Pythonic APIs

Anthony Baxter

anthonybaxter@gmail.com

@anthonybaxter

Sunday, November 21, 2010

notes

Some good and bad examples

and speaking of bad examples...

O Hai!

started with python ‘92 or ‘93

Sunday, November 21, 2010

I forget which. I do recall that back then, Guido wrote a webcrawler in Python that crawled the entire web, and crashed a lot of machines.

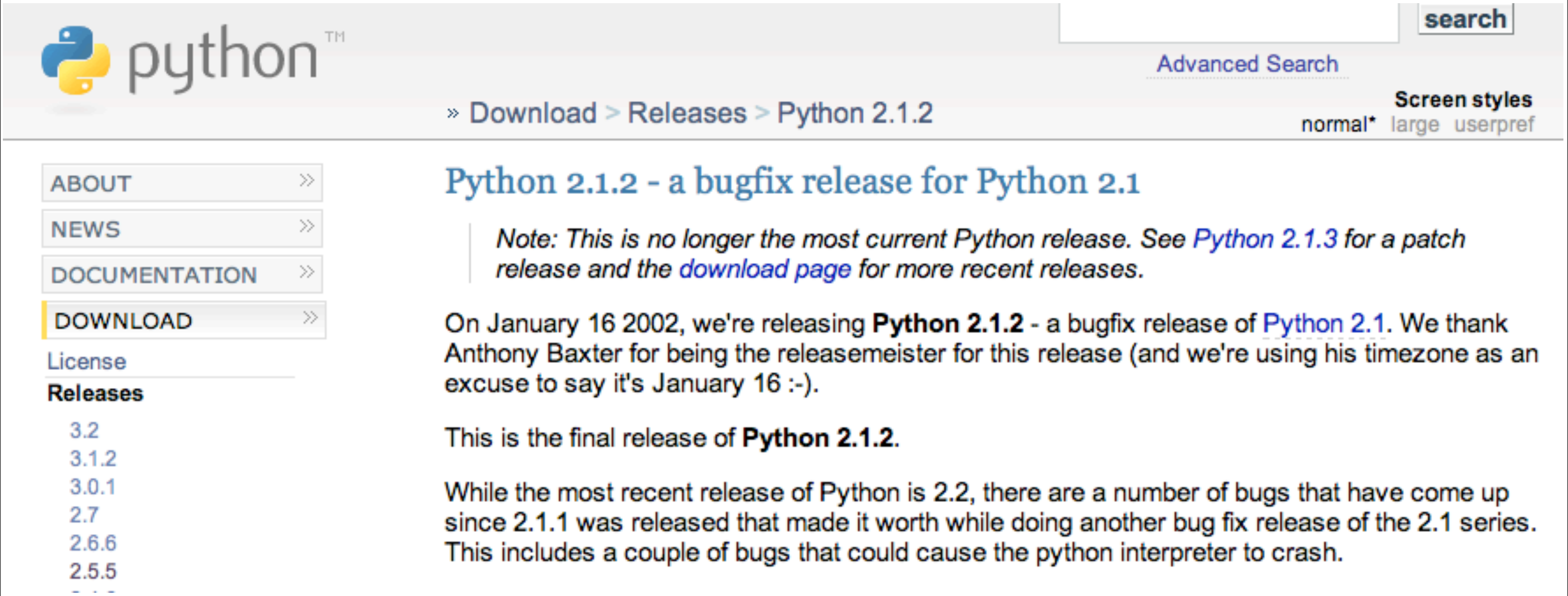
I’ve written a lot of Python

and used a lot more

Python is like a virus

Python gets into your brain and makes you annoyed at other languages. And like all good viruses, it makes you want to spread it.

inevitably



Python 2.1.2 - a bugfix release for Python 2.1

> *Note: This is no longer the most current Python release. See Python 2.1.3 for a patch release and the download page for more recent releases.*

On January 16 2002, we're releasing **Python 2.1.2** - a bugfix release of Python 2.1. We thank Anthony Baxter for being the releasemeister for this release (and we're using his timezone as an excuse to say it's January 16 :-).

This is the final release of **Python 2.1.2**.

While the most recent release of Python is 2.2, there are a number of bugs that have come up since 2.1.1 was released that made it worth while doing another bug fix release of the 2.1 series. This includes a couple of bugs that could cause the python interpreter to crash.

did that for 5 years

primary language from 1996-2007, now have to do Java and C++ as well, boo

Then Google ate my free time

some APIs

On of the ways of spreading Python is to make it possible to do new stuff, simply.
I’ve built far too many APIs and libraries for Python. Mostly I build them to get something done, then eventually get bored of them and hand them on or move on.
Learn from my mistakes.

miniSQL

Sunday, November 21, 2010

mid 90s – first of the small open embedded databases, by David Hughes. wrapped it in custom C code wrapper. Eventually taken over by Mark Shuttleworth, who used it at Thawte.

snmpy

wrapper around UCD-SNMP. wanted to manage some small ethernet switches. subsequently, used it at a large ISP to collect stats from thousands of interfaces for billing purposes.
got a bit silly eventually, I used it as a backend to gadfly so I could do SQL-style queries across network devices and interfaces.

shtoom

Sunday, November 21, 2010

voice over IP

lots of other stuff...

whenever I come across a new protocol, I spend some time implementing it. sometimes I even get around to releasing that code.

So you’re building an API

Why would you do this?

Wrap existing code

New code to achieve a purpose

Generalise existing code

Learning

I’ve done a lot of this – the easiest way I find to understand something new, especially a network protocol, is to implement it.

for the hell of it

sheer bloodymindedness

shtoom was originally written for one reason. I got sick of hearing Python called a “scripting language”. So I thought implement voice over IP in it and write a talk, just to say “shut up”

“Scripting Language, My Arse”

Sunday, November 21, 2010

I kept working on it, adding all sorts of stuff to it. It kinda grew out of control.

SIP

IVRs

audio

tftp

STUN

rtp

UPnP

codecs

ulaw
gsm06.10
G.72x
speex
...

UIs

tk
qt
gnome
wx
cocoa
text
...

by the time I got bored, I had implemented about 5 different audio capture/playback interfaces, and 7 or 8 different UI layers. As well as about 4 different firewall avoidance mechanisms. It worked on Unixes, Windows, the Mac, and sometimes even Gentoo.

~ 25K LOC

“Pythonic”

Guido is Watching You

Guido has been successful as BDFL because of his excellent taste.

“I know it when I see it”

Justice Potter Stewart,
Jacobellis v. Ohio 378 U.S. 184 (1964)

of course he was talking about obscenity, but the principle remains

What makes an API Pythonic?

principle of least surprise

EDIBLE IMAGES

principle of least surprise

Sunday, November 21, 2010

it should feel safe and comfortable and familiar to a Python programmer, even if they’ve never come across it before.

Zen of Python

Sunday, November 21, 2010

when designing an API, bear the words of Sensei Tim Peters in mind. Type ‘import this’ if you haven’t already

“Python fits in my brain”

Your API should also fit in my brain. I shouldn’t have to keep referring to the docs. Also, if I learn one bit of the API, it should be consistent enough that the rest of the API naturally follows.

