What’s new in Python

Veit Schiele
Cusy GmbH, Berlin
Python Users Berlin, 10 March 2022

github.com/cusyio/jupyter-reveal/pub/pub_2022-03.pdf

Python 3.11.0a6 released

On Monday, 7 March, Python 3.11 a6 was released

Python 3.11.0a6 released

On Monday, 7 March, Python 3.11 a6 was released

Release schedule

Final release of Python 3.11 is planned for October 2022,

see PEP 664 – Python 3.11 Release Schedule

Python 3.11.0a6 released

New features

see Python 3.11.0a6 release notes

Python 3.11.0a6 released

New features

PEP 657 – Include Fine Grained Error Locations in Tracebacks

The traceback contains visual feedback about the exact location in a line that threw an exception, for example

```
Traceback (most recent call last):
  File "test.py", line 2, in <module>
    x['a']['b']['c']['d'] = 1
    ~~~~~~~~~~~^^^^^
TypeError: 'NoneType' object is not subscriptable
```

Python 3.11.0a6 released

New features

PEP 657 – Include Fine Grained Error Locations in Tracebacks

Opt-out mechanism

... to save memory and still allow tools to automatically analyse tracebacks:

- Environment variable: `PYTHONNODEBUGRANGES`
- Command line option: `python -Xno_debug_ranges`

Python 3.11.0a6 released

New features

PEP 654 – Exception Groups and except*

There will be a new exception type called *exception group*, which will allow multiple exceptions to be grouped into one container. This essentially simplifies error handling in concurrent code, especially with Async IO, but also, for example, when grouping multiple exceptions that occur during data validation.

Python 3.11.0a6 released

New features

PEP 654 – Exception Groups and except*

ExceptionGroup and BaseExceptionGroup

```
>>> eg = ExceptionGroup(
...     "one",
...     [
...         TypeError(1),
...         ExceptionGroup(
...             "two",
...              [TypeError(2), ValueError(3)]
...         ),
...         ExceptionGroup(
...              "three",
...               [OSError(4)]
...         )
...     ]
... )
```

Python 3.11.0a6 released

New features

PEP 654 – Exception Groups and except*

ExceptionGroup and BaseExceptionGroup

```
>>> import traceback
>>> traceback.print_exception(eg)
  | ExceptionGroup: one (3 sub-exceptions)
  +-+---------------- 1 ----------------
    | TypeError: 1
    +---------------- 2 ----------------
    | ExceptionGroup: two (2 sub-exceptions)
    +-+---------------- 1 ----------------
      | TypeError: 2
      +---------------- 2 ----------------
      | ValueError: 3
      +------------------------------------
```

Python 3.11.0a6 released

New features

PEP 654 – Exception Groups and except*

except*

This introduces a new variant of the try except syntax to simplify working with exception groups.

`*` indicates that multiple exceptions can be handled by each `except*` clause:

Python 3.11.0a6 released

New features

PEP 654 – Exception Groups and except*

except*

This introduces a new variant of the try except syntax to simplify working with exception groups.

`*` indicates that multiple exceptions can be handled by each `except*` clause:

```
try:
    ...
except* SpamError:
    ...
except* FooError as e:
    ...
except* (BarError, BazError) as e:
    ...
```

Python 3.11.0a6 released

New features

PEP 673 – Self Type

PEP 673 wants to introduce a simpler and more intuitive way to annotate methods that return an instance of their class. This corresponds to the TypeVar-based approach in PEP 484 – Type Hints.

Python 3.11.0a6 released

New features

PEP 673 – Self Type

PEP 673 wants to introduce a simpler and more intuitive way to annotate methods that return an instance of their class. This corresponds to the TypeVar-based approach in PEP 484 – Type Hints.

Reference implementation: PyRight

Python 3.11.0a6 released

New features

PEP 673 – Self Type

PEP 673 wants to introduce a simpler and more intuitive way to annotate methods that return an instance of their class. This corresponds to the TypeVar-based approach in PEP 484 – Type Hints.

Reference implementation: PyRight

> **See also:**
>
> Mypy thread from 2016 talking about ways to tackle the self-typing problem SelfType or another way to spell "type of self" (or, How to define a copy() function).

Python3.11.0a6 released

New features

PEP 646 – Variadic Generics

Variadic generics in the form of `TypeVarTuple` want to solve a very special use case in scientific libraries like NumPy and simplify the handling of multidimensional arrays. Thus, the shape of an array is to be defined by parameterising its type with a variable number of placeholder types.

Python 3.11.0a6 released

New features

PEP 680 – tomllib: Support for Parsing TOML in the Standard Library

The Steering Committee recently adopted PEP 680, which justifies the need for a TOML parser in the Python standard library, as PEP 517 – A build-system independent format for source trees leads to a new build system based on `pyproject.toml`. Previously, packaging tools had to bundle a library to read project metadata from that file.

