Python
C API

Victor
Stinner

In Python, there is something called the "C API"...

Past

Break API at each release

- 3.8: Remove PyTypeObject.**tp_print**
- 3.10: Remove PyObject_**AsWriteBuffer**()
- 3.11: Remove **PyFrameObject** members
- etc.

ABI issues

- **#73323: fpectl module**
  Module removed in Python 3.7

- **#83780: PyGC_Head in Python 3.7.5**
  Structure made internal in Python 3.9

- **3.8: Debug & release builds compatible**
  Py_TRACE_REFS off by default

- **3.12: Immortal objects (PEP 683): refcount**
  Py_INCREF() opaque in limited API 3.12

Delay after Python releases

Python 3.12 alpha1 (Oct 24, 2022)

- Cython: **7 months** (May, 24)
- numpy: **10 months** & 3 weeks (Sep, 16)
- cffi: **11 months** (Sep, 28)

Python 3.12 rc1: August, 1
Python 3.12.0 final: October, 2

Ship many Python versions

- Python 3.6 (2016-20**21**)
- RHEL 8 (2019-20**29**+) uses Python 3.6
- Python 2.7, 3.6, 3.8, 3.9, 3.11 (**5 versions**)
- Fedora 38: Python 2.7, 3.6-3.12 (**8 versions**)
- RHEL: one numpy package per Python version (**5 packages**)

♥️ Stable ABI ♥️

- Added to **Python 3.2** (2011)
- Used by **550+** PyPI packages
- cryptography, PySide, PyO3 (**Rust**), vim, ...
- WIP: **nanobind** (C++), pybind11 fork
- One package per {OS+libc, arch}:
  ```
  manylinux2_aarch64
  musllinux_x86_64 (Alpine Linux)
  win32
  macosx-universal2
  ```

Red Hat

C API Statistics

- **37 000** lines
- **227** header files
- **941** public exported functions

`https://pythoncapi.readthedocs.io/stats.html`

Present

Complete Limited C API

Python 3.11 adds:
- **Py_buffer**

Python 3.12 adds:
- PyType_From**Metaclass**()
- Py**Vectorcall**_Call()
- PEP 697: Extending Opaque Types

Test Limited C API

- Python 3.10: add `Misc/`**`stable_abi.txt`** and build **xxlimited** with `Py_LIMITED_API`

- Python 3.11: add **test_stable_abi_**`ctypes`

- Python 3.12: add **heap types relative** and **vectorcall** tests

- Python 3.13: add **Argument Clinic** tests

Remove private funcs
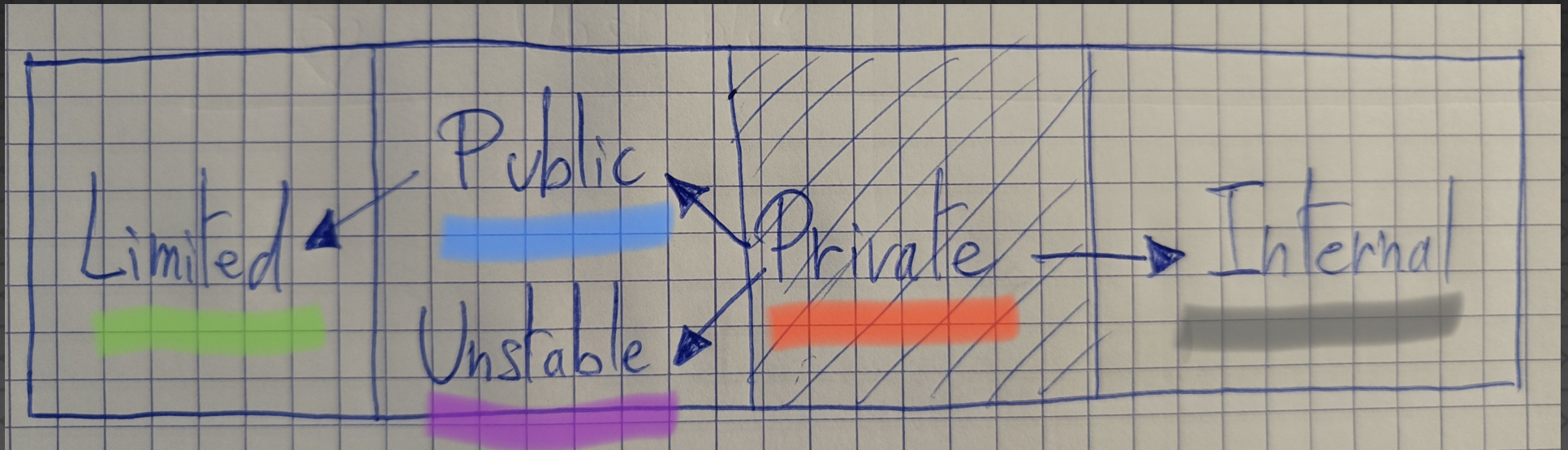
- Move **300** private functions to the internal C API:

