Tales of rejected PEPs

SORT OF

Jacek Szpot

10CLOUDS

and @maligree after hours

me!

me me me me me me

let's talk about me

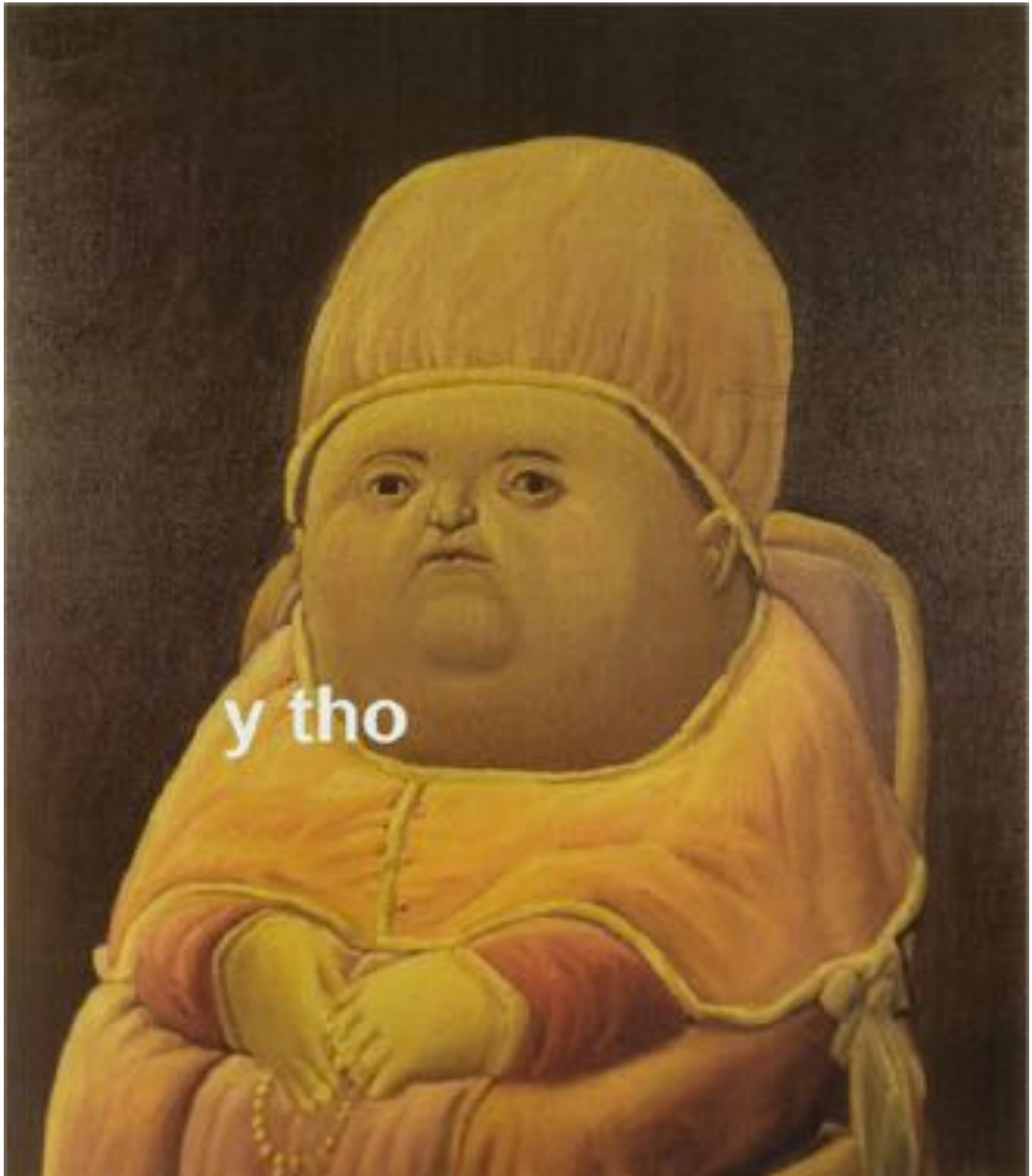
PEPs

What the hell are PEPs

python enhancement proposals

python enhancement proposals

of course you know this


y tho


metro
metroPCS


I AM
A
LITTLE
UPSET.