Python 3.11.0a6 released

New features

PEP 680 – tomllib: Support for Parsing TOML in the Standard Library

The Steering Committee recently adopted PEP 680, which justifies the need for a TOML parser in the Python standard library, as PEP 517 – A build-system independent format for source trees leads to a new build system based on `pyproject.toml`. Previously, packaging tools had to bundle a library to read project metadata from that file.

> **Note:**
>
> The current plan only provides for a TOML parser, without the corresponding serialiser. To write TOML data, an external library must still be installed and imported.

Python 3.11.0a6 released

Performance optimisations

According to Anthony Shaw from the Faster Cpython Project, CPython 3.11 is up to 45% faster at creating instances of base classes and calling methods (most Python code!) 🥳

Python Issues Moving to GitHub

If you have found a bug in Python 3.11.0 a6, you can currently report it to the Roundup-based Python Bug Tracker (BPO). From 24 March, however, the move to GitHub Issues will begin.

Python Issues Moving to GitHub

If you have found a bug in Python 3.11.0 a6, you can currently report it to the Roundup-based Python Bug Tracker (BPO). From 24 March, however, the move to GitHub Issues will begin.

See also:

- Github Issues Migration is coming soon
- PEP 581 – Using GitHub Issues for CPython

PyCon US 2022 schedule announced

- After two years in which PyCon US was held virtually due to the pandemic, PyCon US 2022 seems to be able to take place in person again.
- This time the conference will be held in Salt Lake City, Utah.
- At the same time, there will be an online streaming option.

PyCon US 2022 schedule announced

Tutorials, Summits etc.

Dates	Events	Notes
27-28 April	Tutorials	150 USD per session; there are two timeslots per day
28 April	Sponsor Workshops	No entry fee
28 April	Education Summit	No entry fee
29 April	Maintainers Summit	No entry fee
30 April	Mentored Sprints for Diverse Beginners	No entry fee
30 April	PyLadies Auction	35 or 50 USD Entry; supporter rates are offered

PyCon US 2022 schedule announced

Tutorials, Summits etc.

Dates	Events	Notes
27-28 April	Tutorials	150 USD per session; there are two timeslots per day
28 April	Sponsor Workshops	No entry fee
28 April	Education Summit	No entry fee
29 April	Maintainers Summit	No entry fee
30 April	Mentored Sprints for Diverse Beginners	No entry fee
30 April	PyLadies Auction	35 or 50 USD Entry; supporter rates are offered

> **See also:**
> - Conference Schedule
> - Registration

GeoPython 2022

Due to an overlap with a major event in Basel, GeoPython 2022 had to be postponed to 20–22 June 2022.

GeoPython 2022

Due to an overlap with a major event in Basel, GeoPython 2022 had to be postponed to 20–22 June 2022.

Other important dates are:

- Submission deadline workshops: 15 March 2022 (12:00 AoE)
- Registration deadline: 15 March 2022
- Timetable for the first draft: 20 March 2022

PSF hires to improve PyPI

The Python Software Foundation (PSF) is looking to hire two developers to develop new features in the Python Package Index (PyPI). This decision was made after some surveys in the Python community identified the main user requirements that are currently missing in PyPI: the most requested feature was organisational accounts in PyPI. This is to become a paid service offered to companies.

PSF hires to improve PyPI

The Python Software Foundation (PSF) is looking to hire two developers to develop new features in the Python Package Index (PyPI). This decision was made after some surveys in the Python community identified the main user requirements that are currently missing in PyPI: the most requested feature was organisational accounts in PyPI. This is to become a paid service offered to companies.

One of the contractors will focus on the back-end while the other will take care of the front-end. Both roles will be location independent and contract work is expected to start in early April 2022. The budget for each role is up to $98k for approximately 560 hours, with 35 hours per working week.

PSF hires to improve PyPI

The Python Software Foundation (PSF) is looking to hire two developers to develop new features in the Python Package Index (PyPI). This decision was made after some surveys in the Python community identified the main user requirements that are currently missing in PyPI: the most requested feature was organisational accounts in PyPI. This is to become a paid service offered to companies.

One of the contractors will focus on the back-end while the other will take care of the front-end. Both roles will be location independent and contract work is expected to start in early April 2022. The budget for each role is up to $98k for approximately 560 hours, with 35 hours per working week.

> **See also:**
> - We are hiring to expand our Infrastructure staff!
> - Evaluation criteria

`pythoncapi_compat` is now a project of the GitHub Python Organisation

pythoncapi_compat is used to write a C extension for different Python versions with a single code base.