Python isn’t always Pythonic

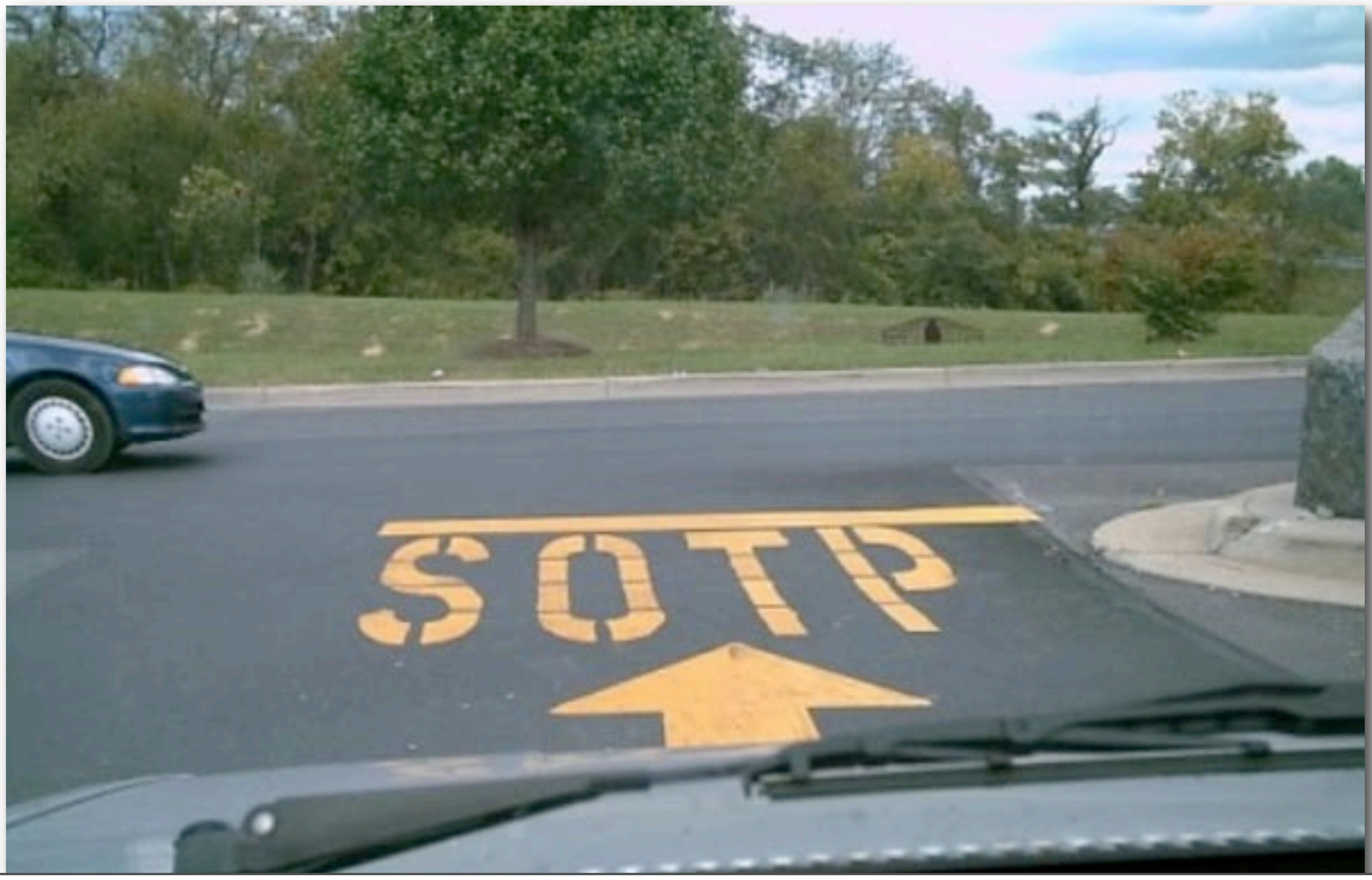
map, filter, reduce

we fixed that. list comprehensions for map and filter. reduce, we have sum(). map(None, foo), we have zip()

print >> fp

lambda

Common mistkaes



Over-ambition

I'm gonna TAKE OVER THE WORLD

“Frameworks”

Why don’t I like them?

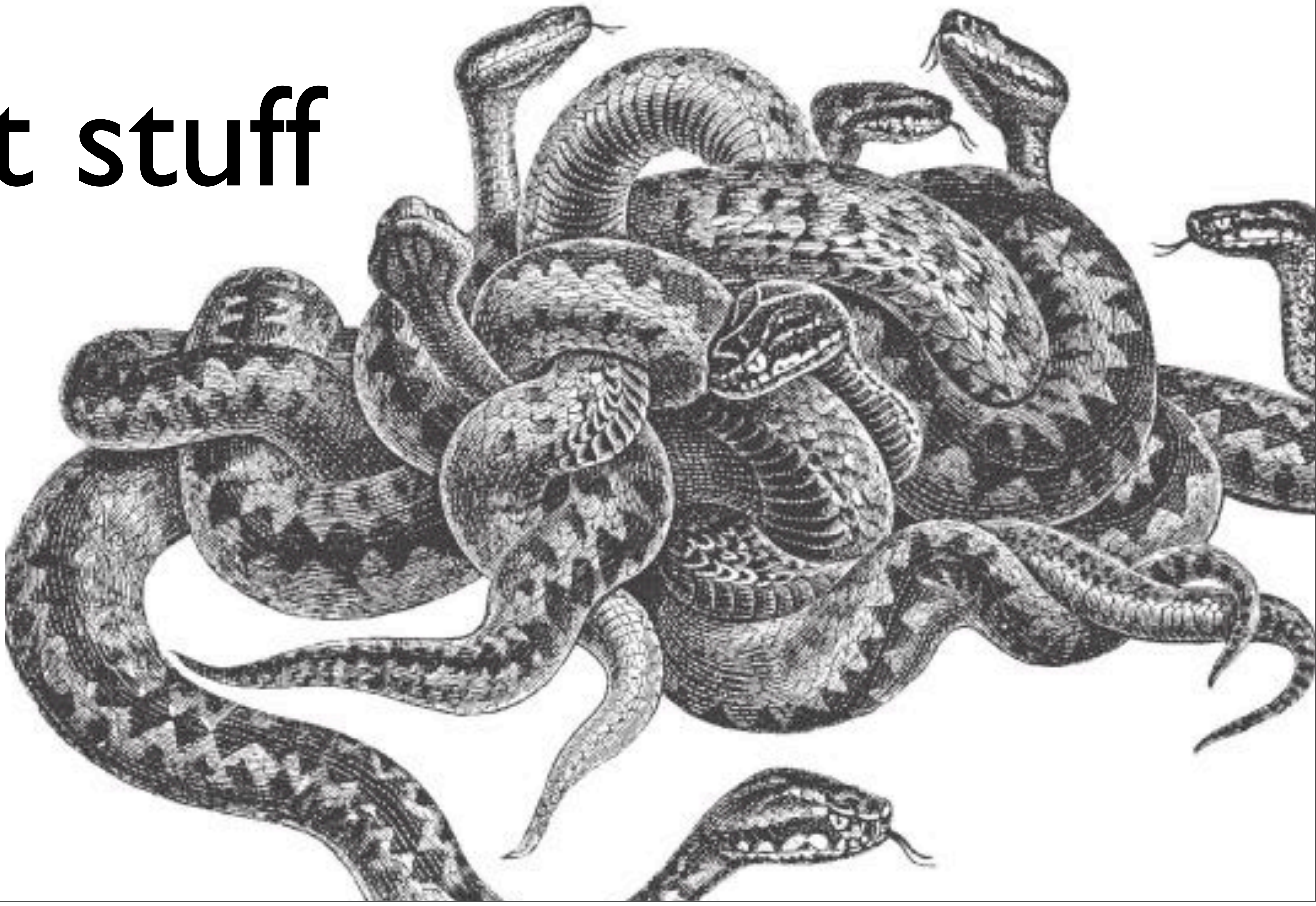
I just want one piece

e.g.

twisted

great stuff

but...



twisted has a wide variety of protocol implementations, often of high quality. but interfacing with them means buying into the whole framework in most cases. and there's often semi-heroic efforts needed to interface with other APIs that weren't designed for twisted. Twisted evangelists will probably disagree with me.

