

REPRODUCIBLE WORKFLOWS

Version Control and Computational Notebooks

John Little 

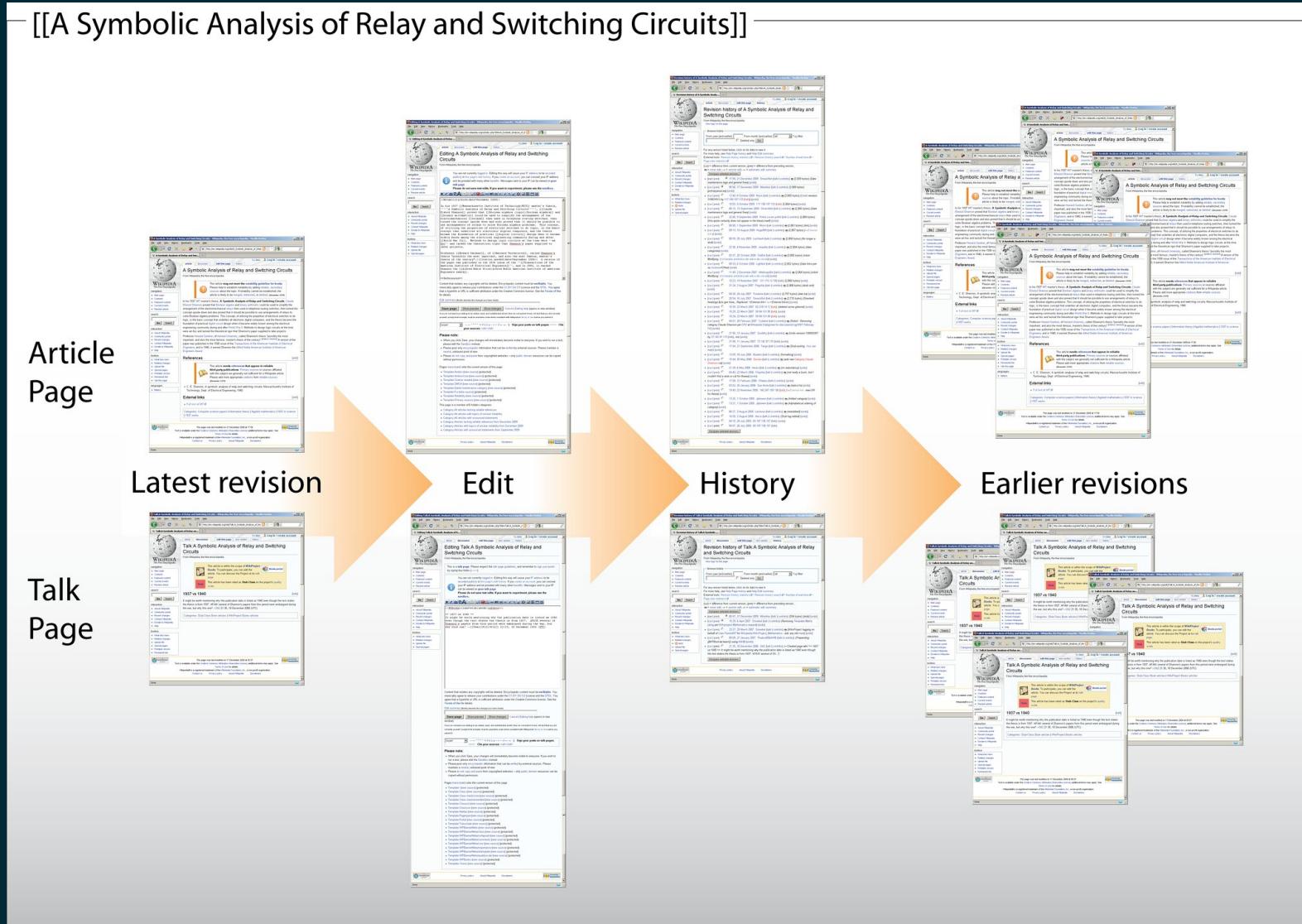
Duke University Libraries

Center for Data & Visualization Sciences

2024-02-06



ARTICLE PRODUCTION



REPRODUCTION

Authoring and computation environment should enable the articulation of scholarship within a reproducible context



FEATURES

- Support composable recombination
- Accommodate multimedia expression
- Provide rich reporting expressions
- Support economical portability and degrade gracefully
- Support extensibility
- Ensure transparency
- Support a documentary-style project history
- Accommodate change and collaboration
- Be citable

THREE POINTS

1. Notebooks (Literate Coding)
2. Version Control (Git & GitHub)
3. Sharing (Zenodo, Containers)

NOTEBOOKS



REPRODUCIBILITY

- Do everything with code!
 - Helps reduce repetition errors
 - Helps avoid copy/paste barriers
 - Orchestrate workflows

COMPUTATIONAL NOTEBOOKS

- Authoring environment
 - Code chunks interspersed with natural language
 - aka *Literate Coding*
- Easy to read and compose
- Graceful degradation

REPORTS AND EXPRESSIONS

Reports expressions are rendered at code execution



INTERACTIVITY AND WEB APPLICATIONS

- Shiny
- Flask
- WebR
- Plotly Dash
- ObservableJS

computation
rce Visu
title: "Quarto Computations"

his dataset contains a subset of the fuel economy data from the EPA.
pecifically, we use the `mpg` dataset from the `ggplot2` package.

```
[r]  
| label: load-packages  
  
library(ggplot2)
```