  `Include/cpython/` → `Include/`**`internal`**`/`

- **No longer exported**

- Exported: **385** (3.12) → **85** (main)

- Public API requires **doc** and **tests**

- Add **`PyLong_AsInt`**`()` and `Py_IsFinalizing()`

Remove private funcs

Stdlib issues

- **FIXED** (3.10): Limited API incompatible with `debug build`

- **FIXED** (3.12): Missing **PyMemberDef**

- Mostly **FIXED** (3.12): **static** types

- WIP: **Performance**: vectorcall, Argument Clinic

Modernize Stdlib ext

- Static types: 22% (59/226)
  Heap types: 78% (207/266)

- Legacy init API: 15% (21/141)
  Multi-phase init: 85% (120/141) PEP 489

- 5/363 C files (tests) use Py_LIMITED_API

Fix the C API

- **Add** new APIs fixing known issues:
  `PyWeakref_Get`**`Ref`**`()`: borrowed ref
  `PyDict_GetItemRef()`: error checking

- Add to **pythoncapi-compat**

- **Deprecate** old API

- **Help** affected projects (report issue, fix)

- If needed, keep old API longer

- **Remove** old API

No Python 4 needed

- New API usable on old Python with **pythoncapi-compat**

- Updated code **works on old and new** Python

- **Limitation**: new features not available on old Python
  3.8: Vectorcall
  3.12: PyType_From**Metaclass**()