Frameworks

==

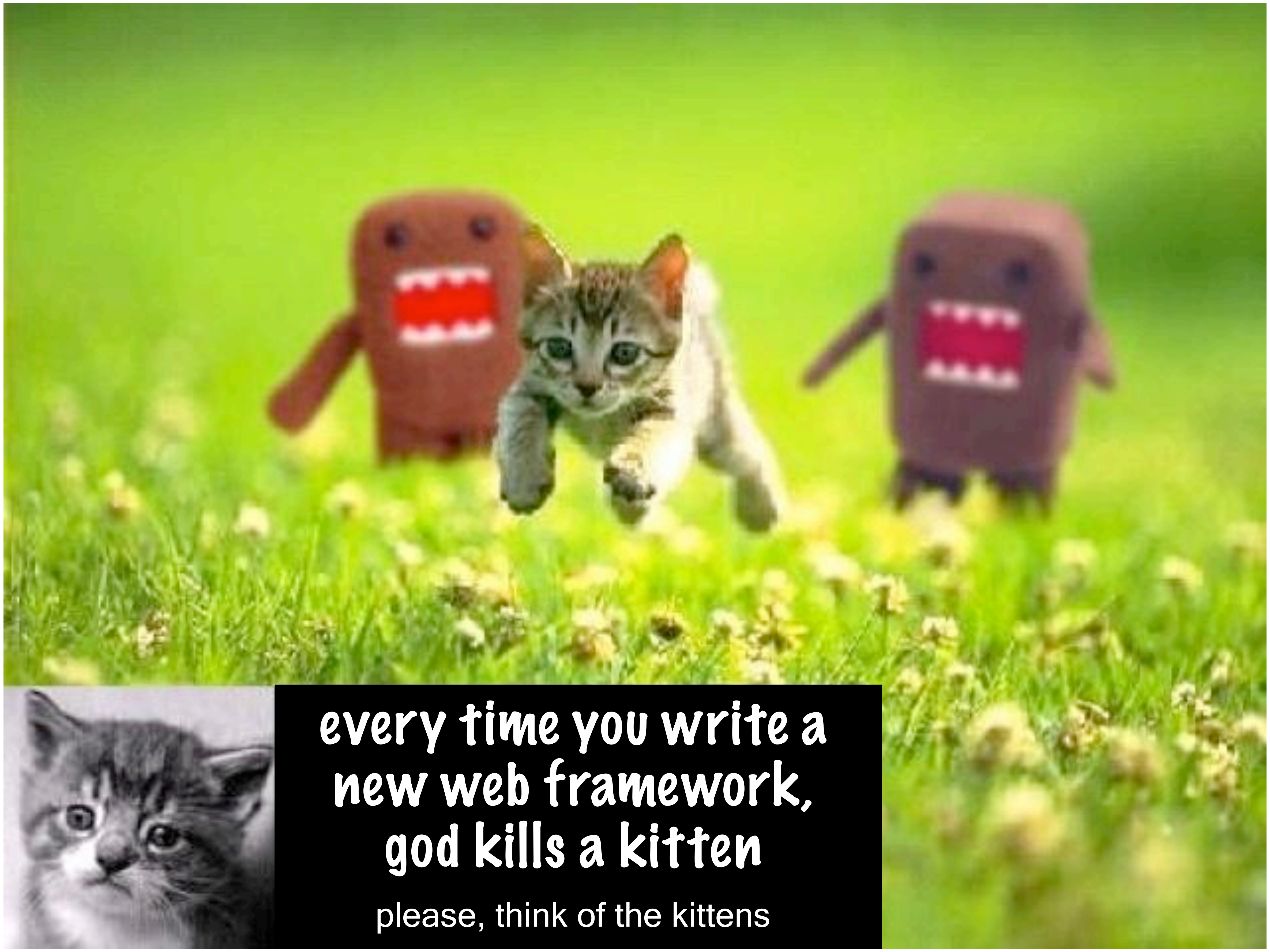
do it my way

I want to be able to pick up an individual piece of code or a library, and run with it. I don't want to have to restructure my entire application around a new framework.
This also often leads to a situation where I'm forced to pick a framework at the start of coding, before I've figured out how exactly I want to achieve my goal.

Mistaik #2: Reinventing the wheel

We don’t need:

another web
framework



every time you write a new web framework, god kills a kitten

please, think of the kittens

another #!*$^#?@& templating language

you make KITTY SAD.

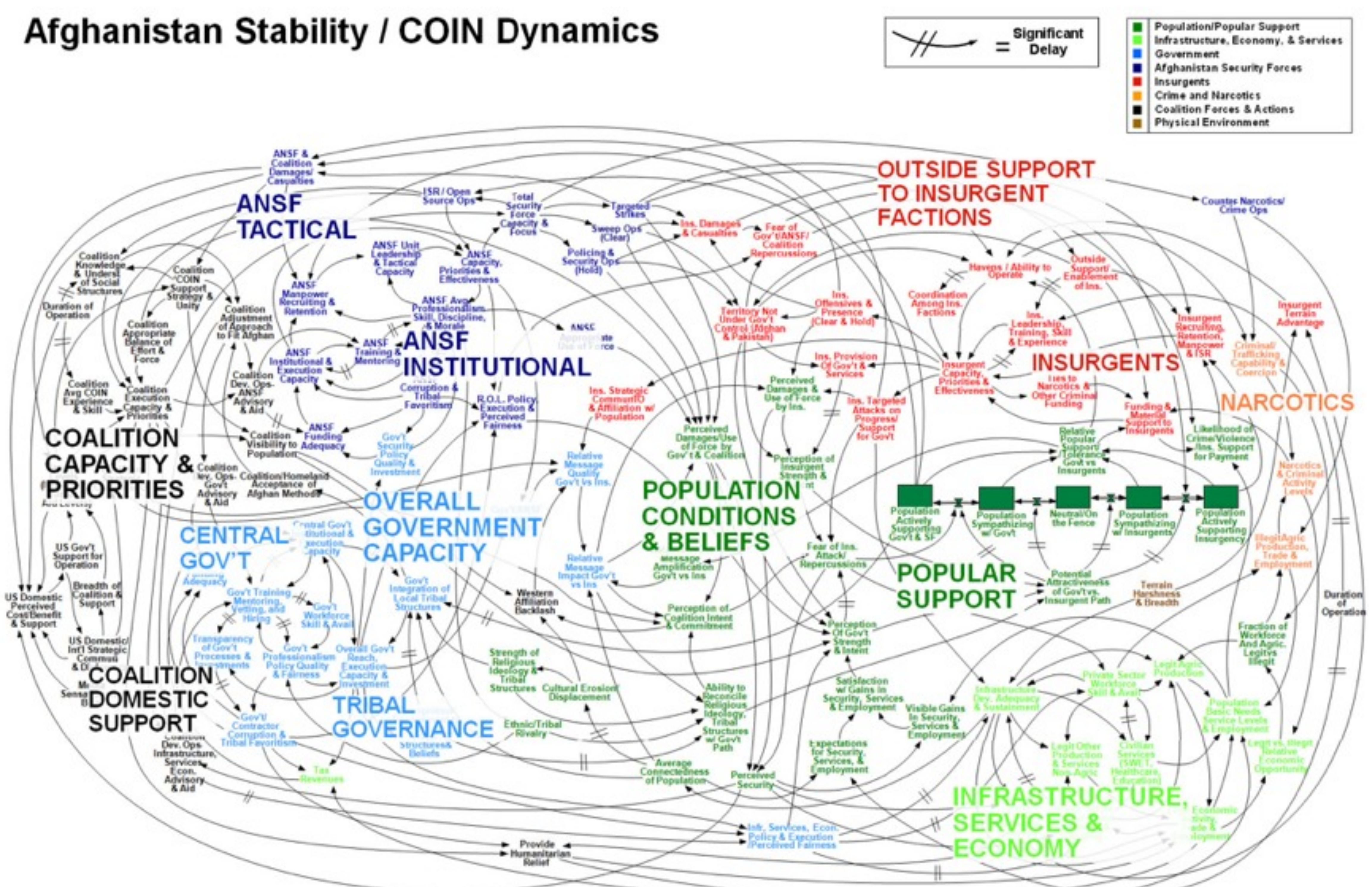
another interface for postgresql or mysql