CELEBRITY QUIZ

Who's your daddy?

Who's your daddy?

BDFL

Benevolent Dictator for Life

idea

+

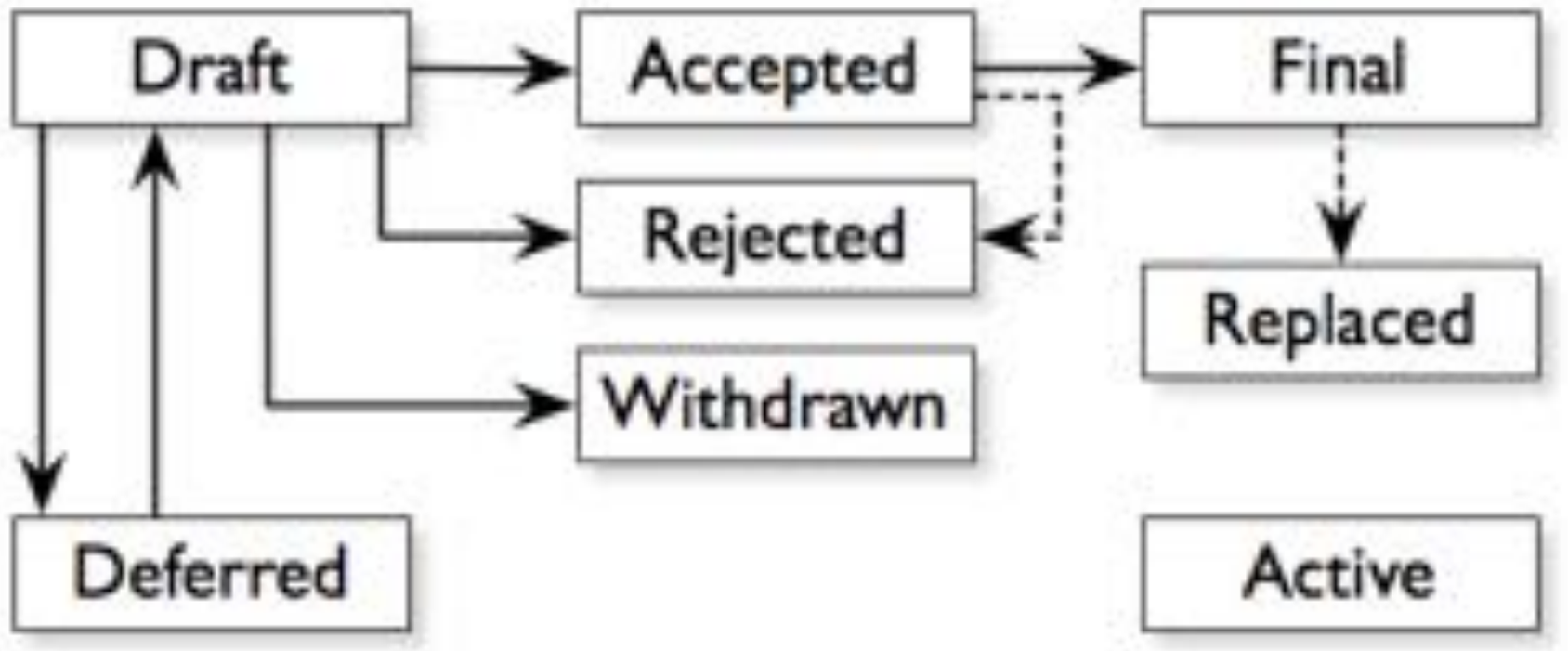
champion

+

github PR

=

new pep draft


Draft
Accepted
Final
Rejected
Replaced
Withdrawn
Deferred
Active

What's the PEP that everyone knows?

