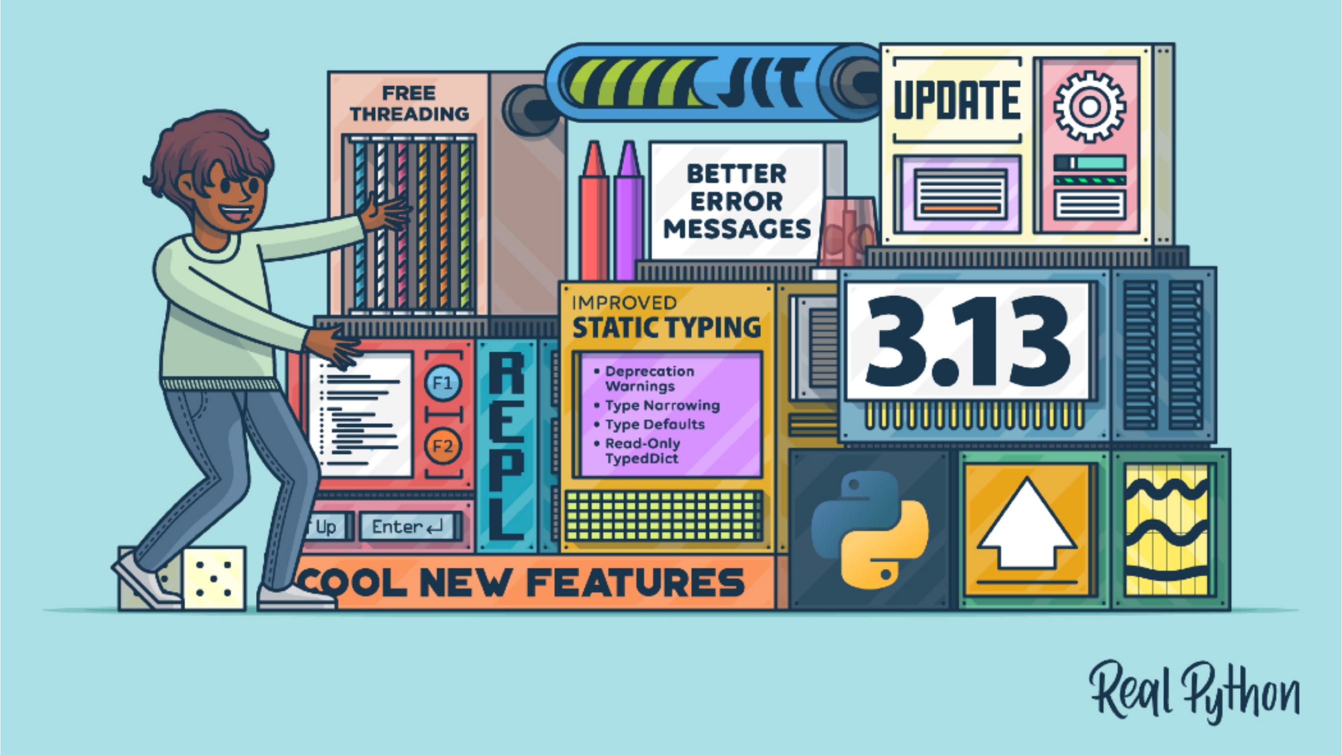
WHAT'S NEW IN PYTHON 3.13



OVERVIEW

- Python 3.13 is the 2024 release
- Improvements:
  - The new REPL
  - Experimental performance improvements
  - New additions to the typing system
  - Improvements to `copy` and `pathlib`
  - Loads more!

VERSIONS

Note:

- Sample code tested using:
  - Python 3.13 RC 2

NEXT UP...

→ REPL

TABLE OF CONTENTS

1. **Overview**
2. ▶ **REPL**
3. Free Threading and JIT
4. Better Error Messages
5. Typing: Deprecations and Defaults
6. Typing: ReadOnly and TypeIs
7. Random and Docstrings
8. Copy’s Replace
9. Pathlib

10.Other Goodness

11.Summary

REPL

- When you run `python` on its own you get the REPL
- **R**ead **E**valuate **P**rint **L**oop
- Python 3.13 has re-written the REPL adding:
  - Multi-line editing with history
  - REPL commands (help, exit, and quit)
  - Color highlighting for tracebacks
  - History browsing
  - Paste mode
- `PYTHON_BASIC_REPL=1`

NEXT UP...

