10年前から来たphp.py

"php.py" came from the 2010's

pixiv Inc.
USAMI Kenta

pixiv

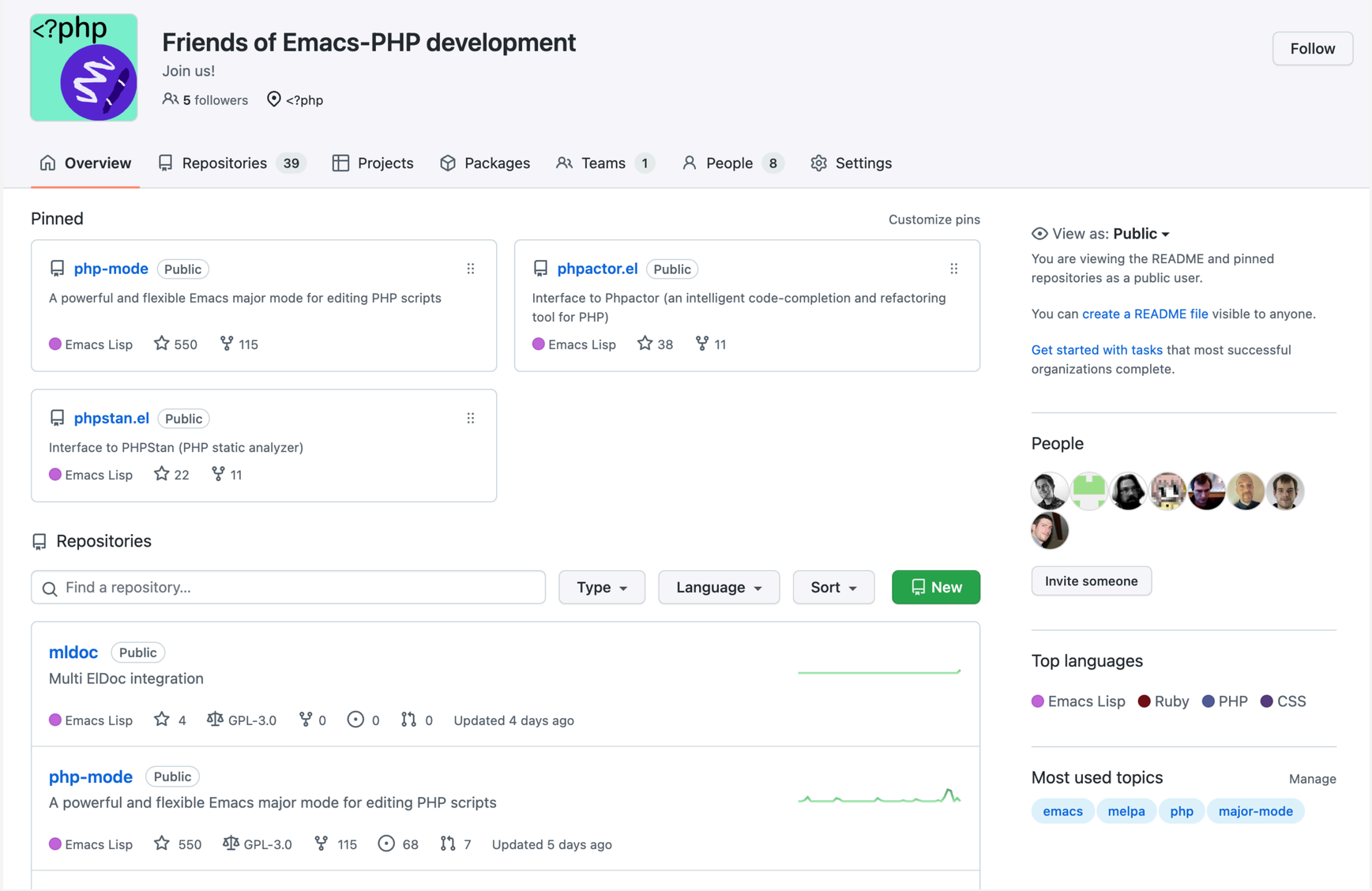
2022-10-15

お前誰よ

- うさみけんた（@tadsan）/ **Zonu.EXE** / **にゃんだーすわん**
- ピクシブ株式会社 pixiv事業本部 エンジニア
  - 最近はピクシブ百科事典(dic.pixiv.net)を開発しています
  - **会社ではPython使われてますが私は触ってません！！！！**
- Emacs Lisper, PHPer
  - Emacs PHP Modeを開発しています（2017年-）

3週間前はPHP
カンファレンスでした
(両方に携っているスタッフ
おつかれさまです)