I counted at least 6 PG adapters on pypi, and a similar number of mysql adapters.

makes no-one's life better

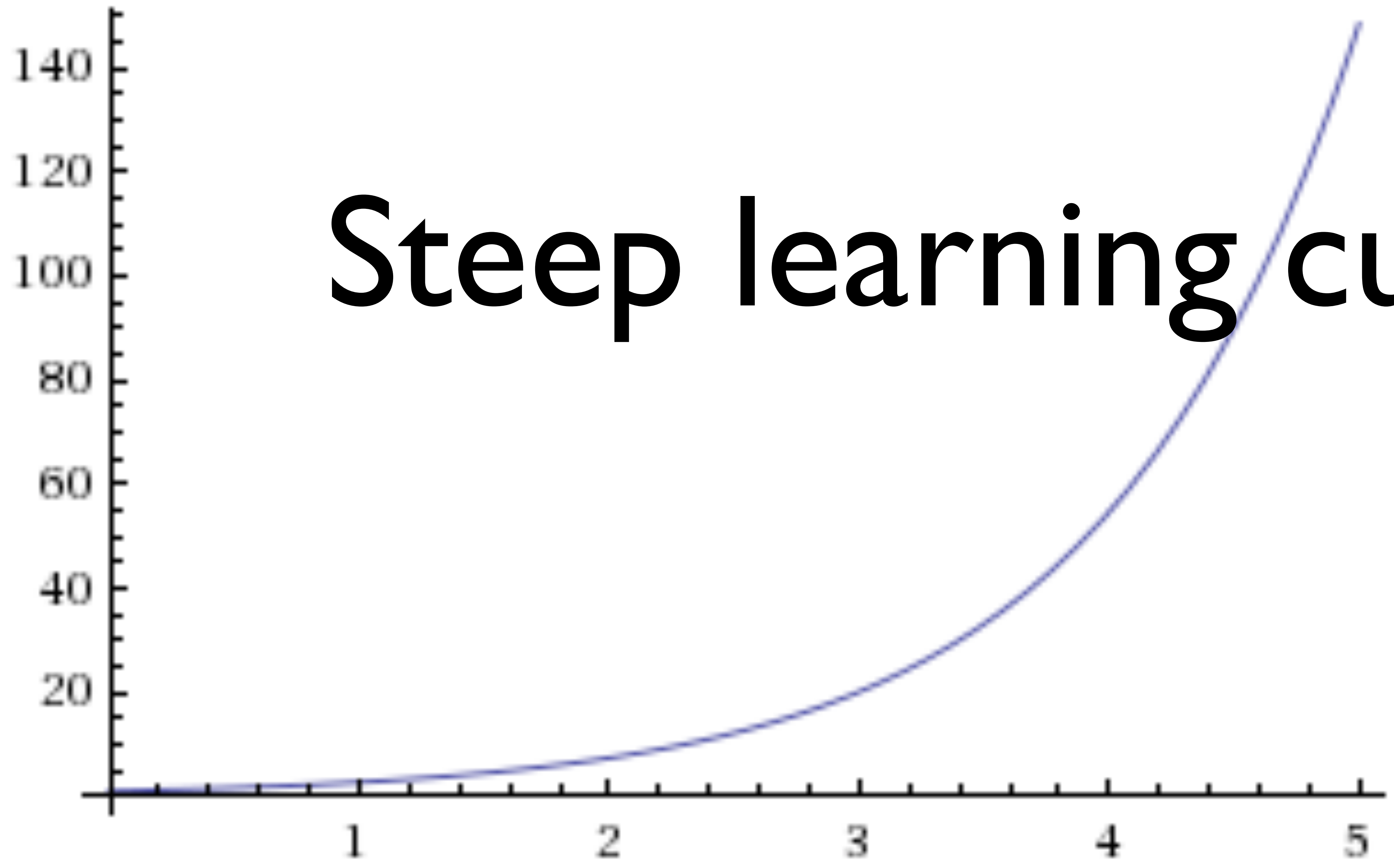
it's harder for python developers.
it's harder for you (less users).

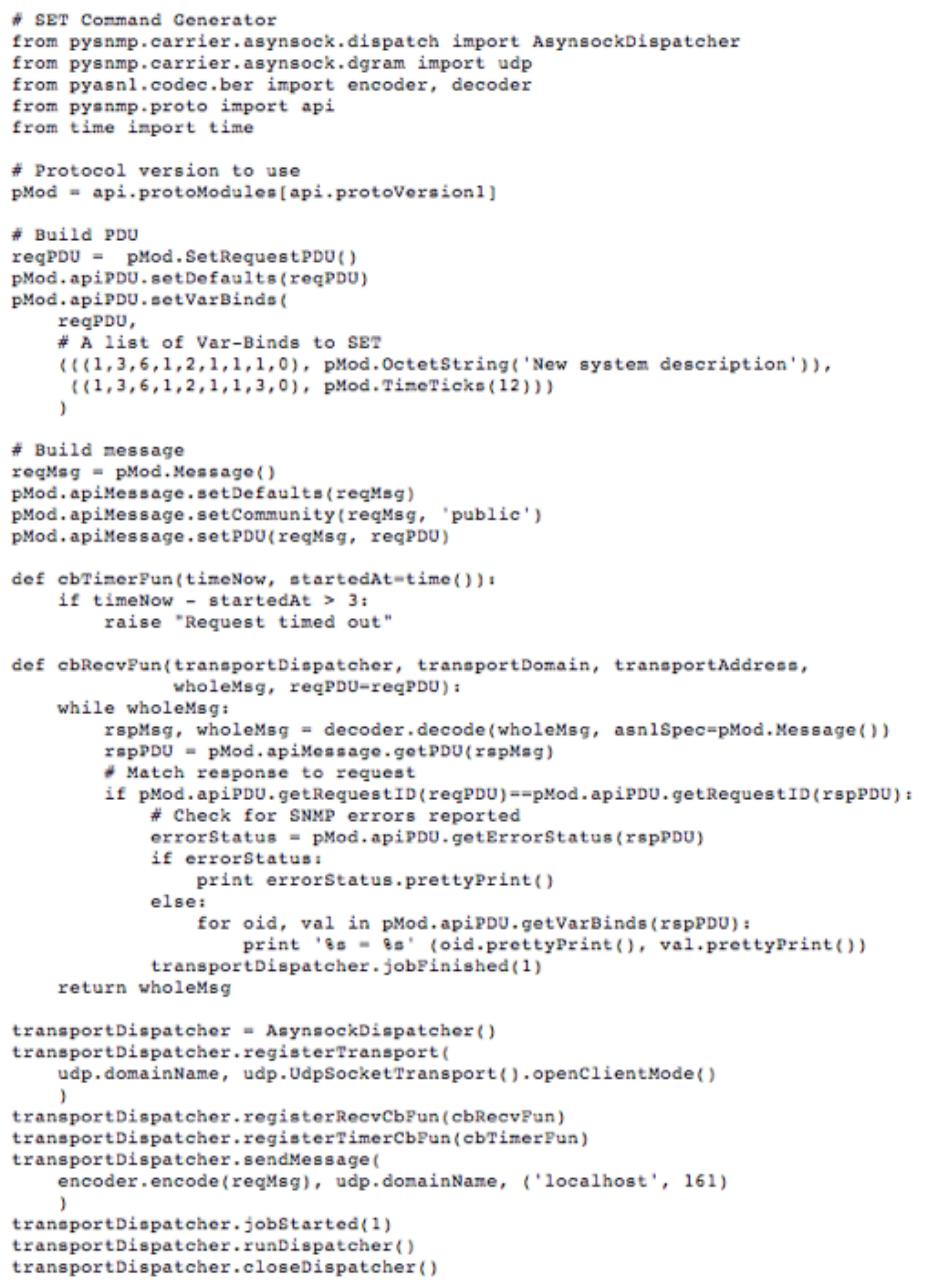
#3: complexity



Afghanistan Stability / COIN Dynamics
= Significant Delay
Population/Popular Support
Infrastructure, Economy, & Services
Government
Afghanistan Security Forces
Insurgents
Crime and Narcotics
Coalition Forces & Actions
Physical Environment
ANSF TACTICAL
ANSF INSTITUTIONAL
OUTSIDE SUPPORT TO INSURGENT FACTIONS
INSURGENTS
NARCOTICS
COALITION CAPACITY & PRIORITIES
CENTRAL GOV'T
OVERALL GOVERNMENT CAPACITY
POPULATION CONDITIONS & BELIEFS
POPULAR SUPPORT
COALITION DOMESTIC SUPPORT
TRIBAL GOVERNANCE
INFRASTRUCTURE, SERVICES & ECONOMY
Population Actively Supporting Gov't & SF
Population Sympathizing w/ Gov't
Neutral/On the Fence
Population Sympathizing w/ Insurgents
Population Actively Supporting Insurgency
Provide Humanitarian Relief
Perceived Security