PEP554

PEP8

```
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

```
…
…
…
…
…
Sparse is better than dense.
…
…
…
…
…
In the face of ambiguity, refuse the temptation to guess.
…
…
…
…
…
…
…
```

“Don't try to stick too much code on one line.”

No one knows about the other one.

Let’s talk about ~~pepes~~ peps!

PEP 504 -- Using the System RNG by default

PEP:	504
Title:	Using the System RNG by default
Author:	Nick Coghlan «ncoghlan at gmail.com»
Status:	Withdrawn
Type:	Standards Track
Created:	15-Sep-2015
Python-Version:	3.6
Post-History:	15-Sep-2015

Abstract

Python currently defaults to using the deterministic Mersenne Twister random number generator for the module level APIs in the `random` module, requiring users to know that when they're performing "security sensitive" work, they should instead switch to using the cryptographically secure `os.urandom` or `random.SystemRandom` interfaces or a third party library like `cryptography`.

Unfortunately, this approach has resulted in a situation where developers that aren't aware that they're doing security sensitive work use the default module level APIs, and thus expose their users to unnecessary risks.

serious business?

os.urandom / random.SystemRandom

over

random.* whatever

```
import random
>>> random._inst
<random.Random object at 0x7fc536051618>

# PEP proposed: make _inst a SystemRandom instance
# by default and provide a call to
# switch to random.Random:
# random.ensure_repeatable()


# oooh and this has a performance impact
```

PEP 506 -- Adding A Secrets Module To The Standard Library

PEP:	506
Title:	Adding A Secrets Module To The Standard Library
Author:	Steven D'Aprano <steve at pearwood.info>
Status:	Accepted
Type:	Standards Track
Created:	19-Sep-2015
Python-Version:	3.6
Post-History:	

15.3. `secrets` — Generate secure random numbers for managing secrets ¶

New in version 3.6.

Source code: Lib/secrets.py

The `secrets` module is used for generating cryptographically strong random numbers suitable for managing data such as passwords, account authentication, security tokens, and related secrets.

In particularly, `secrets` should be used in preference to the default pseudo-random number generator in the `random` module, which is designed for modelling and simulation, not security or cryptography.

See also: PEP 506

depressingly civil and polite and nice.

PEP 666 -- Reject Foolish Indentation

PEP:	666
Title:	Reject Foolish Indentation
Author:	lac at strakt.com (Laura Creighton)
Status:	Rejected
Type:	Standards Track
Created:	3-Dec-2001
Python-Version:	2.2
Post-History:	5-Dec-2001

- `python -TNone` will refuse to run when there are any tabs.
- `python -Tn` will refuse to run when tabs are not exactly `n` spaces
- `python -TOnly` will refuse to run when blocks are indented by anything other than tabs

People who mix tabs and spaces, naturally, will find that their programs do not run. Alas, we haven't found a way to give them an electric shock as from a cattle prod remotely. (Though if somebody finds out a way to do this, I will be pleased to add this option to the PEP.)

This proposal, if accepted, will probably mean a heck of a lot of work for somebody. But since I don't want it accepted, I don't care.

PEP 3117 -- Postfix type declarations

PEP:	3117
Title:	Postfix type declarations
Author:	Georg Brandl <georg at python.org>
Status:	Rejected
Type:	Standards Track
Created:	01-Apr-2007
Python-Version:	3.0
Post-History:	

Therefore, this PEP combines the move to type declarations with another bold move that will once again prove that Python is not only future-proof but future-embracing: the introduction of Unicode characters as an integral constituent of source code.

Instead of the single Unicode character, they can type `name${UNICODE NAME OF THE DECLARATOR}$` . For example, these two function definitions are equivalent:

```
def fooλ(x℃):
    return None
```

and

```
def foo${LATIN SMALL LETTER LAMBDA WITH STROKE}$(x${DOUBLE-STRUCK CAPITAL C}$):
    return None${ZERO WIDTH NO-BREAK SPACE}$
```

This is still easy to read and makes the full power of type-annotated Python available to ASCII believers.

Example

This is the standard `os.path.normpath` function, converted to type declaration syntax:

