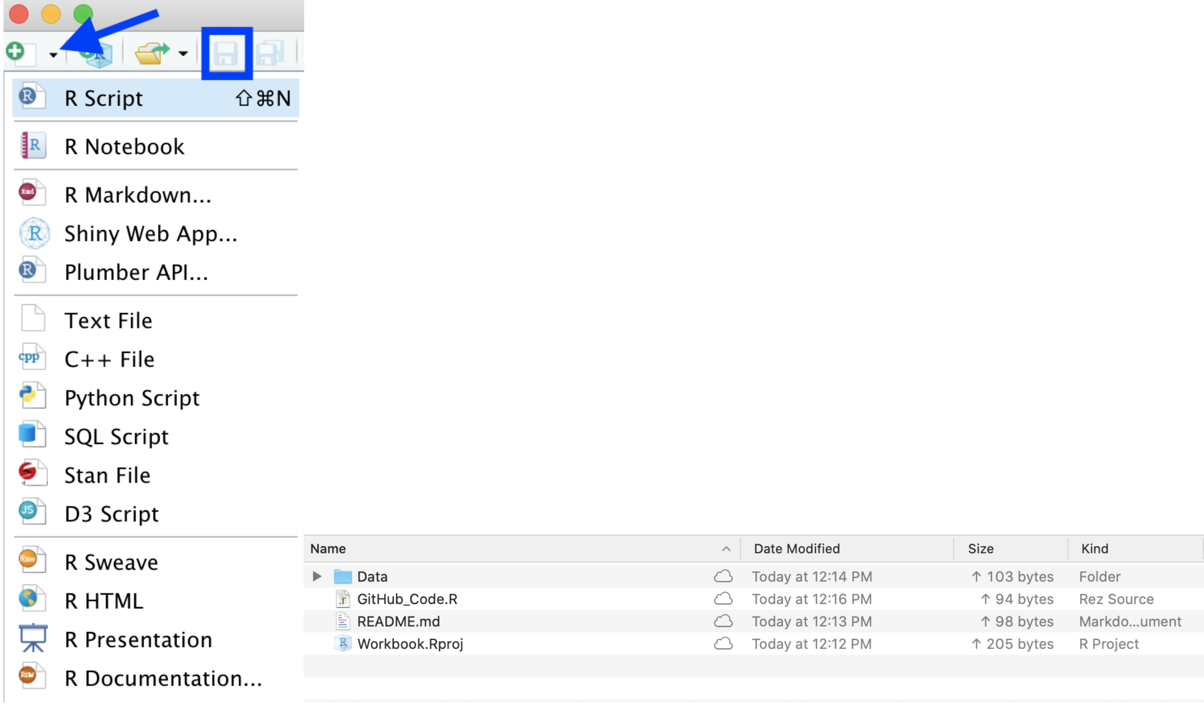
Name	Date Modified	Size	Kind
▶ Data	Today at 9:32 AM	↑ 103 bytes	Folder
GitHub_Tutorial.Rproj	Today at 9:15 AM	↑ 205 bytes	R Project
README.md	Yesterday at 11:50 AM	↑ 97 bytes	Markdo...ument

The project's directory now contains the subdirectory "Data" which contains the data file "CountMatrix.csv".

2. Create a new R Script

- Click on the green plus symbol in top left of RStudio window
- Select "R Script" from the dropdown menu

- Click the floppy disk icon and name the script "GitHub_Code"



Name	Date Modified	Size	Kind
▶ Data	Today at 12:14 PM	↑ 103 bytes	Folder
GitHub_Code.R	Today at 12:16 PM	↑ 94 bytes	Rez Source
README.md	Today at 12:13 PM	↑ 98 bytes	Markdo...ument
Workbook.Rproj	Today at 12:12 PM	↑ 205 bytes	R Project

The project's directory now contains the "GitHub_Code.R" script file.

3. Write the initial code in the R Script. For this example, the basic code to load two R Packages and import the data set is provided. Type this code in the "GitHub_Code.R" script file and click the floppy disk icon to save.