Steep learning curve





```
# SET Command Generator
from pysnmp.carrier.asynsock.dispatch import AsynsockDispatcher
from pysnmp.carrier.asynsock.dgram import udp
from pyasn1.codec.ber import encoder, decoder
from pysnmp.proto import api
from time import time

# Protocol version to use
pMod = api.protoModules[api.protoVersion1]

# Build PDU
reqPDU =  pMod.SetRequestPDU()
pMod.apiPDU.setDefaults(reqPDU)
pMod.apiPDU.setVarBinds(
    reqPDU,
    # A list of Var-Binds to SET
    (((1,3,6,1,2,1,1,1,0), pMod.OctetString('New system description')),
     ((1,3,6,1,2,1,1,3,0), pMod.TimeTicks(12)))
    )

# Build message
reqMsg = pMod.Message()
pMod.apiMessage.setDefaults(reqMsg)
pMod.apiMessage.setCommunity(reqMsg, 'public')
pMod.apiMessage.setPDU(reqMsg, reqPDU)

def cbTimerFun(timeNow, startedAt=time()):
    if timeNow - startedAt > 3:
        raise "Request timed out"

def cbRecvFun(transportDispatcher, transportDomain, transportAddress,
              wholeMsg, reqPDU=reqPDU):
    while wholeMsg:
        rspMsg, wholeMsg = decoder.decode(wholeMsg, asn1Spec=pMod.Message())
        rspPDU = pMod.apiMessage.getPDU(rspMsg)
        # Match response to request
        if pMod.apiPDU.getRequestID(reqPDU)==pMod.apiPDU.getRequestID(rspPDU):
            # Check for SNMP errors reported
            errorStatus = pMod.apiPDU.getErrorStatus(rspPDU)
            if errorStatus:
                print errorStatus.prettyPrint()
            else:
                for oid, val in pMod.apiPDU.getVarBinds(rspPDU):
                    print '%s = %s' (oid.prettyPrint(), val.prettyPrint())
            transportDispatcher.jobFinished(1)
    return wholeMsg

transportDispatcher = AsynsockDispatcher()
transportDispatcher.registerTransport(
    udp.domainName, udp.UdpSocketTransport().openClientMode()
    )
transportDispatcher.registerRecvCbFun(cbRecvFun)
transportDispatcher.registerTimerCbFun(cbTimerFun)
transportDispatcher.sendMessage(
    encoder.encode(reqMsg), udp.domainName, ('localhost', 161)
    )
transportDispatcher.jobStarted(1)
transportDispatcher.runDispatcher()
transportDispatcher.closeDispatcher()
```

Sunday, November 21, 2010

this is a ‘hello world’ for pysnmp. OMG.
ideally you should be able to play at the interactive interpreter.

#4: being too clever

back to snmpy...

SNMP OIDs

```
interfaces.ifTable.ifEntry.ifIndex.1 = 1
interfaces.ifTable.ifEntry.ifDescr.1 = "Ethernet"
interfaces.ifTable.ifEntry.ifType.1 = ethernet-csmacd(6)
interfaces.ifTable.ifEntry.ifMtu.1 = 1536
interfaces.ifTable.ifEntry.ifSpeed.1 = Gauge: 10000000
```

woot - __getattr__!

so I thought using getattr would be cute. but there's a problem...

.iso.org.dod.internet.mgmt.mib-2.system.sysUpTime

here’s a full OID. where is the problem as far as using getattr?

.iso.org.dod.internet.mgmt.**mib-2**.system.sysUpTime

impedence mismatch

OID names != python identifiers

.iso.org.dod.internet.mgmt[‘mib-2’].system.sysUpTime

suddenly things got nasty

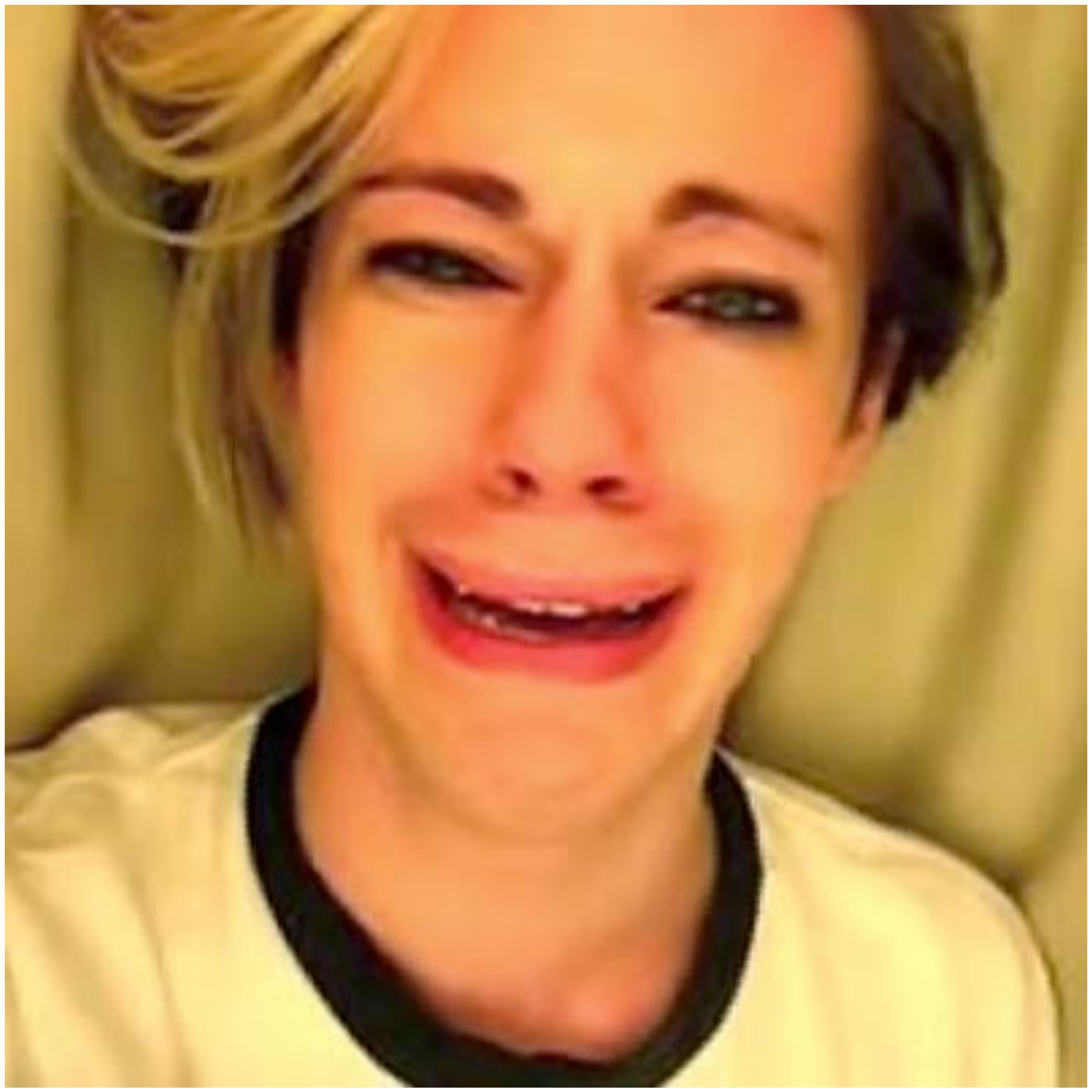
worse yet

overloading attribute lookup meant all sorts of horrible and strange errors would pop up. they made debugging awful.

cleverness will chew off your face

cleverness will bite your users

similarly



leave import alone!!!!