Emacs-PHP



pixiv

PHPの静的解析とか言語仕様の紹介をして暮らしています

普段はEmacsの
php-mode開発したり

pixiv

WEB+DB PRESS総集編

pixiv

Software Design 11月号

この記事は執筆時点で10月2～3日に開催を予定しているPHPカンファレンス2021と連動した短期集中連載の最終回です。つまり、この記事が掲載されたSoftware Design 2021年11月号をみなさんが手にとられるころにはすでに開催されています。カンファレンスで発表されたセッションはYouTubeチャンネル[注1]にアーカイブが公開されているはずですので、興味のある方はぜひご覧ください。筆者はPHPカンファレンス2021で「**配列、ジェネリクス、PHPで書けない型**」と題したトークを発表します。

最終回では近年の動的言語の型検査の動向についての概観に触れたあと、PHPの動向について説明します。

静的型付き(statically typed)の言語と呼びます。事前にデータの種類を固定せずに処理することを**動的型検査**(dynamic type checking)と呼び、実行時にデータに応じた処理をすることが基本のプログラミング言語を**動的プログラミング言語**(dynamic programming language)と呼びます。

メジャーなものでは、C言語、Java、C++、C#、Go、Swift、Rust、Scala、Kotlin、Haskell、OCamlなどは静的型付きの言語です。また、Python、JavaScript、Ruby、シェルスクリプト、Clojureなど Lisp系の言語、Smalltalk、そしてPHPは動的言語と呼ばれます。俗に実行ファイルを生成するものをコンパイル言語、ソースコードを直接実行するものをインタプリタ言語として

pixiv

Emacsで動的言語をリアルタイム静的解析しながら快適に開発できるようにしています

Pythonは1行くらい書いたことあります

nishioさんに影響されて…

西尾泰和のブログ
NISHIO HIROKAZU's Blog
現在Jython本執筆中〜
西尾泰和のプロフィール 西尾泰和の活動ダイジェスト 西尾泰和のブログ@Cybozu Labs

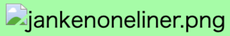
« 日記 |Main| 日記 »

« Pythonでイテレータの復習 | Python | Pythonの構文木を見る »

Pythonでワンライナーを作成する際のノウハウ集

これはLL Ringというイベントの「じゃんけん2.0」に出場する際に「多くの構文に改行が必須であるPythonで書かれたじゃんけんエージェントをワンライナーにしていたらウケるかな」と思ってワンライナー化しているときに書いたメモです。自分用のメモのつもりだったので書き殴ってありますが、意外と人気のようなので近いうちに加筆します。 実は後から書いた英語版(How to make oneliner in Python?)の方が整理されているのかも。

完成したワンライナー


jankenoneliner.png

def文を式にする

defは改行を要求するのでlambdaに置き換える必要がある。

```
def foo(x): return x + 1(ここに改行)
```

```
foo = lambda x: x + 1
```

```
globals().__setitem__("foo", lambda x: x + 1)
```

