





# Reproduction-ready: A proper project setup for Python and beyond

## A Reproducible Research workshop

Simon Stone

*Research Data Services*

*Dartmouth College Libraries*





# About Research Data Services

## Research Data Management

Data Management Plans (DMPs) for sponsored projects

Finding and using 3rd party data

Collection and cleaning of data

Organization and documentation

Publishing and Repositories

## Data Analysis/Visualization

Textual, numeric, spatial data

Reproducible research workflows

Scripting in R: tidyverse core package (i.e., ggplot, dplyr, tydr, tibble, etc.)

Scripting in Python: NumPy, SciPy, Pandas, Scikit-learn, Matplotlib, Seaborn, (OpenCV, PyTorch, TensorFlow, Tesseract, NLTK, etc.)

## Computational Scholarship

Computational project planning

Collections as Data

Storytelling with data and visualizations

Text and data mining

Digital Humanities support

Computational Pedagogy



# Work with us

[ResearchDataHelp@groups.dartmouth.edu](mailto:ResearchDataHelp@groups.dartmouth.edu)

Jeremy Mikecz  
Research Data Science Specialist  
[jeremy.m.mikecz@dartmouth.edu](mailto:jeremy.m.mikecz@dartmouth.edu)  
[dartgo.org/jeremyappts](https://dartgo.org/jeremyappts)

Simon Stone  
Research Data Science Specialist  
[simon.stone@dartmouth.edu](mailto:simon.stone@dartmouth.edu)  
[dartgo.org/meetwithsimon](https://dartgo.org/meetwithsimon)

Lora Leligdon  
Head of Research Data Services  
[lora.c.leligdon@dartmouth.edu](mailto:lora.c.leligdon@dartmouth.edu)  
[dartgo.org/lora](https://dartgo.org/lora)

# What you will learn in this session

- 🌀 What makes managing a computational project challenging
- ↺ What makes a project reproducible
- 🔧 What tools and strategies can you use to make your work with such projects easier
- 🎯 Session Goal: Inspire thinking about project organization

# Components of computational projects

A typical computational project may contain any or all of these components:



Data



Code



Artifacts (e.g., trained models)



Documentation



Administrative documents



Scholarly writing



How to keep track of all of this?



# Challenges of reproducibility

Each of the components faces individual challenges:



Data:

Availability and access



Code and artifacts:

Dependencies, compatibility and portability issues



Documentation:

Sufficient coverage



Administrative documents: Part of the project, but probably should not be shared



Scholarly writing:

Copyright issues and limited access

How to face those challenges without  
too much additional overhead?

# Strategies to improve productivity and reproducibility

## Good strategies:

- 🤝 Don't get in the way of doing the actual work
- 😎 Need little to no extra work
- 😴 Become second nature

## In today's session:

- 👉 Organization of your project folder
- 🏢 Using Visual Studio Code as your project's central HQ



# The project folder: One folder to hold them all

- 🍪 Keep all files related to a project in a single folder
- 🍪 Simple, yet effective: Everything in one place

## Enables:

- 🍪 Version control (git)
- 🍪 Working directory in code editor (e.g., VS Code)
- 🍪 Instantly shareable
- 🍪 Easy context switching between projects

# Using Visual Studio Code and your project folder

- ✧ Integrated Development Environments (IDEs) like VS Code are a great tool to work in a project folder
- ✧ VS Code is not the only such tool, but it is one of the leading ones
- ✧ Combines file browser, text editor, code debugger, PDF viewer, LaTeX support, and more in a single interface
- ✧ For many workflows, you don't have to leave VS Code at all
- ✧ VS Code keeps track of your session (i.e., which files were last open)
- ✧ Switching between projects only takes two clicks



# Installing and setting up VS Code

## Instructions for MacOS Ventura or later:

- 🍏 Navigate to <https://code.visualstudio.com>
- 🍏 Download the latest stable version
- 🍏 Open your Downloads folder
- 🍏 Drag and drop the downloaded file Visual Studio Code onto your Applications folder on the left pane

## Instructions for Windows 11

- 🪄 Navigate to <https://code.visualstudio.com>
- 🪄 Download the latest stable version
- 🪄 Open the downloaded installer
- 🪄 Follow the instructions and click Next three times
- 🪄 When prompted, select (or leave selected) all checkboxes under Other:
- 🪄 Click Install



# A quick tour of VS Code

- UI elements:
  - Activity bar
  - Primary side bar
  - Editor groups
  - Panel
  - Status bar



# A quick tour of VS Code

- Opening a folder
- Open files
- Preview vs sticky tabs
- Rearranging tabs
- Accessing a Terminal/Shell



# Some useful extensions for VS Code

- Python
- Jupyter
- File viewers (CSV, Excel, JSON, PDF, ...)
- SQL database browser
- LaTeX support
- ...



# Setting up a Python project

- Git
- Venv
- Folder structure
- README
- requirements.txt
- .gitignore



# An example project folder

- [www.dartgo.org/rds-example-project](http://www.dartgo.org/rds-example-project)

**Disclaimer:** Nothing in here should be understood as a one-size-fits-all or mandatory.

Depending on your project, your team, and your personal preferences, you can (and should!) adapt this to fit your needs.

The important point is to think about structure and make conscious decisions that benefit the reproducibility and intelligibility of your analyses and modeling for all stakeholders (yourself included!).





# Selectively sharing your project





- Publishing your project on GitHub is a great way to share it:
  - Create a new empty repository and clone it *before* you start your project
  - Create a new repository from an *existing* local repository or folder
- If parts of your project should not be shared:
  - Put the folder path containing the private files in the `.gitignore` file
- You can alternatively have multiple repositories in your project:
  - Make one subfolder per subproject (e.g., `admin`, `code`, `manuscript`)
  - Create separate repos in each subfolder

# Adventure Time!


- Pick one of your existing projects or think of a new one
- Create a project folder and open it with VS Code
- Copy (!) existing files over or create new ones
- Find some extensions that might be useful to your project
- Set up a virtual environment
- Set up a git repository
- Explore!



# Summary and next steps

-  All files for a project are best kept in a single folder tree
-  Using an IDE can help you navigate the project folder efficiently
-  Context switching is as easy as opening another folder
-  Version control (git) can help (selectively) sharing your project

## Next steps:

-  Learn about good practice for cross-platform compatibility (Linux, MacOS, Windows)
-  Try it out and tell us how it went!



# Thank you