Python

here, there, and everywhere

Where should you run Python

- It's best to install it on your own computer
  - You'll have more control, can run Jupyter notebooks and learn more about it
- You can also run it on UMBC's gl unix system
  - Installing new packages and modules as needed
- You can also use remote notebook servers
  - At UMBC, Google Colab, Binder, …
- And use other remote options
  - Like Repl.it
- We'll give some details here

Installing Python 3

- Assuming you have a computer running Unix, OS X, or Windows install the newest version of Python (e.g., 3.8) from python.org
- You can install it on iOS and Android, too
- Here's a tutorial if you need help
- Running it on your own computer makes it easier to install packages, IDEs, and use notebooks
- And will give you more experience

IDE or not?

- Python's an interpreted language so it comes with a read-eval-print-loop environment
- I'll admit to mostly using emacs to edit code in one window and the Python REPL in another
  - Emacs comes with a python-mode that's invoked when you edit a file ending in .py
- But you may prefer a Python IDE
  - Python comes with a simple one, IDLE
  - PyCharm is very popular and good
- Here's a guide to Python editors and IDEs

Loading code into Python

- Load file foo.py from the current directory

  >>> import foo

- Each expression in the file is evaluated, but the value is not printed (i.e., it's a read-eval loop)
- Python will also search directories in the environment variable PYTHONPATH

  ~> echo $PYTHONPATH

  /afs/umbc.edu/users/f/i/finin/pub/ai/aima-python:
  /afs/umbc.edu/users/f/i/finin/pub/ai/:

- And load installed **packages**, which can be simple or complex with sub-packages

  >>> import tensorflow

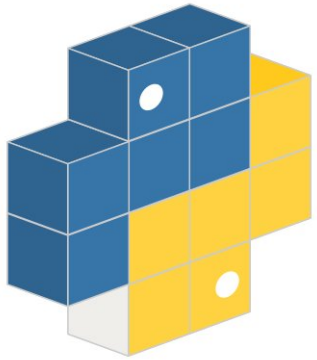
Import variations

File: example.py

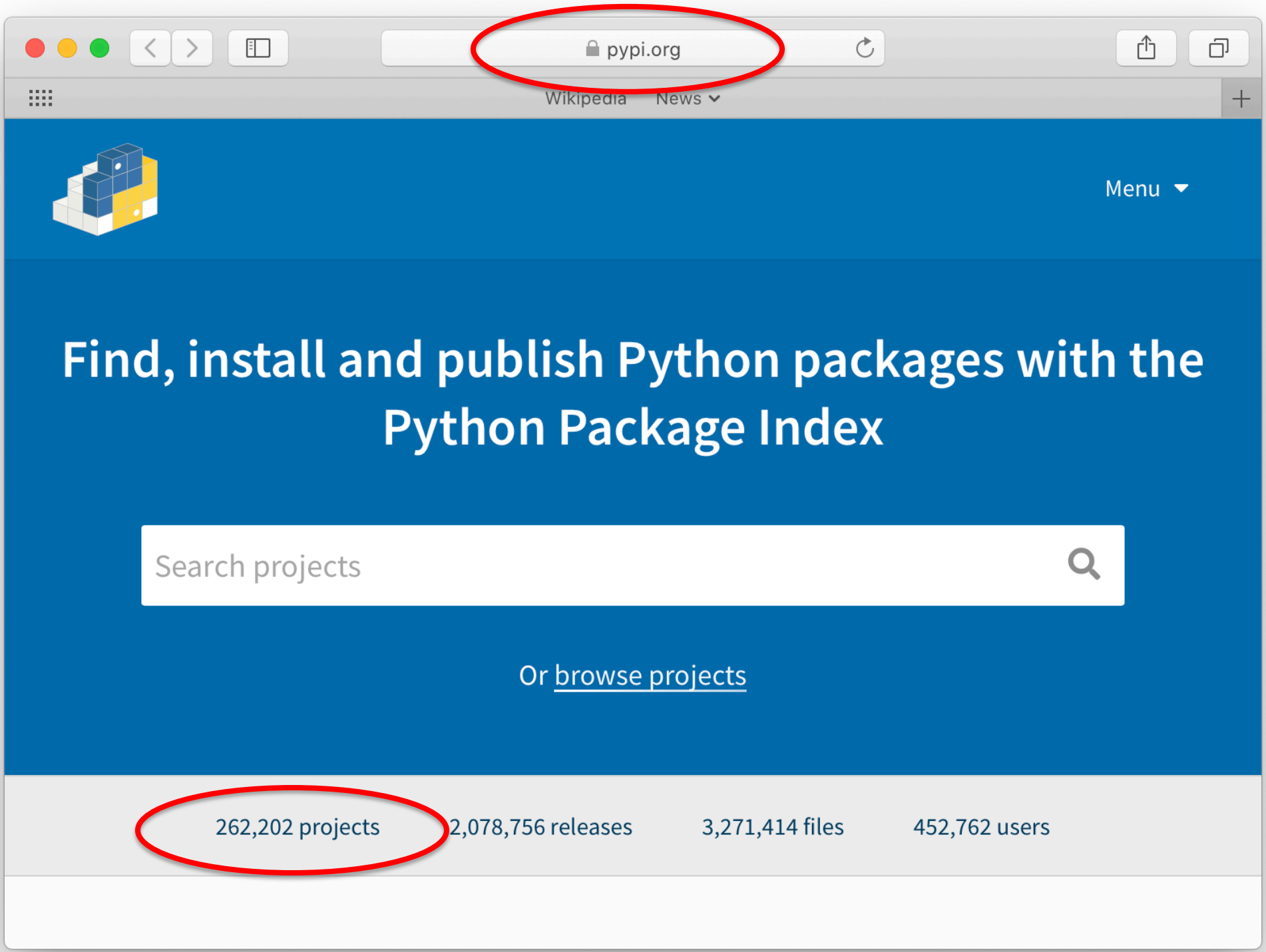
```
name = "Bob"

def hello():
  print("hello, I'm", name)

def bye():
    print("goodbye")
```