The visualization below shows a positive, strong, and linear relationship between the city and highway mileage of these cars. Additionally, mileage is higher for cars with fewer cylinders.

```
[r]  
| label: scatterplot  
  
ggplot(mpg, aes(x = hwy, y = cty, color = cyl)) +  
  geom_point(alpha = 0.5, size = 2) +  
  scale_color_viridis_c() +  
  theme_minimal()
```

QUARTO NOTEBOOK IN RSTUDIO

Computations

File Go to file/function Addins

Plots Packages Help Viewer Presentation Edit Publish

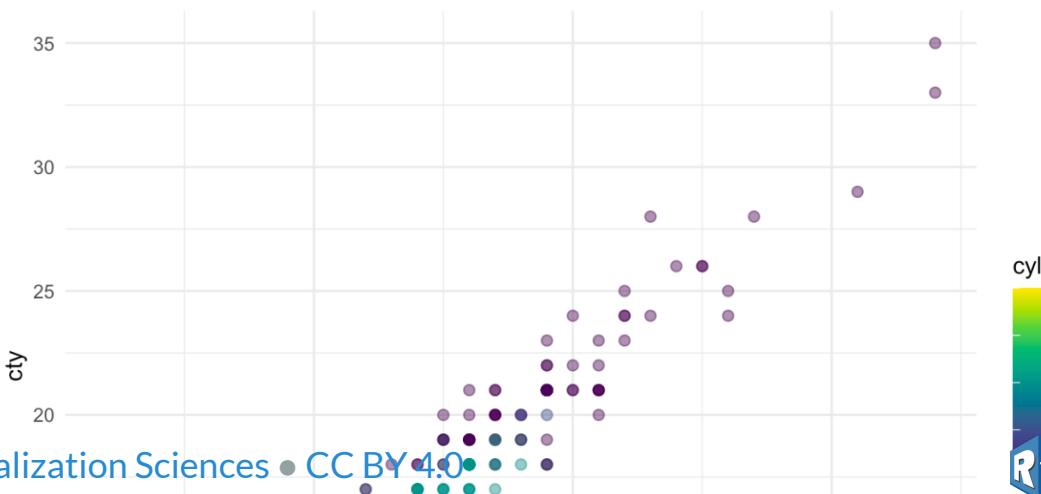
Quarto Computations

This dataset contains a subset of the fuel economy data from the EPA. Specifically, we use the `mpg` dataset from the `ggplot2` package.

```
library(ggplot2)
```

The visualization below shows a positive, strong, and linear relationship between the city and highway mileage of these cars. Additionally, mileage is higher for cars with fewer cylinders.

```
ggplot(mpg, aes(x = hwy, y = cty, color = cyl)) +  
  geom_point(alpha = 0.5, size = 2) +  
  scale_color_viridis_c() +  
  theme_minimal()
```



John R Little • Center for Data & Visualization Sciences • CC BY 4.0

R fun 4

JUPYTER NOTEBOOKS

(yes Classification) is a good starting point for classification tasks, linear regression models are a good starting point for can be fit very quickly, and are very interpretable. You are probably familiar with the simplest form of a linear regression model extended to model more complicated data behavior.

In this section we will start with a quick intuitive walk-through of the mathematics behind this well-known problem, before seeing how before moving on to see how linear models can be generalized to account for more complicated patterns in data.

We begin with:

```
[1]: %matplotlib
import matplotlib.pyplot as plt
notebook</h1>
```

Simple Linear Regression

We will start by fitting a simple linear regression model where a is the slope and b is the intercept.

Consider the following data:

```
[2]: rng = np.random.RandomState(42)
x = 10 * rng.rand(50) - 5
y = 2 * x + 1 + rng.randn(50)
```

Notebook Metadata

```
[3]: {"kernelspec": {"display_name": "Python 3", "language": "python", "name": "python3"}, "language_info": {"codemirror_mode": {"name": "ipython", "version": 3}, "file_extension": ".py", "mimetype": "text/x-python", "name": "python", "nbconvert_exporter": "python", "pygments_lexer": "ipython3", "version": "3.6.7"}, "toc-autonumbering": false, "toc-showcode": true, "toc-showmarkdowntxt": true}
```

We can use this to fit a simple linear regression model:

```
[3]: from sklearn import linear_model
```

Python 3 | Idle

Launcher

- Python 3
- C++11
- C++14
- C++17
- Julia 1.1.0
- phylogenetics (Python 3.7)
- R

Console

- Python 3
- C++11
- C++14
- C++17

Seattle Weather: 2012-2015

Julia.ipynb

```
[10]: using RDatasets, Gadfly
plot(dataset("datasets","iris"), x="Sepal.Length", y="Sepal.Width")
```

```
[10]: eigen(x)
```

```
[10]: Eigen{Complex{Float64}, Complex{Float64}, Array{Complex{Float64}, 2}, Array{Complex{Float64}, 1}}
eigenvalues:
10-element Array{Complex{Float64}, 1}:
 4.793881566545466 + 0.0im
 -0.9445989635995898 + 0.0im
```

Julia

Lorenz.ipynb

```
[1]: %matplotlib inline
from ipywidgets import interactive, fixed
```

python notebook

We explore the Lorenz system of differential equations:

$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$

Let's change (σ, β, ρ) with ipywidgets and examine the trajectories.

```
[2]: from lorenz import solve_lorenz
w = interactive(solve_lorenz, sigma=(0.0, 50.))
w
```

```
[1]: head(iris)
```

