よくある質問

■ 今から勉強するなら Python 2.x と Python 3.x のどちらをやればいいですか？

両方とも一長一短があるので場合によります。

- 全く初めて Python に触れるなら 3.x の方が混乱が少ないと考えられます
  - 2.x は Python が始まって以来、互換性を維持しているので、あまり美しくない部分が少しあります
- 外部のライブラリの中にはまだ 3.x に対応していないものがある
  - 有名どころとしては
    - wxPython(ウィジェットツールキット)
    - MySQL-python(MySQL のドライバー。対応作業中。フォーク版は対応済み)
    - mechanize(Web アクセス自動化。開発が停滞しており対応見込みは薄い。難解だが高機能な代替物に Selemium がある)
    - pygame(ゲームフレームワーク)
    - OpenCV(画像処理ツールキット)
- 3.x の方が日本語の扱いはよくなった
  - ごくたまに 3.x の方が後退しているライブラリもあるにはある（例：zipfile）
  - 2.x では str と unicode をきちんと区別して扱う必要があるが、これが初学者には少し難しい
  - 3.x では str に統一されたので単純になった
- 書籍や情報は 2.x の方がまだ多い
- すぐに 2.x が廃れることはない
- 今後数年をかけて 2.x から 3.x に移行していくと思われる
- ある程度 Python を習得すれば違いはすぐに理解できる
- 2.x と 3.x の両方をシステムにインストールすることはできる
- Python.org によってサポートされている 2.x は 2.7 だけである。2.6 以前を公開用途に用いることは薦められない

Python2orPython3(英語) も参照のこと。

■ プログラムを実行したのですが黒い画面が一瞬出るだけです。

コマンドプロンプトから起動させてください。

```
# Python 2.x
raw_input()

# Python 3.x
input()


import os; os.system('pause')
```

などのコードをソースの最後に挿入する方法でも大丈夫です。

■ Python ってなんで日本ではこんなにマイナーなの？

回答１：使っている人はそこそこいるとは思うんだけど、あまり大々的に表に出てくるようなものを作るのには使ってないのかもしれないね。国内では Perl, PHP, Ruby の日本語の情報が充実してるし、Python ででしかできないことは無いしね。結局、判子押す権能を持っている人が良く知ってる（名前を良く聞く）という言語が一見メジャーに見えているだけのような気もするよ。

回答２：あまり CGI で使うもんじゃないから厨房が使いたがらない無料レンタル鯖がどれだけ対応してるかと本屋の棚の占有率は比例する

■ お勧めのエディタはありますか？

初心者には IDLE をお勧めします。Python の配布物に同梱されています。パッケージ管理システム（deb や rpm など）を使っている場合は Python とは別に python-idle-* などの名前で配布されているかもしれません。

その他のエディタや統合開発環境は IDE&Editor で紹介しています。

■ IDLE（開発環境）のヘルプとか日本語の情報とかありますか？

調べてみた。残念ながら日本語のはないなー。IDLE 内蔵のデバッガの説明なんかは、英語のマニュアルを読むほかない。

基本的な使い方を知るだけでよいなら、「Pygame 実況中継 第一回」のムービーを見ると良いよ。(日本語)

英語でもよければ、こんなのもあるよ。

- One Day of IDLE Toying: (簡単なガイド)
- Using IDLE: (詳細なマニュアル)

■ PythonWin って人気ないの？

回答１：pythonwin は、"open(" とか打った時に、関数の引数リストとか簡単な説明が出てこないから使わなくなった。

回答２：calltip がでないのはアレだけど、pythonwin のデバッガは便利だった気がする。

■ python-mode(Emacs) 使ってる？どんな感じですか？

python-mode 入れてみた。○ speedbar でクラスの内部構造とかも表示してくれるのは便利だ。○ emacs 内で PythonShell も動かせて便利だ。△デバッグとかは pdb との連携でやるらしい (pdb は入れてない)○手がなじんでしまった emacs でコードが書けるのがイイ！×コード補完がない…。インテリセンスがほしい…。