lambdaは式しか含むことが出来ないので、代入などの文は全部式に置き換える必要がある。

if文を式にする

```
if condition:
    p("True")
else:
    p("False")
```

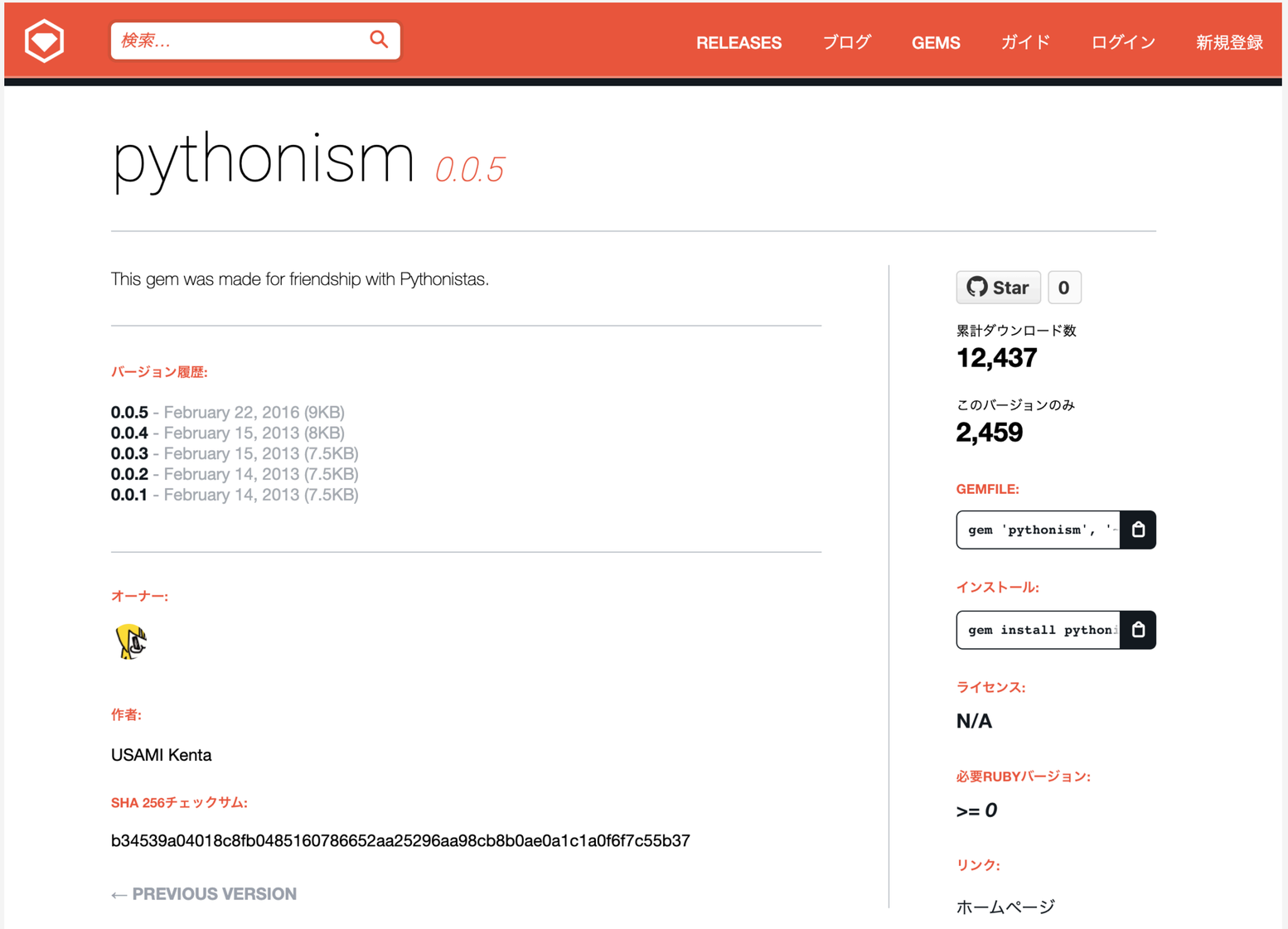
```
condition and p("True") or p("False")
```

