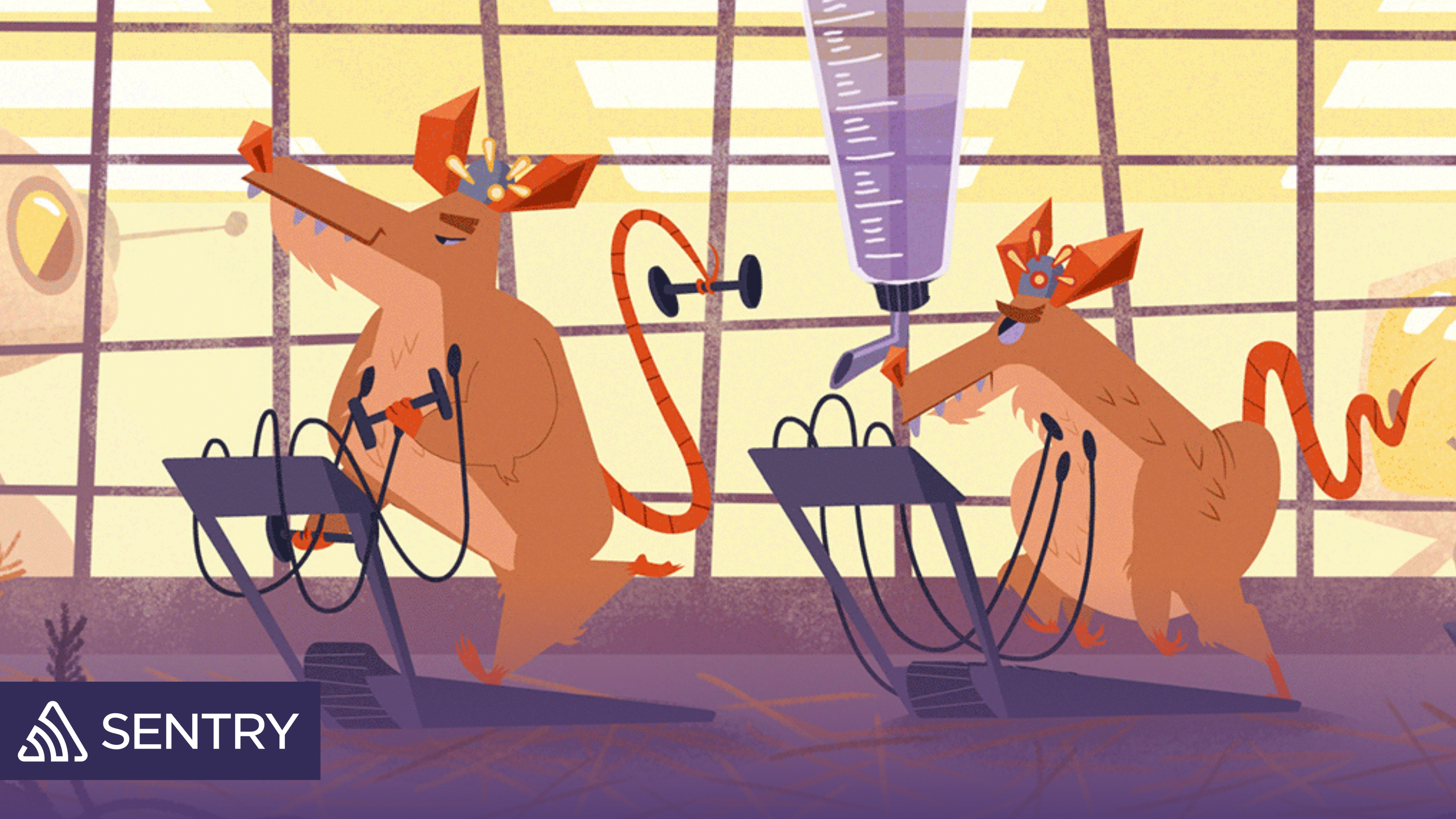
DEBUG IS
THE NEW
RELEASE

THE UNEXPECTED BENEFITS OF SLOW LANGUAGES

ARMIN @MITSUHIKO
RONACHER / FLASK,
WERKZEUG, JINJA,
CLICK, ... / DIRECTOR
OF ENGINEERING AT
SENTRY / LOVES OSS


SENTRY

1. developer experience matters
2. the ability to debug matters
3. debugging does not stop when shipping a release

```
Python 3.7.4 (default, Jul  9 2019, 18:13:23)
[Clang 10.0.1 (clang-1001.0.46.4)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import sys
>>> a_variable = 'Hello World!'
>>> sys._getframe().f_locals['a_variable']
'Hello World!'
```