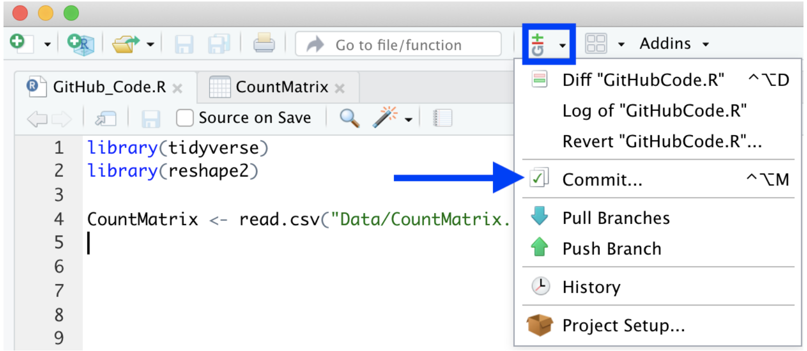
```r
# Load packages
library(tidyverse)
library(reshape2)

# Import data into R
CountMatrix <- read.csv("Data/CountMatrix.csv")
```

Initial GitHub commit and push to repository

You have now made several additions to your R Project. However, these changes have only been made on your local computer. Remember, the "main" branch of the GitHub repository solely contains the "README.md" file at this time. The following steps describe how to "commit" the changes you have made and "push" them to the online repository.

1. Click the "Git" dropdown menu at the top of the RStudio window

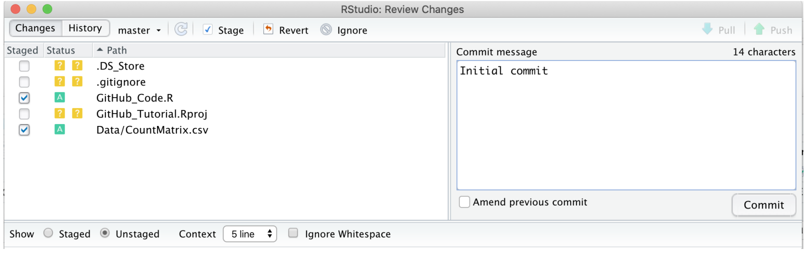
- Select "Commit"



```
library(tidyverse)
library(reshape2)

CountMatrix <- read.csv("Data/CountMatrix.
```

2. Check the boxes next to the files you want to transfer to the online repository. For this example, select the "GitHub_Code.R" and "Data/CountMatrix.csv" files.

- Boxes will turn green
- Add a short and descriptive commit message to serve as an identifier. For this example, type "Initial commit".
- Click the "Commit" button under the message section
- Close the subsequent windows



Staged	Status	Path
□	? ?	.DS_Store
□	? ?	.gitignore
☑	A	GitHub_Code.R
□	? ?	GitHub_Tutorial.Rproj
☑	A	Data/CountMatrix.csv

Commit message 14 characters

```
Initial commit
```

□ Amend previous commit

Show ○ Staged ● Unstaged Context 5 line □ Ignore Whitespace

You can make a "commit" at any time, however, only "commit" components that you are confident are correct and fully functional. These will not be stored in your online repository until you "push" the "commit" (which requires an internet connection). Always prioritize "committing" changes on your local computer first, especially when collecting data or writing code in the field.

3. Navigate to the shell terminal

- Use the previously opened terminal or navigate Tools -> Shell
- Type the following code and press enter:

  git push -u origin master

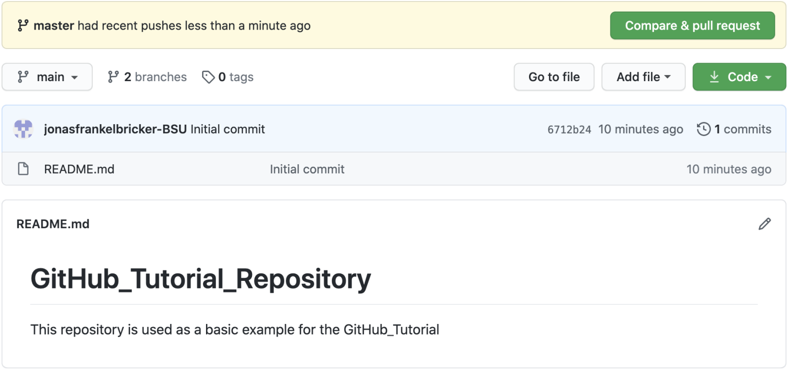
This will push "Initial commit" to the online repository in a new branch called "master".

