ALLEN INSTITUTE *for*
BRAIN SCIENCE

uv

Ben Pedigo
(he/him)
Scientist I
Allen Institute for Brain Science
ben.pedigo@alleninstitute.org

Features

- Fast `pip` alternative with drop-in syntax
- Can get Python versions for you as needed
- Allows for inline script dependencies (there is a PEP for this) `uv run`
- Will install and run command line tools for you `uvx black ...`
- Can handle some dev tasks like building a package, updating dependencies, etc.

Concepts

venvs

- Often created dynamically since they are so quick to install
- Commands like `uv run` will look for a nearby .venv directory and use it if it exists, creating it otherwise

lockfile

https://docs.astral.sh/uv/concepts/projects/layout/#the-lockfile

> uv.lock is a universal or cross-platform lockfile that captures the packages that would be installed across all possible Python markers such as operating system, architecture, and Python version.

> Unlike the pyproject.toml, which is used to specify the broad requirements of your project, the lockfile contains the exact resolved versions that are installed in the project environment. This file should be checked into version control, allowing for consistent and reproducible installations across machines.

Random tips

Install from requirements, not looking at lock file

```
uv pip install --requirements pyproject.toml
```

Avoid building anything from source

`--no-build`

Just see what would be installed

```
--dry-run
```

Use the oldest version of packages

```
--resolution lowest
```

Upgrading

> If resolution output file exists, i.e. a uv lockfile (uv.lock) or a requirements output file (requirements.txt), uv will prefer the dependency versions listed there. Similarly, if installing a package into a virtual environment, uv will prefer the already installed version if present. This means that locked or installed versions will not change unless an incompatible version is requested or an upgrade is explicitly requested with `--upgrade`.

`--upgrade` is the same as deleting the lockfile and starting over(?)

See what would get installed for another platform

```
uv pip install ... --dry-run --python-platform windows
```

View dependency tree

```
uv tree
```

For a package, see who has opinions about it

```
uv tree --invert --package numpy
```