Positron Setup Guide
Sociol 8607: Causal Modeling

Jan 15, 2026

Agenda

Outline

Today we will install

- Python (and uv)
    - virtual environments
- Positron (**I**ntegrated **D**evelopment **E**nvironment) for Python & R
- LaTeX for creating PDFs
- GitHub Desktop (and git)

(there are different steps for Windows vs. MacOS)

Python

Windows dependency

Positron Setup
Guide

Agenda
Python
Positron
Virtual
Environments
LATEX
GitHub
Extra

- Windows dependency: latest C++ Redistributable
  positron.posit.co/install.html
  - Prerequisites ⟶ Windows

Overview

Setting up Python is a two-step process...

- First we will install uv, a tool for managing Python and Python projects

- Then we can use uv to install Python (& virtual environments inside Positron)

  - later on we will use also uv to install Python packages

- To get started, open: positron.posit.co/install.html

  - **Prerequisites** $\longrightarrow$ Python setup $\longrightarrow$ uv
  - click on **Installation** left-hand side (ToC)

uv - MacOS

- Open a Terminal

  - command-space (to open Spotlight) and type Terminal.app (auto-complete should kick in)

  - the Terminal is a powerful tool for managing your file system and running programs (Positron has a built-in terminal!)

- Inside the terminal, type the following command

```
curl -LsSf https://astral.sh/uv/install.sh | sh
```

  - (copy and paste from website)

  - If curl is not available, then try

```
wget -qO- https://astral.sh/uv/install.sh | sh
```

uv - Windows

- Open a PowerShell (Windows equivalent to a terminal)
- Inside the PowerShell, type the following command

```
powershell -ExecutionPolicy ByPass -c "irm https://astral.sh/uv/install.ps1 | iex"
```

  - (copy and paste from website)
- Now close the PowerShell and open a new one
  - (this should ensure that PowerShell can find the newly installed uv tool)

Python

Positron Setup
Guide

Agenda

Python

Positron

Virtual
Environments

LaTeX

GitHub

Extra

- Now we can use uv to install Python!
- Inside the terminal or powershell, simply run

```
uv python install
```

  - this should install the latest version (3.14)

- In the future (if needed), you can install a different version of Python

```
uv python install 3.13
```

Positron

Positron

- Download Positron installer for Windows or MacOS positron.posit.co/download.html

  - on Windows 11, this required 1.34GB of disk space

- After installing Positron and starting the program, Windows 11 security feature may ask if you want to allow *KCserver* (you do)

Virtual Environments

Motivation

- Python development moves fairly rapidly
  - Not all projects/packages are moving at the same speed, which can cause headaches.
- Consider the following...
  - Your Project 1 depends on Package A (version 2.7)
  - Your new Project 2 depends on Package A (version 3.1)
  - When you update Package A, Project 2 is good, but it breaks Project 1!

- We can mitigate some of these problems by installing packages in separate locations

  - Project 1 looks in
    /home/l_yamal/proj1/packages/pkga_v27
    to find Packge A (version 2.7)

  - while Project 2 looks in
    /home/l_yamal/proj2/packages/pkga_v31
    where version 3.1 is installed

- These separate locations are implemented as
  Virtual Environments

Python Virtual Environment

- For each project involving Python, you should set up a *virtual environment*

- Inside Positron, open (or create) the folder where you want to set up your project: `File` ⟶ `Open Folder...`

- Open the *Command Palette*
    - MacOS: command-shift-P
    - Windows: control-shift-P

- Type: `Python: Create Virtual Environment`
    - choose the uv option

LATEX

LaTeX

- Positron comes installed with *quarto*, a publishing system that can create documents, presentation, webpages, etc.

- Quarto uses LaTeX to create PDFs, which we can install by running the following command in the Positron terminal

```
quarto install tinytex
```

  - (takes a few minutes to run)

GitHub

Gintro

- Messy projects, messy code, and sharing/collaboration?

- GitHub to the rescue
  - GitHub is an on-line service that stores files, tracks versions, and more!

- `git` is the program used for *version control* (i.e., managing and tracking different versions of your files)
  - Positron adds features for running git on your computer Positron's VC docs
  - an excellent resource: git book

Gintro (cont.)

- "and more" you say??

  - renders *markdown* code (used to format text)
    test drive: https://markdownlivepreview.com

  - websites,
    e.g. https://buckipr.github.io/R_Working_Group/

  - create different *branches* of you project

  - advanced features for software developers (making it
    popular and a common source for latest versions)

GitHub

Sign up for a free GitHub account:

- https://github.com/signup
    - you will need to create a new username and password
    - the *Free* subscription will work just fine
    - there is also an educational account that provides more resources
    - a verification email will be sent (check spam folder)
- You may need to set up 2-Factor Authentication

GitHub Desktop

- GitHub Desktop is a user-friendly interface for `git`
- Download and install the GitHub Desktop application
- https://desktop.github.com/
    - for Windows & MacOS
- GitHub Desktop will present a Welcome Screen
    - sign in using the username and password you created previously

git: concepts and actions

- **Clone** - copy a repository to your computer for the first time

- **Pull** - update your local copy of the repo with any new changes on the remote repo

  - clone is basically a pull for the first time

  - you can also *fetch* - download changes (so you can inspect them) but do not sync them with your local repo

- Make new files and changes to your local repo, then **STAGE** all the files you want to keep track of

  - this includes adding, renaming/moving, & removing files with any changes that you want to track

- **Commit** - save your changes (they are now tracked)
    - typically include a message describing the commit
    - this creates a new version (or snap shot) of your project
    - you can go back and look at previous commits (or revert to that version) and look at differences between commits
- **Push** - send your changes to GitHub
    - local repo will now be identical to remote repo

Extra

- open PowerShell and run the following command

  ```
  Get-FileHash installer.exe -Algorithm SHA256 | Set-ContentPath f1.txt
  ```

- copy & paste SHA-256 from website into a text file f2.txt

- then run the following in ps

  ```
  Compare-Object (Get-Content f1.txt) (Get-Content f2.txt)
  ```

SHA-256 - MacOS

- open Terminal and run the following commands

```
sha256sum installer.dmg > f1.txt
echo SHA-256-from-website > f2.txt1x
```