

>>> network `.toCode()`

Network to Code

Development Environments

Agenda

- Introduction to DevEnvs
- Code Editors & IDEs
- Other Tools: Terminals, Git
- Windows Subsystem for Linux
- Vagrant and Virtual Machines
- Python Tooling
- Docker
- Hands-on Labs

What is a Development Environment

A DevEnv is a collection of tools and machines on which you (the developer) work on your projects. It may contain, but not limited to:

- Code Editors and IDEs (Integrated Development Environments)
- One or more programming languages and their tooling (interpreters, compilers)
 - Third party libraries for these languages
- Hardware - your PC, network devices
 - Often in virtual format as well - VMs of various platforms
- Supporting software and services
 - Centralized code repositories
 - “Source of Truth” for input data (more commonly known as databases)
- Workflows, Standards, Processes

Code Editors & IDEs

These (opinionated) tools are your bread-and-butter

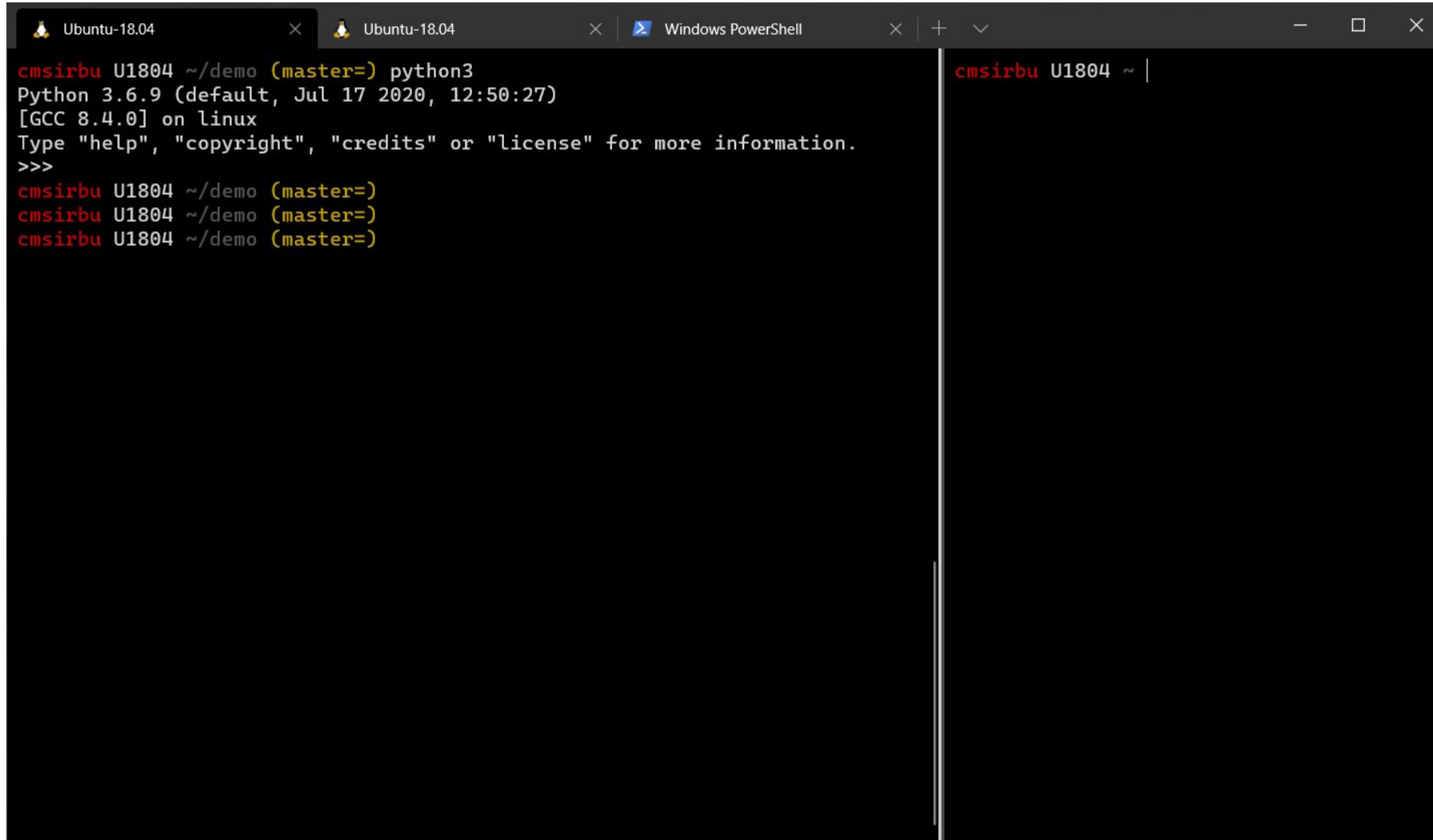
- A lot of choice, both free and paid software
- Beginners: start with popular options, expand as you gain knowledge
 - Watch how others work, you can always learn new things
- VSCode, SublimeText3, PyCharm, Atom, VIM
 - Research plugins and configuration
 - Learn shortcuts!