andやorが遅延評価することを利用している。このandとorを使った評価順の制御が、式しか使えないlambdaの中ではすべての基礎になる。

p("True")が0, NoneなどのFalseと判定される値になりうる場合は例えばこうする。

PythonでRubyを書けるようにしたり

RubyをPythonっぽく置き換える

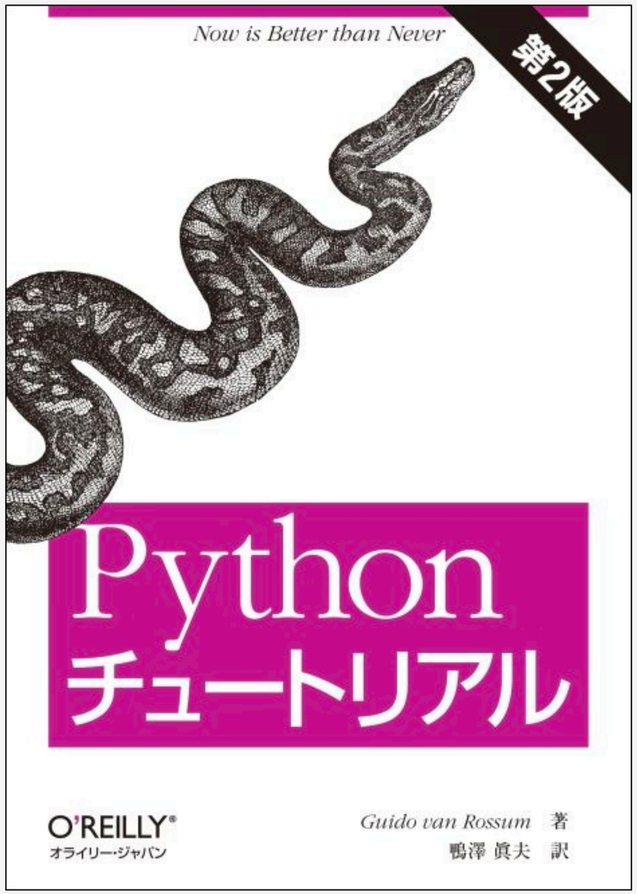


pixiv

冗談は置いておいて

2011年くらいに Pythonやってました

Pythonを学んだ本



pixiv

札幌で読書会をやっていた

pixiv

最初はRuby
並行してPython
就職してPHP

いわゆるLL言語を浅く広く触っている

RubyとPHPの処理系実装のことはちょっとだけわかる

PyConは初参加！

~~隣の芝は青い~~

専門分野じゃない
カンファレンスに
飛び込むのは良い

というわけでPython ニワカなりに10年前の ことを知ってます

~~Emacsで.py書くとき~~
~~どっちのpython-mode~~
~~使ってるんです？~~

PyCon楽しい！！

とはいえ
Pythonコミュニティに
影響を受けてきた

魂に刻まれた Pythonのヤバい機能

codec

pixiv

文字コード変換する
やつでしょ？

昔はスクリプトが UTF-8以外で 書かれていた

Pythonは3で文字列=Unicodeに全振りした

ちなみにPHP6もやろうとしたけど失敗した

Pythonは任意のエンコーディングで書かれたコードを実行できる仕組みがある

LL Tiger (2010年)

渋日記@shibu.jp

渋川よしきの日記です。ソフトウェア開発とか、ライフハックを中心に記事を書いていきます。

Top | Blog | Book List | Photos | Works | Translations | Profile

<<Sphinx 1.0リリースパーティを開催しました！ | TOP | 日本語でSphinxを使う時のストレスを減らす拡張機能 >>

2010年08月01日

LL TigerのLTでRuby on Python(仮)の発表してきました。

検索ボックス

検索

Twitter

www.flickr.com

What is this?