知らないライブラリをあれこれ探りながらやるときには、補完機能があると安心感がちがうんだよなー。

■ Python なら Apache より Zope とかじゃないの？

回答１：mod_python 侮れないぞ。

回答２：Python なら WSGI だよ。

回答３：werkzeug

■ print 関数で表示したで文字化けするんだけど、チェックリストみたいなものは無い？

回答１：この辺をチェック：

- ファイルのエンコーディング指定 （ファイル先頭に "# -*- coding: euc-jp -*-" とか）
- 文字列代入は strhoge = " 日本語 ".decode("euc-jp") もしくは strhoge = u" 日本語 "
- 外部から入力した文字列 strinput のエンコーディングが shift_jis なら、 hoge = strinput.decode("shift_jis") と明示的に書く
- print する際のエンコード指定 : 例 strhoge.encode("utf-8") #utf-8 エンコードで出力

回答２：

- Python で UTF-8, shift_jis, euc_jp など日本語を使う方法 を見てはいけない。(コメント内容間違ってるので注意)。見てしまった人は内容を忘れること。
- このコメントが最大の間違いです。「日本語の入った文字は、u'....' のように、頭に "u" をつけて、 この文字列が UTF-8 で書かれている事を明言します。」間違って覚えると混乱するので注意。逆に言えば、これがなぜ間違ってるかを説明出来るひとは、Python での日本語の扱いのマスターに一歩近付いています。

詳細は japaneseCharset を参照してください。

■ len 関数がオブジェクト指向ぽくない。

len は単なるアクセス・ルーチンです。特殊メソッド __len__ を直接呼ぶこともできます。

```
>>> s = "hello"
>>> len(s)
5
>>> s.__len__()
5


>>> L = (2, 3, 5, 7, 11, 13)
>>> len(L)
6
>>> L.__len__()
6


>>> d = {"alpha" : " α ", "beta" : " β "}
>>> len(d)
2
>>> d.__len__()
2
```

他にも + は __ad__、hash は __hash__ などで置き換えることができます。ただし、こうしたメソッドは型が一致していることが前提になるので、自前で型検査を行わないとわかりにくい例外が発生します。なるべく標準の方法を使ったほうがいいでしょう。

■ str.split() の結果が変です。

```
>>> "hoge".split()
['hoge']
>>> "hoge".split('')
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: empty separator
```

長い間 Perl を使っていませんでしたか？

```
>>> list("hoge")
['h', 'o', 'g', 'e']
>>> "hoge"[2]
'g'
```

■ hello world でエラーになります。

```
>>> print "hello world"
  File "<stdin>", line 1
    print "hello world"
                  ^
SyntaxError: invalid syntax
```

Python 3.x を使っていませんか？ Python 2.x と 3.x は互換性がありません。Python 3.x から print はステートメントではなく関数になりました。よって以下のように書いてください。

```
>>> print("hello world")
hello world
```

ちなみに Python はいつも文法を変更しているわけではありません。Python が後方互換を破壊するような変更を行ったのは今回の１度だけです。変更の経緯は「Python3000」や「py3k」を参照のこと。

■ イースター・エッグのようなものはありますか？

あります。

```
# Python 哲学とは ...
import this

# プログラマにとって世界一有名な言葉とは ...
import __hello__

# エラーメッセージがいつもと違うような ...
from __future__ import braces
```

詳しくは Easter eggs in Python を参照のこと。

■ 2ch にソースコードを貼るとインデントがなくなってしまいます。

HTML の仕様で連続したスペースは１つにまとめられ、さらに行頭のスペースは削除されます。事前に &#32； に変換してから投稿してください。

既に投稿されたソースコードも、dat ファイルや HTML ソースを直接見れば整形前のものを確認することができます。