- How you import effects how you access its named functions and variables
  - import example
    `example.hello()`
  - import example as ex
    `ex.hello()`
  - from example import hello
    `hello()`
  - from foo import *
    `hello()`
- Python only imports a file once, subsequent imports do nothing

Installing software packages



- Python's got a huge user base and is the most popular language for AI today
  - So there are many great SW packages to install
- Your Python probably came with pip, the standard python package install program
- Search for packages on pypi.org and install/update them with the pip command


pypi.org
Wikipedia
News
Menu
Find, install and publish Python packages with the Python Package Index
Search projects
Or browse projects
262,202 projects
2,078,756 releases
3,271,414 files
452,762 users

Using pip

- For HW3 you'll need the python package python-constraint
- Install it on your own Linux or Mac system

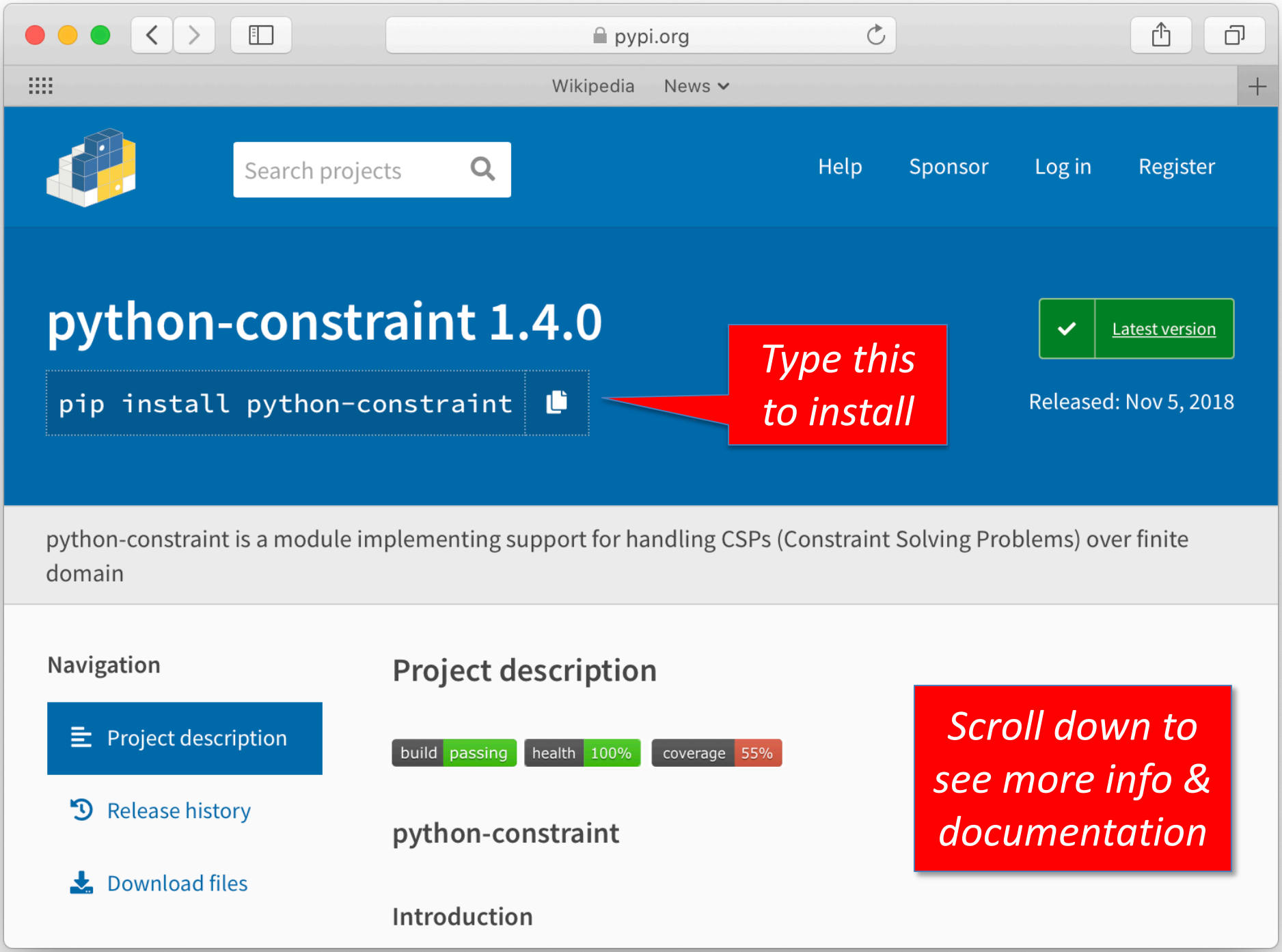
  pip install python-constraint

- If your acct is not an admin:

  sudo pip install python-constraint

- Install without sudo privileges (e.g., on gl)

  pip install python-constraint --user


pypi.org
Wikipedia
News
Search projects
Help
Sponsor
Log in
Register
python-constraint 1.4.0
pip install python-constraint
Type this to install
Latest version
Released: Nov 5, 2018
python-constraint is a module implementing support for handling CSPs (Constraint Solving Problems) over finite domain
Navigation
Project description
Release history
Download files
Project description
build passing
health 100%
coverage 55%
python-constraint
Introduction
Scroll down to see more info & documentation

Working on gl

- On gl, you tell Python to look in the directory we've set up for AIMA python code
- Or set up your own directory (e.g., ~/mypy) in which you install new packages
- For either, you must first add appropriate directories to your PYTHONPATH environment variable
  - Do this by modifying your shell initialization file (e.g., ~/.cshrc or ~/.bashrc)

Python and PYTHONPATH

- Python's import command looks for modules to load in a list of places
- sys.path is the list, with ‘ ‘ as the current directory

```
>>> import sys
>>> sys.path
[‘ ‘, '/usr/lib64/python26.zip', …]
```

