Python 3
10 years later
EuroPython 2018, Edinburgh
Victor Stinner
Victor Stinner

- CPython core developer since 2010
- Maintain Python for Red Hat
- Very happy user of Fedora and vim!
Autumn
Birth of Python 3000

2006: PEP 3000 “Python 3000”

Fix "Python warts":
- long vs int; new class vs old class
- int vs float division
- Unicode mess
- Comparisons
- Relative imports
Risk management

- Don't break everything, only acknowledged warts
- Have an open community process for deciding what to change
- Don't reimplement the interpreter from scratch
- Plan end of life for Python 2
2008: Python 3.0 released
First migration plan

- Run **2to3** to port your whole code base at once: you're done! ...
- Drop Python 2 is a **no-go**, modules authors care of Python 2 users!
- All **dependencies** must be Python 3 compatible
- Python 2.7 was heavily used in **production**
Technical debt

- Why should I let you work on Python 3 support?
- For all these new cool Python 3 features!
- Can we use these features?
- Well.... since we still have to support Python 2... no
Two branches in Git?

Some projects were forked to add Python 3 support.

- Same upstream, two names: dnspython → dnspython3
- Community fork: PIL → Pillow
- Upstream does not reply: MySQL-python → mysqlclient
Python 2.6 and 3.2

- Python 2.6 was the stable version when 3.0 was released
- It requires `unittest2` and more backports
- It requires heavy usage of the `six` module
- Python 3.2 requires `six.u("unicode")`; `u"unicode"` is a syntax error
Winter
Python 3.0 was released December 3, 2008. As listed on PyPI - packages in **red** don't support python 3, packages in **green** do. Hopefully one day everything will be greener.

Status: 18/200 Updated: 2011-02-15T07:51:56.183000

<table>
<thead>
<tr>
<th>Package</th>
<th>Downloads</th>
</tr>
</thead>
<tbody>
<tr>
<td>setuptools</td>
<td>3939823</td>
</tr>
<tr>
<td>zc.buildout</td>
<td>2022655</td>
</tr>
<tr>
<td>lxml</td>
<td>1476142</td>
</tr>
<tr>
<td>Paste</td>
<td>845570</td>
</tr>
<tr>
<td>distribute</td>
<td>784171</td>
</tr>
<tr>
<td>PasteDeploy</td>
<td>665090</td>
</tr>
<tr>
<td>pytz</td>
<td>613757</td>
</tr>
<tr>
<td>pip</td>
<td>569017</td>
</tr>
<tr>
<td>virtualenv</td>
<td>500445</td>
</tr>
</tbody>
</table>

Only **9%** :-((18/200)

2011: “an attempt at motivating package maintainers to port to python 3”
Big Python 2 projects

Twisted

Mercurial

Heavy usage of bytes

Incomplete Unicode support
Python 3 trolls

- Python 3 *doesn’t* bring anything
- Python 3 introduces new *Unicode* issues
- Using *bytes* is simpler
Python 2.8 idea

- Python 2.7 is alive: let’s continue the development!
- 2014: LWN article “Debating a "transitional" Python 2.8"
- “concerns that Python 3 would never take off”
- “Python 3 represents under 2% of package”
No Python 2.8!

- 2011: **PEP 404** ("PEP not found ;-)")
  "Python 2.8 Un-release Schedule"
- 2013: **80%** of top 50 projects supports Python 3
- 2014: Python 2.7 end of life extended by 5 years to **2020**
Spring
Problem #1 solved!

“How to install a dependency?”
“How to install setuptools?”

- 2011: pip 1.0 released
- 2014: Python 2.7.9 and 3.4 now come with ensurepip
- pip: defacto installer
- Linux distros with pip
New approach

- Stop promoting 2to3: don’t remove Python 2 support
- Add Python 3 support
- New tools like modernize and sixer
- Incremental changes tested by a CI
Large code base

- For legacy code bases: first add new tests to reduce the risk of regression
- Dropbox is working on mypy and typing to annotate types in their large code base
Building bridges

- 2012: Python 3.3 reintroduces \texttt{u"unicode"}
- 2015: Python 3.5 adds \texttt{bytes \% args} (PEP 461)
- More \texttt{py3k warnings} added to Python 2.7
- More 2.7 \texttt{backports}: unittest2, enum34, ...
Summer
Python 3.0 was released December 3, 2008.

As listed on PyPI, packages in red do not support Python 3, packages in green do. Hopefully, everything will be greener.

