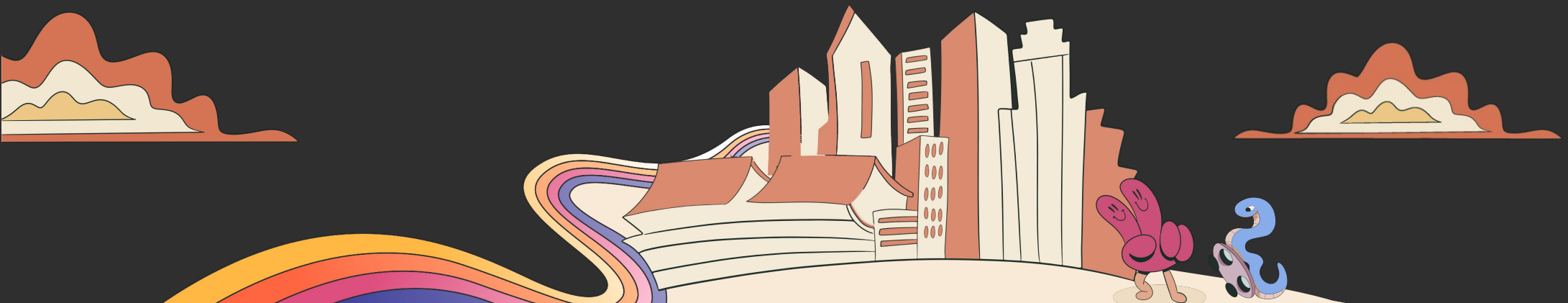
PyConUS 2025

Pittsburgh

Reinventing the Wheel: A Community-Driven Roadmap for Python Packaging

Jonathan Dekhtiar, Barry Warsaw

PyCon 2025



PyCon US 2025

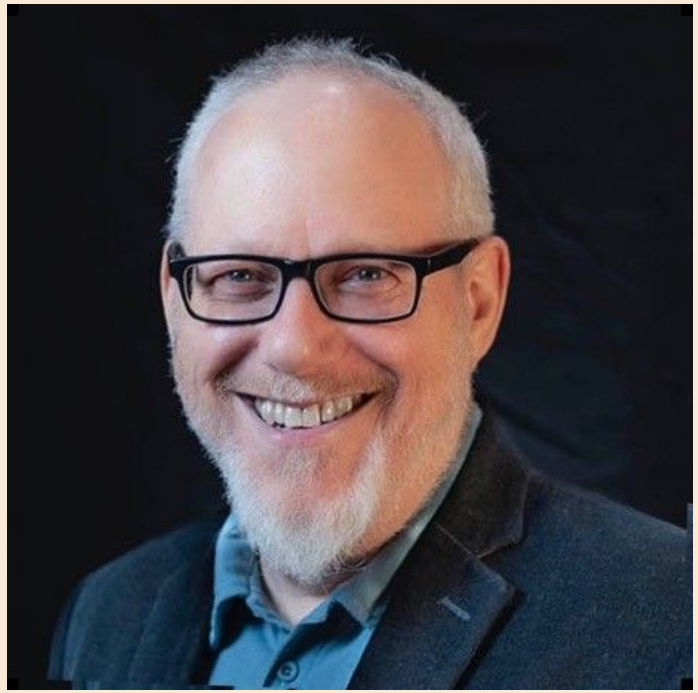
Who we are

Jonathan Dekhtiar

WheelNext - NVIDIA

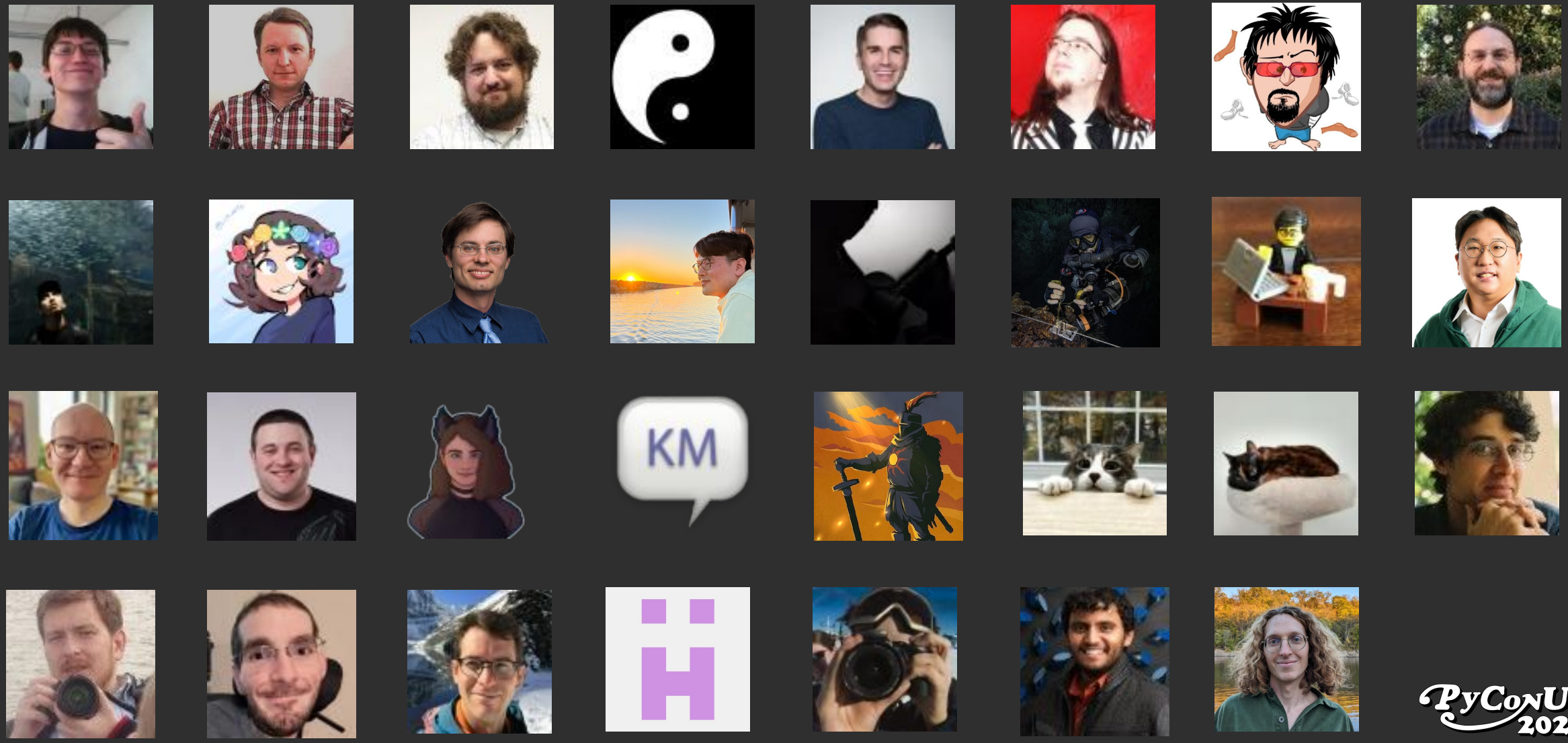
Barry Warsaw

Core Dev - WheelNext - NVIDIA



The work of so many

Alphabetic Order



PyConUS 2025

1 Celebrating Wins

PyConUS 2025

Celebrating our wins

633,415 projects 6,904,556 releases 14,162,240 files 920,172 users

python™ Package Index

The Python Package Index (PyPI) is a repository of software for the Python programming language.

PyPI helps you find and install software developed and shared by the Python community. Learn about installing packages.

Package authors use PyPI to distribute their software. Learn how to package your Python code for PyPI.