RUNTIME INTROSPECTION IS POWERFUL

RUNTIME
INTROSPECTION
IS SLOW

RUNTIME
INTROSPECTION
IS IMPORTANT

“WORKS ON MY MACHINE”

MILLIONS OF BROWSER SESSIONS
HUNDREDS OF COUNTRIES

IOT DEVICES
MOBILE PHONES

LETS

DEBUG

IN PROD

you need basic debugging and introspection in production

PROD VS DEBUG PERF

let's talk about runtimes ...

Simple Interpreter
JIT compiled
AOT compiled

- these are examples
- not scientific
- not entirely comparable


INTERR
PRETER

A Simple Interpreter: CPython

```
>>> import dis
>>>
>>> def add_numbers(a, b):
...  return a + b
...
>>> dis.dis(add_numbers)
  2           0 LOAD_FAST                0 (a)
              2 LOAD_FAST                1 (b)
              4 BINARY_ADD
              6 RETURN_VALUE
```

```
while ... {
    switch (...) {
        case TARGET(LOAD_FAST): {
            PyObject *value = GETLOCAL(oparg);
            if (value == NULL) {
                format_exc_check_arg(tstate, PyExc_UnboundLocalError,
                                     UNBOUNDLOCAL_ERROR_MSG,
                                     PyTuple_GetItem(co->co_varnames, oparg));
                goto error;
            }
            Py_INCREF(value);
            PUSH(value);
            FAST_DISPATCH();
        }

}
```

```
case TARGET(BINARY_ADD): {
    PyObject *right = POP();
    PyObject *left = TOP();
    PyObject *sum;
    if (PyUnicode_CheckExact(left) &&
             PyUnicode_CheckExact(right)) {
        sum = unicode_concatenate(tstate, left, right, f, next_instr);
    }
    else {
        sum = PyNumber_Add(left, right);
        Py_DECREF(left);
    }
    Py_DECREF(right);
    SET_TOP(sum);
    if (sum == NULL)
        goto error;
    DISPATCH();
}
```

there is a lot of compiled code executing every instruction

```
import sys

def failing_func():
    raise Exception('Oh noes')

def catching_func():
    try:
        failing_func()
    except Exception:
        pass

def stacktrace_making_func():
    try:
        failing_func()
    except Exception:
        sys.exc_info()
```

```
mitsuhiko at argus in /tmp
$ python -mtimeit -s 'from test import catching_func as x' 'x()'
1000000 loops, best of 3: 1.34 usec per loop

mitsuhiko at argus in /tmp
$ python -mtimeit -s 'from test import stacktrace_making_func as x' 'x()'
1000000 loops, best of 3: 1.44 usec per loop
```

7% SLOWER


JIT

JIT Compiled Interpreter: V8

```
function throwingFunc() {
  throw new Error('Oh noes');
}

function catchingFunc() {
  try {
    throwingFunc();
  } catch (err) {}
}

function stacktraceMakingFunc() {
  try {
    throwingFunc();
  } catch (err) {
    return err.stack;
  }
}
```

```
catching           x 160,895 ops/sec ±2.30% (60 runs sampled)
stacktrace making x  26,495 ops/sec ±1.98% (86 runs sampled)
```


AOT

Native Code: clang

well what's a stack trace anyways?