Sepal.Length	Sepal.Width	Petal.Length
5.1	3.5	1.4
4.9	3.0	1.4

R.ipynb

```
[3]: ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width))
```

R

QUARTO

- A scientific publishing system
- R, Python, ObservableJS
- Compose with standard text editors, or basic IDEs
 - IDEs: RStudio, Jupyter, VSCode

RENDERED OUTPUTS

- Artifacts that document a body of work
- Are reproducible and modifiable when data or techniques change
- Easy to update natural language explanations and re-render outputs
- Schedule emails based on report parameters

SUMMARY OF BENEFITS

- Using natural language clearly explain data, models, and workflows
- Reduce dependencies on outside and undocumented steps
- Ability to expose technical code chunks depending on audience focus
- State of the art reproducibility
 - 21st century **container** for evidence-based, computationally-processed research

VERSION CONTROL



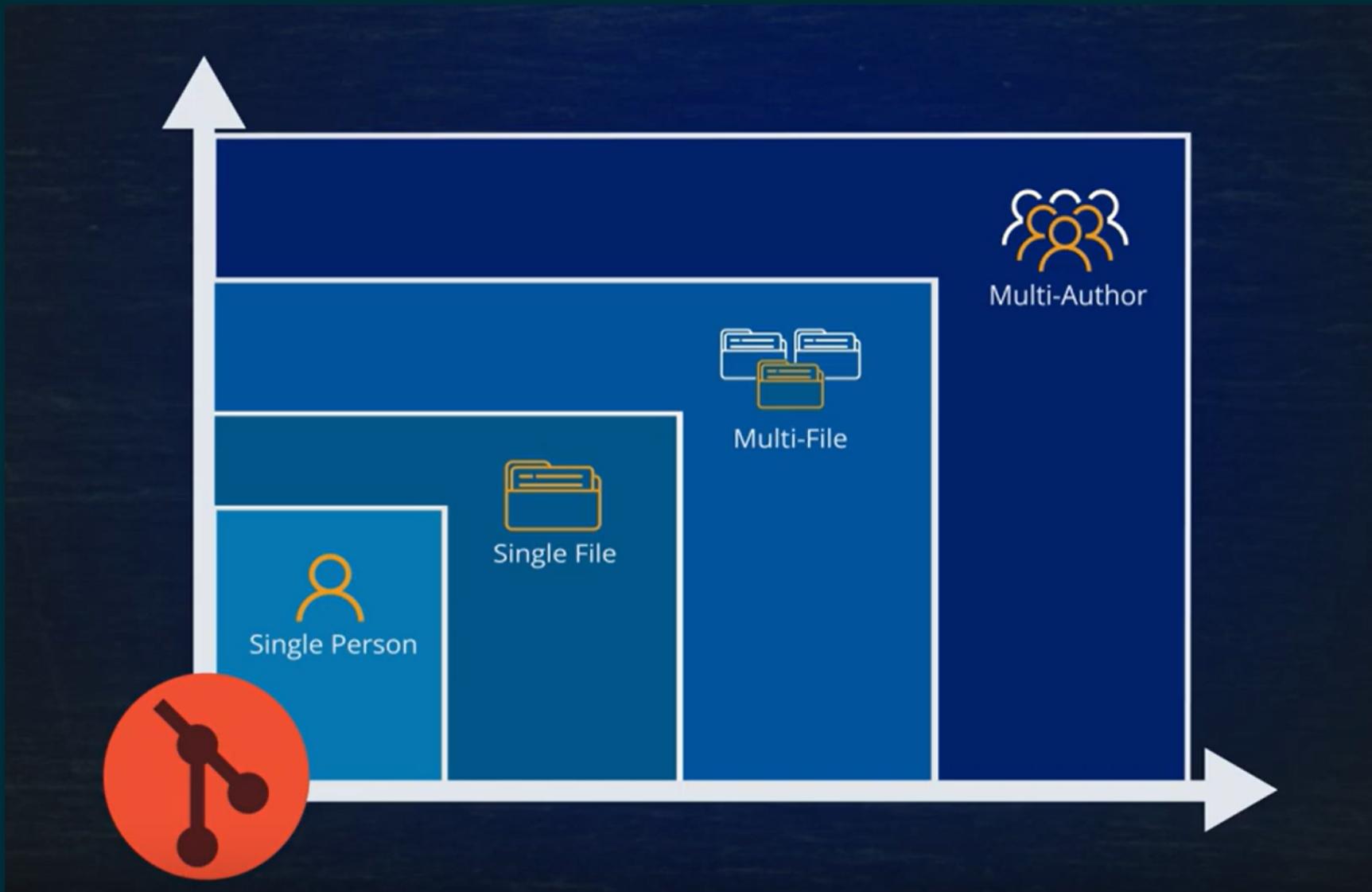
DEFINITION

- A system to manage projects (repo)
- A system to track how computer files change over time
- A system that supports collaborative revision
- More than file synchronization
- Assists in project back-ups