Celebrating our wins

PyPI Stats

Search

All packages

Top packages

Track packages

__all__

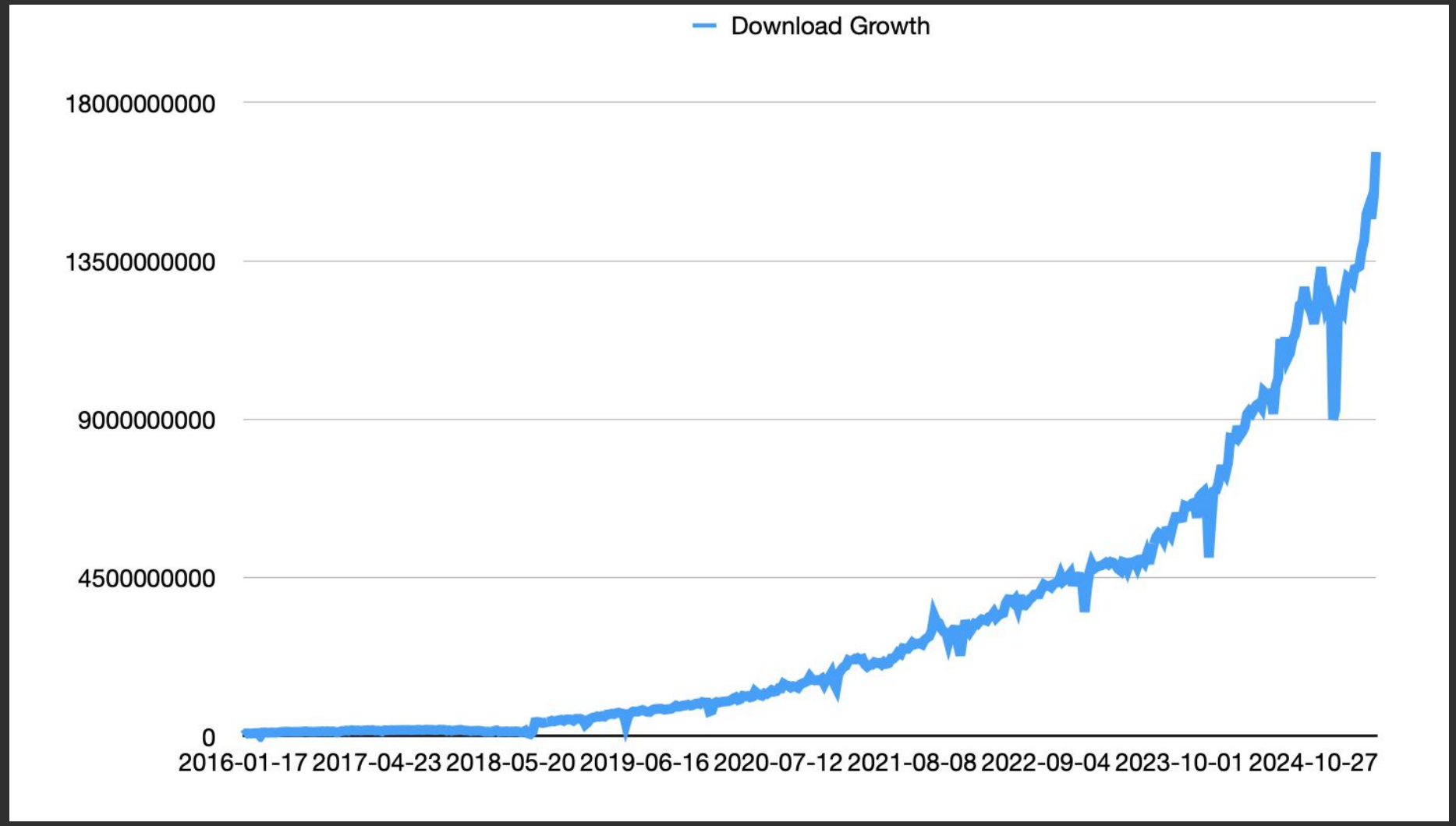
Download stats for __all__ indicate downloads across all packages on PyPI.

