2023 INTERNET2 TECHNOLOGY exchange

Network Automation Tapas

Frank Seesink, Title, Organization
Maria Isabel Gandia, CSUC/RedIRIS (GÉANT project),
Amy Liebowitz, Network Architect, University of Michigan
AJ Ragusa, Manager - Network Automation and Performance, GlobalNOC @ IU
James Harr, NetDevOps Engineer, Internet2
Shannon Byrnes, NetDevOps Engineer, Internet2

Network Automation Tapas

Bite-sized talks to give the audience a little something to chew on

2023 INTERNET2 TECHNOLOGY exchange

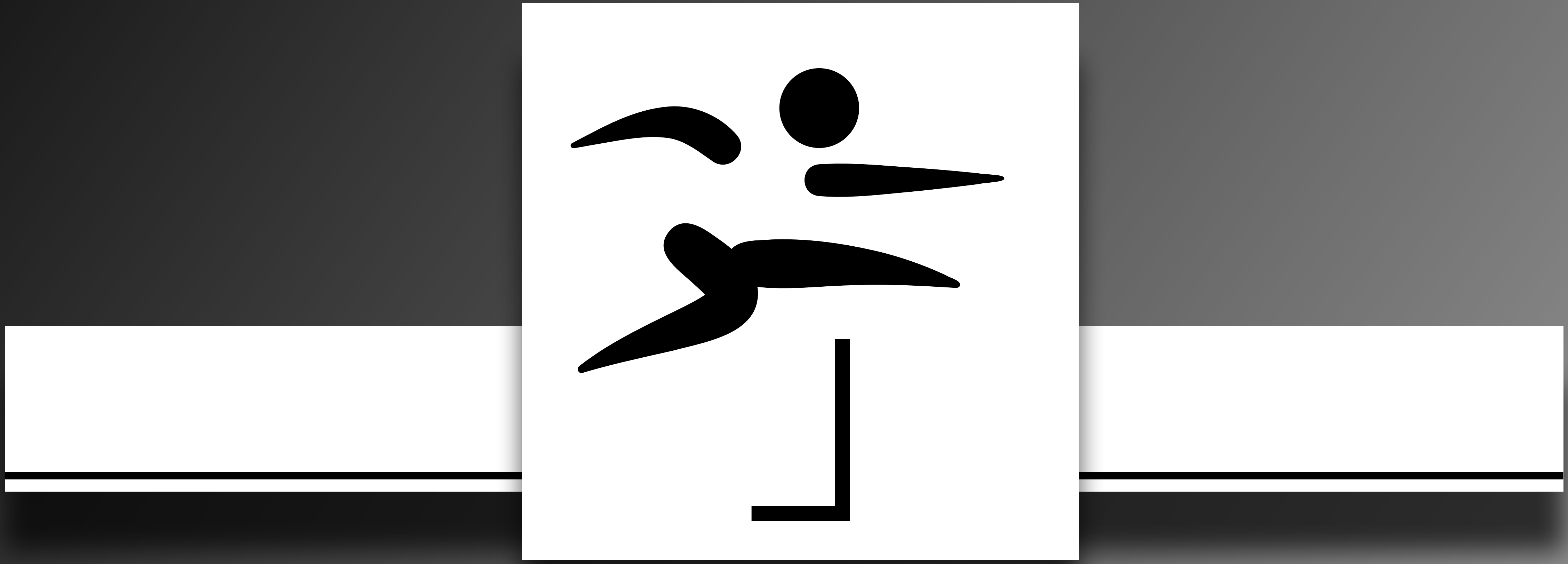
THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Network Automation Tapas

- **Frank Seesink**, Senior Network Engineer
  UNC Chapel Hill
- **Maria Isabel Gandia**
  CSUC/RedIRIS (GÉANT project)
- **Amy Liebowitz**
  University of Michigan
- **AJ Ragusa**
  GlobalNOC
- **James Harr**
  Internet2
- **Shannon Byrnes**, NetDevOps Engineer
  Internet2

Why this session?

Network Automation Tapas

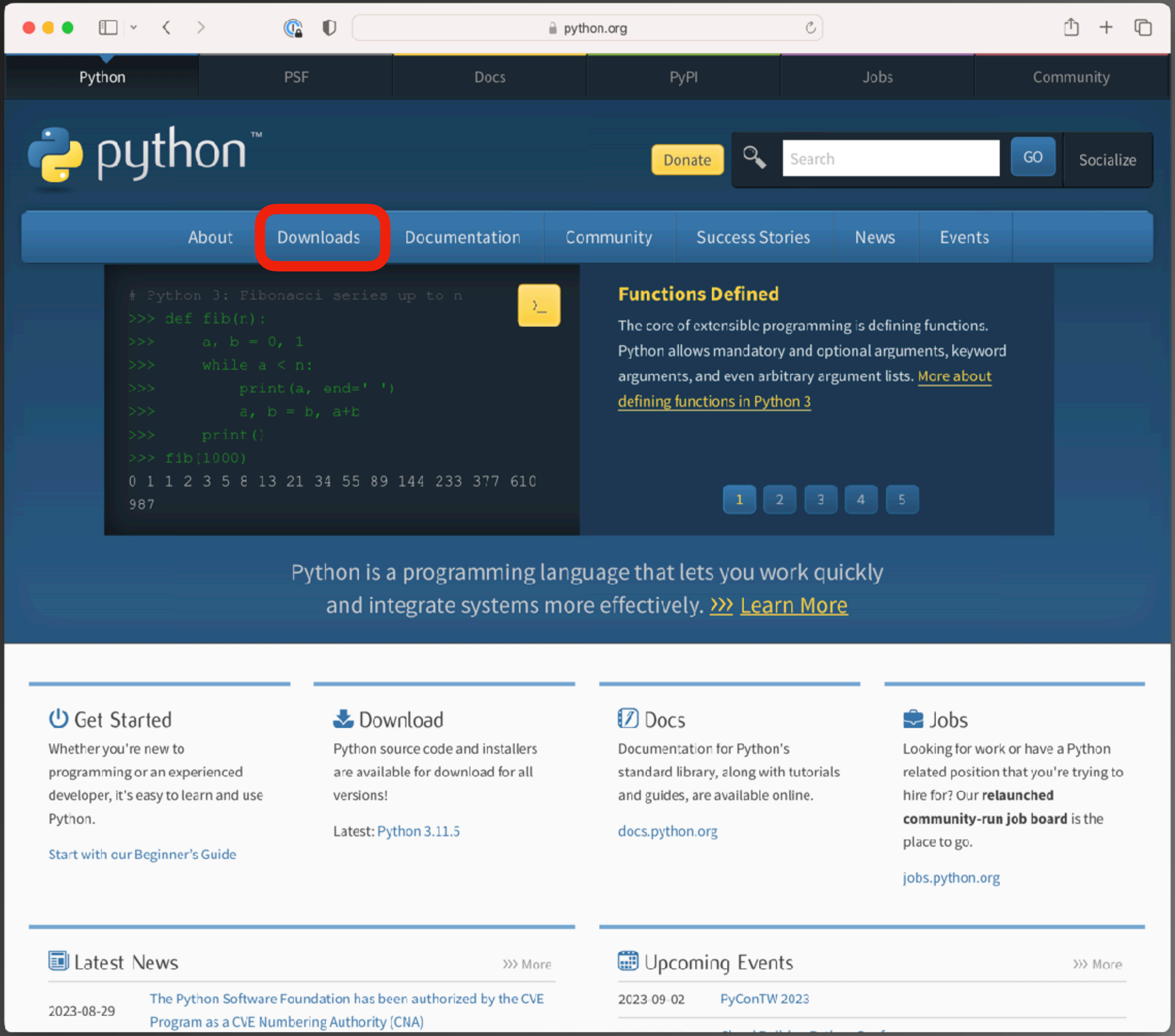


YAML
xml
$_
Yum!

Getting Started with Python

Python Software Foundation

Option #1



```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
>>>     print()
>>> fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
987
```

Functions Defined

The core of extensible programming is defining functions. Python allows mandatory and optional arguments, keyword arguments, and even arbitrary argument lists. More about defining functions in Python 3

1 2 3 4 5

Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

Get Started
Whether you're new to programming or an experienced developer, it's easy to learn and use Python.
Start with our Beginner's Guide

Download
Python source code and installers are available for download for all versions!
Latest: Python 3.11.5

Docs
Documentation for Python's standard library, along with tutorials and guides, are available online.
docs.python.org

Jobs
Looking for work or have a Python related position that you're trying to hire for? Our **relaunched community-run job board** is the place to go.
jobs.python.org

Latest News >>> More
2023-08-29 The Python Software Foundation has been authorized by the CVE Program as a CVE Numbering Authority (CNA)

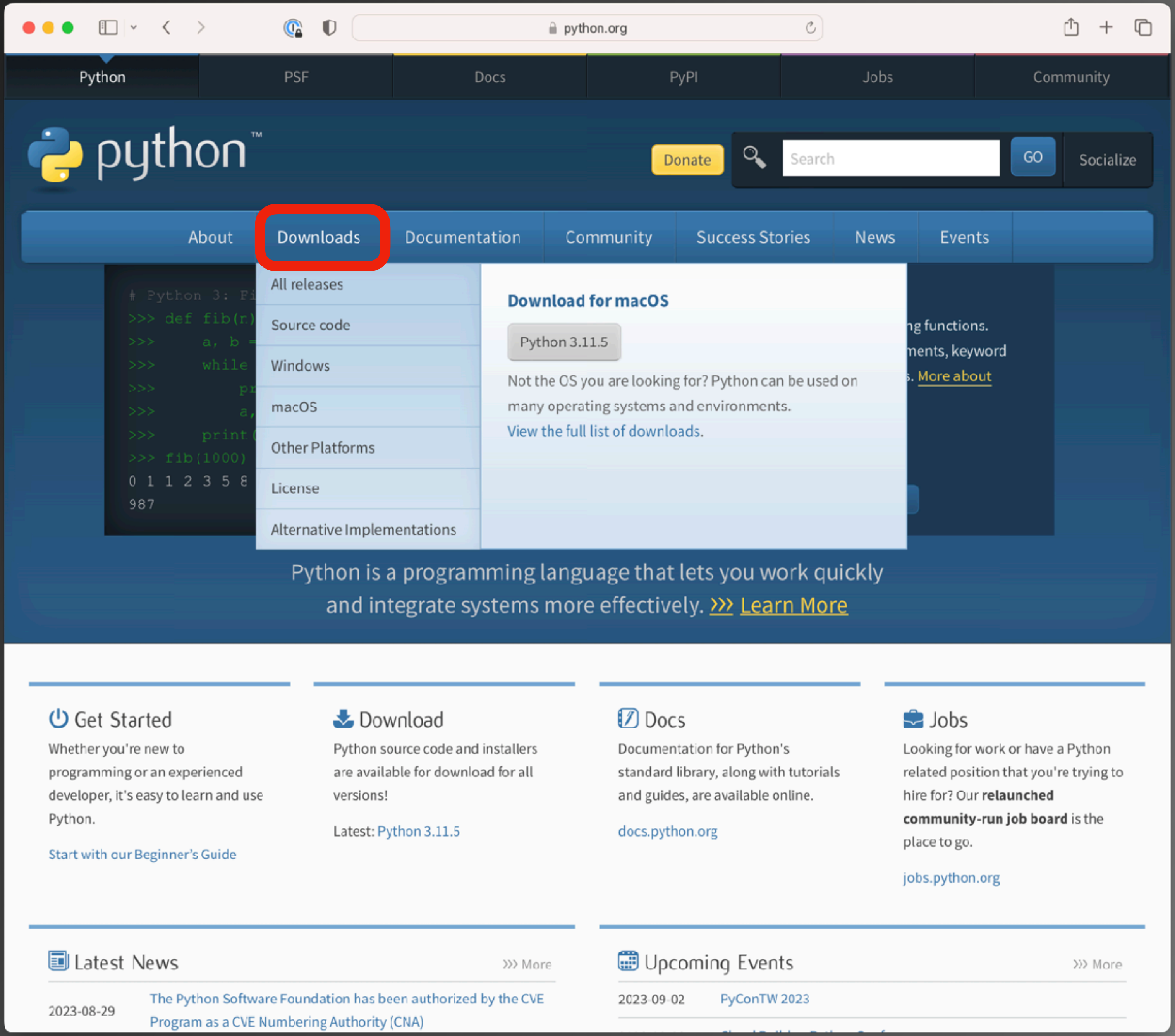
Upcoming Events >>> More
2023-09-02 PyConTW 2023

https://www.python.org/

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Python Software Foundation

Option #1



```
# Python 3: Fi
>>> def fib(n)
>>>     a, b =
>>>     while
>>>         pr
>>>         a,
>>>     print(
>>> fib(1000)
0 1 1 2 3 5 8
987
```

Download for macOS

Python 3.11.5

Not the OS you are looking for? Python can be used on many operating systems and environments. View the full list of downloads.

ng functions.
ments, keyword
s. More about

Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

Get Started

Whether you're new to programming or an experienced developer, it's easy to learn and use Python.

Start with our Beginner's Guide

Download

Python source code and installers are available for download for all versions!

Latest: Python 3.11.5

Docs

Documentation for Python's standard library, along with tutorials and guides, are available online.

docs.python.org

Jobs

Looking for work or have a Python related position that you're trying to hire for? Our **relaunched community-run job board** is the place to go.

jobs.python.org

Latest News >>> More

2023-08-29 The Python Software Foundation has been authorized by the CVE Program as a CVE Numbering Authority (CNA)

Upcoming Events >>> More

2023-09-02 PyConTW 2023

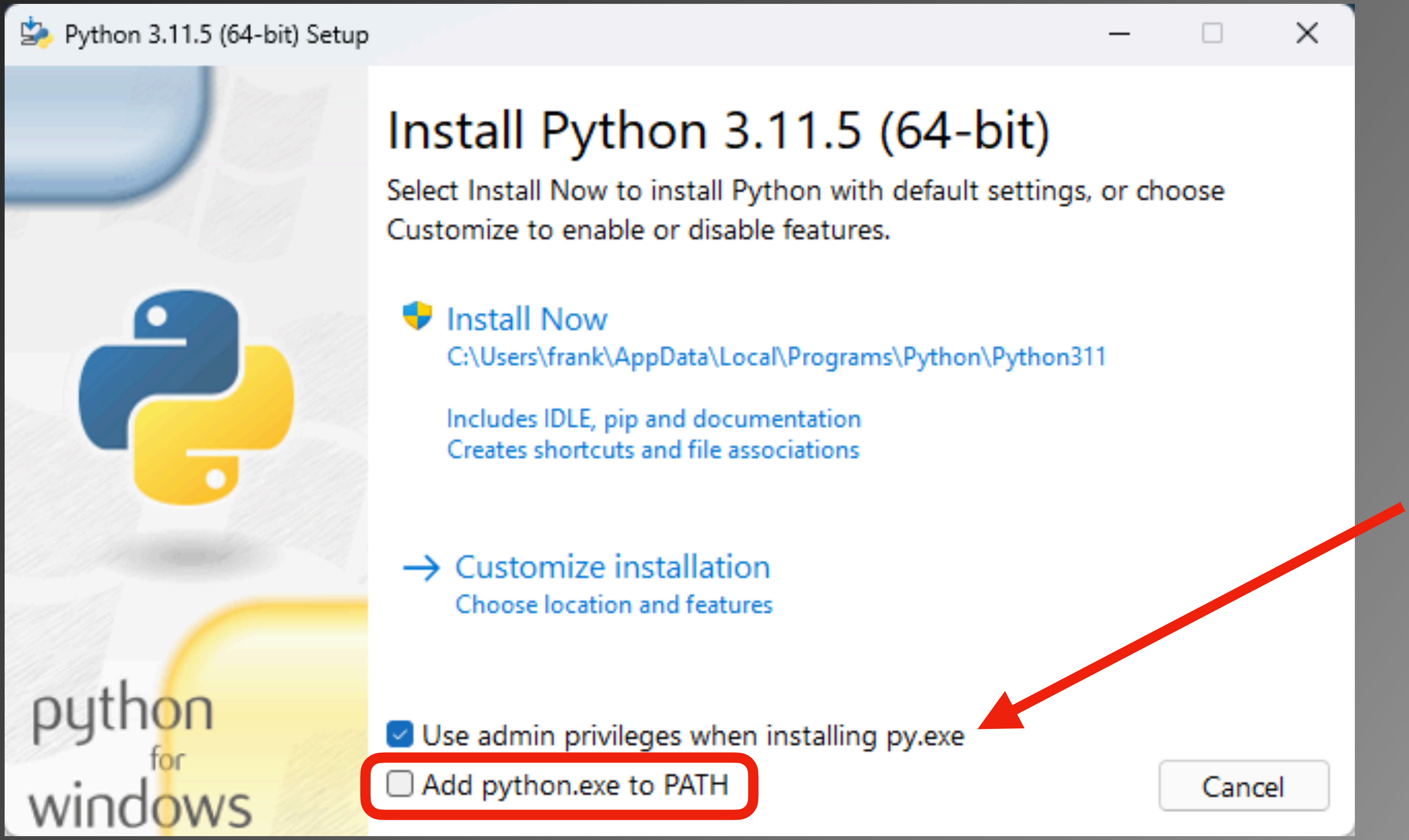
mac OS

https://www.python.org/

Installing Python

for Windows

Install Python - Windows



Python 3.11.5 (64-bit) Setup

Install Python 3.11.5 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

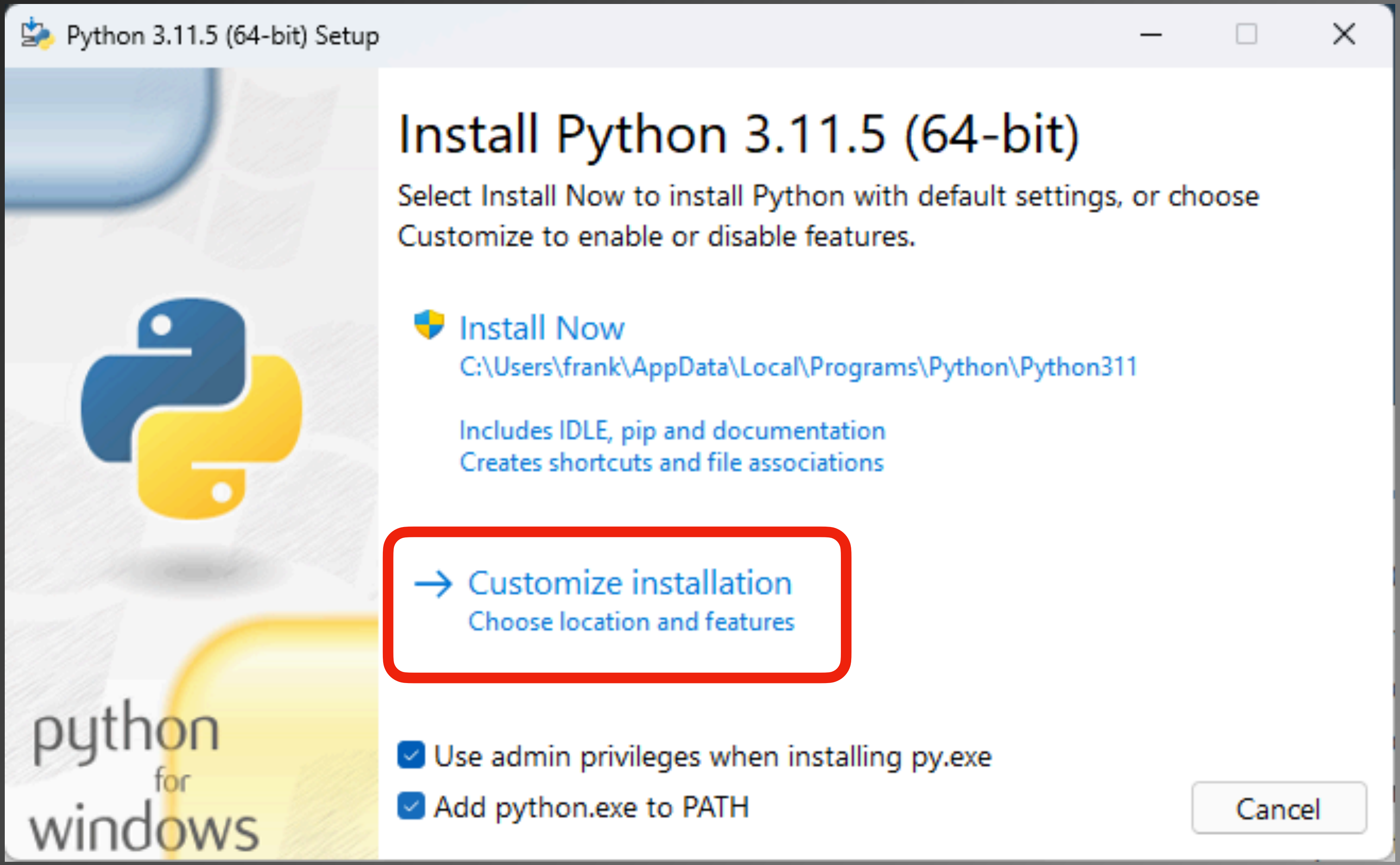
Install Now
C:\Users\frank\AppData\Local\Programs\Python\Python311

Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

☑ Use admin privileges when installing py.exe
☐ Add python.exe to PATH

Install Python - Windows



Python 3.11.5 (64-bit) Setup

Install Python 3.11.5 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

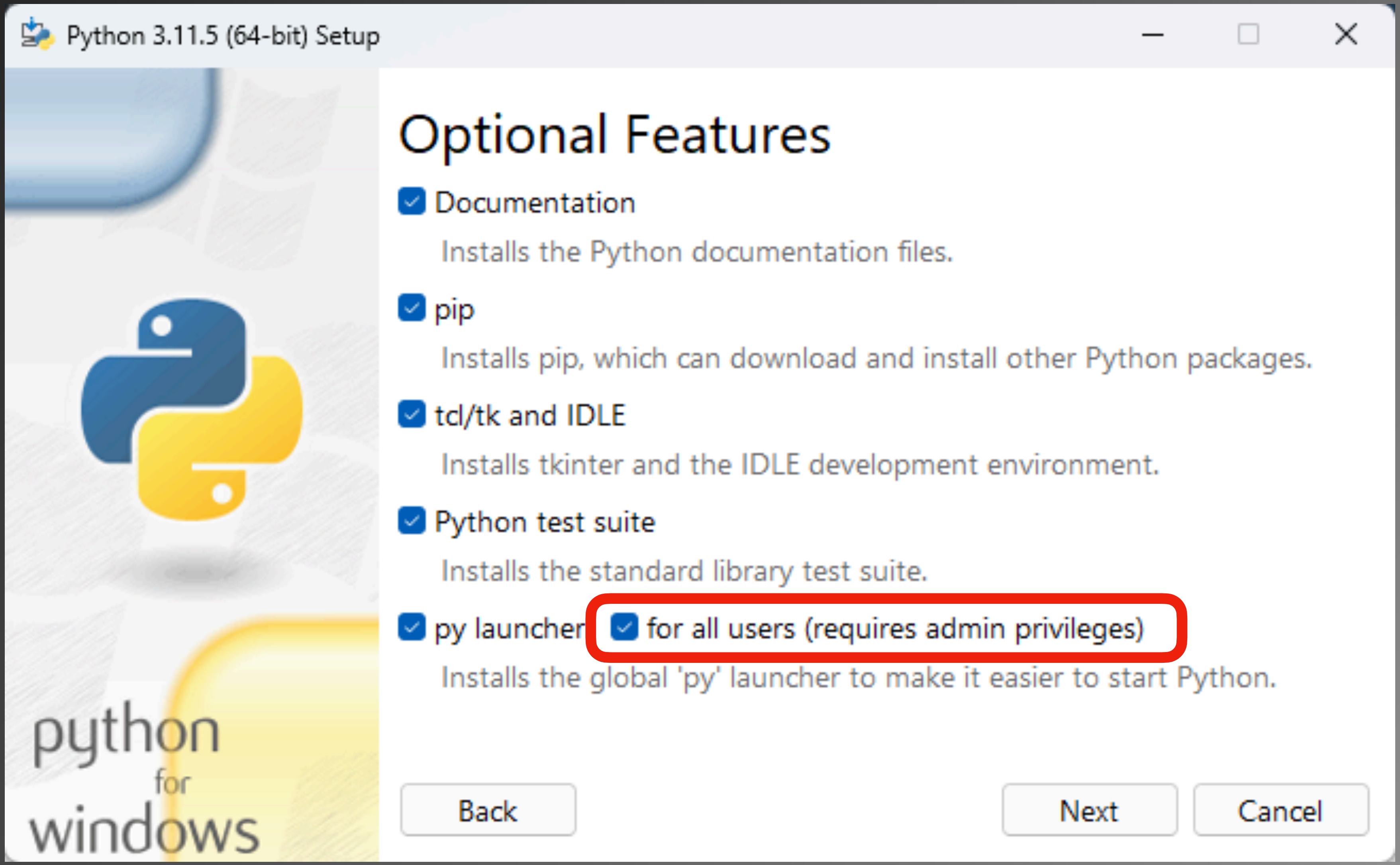
Install Now
C:\Users\frank\AppData\Local\Programs\Python\Python311

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

✓ Use admin privileges when installing py.exe
✓ Add python.exe to PATH

Install Python - Windows

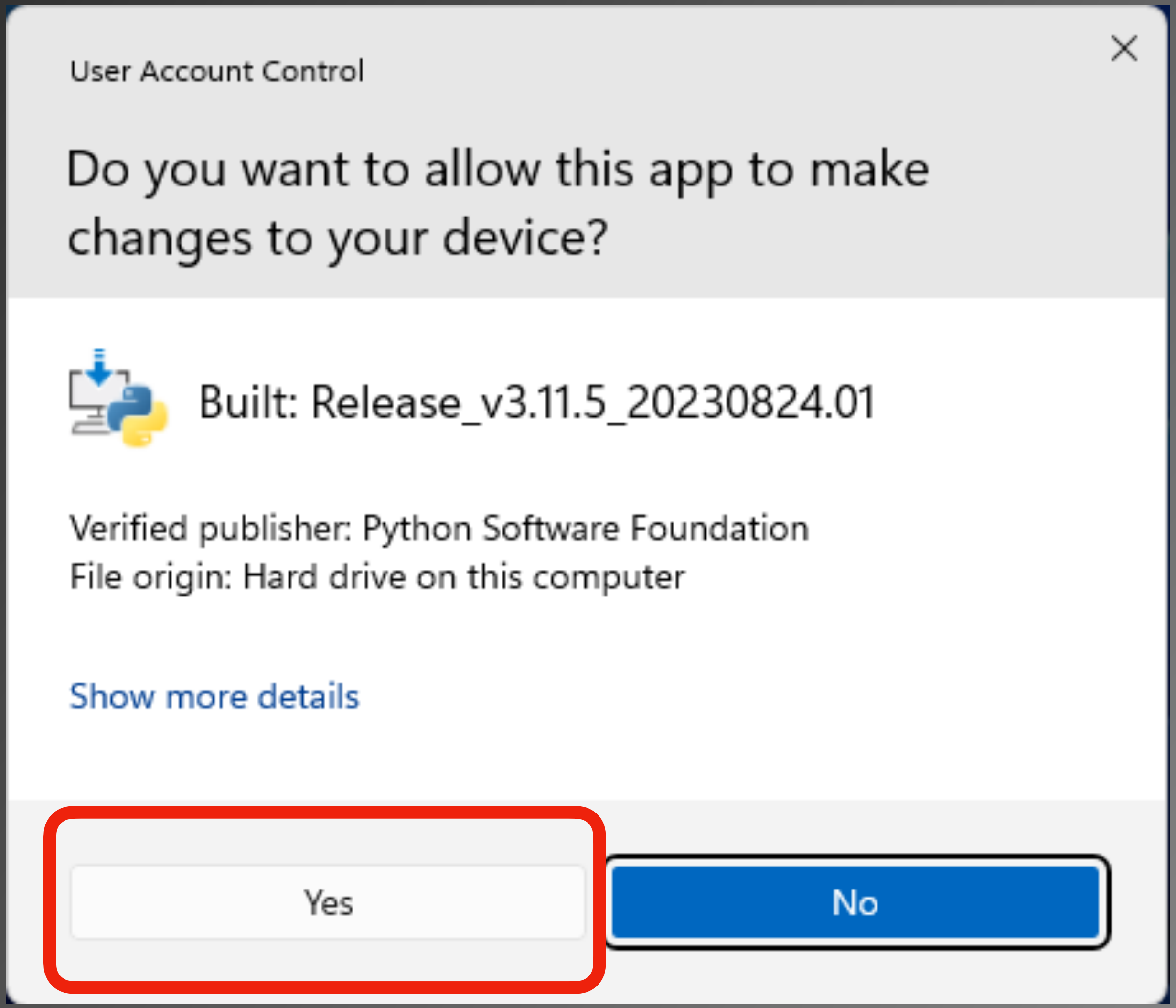


Python 3.11.5 (64-bit) Setup

Optional Features

- [✓] Documentation
  Installs the Python documentation files.
- [✓] pip
  Installs pip, which can download and install other Python packages.
- [✓] tcl/tk and IDLE
  Installs tkinter and the IDLE development environment.
- [✓] Python test suite
  Installs the standard library test suite.
- [✓] py launcher [✓] for all users (requires admin privileges)
  Installs the global 'py' launcher to make it easier to start Python.