No Python 4 needed

- Spread API changes over multiple Python versions

- **Limit** changes **per release**: give time to 3rd party code to update

- Tooling and good doc help

- **Code search** to list affected projects: search_pypi_top.py

- Tool for **text replace**: upgrade_pythoncapi.py

Red Hat

No Python 4 needed

- **Incremental** approach: **adapt speed** depending on feedback

- **Revert** if maintainers **need more time**

- `Py_TYPE()` change took **3 years**: May 2020-October 2022

- It takes **months** to get a Cython/numpy release, longer for other projects

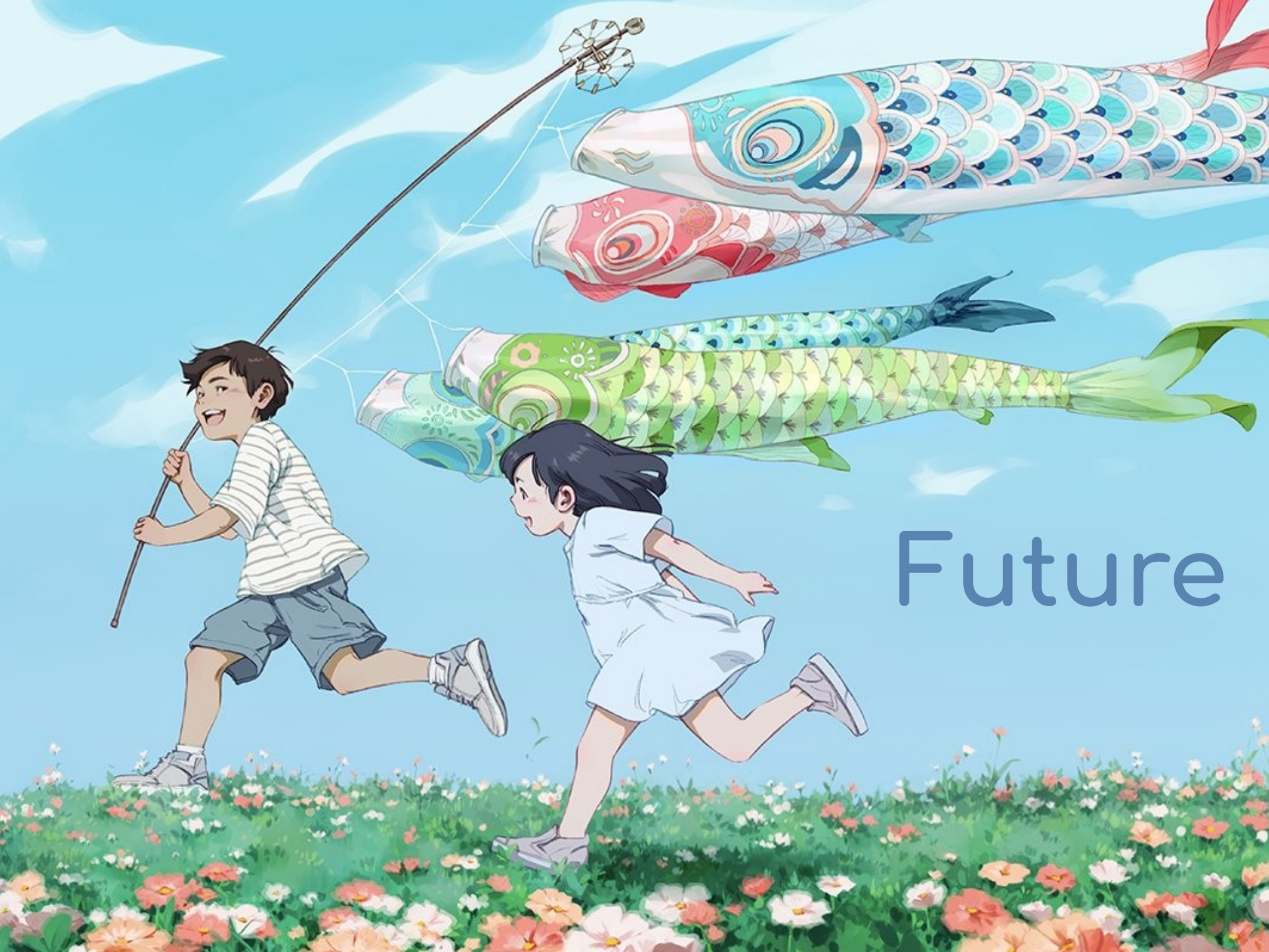- C extensions maintainers **burnout**

Red Hat

Guidelines

- Don't return **borrowed references** (NoGIL)

- Design with stable ABI in mind

- `PyDict_GetItemRef(dict, key, &value)`
  -1: error
   0: not found
   1: found
  No need to check for `PyErr_Occurred()`

Future

Stdlib uses limited API

- If no performance impact

- If not using the internal C API

- **Better tests** than xxlimited: more realistic code with existing tests

- **Candidates**: _bisect, _scproxy, _stat, _testmultiphase, _uuid, fcntl, grp, pwd, resource, syslog

- Perf impact: **_statistics** (1.8x slower). It can be fixed!

Red Hat

Stable ABI by default

- When? **Now!**
- How? Change **setuptools**?
- Who? Me?
- Need an easy way to **opt-out**

Cython uses limited API

- **Experimental** build mode
- Enable it by **default**?

HPy

- Common goal: move away from implementation details

- Different implementation

- HPy also takes PyPy in account

- TODO: make **PyObject*** opaque to **converge** to a similar API, but HPy has an additional **ctx** parameteer

C extensions on day 1

C extensions built with stable ABI become available on day 1 of Python 3.13 **alpha 1**

Rather than having to wait for 1 year for Python 3.13 **final**, if not longer

Less friction

- More freedom to change **C structures**

- More freedom to **change internals**

- **Less friction** between core devs "breaking things" and users

- "It just works™"

- Past examples:
PyGC_Head and **PyInterpreterState**
Py_INCREF()/DECREF() as functions

Working Group

- Steering Council wants to delegate
- Governance to make sure that **decisions** can be taken in a **timely manner**
- PEP: **Known issues** of the C API
- PEP: **Guidelines** for new C APIs
- PEP: **Roadmap** for the C API (5 years)

Let's talk!

Sources

- Drawing by Djamila Knopf
- Python and Red Hat are registered trademarks