Downloads last day: 1,674,365,442

Downloads last week: 3,413,878,375

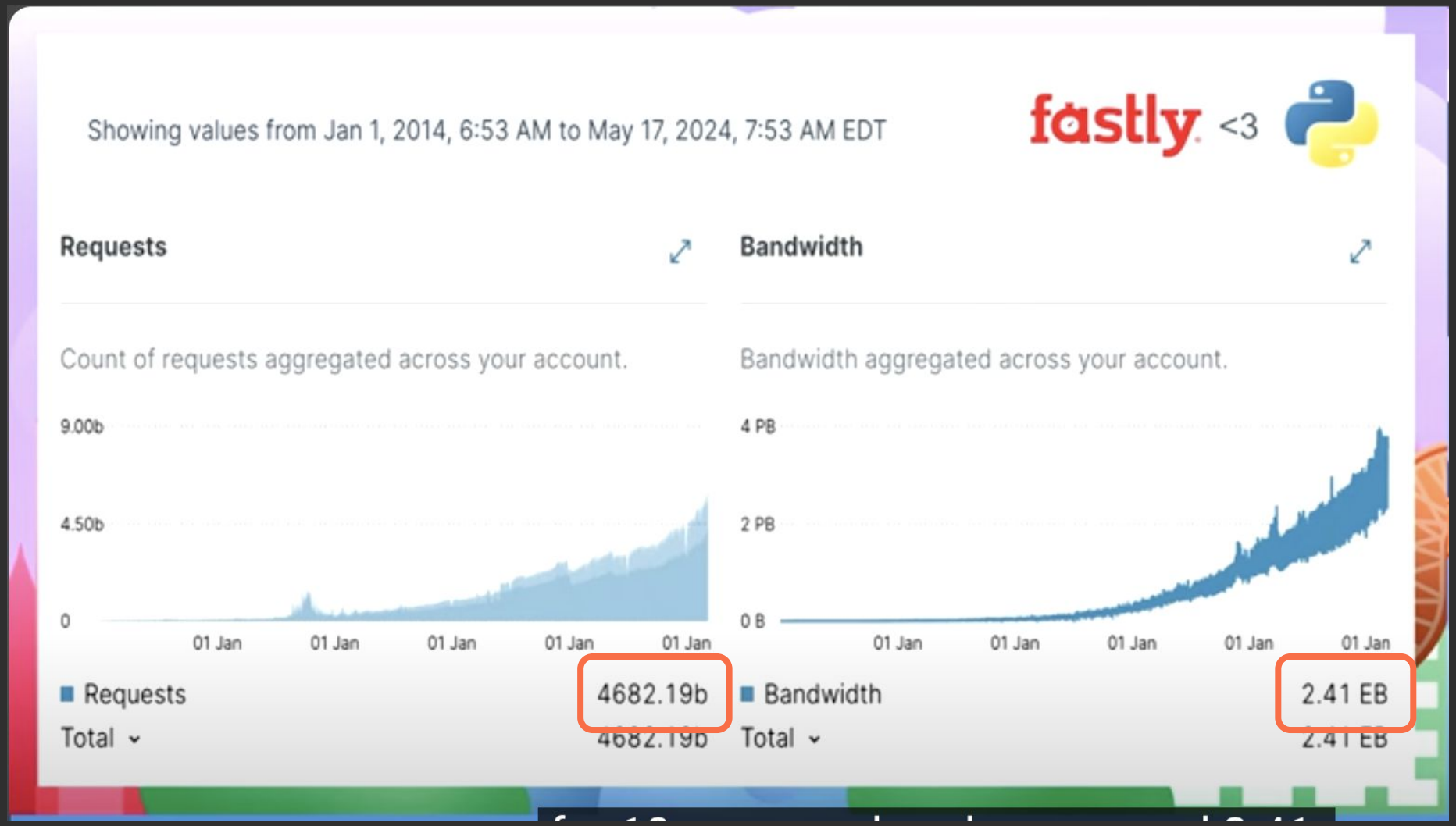
Downloads last month: 19,365,157,143

Celebrating our wins



PyConUS 2025

Celebrating our wins



PyConUS 2025

Celebrating our wins

- Hard to argue that Python packages and wheels aren't hugely successful
- The results of much hard work spanning years across the ecosystem
- Wheels serve the needs of most users most of the time
- But…
- …cracks are beginning to show for some important use cases

2 | WheelNext

What is “WheelNext”?

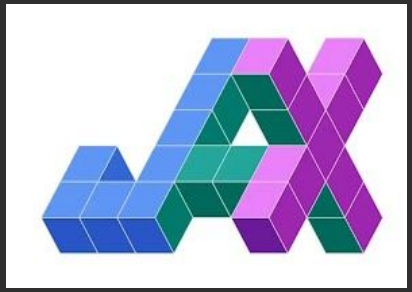
- Open-Source initiative aiming to “Reinvent the Wheel”
- Evolving the Python packaging ecosystem
- Many participants across:
  - Companies, organizations, and individuals
  - Users, tool makers, consumers/installers, producers/builders

contribute.wheelnext.dev

github.com/wheelnext

PyConUS 2025

WheelNext - Who are we ?



PyConUS 2025

WheelNext Inspiration

- pypackaging-native.github.io
- GPUs, CPU microarchitectures, OpenMP/MPI/BLAS/LAPACK
- Wheel hosting size limitations on PyPI
- Native dependencies

PyConUS 2025

WheelNext Design Axioms

- If it works for you now, it'll work for you later

PyConUS 2025

WheelNext Design Axioms

- If it works for you now, it'll work for you later
- Prioritize the UX; push complexity into the tooling

WheelNext Design Axioms

- If it works for you now, it'll work for you later
- Prioritize the UX; push complexity into the tooling
- Meet users where they are

WheelNext Design Axioms

- If it works for you now, it'll work for you later
- Prioritize the UX; push complexity into the tooling
- Meet users where they are
- Ecosystem-wide; no single tool or service focus

PyConUS 2025

WheelNext Design Axioms

- If it works for you now, it'll work for you later
- Prioritize the UX; push complexity into the tooling
- Meet users where they are
- Ecosystem-wide; no single tool or service focus
- Prioritize backward compatibility

WheelNext Design Axioms

- If it works for you now, it'll work for you later
- Prioritize the UX; push complexity into the tooling
- Meet users where they are
- Ecosystem-wide; no single tool or service focus
- Prioritize backward compatibility
- If something *must* break, do so intentionally and explicitly