Status: 18/200 Updated: 2011-02-15T07:51:56.183009

<table>
<thead>
<tr>
<th>Package</th>
<th>Downloads</th>
</tr>
</thead>
<tbody>
<tr>
<td>setuptools</td>
<td>3939823</td>
</tr>
<tr>
<td>zc.buildout</td>
<td>2022655</td>
</tr>
<tr>
<td>lxml</td>
<td>1476142</td>
</tr>
<tr>
<td>Paste</td>
<td>845570</td>
</tr>
<tr>
<td>distribute</td>
<td>784171</td>
</tr>
<tr>
<td>PasteDeploy</td>
<td>665090</td>
</tr>
<tr>
<td>pytz</td>
<td>613757</td>
</tr>
<tr>
<td>pip</td>
<td>569017</td>
</tr>
<tr>
<td>virtualenv</td>
<td>500445</td>
</tr>
</tbody>
</table>

2011: 9% :-( (18/200)
Python 3.0 was released December 3, 2008. As listed on PyPI - packages in red don't support Python 3, packages in green do. Hopefully one day everything will be greener.

Status: 190/200 Updated: 2018-02-01T04:31:06.140930

**Package** | **Downloads**
--- | ---
`simplejson (py3k)` | 232795919
`six` | 110953835
`requests` | 103009497
`pip` | 82348965
`python-dateutil` | 68994216
`virtualenv` | 64726493
`boto` | 60814740
`pyasn1` | 57167291
`pbr` | 53989984

2018: **95% :-)** (190/200)
3.6 faster than 2.7

Results normalized to lto-pgo latest in branch '2.7'

3.6 faster than 2.7
lower = faster
INSTAGRAM ON PYTHON3

CPU

Saving of 12% (on uwsgi/django)

MEMORY

Saving of 30% (on celery)

PORTLAND, OREGON
MAY 17 - 25 2017

Lisa Guo and Hui Ding Keynote
Python 2.7 WONTFIX

Backward compatibility prevents to fix Python 2.7 bugs:

- **Unicode** support
- Hash **not randomized** by default
- subprocess is **not thread safe**
- threading.RLock is **not signal safe**
- Internal clocks are **not monotonic**
Fixed in Python 3

- 3.3: time.monotonic() (PEP 418)
- 3.4: file descriptors non-inheritable, fork+exec safety (PEP 446)
- 3.5: retry syscalls on EINTR (PEP 475)
- “We are aware of the code breakage this is likely to cause, and doing it anyway for the good of mankind.” – Guido van Rossum PEP 446 approval
2.7 → 3.7 new modules

asyncto, concurrent.futures, contextvars, dataclasses, enum, ensurepip, faulthandler, importlib, importlib.resources, ipaddress, lzma, pathlib, secrets, selectors, statistics, tkinter.ttk, tracemalloc, typing, unittest.mock, venv, zipapp

😍 21 new modules 😍
f-string (PEP 498)

>>> name = "world"; print(f"Hello {name}!")
Hello world!

>>> print(f"Hello {name.title()}!")
Hello World!

>>> x = 1; y = 2; print(f"{x} + {y} = {x + y}")
1 + 2 = 3

>>> msg = f"{1+2}"; print(msg)
3
```
def generator():
    yield from range(5)

async def coroutine():
    return await async_read() 

async def async_generator():
    yield ...

[... async for it in async_gen()]
[await func() for func in funcs()]```
New Python 3 syntax

- def func(arg, *, kw_only=None): ...
- print(msg, file=sys.stderr, end="")
- one, *tail = range(5)
  cmd = ['python3', *args, 'script.py']
- mydict = {"key": "value", **other_dict}
New Python 3 syntax

- `million = 1_000_000`
- `x: int = 5`
- `with open(...) as infp, open(...) as outfp: ...`
- `bytes % args`
- `matrix_multiplication = a @ b`
Bury Python 2?

- Fedora 23 (2015), Ubuntu 17.10 (2017): no python2 in the base system
- python3statement.org
- pythononclock.org
- 2017: IPython 6.0 and Django 2 are Python 3 only
RHEL 7.5: Python 2 has been deprecated and will be replaced with Python 3 in the next Red Hat Enterprise Linux major release.

Software Collections (SCL):
- Python 3.6: RHEL/Centos 7
- Python 3.5: RHEL/Centos 6 & 7
- Supported 3 years (ex: 2.7 SCL gone)
Javascript approach

- Daniel Esposti’s talk "Evolution or stagnation programming languages" (last Friday at Pycon IT)
- Javascript language evolved without breaking the backward compatibility
- Transpilling using Babel & polyfill: run newer Javascript on old JS VM
Python 4?
Questions?

Reach me at:

- Email: vstinner@redhat.com
- Twitter: @VictorStinner
- IRC (Freenode): vstinner
CPython sprint this weekend: join us!
Sources, copyrights

- Autumn: https://www.flickr.com/photos/visualpanic/3035384225/
- Winter: https://www.flickr.com/photos/41848869@N04/8511091946/
- Spring: https://www.flickr.com/photos/kubina/448485266/
- Summer: https://www.flickr.com/photos/freaky_designz/14385194484/
- Red Hat and Python are registered trademarks.