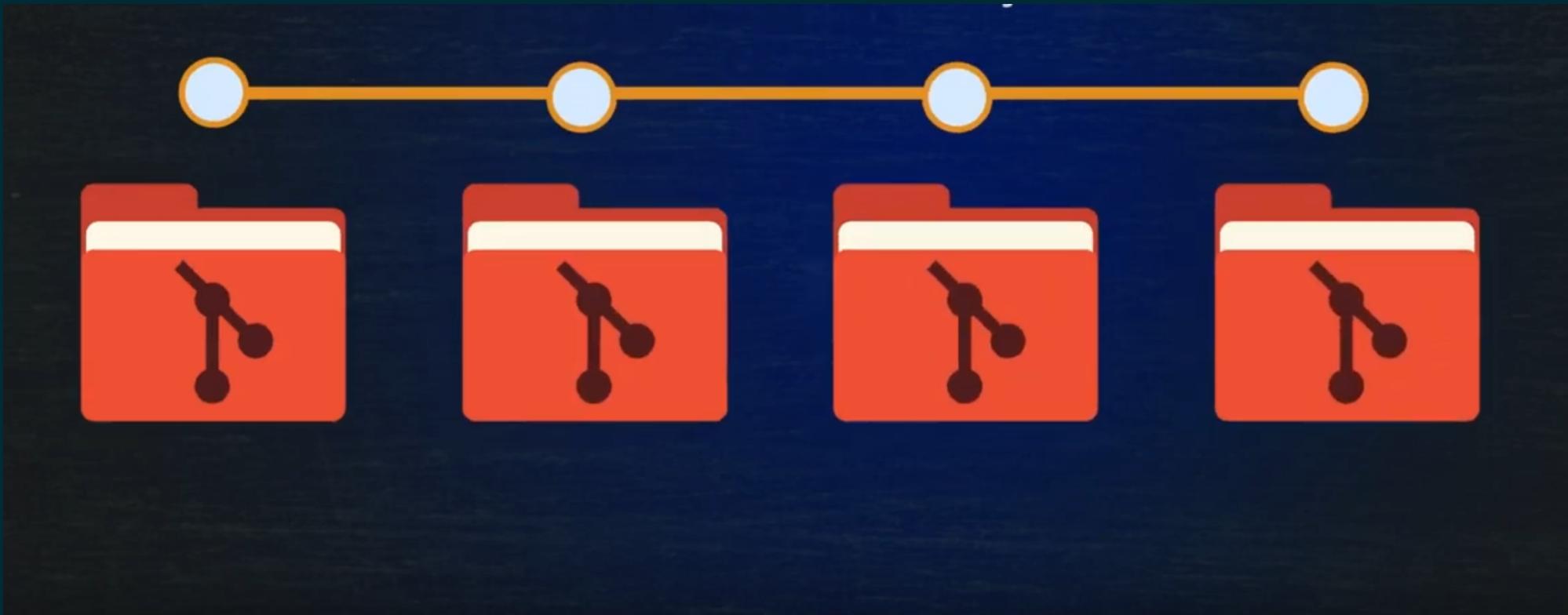
GIT

- Free open source
- Wildly successful; most broadly implemented
- In use across the globe
- Use it on any file system
- Track any file
- Use it in any environment

SCALABLE TO PROJECT SIZE



PROJECT REPOSITORIES



- Work on any file system
- Operates on at the folder level

ARCHIVAL VS VERSION-CONTROL



Zenodo logo - Postery of milestones



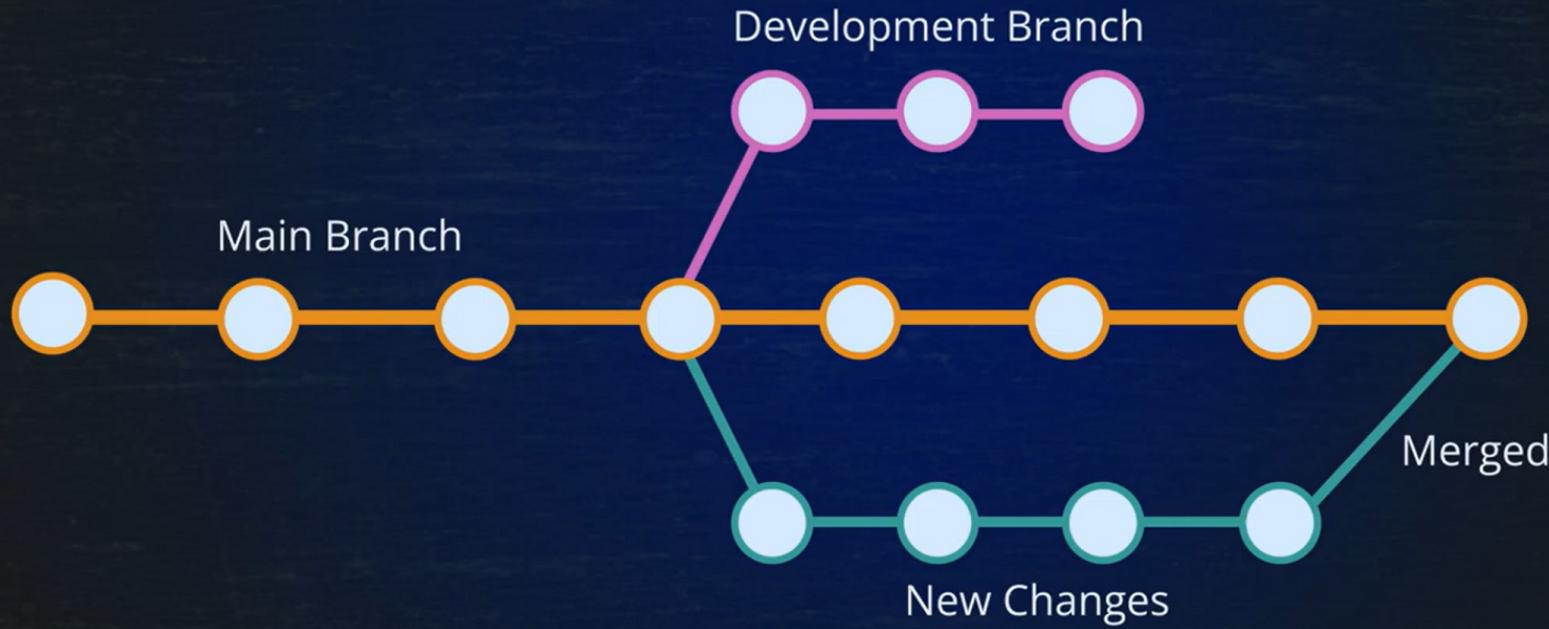
Git - track evolution of workflow
(i.e. transparency)