→ Experimental performance features

TABLE OF CONTENTS

1. Overview

2. REPL

▶ **3. Free Threading and JIT**

4. Better Error Messages

5. Typing: Deprecations and Defaults

6. Typing: ReadOnly and TypeIs

7. Random and Docstrings

8. Copy’s Replace

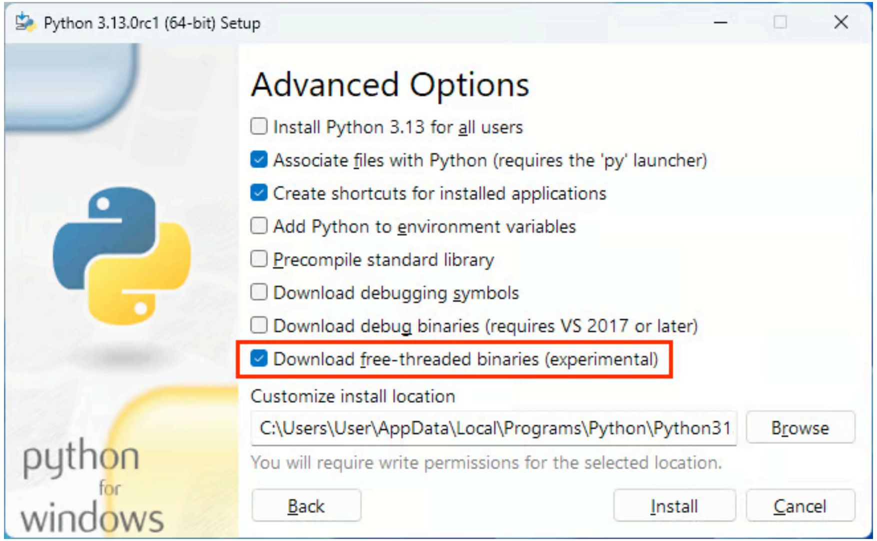
9. Pathlib

10.Other Goodness

11.Summary

EXPERIMENTAL FEATURES

- Python 3.13 “includes” two experimental features:
  - Free threading
  - JIT
- “Includes” means CPython compile flags
- On macOS or Windows use *Customize Installation* and *Advanced Options*
- Build it yourself



Python 3.13.0rc1 (64-bit) Setup

Advanced Options

☐ Install Python 3.13 for all users
☑ Associate files with Python (requires the 'py' launcher)
☑ Create shortcuts for installed applications
☐ Add Python to environment variables
☐ Precompile standard library
☐ Download debugging symbols
☐ Download debug binaries (requires VS 2017 or later)
☑ Download free-threaded binaries (experimental)

Customize install location

C:\Users\User\AppData\Local\Programs\Python\Python31

You will require write permissions for the selected location.

FREE THREADING

- Free from the GIL
- Global Interpreter Lock
  - A lock to prevent race-conditions during memory operations
  - Deep within Python and expected by extensions
  - Limits amount of concurrency on a single thread on a single CPU
- Roadmap:
  - **Experimental:** Not included in the official distribution
  - **Enabled:** GIL becomes optional
  - **Disabled:** GIL disabled by default, but still available

FREE THREADING

- Roadmap subject to change
- Multi-year plan
- Application Binary Interface (ABI) incompatible
  - C-extensions written for the free threaded version will not be compatible with the GIL enabled versions
  - Maintainers will have to ship two packages

JIT

- **JIT: J**ust **I**n **T**ime compilation
- Python currently compiles to byte code
- Byte code executed by the interpreter
- JIT compiles byte code to machine code
- Copy-and-patch mechanism
  - Pre-compiled templates of machine code for common patterns
  - JIT compiler fills in specifics for your program
  - Fast compilation
- Currently hoping for JIT on par with regular existing Python
- Will optimize over time

NEXT UP...