PyConUS 2025

3 Problems

Problem: PyPI quotas

- Package size limitation
  - 100 MiB / file by default
  - 1GB / file hard limit

- Project size limitation
  - 10 GiB / project by default

Problem: Multiple Indexes

- Why use multiple indexes?
- Installing from multiple indexes can be unintuitive
  - Users expect installer flags to express priority, but that does not match reality
  - How do users ensure that they are getting what they want?
  - How do projects provide instructions for reliable package installation?
- Confusing at best. Prone to exploitation at worst.
  - Name collision across different indexes; dependency confusion attacks

Problem: Wheel 1.0

- Difficult to adopt new features, e.g.
  - symlinks
  - Zstandard compression
  - METADATA.json
- Backward compatibility
- Long tail of adoption

Problem: “shared libraries”

- No symlinks in Python wheels => reducing “packaging bloat”
  https://pypackaging-native.github.io/other_issues/#lack-of-support-for-symlinks-in-wheels
- No safe, robust, and portable approach for sharing native libraries
  - Ensuring all dependents load “the right” shared library
- Duplication (in whls) of common Dynamic Shared Objects (DSOs)
  - Increase “package bloat” and load on PyPI infrastructure.

Problem: Default Extras

`pip install package[default_extra]`

- Optional dependencies often used to express different “backends”
- Some packages *require* at least one extra to be installed
- But which one?
- Make it easy by providing a default extra if none is specified
- `pip install pkg` == `pip install pkg[default_extra]`

Problem: Specialized "Hardware"

Going beyond Python version + ABI + Platform

- No way to more finely describe the operating environment
  - What type of hardware do you have *(e.g. GPU, FPGA, ASIC, etc.) ?*
  - What x86-64 / ARM version (e.g. x86-64v3, ARMv7, ARMv8, etc.)?
  - What special instruction sets (e.g. AVX512)?
  - Specialized libraries (BLAS, MPI, etc.)

Problem: Specialized “Hardware”

Going beyond Python version + ABI + Platform

PyTorch Build	Stable (2.5.1)			Preview (Nightly)	
Your OS	Linux	Mac			
Package	Conda	Pip			Source
Language	Python			C++ / Java	
Compute Platform		CUDA 12.1	CUDA 12.4	ROCm 6.2	CPU
Run this Command:	pip3 install torch torchvision torchaudio --index orch.org/whl/cpu				

This can not be the best answer our community has - We must do better.

PyTorch

PyConUS 2025

Problem: Specialized “Hardware”

Going beyond Python version + ABI + Platform

	Linux, x86_64	Linux, aarch64	Mac, x86_64	Mac, aarch64	Windows, x86_64	Windows WSL2, x86_64
CPU	yes	yes	jax≤0.4.38 only	yes	yes	yes
NVIDIA GPU	yes	yes	no	n/a		experimental
Google Cloud TPU	yes	n/a		n/a	n/a	n/a
AMD GPU	[illegible]	no	experimental	n/a	no	no
[illegible] GPU	n/a	no	n/a	experimental	n/a	
Intel GPU	experimental	n/a	n/a	n/a	no	

This can not be the best answer our community has - We must do better.

PyConUS 2025

04

WheelNext Proposals

PyConUS 2025

Proposals: Honorable Mentions

PEP	Title
PEP 759	External wheel hosting
PEP 771	Default extras for Python software packages
PEP 777	How to re-invent the wheel
PEP 778	Supporting symlinks in wheels
PEP XXX	Build isolation bypass for specific dependencies

Proposed: PEP 772

Packaging governance process

- A formal packaging governance council, like the PSC
- Transfer standing delegations for Packaging PEPs
- Works with PyPA, dissolves and replaces Packaging WG
- Large voting community:
  - PyPA members
  - Packaging WG
  - core devs, wider community members, etc.

Proposed: Informational PEP 766

Explicit Priority Choices Among Multiple Indexes

- Defines "version priority" and "index priority":
  - *Version priority (like pip):* *combines indexes then finds the highest version*
  - *Index priority (like uv and conda):* *resolves packages one-index-at-a-time*
- Terminology and descriptions are provided to help package providers and end users differentiate installer behaviors.
- Avoids rigid implementation requirements to allow innovation
  - Expect that pip, uv, and other installers would follow user demand

PyConUS 2025

Proposed: Future PEP

Native Library Loader

- Safely use shared libraries distributed in wheels
- No conflict between wheels and system libraries (if present)
- Multiple approaches exist to address this problem