<< 2019年02月 >>

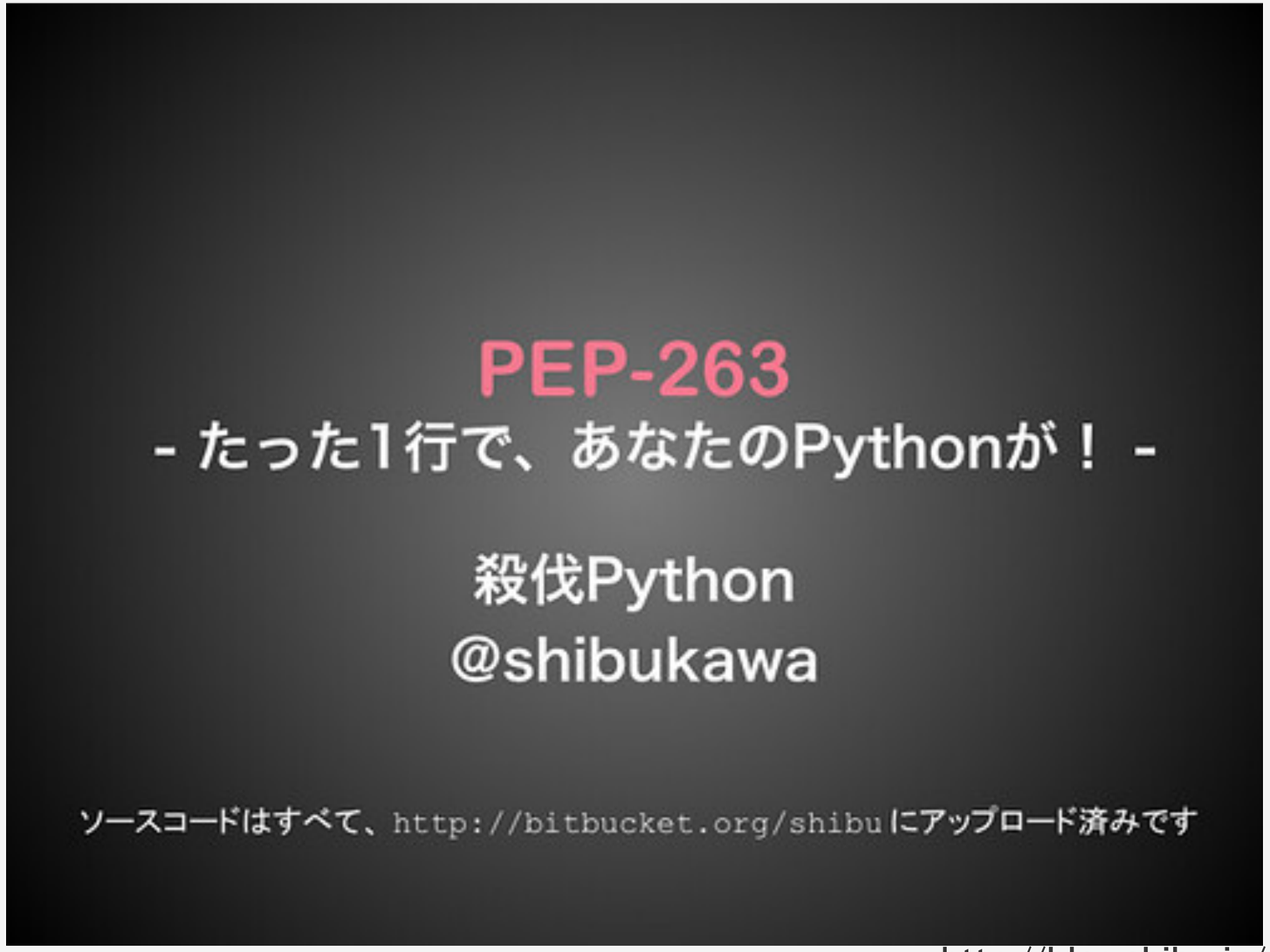
日 月 火 水 木 金 土

1 2

3 4 5 6 7 8 9

http://blog.shibu.jp/article/39920783.html より引用

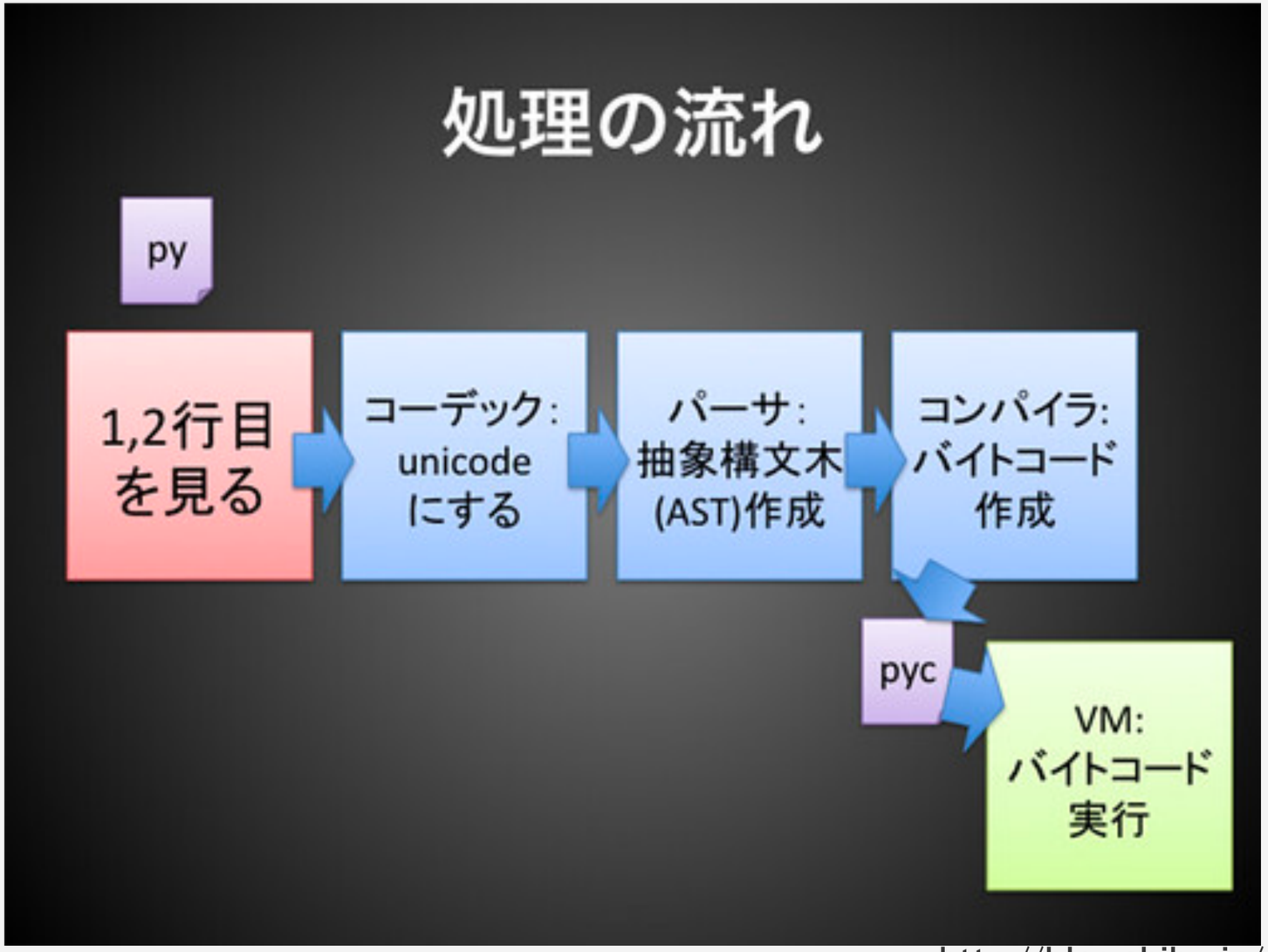
殺伐Python



pixiv

http://blog.shibu.jp/article/39920783.html より引用

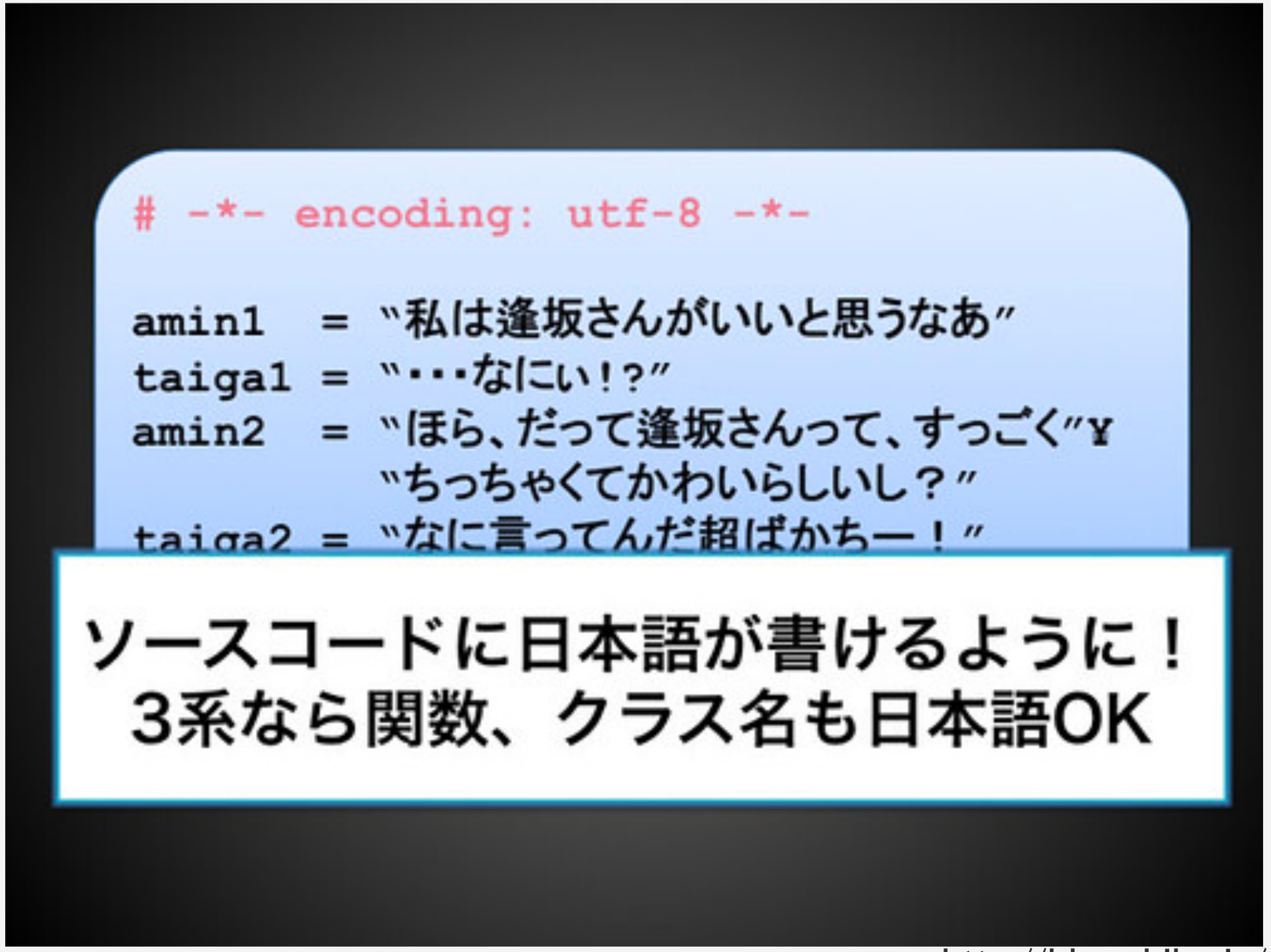
殺伐Python



pixiv

http://blog.shibu.jp/article/39920783.html より引用

殺伐Python



pixiv

http://blog.shibu.jp/article/39920783.html より引用

codecは追加できる

なんとcodecはプリプロセッサだったのだ

C言語風のマクロを実現するコーデックを実装してみました。(本番ではここでデモしました)

```
view plain  copy to clipboard  print  ?
01. # encoding: define
02.
03. #define MSG "ll tiger"
04.
05. print "hello", MSG
```