➡ More better error

TABLE OF CONTENTS

1. Overview

2. REPL

3. Free Threading and JIT

▶ **4. Better Error Messages**

5. Typing: Deprecations and Defaults

6. Typing: ReadOnly and TypeIs

7. Random and Docstrings

8. Copy’s Replace

9. Pathlib

10.Other Goodness

11.Summary

BETTER ERROR MESSAGES

- Command-line errors are now colorized by default
- Scripts with the same name as a library can be confusing
    - Bad imports result in extra hints
    - Specific mention of standard library modules
    - General mention of third party modules
- “Did you mean” errors now apply to keyword arguments

NEXT UP...

→ Typing

TABLE OF CONTENTS

1. **Overview**
2. **REPL**
3. **Free Threading and JIT**
4. **Better Error Messages**
5. ▶ **Typing: Deprecations and Defaults**
6. Typing: ReadOnly and TypeIs
7. Random and Docstrings
8. Copy’s Replace
9. Pathlib
10. Other Goodness
11. Summary

TYPING IMPROVEMENTS

- The Python typing system provides tools with type information to help prevent bugs
- Most IDEs will perform type checking, or you can use third party libraries

```
$ python -m pip install pyright
$ pyright --pythonversion 3.13 code.py
```

DEPRECATIONS

- The `warnings` module now has a `deprecated` decorator
- Decorate items you plan on removing from your code
- Only used by typing systems

```toml
# pyproject.toml
[tool.pyright]
reportDeprecated = true
```

DEFAULT TYPING

- The typing system allows you to declare typed templates
- Typically for containers
- Establish the type of the contents at instantiation time
- Python 3.13 has added default typing for templates

NEXT UP...

➡ More typing

TABLE OF CONTENTS

1. Overview

2. REPL

3. Free Threading and JIT

4. Better Error Messages

5. Typing: Deprecations and Defaults

▶ **6. Typing: ReadOnly and TypeIs**

7. Random and Docstrings

8. Copy’s Replace

9. Pathlib

10.Other Goodness

11.Summary

READONLY TYPE

- Can now declare a `TypeDict` attribute to have a `ReadOnly` type
- Only a type annotation

TYPEIS

- The typing system supports type guards:
    - A way of narrowing down acceptable types
    - Done automatically in simple cases
    - Can declare a function as a `TypeGuard` for more complex cases
    - A `TypeGuard` function returns a `Boolean` based on a type check
- Python 3.13 introduces `TypeIs`
- Also for type narrowing
- More restricted case, makes it narrower

NEXT UP...

→ Random and docstrings

TABLE OF CONTENTS

1. **Overview**
2. **REPL**
3. **Free Threading and JIT**
4. **Better Error Messages**
5. **Typing: Deprecations and Defaults**
6. **Typing: ReadOnly and TypeIs**
7. ▶ **Random and Docstrings**
8. Copy’s Replace
9. Pathlib
10. Other Goodness
11. Summary

RANDOM MODULE TOOL

- Can now invoke the `random` module as a command line tool
- Handy for generating random values on the fly

DOCSTRINGS

- Docstrings are now automatically de-indented
- Removes padding used for programming aesthetics from output
- If you’re running tests on your docstrings this is a breaking change!

NEXT UP...

➡ Copy and edit

TABLE OF CONTENTS

1. **Overview**
2. **REPL**
3. **Free Threading and JIT**
4. **Better Error Messages**
5. **Typing: Deprecations and Defaults**
6. **Typing: ReadOnly and TypeIs**
7. **Random and Docstrings**
8. ▶ **Copy’s Replace**
9. Pathlib
10. Other Goodness
11. Summary

COPY MODULE

- `copy.replace()` function in the `copy` module
- Copies object while replacing some attributes
- Example: copy a `NamedTuple` but change one of the values

NEXT UP...

➡ Pathlib

TABLE OF CONTENTS

1. Overview
2. REPL
3. Free Threading and JIT
4. Better Error Messages
5. Typing: Deprecations and Defaults
6. Typing: ReadOnly and TypeIs
7. Random and Docstrings
8. Copy’s Replace

▶ 9. Pathlib

10.Other Goodness

11.Summary

CHANGES TO `pathlib`

- Construct a path from a file URI
- Path matching using `*` and `**` patterns
- Consistent behavior for `**` in `Path.glob()` and `Path.rglob()`
- `recurse_symlinks` argument added to `Path.glob()` and `Path.rglob()`
- `follow_symlinks` argument for `Path.is_file`, `Path.is_dir`, `Path.owner()`, and `Path.group()`

CHANGES TO pathlib

```
music
├── opera
│   ├── flower_duet.txt
│   ├── habanera.txt
│   └── nabucco.txt
└── rap
    ├── bedlam_13-13.txt
    └── fight_the_power.txt
```

NEXT UP...