Other Tools: Git

Git is a distributed version-control system.

- It's massively popular, thanks to wide adoption through Internet based centralized platforms like **GitHub** and **GitLab**.
- But there are others: CVS, Mercurial, Subversion, Bazaar (Open source)
- CLI-driven - so **learn the CLI!** Later on, add GUI tools for complex projects.
- Get comfy with git commands - very extensible and configurable. (Shell) aliases are your best friends.
 - alias gc='git commit'
 - alias gs='git status --short'

Other Tools: Terminals



The image shows a terminal window with three tabs: 'Ubuntu-18.04', 'Ubuntu-18.04', and 'Windows PowerShell'. The active tab is the first 'Ubuntu-18.04' terminal. It displays a shell session where the user has run 'python3'. The output shows 'Python 3.6.9 (default, Jul 17 2020, 12:50:27)' and '[GCC 8.4.0] on linux'. The user is prompted to type 'help', 'copyright', 'credits', or 'license' for more information. The prompt is '>>>'. Below this, the user has entered 'cmsirbu U1804 ~/demo (master=)' three times, which appears to be a command or a prompt that is not recognized by the Python shell.

```
cmsirbu U1804 ~/demo (master=) python3
Python 3.6.9 (default, Jul 17 2020, 12:50:27)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
cmsirbu U1804 ~/demo (master=)
cmsirbu U1804 ~/demo (master=)
cmsirbu U1804 ~/demo (master=)
```

