A Python you can reinstall without internet

Why the system Python is the wrong place to work, what a virtual environment actually is, and how to carry a working install to a field laptop that has never seen a package index.

Pierre Robentz Cassion

Lesson 1 of 8

Python for Programme Data — An environment that will still run

What this lesson covers

- The failure this lesson prevents
- Do not work in the system Python
- What a virtual environment is
- uv, and why it matters more here than elsewhere
- Installing where there is no internet
- Pinning the Python version too
- Verifying the environment before trusting it
- What to hand to a colleague
- What comes next

The failure this lesson prevents

- An analysis runs on your laptop.

Do not work in the system Python — Example

```
error: externally-managed-environment

x This environment is externally managed
+-> To install Python packages system-wide, try apt install
    python3-xyz, where xyz is the package you are trying to
    install.
```

Do not work in the system Python

- That error is correct and should not be worked around — There is a flag that overrides it

What a virtual environment is — Shell

```
python3 -m venv .venv
source .venv/bin/activate       # Windows: .venv\Scripts\activate
python -m pip install pandas
```

What a virtual environment is

- **Per project, not per machine.** Two analyses needing different pandas versions coexist without either knowing about. . .
- **Disposable.** If an environment gets into a bad state, delete the directory and rebuild it. Nothing of value lives. . .

uv, and why it matters more here than elsewhere — Shell

```
# Install uv itself, once, on a machine that does have internet.
curl -LsSf https://astral.sh/uv/install.sh | sh

uv init muac-analysis
cd muac-analysis
uv add pandas openpyxl
uv run python -c "import pandas; print(pandas.__version__)"
```

uv, and why it matters more here than elsewhere — Shell

```
uv sync                    # reads uv.lock, installs exactly those versions
```

Installing where there is no internet — Shell

```
mkdir wheels
uv pip download pandas openpyxl --dest wheels
```

Installing where there is no internet — Shell

```
uv venv
uv pip install --no-index --find-links wheels pandas openpyxl
```

Installing where there is no internet

Wheels are platform-specific. A wheel downloaded on macOS will not install on a Windows laptop, and one built for Python 3.12 will not install into 3.11. Download on a machine matching the target, or pass `-python-platform` and `-python-version` explicitly.

Pinning the Python version too — Example

```
# pyproject.toml
[project]
name = "muac-analysis"
requires-python = ">=3.11"
dependencies = ["pandas>=2.2", "openpyxl>=3.1"]
```

Pinning the Python version too — Shell

```
uv python install 3.12
uv python pin 3.12
```

Verifying the environment before trusting it — In Python

```python
import sys
import pandas as pd

print(sys.executable)          # should be inside .venv, not /usr/bin
print(sys.version)
print(pd.__version__)
```

Verifying the environment before trusting it — In Python

```
import sys

assert sys.version_info >= (3, 11), f"needs Python 3.11+, running {sys.version}"
```

What to hand to a colleague

Commit	Do not commit
`pyproject.toml`	`.venv/`
`uv.lock`	`wheels/`
`README.md` with the two commands	`__pycache__/`

What to hand to a colleague — Example

```
uv sync
uv run python scripts/indicator_table.py --input data/raw/muac.csv
```

What comes next

- The environment is reproducible.

Read the full lesson, with runnable code

https://data-analysis.cassion.dev/courses/python-for-programme-data/python-01-environment