TRACK CHANGE



When, Who, Why, What

BRANCHES



- Main branch
- Experimentation
- Developments
- Merging

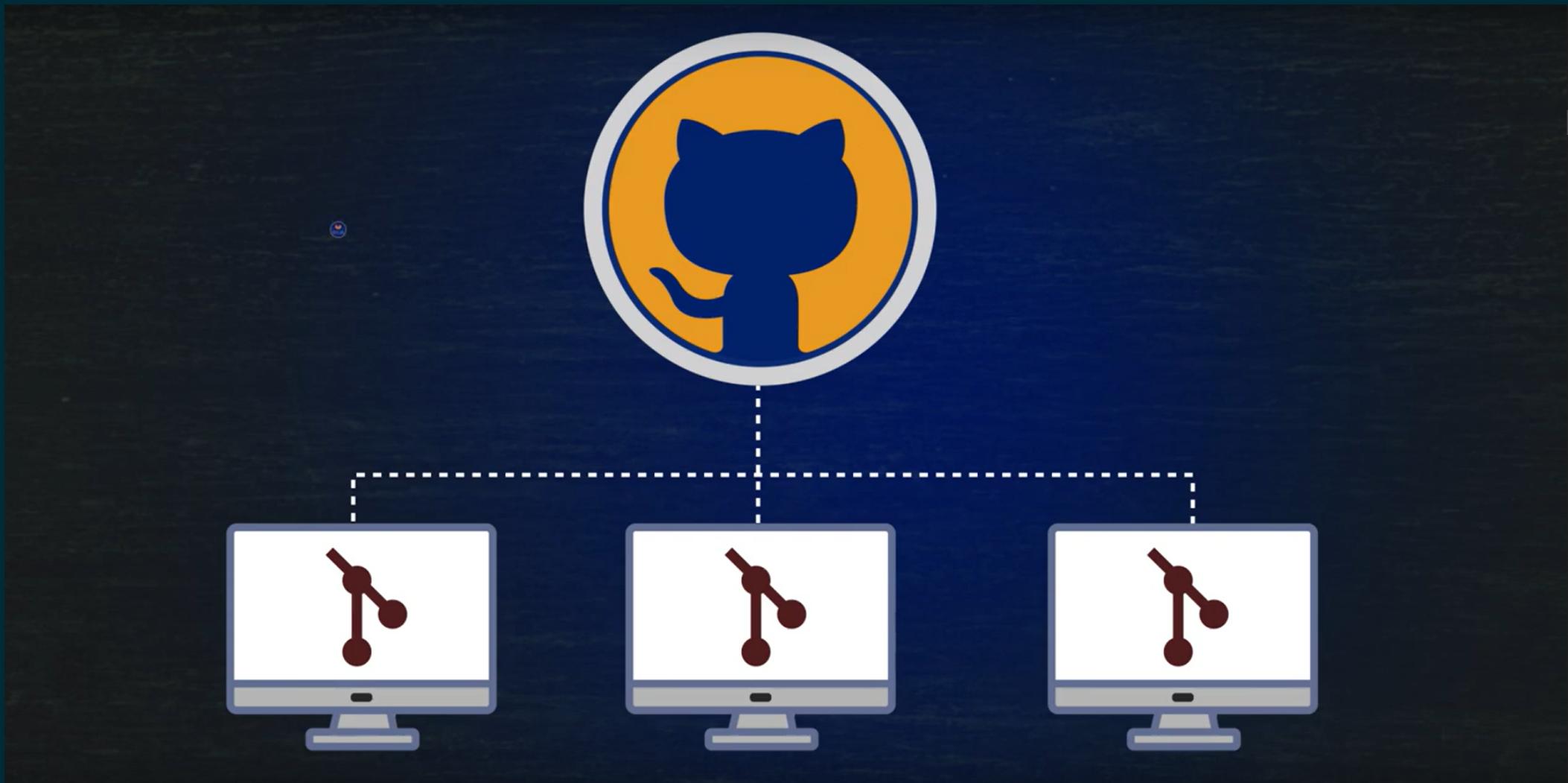
GITHUB

- Profile (store and host) git repos
- Enable collaboration across the globe or private
- Editorial and fine-grain control