  ⇒ Help us converge and test
- We want a standardized comprehensive solution
- “Good enough” is better than “nothing at all”

Proposed: Future PEP

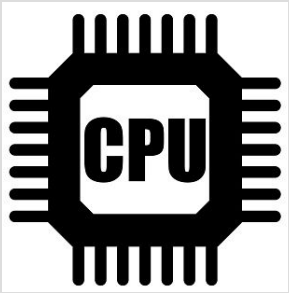
Wheel Variants

Today's (partial) Platform (tags)

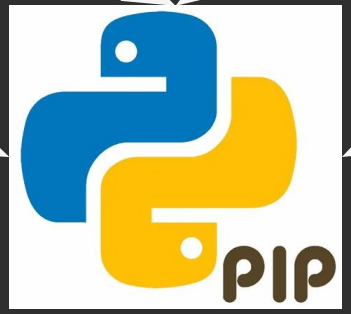
Python (+ ABI)

OS

GLibC



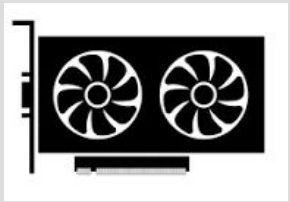
CPU Arch

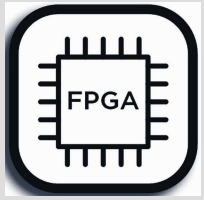


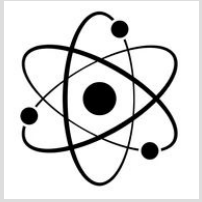
Proposed: Future PEP

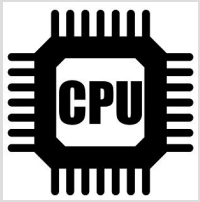
Wheel Variants

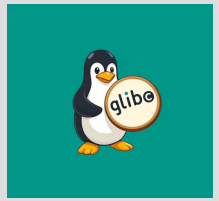
Complete Platform

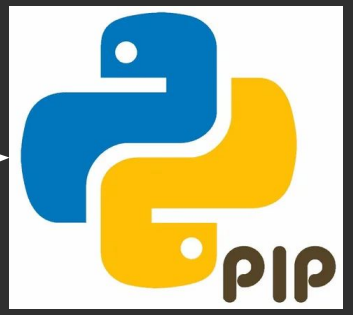










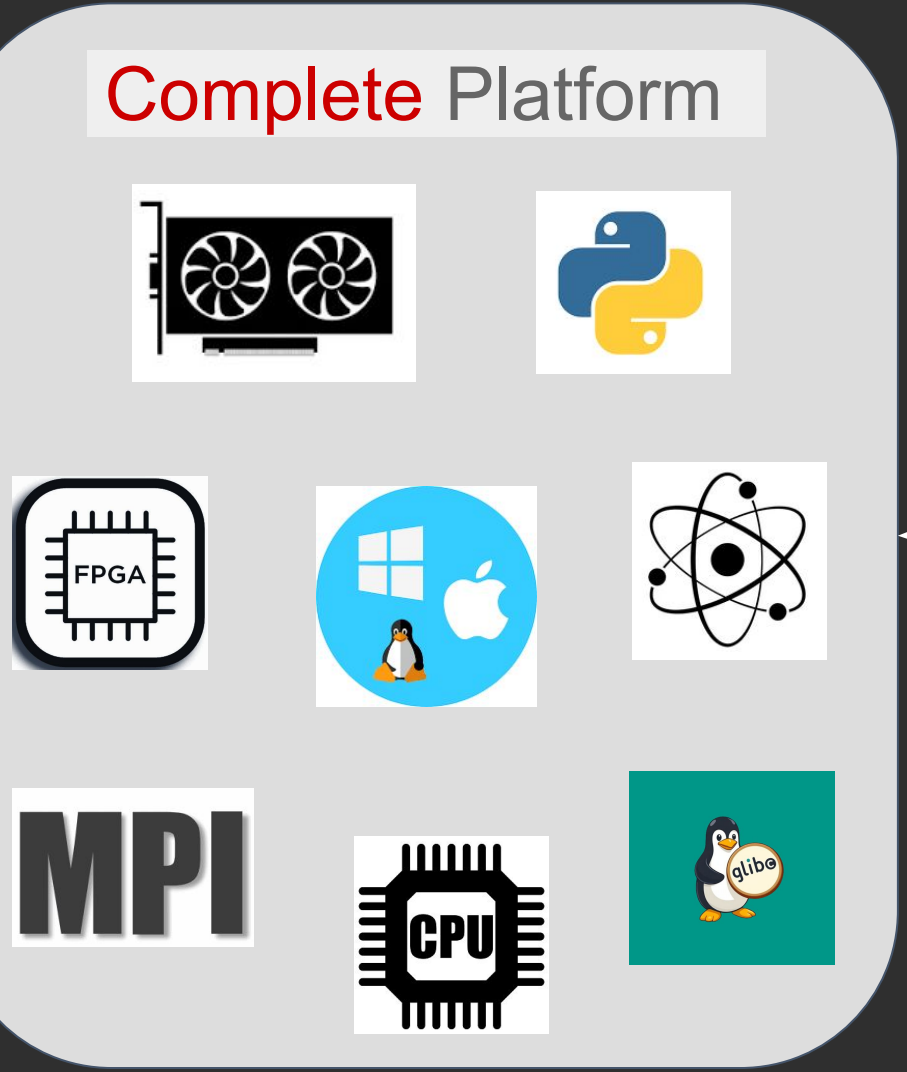


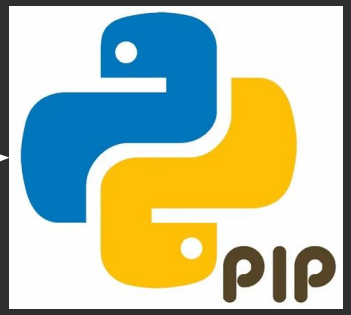
PyConUS 2025

Proposed: Future PEP

Wheel Variants

Complete Platform






```
# ========= dependent on ========= #

 - x86_64:    v1, v2, v3, v4
 - ARM:       v7, v8, v9
 - BLAS:      OpenBLAS, MKL, etc.

pip install numpy


# ========= dependent on ========= #

 - NVIDIA CUDA:  11.8, 12.6, 12.8
 - TPU:          v4, v5
 - CPU Instr. :  AVX512 Yes/No

pip install jax/torch
```

PyConUS 2025

Proposed: Future PEP

Wheel Variants

https://variants-demo.wheelnext.dev

Proposed: Future PEP

Wheel Variants

```
$ pip install numpy

Installing variant-provider-plugins in current environment:
  - provider-variant-aarch64 == 0.0.1;

  Variant `09300f2f` rejected `[aarch64 :: version :: 8.4a]` is not supported.
  Variant `c87a4099` rejected `[aarch64 :: version :: 8.5a]` is not supported.

  Total Number of Compatible Variants: 4

######################### Selected Variant: `522ebbc7` #########################
Variant-property: aarch64 :: version :: 8.3a
##############################################################################