Processing commit in GitHub

You have now pushed the changes you made in your R Project back to the online repository. The next steps involve processing these changes in GitHub.

1. Navigate back to the online GitHub repository

- A new message at the top of the window states “Master had recent pushes. . . ”

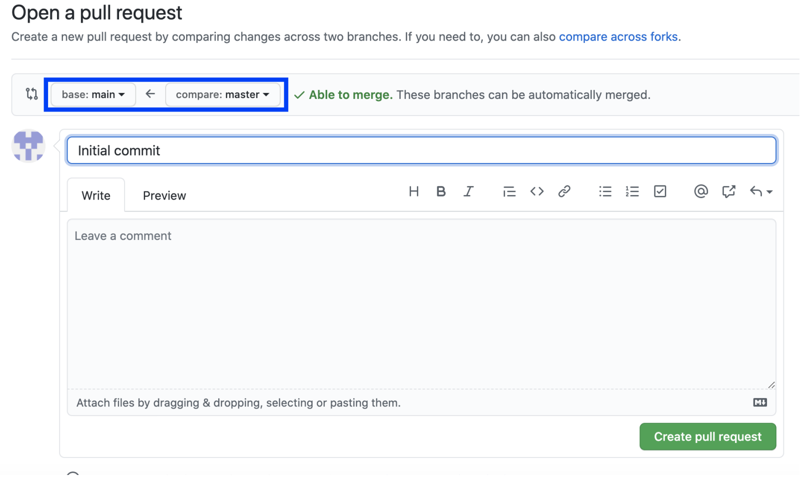

master had recent pushes less than a minute ago

GitHub_Tutorial_Repository

This repository is used as a basic example for the GitHub_Tutorial

2. Click the green “Compare & pull request” at the top of the window

- Set “base:” to “main” and “compare:” to “master”


Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base: main ← compare: master

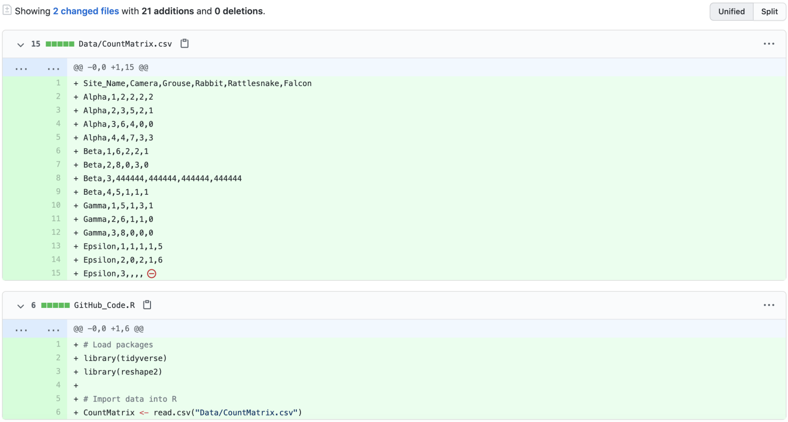
✓ Able to merge. These branches can be automatically merged.

Initial commit

Write
Preview
Leave a comment
Attach files by dragging & dropping, selecting or pasting them.
Create pull request

This designates that you want to compare master (which contains “Initial commit”) with “main” (your repository containing the most up-to-date code).

3. View the changes “Initial commit” will make to “main”
   - Scroll down to the bottom of the page to view all components added or revised in “Initial commit”



Showing 2 changed files with 21 additions and 0 deletions.

15 Data/CountMatrix.csv

```
@@ -0,0 +1,15 @@
+ Site_Name,Camera,Grouse,Rabbit,Rattlesnake,Falcon
+ Alpha,1,2,2,2,2
+ Alpha,2,3,5,2,1
+ Alpha,3,6,4,0,0
+ Alpha,4,4,7,3,3
+ Beta,1,6,2,2,1
+ Beta,2,8,0,3,0
+ Beta,3,444444,444444,444444,444444
+ Beta,4,5,1,1,1
+ Gamma,1,5,1,3,1
+ Gamma,2,6,1,1,0
+ Gamma,3,8,0,0,0
+ Epsilon,1,1,1,1,5
+ Epsilon,2,0,2,1,6
+ Epsilon,3,,,,
```

