Unlocking Hidden Power of Conda with Pixi

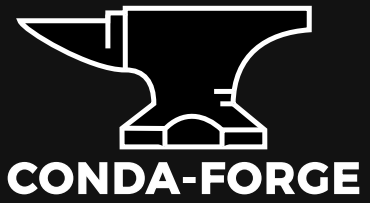
By Prefix.dev GmbH

About Us

- 📦 Focused on solving Package Management
- 🚀 Startup, 2.5 years old
- 🌐 Fully remote, spread around Europe
- 🐍 Dedicated to make the conda ecosystem amazing!
- 🦀 Everything in Rust

What is Conda?

- 📦 Package ecosystem:
  - Cross-platform
  - Cross-language
- 🔬 Commonly used for scientific Python
- 🌐 Decentralized channels like:
  - conda-forge: Most popular channel
  - bioconda: Seperatly managed channel for bioinformatics
  - RoboStack: Automated channel for robotics
- Gifted to the community by Anaconda, Inc.
- Prefix.dev GmbH: revolutionizing the conda ecosystem



Installing NumPy via Pip

From the NumPy contributor docs:

- Install NumPy as a user:

```
pip install numpy
```

- Install NumPy as a developer:

```
# Debian
sudo apt build-dep numpy
# Fedora
sudo dnf builddep numpy
# Arch
sudo pacman -S gcc-fortran openblas pkgconf
# macOS
brew install openblas pkg-config gfortran
```

Finally

```
pip install . --no-build-isolation
```

Installing NumPy via Conda

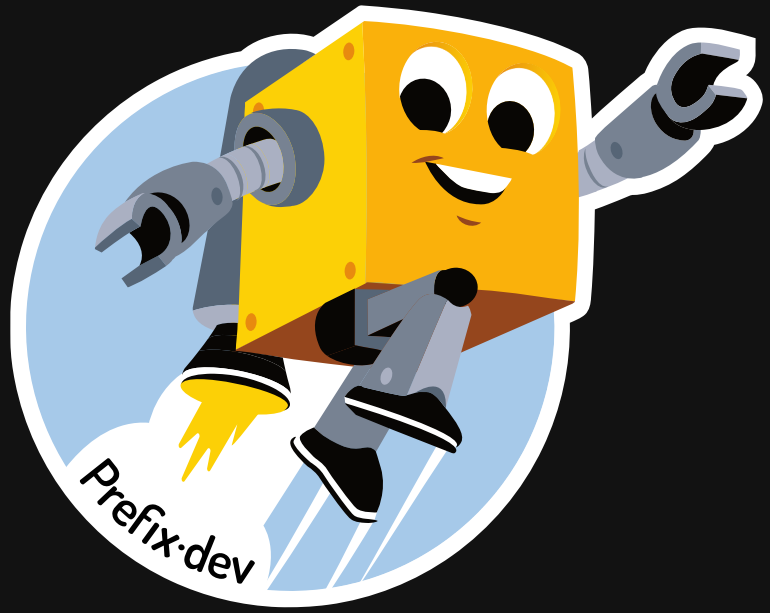
> From the **NumPy** contributor docs:
> If you are using conda, you can skip the steps in this section - with the exception of installing compilers for Windows or the Apple Developer Tools for macOS. All other dependencies will be installed automatically [...]

```
conda env create -f environment.yml

# or
pixi init --import environment.yml
```

Introducing Pixi

- ⚡ Fast
- 🆓 Open-Source
- 🛠️ Workflow management
- 🌐 Multi-environments
- 🔒 Reproducible thanks to lock-files
- 🐍 Supports conda and PyPI ecosystem



Ecosystem Comparison

Feature	conda	PyPI
Official Python Index	⚠️	✅
Cross-Platform	✅	✅
Cross-Language	✅	⚠️
Decentralized	✅	⚠️
Traditional Package Manager	conda	pip (conda)
Modern Package Manager	pixi	uv (pixi)

What About `uv`?

Or `hatch`, `poetry`, ...

- 💛 Amazing tool, highly appreciate their work.
- Pixi supports PyPI by integrating `uv`.
- Like `pixi`, `uv` uses the workspace model.
- Support both by using `pyproject.toml`.