→ Miscellaneous goodness

TABLE OF CONTENTS

1. **Overview**
2. **REPL**
3. **Free Threading and JIT**
4. **Better Error Messages**
5. **Typing: Deprecations and Defaults**
6. **Typing: ReadOnly and TypeIs**
7. **Random and Docstrings**
8. **Copy’s Replace**
9. **Pathlib**

▶ **10.Other Goodness**

11.Summary

CHANGES TO `locals()`

- The `locals()` built-in function displays what is in context
- Several versions ago, the underlying container was modified for performance reasons
- PEP 667: Consistent views of namespaces

```
>>> import sys
>>> frame_locals = None
>>> [frame_locals := sys._getframe().f_locals for number in range(3)]
[{}, {}, {}]
```

Not empty in 3.12

True in 3.13

OTHER GOODNESS

- Performance improvements:
  - Garbage collector (large heaps)
  - Import times on some standard library modules
- `argparse` supports a `deprecated` parameter
- Colorization of `doctest`
- `importlib.resources()` now supports directories
- `breakpoint()` and `set_trace()` are now the debugger’s entry points

OTHER GOODNESS

- Mobile platforms
  - Apple iOS now a supported platform
  - Android support being worked on
- Dead batteries have been removed

NEXT UP...

➡ Summary

TABLE OF CONTENTS

1. Overview
2. REPL
3. Free Threading and JIT
4. Better Error Messages
5. Typing: Deprecations and Defaults
6. Typing: ReadOnly and TypeIs
7. Random and Docstrings
8. Copy’s Replace
9. Pathlib
10.Other Goodness

▶ 11.Summary

SUMMARY

- Python continues to improve
- REPL is now much friendlier: block edits and paste mode
- Experimental features: No-GIL and JIT
- Improved error messages
- Typing:
  - Deprecation decorator
  - Template typing defaults
  - `ReadOnly` attribute for `TypeDict`
  - `TypeIs` guard

SUMMARY

- Random can now be called from the command line
- Docstrings get de-dented
- Copy and edit with `copy.replace()`
- `*` and `**` improvements in `pathlib`
- iOS now a supported platform

MORE INFORMATION

- What’s New In Python 3.13 (docs)

  https://docs.python.org/3.13/whatsnew/3.13.html

- Python 3.13 Preview: Free Threading and a JIT Compiler

  https://realpython.com/python313-free-threading-jit/ (tutorial)

- Python 3.13 Preview: A Modern REPL

  https://realpython.com/python313-repl/

Thanks!

Dankie ju faleminderit faleminderit شكرا Grazias Շնորհակալություն Sağ ol eskerrik asko Дзякуй তোমাকে ধন্যবাদ hvala trugéré

благодаря Akeva Chezu ba gràcies Salamat zikomo 谢谢 hvala děkuji Tak dank u Dankon aitäh takk fyri salamat kiitos Merci

Grazas დიდი მადლობა Danke σας ευχαριστώ આભાર Mèsi poutèt ou Na gode Mahalo תודה Dhanyawaad köszönöm þakka þér

Daalụ terima kasih Go raibh maith agat grazie ありがとう matur nuwun ದನ್ಯವಾದಗಳು សូមអរគុណអ្នក Kamsahamnida ຂໍຂອບໃຈທ່ານ

Ngiyabonga paldies ačiū ви благодариме misaotra Terima kasih Grazzi xie Mauruuru Dhanyawaadh Welálin баярлалаа barka

Ahéhee' Dhanyabaad miigwetch manana تشکر از شما dziękuję obrigado ਤੁਹਾਡਾ ਧੰਨਵਾਦ mulţumesc спасибо tapadh leibh хвала

ďakujem hvala Waad ku mahadsan tahay Gracias Asante Tack Salamat rahmat நன்றி ధన్యవాదాలు ขอบคุณ tualumba teşekkür

ederim Спасибі آپ کا شکریہ rahmat cảm ơn bạn Diolch yn fawr אַ דאַנק Balika o ṣeun