"define"というコーデックを指定すると、C言語のように、プリプロセッサとしてソースコードを加工することができます。

Shibu's Diary: LL TigerのLTでRuby on Python(仮)の発表してきました。
http://blog.shibu.jp/article/39920783.html より引用

バイトコンパイル前に介入できれば事実上何でもできる

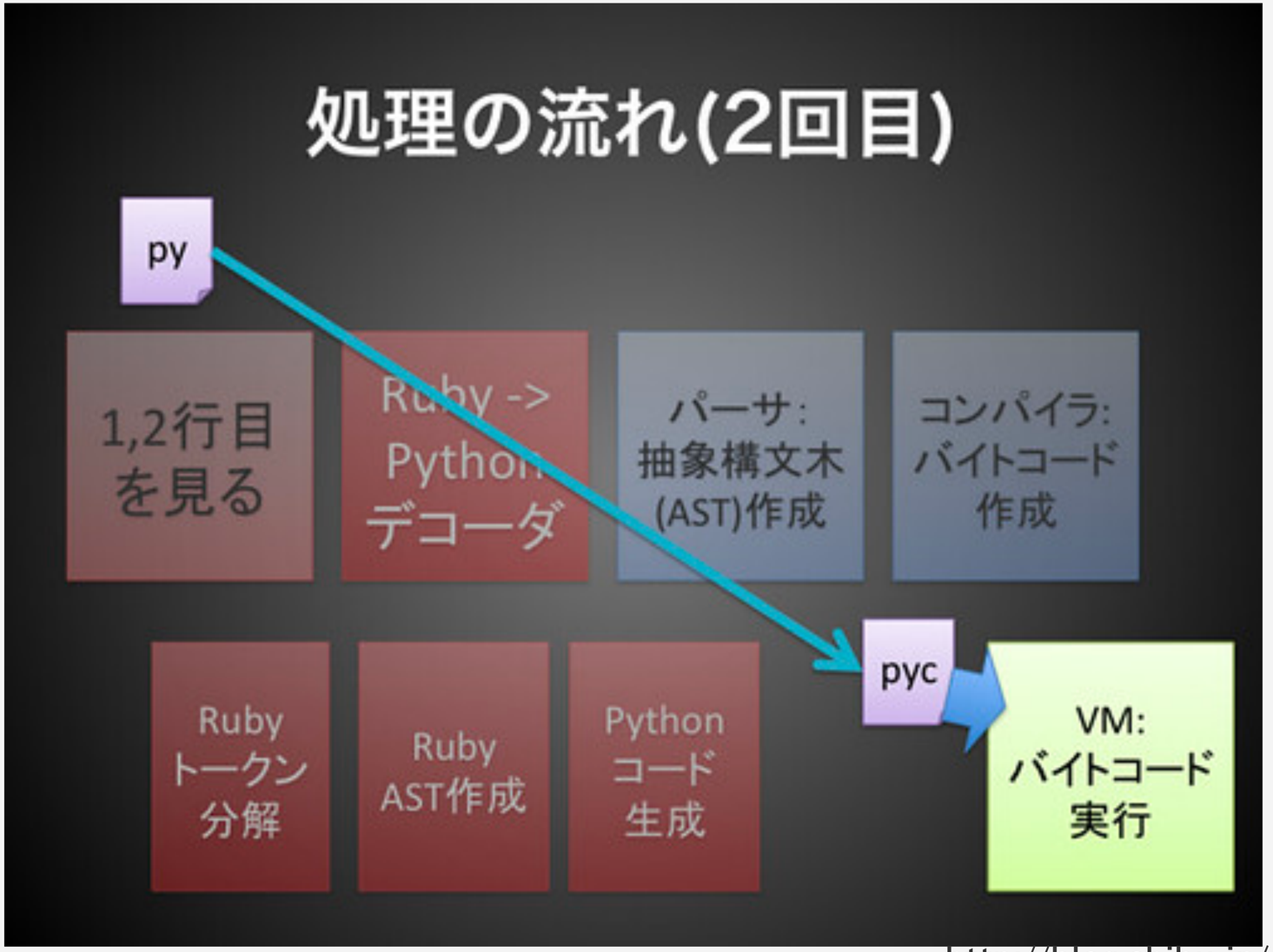
殺伐Python

pixiv

http://blog.shibu.jp/article/39920783.html より引用

渋川さんはガチなので Rubyのコードを Pythonに翻訳します

殺伐Python



pixiv

http://blog.shibu.jp/article/39920783.html より引用

殺伐Python

ちなみに、重そうな処理に見えますが、うまくバイトコードファイルができて、一度キャッシュされてしまえば時間のロスはなくなります。(本番ではここでfizzbuzzのデモをしました)