- On Unix, when python starts, it prepends directories on your PYTHONPATH environment variable
- Add new directories for python to search by setting PYTHONPATH in the init file used by your shell
- The Unix command *echo $SHELL* shows what shell you are using

virtualenv

Notebooks

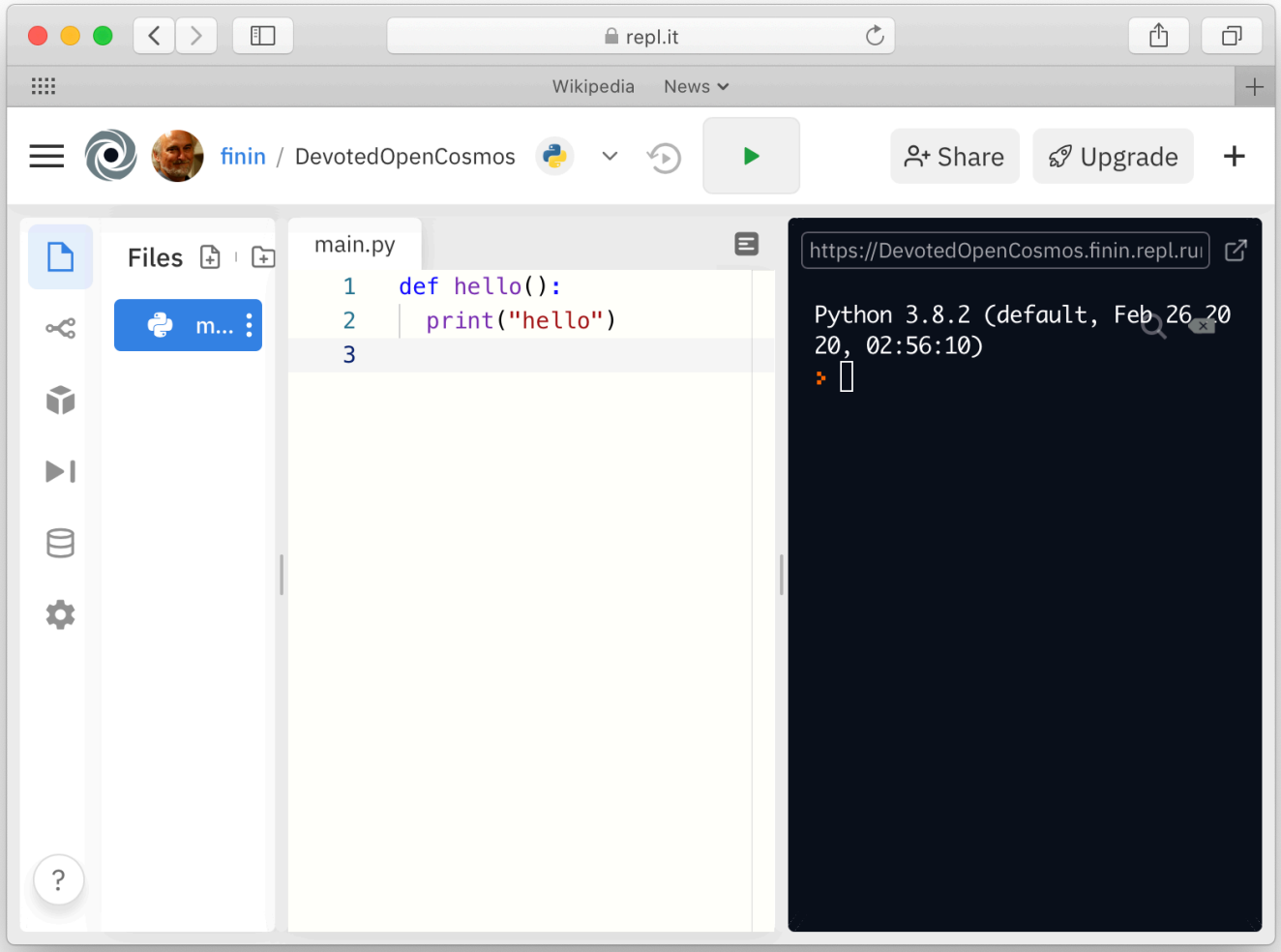
On your own computer

Using a notebook server

Using Binder

Using Google Colab

Using repl.it



Using repl.it

I’m unfamiliar with this, but it looks interesting

- Web-based IDE startup for 60+ languages
- Including Python
- Free for public and limited use
- Good for trying new languages?
- Supports teams