(Things we learned and approaches we took)

Converting 300,000 Lines of Code from Python 2 to 3

Nick Radcliffe
Stochastic Solutions Limited
http://stochasticsolutions.com

1st January, 2020

2 RIP

Stochastic Solutions

Miró

300k LOC

Python2

UTF-8 Sandwich

GOAL

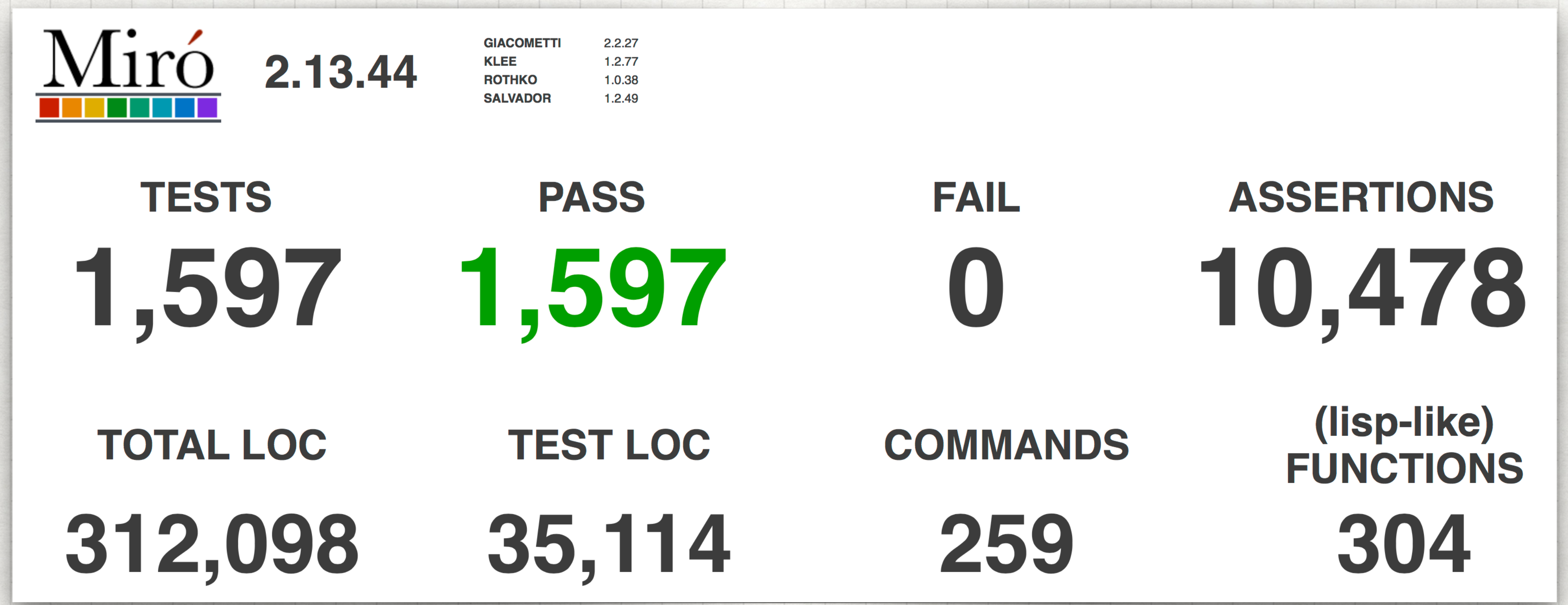
single code base

Run in Python2 & 3 . . .