Install Python - Windows

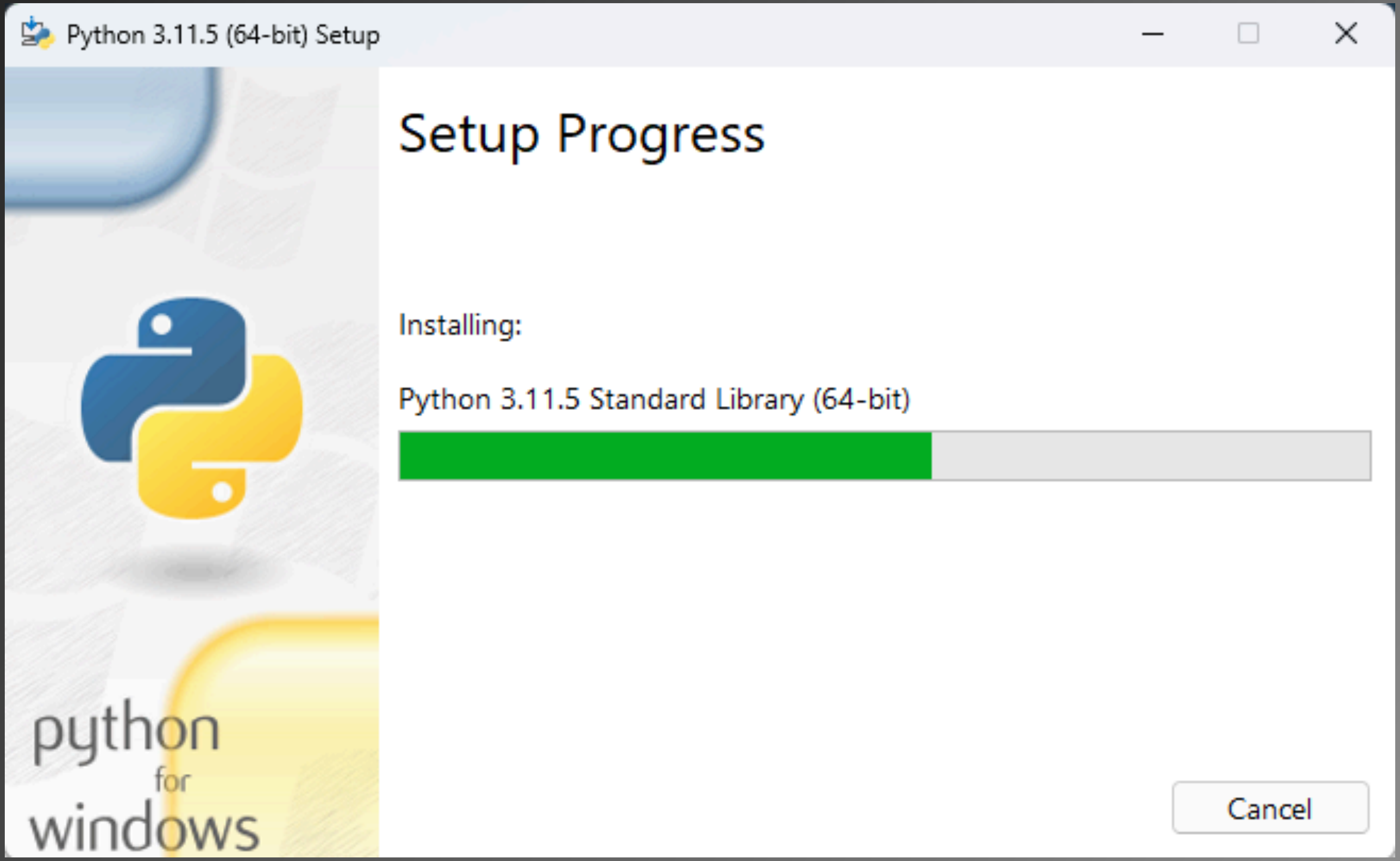

User Account Control

Do you want to allow this app to make changes to your device?

Built: Release_v3.11.5_20230824.01

Verified publisher: Python Software Foundation
File origin: Hard drive on this computer

Install Python - Windows



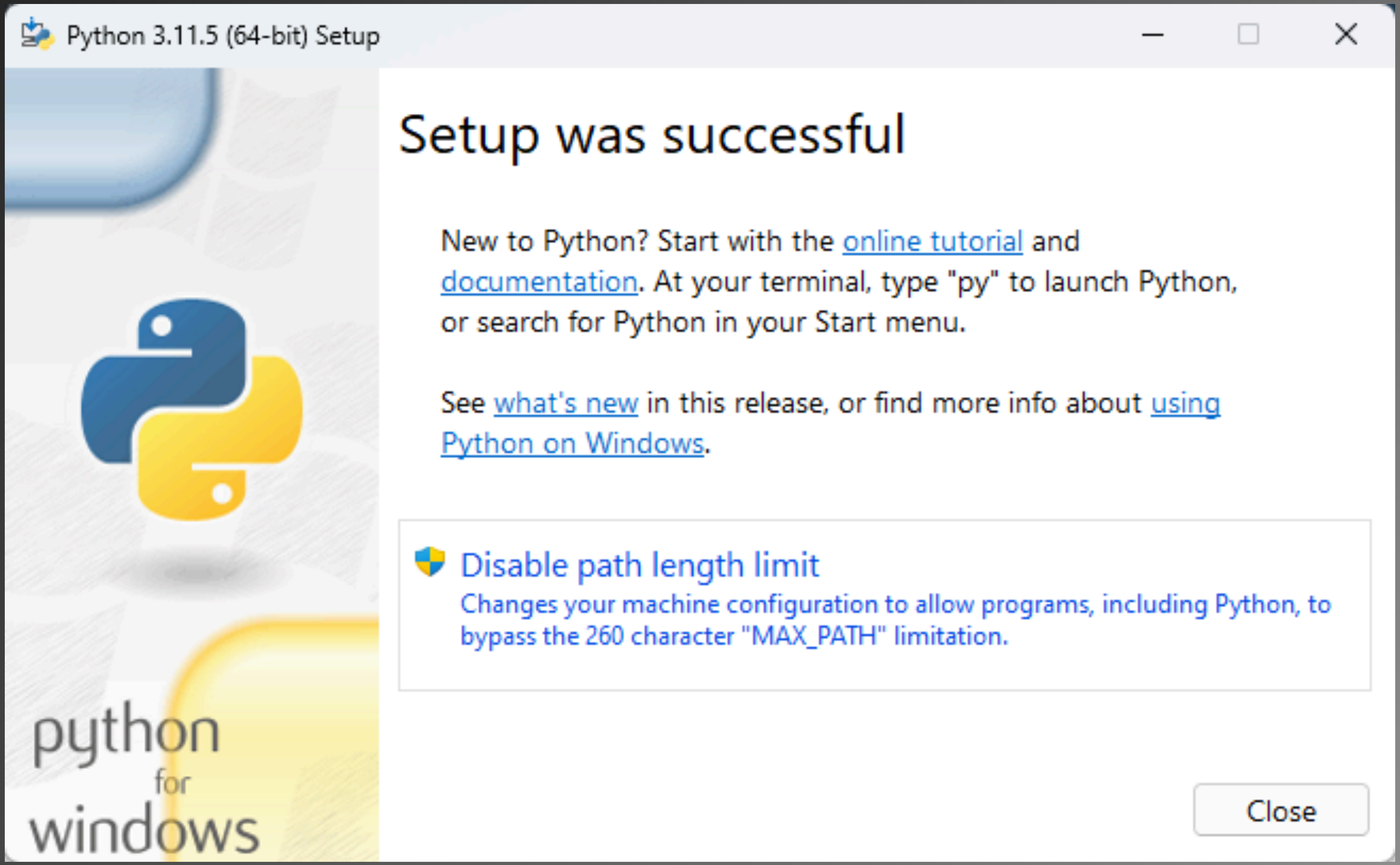
Python 3.11.5 (64-bit) Setup

Setup Progress

Installing:

Python 3.11.5 Standard Library (64-bit)

Install Python - Windows


Python 3.11.5 (64-bit) Setup

Setup was successful

New to Python? Start with the online tutorial and documentation. At your terminal, type "py" to launch Python, or search for Python in your Start menu.

See what's new in this release, or find more info about using Python on Windows.

Disable path length limit
Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.

Install Python - Windows

Python.org Windows Installer installs Python in

```
C:\Users\<user>\AppData\Local\Programs\Python\Python311\
```

Python modules (e.g., seen using **`pip list -v`**) are located in

```
C:\Users\<user>\AppData\Local\Programs\Python\Python311\
Lib\site-packages\
```

Install Python - Windows

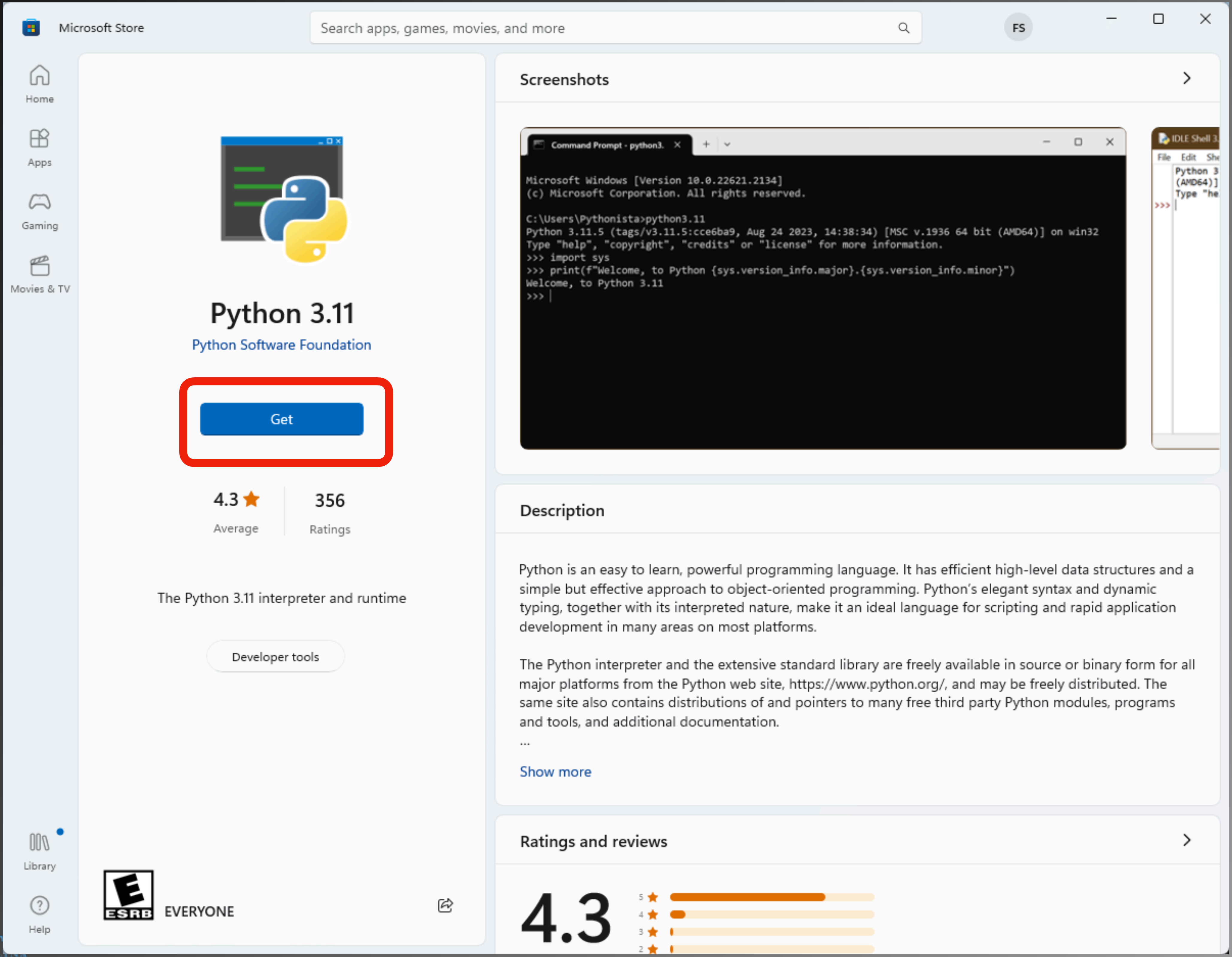
Option #2: Microsoft Store

Simply

1. open the Microsoft Store and search for “python”, or

2. open PowerShell/Command Prompt and just type `python` to bring up the Store.

Install Python - Windows



```
Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Pythonista>python3.11
Python 3.11.5 (tags/v3.11.5:cce6ba9, Aug 24 2023, 14:38:34) [MSC v.1936 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import sys
>>> print(f"Welcome, to Python {sys.version_info.major}.{sys.version_info.minor}")
Welcome, to Python 3.11
>>>
```

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python web site, https://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

...

Install Python - Windows

Microsoft Store installs Python in

```
C:\Users\<user>\AppData\Local\Microsoft\WindowsApps\
```

Python modules (e.g., seen using **`pip list -v`**) are located in

```
C:\Users\<user>\AppData\Local\Packages\
PythonSoftwareFoundation.Python.3.11_…\LocalCache\
local-packages\Python311\site-packages\
```

Install Python - Windows

Option #3: Chocolatey

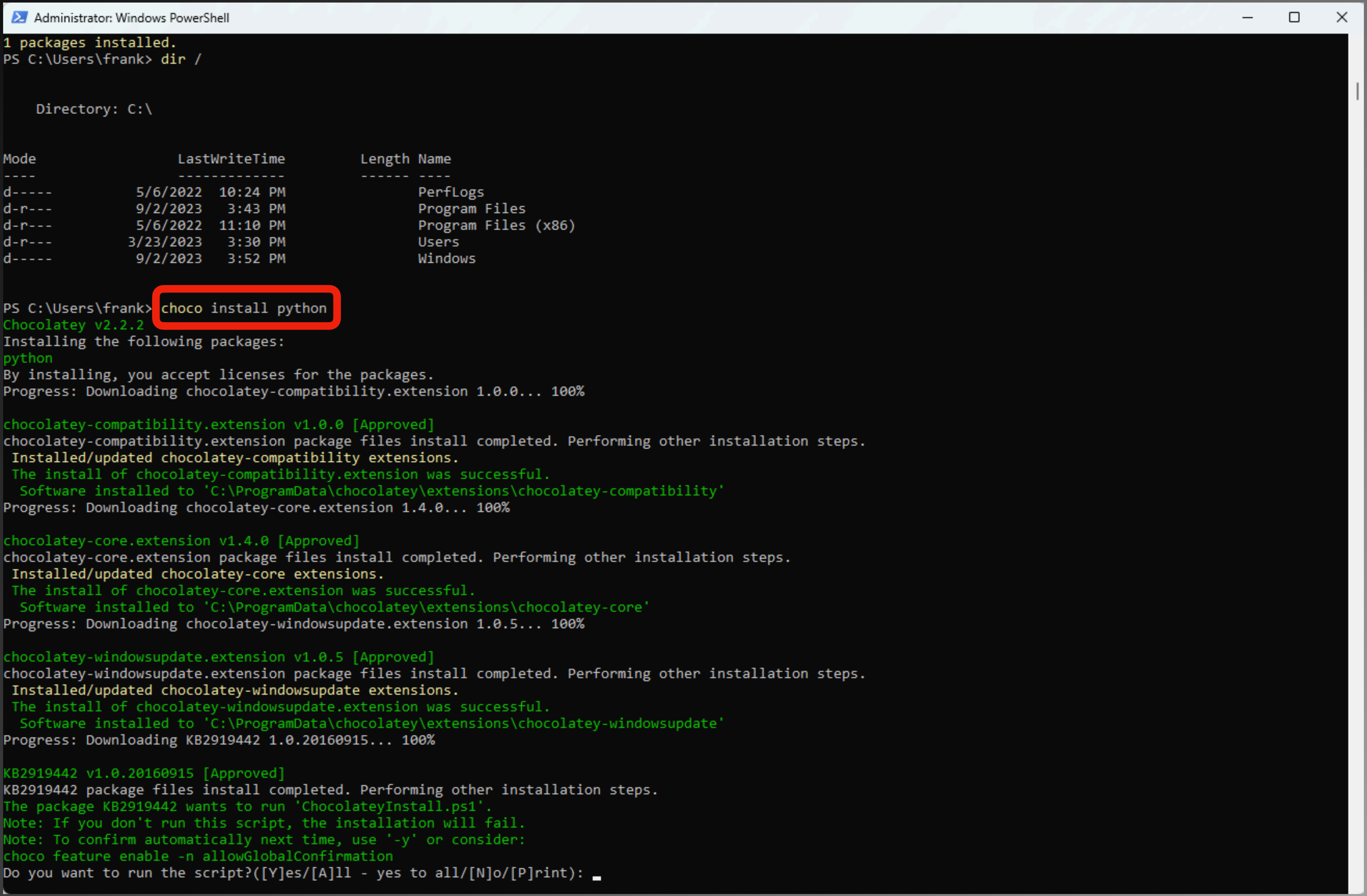
The Package Manager for Windows

https://chocolatey.org/

Simply open PowerShell as an administrative shell (i.e., “Run as Administrator”) and enter

```
choco install python
```

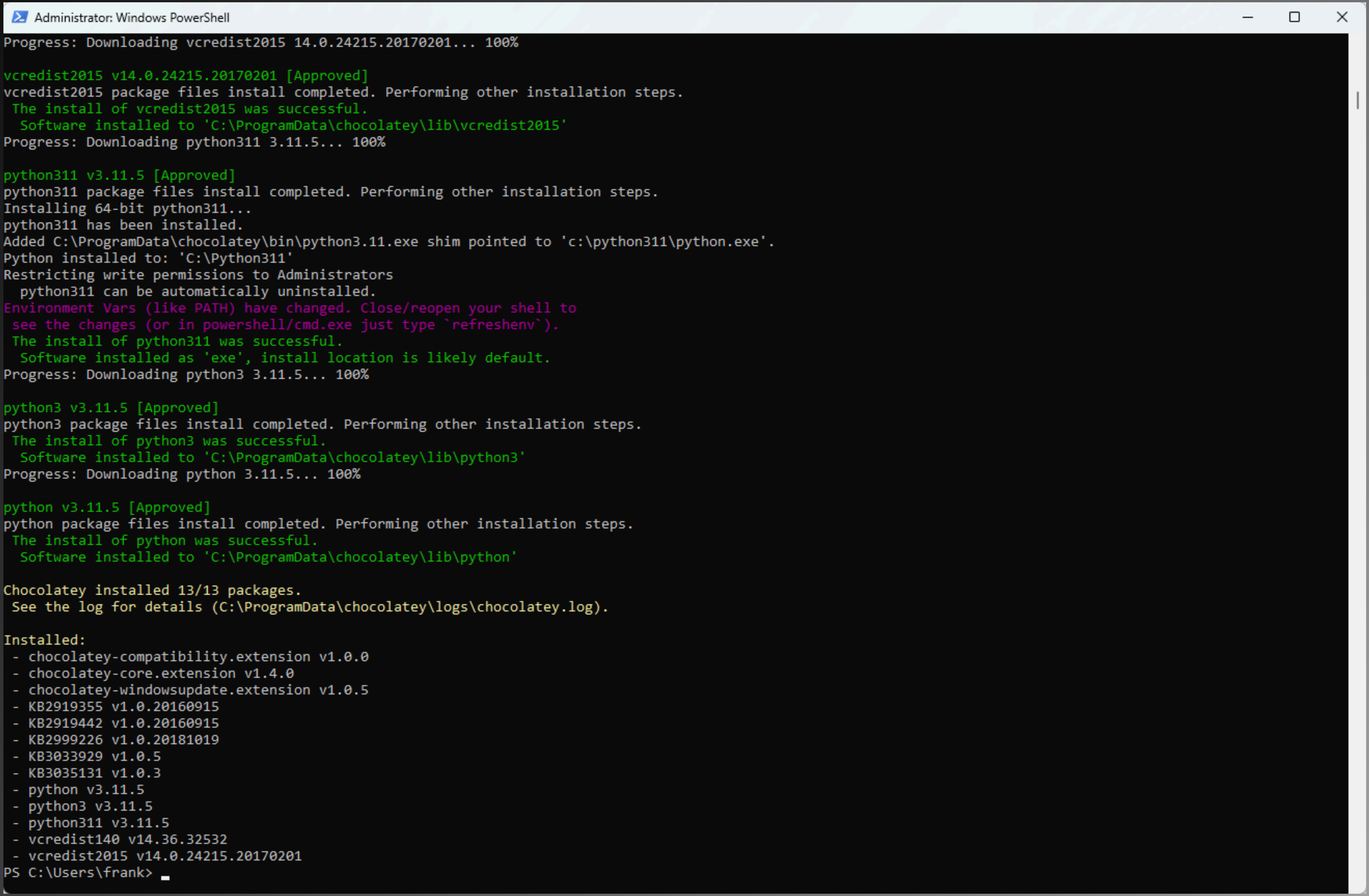
Install Python - Windows



Administrator: Windows PowerShell

```
1 packages installed.
PS C:\Users\frank> dir /


    Directory: C:\


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          5/6/2022  10:24 PM                PerfLogs
d-r---          9/2/2023   3:43 PM                Program Files
d-r---          5/6/2022  11:10 PM                Program Files (x86)
d-r---         3/23/2023   3:30 PM                Users
d-----          9/2/2023   3:52 PM                Windows


PS C:\Users\frank> choco install python
Chocolatey v2.2.2
Installing the following packages:
python
By installing, you accept licenses for the packages.
Progress: Downloading chocolatey-compatibility.extension 1.0.0... 100%

chocolatey-compatibility.extension v1.0.0 [Approved]
chocolatey-compatibility.extension package files install completed. Performing other installation steps.
 Installed/updated chocolatey-compatibility extensions.
 The install of chocolatey-compatibility.extension was successful.
  Software installed to 'C:\ProgramData\chocolatey\extensions\chocolatey-compatibility'
Progress: Downloading chocolatey-core.extension 1.4.0... 100%

chocolatey-core.extension v1.4.0 [Approved]
chocolatey-core.extension package files install completed. Performing other installation steps.
 Installed/updated chocolatey-core extensions.
 The install of chocolatey-core.extension was successful.
  Software installed to 'C:\ProgramData\chocolatey\extensions\chocolatey-core'
Progress: Downloading chocolatey-windowsupdate.extension 1.0.5... 100%

chocolatey-windowsupdate.extension v1.0.5 [Approved]
chocolatey-windowsupdate.extension package files install completed. Performing other installation steps.
 Installed/updated chocolatey-windowsupdate extensions.
 The install of chocolatey-windowsupdate.extension was successful.
  Software installed to 'C:\ProgramData\chocolatey\extensions\chocolatey-windowsupdate'
Progress: Downloading KB2919442 1.0.20160915... 100%

KB2919442 v1.0.20160915 [Approved]
KB2919442 package files install completed. Performing other installation steps.
The package KB2919442 wants to run 'ChocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N]o/[P]rint): _
```

Install Python - Windows



```
Progress: Downloading vcredist2015 14.0.24215.20170201... 100%

vcredist2015 v14.0.24215.20170201 [Approved]
vcredist2015 package files install completed. Performing other installation steps.
 The install of vcredist2015 was successful.
  Software installed to 'C:\ProgramData\chocolatey\lib\vcredist2015'
Progress: Downloading python311 3.11.5... 100%

python311 v3.11.5 [Approved]
python311 package files install completed. Performing other installation steps.
Installing 64-bit python311...
python311 has been installed.
Added C:\ProgramData\chocolatey\bin\python3.11.exe shim pointed to 'c:\python311\python.exe'.
Python installed to: 'C:\Python311'
Restricting write permissions to Administrators
  python311 can be automatically uninstalled.
Environment Vars (like PATH) have changed. Close/reopen your shell to
 see the changes (or in powershell/cmd.exe just type `refreshenv`).
 The install of python311 was successful.
  Software installed as 'exe', install location is likely default.
Progress: Downloading python3 3.11.5... 100%

python3 v3.11.5 [Approved]
python3 package files install completed. Performing other installation steps.
 The install of python3 was successful.
  Software installed to 'C:\ProgramData\chocolatey\lib\python3'
Progress: Downloading python 3.11.5... 100%

python v3.11.5 [Approved]
python package files install completed. Performing other installation steps.
 The install of python was successful.
  Software installed to 'C:\ProgramData\chocolatey\lib\python'

Chocolatey installed 13/13 packages.
 See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).

Installed:
 - chocolatey-compatibility.extension v1.0.0
 - chocolatey-core.extension v1.4.0
 - chocolatey-windowsupdate.extension v1.0.5
 - KB2919355 v1.0.20160915
 - KB2919442 v1.0.20160915
 - KB2999226 v1.0.20181019
 - KB3033929 v1.0.5
 - KB3035131 v1.0.3
 - python v3.11.5
 - python3 v3.11.5
 - python311 v3.11.5
 - vcredist140 v14.36.32532
 - vcredist2015 v14.0.24215.20170201
PS C:\Users\frank> _
```

Install Python - Windows

Microsoft Store installs Python in

```
C:\Python311\
```

Python modules (e.g., seen using **`pip list -v`**) are located in

```
C:\Python311\Lib\site-packages\
```

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

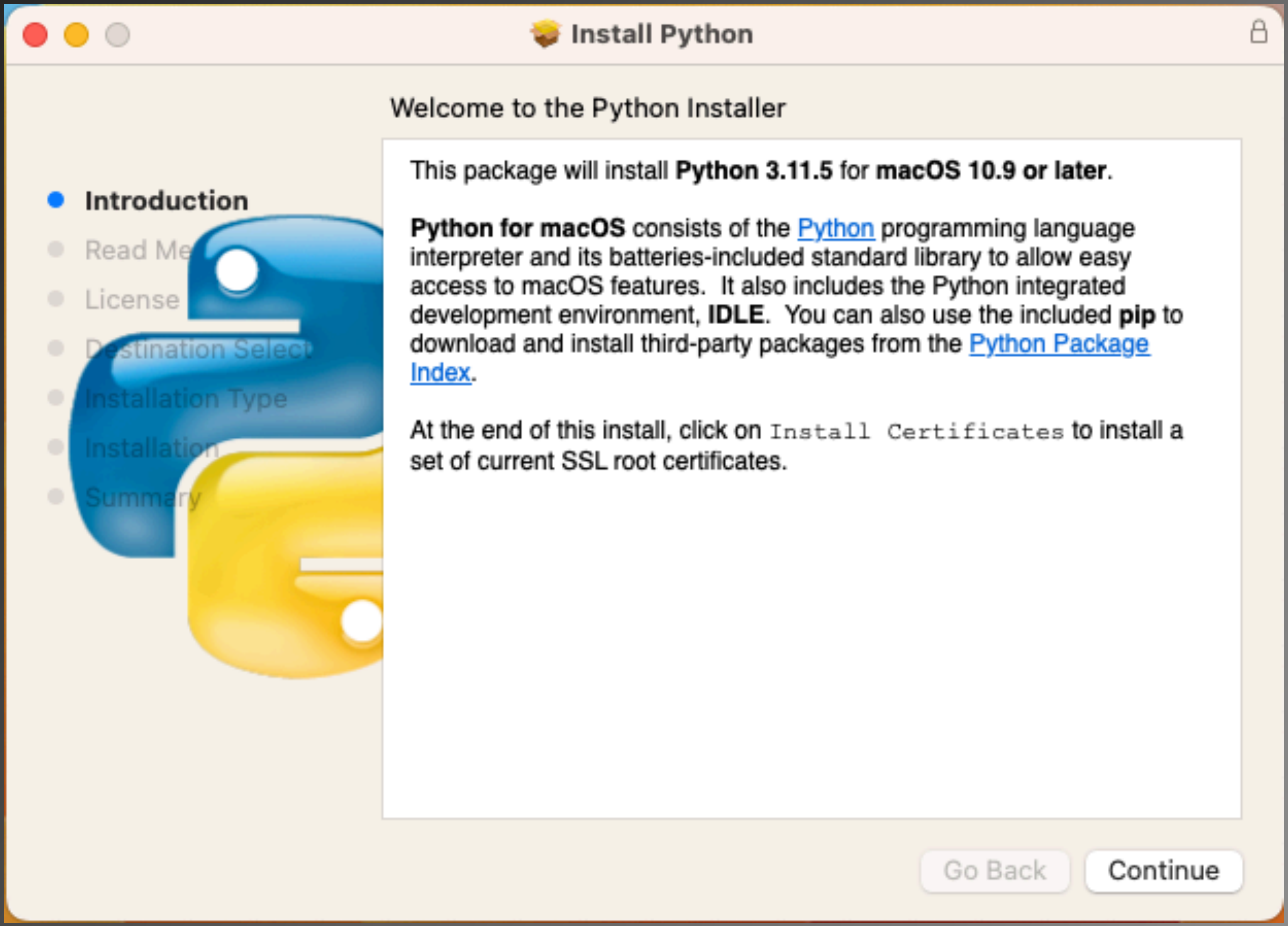
mac
OS

Installing Python

for macOS

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

Install Python - macOS



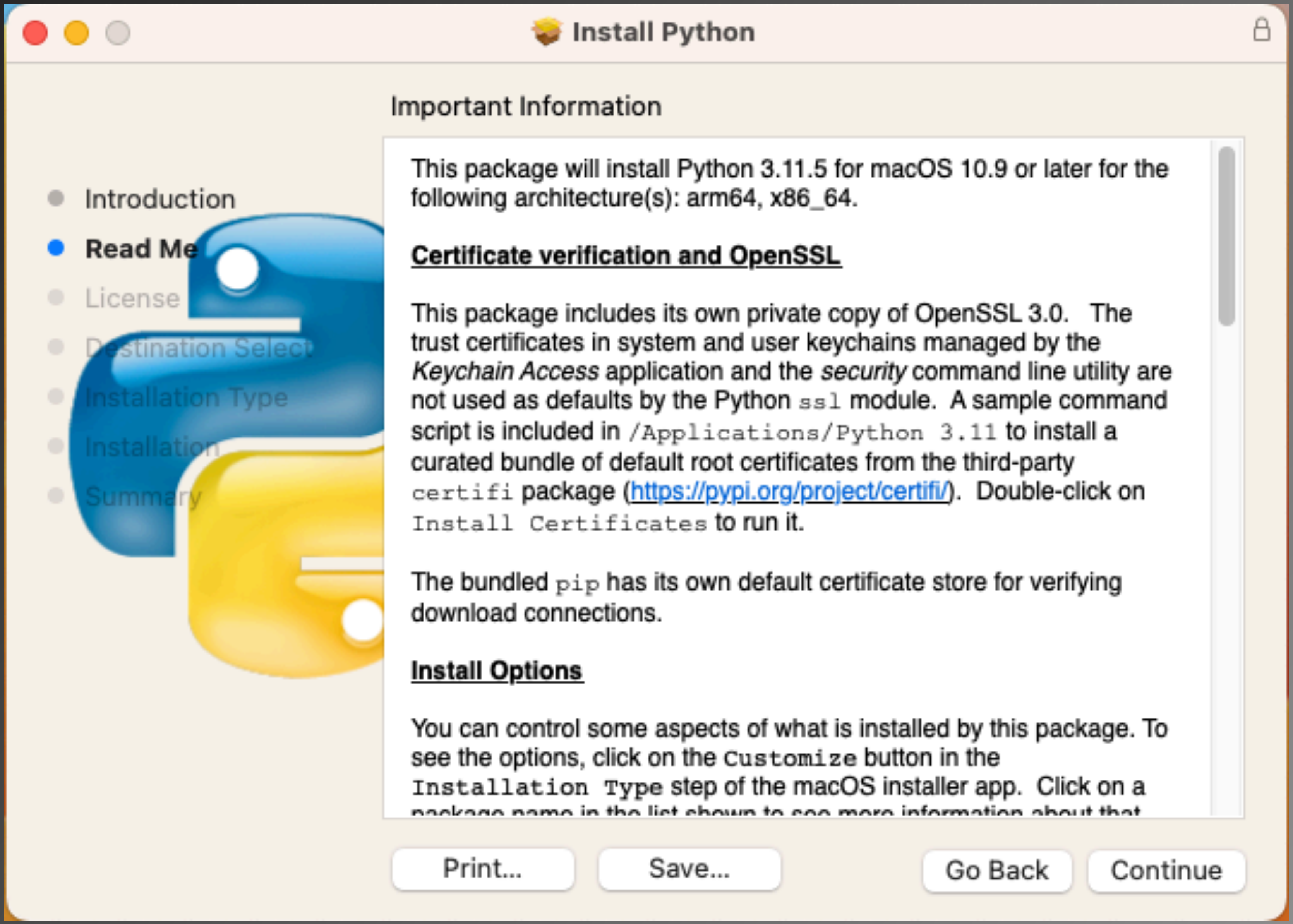
Welcome to the Python Installer

This package will install **Python 3.11.5** for **macOS 10.9 or later**.