GitHub

GIT + GITHUB



HUBS

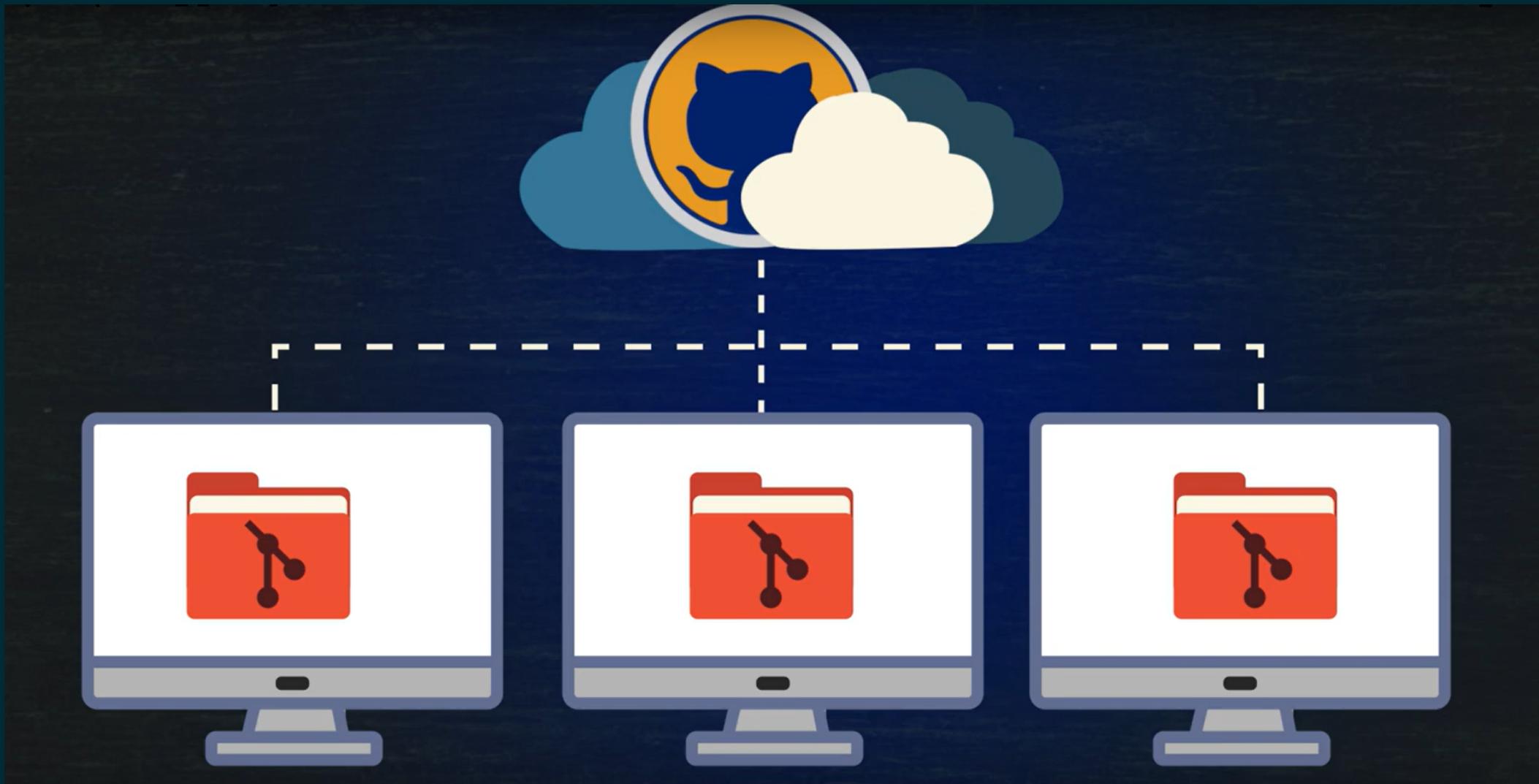
- GitHub
- GitLab
- BitBucket



DUKE SPECIFIC HUBS

- gitlab.oit.duke.edu (netId)
- PACE
- Anywhere that data and coding happens.