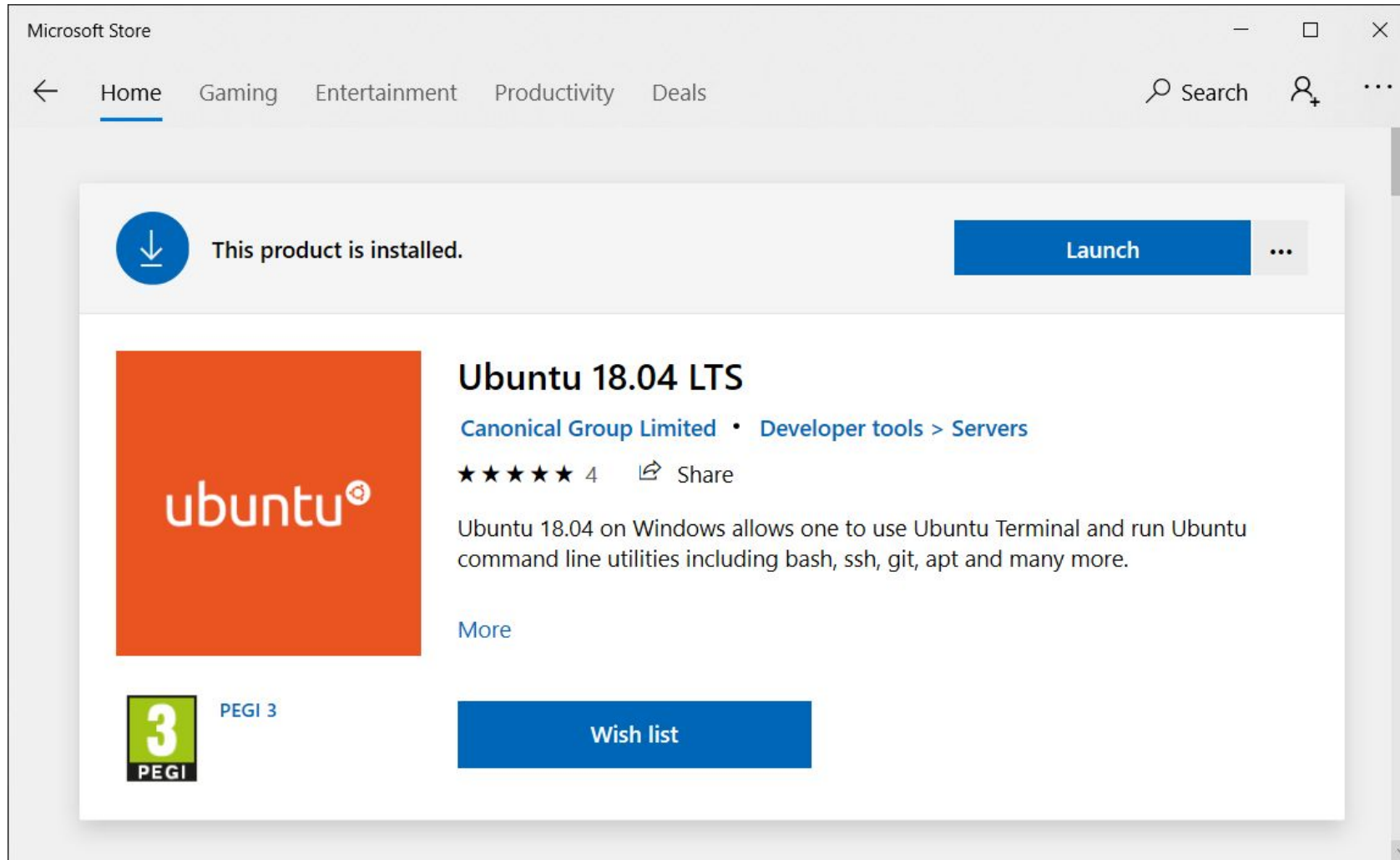
Windows Subsystem for Linux

Git is a distributed version-control system.

- “The Windows Subsystem for Linux lets developers run a GNU/Linux environment -- including most command-line tools, utilities, and applications -- directly on Windows, unmodified, without the overhead of a traditional virtual machine or dualboot setup.”
- WSL1 from Windows 10 version 1709 (should be everywhere)
- WSL2 from Windows 10 version 2004 (very recent)
 - Ships a Linux kernel, makes performance improvements
 - You can run Docker more “natively”!

<https://docs.microsoft.com/en-us/windows/wsl/install-win10>

Windows Subsystem for Linux



>>> network .toCode()

DEMO TIME!

(VSCode, WSL)

Vagrant and Virtual Machines

Vagrant is an opensource tool for building and maintaining portable virtual software development environments.

- Made by HashiCorp (they make great tools!) <https://www.vagrantup.com/>
- Works with VirtualBox, Hyper-V, KVM, VMWare
- Many “boxes” publicly available <https://app.vagrantup.com/boxes/search>
- There can (usually) be only one hypervisor!
 - On Win10 - Hyper-V is used for WSL
 - VMWare 15.5.5+ allegedly co-exists with Hyper-V

```
Administrator: Windows PowerS...
PS C:\stuff\vm\ubuntu> vagrant up
Bringing machine 'default' up with 'hyperv' provider...
==> default: Verifying Hyper-V is enabled...
==> default: Verifying Hyper-V is accessible...
==> default: Box 'generic/ubuntu2004' could not be found. Attempting to find and install...
    default: Box Provider: hyperv
==> default: Loading metadata for box 'generic/ubuntu2004'
    default: URL: https://vagrantcloud.com/generic/ubuntu2004
==> default: Adding box 'generic/ubuntu2004' (v3.0.20) for provider: hyperv
    default: Downloading: https://vagrantcloud.com/generic/boxes/ubuntu2004/versions/3.0.20/providers/hyperv.box
Download redirected to host: vagrantcloud-files-production.s3.amazonaws.com
    default:
    default: Calculating and comparing box checksum...
==> default: Successfully added box 'generic/ubuntu2004' (v3.0.20) for 'hyperv'!
==> default: Importing a Hyper-V instance
    default: Creating and registering the VM...
    default: Successfully imported VM
    default: Configuring the VM...
==> default: Starting the machine...
==> default: Waiting for the machine to report its IP address...
    default: Timeout: 120 seconds
    default: IP: 172.20.191.155
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 172.20.191.155:22
    default: SSH username: vagrant
    default: SSH auth method: private key
    default:
    default: Vagrant insecure key detected. Vagrant will automatically replace
    default: this with a newly generated keypair for better security.
    default:
    default: Inserting generated public key within guest...
    default: Removing insecure key from the guest if it's present...
vagrant@ubuntu2004:~$ ls
vagrant@ubuntu2004:~$
vagrant@ubuntu2004:~$
vagrant@ubuntu2004:~$ logout
Connection to 172.20.191.155 closed.
```