Collecting numpy
  numpy-2.2.5-cp312-cp312-macosx_14_0_arm64-522ebbc7.whl (5.1 MB)

Installing collected packages: numpy
Successfully installed numpy-2.2.5-522ebbc7
```

ARM v8.3a

NumPy

Proposed: Future PEP

Wheel Variants

```
$ pip install numpy

Installing variant-provider-plugins in current environment:
  - provider-variant-aarch64 == 0.0.1;

  Variant `09300f2f` rejected `[aarch64 :: version :: 8.4a]` is not supported.
  Variant `c87a4099` rejected `[aarch64 :: version :: 8.5a]` is not supported.

  Total Number of Compatible Variants: 4

######################### Selected Variant: `522ebbc7` #########################
Variant-property: aarch64 :: version :: 8.3a
##############################################################################

Collecting numpy
  numpy-2.2.5-cp312-cp312-macosx_14_0_arm64-522ebbc7.whl (5.1 MB)

Installing collected packages: numpy
Successfully installed numpy-2.2.5-522ebbc7
```

ARM v8.3a

NumPy

Proposed: Future PEP

Wheel Variants

```
$ pip install numpy

Installing variant-provider-plugins in current environment:
  - provider-variant-aarch64 == 0.0.1;

  Variant `09300f2f` rejected `[aarch64 :: version :: 8.4a]` is not supported.
  Variant `c87a4099` rejected `[aarch64 :: version :: 8.5a]` is not supported.

  Total Number of Compatible Variants: 4

######################### Selected Variant: `522ebbc7` ########################
Variant-property: aarch64 :: version :: 8.3a
##############################################################################

Collecting numpy
  numpy-2.2.5-cp312-cp312-macosx_14_0_arm64-522ebbc7.whl (5.1 MB)

Installing collected packages: numpy
Successfully installed numpy-2.2.5-522ebbc7
```