`pyproject.toml`

```toml
[project]
name = "my-project"
version = "0.1.0"
dependencies = [
    "matplotlib",
    "numpy",
]

[tool.pixi.workspace]
channels = ["conda-forge"]
platforms = ["linux-64", "osx-arm64", "win-64"]
```

`Terminal`

```
pixi run python -c "import matplotlib; import numpy"
# or
uv run python -c "import matplotlib; import numpy"
```

Pixi Workflow

Initialization

```
pixi init demo
```

Adding `cowpy` and `python`

```
pixi add cowpy
```

Running a task

```
pixi run python -m demo
```

```
 --------------
< Hello All >
 --------------
    \   ^__^
     \  (oo)\_______
        (__)\       )\/\
          ||----w |
          ||     ||
```

`pyproject.toml`

```toml
[build-system]
requires = ["hatchling"]
build-backend = "hatchling.build"

[project]
name = "demo"
version = "0.1.0"
requires-python = ">=3.9"
dependencies = ["cowpy"]

[tool.pixi.workspace]
channels = ["conda-forge"]
platforms = ["linux-64"]
```

`demo/__main__.py`

```python
from cowpy.cow import Cowacter

message = Cowacter().milk("Hello All!")
print(message)
```

Tasks

Add a task

```
pixi task add hello "python hello.py"
```

Running a task

```
pixi run hello
```

```
 --------------
< Hello All >
 --------------
     \   ^__^
      \  (oo)\_______
         (__)\       )\/\
           ||----w |
           ||     ||
```

`pyproject.toml`

```toml
[build-system]
requires = ["hatchling"]
build-backend = "hatchling.build"

[project]
name = "demo"
version = "0.1.0"
requires-python = ">=3.9"
dependencies = ["cowpy"]

[tool.pixi.workspace]
channels = ["conda-forge"]
platforms = ["linux-64"]

[tool.pixi.tasks]
hello = "python -m demo"
```

Multiple Environments

```
pixi run --environment py312 hello
```

```
 _________________________
< Hello from Python 3.12! >
 -------------------------
    \   ^__^
     \  (oo)\_______
        (__)\       )\/\
          ||----w |
          ||     ||
```

```
pixi run --environment py313 hello
```

```
 _________________________
< Hello from Python 3.13! >
 -------------------------
    \   ^__^
     \  (oo)\_______
        (__)\       )\/\
          ||----w |
          ||     ||
```

pyproject.toml

```toml
[tool.pixi.feature.py312.dependencies]
python = "3.12.*"

[tool.pixi.feature.py313.dependencies]
python = "3.13.*"

[tool.pixi.environments]
py312 = ["py312"]
py313 = ["py313"]
```

demo/__main__.py

```python
from sys import version_info as vi
from cowpy.cow import Cowacter

python_version = f"Python {vi.major}.{vi.minor}"
message = Cowacter().milk(f"Hello from {python_version}!")
print(message)
```

Pixi Build

👷 Under construction

- 🏗️ Build your own conda packages directly from a `pixi.toml` or `pyproject.toml`
- 🐍 Using the build-backend principle like PyPI
- 🛠️ Support multi language monorepos
- 📦 Build and publish to custom channels

`pyproject.toml`

```toml
[project]
name = "package-name"
version = "0.1.0"

[tool.pixi.workspace]
channels = ["conda-forge"]
platforms = ["win-64", "linux-64", "osx-arm64"]

[tool.pixi.dependencies]
package-name = { path = "." }

[tool.pixi.package]
name = "package-name"
version = "0.1.0"

[tool.pixi.package.build]
backend = "pixi-build-python"
```

Lock file

- 👐 `pyproject.toml` : direct dependencies
- 🔒 `pixi.lock` : whole dependency graph
- 🎆 Fully reproducible setup

```
    license: bzip2-1.0.6
    license_family: BSD
    purls: []
    size: 252783
    timestamp: 1720974456583
  - conda: https://conda.anaconda.org/conda-forge/linux-64/ca-certificates-2024.12.14-hbcca054_0.conda
    sha256: 1afd7274cbc9a334d6d0bc62fa760acc7afdaceb0b91a8df370ec01fd75dc7dd
    md5: 720523eb0d6a9b0f6120c16b2aa4e7de
    license: ISC
    purls: []
    size: 157088
    timestamp: 1734208393264
  - pypi: https://files.pythonhosted.org/packages/29/a2/76daec910034d765f1018d22660c0970fb99f77143a42841d067b522903e/co
```


C
C++
C#
F
Java
JS
Lua
node
php
R
TS
WA
ZIG

Cross-language

- 📝 Version-control with `git` (written in C)