Sunday, November 21, 2010

please don’t play games with module import.
yes, you can write something that will fetch modules from a URL. But you shouldn’t.
if you’ve ever been bitten by this, you know how utterly irritating it can be
multiple custom importers make everyone sad

and don’t monkeypatch the stdlib

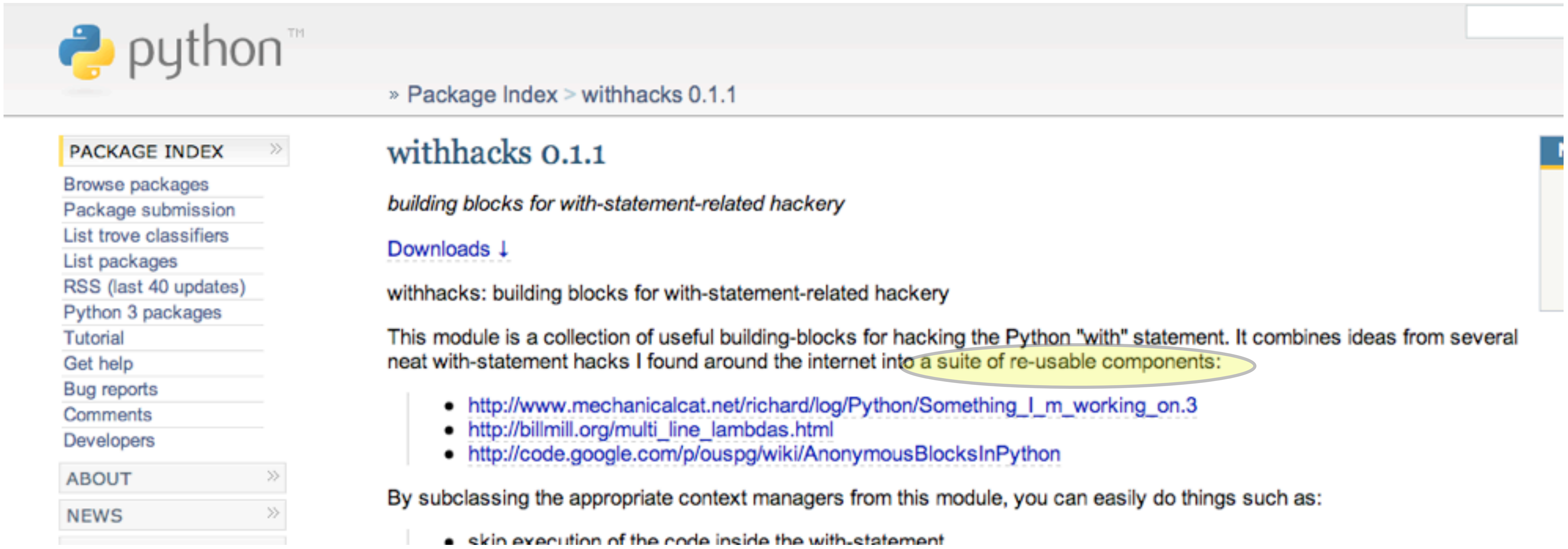
Sunday, November 21, 2010

patching the stdlib summons hideous demons that will eat your soul.

so don’t be too clever or cute

making things that “act like a list, sort of” is all well and good until you discover that it’s not quite list-like enough.

just because you **can** abuse ‘with’ doesn’t mean you should



withhacks 0.1.1

building blocks for with-statement-related hackery

Downloads ↓

withhacks: building blocks for with-statement-related hackery

This module is a collection of useful building-blocks for hacking the Python "with" statement. It combines ideas from several neat with-statement hacks I found around the internet into a suite of re-usable components:

- http://www.mechanicalcat.net/richard/log/Python/Something_I_m_working_on.3
- http://billmill.org/multi_line_lambdas.html
- http://code.google.com/p/ouspg/wiki/AnonymousBlocksInPython

By subclassing the appropriate context managers from this module, you can easily do things such as:

- skip execution of the code inside the with-statement

I question the use of the phrase "reusable components" there.

what you do in the privacy of your own codebase is your own business

don’t make it mine...

Next big mistake

Writing in the wrong language

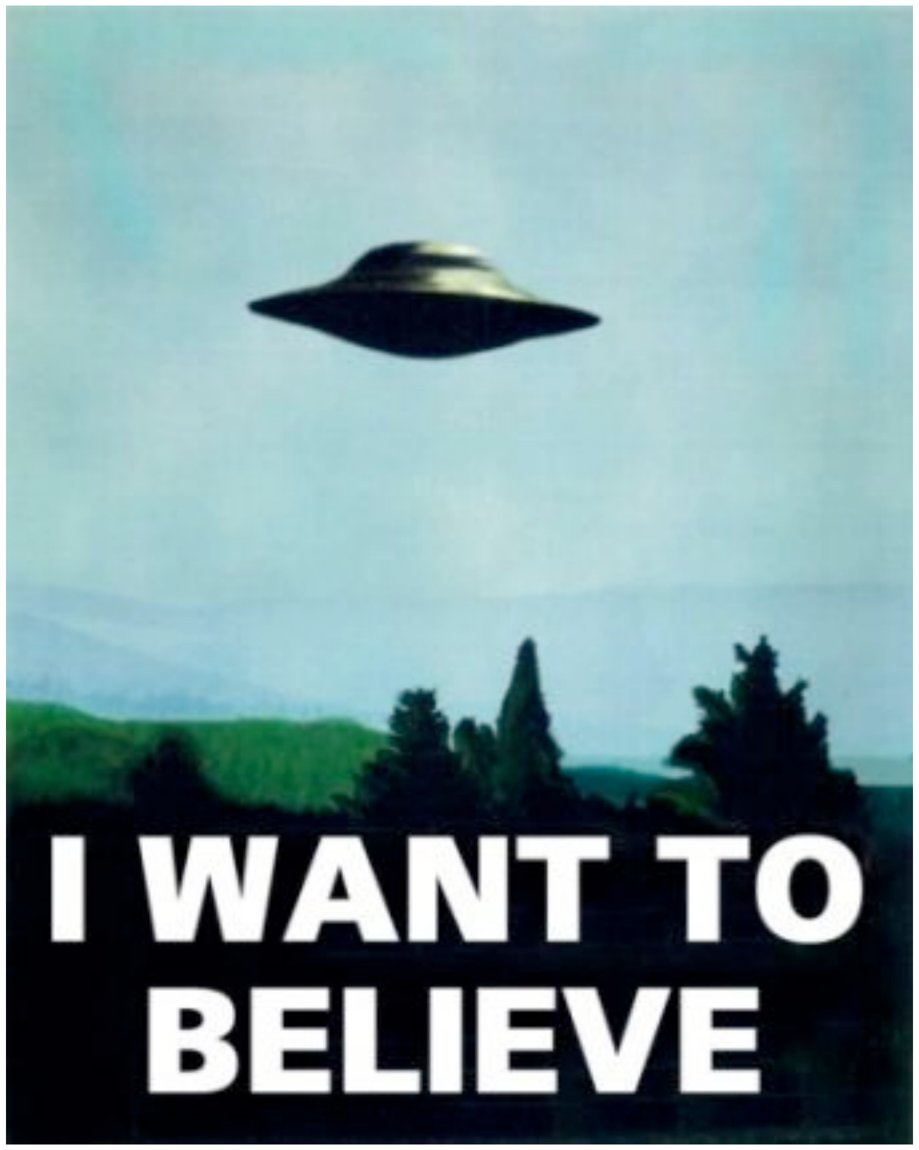
you can tell

after a while, you can almost spot someone who's not a python programmer from their code. at google, I work with a bunch of super-smart people. but often Java or C++ has polluted their brains.

don’t port APIs directly

unfortunately, the stdlib has a few great examples of how and why this is a terrible terrible idea:

don’t believe me?



import xml.dom

a direct port of the “standard” DOM APIs.