STACK
WALKING

there is a little DWARF
in your computer

stack unwinding: go to where a function would return to

```
0   libsystem_kernel.dylib          0x00007fff61bc6c2a 0x7fff61bc6000 + 3114
1   CoreFoundation                  0x00007fff349f505e 0x7fff349b9000 + 245854
2   CoreFoundation                  0x00007fff349f45ad 0x7fff349b9000 + 243117
3   CoreFoundation                  0x00007fff349f3ce4 0x7fff349b9000 + 240868
4   HIToolbox                       0x00007fff33c8d895 0x7fff33c83000 + 43157
5   HIToolbox                       0x00007fff33c8d5cb 0x7fff33c83000 + 42443
6   HIToolbox                       0x00007fff33c8d348 0x7fff33c83000 + 41800
7   AppKit                          0x00007fff31f4a95b 0x7fff31f30000 + 108891
8   AppKit                          0x00007fff31f496fa 0x7fff31f30000 + 104186
9   AppKit                          0x00007fff31f4375d 0x7fff31f30000 + 79709
10  YetAnotherMac                   0x0000000108b7092b 0x10864e000 + 5384491
11  YetAnotherMac                   0x0000000108b702a6 a_function_here + 64
12  libdyld.dylib                   0x00007fff61a8e085 start + 0
13  YetanotherMac                   0x00000000000ea004 main (main.m:16)
```

okay it's “fast”, but it's also pretty bad

because better would be much slower

- unwinding on device
- deferred symbolication
- pain, suffering and disappointment

want stack traces? need to capture when exceptions are thrown

1. debuggability incurs runtime cost
2. JIT/AOT optimizations break down
3. If you want debug functionality in production, percentage performance loss matters

Sentry exists, because cheap in-production debugging is amazing and not much slower in Python

BUT ARMIN, STACK TRACES ARE FAST!!!111

but we expect more

ReadTimeout

SafeHTTPSConnectionPool(host='hooks.slack.com', port=443): Read timed out. (read timeout=5)

mechanism	logging	handled	yes

sentry/net/http.py in **request** at line **150**

```
145.
146.
147. class Session(_Session):
148.     def request(self, *args, **kwargs):
149.         kwargs.setdefault("timeout", 30)
150.         response = _Session.request(self, *args, **kwargs)
151.         # requests' attempts to use chardet internally when no encoding is
152.         # and we want to avoid that slow behavior
153.         if not response.e
154.             response.enco
155.         return response
```

args	[]
kwargs	{ allow_redirects: False, data: { payload: '{"username":"[redacted]","attachments":[{"color":"#f43f20","fields": [{"short":false,"value":"t.selector(lNDM86b3a!s!utf-8/http://[redacted]/index)","title":"Culprit"}, {"short":true,"value":"factoring","title":"Project"}],"fallback":"[factoring] Error: http error with status code: 0 and body: {}","title_link":"https://sentry.io/organizations/[redacted]? referrer=slack","title":"Error: http error with status code: 0 and body: {}"}]}' }, method: 'POST', timeout: 5, url: u'https://hooks.slack.com/services/[redacted]', verify: True }
self	<sentry.net.http.SafeSession object at 0x7f532fd28fd0>

sentry/http.py in **safe_urlopen** at line **117**

sentry_plugins/slack/plugin.py in **notify** at line **244**

sentry/plugins/bases/notify.py in **rule_notify** at line **112**

sentry/utils/safe.py in **safe_execute** at line **24**

Exception Data

Stacktraces

Source Code

Local Variables

[SHAMELESS PLUG BUT IT'S FOR CONTEXT]

value what you have,
Python developers

WHAT DO
WE HAVE?

```
>>> import sys
>>> sys._current_frames()
{4656870848: <frame at 0x109177d50, file '<stdin>', line 1, code <module>>}
```

```
>>> import sys
>>> sys._getframe().f_locals
{'__annotations__': {},
 '__builtins__': <module 'builtins' (built-in)>,
 '__cached__': None,
 '__doc__': None,
 '__loader__': <_frozen_importlib_external.SourceFileLoader object at 0x1090d55d0>,
 '__name__': '__main__',
 '__package__': None,
 '__spec__': None,
 'sys': <module 'sys' (built-in)>}
```

```
>>> try:
...     1/0
... except Exception as e:
...     e.__traceback__
...
<traceback object at 0x1093559b0>
```

```
from threading import Thread
old_start = Thread.start
Thread.start = make_new_start(old_start)
```

you can also attach a debugger, run some code and start a reverse python shell on a running process

& Python 3.7 has execution contexts (context vars)

WHAT WILL THE FUTURE BRING

ASK

YOUR

QUESTIONS

— I DON'T BITE —