view plain copy to clipboard print ?

```ruby
# encoding: ruby

(1..100).each do |i|
  if i % 15 == 0
    puts "fizzbuzz"
  elsif i % 3 == 0
    puts "fizz"
  elsif i % 5 == 0
    puts "buzz"
  else
    puts i
  end
end
```


pixiv

http://blog.shibu.jp/article/39920783.html より引用

いま読み直しても
やばい

じゃあPHPでも動かすか(安直)

私はガチではないので
お手軽実装で済ませる

というわけで
Pythonを書きました

1行くらい…

```
import encodings,codecs,sys,re,subprocess;from importlib import ma
chinery;phpdecode=lambda input:('import helper;helper.run(b"""{}""
")\n'.format("\n".join(str(input.tobytes(),'utf-8').split("\n")[1:
]).replace('"','\\"')),len(input));search_function=lambda s:None i
f(s != "php")else codecs.CodecInfo(None,phpdecode,name='php');code
cs.register(search_function);loader=machinery.SourceFileLoader('ph
p',sys.argv[1]);loader.load_module()
```

読みにくい…

phpcodec.py

```
import encodings, codecs,sys, re, subprocess
from importlib import machinery

def phpdecode(input):
    return (
        'import helper;helper.run(b"""{}""")\n'
            .format("\n".join(str(input.tobytes(),'utf-8')
            .split("\n")[1:]).replace('"','\\"')),
        len(input)
    )

def search_function (s):
    if (s != "php"):
        return None

    return codecs.CodecInfo(None, phpdecode, name='php')

codecs.register(search_function)

if (__name__ == '__main__'):
    loader = machinery.SourceFileLoader('php',sys.argv[1])
    loader.load_module()
```

(ファイルに書かれた
コードをPHPプロセス
に流すだけの雑実装)

みなさまおなじみのPHP

```
# encoding: php
<?=implode("\n",array_map(fn($n)=>match([($n%3==0)+0,($n%5==0)+0])
{[1,1]=>'FizzBuzz',[1,0]=>'Fizz',[0,1]=>'Buzz',default=>$n},range(
1,100)))?>
```

改行したfizzbuzz.py

```
# encoding: php
<?php
foreach (range(1,100) as $n) {
    print match ($n % 3 == 0) {
        true => match ($n % 5 == 0) {
            true => 'FizzBuzz',
            false => 'Fizz',
        },
        false => match ($n % 5 == 0) {
            true => 'Buzz',
            false => (string)$n,
        },
    } . PHP_EOL;
}
```

実行

```
Welcome to the Emacs shell

~/repo/python/phppy $ python3 -m phpcodec fizzbuzz2.py
1
2
Fizz
4
Buzz
Fizz
7
8
```

codecを現在動かすにあたって

- 公開されていた渋川さんのコードが読めない（!!?）
  - BitBucketはMercurialサポート終了 https://bitbucket.org/shibu
  - 10年前はGitHub普及前でMercurialとBitBucket使われてましたね…
- 渋川さんのコードを再利用したと思しき記事を断片的に参考に
  - https://doloopwhile.hatenablog.com/entry/20100801/1280688409

codecを現在動かすにあたって

- 元コードではStreamReaderでコードを書き換えていたが動かなかった
  - Pythonの処理系実装を軽く読んでみたけど理由わからず、教えてください
- sitecustomizeとかusercustomizeで暗黙的にコーデック追加して
  #coding: php なファイルをダイレクトに起動できるようにしたい…
  - 仕方ないので現在は -m phpcodec で実行したらダイナミックインポート
  - もっといい感じにしたいね！

おまけ

エンコーディング指定

```
# encoding: utf-8
```

テキストエディタのエンコーディング指定に由来
（Rubyもだいたい同じ仕様）

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

エンコーディング指定

```
# encoding: utf-8
```

これなに

Emacs記法

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


pixiv

PEP 263

```
^[ \t\f]*#.*?coding[:=][ \t]*([-_.a-zA-Z0-9]+)
```

それっぽい指定をいい感じに読み込んでくれる

Vim記法

```
# vim: set fileencoding=utf-8 :
```


pixiv