```
def normpathλ(path✎)✎:
    """Normalize path, eliminating double slashes, etc."""
    if path✎ == '':
        return '.'
    initial_slashes✓ = path✎.startswithλ('/')✓
    # POSIX allows one or two initial slashes, but treats three or more
    # as single slash.
    if (initial_slashes✓ and
        path✎.startswithλ('//')✓ and not path✎.startswithλ('///')✓)✓:
        initial_slashesℕ = 2
comps♨ = path✎.splitλ('/')♨
new_comps♨ = []♨
for comp✎ in comps♨:
    if comp✎ in ('', '.')ℂ:
        continue
    if (comp✎ != '..' or (not initial_slashesℕ and not new_comps♨)✓ or
        (new_comps♨ and new_comps♨[-1]✎ == '..')✓)✓:
```

PEP 3125 -- Remove Backslash Continuation

PEP:	3125
Title:	Remove Backslash Continuation
Author:	Jim J. Jewett <JimJJewett at gmail.com>
Status:	Rejected
Type:	Standards Track
Created:	29-Apr-2007
Post-History:	29-Apr-2007, 30-Apr-2007, 04-May-2007


I'M HELPING

PEP 394 -- The "python" Command on Unix-Like Systems

PEP:	394
Title:	The "python" Command on Unix-Like Systems
Author:	Kerrick Staley <mail at kerrickstaley.com>, Nick Coghlan <ncoghlan at gmail.com>, Barry Warsaw <barry at python.org>
Status:	Active
Type:	Informational
Created:	02-Mar-2011
Post-History:	04-Mar-2011, 20-Jul-2011, 16-Feb-2012, 30-Sep-2014
Resolution:	https://mail.python.org/pipermail/python-dev/2012-February/116594.html

PEP 374 -- Choosing a distributed VCS for the Python project

PEP:	374
Title:	Choosing a distributed VCS for the Python project
Author:	Brett Cannon <brett at python.org>, Stephen J. Turnbull <stephen at xemacs.org>, Alexandre Vassalotti <alexandre at peadrop.com>, Barry Warsaw <barry at python.org>, Dirkjan Ochtman <dirkjan at ochtman.nl>
Status:	Final
Type:	Process
Created:	07-Nov-2008
Post-History:	07-Nov-2008 22-Jan-2009

Okay let’s find some treasure.

```
git clone $pep_repo
```

```
# Find me some dirt!
```

```
$ git grep -i fuck
# nothing :(
```

```
$ git grep -i shit
# nothing :(
```

```
$ git grep -i dick
# ...
```

```
$ git grep -i dick
# …
```

```
pep-3133.txt:express the works of Charles Dickens in Python: ::
pep-3133.txt:  >>> from dickens import Urchin, Gentleman
```

```
$ git grep -i dick
# …
```

```
pep-3133.txt:express the works of Charles Dickens in Python: ::
pep-3133.txt:  >>> from dickens import Urchin, Gentleman
```

C’est la vie.
I look for profanity, I find **Dickens**.

```
$ git grep -i dogs
# …
```

```
pep-3133.txt:With the invention of both dogs and trees, we were
no longer able to …
```


WHAT'S GOING ON IN THIS THR-
OH LAWD

PEP 3133 -- Introducing Roles

PEP:	3133
Title:	Introducing Roles
Author:	Collin Winter <collinwinter at google.com>
Status:	Rejected
Type:	Standards Track
Requires:	3115 3129
Created:	1-May-2007
Python-Version:	3.0
Post-History:	13-May-2007

```
pies.szczeka()
```

Okay let’s look for envy.