Python for macOS consists of the Python programming language interpreter and its batteries-included standard library to allow easy access to macOS features. It also includes the Python integrated development environment, **IDLE**. You can also use the included **pip** to download and install third-party packages from the Python Package Index.

At the end of this install, click on `Install Certificates` to install a set of current SSL root certificates.

Install Python - macOS



Important Information

This package will install Python 3.11.5 for macOS 10.9 or later for the following architecture(s): arm64, x86_64.

Certificate verification and OpenSSL

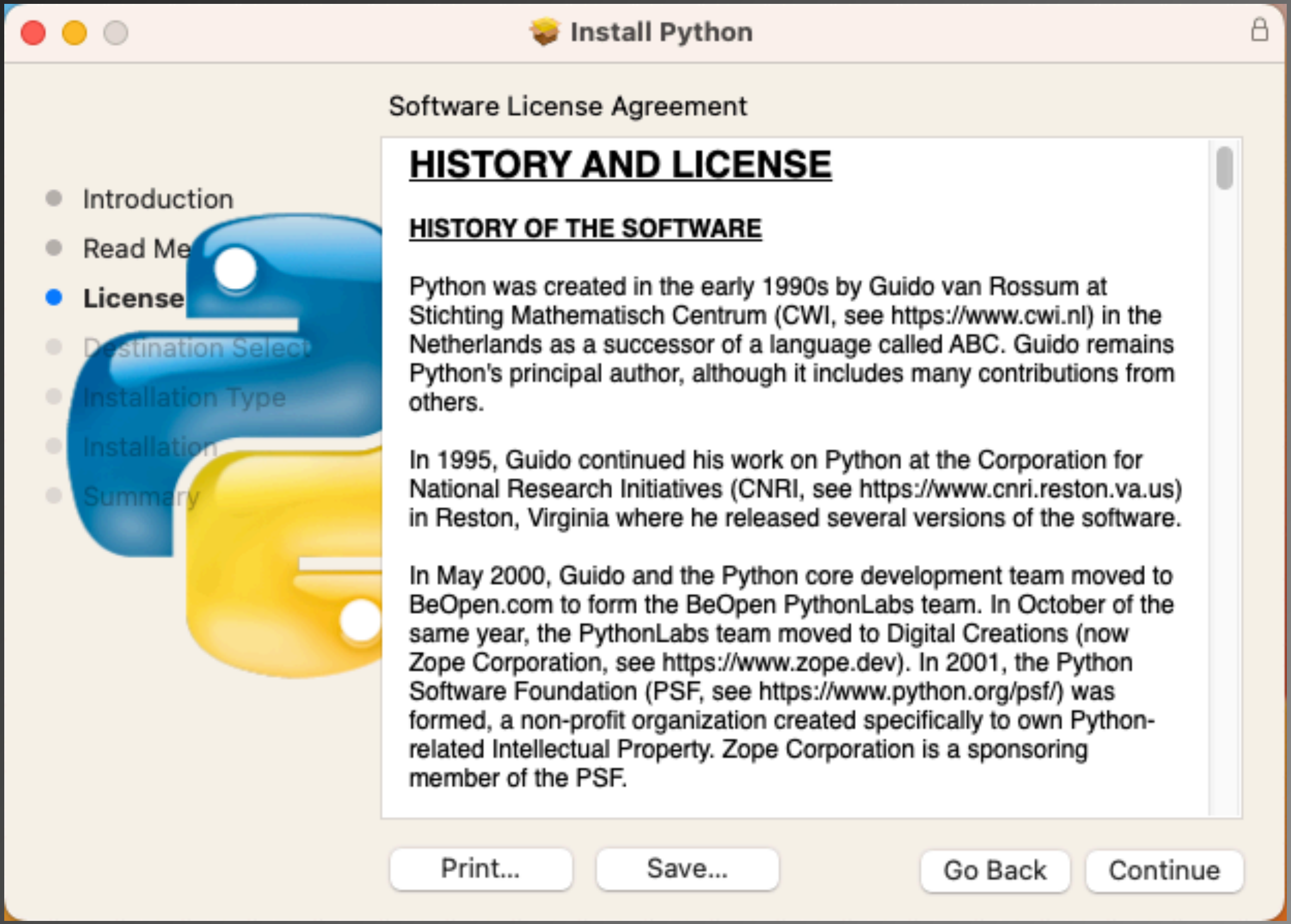
This package includes its own private copy of OpenSSL 3.0. The trust certificates in system and user keychains managed by the *Keychain Access* application and the *security* command line utility are not used as defaults by the Python `ssl` module. A sample command script is included in `/Applications/Python 3.11` to install a curated bundle of default root certificates from the third-party `certifi` package (https://pypi.org/project/certifi/). Double-click on `Install Certificates` to run it.

The bundled `pip` has its own default certificate store for verifying download connections.

Install Options

You can control some aspects of what is installed by this package. To see the options, click on the `Customize` button in the `Installation Type` step of the macOS installer app. Click on a

Install Python - macOS



Software License Agreement

HISTORY AND LICENSE

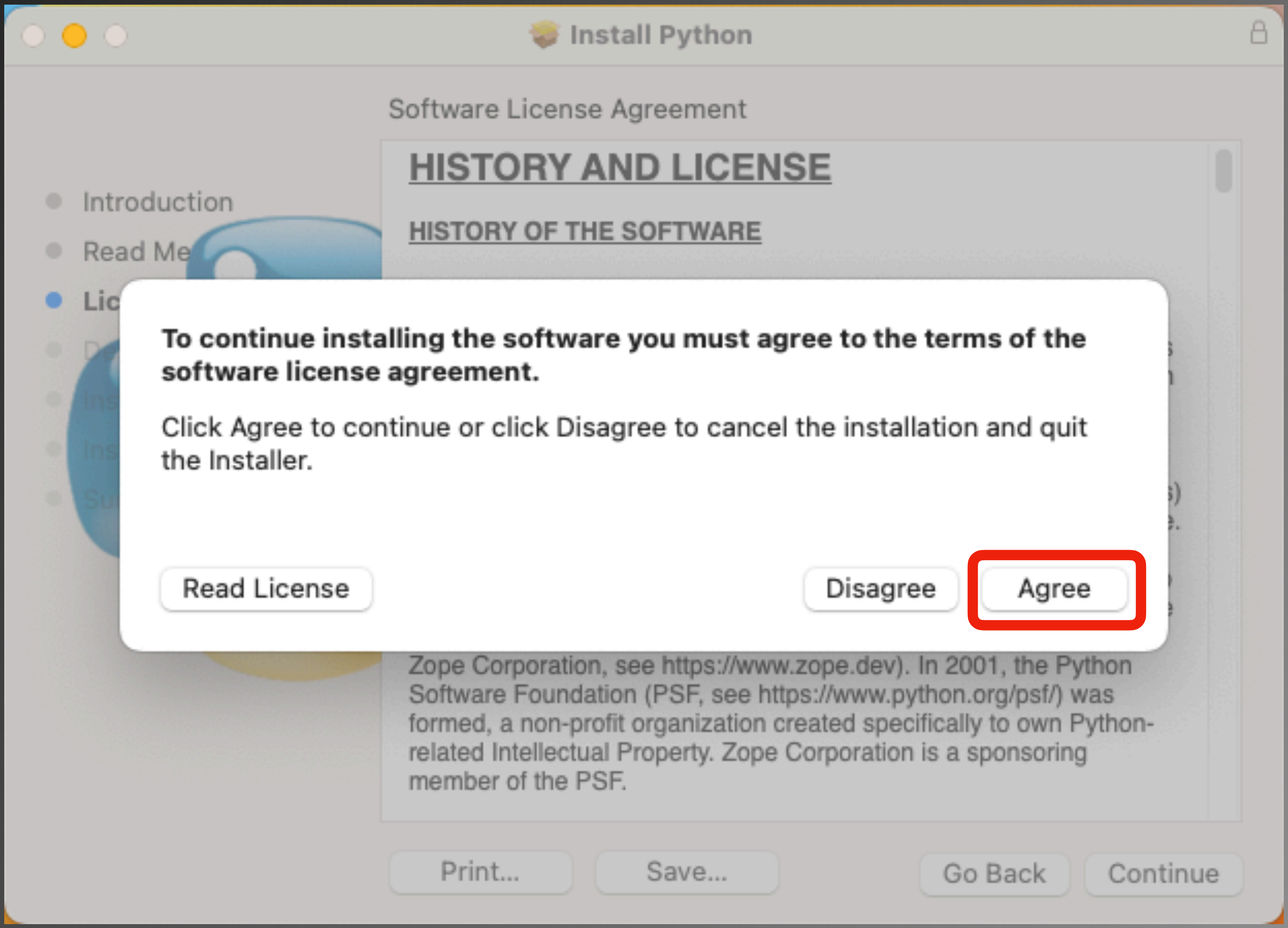
HISTORY OF THE SOFTWARE

Python was created in the early 1990s by Guido van Rossum at Stichting Mathematisch Centrum (CWI, see https://www.cwi.nl) in the Netherlands as a successor of a language called ABC. Guido remains Python's principal author, although it includes many contributions from others.

In 1995, Guido continued his work on Python at the Corporation for National Research Initiatives (CNRI, see https://www.cnri.reston.va.us) in Reston, Virginia where he released several versions of the software.

In May 2000, Guido and the Python core development team moved to BeOpen.com to form the BeOpen PythonLabs team. In October of the same year, the PythonLabs team moved to Digital Creations (now Zope Corporation, see https://www.zope.dev). In 2001, the Python Software Foundation (PSF, see https://www.python.org/psf/) was formed, a non-profit organization created specifically to own Python-related Intellectual Property. Zope Corporation is a sponsoring member of the PSF.

Install Python - macOS



Software License Agreement

HISTORY AND LICENSE

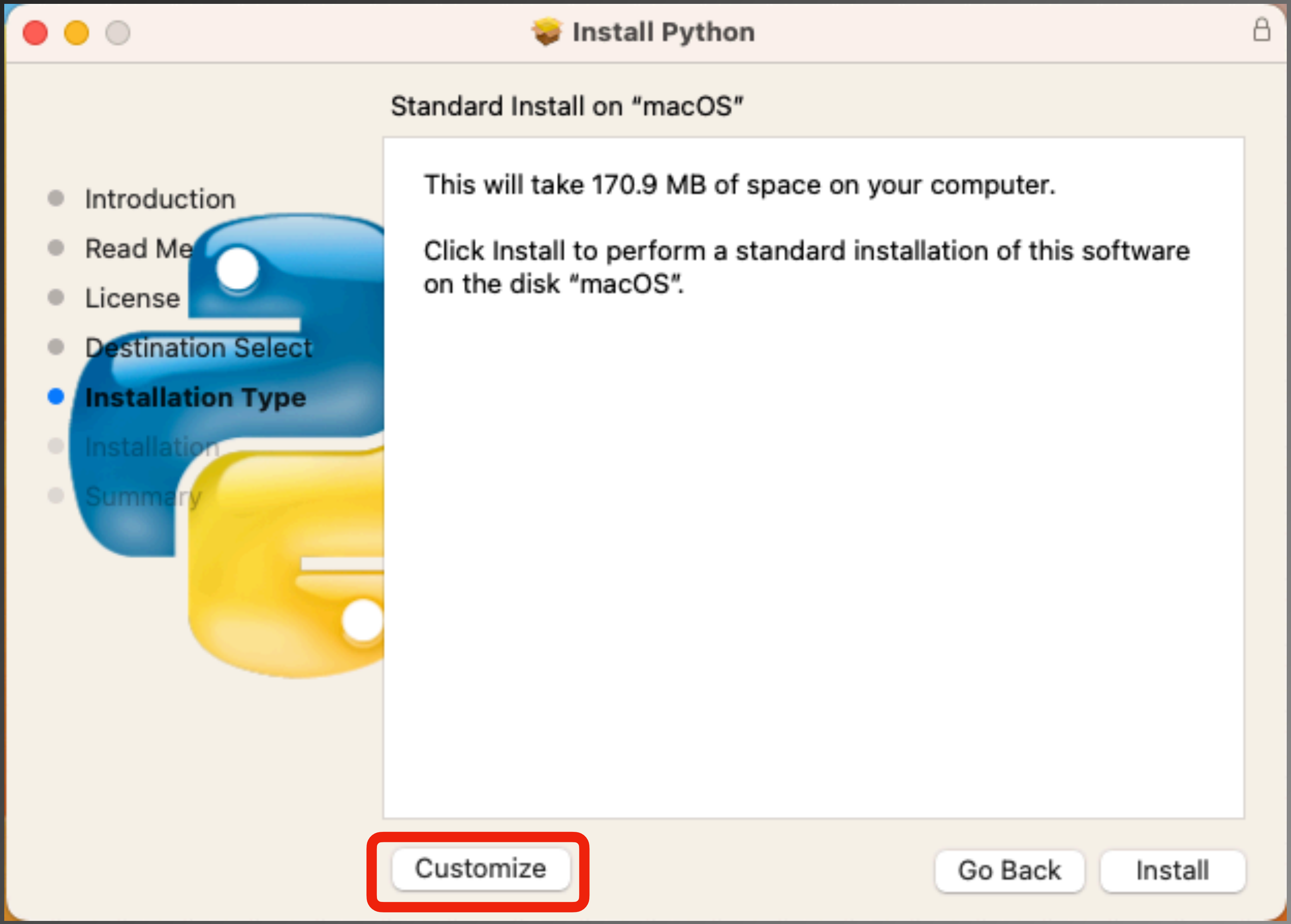
HISTORY OF THE SOFTWARE

To continue installing the software you must agree to the terms of the software license agreement.

Click Agree to continue or click Disagree to cancel the installation and quit the Installer.

Zope Corporation, see https://www.zope.dev). In 2001, the Python Software Foundation (PSF, see https://www.python.org/psf/) was formed, a non-profit organization created specifically to own Python-related Intellectual Property. Zope Corporation is a sponsoring member of the PSF.

Install Python - macOS

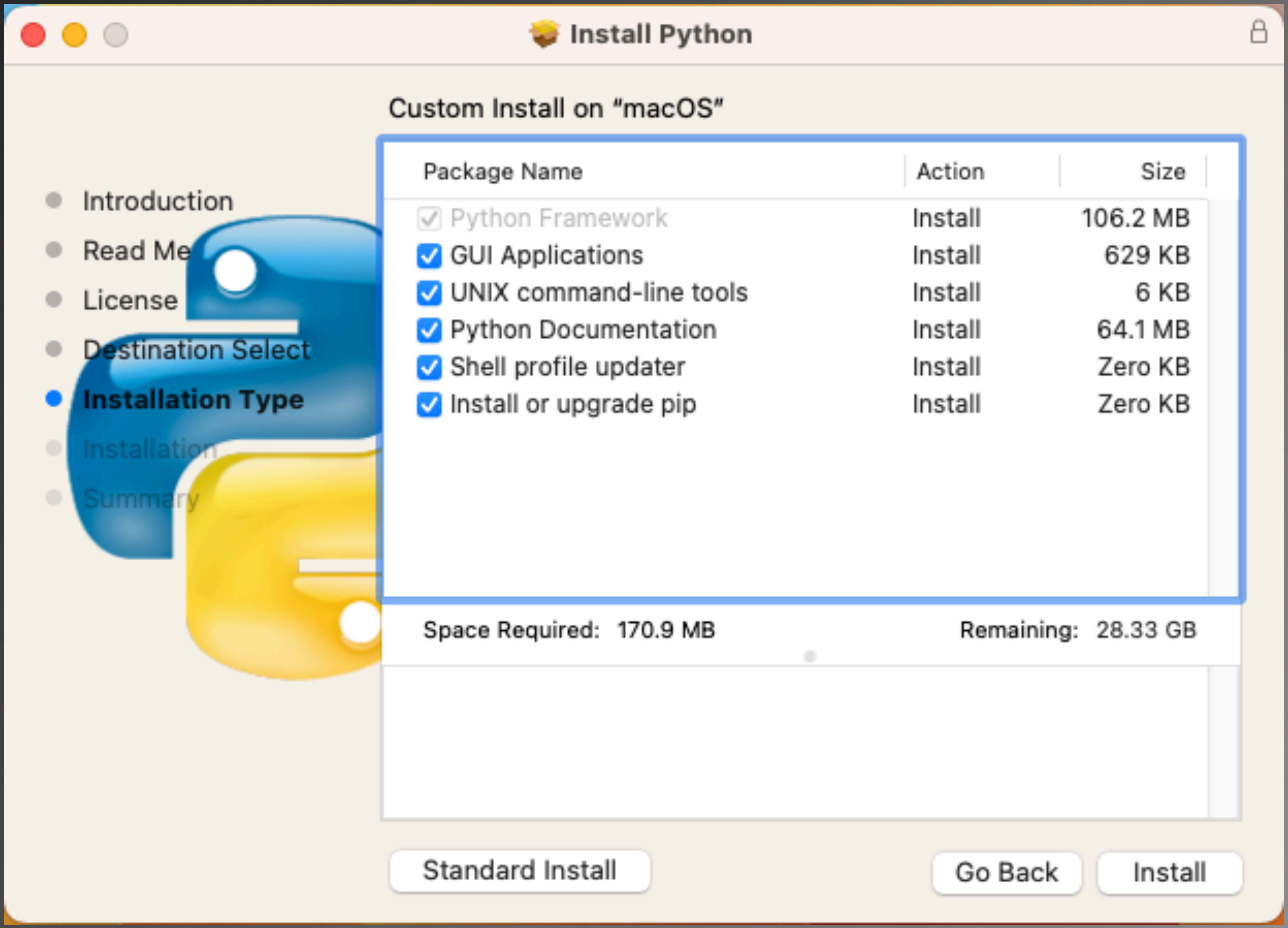


Standard Install on "macOS"

This will take 170.9 MB of space on your computer.

Click Install to perform a standard installation of this software on the disk "macOS".

Install Python - macOS



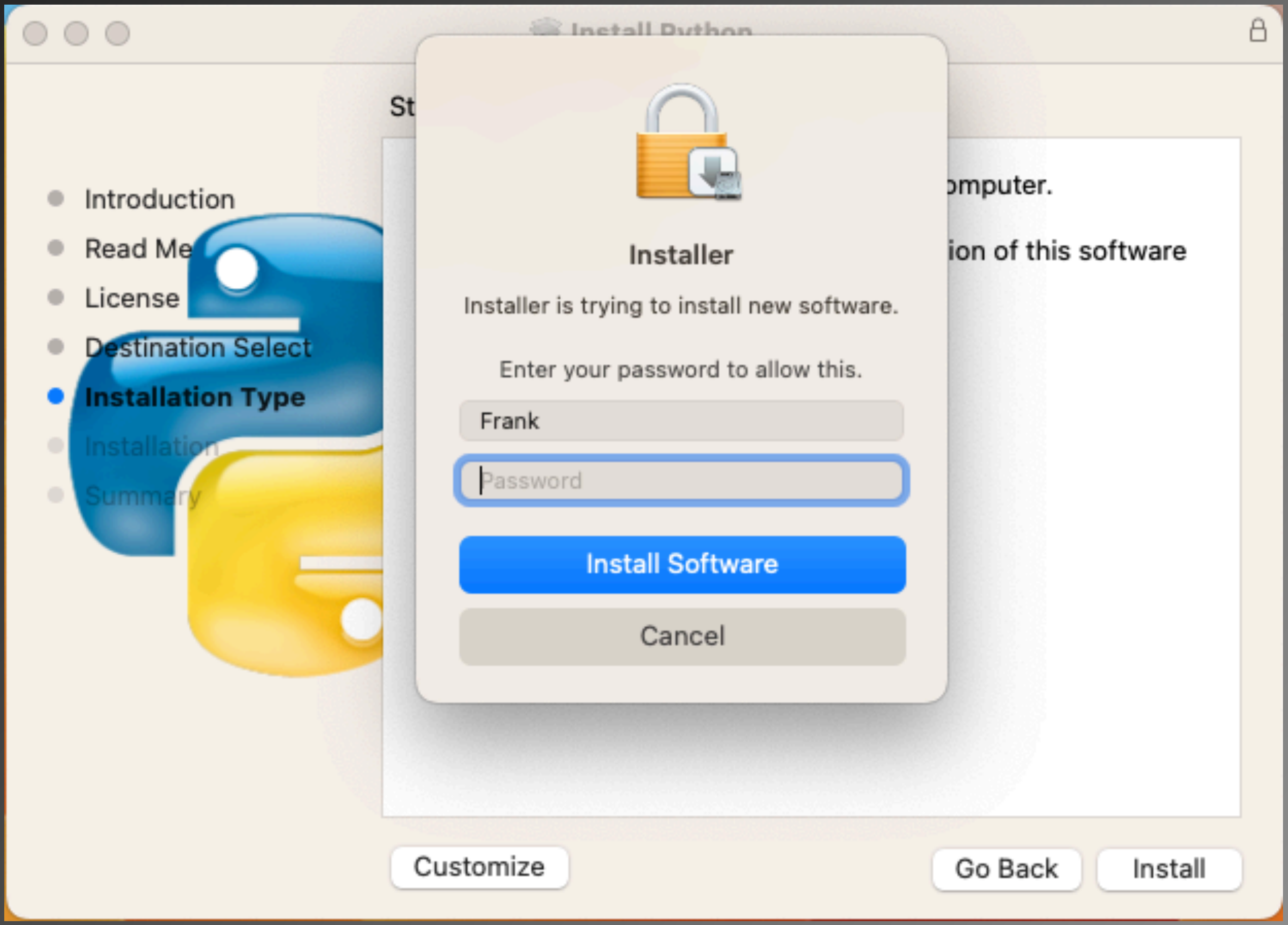
Install Python

Custom Install on "macOS"

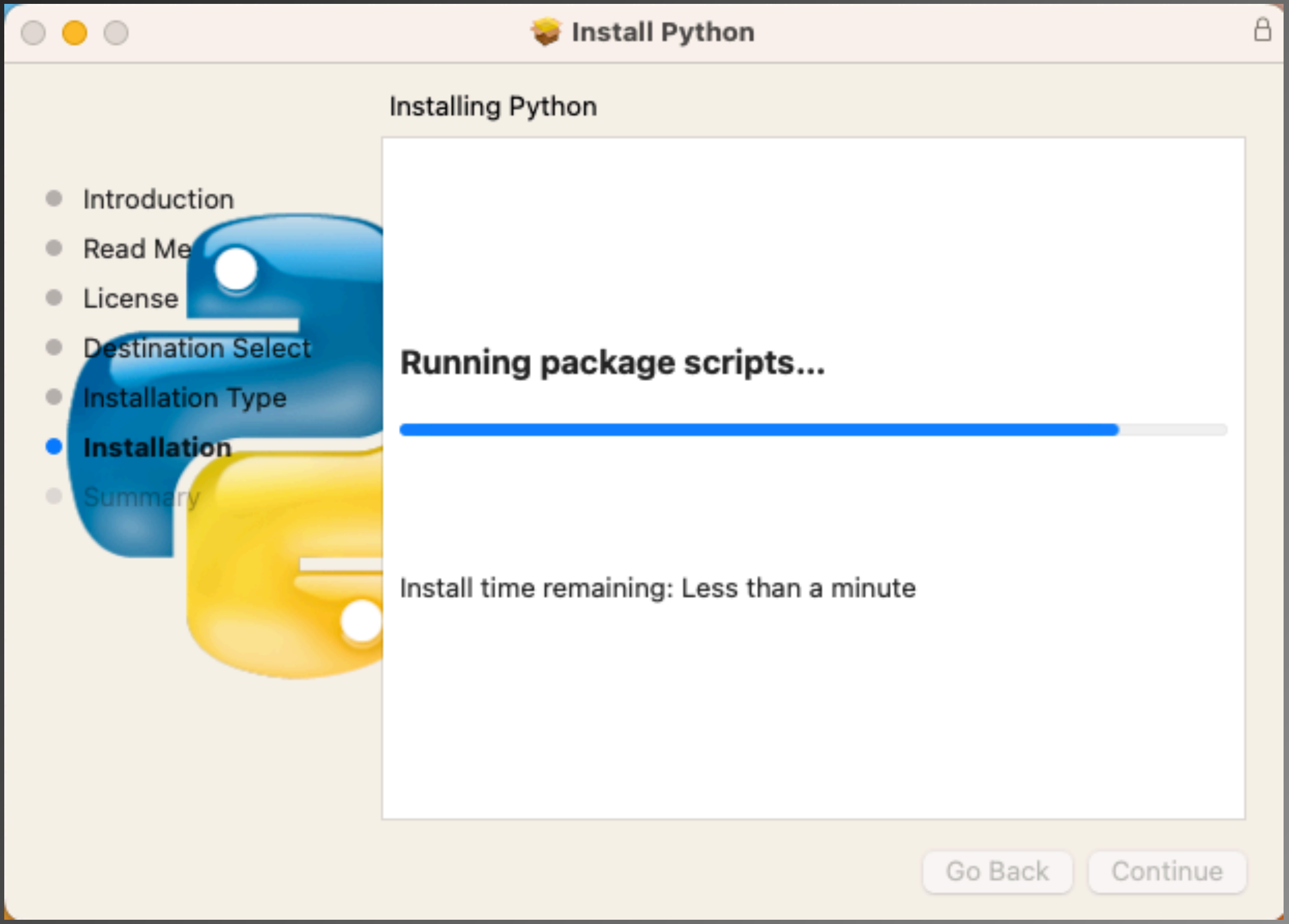
Package Name	Action	Size
✓ Python Framework	Install	106.2 MB
✓ GUI Applications	Install	629 KB
✓ UNIX command-line tools	Install	6 KB
✓ Python Documentation	Install	64.1 MB
✓ Shell profile updater	Install	Zero KB
✓ Install or upgrade pip	Install	Zero KB