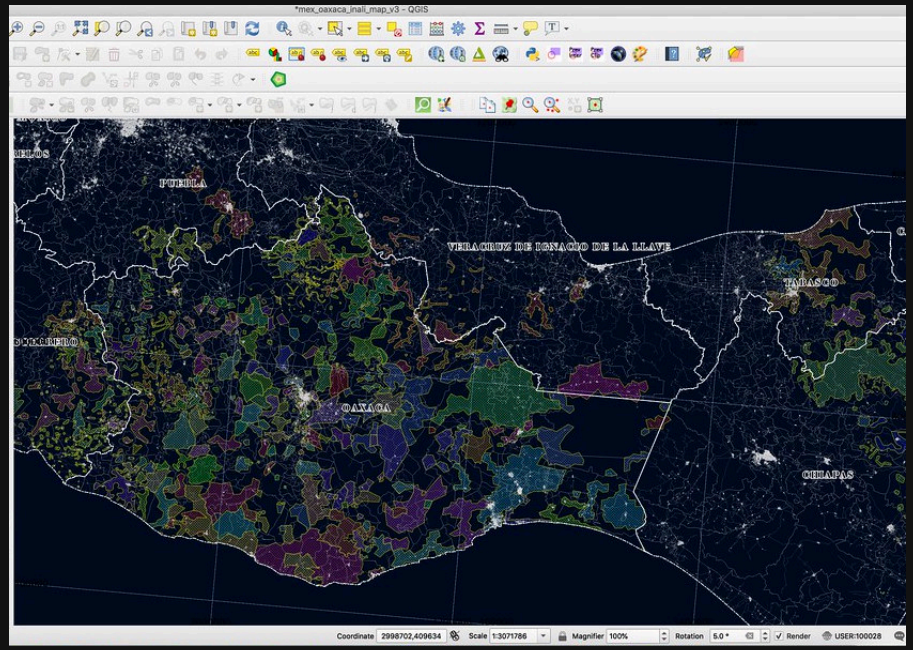
```
pixi add git
```

- 🐙 Manage GitHub repos with `gh` (written in Go)

```
pixi add gh
```

- 🌍 Geoscience with `QGIS` (written in C++)

```
pixi add qgis
```

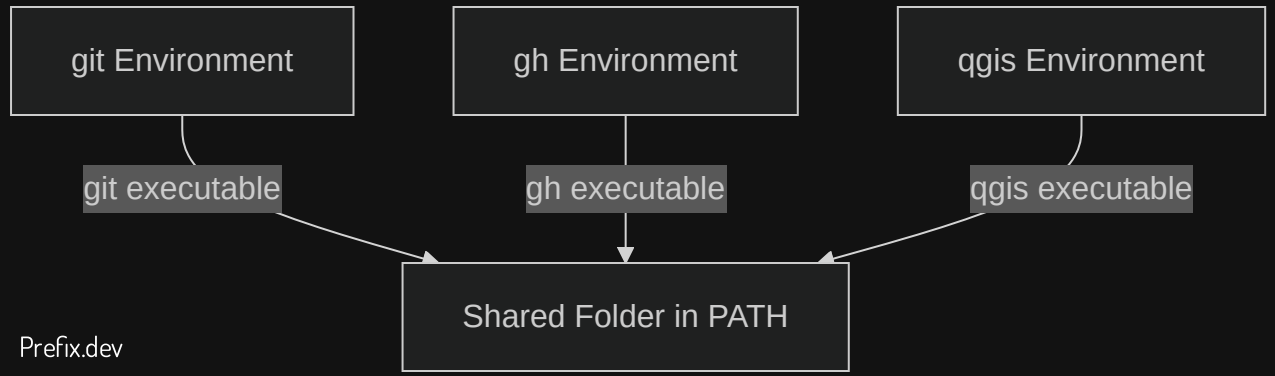


Pixi global

- 📦 Replace `apt`, `brew` or `winget` with `pixi`.
- ➕ `pixi add` adds dependencies to a workspace.
- 🌍 `pixi global install` installs tools globally on your system.
- 🛠️ Each tool is isolated in its own virtual environment.

Usage:

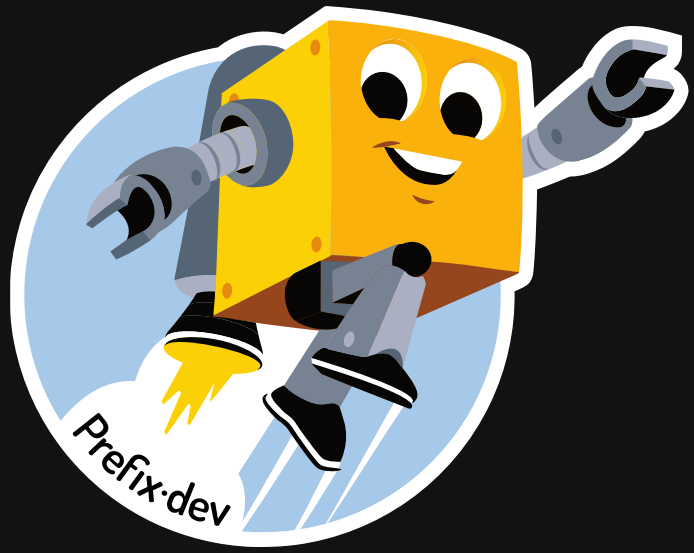
```
pixi global install git gh qgis
```



Projects using Pixi

Conclusion

- Modernize your workflow
  - Reproducible
  - Fast
  - Cross language
- One tool for all your development needs
- Free & Open-Source



Thank you for your attention!

Pixi Website

LinkedIn

Discord