Even Javascript programmers don’t like working with the DOM – this is why jquery &c are winning out. it’s an awful awful API to work with. use elementree instead if at all possible.

import logging

Sunday, November 21, 2010

logging was a direct port of Java's log4j. I just want to log something to a file. Why should this be so hard? Fortunately, logging now has a .basicConfig() function to do this, but it really, really shouldn't be so hard. It's also huge and complicated. I'd rather use print.

import unittest

a direct port of junit, initially. it’s getting better. but pretty much every major project ends up fixing it.
consider also py.test. I plan on trying to open source google’s wrapper around unittest this year.

Python is not C/C++

...but makes a good wrapper around them

important:

provide higher level wrappers

At Google, much of the lower level magic is exposed via SWIG. This is a blessing and a curse. A blessing, because we *have* the lower level magic exposed in Python. A curse, because the SWIG interfaces are often extremely unintuitive to a Python programmer, and can be a bit of a nightmare.

recommended:

ctypes

SWIG

pyrex

maintainability
readability

cross platform

Sunday, November 21, 2010

compiling C extensions on Windows is one of the worst things you can do to a unix developer. ctypes makes this pain go away.

classic example:

pygame vs pyglet

pygame is custom wrappers around SDL.
pyglet is ctypes around opengl
I understand pygame reloaded has made more efforts here to fix this.

more learnings from pyglet

sorry. I work for a large corporation that also has marketing people. these words slip in occasionally.

wrap the lower level code

provide higher-level interfaces

Sunday, November 21, 2010

we’re not writing C code for a reason. if you have to do all the same calls, and all you’re saving yourself is the compile cycle, you’re missing out.
additionally, providing the higher level interfaces lets you maintain some level of sane compatibility when your underlying libraries change under you.

Python is *really* not Java

getters and setters?
seriously?

dozens and dozens of classes to do anything

method overloading

I’ve seen this too often – a method that can take a Foo, or a Bar, or a Baz, and does different things based on them.

@classmethod
@staticmethod

Sunday, November 21, 2010

There's only a couple of places to use a classmethod - alternate constructors. There's almost never a use for a staticmethod.

How to design an API

Sunday, November 21, 2010

this is all from personal experience, of course, and from watching other people.

Have a use-case

better:

have multiple use cases

think about how it will be used

yes, including unicode

```
_ = str.upper
```

start off:
solve one problem

start off simple

don’t overdesign

YAGNI

what worked? what didn’t?

now solve a second problem

or ask a friend to try

Sunday, November 21, 2010

your first couple of users are the equivalent of doing user testing. do they get what you are trying to achieve?

be ruthless early on with refactoring

it’s much much harder once you have users.

build on what’s there

lists, dicts, sets

go go duck typing

more on ducks.
be sensible.

ducks are widely known for their common sense.
but see earlier, be rational about duck typing. don't do it for the hell of it.
caches - dict.
result sets - iter
there is almost no case for pretending to be a string, or a tuple, or an int

getattr and setattr

Sunday, November 21, 2010

just say no.

start with a bunch of functions

don’t over-engineer

aggregate common functionality

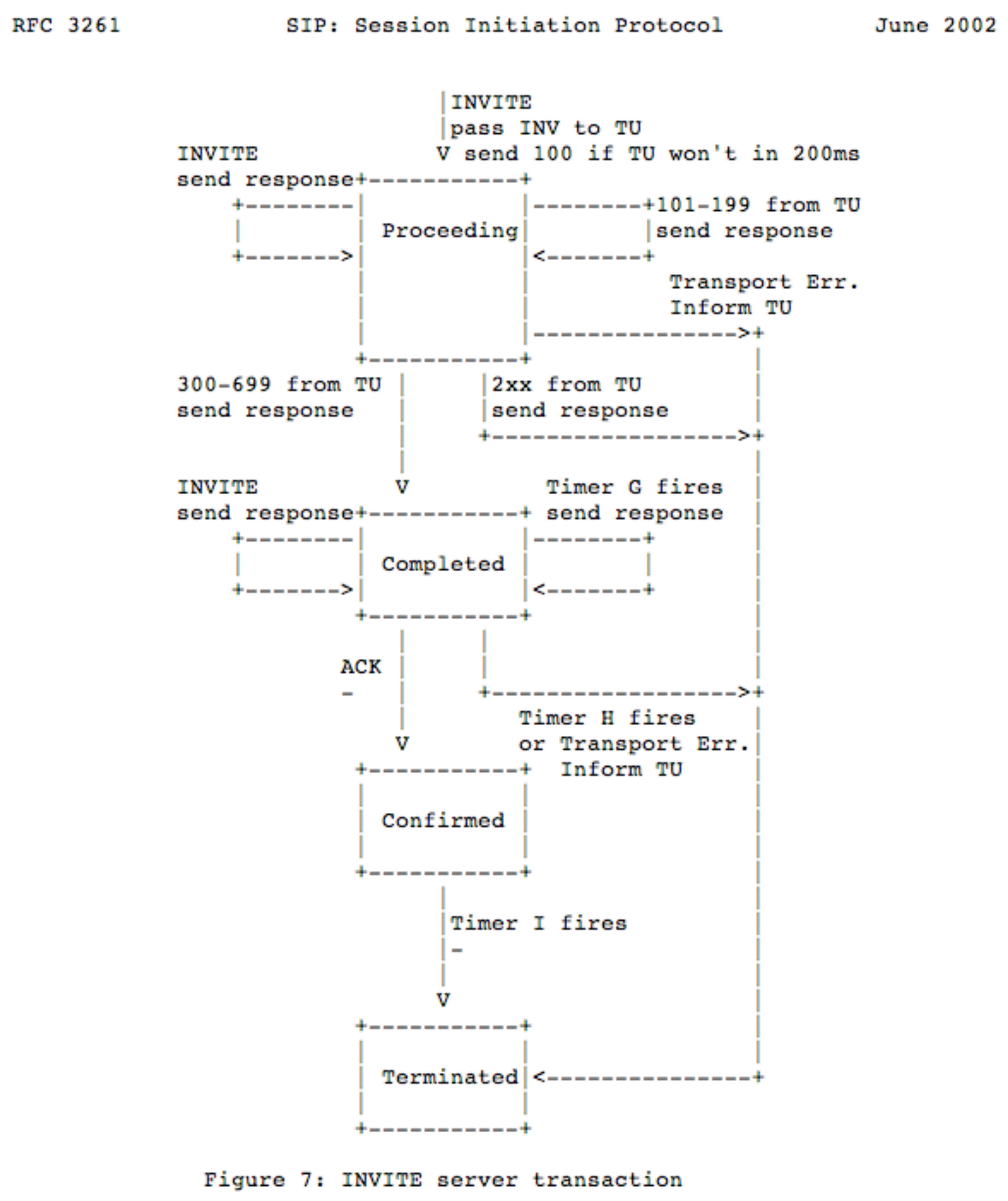
think about testing

I learned this the hard way

SIP is incredibly complex

shtoom’s SIP support: one long spike

I implemented SIP based on reading packets and making it work, rather than implementing the hundreds and hundreds of pages of RFCs. Bad mistake.



```
RFC 3261            SIP: Session Initiation Protocol           June 2002

                               |INVITE
                               |pass INV to TU
            INVITE             V send 100 if TU won't in 200ms
            send response+-----------+
                +--------|           |--------+101-199 from TU
                |        | Proceeding|        |send response
                +------->|           |<-------+
                         |           |          Transport Err.
                         |           |          Inform TU
                         |           |--------------->+
                         +-----------+                |
            300-699 from TU |     |2xx from TU        |
            send response   |     |send response      |
                            |     +------------------>+
                            |                         |
            INVITE          V          Timer G fires  |
            send response+-----------+ send response  |
                +--------|           |--------+       |
                |        | Completed |        |       |
                +------->|           |<-------+       |
                         +-----------+                |
                            |     |                   |
                        ACK |     |                   |
                        -   |     +------------------>+
                            |        Timer H fires    |
                            V        or Transport Err.|
                         +-----------+  Inform TU     |
                         |           |                |
                         | Confirmed |                |
                         |           |                |
                         +-----------+                |
                               |                      |
                               |Timer I fires         |
                               |-                     |
                               |                      |
                               V                      |
                         +-----------+                |
                         |           |                |
                         | Terminated|<---------------+
                         |           |
                         +-----------+

                Figure 7: INVITE server transaction
```