Space Required: 170.9 MB Remaining: 28.33 GB

Install Python - macOS



Install Python - macOS

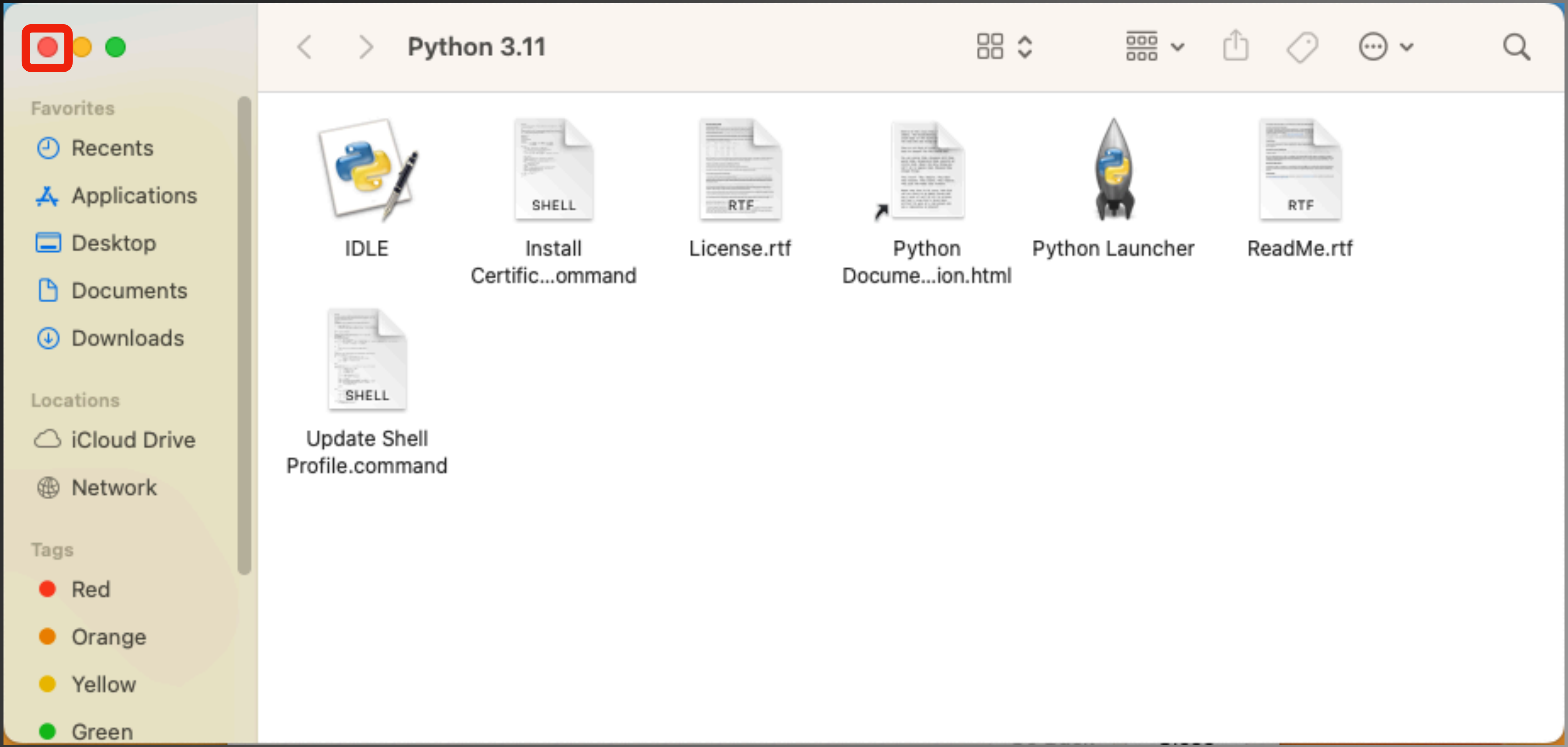

Install Python

Installing Python

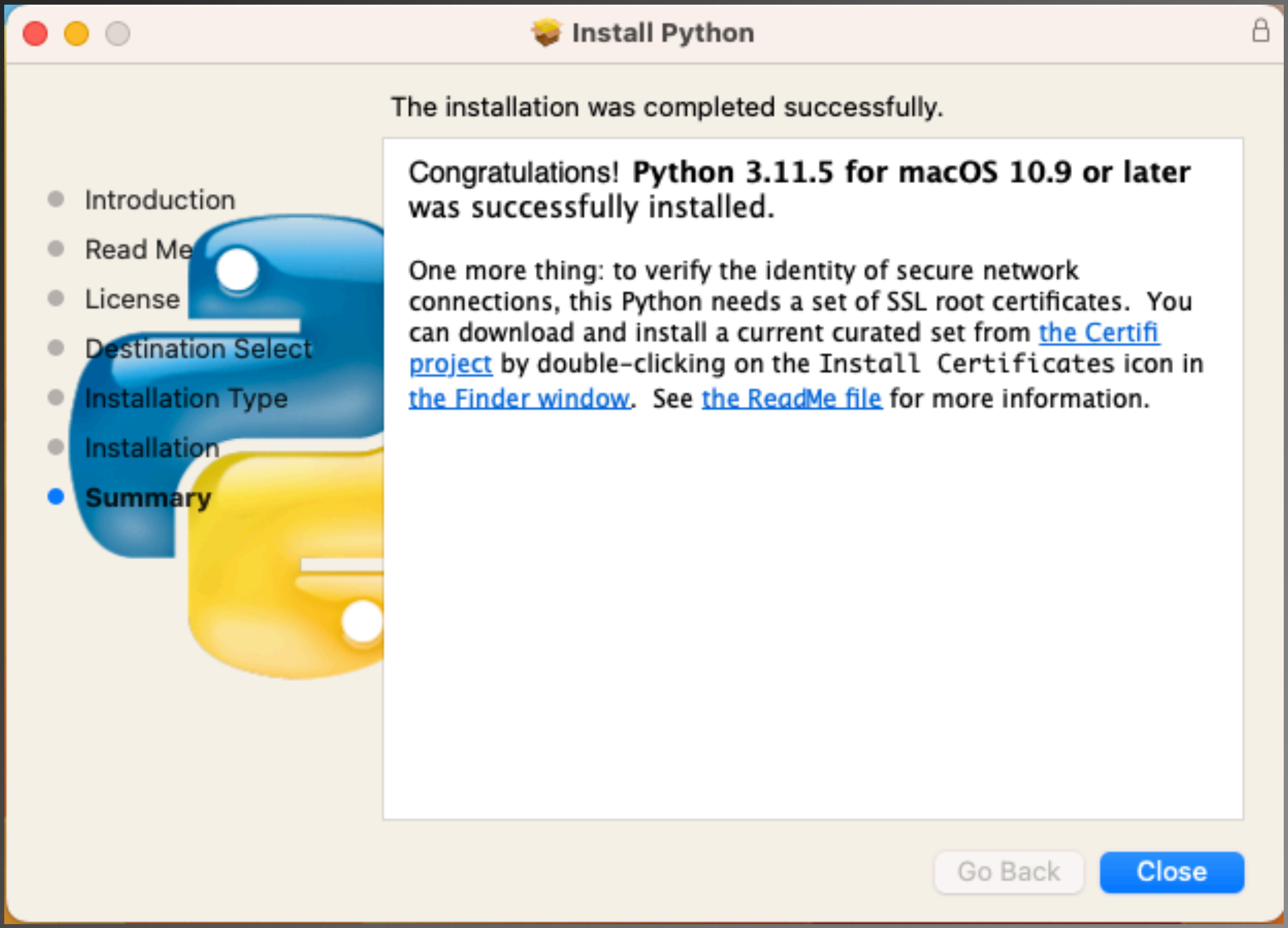
Running package scripts...

Install time remaining: Less than a minute

Install Python - macOS



Install Python - macOS

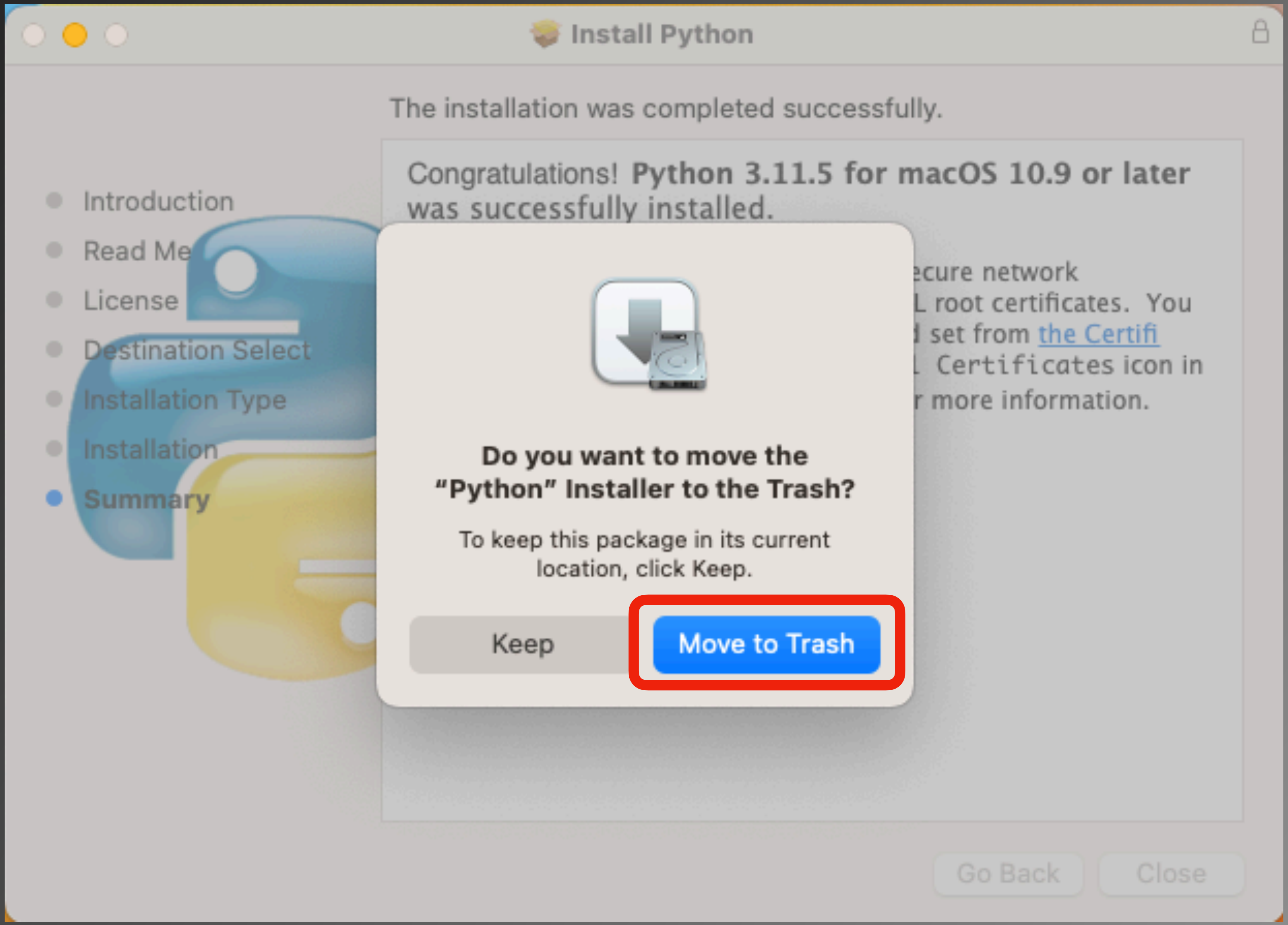


The installation was completed successfully.

Congratulations! **Python 3.11.5 for macOS 10.9 or later** was successfully installed.

One more thing: to verify the identity of secure network connections, this Python needs a set of SSL root certificates. You can download and install a current curated set from the Certifi project by double-clicking on the `Install Certificates` icon in the Finder window. See the ReadMe file for more information.

Install Python - macOS



The installation was completed successfully.

Congratulations! **Python 3.11.5 for macOS 10.9 or later was successfully installed.**

ecure network
L root certificates. You
l set from the Certifi
L Certificates icon in
r more information.

Do you want to move the "Python" Installer to the Trash?

To keep this package in its current location, click Keep.

Install Python - macOS

Python.org macOS Installer installs Python in

```
/Library/Frameworks/Python.framework/Versions/3.11/
```

Python modules (e.g., seen using **`pip3 list -v`**) are located in

```
/Library/Frameworks/Python.framework/Versions/3.11/lib/
python3.11/site-packages/
```

Install Python - macOS

Option #2: Homebrew

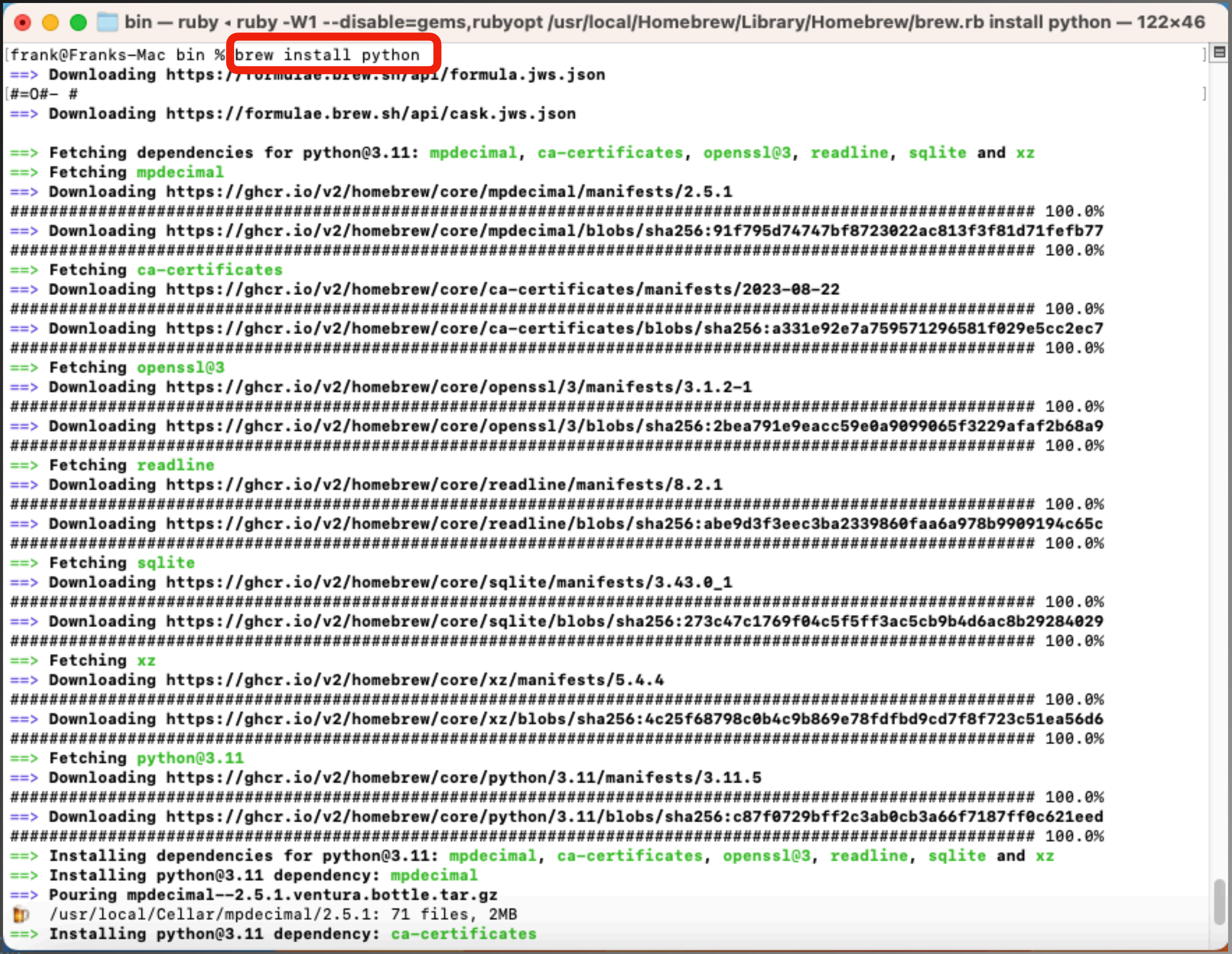
The Missing Package Manager for macOS (or Linux)

https://brew.sh/

Simply having Homebrew installed provides you with a version of Python3 (it comes with the XCode Command Line Tools that Homebrew installs). However, it is not the latest. To update to the current version, simply open Terminal and enter

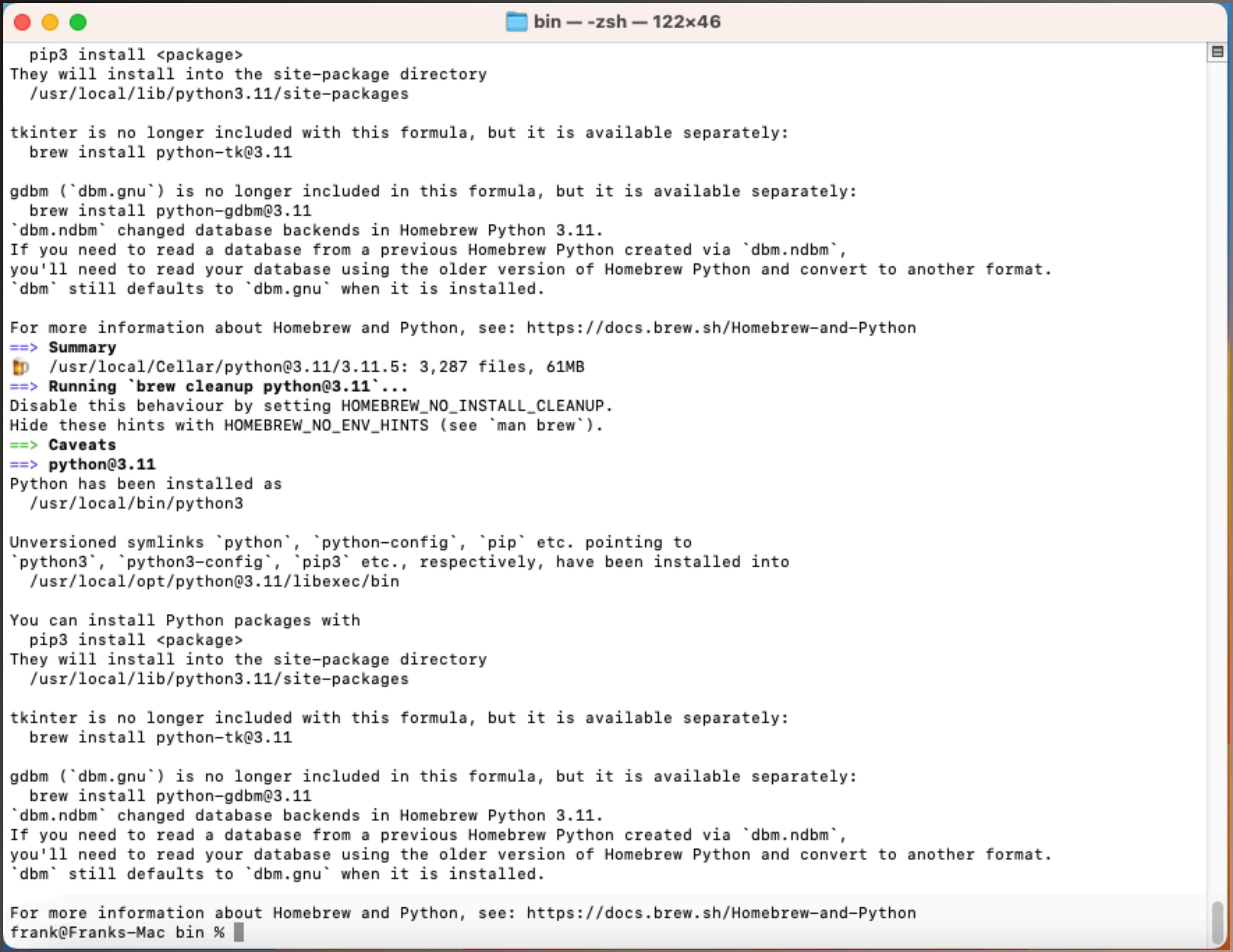
```
brew install python
```

Install Python - macOS



```
bin — ruby ◂ ruby -W1 --disable=gems,rubyopt /usr/local/Homebrew/Library/Homebrew/brew.rb install python — 122×46
frank@Franks-Mac bin % brew install python
==> Downloading https://formulae.brew.sh/api/formula.jws.json
#=O#- #
==> Downloading https://formulae.brew.sh/api/cask.jws.json

==> Fetching dependencies for python@3.11: mpdecimal, ca-certificates, openssl@3, readline, sqlite and xz
==> Fetching mpdecimal
==> Downloading https://ghcr.io/v2/homebrew/core/mpdecimal/manifests/2.5.1
######################################################################################################### 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/mpdecimal/blobs/sha256:91f795d74747bf8723022ac813f3f81d71fefb77
######################################################################################################### 100.0%
==> Fetching ca-certificates
==> Downloading https://ghcr.io/v2/homebrew/core/ca-certificates/manifests/2023-08-22
######################################################################################################### 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/ca-certificates/blobs/sha256:a331e92e7a759571296581f029e5cc2ec7
######################################################################################################### 100.0%
==> Fetching openssl@3
==> Downloading https://ghcr.io/v2/homebrew/core/openssl/3/manifests/3.1.2-1
######################################################################################################### 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/openssl/3/blobs/sha256:2bea791e9eacc59e0a9099065f3229afaf2b68a9
######################################################################################################### 100.0%
==> Fetching readline
==> Downloading https://ghcr.io/v2/homebrew/core/readline/manifests/8.2.1
######################################################################################################### 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/readline/blobs/sha256:abe9d3f3eec3ba2339860faa6a978b9909194c65c
######################################################################################################### 100.0%
==> Fetching sqlite
==> Downloading https://ghcr.io/v2/homebrew/core/sqlite/manifests/3.43.0_1
######################################################################################################### 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/sqlite/blobs/sha256:273c47c1769f04c5f5ff3ac5cb9b4d6ac8b29284029
######################################################################################################### 100.0%
==> Fetching xz
==> Downloading https://ghcr.io/v2/homebrew/core/xz/manifests/5.4.4
######################################################################################################### 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/xz/blobs/sha256:4c25f68798c0b4c9b869e78fdfbd9cd7f8f723c51ea56d6
######################################################################################################### 100.0%
==> Fetching python@3.11
==> Downloading https://ghcr.io/v2/homebrew/core/python/3.11/manifests/3.11.5
######################################################################################################### 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/python/3.11/blobs/sha256:c87f0729bff2c3ab0cb3a66f7187ff0c621eed
######################################################################################################### 100.0%
==> Installing dependencies for python@3.11: mpdecimal, ca-certificates, openssl@3, readline, sqlite and xz
==> Installing python@3.11 dependency: mpdecimal
==> Pouring mpdecimal--2.5.1.ventura.bottle.tar.gz
🍺  /usr/local/Cellar/mpdecimal/2.5.1: 71 files, 2MB
==> Installing python@3.11 dependency: ca-certificates
```

Install Python - macOS



bin — -zsh — 122×46

```
  pip3 install <package>
They will install into the site-package directory
  /usr/local/lib/python3.11/site-packages

tkinter is no longer included with this formula, but it is available separately:
  brew install python-tk@3.11

gdbm (`dbm.gnu`) is no longer included in this formula, but it is available separately:
  brew install python-gdbm@3.11
`dbm.ndbm` changed database backends in Homebrew Python 3.11.
If you need to read a database from a previous Homebrew Python created via `dbm.ndbm`,
you'll need to read your database using the older version of Homebrew Python and convert to another format.
`dbm` still defaults to `dbm.gnu` when it is installed.

For more information about Homebrew and Python, see: https://docs.brew.sh/Homebrew-and-Python
==> Summary
🍺  /usr/local/Cellar/python@3.11/3.11.5: 3,287 files, 61MB
==> Running `brew cleanup python@3.11`...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).
==> Caveats
==> python@3.11
Python has been installed as
  /usr/local/bin/python3

Unversioned symlinks `python`, `python-config`, `pip` etc. pointing to
`python3`, `python3-config`, `pip3` etc., respectively, have been installed into
  /usr/local/opt/python@3.11/libexec/bin

You can install Python packages with
  pip3 install <package>
They will install into the site-package directory
  /usr/local/lib/python3.11/site-packages

tkinter is no longer included with this formula, but it is available separately:
  brew install python-tk@3.11

gdbm (`dbm.gnu`) is no longer included in this formula, but it is available separately:
  brew install python-gdbm@3.11
`dbm.ndbm` changed database backends in Homebrew Python 3.11.
If you need to read a database from a previous Homebrew Python created via `dbm.ndbm`,
you'll need to read your database using the older version of Homebrew Python and convert to another format.
`dbm` still defaults to `dbm.gnu` when it is installed.

For more information about Homebrew and Python, see: https://docs.brew.sh/Homebrew-and-Python
frank@Franks-Mac bin %
```

Install Python - macOS

Homebrew macOS Installer installs Python in

```
/usr/local/bin/
```

Python modules (e.g., seen using **pip3 list -v**) are located in

```
/usr/local/lib/python3.11/site-packages/
```

Install Python - macOS

Option #3: MacPorts

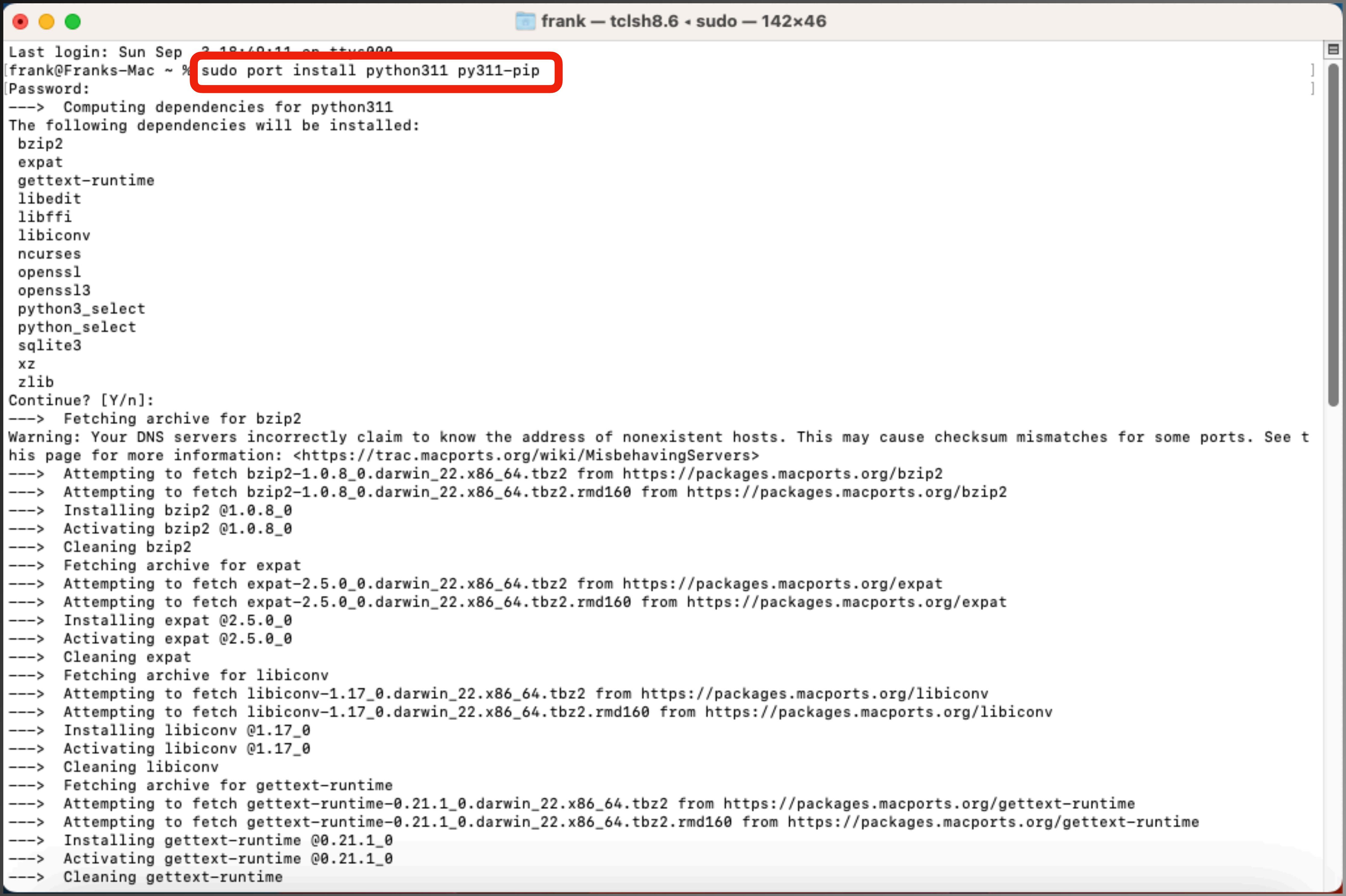
An open-source community initiative to design an easy-to-use system for compiling, installing, and upgrading either command-line, X11 or Aqua based open-source software on the Mac operating system

https://www.macports.org/

To install Python, simply open Terminal and enter

```
sudo port install python311 py311-pip
```