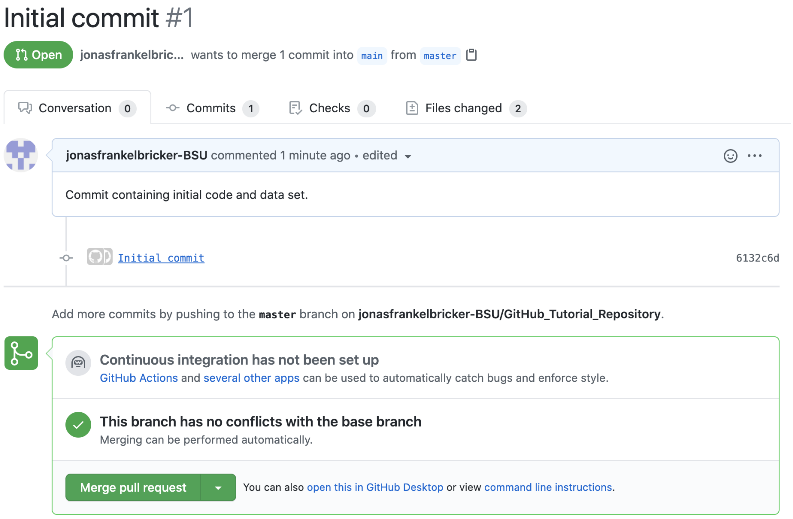
6 GitHub_Code.R

```
@@ -0,0 +1,6 @@
+ # Load packages
+ library(tidyverse)
+ library(reshape2)
+
+ # Import data into R
+ CountMatrix <- read.csv("Data/CountMatrix.csv")
```

Always double-check all parts of a new "commit" before proceeding to the next step.

4. Create a "pull request"
    - Write a descriptive comment for "Initial commit"
    - Click the green "Create pull request button"
5. Merge the "pull request"

- Click the green "Merge pull request" button
- Click the green "Confirm merge" button
- Click the "Delete branch" button next to the message "Pull request successfully merged and closed"



Initial commit #1

Open jonasfrankelbric... wants to merge 1 commit into main from master

jonasfrankelbricker-BSU commented 1 minute ago • edited

Commit containing initial code and data set.

Initial commit 6132c6d

Add more commits by pushing to the master branch on jonasfrankelbricker-BSU/GitHub_Tutorial_Repository.

Continuous integration has not been set up
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

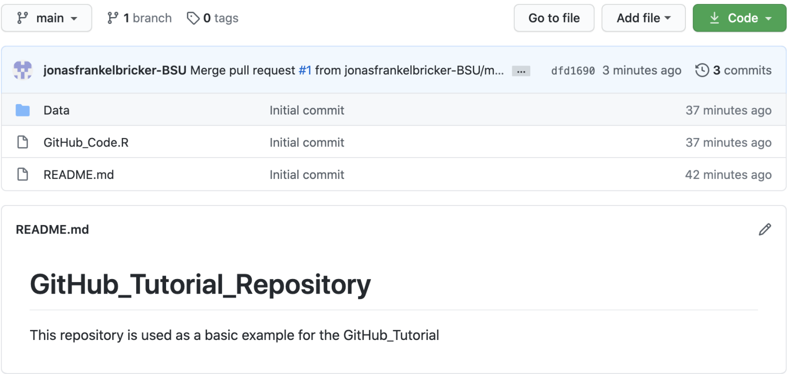
This branch has no conflicts with the base branch
Merging can be performed automatically.

You can also open this in GitHub Desktop or view command line instructions.

These steps incorporate "Initial commit" with the "main" branch and delete the temporary "master" branch.

6. Confirm that "Initial commit" was successfully processed

- Navigate to the repository's home page
- Confirm that all components were successfully transferred to the "main" branch



jonasfrankelbricker-BSU Merge pull request #1 from jonasfrankelbricker-BSU/m... … dfd1690 3 minutes ago 3 commits

Data	Initial commit	37 minutes ago
GitHub_Code.R	Initial commit	37 minutes ago
README.md	Initial commit	42 minutes ago

README.md

GitHub_Tutorial_Repository

This repository is used as a basic example for the GitHub_Tutorial

"GitHub_Code.R" and the "Data" directory containing "CountMatrix.csv" should now be located in the "main branch" along with the original "README.md" file.