Sunday, November 21, 2010

SIP is incredibly complex. core RFC is 269 pages.
lack of testing killed me over and over again.
lack of testability killed me over and over again – it eventually killed my will to live and/or work on shtoom any more.

testable APIs

an interesting discovery

testing is good

motherhood, apple pie, american flag, all that stuff but it turns out

testing actually makes an API *better*

and not just how you might think...

testing using mocks

Sunday, November 21, 2010

there's a bunch of mock libraries - I like "mox", but pick and choose what works for you.

python uses duck-typing

duck-typing + testable APIs = :-)

Sunday, November 21, 2010

if your code is in small, testable pieces – pieces that rely on something shaped like a particular type of duck, people can find new and interesting ways to use it.

people will find new ways to use your code oh and btw,

don’t rely on _methods

people are bastards. you can’t try and mark part of your API “off limits”
python’s standard library had this problem at multiple points - to get some stuff done, you had to override an _ prefixed method.

and __ methods are *right out*

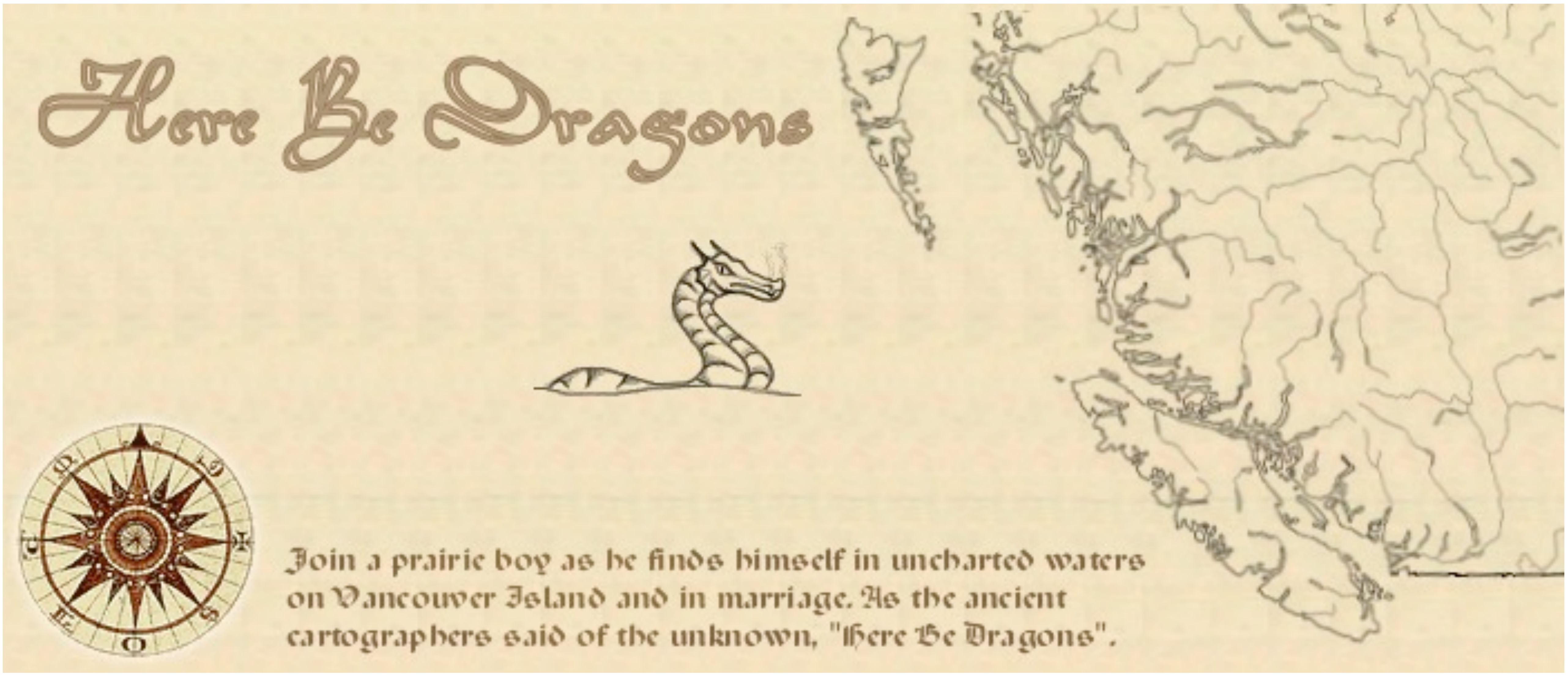
Sunday, November 21, 2010

double underscore is a namespace mangling thing. it doesn't protect you. it just means users who just want to do stuff have to mess about a little. I rate __ methods as one of Python's bigger mistakes.

all of your API is public

we got burnt on this in appengine - we had _methods for internal implementation details. People used these, overrode these and generally made it impossible for us to change them. you can say “don’t do this” but people will.

Python’s ethos is we’re all “consenting adults” sometimes it's best not to look too closely the best you can do is say



Here Be Dragons

Join a prairie boy as he finds himself in uncharted waters on Vancouver Island and in marriage. As the ancient cartographers said of the unknown, "Here Be Dragons".

How to make your API popular

Documentation

I can't emphasise enough how important this is. I will actually make choices on an lib based on the quality of the docs.

anything is better than nothing

even just generated from docstrings

Sunday, November 21, 2010

this will of course encourage you to do docstrings.
but the main thing is to make sure your docs aren't just a reference.
logging used to have that. it didn't answer the "HOW DO I ACTUALLY USE THIS THING!?1"

Examples

Sunday, November 21, 2010

You should have a few hello world type examples – over time, build these up. If you are lucky, people will contribute more.

Frequent releases

more importantly, make it clear what’s going on
this is something I’ve done badly

Frequent, or at least regular, releases.

Frequent, or at least *some*, releases.

Sunday, November 21, 2010

“just build from the revision control” isn’t a great plan.

Be open to feedback

... and patches. as soon as you release code, suddenly you have more users. They can think of use cases you've never, ever thought of. Don't be precious about your code or your library. especially in the early days...

Think about stability

once your library gets users, they want their apps to keep working. This becomes harder once you have paying users - your entire API becomes somewhat frozen.

Python releases

linux kernel

plan for extensibility

your users will find new ways to use your library than you could have ever imagined.

finally:

what happens if you don’t succeed?

nothing bad

this goes back to why you wrote an API in the first place.
assuming you haven't based a business model on a library or API you're releasing...

the good ideas will live on

Sunday, November 21, 2010

bits of shtoom turned up in unexpected places.
UPnP was based on a module called ‘soapsucks’ which turned up in a number of other projects, although they usually renamed it.

and you might be able to get a talk or two out of it finally...

the most important thing about API design

no matter what you think

no matter how much
you plan

expect the unexpected