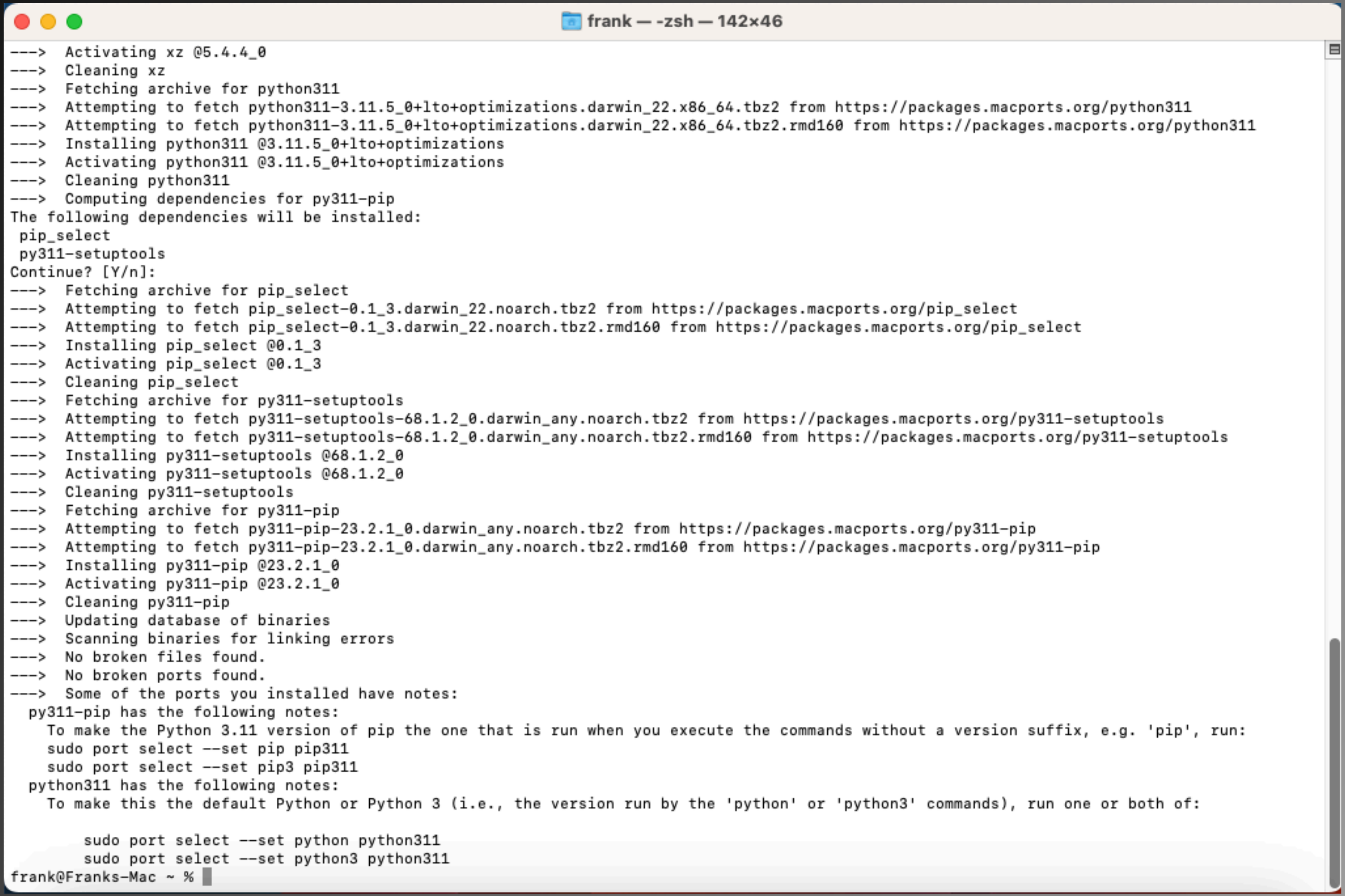
Install Python - macOS



```
Last login: Sun Sep  3 18:49:11 on ttys000
frank@Franks-Mac ~ % sudo port install python311 py311-pip
Password:
--->  Computing dependencies for python311
The following dependencies will be installed:
 bzip2
 expat
 gettext-runtime
 libedit
 libffi
 libiconv
 ncurses
 openssl
 openssl3
 python3_select
 python_select
 sqlite3
 xz
 zlib
Continue? [Y/n]:
--->  Fetching archive for bzip2
Warning: Your DNS servers incorrectly claim to know the address of nonexistent hosts. This may cause checksum mismatches for some ports. See t
his page for more information: <https://trac.macports.org/wiki/MisbehavingServers>
--->  Attempting to fetch bzip2-1.0.8_0.darwin_22.x86_64.tbz2 from https://packages.macports.org/bzip2
--->  Attempting to fetch bzip2-1.0.8_0.darwin_22.x86_64.tbz2.rmd160 from https://packages.macports.org/bzip2
--->  Installing bzip2 @1.0.8_0
--->  Activating bzip2 @1.0.8_0
--->  Cleaning bzip2
--->  Fetching archive for expat
--->  Attempting to fetch expat-2.5.0_0.darwin_22.x86_64.tbz2 from https://packages.macports.org/expat
--->  Attempting to fetch expat-2.5.0_0.darwin_22.x86_64.tbz2.rmd160 from https://packages.macports.org/expat
--->  Installing expat @2.5.0_0
--->  Activating expat @2.5.0_0
--->  Cleaning expat
--->  Fetching archive for libiconv
--->  Attempting to fetch libiconv-1.17_0.darwin_22.x86_64.tbz2 from https://packages.macports.org/libiconv
--->  Attempting to fetch libiconv-1.17_0.darwin_22.x86_64.tbz2.rmd160 from https://packages.macports.org/libiconv
--->  Installing libiconv @1.17_0
--->  Activating libiconv @1.17_0
--->  Cleaning libiconv
--->  Fetching archive for gettext-runtime
--->  Attempting to fetch gettext-runtime-0.21.1_0.darwin_22.x86_64.tbz2 from https://packages.macports.org/gettext-runtime
--->  Attempting to fetch gettext-runtime-0.21.1_0.darwin_22.x86_64.tbz2.rmd160 from https://packages.macports.org/gettext-runtime
--->  Installing gettext-runtime @0.21.1_0
--->  Activating gettext-runtime @0.21.1_0
--->  Cleaning gettext-runtime
```

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Install Python - macOS



```
--->  Activating xz @5.4.4_0
--->  Cleaning xz
--->  Fetching archive for python311
--->  Attempting to fetch python311-3.11.5_0+lto+optimizations.darwin_22.x86_64.tbz2 from https://packages.macports.org/python311
--->  Attempting to fetch python311-3.11.5_0+lto+optimizations.darwin_22.x86_64.tbz2.rmd160 from https://packages.macports.org/python311
--->  Installing python311 @3.11.5_0+lto+optimizations
--->  Activating python311 @3.11.5_0+lto+optimizations
--->  Cleaning python311
--->  Computing dependencies for py311-pip
The following dependencies will be installed:
 pip_select
 py311-setuptools
Continue? [Y/n]:
--->  Fetching archive for pip_select
--->  Attempting to fetch pip_select-0.1_3.darwin_22.noarch.tbz2 from https://packages.macports.org/pip_select
--->  Attempting to fetch pip_select-0.1_3.darwin_22.noarch.tbz2.rmd160 from https://packages.macports.org/pip_select
--->  Installing pip_select @0.1_3
--->  Activating pip_select @0.1_3
--->  Cleaning pip_select
--->  Fetching archive for py311-setuptools
--->  Attempting to fetch py311-setuptools-68.1.2_0.darwin_any.noarch.tbz2 from https://packages.macports.org/py311-setuptools
--->  Attempting to fetch py311-setuptools-68.1.2_0.darwin_any.noarch.tbz2.rmd160 from https://packages.macports.org/py311-setuptools
--->  Installing py311-setuptools @68.1.2_0
--->  Activating py311-setuptools @68.1.2_0
--->  Cleaning py311-setuptools
--->  Fetching archive for py311-pip
--->  Attempting to fetch py311-pip-23.2.1_0.darwin_any.noarch.tbz2 from https://packages.macports.org/py311-pip
--->  Attempting to fetch py311-pip-23.2.1_0.darwin_any.noarch.tbz2.rmd160 from https://packages.macports.org/py311-pip
--->  Installing py311-pip @23.2.1_0
--->  Activating py311-pip @23.2.1_0
--->  Cleaning py311-pip
--->  Updating database of binaries
--->  Scanning binaries for linking errors
--->  No broken files found.
--->  No broken ports found.
--->  Some of the ports you installed have notes:
  py311-pip has the following notes:
    To make the Python 3.11 version of pip the one that is run when you execute the commands without a version suffix, e.g. 'pip', run:
    sudo port select --set pip pip311
    sudo port select --set pip3 pip311
  python311 has the following notes:
    To make this the default Python or Python 3 (i.e., the version run by the 'python' or 'python3' commands), run one or both of:

        sudo port select --set python python311
        sudo port select --set python3 python311
frank@Franks-Mac ~ %
```

Install Python - macOS

MacPorts macOS Installer installs Python in

```
/opt/local/bin/
```

Python modules (e.g., seen using **`pip3 list -v`**) are located in

```
/opt/local/Library/Frameworks/Python.framework/Versions/
3.11/lib/python3.11/site-packages/
```

Installing Python

for Linux

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

Install Python - Linux

RHEL/CENTOS/Rocky/Alma Linux

```
rpm/yum/dnf install python3
```

Ubuntu/Debian Linux

```
apt install python3
```

WSL

Python Basics

Python REPL

REPL = Read, Evaluate, Print, and Loop

```
$ python3
Python 3.11.5 (v3.11.5:cce6ba91b3, Aug 24
2023, 10:50:31) [Clang 13.0.0
(clang-1300.0.29.30)] on darwin
Type "help", "copyright", "credits" or
"license" for more information.
>>> print("Hello world")
Hello world
>>>
```

To exit the REPL, hit [CTRL][D] or type `exit()`.

First Python Script

1. In a text editor write

```
#!/usr/bin/python3
print(“Hello world!”)
```

2. Save this to **myfirst.py**
3. Open a terminal, navigate to where this file is located, and run

```
python3 myfirst.py
```

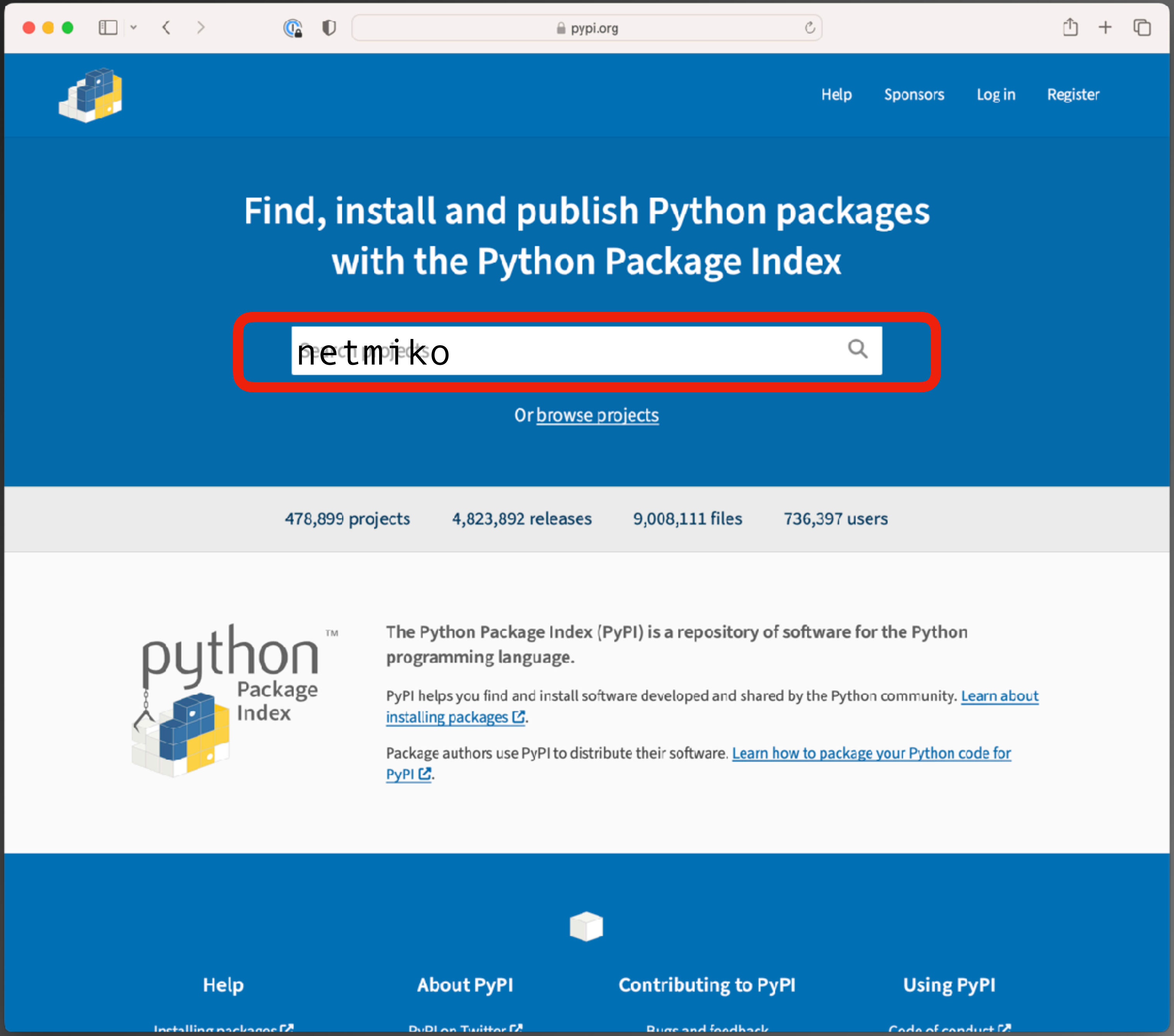
pip

pip is the package installer for Python. You can use pip to install packages from the Python Package Index and other indexes.

e.g.,

```
pip install requests
pip install netmiko
pip install gspread
```

Python Package Index (PyPI)



pypi.org

Help Sponsors Log in Register

Find, install and publish Python packages with the Python Package Index

netmiko

Or browse projects

478,899 projects 4,823,892 releases 9,008,111 files 736,397 users

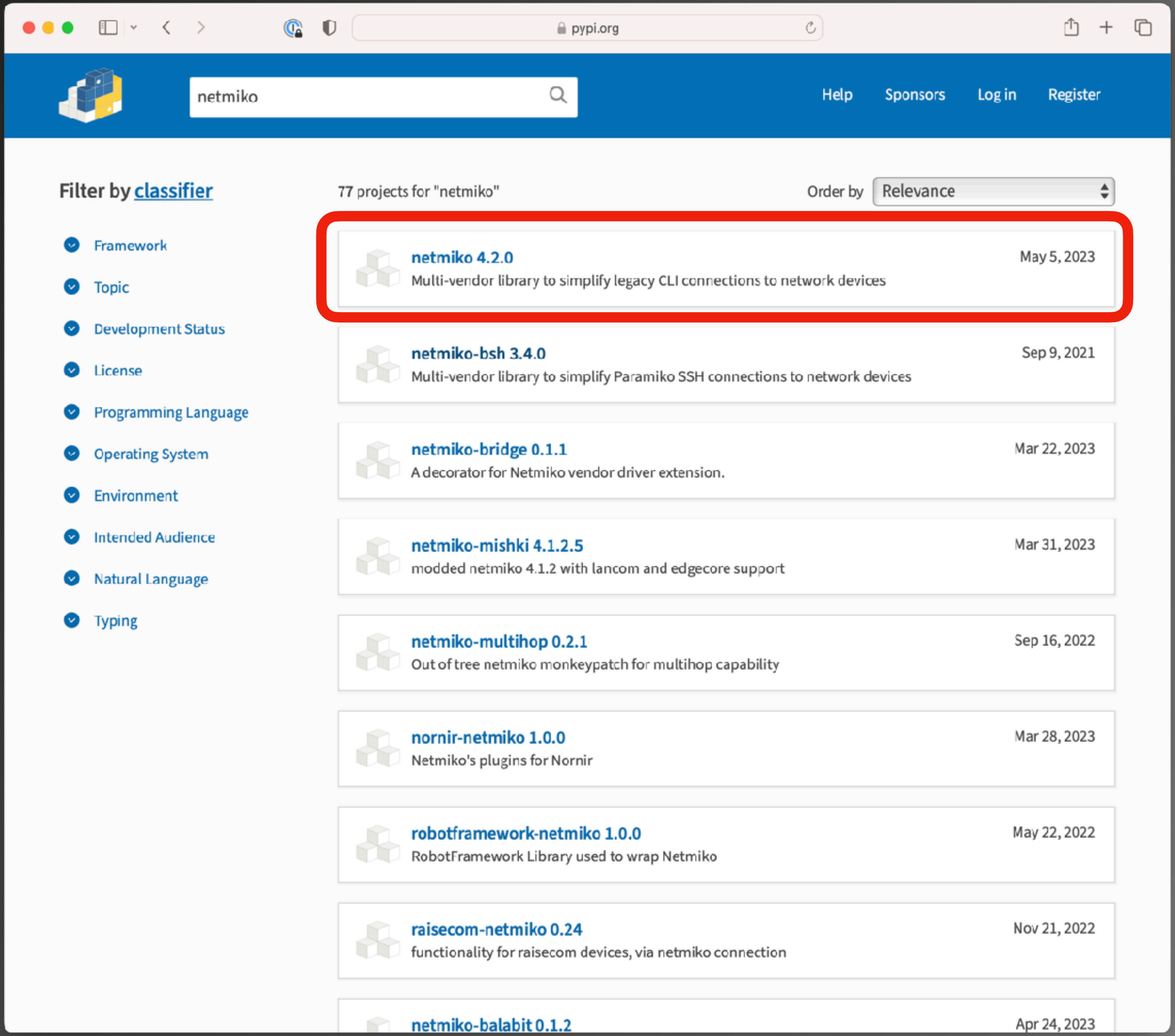
The Python Package Index (PyPI) is a repository of software for the Python programming language.

PyPI helps you find and install software developed and shared by the Python community. Learn about installing packages.

Package authors use PyPI to distribute their software. Learn how to package your Python code for PyPI.

https://www.pypi.org/

Python Package Index (PyPI)


pypi.org

netmiko

Help Sponsors Log in Register

Filter by classifier

- Framework
- Topic
- Development Status
- License
- Programming Language
- Operating System
- Environment
- Intended Audience
- Natural Language
- Typing

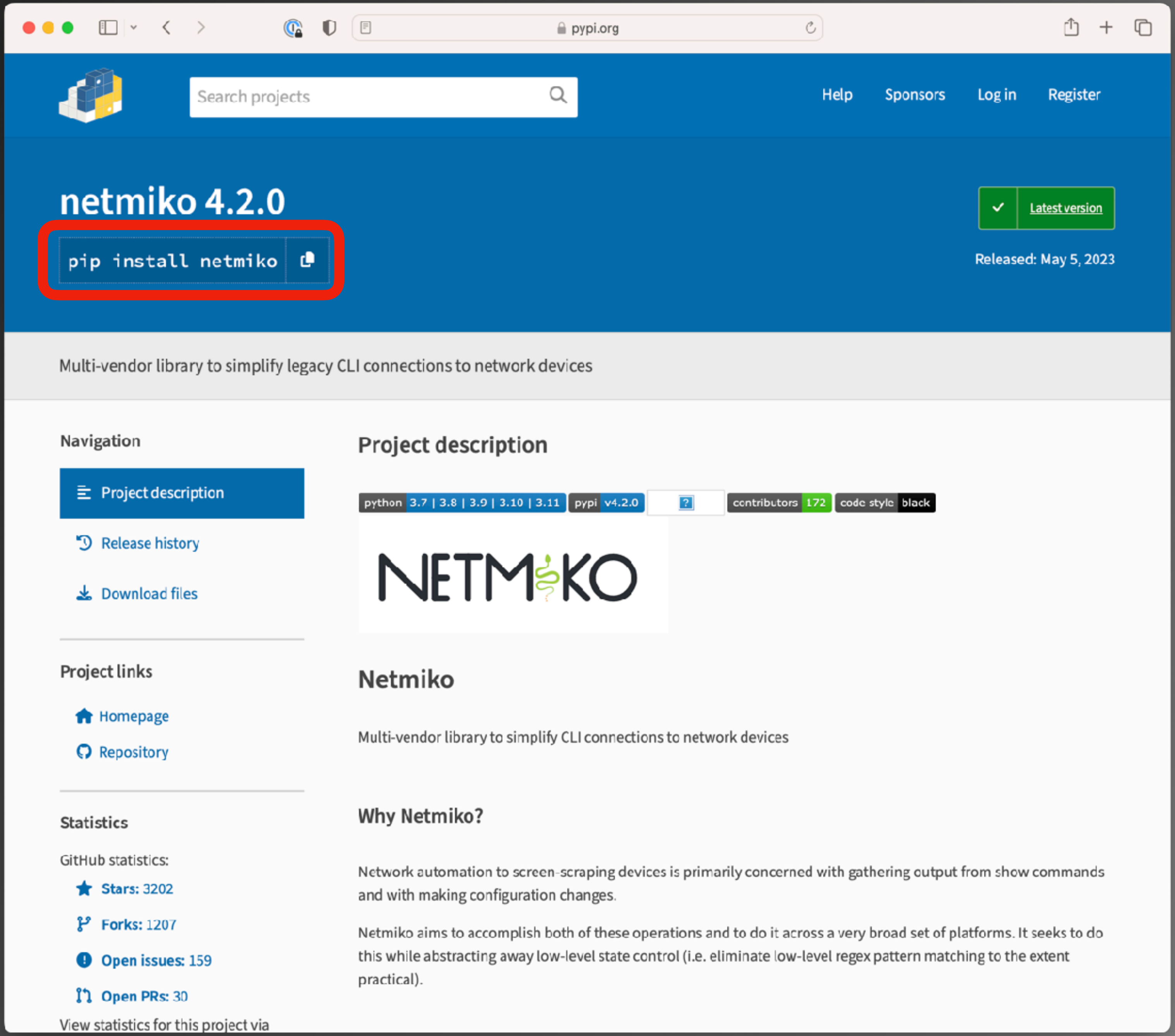
77 projects for "netmiko"

Order by Relevance

netmiko 4.2.0 Multi-vendor library to simplify legacy CLI connections to network devices	May 5, 2023
netmiko-bsh 3.4.0 Multi-vendor library to simplify Paramiko SSH connections to network devices	Sep 9, 2021
netmiko-bridge 0.1.1 A decorator for Netmiko vendor driver extension.	Mar 22, 2023
netmiko-mishki 4.1.2.5 modded netmiko 4.1.2 with lancom and edgecore support	Mar 31, 2023
netmiko-multihop 0.2.1 Out of tree netmiko monkeypatch for multihop capability	Sep 16, 2022
nornir-netmiko 1.0.0 Netmiko's plugins for Nornir	Mar 28, 2023
robotframework-netmiko 1.0.0 RobotFramework Library used to wrap Netmiko	May 22, 2022
raisecom-netmiko 0.24 functionality for raisecom devices, via netmiko connection	Nov 21, 2022
netmiko-balabit 0.1.2	Apr 24, 2023

https://www.pypi.org/

Python Package Index (PyPI)



pypi.org

netmiko 4.2.0

```
pip install netmiko
```

Released: May 5, 2023

Multi-vendor library to simplify legacy CLI connections to network devices

Statistics

GitHub statistics:

- Stars: 3202
- Forks: 1207
- Open issues: 159
- Open PRs: 30

View statistics for this project via

Project description

Netmiko

Multi-vendor library to simplify CLI connections to network devices

Why Netmiko?

Network automation to screen-scraping devices is primarily concerned with gathering output from show commands and with making configuration changes.

Netmiko aims to accomplish both of these operations and to do it across a very broad set of platforms. It seeks to do this while abstracting away low-level state control (i.e. eliminate low-level regex pattern matching to the extent practical).

https://www.pypi.org/

venv

The venv module supports creating lightweight "virtual environments", each with their own independent set of Python packages installed in their site directories. A virtual environment is created on top of an existing Python installation, known as the virtual environment's "base" Python, and may optionally be isolated from the packages in the base environment, so only those explicitly installed in the virtual environment are available.

- https://docs.python.org/3/library/venv.html

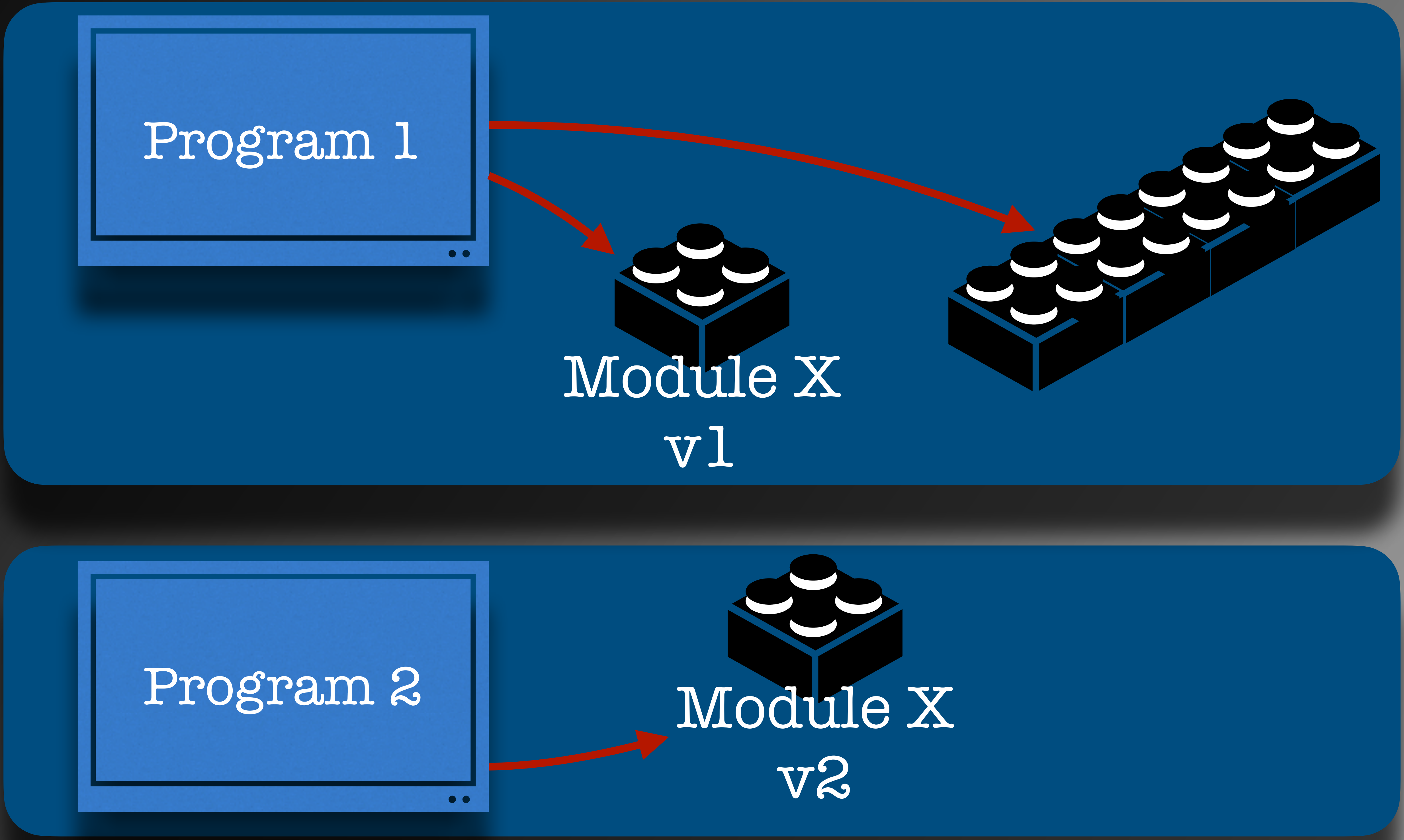
venv

So… why?

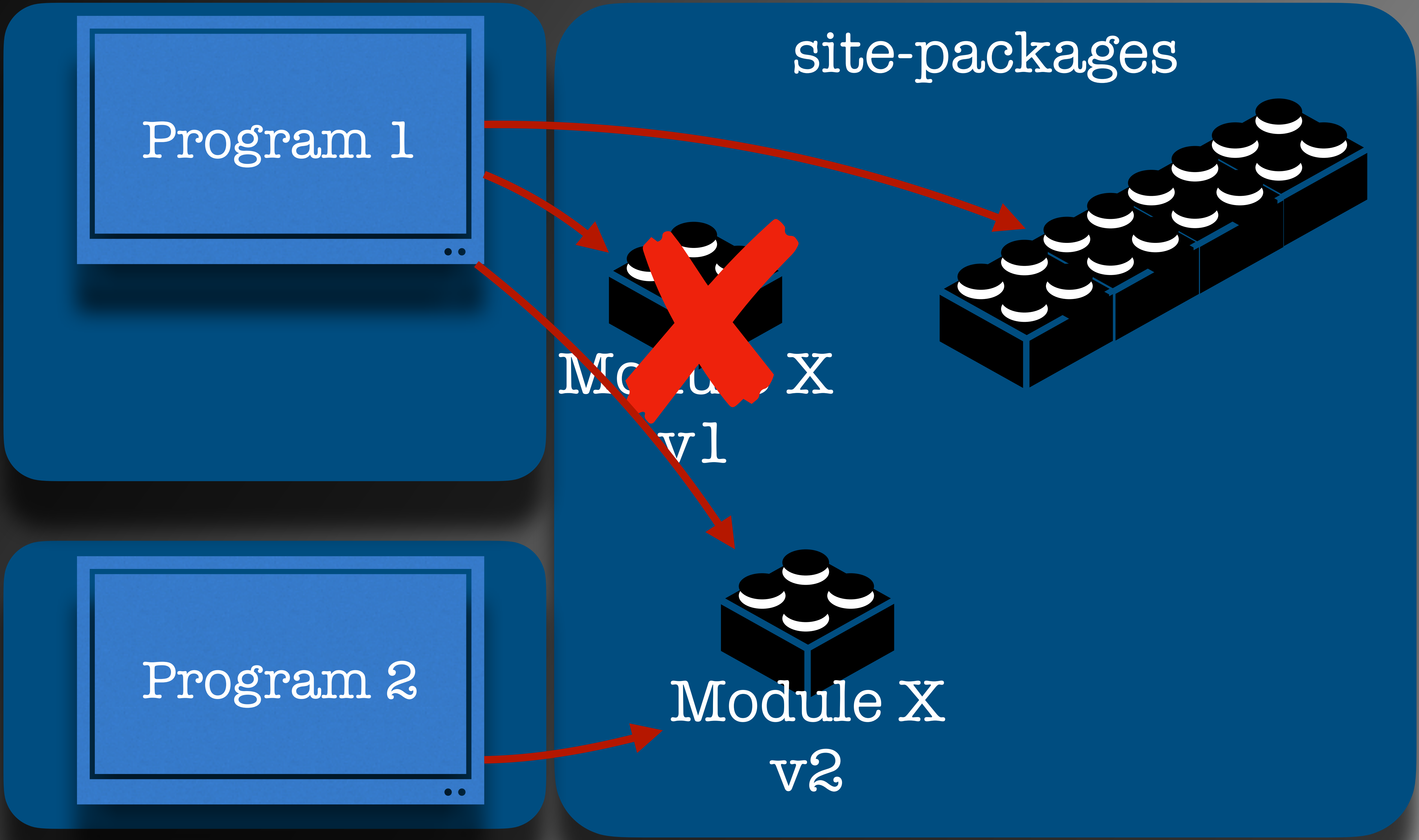
Once you begin using Python, you will inevitably encounter situations where one Python program expects a module v1 while another only works with v2. If all Python scripts are in the same environment… KABOOM!

Virtual environments allow you to isolate/separate different Python programs from each other and provide each Python program with the modules and versions it expects.

Why We Need venv



Why We Need venv



venv

For example, you might do the following:

```
$ python3 -m venv venv
$ ls -1 venv
bin
include
lib
pyvenv.cfg
$ source venv/bin/activate
(venv) $ pip list
```

This tells the Python interpreter to run module (-m) **venv** and create a new virtual environment in a directory named ‘venv’ in the current directory. We then activate that virtual environment.