```
$ git grep -i $other_lang
```

```
$ git grep -i php | wc -l
74
```

```
$ git grep -i “[^.]php” | wc -l
40
```

PEP 505 -- None-aware operators

PEP:	505
Title:	None-aware operators
Author:	Mark E. Haase <mehaase at gmail.com>
Status:	Draft
Type:	Standards Track
Created:	18-Sep-2015
Python-Version:	3.6

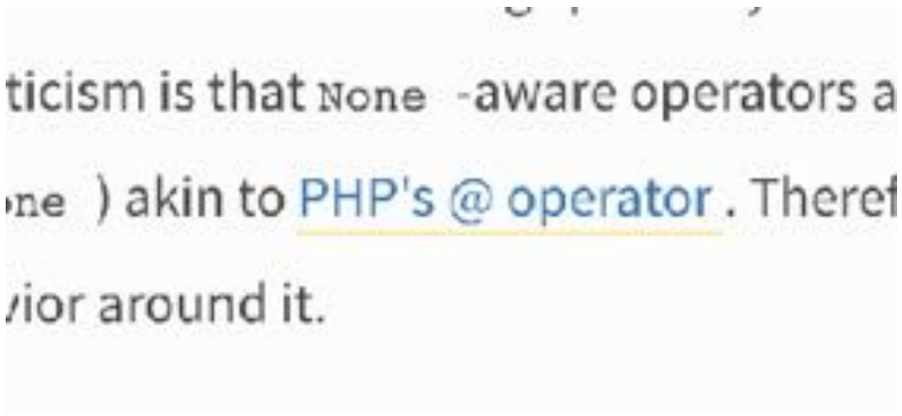
Abstract

Several modern programming languages have so-called " `null` -coalescing" or " `null` - aware" operators, including C# [1] , Dart [2] , Perl, Swift, and PHP (starting in version 7). These operators provide syntactic sugar for common patterns involving null references.

ticism is that `None` -aware operators a

`ne`) akin to PHP's @ operator . Theref

vior around it.


ticism is that None -aware operators a
one) akin to PHP's @ operator . Theref
vior around it.

ticism is that `None` -aware operators a

`ne`) akin to PHP's @ operator . Theref

vior around it.

ticism is that `None` -aware operators a

`ne`) akin to PHP's @ operator . Theref

vior around it.

ism is that `None` -aware operators a

) akin to PHP's @ operator . Theref

r around it.

ism is that `None` -aware operators a

) akin to PHP's @ operator . Theref

r around it.

that None -aware operators

to PHP's @ operator . The

nd it.

t None -aware opera

PHP's @ operator.

t.

P's @ oper

(that was 6 useless slides)

@

▲ 649 ▼ Anonymous

This operator is affectionately known by veteran phpers as the stfu operator.

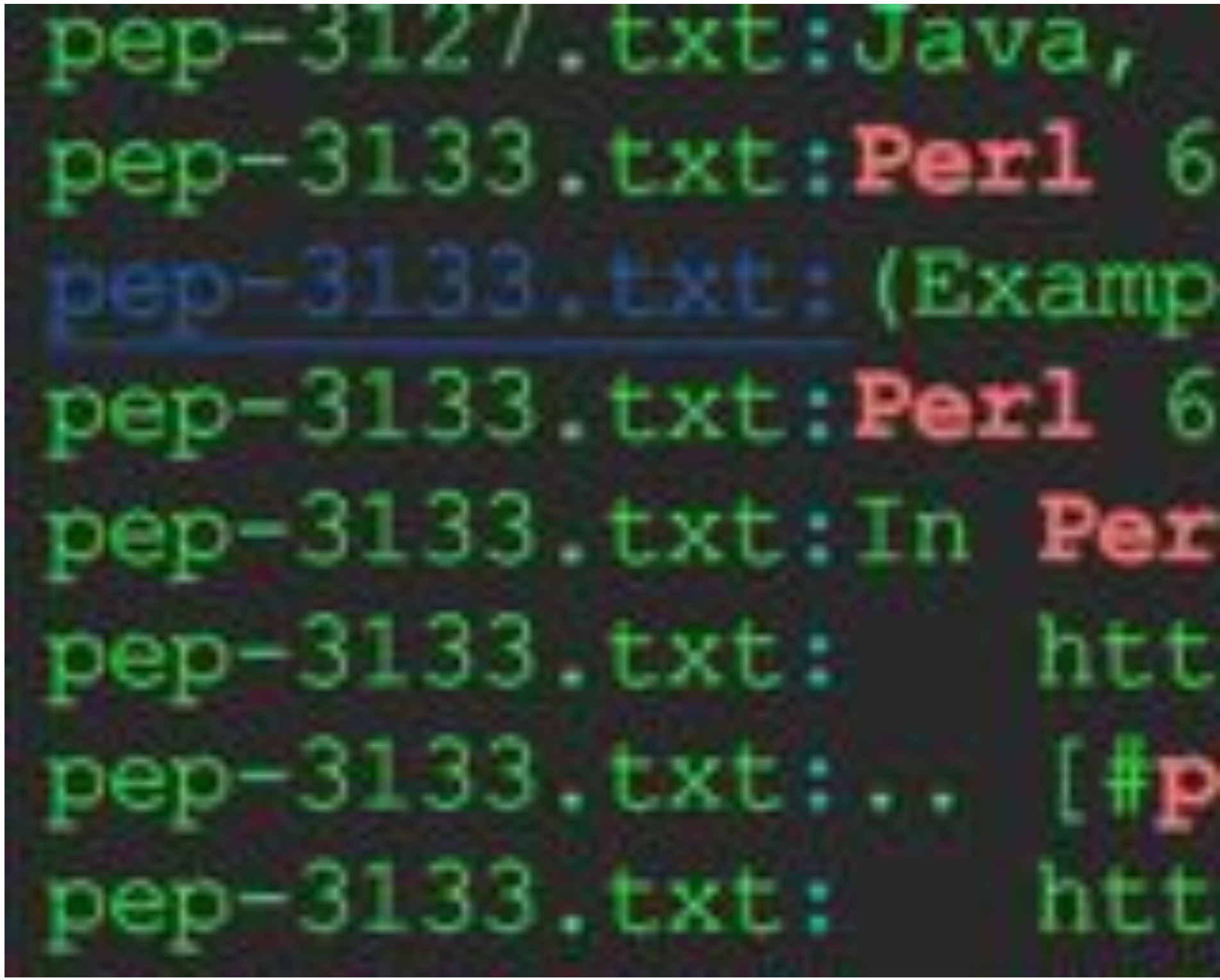
```
git grep -i "[Pp]erl[^(ly|pod|\.org)|ink]" | wc -l
```