. . . & in Python2 with unicode_literals

. . . and share data

1. Tests



Miró 2.13.44

GIACOMETTI	2.2.27
KLEE	1.2.77
ROTHKO	1.0.38
SALVADOR	1.2.49

TESTS	PASS	FAIL	ASSERTIONS
1,597	1,597	0	10,478

TOTAL LOC	TEST LOC	COMMANDS	(lisp-like) FUNCTIONS
312,098	35,114	259	304

2. __future__ imports

```
# -*- coding: utf-8 -*-

from __future__ import division
from __future__ import print_function
from __future__ import absolute_import
```

3. unicode literals

```
from __future__ unicode_literals
```

Code to toggle this on and off during development

4. min/max

```
def listmin(L):
    """
    Returns min of an iterable L,
    ignoring null (None) values.
    If all values are null, return None.
    """
    values = [v for v in L if v is not None]
    return min(values) if values else None
```

5. UTF8Definite, UnicodeDefinite

```
isPython2 = sys.version_info.major < 3

if isPython2:
    bytes_type = str
    unicode_type = unicode
else:
    bytes_type = bytes
    unicode_type = str


def UnicodeDefinite(s):
    return unicode_type(s, UTF8) if type(s) is bytes_type else s


def UTF8Definite(s):
    return s.encode(UTF8) if type(s) is unicode_type else s
```

6. MiroStrDefinite, NonMiroStrDefinite

```
if isPython2 and type('') == bytes_type:
    MiroStrDefinite = UTF8Definite
    NonMiroStrDefinite = UnicodeDefinite
else:
    MiroStrDefinite = UnicodeDefinite
    NonMiroStrDefinite = UTF8Definite
```

7. mirostr, nonmirostr

```
if isPython2 and type('') == bytes_type:
    mirostr = bytes_type
    nonmirostr = unicode_type
else:
    mirostr = unicode_type
    nonmirostr = bytes_type

if isPython2:
    long_type = long
else:
    long_type = int
```

8. cgi.escape, url lib.unquote

```
if isPython2:
    from cgi import escape as htmlescape
    from urllib import unquote
else:
    from html import escape as htmlescape
    from urllib.parse import unquote
```

9. re_escape

r.py:

```
from __future__ import print_function
import re
s = r'^a_(\d)$'
print(re.escape(s))
```

```
$ python2 r.py
\^a\_\(\\d\)\$
```

```
$ python3 r.py
\^a_\(\\d\)\$
```

10. range, keys, items & zip

range, keys, items & zip are all generators in Python3

```
for i in range(10):
    for k in d.keys():
        for z in zip(a, b):
```

✓

```
range(10)[3]
zip(a, b) == [(1, 2), (3, 4)]
for (k, v) in d.items():
    if v is None:
        del d[k]
```

✗

```
list(range(3)) == [0, 1, 2]
```

✓

11. str(float)

p.py:

```
from __future__ import print_function
import math
print(str(math.pi))
```

```
0 godel:$ python2 p.py
3.14159265359
0 godel:$ python3 p.py
3.141592653589793
```

12. it.next()

```
# next.py:
form __future__ import print_function
def g():
    for i in range(10):
        yield i

it = g()
print(next(it))
print(it.next())
```

```
$ python2 next.py
0
1

$ python3 next.py
0
Traceback (most recent call last):
  File "next.py", line 8, in <module>
    print(it.next())
AttributeError: 'generator' object has no attribute 'next'
```

13. Dummy variables

```python
# comp.py
from __future__ import print_function
a = [i * i for i in range(10)]
print(i)
```

```
$ python2 comp.py
9
```

```
$ python3 comp.py
Traceback (most recent call last):
  File "comp.py", line 4, in <module>
    print(i)
NameError: name 'i' is not defined
```

14. Pickles

Pickles are not really compatible between Python 2 & 3

and are not really meant for long-term storage

We switched to JSON.

Faster and more portable

15. Tools

USED EXTENSIVELY

pep8 — *Checks conformance to PEP8*
pyflakes — *Checks for real problems in Python code (almost always real)*
tdda library — *Reference testing capability*

USED OCCASIONALLY

pylint — *Very pedantic linter for Python style & correctness*
coverage.py — *Test coverage checker*

DID NOT USE

six — *Library to aid Python2/Python3 compatible code*
2to3 — *Automatic Python2 to Python3 converter*

njr@StochasticSolutions.com

http://tdda.info

https://github.com/tdda

#tdda*

@tdda0 @njr0

* *tweet (DM) us email address for invitation*
Or email me.

Correct interpretation: Zero

Error of interpretation: Letter "Oh"