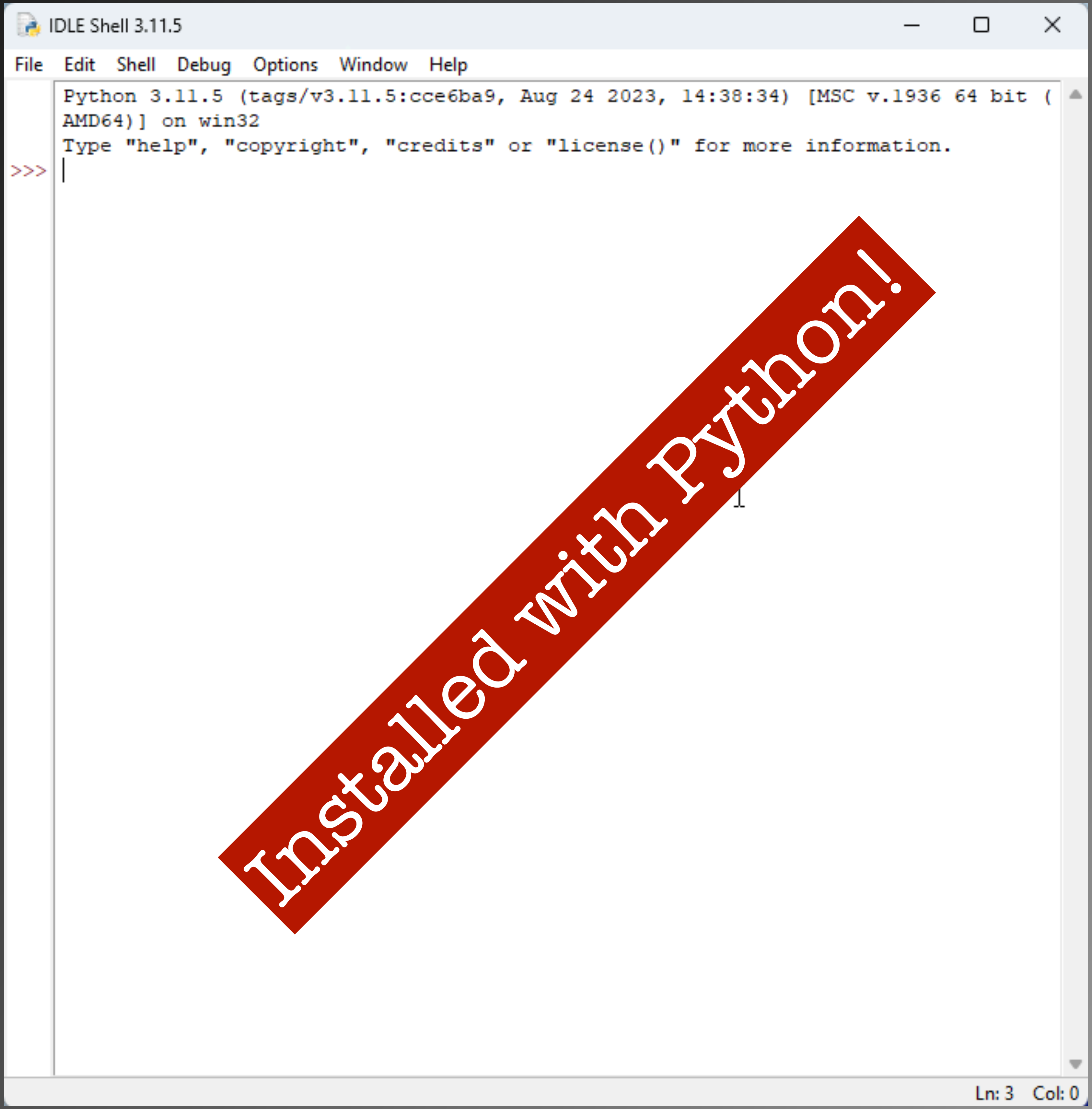
IDE

“An integrated development environment (IDE) is a software application that provides comprehensive facilities for software development. An IDE normally consists of at least a source-code editor, build automation tools, and a debugger.”
- https://en.wikipedia.org/wiki/Integrated_development_environment

Examples:

- IDLE
- Visual Studio Code (VSCode) / VSCodium
- PyCharm

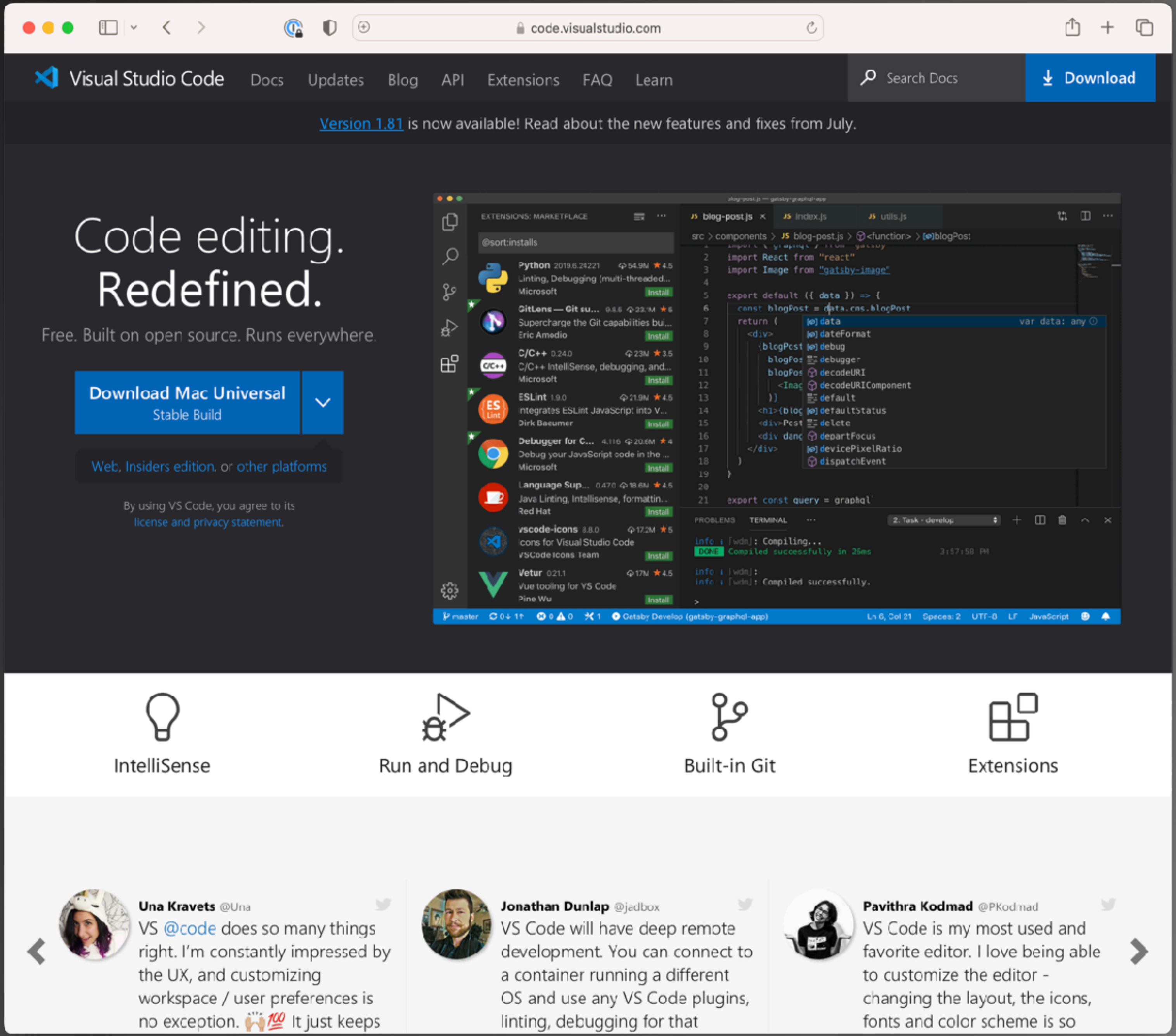
IDLE



```
Python 3.11.5 (tags/v3.11.5:cce6ba9, Aug 24 2023, 14:38:34) [MSC v.1936 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
```

Installed with Python!

Visual Studio Code (VSCode)



code.visualstudio.com

Visual Studio Code Docs Updates Blog API Extensions FAQ Learn Search Docs Download

Version 1.81 is now available! Read about the new features and fixes from July.

Code editing.
Redefined.

Free. Built on open source. Runs everywhere.

Download Mac Universal
Stable Build

Web, Insiders edition, or other platforms

By using VS Code, you agree to its
license and privacy statement.

EXTENSIONS: MARKETPLACE

@sort:installs

Python 2019.6.24221	54.9M	★4.5	Linting, Debugging (multi-threaded... Microsoft
GitLens — Git su... 9.8.5	23.1M	★5	Supercharge the Git capabilities bu... Eric Amodio
C/C++ 0.24.0	23M	★3.5	C/C++ IntelliSense, debugging, and... Microsoft
ESLint 1.9.0	21.9M	★4.5	Integrates ESLint JavaScript into V... Dirk Baeumer
Debugger for C... 4.11.6	20.6M	★4	Debug your JavaScript code in the ... Microsoft
Language Sup... 0.47.0	18.6M	★4.5	Java Linting, Intellisense, formattin... Red Hat
vscode-icons 8.8.0	17.2M	★5	Icons for Visual Studio Code VSCode Icons Team
Vetur 0.21.1	17M	★4.5	Vue tooling for VS Code Pine Wu

blog-post.js × index.js utils.js

src > components > blog-post.js > <function> > blogPost

```
import React from "react"
import Image from "gatsby-image"

export default ({ data }) => {
  const blogPost = data.cms.blogPost
  return (
    <div>
      {blogPost
        blogPos
        blogPos
          <Imag
        )}
      <h1>{blog
      <div>Post
      <div dang
    </div>
  )
}

export const query = graphql`
```

```
data                 var data: any
dateFormat
debug
debugger
decodeURI
decodeURIComponent
default
defaultStatus
delete
departFocus
devicePixelRatio
dispatchEvent
```

PROBLEMS TERMINAL ... 2: Task - develop

```
info i [wdm]: Compiling...
DONE Compiled successfully in 26ms                3:57:58 PM

info i [wdm]:
info i [wdm]: Compiled successfully.
>
```

master 0↓ 1↑ ⊗ 0 ⚠ 0 ✕ 1 Gatsby Develop (gatsby-graphql-app) Ln 6, Col 21 Spaces: 2 UTF-8 LF JavaScript

IntelliSense Run and Debug Built-in Git Extensions

Una Kravets @Una
VS @code does so many things right. I'm constantly impressed by the UX, and customizing workspace / user preferences is no exception. 🙌💯 It just keeps

Jonathan Dunlap @jedbox
VS Code will have deep remote development. You can connect to a container running a different OS and use any VS Code plugins, linting, debugging for that

Pavithra Kodmad @PKodmad
VS Code is my most used and favorite editor. I love being able to customize the editor - changing the layout, the icons, fonts and color scheme is so

https://code.visualstudio.com/

Visual Studio Code (VSCode)

VSCode offers syntax highlighting, auto-completion, integrated Git support, and too many features to list here.

Be sure to check out their extensions which provide almost everything a developer could hope for.

https://marketplace.visualstudio.com/VSCode

https://code.visualstudio.com/

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Thank You

https://frank.seesink.com/presentations/Internet2TechEx-Fall2023/

Frank Seesink
frank@seesink.com
frank@unc.edu

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Data Formats: Reading and writing JSON – YAML - XML

Maria Isabel Gandia Carriedo, CSUC/RedIRIS

network-eacademy@lists.geant.org

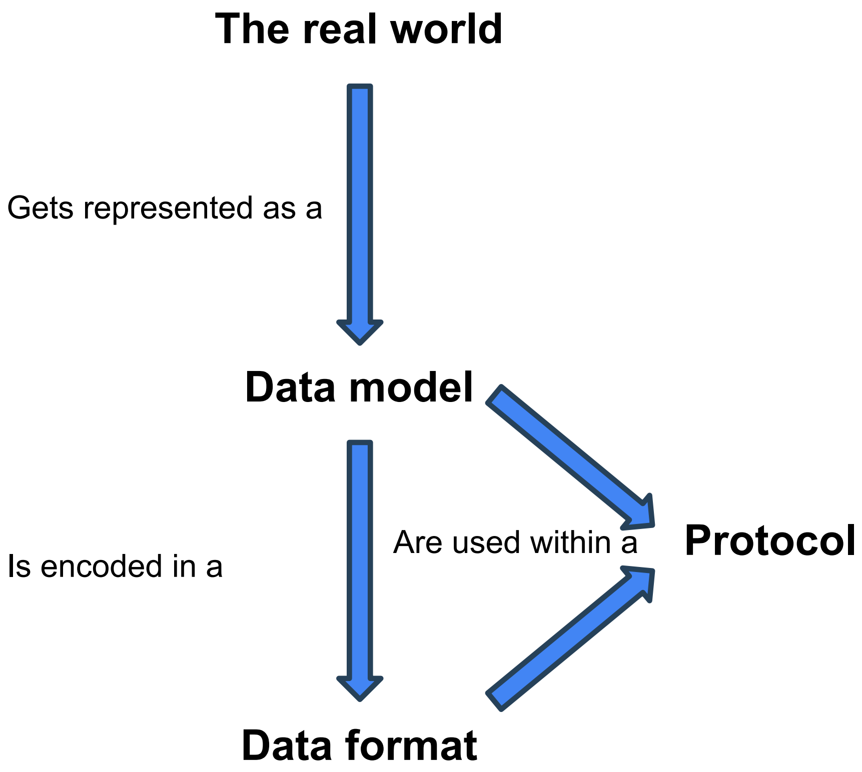
Internet2 Technology Exchange, 19-09-2023
Minneapolis, USA

Public (PU)

GN5-1

Definitions

- Data modelling (YANG, TOSCA)
  - Defines a representation of real-world entities, their relationships and structure
- Data formats (XML, JSON, YAML)
  - Define how to encode the information in a standardized way
- Protocols (NETCONF, RESTCONF, gRPC...)
  - Define the operations, the requests and responses of interactions



Data Serialisation Examples – Human Readable

```
<network>
  <device>
    <type>router</type>
    <vendor>MyOAVvendor</vendor>
    <ports>4</ports>
    <description>Access</description>
 </device>
</network>
```

XML

</>

```
{
  “device”: {
    “type”: “router”,
    “vendor”: “MyOAVvendor”,
    “ports”: 4,
    “description”: “Access”
 }
}
```

JSON

{}

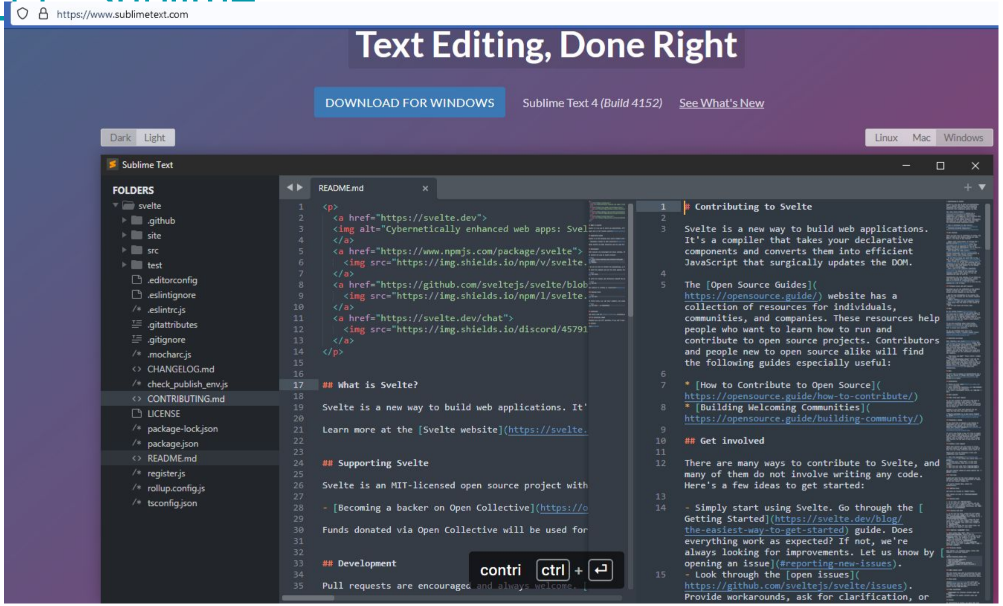
```
---
device:
  type: router
 vendor: MyOAVvendor
 ports: 4
 description: Access
```

YAML

indentation

Writing JSON, XML and YAML files

- You can write JSON, XML and YAML files with any text editor like vim or emacs
- If you like syntax highlighting, editors/IDEs such as Visual Studio Code, Notepad++, Sublime

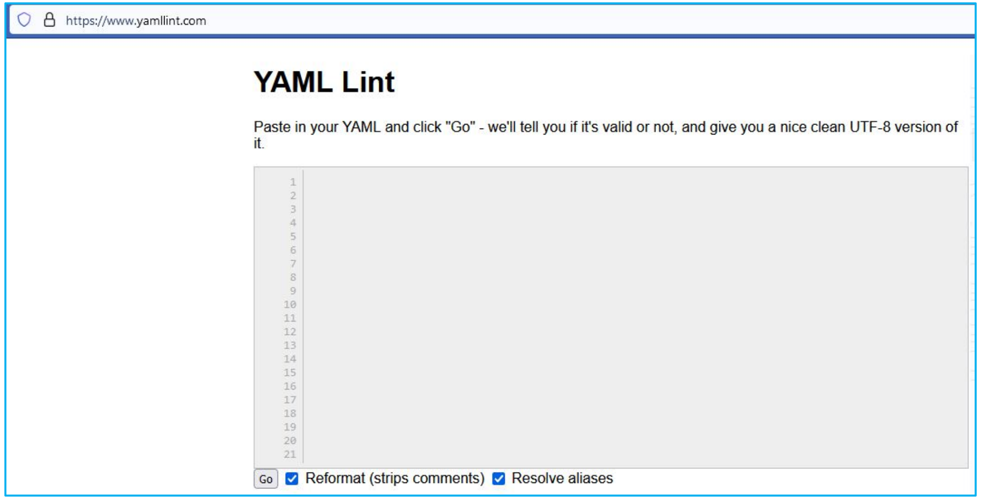


```
<p>
  <a href="https://svelte.dev">
  <img alt="Cybernetically enhanced web apps: Svel
  </a>
  <a href="https://www.npmjs.com/package/svelte">
    <img src="https://img.shields.io/npm/v/svelte.
  </a>
  <a href="https://github.com/sveltejs/svelte/blob
    <img src="https://img.shields.io/npm/l/svelte.
  </a>
  <a href="https://svelte.dev/chat">
    <img src="https://img.shields.io/discord/45791
  </a>
</p>


## What is Svelte?

Svelte is a new way to build web applications. It'

Learn more at the [Svelte website](https://svelte.


## Supporting Svelte

Svelte is an MIT-licensed open source project with

- [Becoming a backer on Open Collective](https://o

Funds donated via Open Collective will be used for


## Development

Pull requests are encouraged and always welcome. [
```

```
# Contributing to Svelte

Svelte is a new way to build web applications.
It's a compiler that takes your declarative
components and converts them into efficient
JavaScript that surgically updates the DOM.

The [Open Source Guides](
https://opensource.guide/) website has a
collection of resources for individuals,
communities, and companies. These resources help
people who want to learn how to run and
contribute to open source projects. Contributors
and people new to open source alike will find
the following guides especially useful:

* [How to Contribute to Open Source](
https://opensource.guide/how-to-contribute/)
* [Building Welcoming Communities](
https://opensource.guide/building-community/)

## Get involved

There are many ways to contribute to Svelte, and
many of them do not involve writing any code.
Here's a few ideas to get started:

- Simply start using Svelte. Go through the [
Getting Started](https://svelte.dev/blog/
the-easiest-way-to-get-started) guide. Does
everything work as expected? If not, we're
always looking for improvements. Let us know by [
opening an issue](#reporting-new-issues).
- Look through the [open issues](
https://github.com/sveltejs/svelte/issues).
Provide workarounds, ask for clarification, or
```

Some Free Tools to Help You Write, Validate and Convert Your Files

- You can check your syntax, format your files or convert them using useful free tools:
    - https://www.freeformatter.com
    - https://www.liquid-technologies.com/online-xml-validator
    - https://onlineyamltools.com/edit-yaml
    - https://www.yamllint.com/
    - https://www.json2yaml.com/



https://www.yamllint.com

YAML Lint

Paste in your YAML and click "Go" - we'll tell you if it's valid or not, and give you a nice clean UTF-8 version of it.

☑ Reformat (strips comments) ☑ Resolve aliases

Some Cases Where We Use JSON, YAML, XML

JSON:

- Web API output (AWS, Google maps, Github, X,...)
- Jenkins
- ELK stack (Elasticsearch, Logstash, Kibana)

XML:

- Jenkins
- NETCONF
- RESTCONF

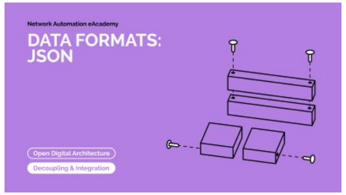
YAML:

- Ansible
- Kubernetes
- Docker

```xml
<copy-config>
        <target>
                <startup/>
        </target>
        <source>
                <running/>
        </source>
</copy-config>
```

```yaml
- hosts: core
  tasks:
  - name: Describe router interfaces
    ios_interface:
      name:        "{{ item.name }}"
      description: "{{ item.description }}"
      state: present
      provider: "{{ credentials }}"
    with_items:
      - { name: Ethernet0/0, description: “One" }
      - { name: Ethernet0/1, description: “Two” }
```

More Information in the Network Automation eAcademy


Network Automation eAcademy

DATA FORMATS: JSON

Open Digital Architecture

Decoupling & Integration

- **Formats: YAML** (30')

Network Automation eAcademy

DATA FORMATS: XML

Open Digital Architecture

Decoupling & Integration

- **Formats: XML** (60’)

Network Automation eAcademy

DATA FORMATS: YAML

Open Digital Architecture

Decoupling & Integration

- **Formats: JSON** (45’)

https://wiki.geant.org/display/NETDEV/OAV+Training+Portal

GÉANT

Thank You!

https://wiki.geant.org/display/NETDEV/NeA
network-eacademy@lists.geant.org
netdev@lists.geant.org

www.geant.org

Co-funded by
the European Union

Automating with Google Sheets

Amy Liebowitz - University of Michigan

- At U of Michigan we use Google Sheets for network projects
  - Cut sheets for network migrations
  - VLAN port assignments for new access layer devices
  - Core point-to-point and loopback assignments
- More convenient than formal tools/databases
  - Easy to use by non-technical people (like PMs)
  - Easy to share and edit
  - Printable for field technicians
- Wouldn’t it be nice if we could derive network configurations from these?
  - You can, and it’s not that hard!
  - Enter gspread - a python api for Google Sheets
  - (NB: If you’re more comfortable with javascript check out Google Apps Script)

Automating with Google Sheets

- Step 1: Set up a Service Account
  - “Bot” account will generate credentials that can be used by your code.
  - Share a spreadsheet with the bot account’s email and your code can access it just like any other user
    - We share our network projects folder with our bot account
    - We store our bot account’s credentials in Cyberark
- Step 2: Create a Spreadsheet
- Step 3: Write code to pull in spreadsheet data
  - gspread’s **get_all_records** method generates a list of dictionaries keyed on column headers
- Step 4: Create a Template
- Step 5: Generate configlets!

Automating with Google Sheets

- References
  - gspread docs: https://docs.gspread.org/en/v5.10.0/index.html
  - gspread example repository: https://github.com/amylieb/gspread-example

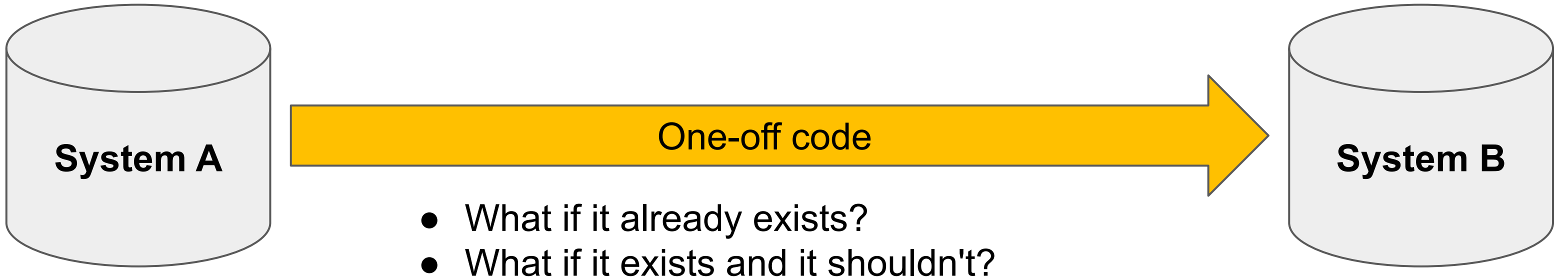
2023 INTERNET2 TECHNOLOGY exchange

Tapas: DiffSync

Compare & Sync two different data-sources

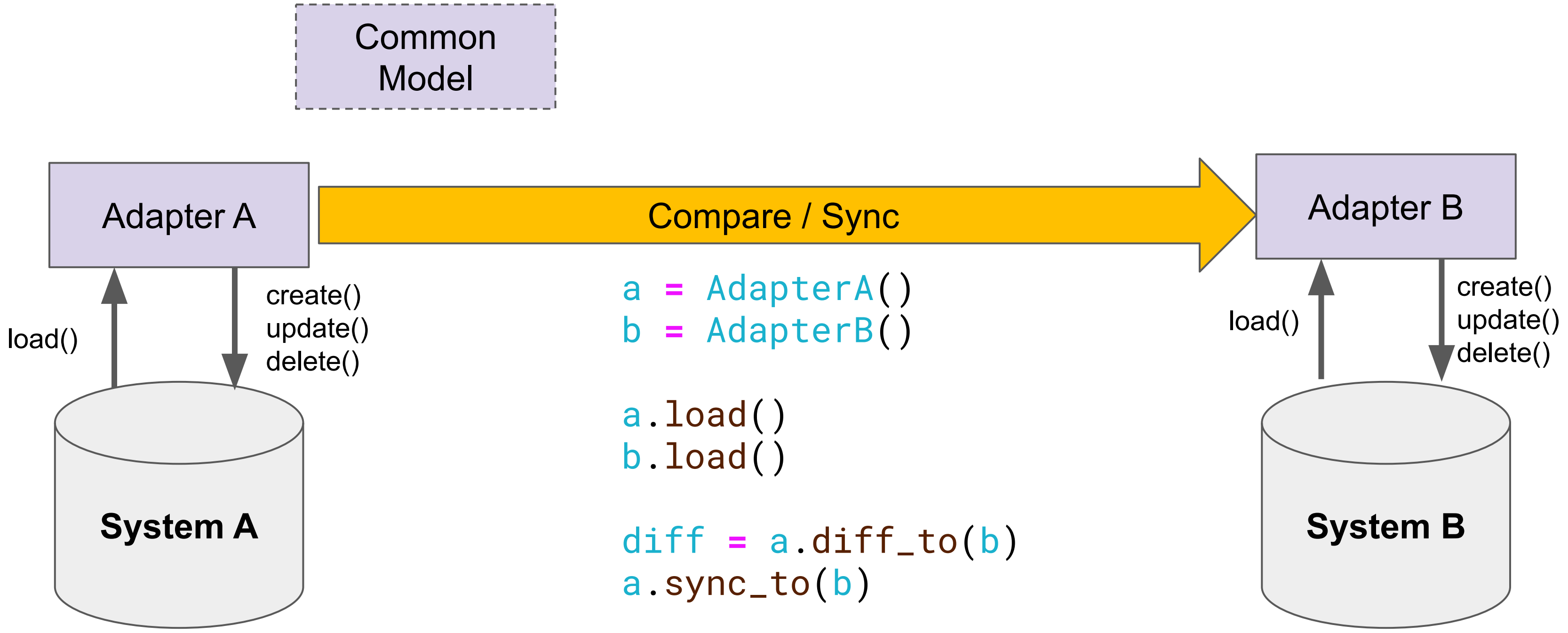
James Harr, Sr NetDevOps Engineer, Internet2

The Typical Pattern



- What if it already exists?
- What if it exists and it shouldn't?
- Should I delete in B if it's missing in A?
  - What if I need to change this?
- What if I only want to update objects that exist in both?