```
83
```

They even mention Perl 6!

```
pep-3127.txt:Java, Perl, and JavaScript, treat a sequence of digits wit
pep-3133.txt:Perl 6 [#perl6-s12]_ where it is called "roles", and it is
pep-3133.txt:(Examples adapted from an article on Perl 6 roles by Curti
pep-3133.txt:Perl 6 allows instances to perform different roles than th
pep-3133.txt:In Perl 6, this is done by creating an anonymous class tha
pep-3133.txt:   http://www.perlmonks.org/?node_id=384858
pep-3133.txt:.. [#perl6-s12]
pep-3133.txt:   http://dev.perl.org/perl6/doc/design/syn/S12.html
```

```
pep-3127.txt:Java, Perl, and JavaScri
pep-3133.txt:Perl 6 [#perl6-s12]_ whe
pep-3133.txt:(Examples adapted from a
pep-3133.txt:Perl 6 allows instances
pep-3133.txt:In Perl 6, this is done
pep-3133.txt:    http://www.perlmonks.
pep-3133.txt:.. [#perl6-s12]
pep-3133.txt:    http://dev.perl.org/p
```


pep-3127.txt:Java,
pep-3133.txt:Perl 6
pep-3133.txt: (Examp
pep-3133.txt:Perl 6
pep-3133.txt:In Per
pep-3133.txt: htt
pep-3133.txt:.. [#p
pep-3133.txt: htt

```
pep-0439.txt:There is a Perl package installer also named "pip".
```

```
git grep -i "JavaScript" | wc -l
29
```

```
git grep -i "ECMAScript" | wc -l
9
```

```
pep-0536.txt:in recognizing escape sequences; ECMAScript 2016 (JavaScript) allows
pep-0536.txt:.. [1] ECMAScript ``IdentifierName`` specification
pep-0536.txt:   Yes, ``const cthulhu = { H\u0042: 42 }`` is valid ECMAScript 2016
maligree:peps/ (master) $
```

```
other languages too
```

other languages too.

let’s wrap this up

what did we learn?

Questions?

the koniec

goodbye

Jacek Szpot

and @maligree after hours