FILE DISTRIBUTION AND COLLABORATION



OTHER PROJECT MANAGEMENT FEATURES



Access
Control



Task
Distribution



Bug
Tracking



Wiki
Documentation



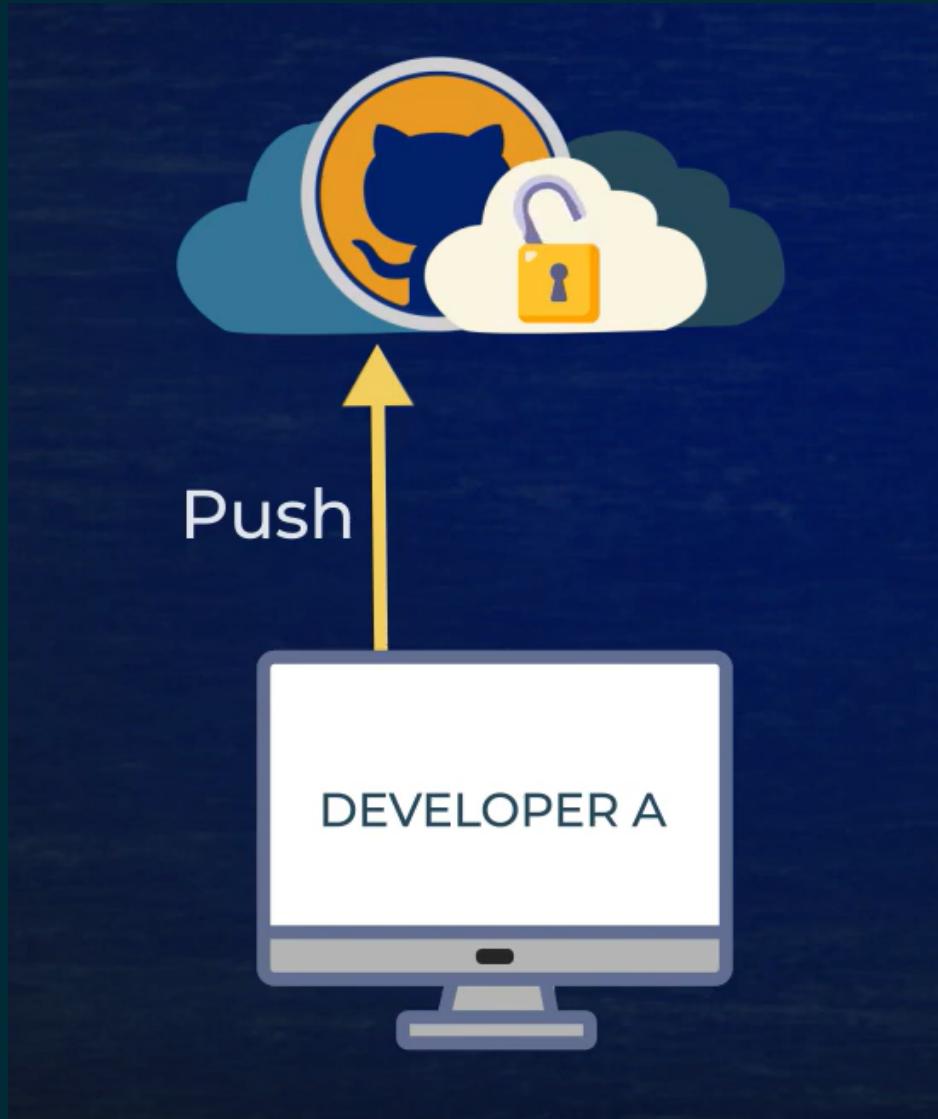
Kanban
Planning

BASIC FEATURES

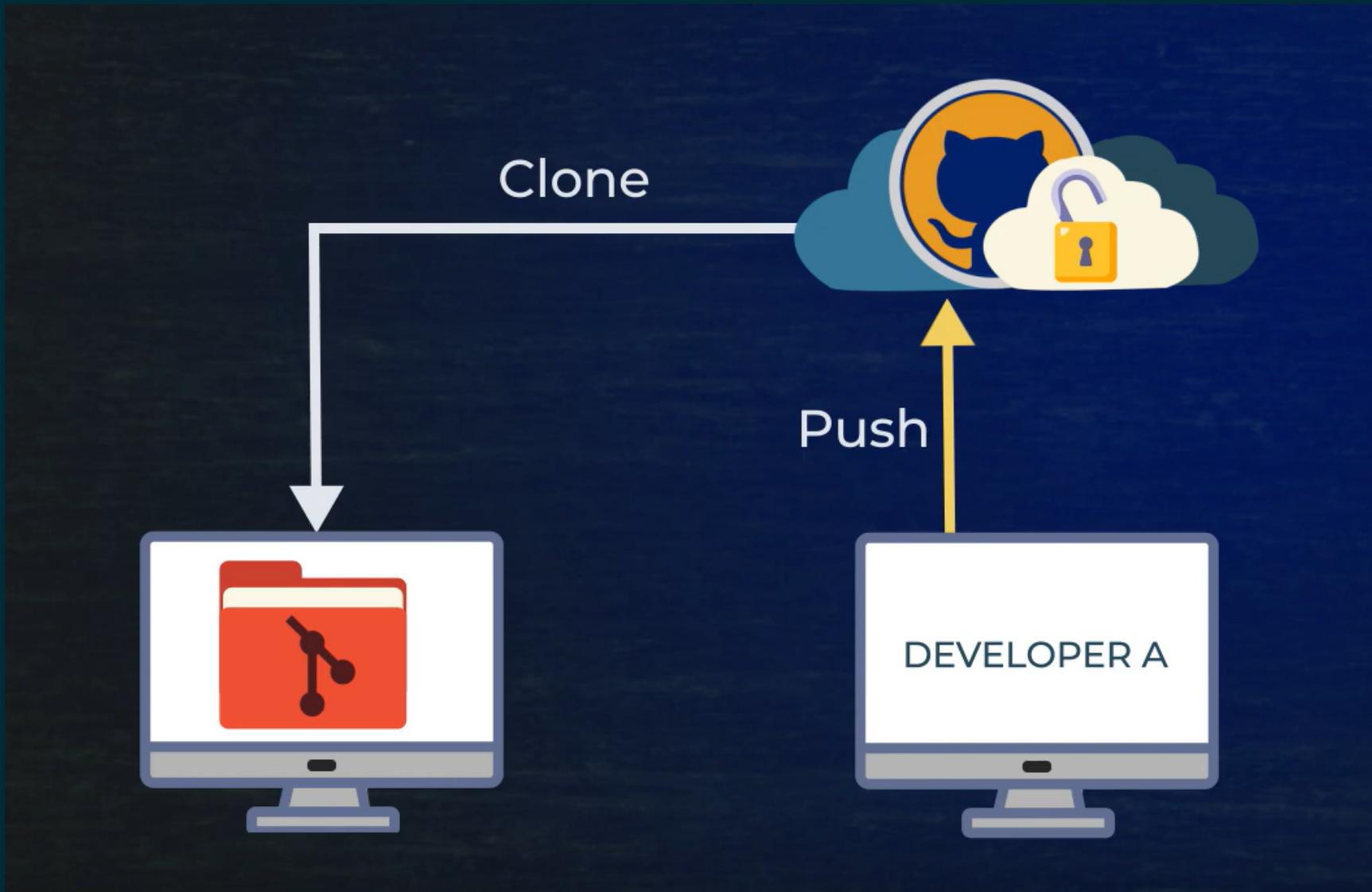
Git features implemented for distribution