DiffSync - The framework



```
a = AdapterA()
b = AdapterB()

a.load()
b.load()

diff = a.diff_to(b)
a.sync_to(b)
```

Defining the Model

```python
from diffsync import DiffSyncModel

class Device(DiffSyncModel):
    _modelname = "device"
    _identifiers = ("name",)
    _shortname = ()
    _attributes = ("addr", "model", "sn")
    _children = {"interface": "interfaces"}

    name: str
    addr: Union[IPv6Address, IPv4Address]
    model: str
    sn: Optional[str]

    interfaces: List[Interface]
```

Defining the Model

```python
class Interface(DiffSyncModel):
    _modelname = "interface"
    _identifiers = ("device_name","intf_name")
    _shortname = ()
    _attributes = ("description", "speed")
    _children = {}

    device_name: str
    intf_name: str
    description: Optional[str]
    speed: Optional[int]   # Mbps
```

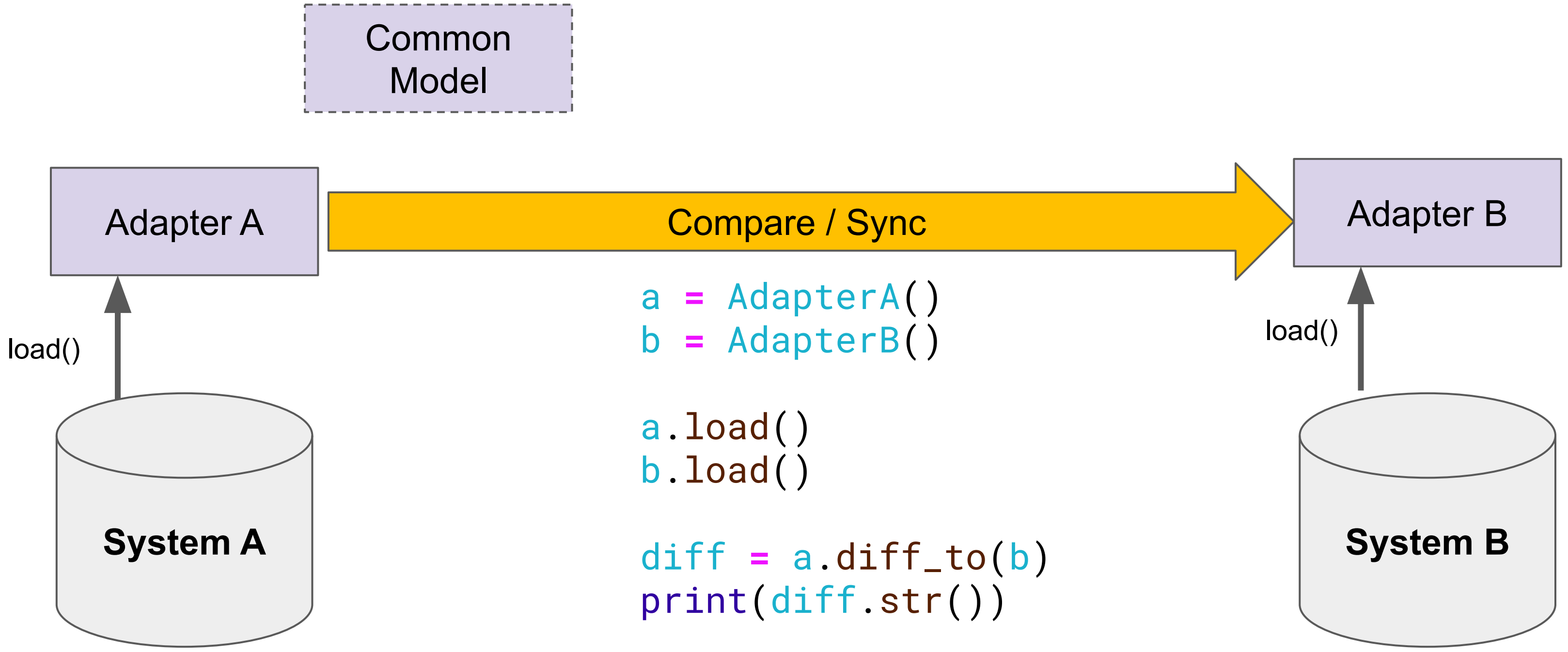
Defining an Adapter

```python
class NautobotDevice(Device):
    pass

class NautobotInterface(Interface):
    pass

class NautobotBackend(diffsync.DiffSync):
    device = NautobotDevice
    interface = NautobotInterface

    def load(self):
        ...
```

Defining an Adapter

```
class NautobotBackend(diffsync.DiffSync):
    def load(self):
        d1 = Device(name="rtr1", addr="2001:db8::1",
                    model="8201", sn="1234")
        self.add(d1)

        intf1 = Interface(device_name="rtr1", name="eth1/1")
        self.add(intf1)
        d1.add_child(intf1)
```

DiffSync - providing a framework

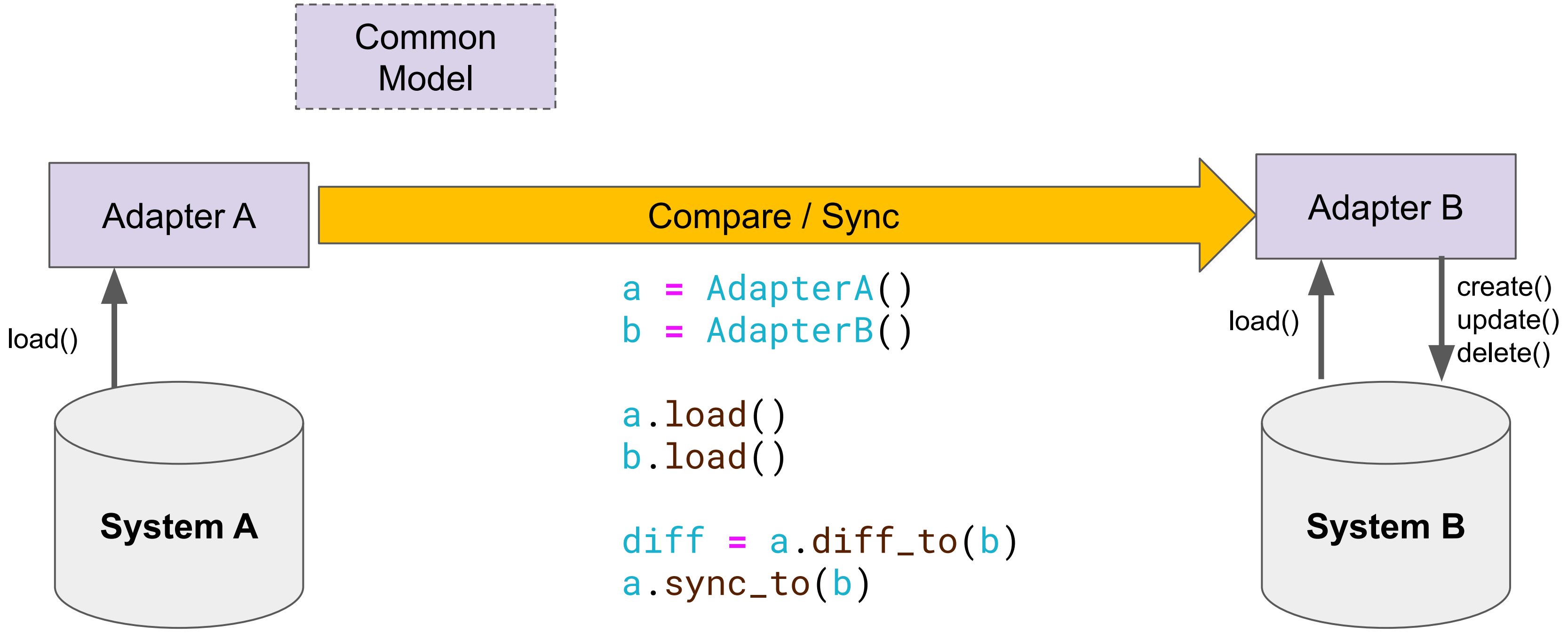


```
a = AdapterA()
b = AdapterB()

a.load()
b.load()

diff = a.diff_to(b)
print(diff.str())
```

Viewing the Diff

```
device
  device: rtr1 MISSING in SNBackend
    interface
      interface: rtr1__eth1/1 MISSING in SNBackend
      interface: rtr1__eth1/2 MISSING in SNBackend
  device: rtr2 MISSING in NautobotBackend
    interface
      interface: rtr2__eth1/1 MISSING in NautobotBackend
      interface: rtr2__eth1/2 MISSING in NautobotBackend
  device: rtr3
    sn        NautobotBackend(abc123)   SNBackend(def456)
    interface
      interface: rtr3__eth1/3 MISSING in SNBackend
```

DiffSync - providing a framework



```
a = AdapterA()
b = AdapterB()

a.load()
b.load()

diff = a.diff_to(b)
a.sync_to(b)
```

Saving Data

```
class SNDevice(Device):
    sn_id: str # Stashed UUID for the Device in SN

    @classmethod
    def create(
        cls,
        diffsync: SNBackend,
        ids: Dict[str, str],
        attrs: Dict[str, str],
    ) -> DiffSyncModel | None:
        sn_id = service_now_api.create(...)
        return cls(**ids, **attrs, sn_id=sn_id)
```

Saving Data

```
class SNDevice(Device):
    def update(
        self,
        attrs: Dict[str, str],
    ) -> DiffSyncModel | None:

        service_now_api.update(id=self.sn_uuid, ...)

        return super().update(attrs)
```

Saving Data

```
class SNDevice(Device):
    def delete(self) -> DiffSyncModel | None:

        service_now_api.update(id=self.sn_uuid, status="DECOM")

        return super().delete()
```

DiffSync - what does this get you?

- Structured development
- Re-run sync process
- Potentially more than just 2 "backends"
- Easier testing

```
@patch("nautobot.api_call")
def test_load(...):
    m = MockBackend(); m.load()  # <-- mock data
    a = MyBackend(); a.load()
    diff = m.diff_to(a)
    assert not diff.has_diffs()  # <-- yay
```

- Selective-sync with (nearly) the same code

```
    a = MyBackend()
    a.load_site("building1")
```

2023 INTERNET2 TECHNOLOGY exchange

Tapas: Bash Incantations

Shannon Byrnes, NetDevOps Engineer, Internet2

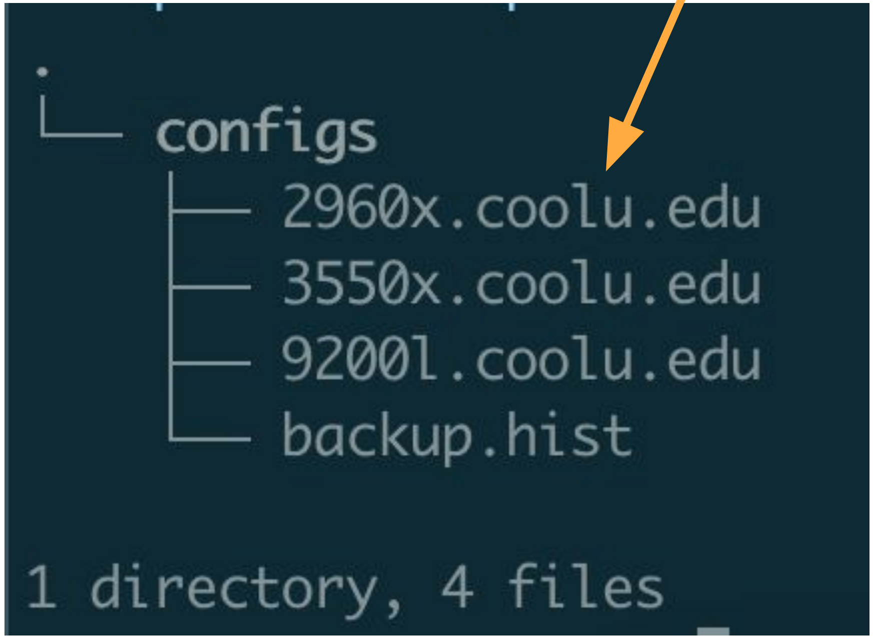
Bash Magic with Config Files

- There is a lot you can do and glean with a folder of configs and bash one-liners. No Python involved.

- This tapa will show a few bash commands using a folder of configs

- *Note*: ChatGPT isn't bad at generating fake configs if you're detailed enough.

Our switch configs are FQDNs



```
.
└── configs
    ├── 2960x.coolu.edu
    ├── 3550x.coolu.edu
    ├── 9200l.coolu.edu
    └── backup.hist

1 directory, 4 files
```

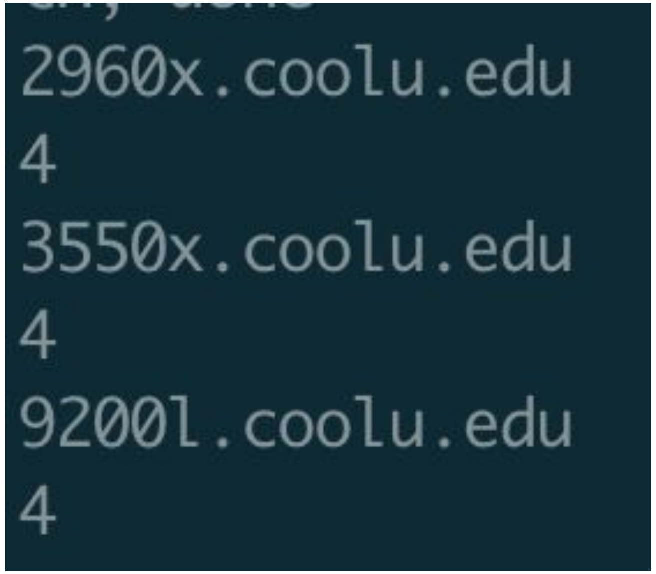
1. Number of Ports by VLAN ID

Incantation Form

```
for SWITCH in $(ls | grep coolu.edu); do echo $SWITCH; grep -c "^
switchport access vlan 100$" $SWITCH; done
```

Script Form

```
for SWITCH in $(ls | grep
coolu.edu)
do
    echo $SWITCH
    grep -c "^ switchport access
    vlan 100$" $SWITCH
done
```



```
2960x.coolu.edu
4
3550x.coolu.edu
4
9200l.coolu.edu
4
```

1. Number of Ports by VLAN ID

Script Form

```
for SWITCH in $(ls | grep coolu.edu)
do
    echo $SWITCH
    grep -c "^ switchport access vlan 100$" $SWITCH
done
```

Print filename so we know which device we are looking at

Return matching lines as a Count.

Only match lines that start with a single space followed by the rest of the pattern.

This is the filename we capture in the top line on each loop.

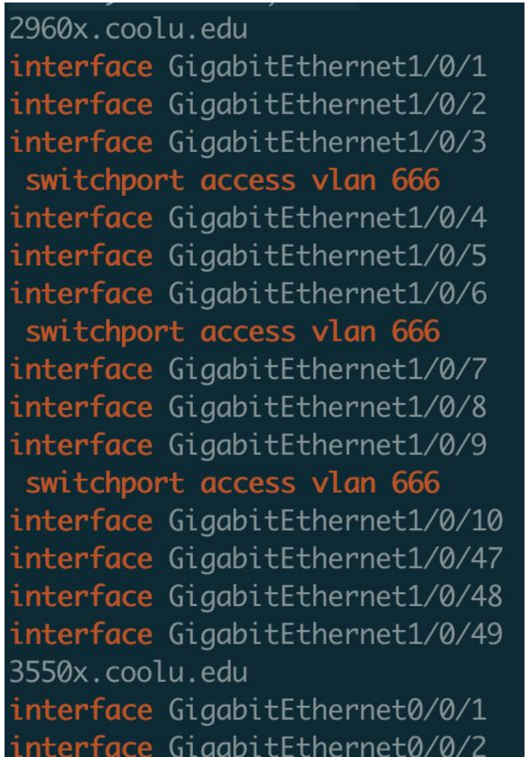
2. Find Available Ports Based on a Black Hole VLAN

Incantation Form

```
for SWITCH in $(ls | grep coolu.edu); do echo $SWITCH; egrep
'^(interface | switchport access vlan 666$)' $SWITCH; done
```

Script Form

```
for SWITCH in $(ls | grep coolu.edu)
do
    echo $SWITCH
    egrep '^(interface | switchport
access vlan 666$)' $SWITCH
done
```



```
2960x.coolu.edu
interface GigabitEthernet1/0/1
interface GigabitEthernet1/0/2
interface GigabitEthernet1/0/3
 switchport access vlan 666
interface GigabitEthernet1/0/4
interface GigabitEthernet1/0/5
interface GigabitEthernet1/0/6
 switchport access vlan 666
interface GigabitEthernet1/0/7
interface GigabitEthernet1/0/8
interface GigabitEthernet1/0/9
 switchport access vlan 666
interface GigabitEthernet1/0/10
interface GigabitEthernet1/0/47
interface GigabitEthernet1/0/48
interface GigabitEthernet1/0/49
3550x.coolu.edu
interface GigabitEthernet0/0/1
interface GigabitEthernet0/0/2
```

2. Find Available Ports Based on a Black Hole VLAN

Same start

Script Form

```
for SWITCH in $(ls | grep coolu.edu)
do
    echo $SWITCH
    egrep '^(interface | switchport access vlan 666$)' $SWITCH
done
```

Match On
Line starts with exactly “interface “
OR
Line exactly matches “ switchport access vlan 666”

3.A Move Switchports From One VLAN to Another

Incantation Form

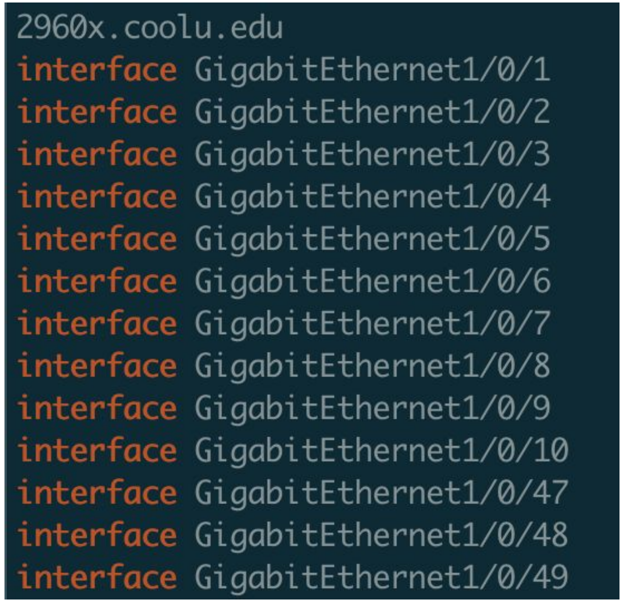
```
for SWITCH in $(ls | grep coolu.edu); do echo $SWITCH; egrep
'^(interface | switchport access vlan 300$)' $SWITCH; done
```

Script Form

```
for SWITCH in $(ls | grep coolu.edu)
do
    echo $SWITCH
    egrep '^(interface | switchport access vlan 300$)' $SWITCH
done
```

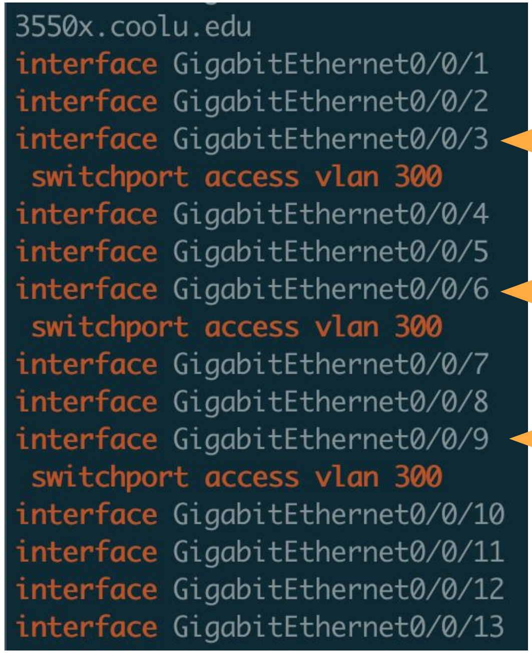
3.A Move Switchports From One VLAN to Another

Output

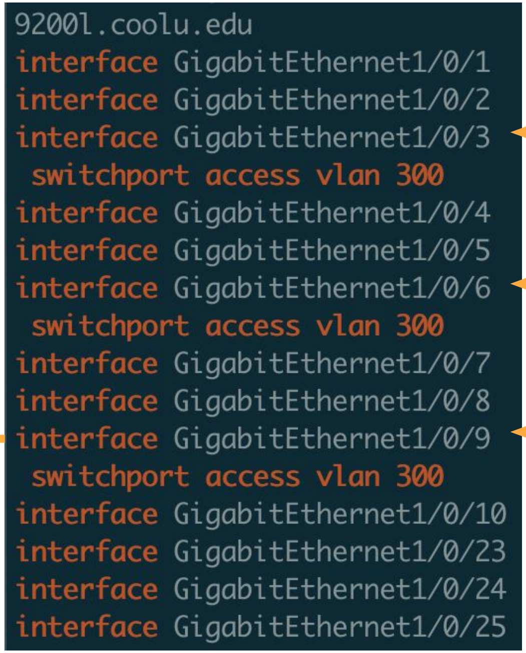


```
2960x.coolu.edu
interface GigabitEthernet1/0/1
interface GigabitEthernet1/0/2
interface GigabitEthernet1/0/3
interface GigabitEthernet1/0/4
interface GigabitEthernet1/0/5
interface GigabitEthernet1/0/6
interface GigabitEthernet1/0/7
interface GigabitEthernet1/0/8
interface GigabitEthernet1/0/9
interface GigabitEthernet1/0/10
interface GigabitEthernet1/0/47
interface GigabitEthernet1/0/48
interface GigabitEthernet1/0/49
```

None here!



```
3550x.coolu.edu
interface GigabitEthernet0/0/1
interface GigabitEthernet0/0/2
interface GigabitEthernet0/0/3
 switchport access vlan 300
interface GigabitEthernet0/0/4
interface GigabitEthernet0/0/5
interface GigabitEthernet0/0/6
 switchport access vlan 300
interface GigabitEthernet0/0/7
interface GigabitEthernet0/0/8
interface GigabitEthernet0/0/9
 switchport access vlan 300
interface GigabitEthernet0/0/10
interface GigabitEthernet0/0/11
interface GigabitEthernet0/0/12
interface GigabitEthernet0/0/13
```



```
9200l.coolu.edu
interface GigabitEthernet1/0/1
interface GigabitEthernet1/0/2
interface GigabitEthernet1/0/3
 switchport access vlan 300
interface GigabitEthernet1/0/4
interface GigabitEthernet1/0/5
interface GigabitEthernet1/0/6
 switchport access vlan 300
interface GigabitEthernet1/0/7
interface GigabitEthernet1/0/8
interface GigabitEthernet1/0/9
 switchport access vlan 300
interface GigabitEthernet1/0/10
interface GigabitEthernet1/0/23
interface GigabitEthernet1/0/24
interface GigabitEthernet1/0/25
```

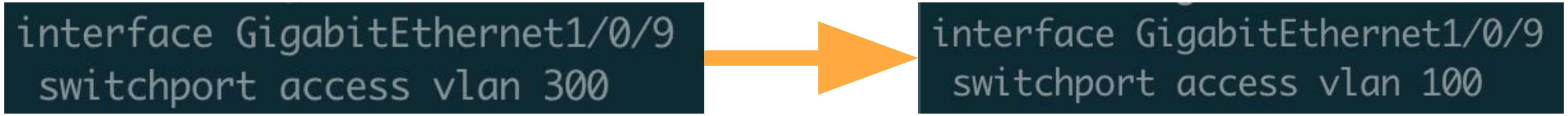
3.B Move Switchports From One VLAN to Another

Incantation Form

```
< OUR LAST COMMAND > | sed 's/vlan 300/vlan 100/g'
```

Full Incantation

```
for SWITCH in $(ls | grep coolu.edu); do echo $SWITCH;

egrep '^(interface | switchport access vlan 300$)' $SWITCH; done |

sed 's/vlan 300/vlan 100/g'
```



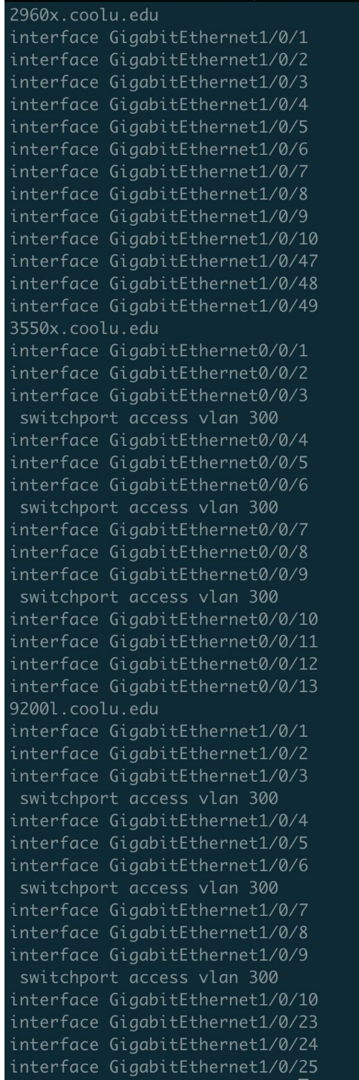
```
interface GigabitEthernet1/0/9
 switchport access vlan 300
```

```
interface GigabitEthernet1/0/9
 switchport access vlan 100
```

3.B Move Switchports From One VLAN to Another

Tada! Now you can copy and paste for each device.

As we know, unless a VLAN change would occur, all the extra lines will be no-ops. However, some cleanup will be easier on the eyes, so I won't stop you.



```
2960x.coolu.edu
interface GigabitEthernet1/0/1
interface GigabitEthernet1/0/2
interface GigabitEthernet1/0/3
interface GigabitEthernet1/0/4
interface GigabitEthernet1/0/5
interface GigabitEthernet1/0/6
interface GigabitEthernet1/0/7
interface GigabitEthernet1/0/8
interface GigabitEthernet1/0/9
interface GigabitEthernet1/0/10
interface GigabitEthernet1/0/47
interface GigabitEthernet1/0/48
interface GigabitEthernet1/0/49
3550x.coolu.edu
interface GigabitEthernet0/0/1
interface GigabitEthernet0/0/2
interface GigabitEthernet0/0/3
 switchport access vlan 300
interface GigabitEthernet0/0/4
interface GigabitEthernet0/0/5
interface GigabitEthernet0/0/6
 switchport access vlan 300
interface GigabitEthernet0/0/7
interface GigabitEthernet0/0/8
interface GigabitEthernet0/0/9
 switchport access vlan 300
interface GigabitEthernet0/0/10
interface GigabitEthernet0/0/11
interface GigabitEthernet0/0/12
interface GigabitEthernet0/0/13
9200l.coolu.edu
interface GigabitEthernet1/0/1
interface GigabitEthernet1/0/2
interface GigabitEthernet1/0/3
 switchport access vlan 300
interface GigabitEthernet1/0/4
interface GigabitEthernet1/0/5
interface GigabitEthernet1/0/6
 switchport access vlan 300
interface GigabitEthernet1/0/7
interface GigabitEthernet1/0/8
interface GigabitEthernet1/0/9
 switchport access vlan 300
interface GigabitEthernet1/0/10
interface GigabitEthernet1/0/23
interface GigabitEthernet1/0/24
interface GigabitEthernet1/0/25
```

2023 INTERNET2 TECHNOLOGY exchange

Tapas: Getting Started with Ansible

AJ Ragusa, Manager Network Automation and Performance - GlobalNOC

What is Ansible?