ARM v8.3a

NumPy

Proposed: Future PEP

Wheel Variants

```
$ pip install numpy

Installing variant-provider-plugins in current environment:
  - provider-variant-aarch64 == 0.0.1;

  Variant `09300f2f` rejected `[aarch64 :: version :: 8.4a]` is not supported.
  Variant `c87a4099` rejected `[aarch64 :: version :: 8.5a]` is not supported.

  Total Number of Compatible Variants: 4

######################### Selected Variant: `522ebbc7` #########################
Variant-property: aarch64 :: version :: 8.3a
##############################################################################

Collecting numpy
  numpy-2.2.5-cp312-cp312-macosx_14_0_arm64-522ebbc7.whl (5.1 MB)

Installing collected packages: numpy
Successfully installed numpy-2.2.5-522ebbc7
```

ARM v8.3a

NumPy

Proposed: Future PEP

Wheel Variants

```
$ pip install numpy

Installing variant-provider-plugins in current environment:
  - provider-variant-x86-64 == 0.0.1;

  Variant `fa7c1393` rejected `[x86_64 :: level :: v3]` is not supported.
  Variant `cfdbe307` rejected `[x86_64 :: level :: v4]` is not supported.

  Total Number of Compatible Variants: 2

######################### Selected Variant: `40aba78e` ########################
Variant-property: x86_64 :: level :: v2
##############################################################################

Collecting numpy
  numpy-2.2.5-cp312-cp312-linux_x86_64-40aba78e.whl (17.6 MB)

Installing collected packages: numpy
Successfully installed numpy-2.2.5-40aba78e
```

X86-64 v2

NumPy

Proposed: Future PEP

Wheel Variants

```
$ pip install torch

Installing variant-provider-plugins in current environment:
  - nvidia-variant-provider == 0.0.1;

  Variant `1065b45d` rejected `[nvidia :: cuda :: 11.8]` is not supported.

  Total Number of Compatible Variants: 3

######################### Selected Variant: `d5784ad6` #########################
Variant-property: nvidia :: cuda :: 12.8
################################################################################

Collecting torch
  torch-2.7.0-cp312-cp312-manylinux_2_28_x86_64-d5784ad6.whl (1096.4 MB)

Installing collected packages: numpy
Successfully installed torch-2.2.5-d5784ad6
```

NVIDIA CUDA

CUDA 12.8

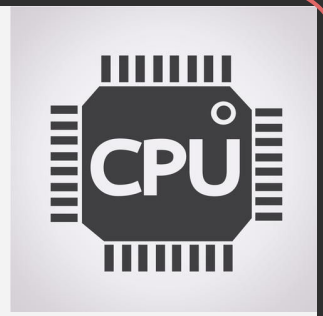
PyTorch

PyConUS 2025

Proposed: Future PEP

Wheel Variants

```
$ pip install torch

Installing variant-provider-plugins in current environment:
  - nvidia-variant-provider == 0.0.1;

  Variant `1065b45d` rejected `[nvidia :: cuda :: 11.8]` is not supported.
  Variant `43331073` rejected `[nvidia :: cuda :: 12.6]` is not supported.
  Variant `d5784ad6` rejected `[nvidia :: cuda :: 12.8]` is not supported.

  Total Number of Compatible Variants: 0

Collecting torch
  torch-2.7.0-cp312-cp312-manylinux_2_28_x86_64.whl (175.4 MB)

Installing collected packages: numpy
Successfully installed torch-2.2.5
```



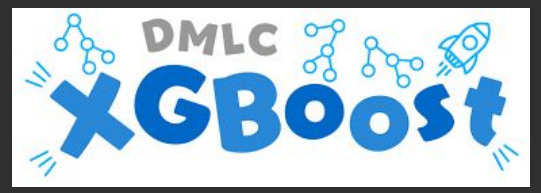
CPU “only”

“No variant found”

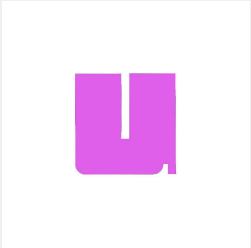
PyTorch

Proposed: Future PEP

Wheel Variants



(Hopefully) Very Soon - Work in Progress



05 | Call to Action

PyConUS 2025

Join us!

- Let's hear from you!
  - wheelnext.dev & GitHub (Use the QR code)
  - PyPA Discord #wheelnext
- PyCon 2025
  - Sprints
  - Language/Packaging Summits (read the blogs)
  - Stickers!
- Try `variants-demo.wheelnext.dev`

PyCon US 2025

Grab a sticker and join the adventure

PyConUS 2025

WheelNext Resources

https://contribute.wheelnext.dev

https://github.com/wheelnext

https://mailing.wheelnext.dev

PyConUS 2025