- Push
- Public or Private
- Clone / Fork
- Pull Request
- Pull

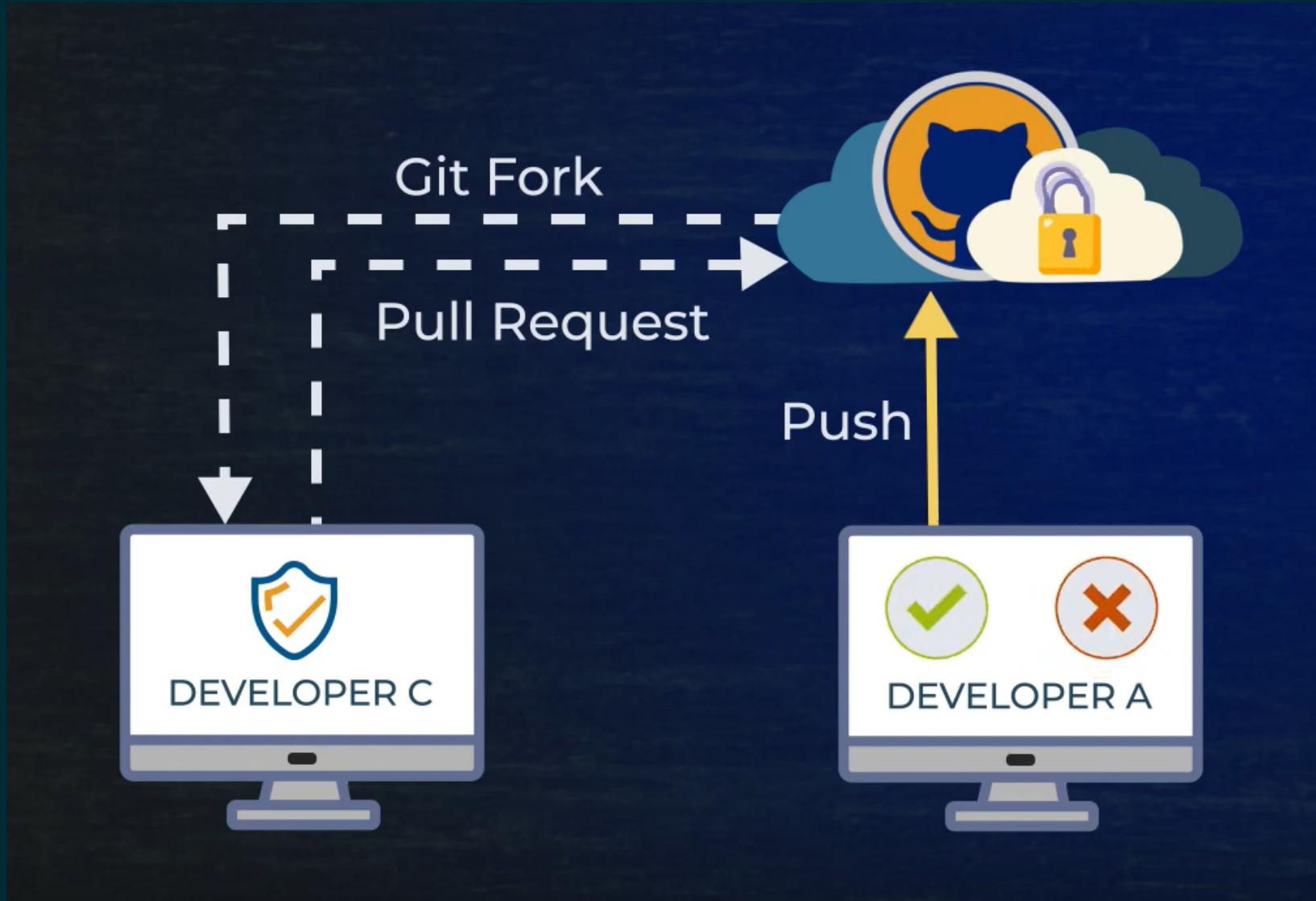
PUSH



CLONE



FORK / PR



SUMMARY

- Git is used to track changes to your repo
- GitHub is used to distribute your git repo and facilitate collaboration

CONTAINERS



SHARING YOUR WORKSPACE

Your computation workspace (i.e. your laptop, desktop, cloud credentials)

Give someone else your laptop so they can play around with your projects

- the code, the data, the settings and configurations?
- Good idea?

Now you can share a copy of your computational environment

HOW

- **Binder:** package and share reproducible computational environments
 - mybinder.org (public BinderHub portal)
- **Zenodo:** general, open repository to deposit research papers, data sets, code, reports and related artifacts and connect to a citable DOI.
- Combine GitHub releases with Zenodo to archive your milestones and share the interactive computation in a binder Hub

BINDER HUB

- Easiest: mybinder.org open and public
 - quarto use binder
- Security demands may push you to use singularity

STEPS

1. Make a GitHub Release at project milestone(s)
2. Connect GitHub to Zenodo
 1. Mint a DOI to a GitHub Release (persistent identifier: citation; milestones)
 2. With DOI, link to ORCID
3. Create a publicly launchable, fully functional computation container of your work

EXAMPLES

- [https://github.com/libjohn/workshop_rfun_iterate?
tab=readme-ov-file#readme](https://github.com/libjohn/workshop_rfun_iterate?tab=readme-ov-file#readme)
- [https://github.com/libjohn/workshop_webscraping?
tab=readme-ov-file#readme](https://github.com/libjohn/workshop_webscraping?tab=readme-ov-file#readme)