- Software tool for simple but powerful automation on cross-platform systems.
- Common use cases:
    - Application Deployment
    - Updates
    - Cloud Provisioning
    - Configuration Management
    - Intra-service orchestration
    - Any reproducible tasks!
- Generally Idempotent - each module is different but most are idempotent
- Support for many different network devices and protocols
    - Cisco (IOS and IOSXR), Juniper, Arista, Aruba

Playbooks and Tasks

- Playbooks are how “tasks” are organized to be executed on the selected devices
  - Playbooks also specify inventories and hosts to be applied to as well as any additional parameters needed for those tasks (variables, roles, collections)
  - Playbooks are written in YAML
  - Can do loops, use blocks, and re-use code using Roles and Collections
- Tasks are “Actions” that should be applied to the selected devices and must be contained inside of a play
  - Tasks should be idempotent in most cases (every time you run it the end state should be the same)
  - Tasks is the smallest unit that can be executed in Ansible

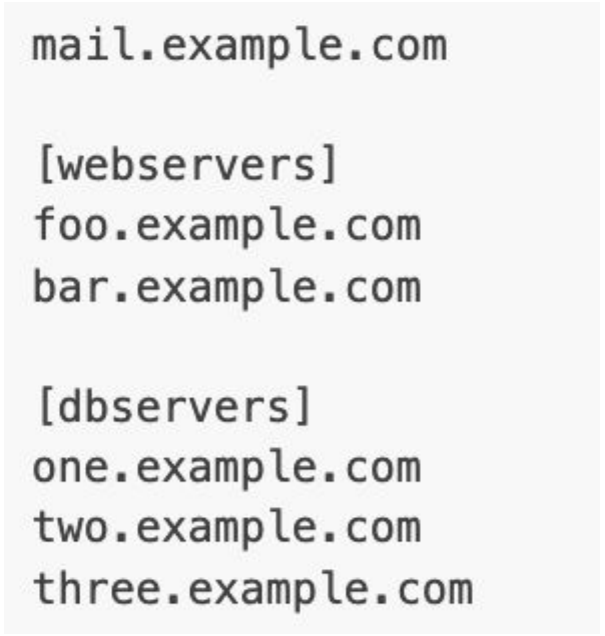
Inventory + Roles + Collection

- The Inventory is the set of hosts that can be executed on by an ansible play
    - Can also contain additional variables for each device
    - Usually specified in JSON/YAML/INI format
- Roles provide the ability to re-use tasks across multiple playbooks
    - Written in YAML can be included into multiple playbooks
    - Individual tasks inside of the role can be executed by the playbook
    - Essentially allows for code re-use
- Collections provide a higher level of re-use, can include playbooks, roles, modules and plugins
    - Similar to roles, collections can be included in your playbook

Lets build an Inventory

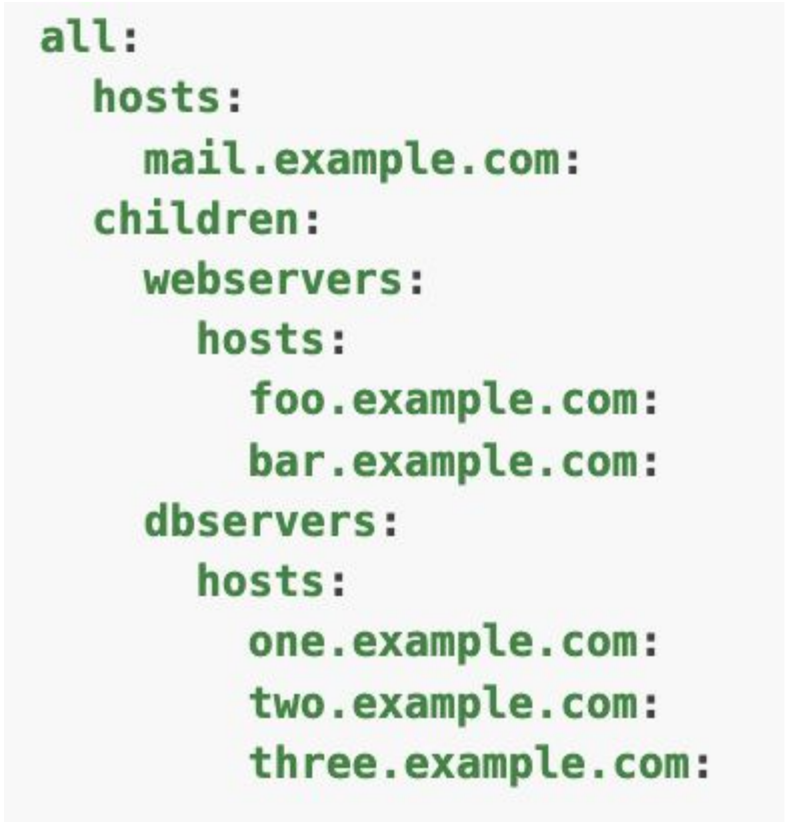
Super Basic inventory

INI format



```
mail.example.com

[webservers]
foo.example.com
bar.example.com

[dbservers]
one.example.com
two.example.com
three.example.com
```

YAML format



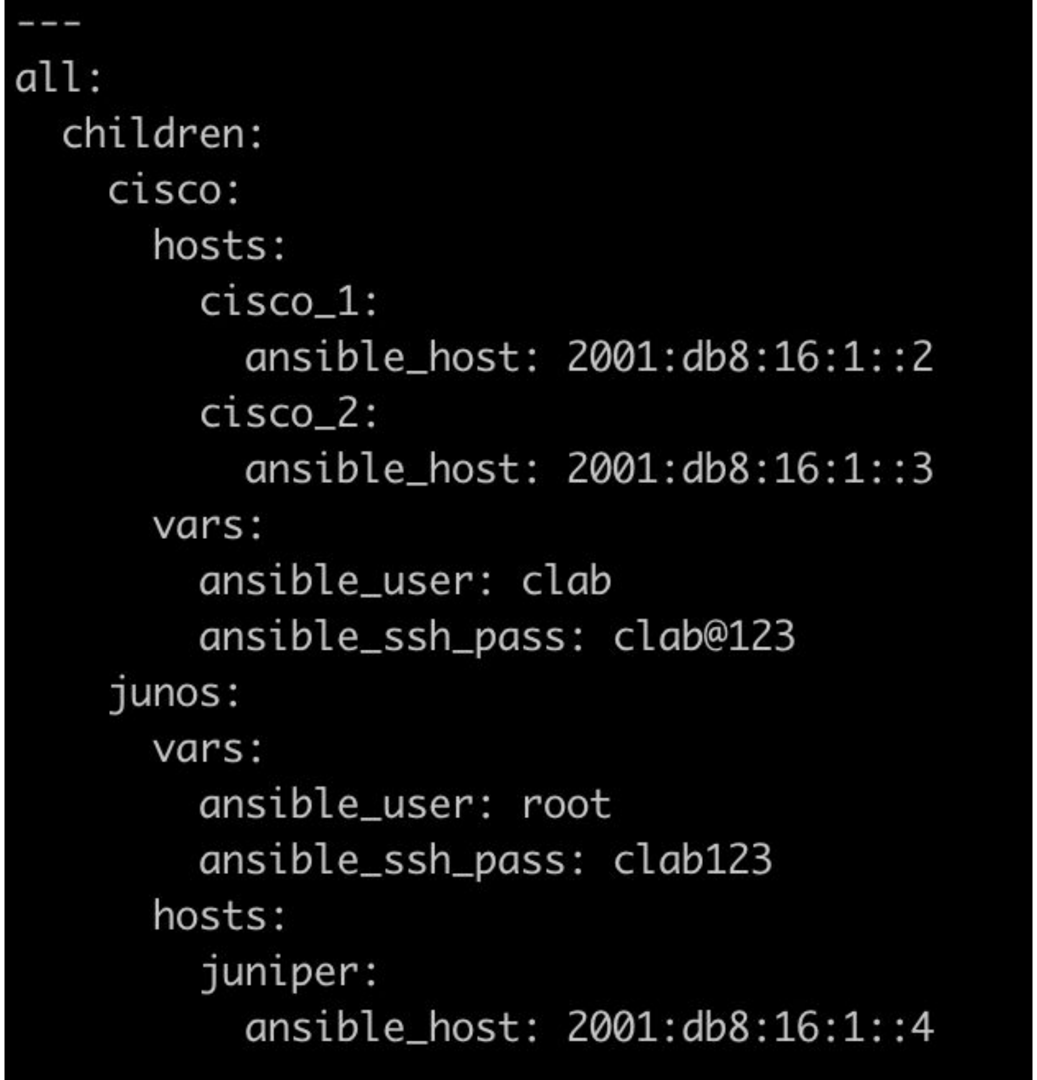
```
all:
  hosts:
    mail.example.com:
  children:
    webservers:
      hosts:
        foo.example.com:
        bar.example.com:
    dbservers:
      hosts:
        one.example.com:
        two.example.com:
        three.example.com:
```

Inventory “assigning variables”

```
[atlanta]
host1 http_port=80 maxRequestsPerChild=808
host2 http_port=303 maxRequestsPerChild=909
```

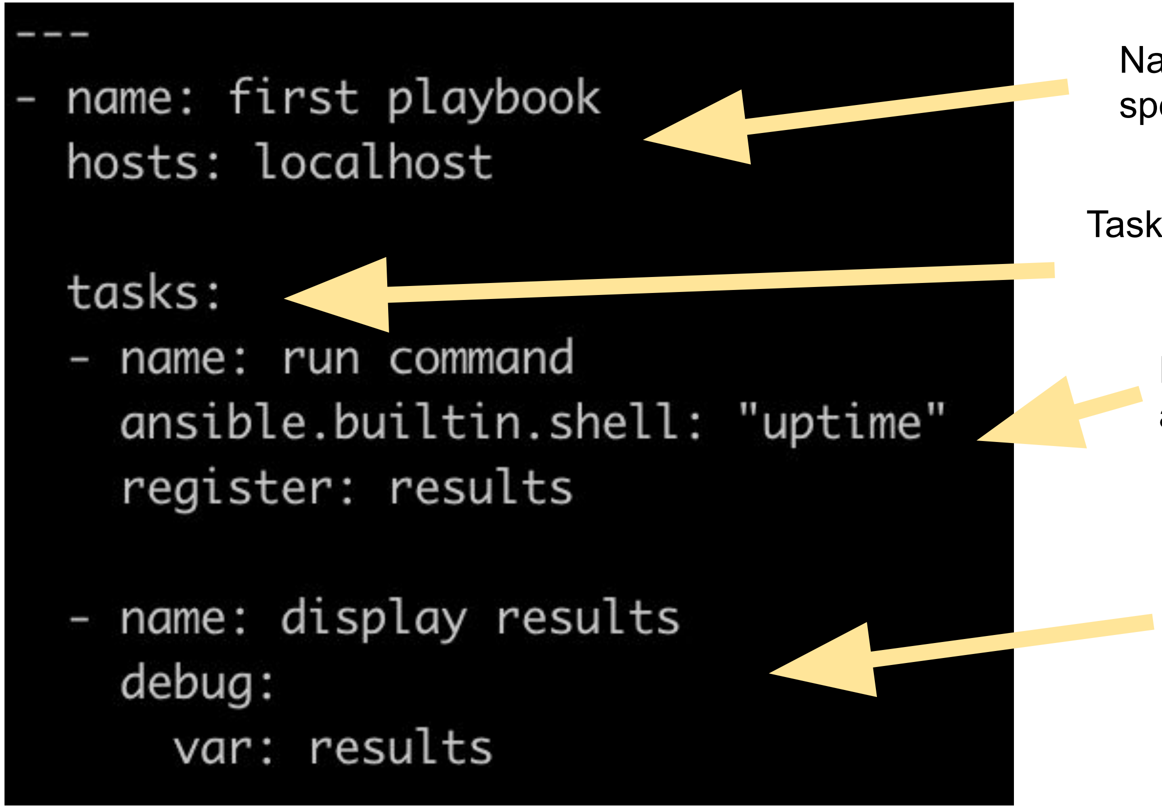
```
atlanta:
  hosts:
    host1:
      http_port: 80
      maxRequestsPerChild: 808
    host2:
      http_port: 303
      maxRequestsPerChild: 909
```

Here is my inventory



```
---
all:
  children:
    cisco:
      hosts:
        cisco_1:
          ansible_host: 2001:db8:16:1::2
        cisco_2:
          ansible_host: 2001:db8:16:1::3
      vars:
        ansible_user: clab
        ansible_ssh_pass: clab@123
    junos:
      vars:
        ansible_user: root
        ansible_ssh_pass: clab123
      hosts:
        juniper:
          ansible_host: 2001:db8:16:1::4
```

My First Playbook



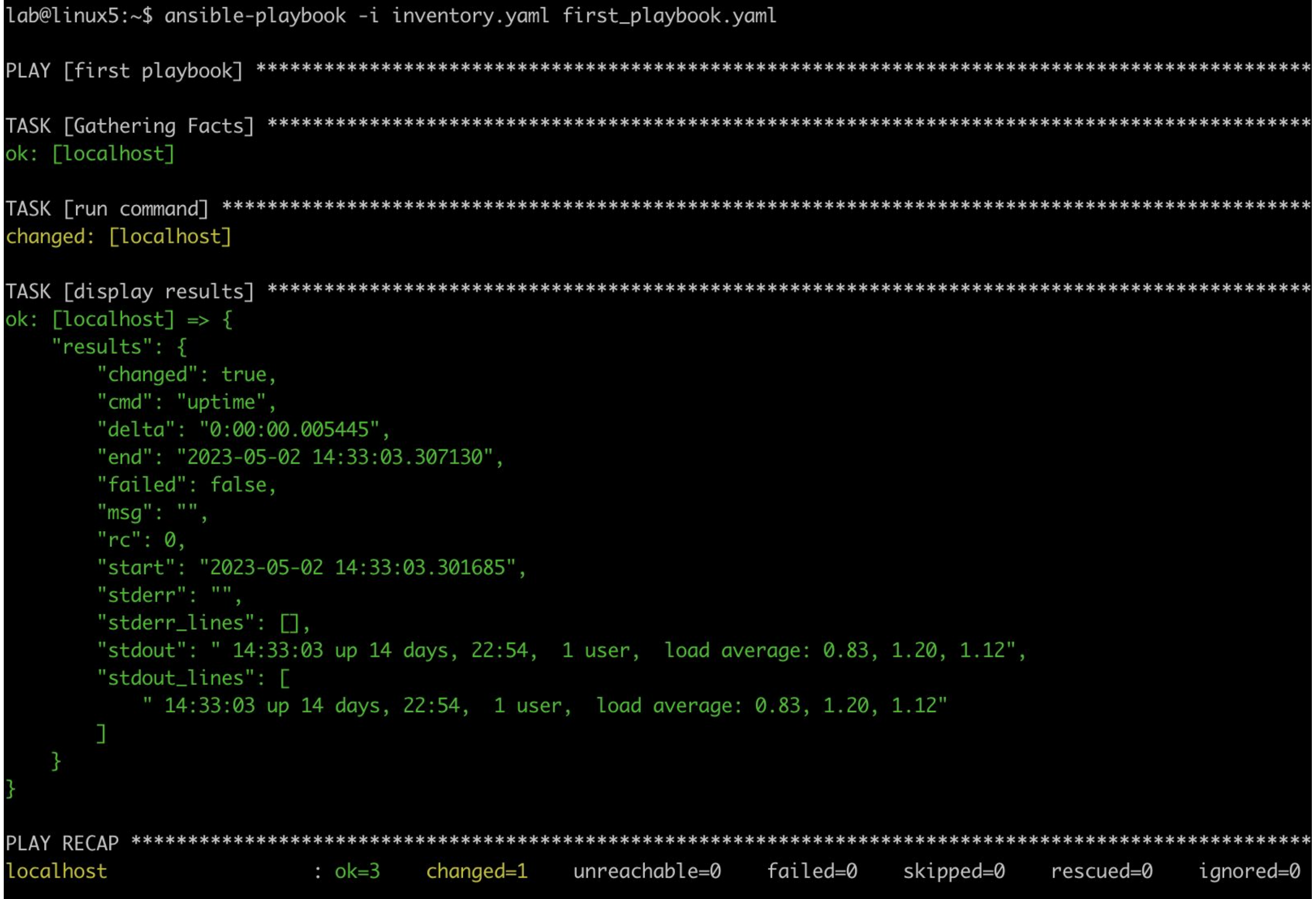
```
---
- name: first playbook
  hosts: localhost

  tasks:
  - name: run command
    ansible.builtin.shell: "uptime"
    register: results

  - name: display results
    debug:
      var: results
```

Name of the playbook (optional) and hosts specifies the hosts to execute it on

Tasks is an array of the tasks

First task - run a shell command “uptime” and store the response as “results”

Display the results using the “debug” task output



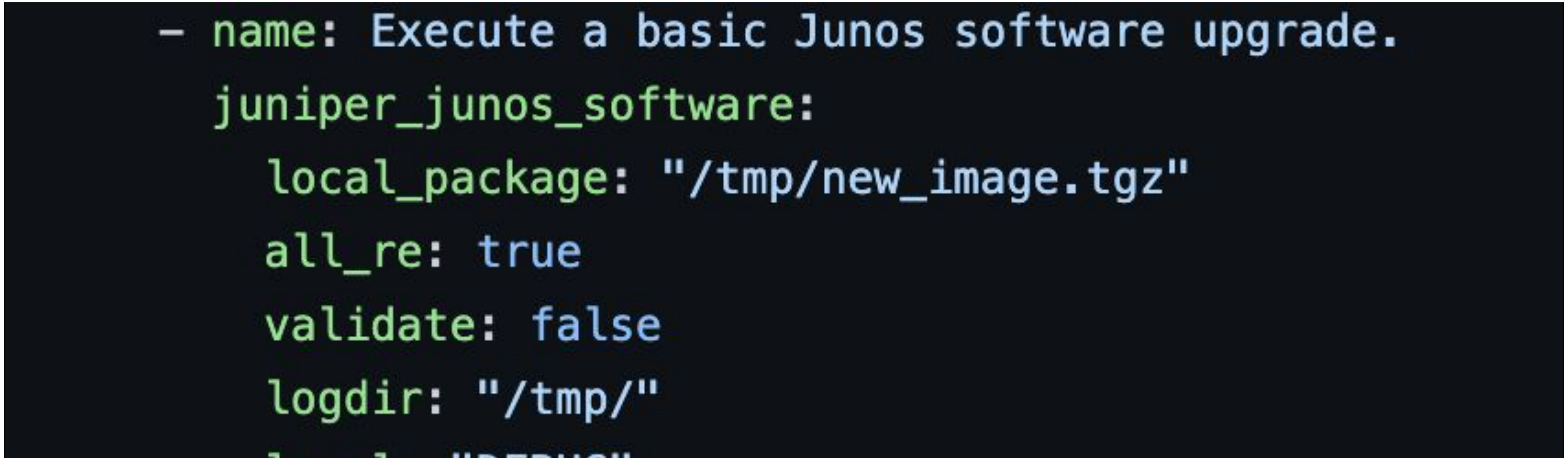
```
lab@linux5:~$ ansible-playbook -i inventory.yaml first_playbook.yaml

PLAY [first playbook] *********************************************************************************************

TASK [Gathering Facts] ********************************************************************************************
ok: [localhost]

TASK [run command] ************************************************************************************************
changed: [localhost]

TASK [display results] ********************************************************************************************
ok: [localhost] => {
    "results": {
        "changed": true,
        "cmd": "uptime",
        "delta": "0:00:00.005445",
        "end": "2023-05-02 14:33:03.307130",
        "failed": false,
        "msg": "",
        "rc": 0,
        "start": "2023-05-02 14:33:03.301685",
        "stderr": "",
        "stderr_lines": [],
        "stdout": " 14:33:03 up 14 days, 22:54,  1 user,  load average: 0.83, 1.20, 1.12",
        "stdout_lines": [
            " 14:33:03 up 14 days, 22:54,  1 user,  load average: 0.83, 1.20, 1.12"
        ]
    }
}

PLAY RECAP ********************************************************************************************************
localhost                  : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Roles/Collections for common devices

use Ansible Galaxy to install some roles

- (Role) ansible-galaxy install Juniper.junos
- (Collection) ansible-galaxy collection install cisco.iosxr

We can then use these Roles/Collections to interact with the devices



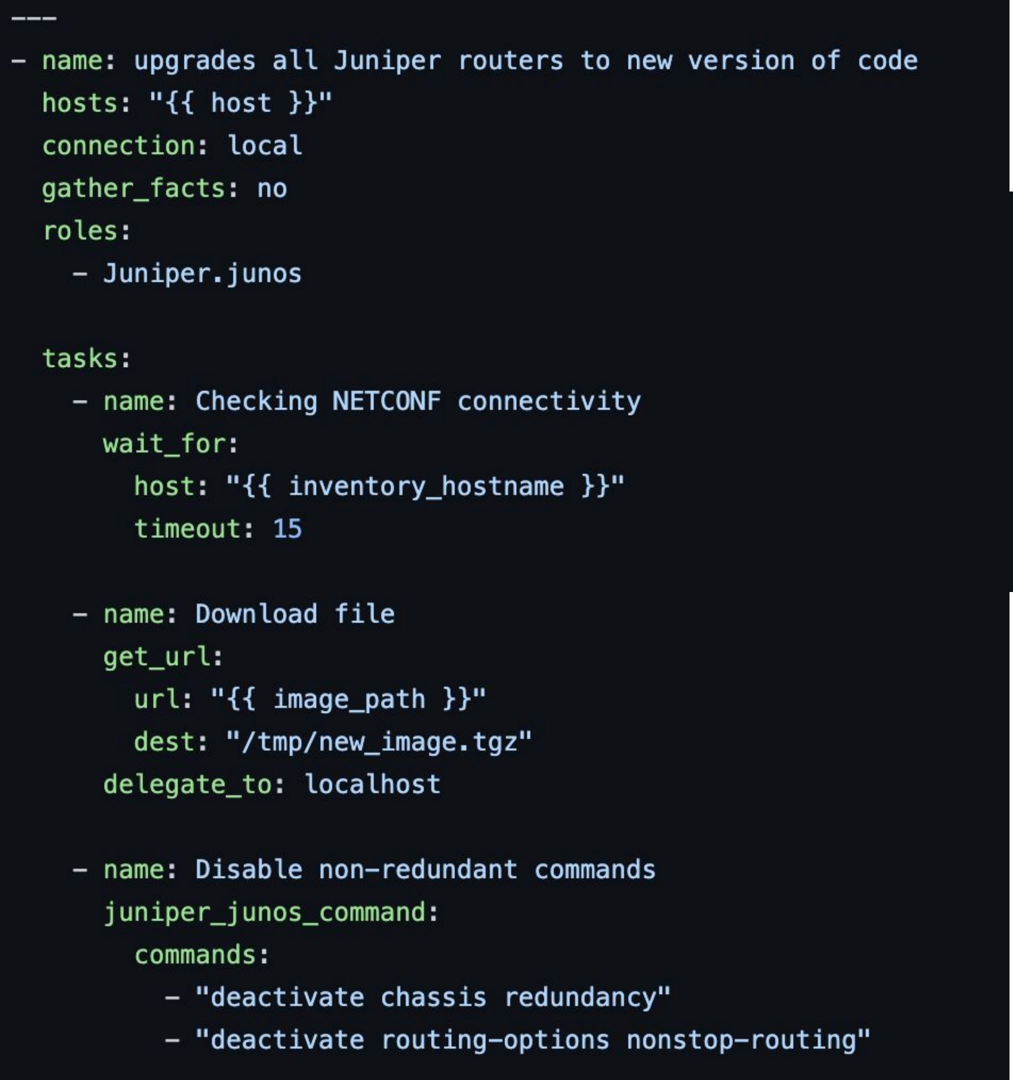
```
    - name: Execute a basic Junos software upgrade.
      juniper_junos_software:
        local_package: "/tmp/new_image.tgz"
        all_re: true
        validate: false
        logdir: "/tmp/"
```

Lets do something more interesting

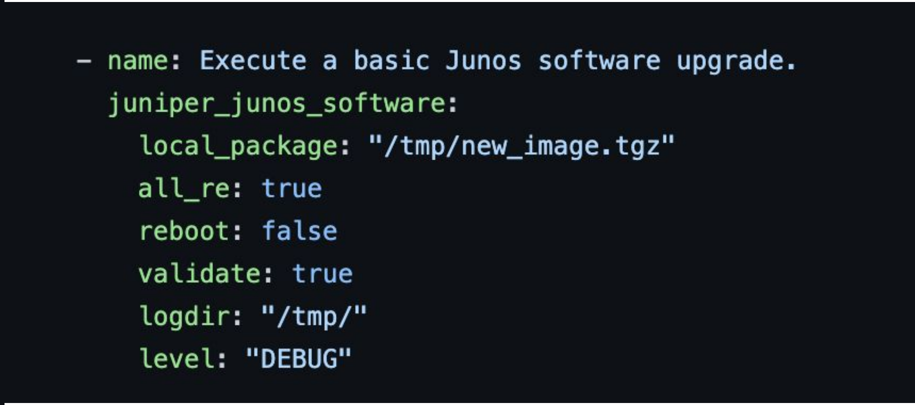
Upgrade a Juniper with redundant REs and do it as hitless as possible

Steps:

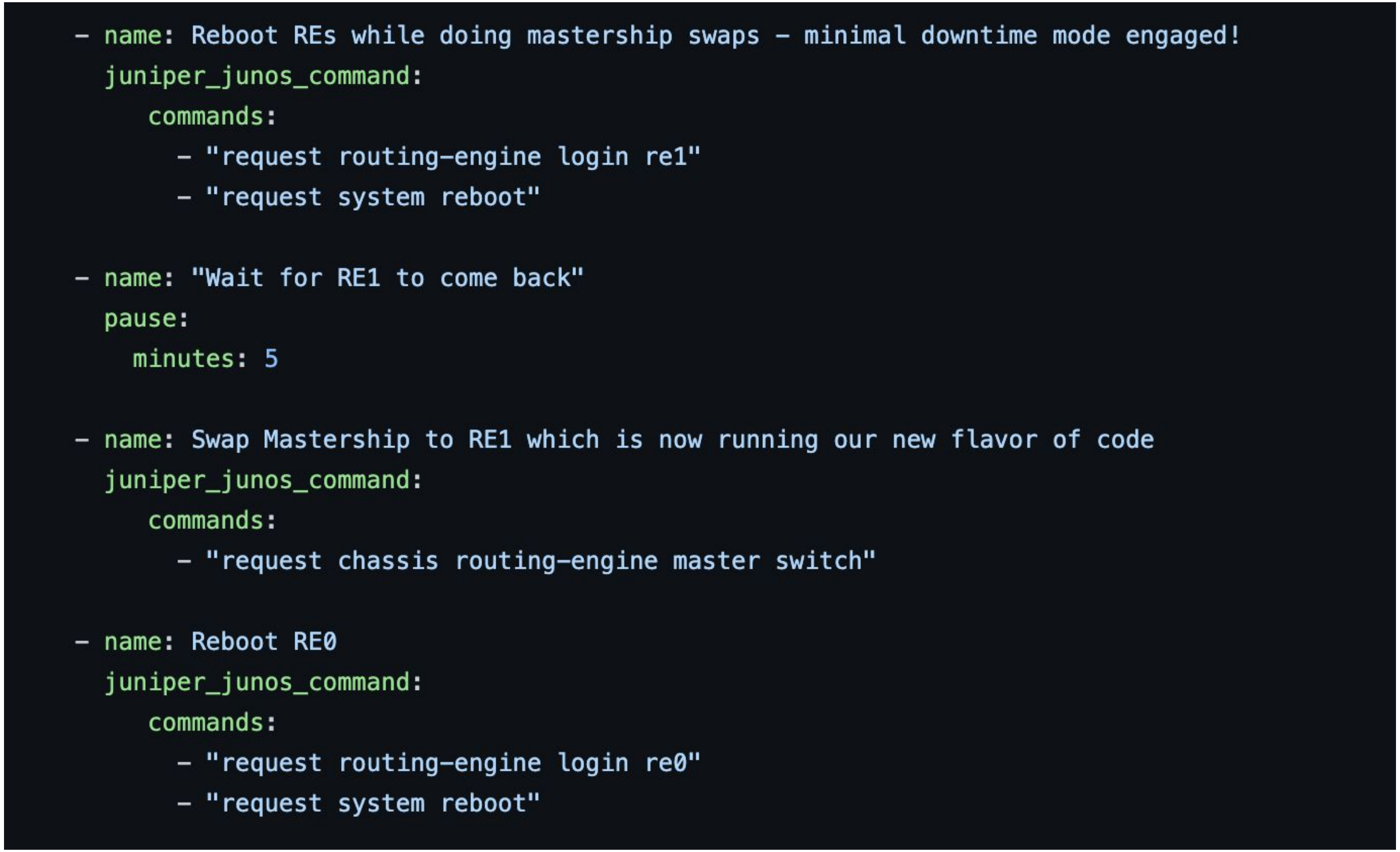
- Download the new code version
- Disable Chassis Redundancy
- Upgrade RE1
- Wait until RE1 comes back up
- Swap Mastership
- Upgrade RE0
- Wait until RE0 Comes back up
- Swap back to RE0
- Re-enable Chassis redundancy

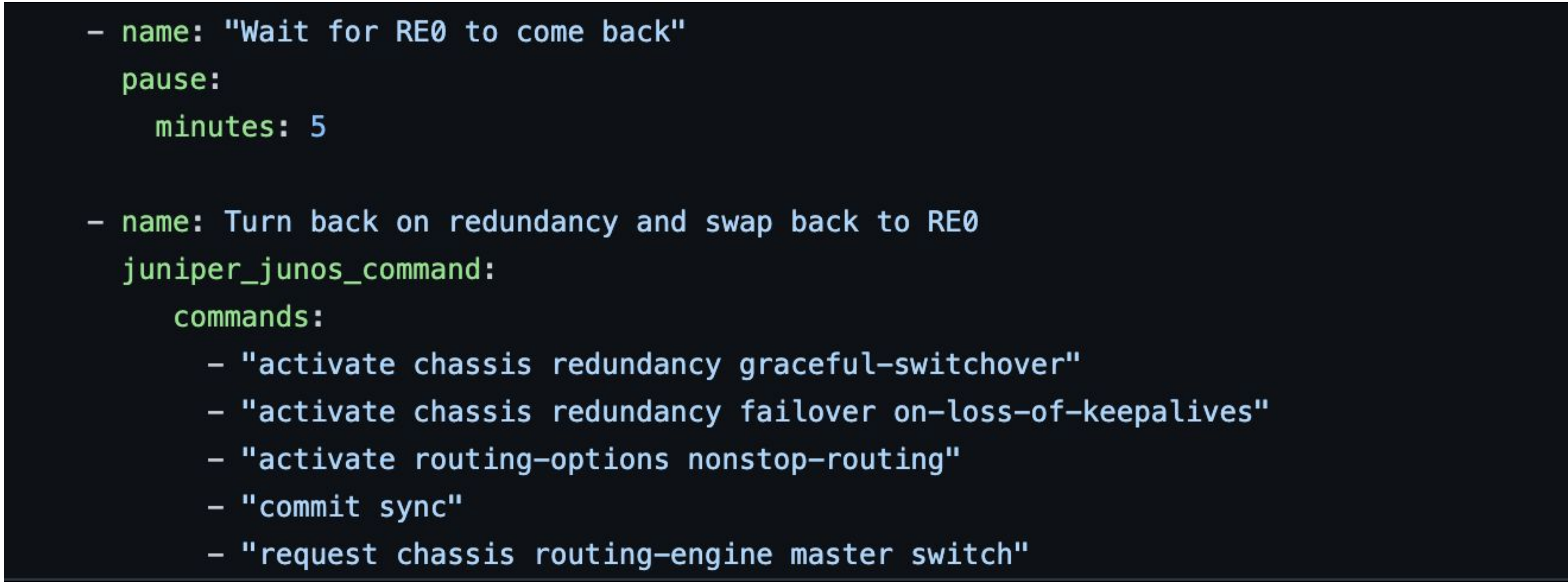


```yaml
---
- name: upgrades all Juniper routers to new version of code
  hosts: "{{ host }}"
  connection: local
  gather_facts: no
  roles:
    - Juniper.junos

  tasks:
    - name: Checking NETCONF connectivity
      wait_for:
        host: "{{ inventory_hostname }}"
        timeout: 15

    - name: Download file
      get_url:
        url: "{{ image_path }}"
        dest: "/tmp/new_image.tgz"
      delegate_to: localhost

    - name: Disable non-redundant commands
      juniper_junos_command:
        commands:
          - "deactivate chassis redundancy"
          - "deactivate routing-options nonstop-routing"
```



```yaml
- name: Execute a basic Junos software upgrade.
  juniper_junos_software:
    local_package: "/tmp/new_image.tgz"
    all_re: true
    reboot: false
    validate: true
    logdir: "/tmp/"
    level: "DEBUG"
```


```yaml
- name: Reboot REs while doing mastership swaps - minimal downtime mode engaged!
  juniper_junos_command:
     commands:
       - "request routing-engine login re1"
       - "request system reboot"

- name: "Wait for RE1 to come back"
  pause:
    minutes: 5

- name: Swap Mastership to RE1 which is now running our new flavor of code
  juniper_junos_command:
     commands:
       - "request chassis routing-engine master switch"

- name: Reboot RE0
  juniper_junos_command:
     commands:
       - "request routing-engine login re0"
       - "request system reboot"
```



```yaml
  - name: "Wait for RE0 to come back"
    pause:
      minutes: 5

  - name: Turn back on redundancy and swap back to RE0
    juniper_junos_command:
       commands:
         - "activate chassis redundancy graceful-switchover"
         - "activate chassis redundancy failover on-loss-of-keepalives"
         - "activate routing-options nonstop-routing"
         - "commit sync"
         - "request chassis routing-engine master switch"
```

Advanced Ansible

- AWX - webUI for your playbooks / workflows
  - Store credentials
  - REST API
- Vaults
  - Encrypted storage of credentials
- Jinja2
  - Templates with REST integrations