File Edit Selection View Go Run Terminal Help vagrant [SSH: vagrant-ubuntu2004] - Visual Studio Code

EXPLORER

> OPEN EDITORS

✓ VAGRANT [SSH: VAGRANT-UBUNTU2004]

- > .cache
- > .ssh
- > .vscode-server
- .bash_history
- .bash_logout
- .bashrc
- .profile
- .python_history
- .wget-hsts

PROBLEMS OUTPUT TERMINAL ... 1: bash

```
vagrant@ubuntu2004:~$ ls
vagrant@ubuntu2004:~$ ll
total 40
drwxr-xr-x 5 vagrant vagrant 4096 Aug  5 16:43 ./
drwxr-xr-x 3 root    root    4096 Jul 29 23:50 ../
-rw-r--r-- 1 vagrant vagrant   60 Aug  5 16:14 .bash_history
-rw-r--r-- 1 vagrant vagrant  220 Jul 29 23:50 .bash_logout
-rw-r--r-- 1 vagrant vagrant 3771 Jul 29 23:50 .bashrc
drwxr-xr-x 2 vagrant vagrant 4096 Jul 29 23:58 .cache/
-rw-r--r-- 1 vagrant vagrant  807 Jul 29 23:50 .profile
-rw-r--r-- 1 vagrant vagrant    0 Aug  5 14:26 .python_history
drwxr-xr-x 2 vagrant vagrant 4096 Aug  5 14:22 .ssh/
drwxrwxr-x 5 vagrant vagrant 4096 Aug  5 16:43 .vscode-server/
-rw-rw-r-- 1 vagrant vagrant  183 Aug  5 16:43 .wget-hsts
vagrant@ubuntu2004:~$
```

> OUTLINE

> TIMELINE

SSH: vagrant-ubuntu2004 0 0

>>> network .toCode()

DEMO TIME!

(VSCode, Vagrant)

Python Tooling

Python has a rich ecosystem of tools for development, testing, packaging, and distribution.

- Virtual Environments - start with the basics “python3 -m venv”
 - Other tools: virtualenv (the original), pipenv, poetry, tox, virtualenvwrapper
- Be careful where you install Python packages
 - Operating system level
 - User level
 - VirtualEnv

Developer Tooling

GNU Make is a build automation tool on Unix & Unix-like systems

- Originally designed to build (compile) files from source code
- Nowadays it's also used as a layer for simple task automation
- Uses a Makefile to describe rules and targets

```
.PHONY: black pylint
```

```
black:  
    black --check --diff get.py
```

```
pylint:  
    pylint get.py
```

Developer Tooling

Invoke is Python task execution tool & library

- It draws inspiration from **make** and **rake** (ruby)
- Defines tasks in Python - by default in a **tasks.py** file

```
from invoke import task

@task
def black(context):
    context.run("black --check --diff get.py")

@task
def pylint(context):
    context.run("pylint get.py")
```



```
>>> network.toCode()
```

DEMO TIME!

(Python VirtualEnvs, Make, Invoke)

Docker

Docker is a set of tools that use OS-level virtualization to package software.

- **Docker Containers** depend on Linux Kernel features for isolation and resource management (cgroups and namespaces)
 - Containers are largely synonymous today with *Linux* Containers
 - Many container orchestrators and runtimes - Kubernetes, OpenShift, LXC/LXD, Rkt, Podman
- **Goal today:** learn the basics of using docker commands
 - Downloading and running someone else's containerized tool
 - Build your own containers with your code
 - Build your own containers with your tools

>>> network `.toCode()`

DEMO TIME!

(Docker)

>>> network .toCode()

LAB TIME!

(Your turn)