

引き継ぎ資料 Vol.2

Python に関する諸々

2016/06/16

目次

1. Python とは?
2. Python の環境構築
3. Python で便利な構文
4. Python プログラムを良くするちょっとしたこと
5. データのロード

注意事項

基本的文法については省略します
適宜質問してください

コードについて

- 今日のコードとプレゼンは
`https://github.com/xi-xi/takeover_document`
- 何かあれば Issues に書いてください
- 公開レポジトリなので実名公開しないように!

Python とは?



*Python is a programming language that lets you
work quickly and integrate systems more effectively*

(<https://www.python.org>)

Python とは?

近年人気なスクリプト言語

Python とは?

近年人気なスクリプト言語

人気なジャンル

- 機械学習
- データ解析...

Python とは?

近年人気なスクリプト言語

人気なジャンル

- 機械学習
- データ解析...

採用実績

- Google
- Dropbox...

Python のいいところ・よくないところ

Pros

- 標準ライブラリが強力
- 標準以外のライブラリも充実
- 書くのがとっても楽
- 同じ意味を書くために1つのやり方

Cons

- 遅い
- (個人的には) 型に厳密であってほしい

Python の環境構築

Python 環境を整えよう

Mac には Python がデフォルト搭載
デフォルトの Python を使えばいい?

Python 環境を整えよう

Mac には Python がデフォルト搭載
デフォルトの Python を使えばいい?
よくない!

デフォルトの Python を使うと...

- Python のバージョンが OS 依存
 - Mac のデフォルトは 2.7
 - イマドキのいけてる人は 3.5
- OS のアップデートで (もしかしたら) 環境が破壊
- 自分で入れたライブラリが OS と競合する可能性
- 気軽に環境のリセットとかできない

じゃあどうするの？

pyenv + Anaconda

pyenv とは

- 複数の Python バージョンを取り替えられる
- Homebrew を使ってインストール可

超簡単な使い方

python 3.5.1 をインストール

```
$ pyenv install 3.5.1
```

カレント以下では 3.5.1 を使用

```
$ pyenv local 3.5.1
```




ANACONDA[®]

*Leading Open Data Science Platform Powered by
Python*

`(https://www.continuum.io/why-anaconda)`

Anaconda とは

- 様々なライブラリをデフォルトで搭載
 - numpy, matplotlib, scipy...
- ライブラリ管理コマンドの conda が付属
 - pip とは別物
 - pip はソースを DL してコンパイル
 - conda はバイナリをインストール

Anaconda とは

- 様々なライブラリをデフォルトで搭載
 - numpy, matplotlib, scipy...
- ライブラリ管理コマンドの conda が付属
 - pip とは別物
 - pip はソースを DL してコンパイル
 - conda はバイナリをインストール
- とっても楽に環境構築が可能

pyenv + Anaconda による環境構築

Mac

1. Anaconda のインストール (時間大)

```
$ pyenv install anaconda3-4.0.0
```

2. カレント以下で使用を宣言

```
$ pyenv local anaconda3-4.0.0
```

(参考)Windows

Anaconda インストーラを実行

Python で便利な構文

- ワンライナー
- 特殊メソッド

ワンライナー

なんでも一行でかける素敵な構文

```
1 normal = []  
2 for i in range(10):  
3     normal.append(i)  
4 one = [i for i in range(10)]
```

ここで normal と one は同じ

使い過ぎると読みにくくなるので注意

やっちゃったワンライナー

```
1 def load_csv(filename):
2     with open(filename) as f:
3         return [
4             [cell for cell in row]
5             for row in csv.reader(f)
6         ]
```


特殊メソッド

特定の構文を使う際に Python によって呼び出される関数
`__init__`, `__call__`, `__str__`など様々

例

```
1 class QuadraticFunction:
2     def __init__(self, a, b, c):
3         self.a = a
4         self.b = b
5         self.c = c
6
7     def __call__(self, x):
8         return self.a * x ** 2 + self.b * x + self.c
9
10    def __str__(self):
11        return "{}x^2+{}x+{}".format(self.a, self.b, self.c)
12 def main():
13     fx = QuadraticFunction(1, 2, 1) #call fx.__init__(1,2,1)
14     print(str(fx)) #call fx.__str__()
15     for x in range(10):
16         print(fx(x)) #call fx.__call__(x)
```

特殊メソッド

特定の構文を使う際に Python によって呼び出される関数

`__init__`, `__call__`, `__str__` など様々

うまく使うと関数のようなクラスインスタンスも作成可能

Python プログラムを良くする ちょっとしたこと

- エディタには PEP チェックを導入
- Google Python Style Guide を参考

<https://google.github.io/styleguide/pyguide.html>

- 使うべき文法
- 使うべきでない文法
- 命名規則
- コメントの書き方

覚えておくと便利な関数

- zip
- enumerate

2つのリストを同時にループ

```
1 def main():  
2     a_list = [1, 2, 3]  
3     b_list = [4, 5, 6]  
4     for a, b in zip(a_list, b_list):  
5         print(a, b)
```

enumerate

index と要素を同時に取得

```
1 def main():
2     a_list = ["a", "b", "c"]
3     for i, a in enumerate(a_list):
4         print(i, a)
```


データのロード

データのロード

Python は一般的なファイル形式をサポート

- csv
- json
- xml
- html
- gzip...

データのロード

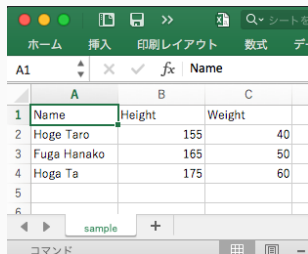
Python は一般的なファイル形式をサポート

- csv
- json
- xml
- html
- gzip...

csv ファイルとは?

- カンマで句切られた値が並ぶテキスト
- 要は表
- Excel などでも読み書き可能

```
Name,Height,Weight  
Hoge Taro,155,40  
Fuga Hanako,165,50  
Hoga Ta,175,60
```



The screenshot shows a spreadsheet application window with a green title bar and menu bar. The menu bar includes options like 'ホーム' (Home), '挿入' (Insert), '印刷レイアウト' (Print Layout), '数式' (Formulas), and 'データ' (Data). The active sheet is named 'sample'. The data is organized into columns A, B, and C, and rows 1 through 6. Row 1 contains headers: 'Name' in column A, 'Height' in column B, and 'Weight' in column C. Rows 2 through 4 contain data: 'Hoge Taro' (155, 40), 'Fuga Hanako' (165, 50), and 'Hoga Ta' (175, 60). Row 5 is empty, and row 6 is partially visible.

	A	B	C
1	Name	Height	Weight
2	Hoge Taro	155	40
3	Fuga Hanako	165	50
4	Hoga Ta	175	60
5			
6			

Python による csv の読み込み

コード

```
1 import sys
2 import csv
3
4
5 def main(filename):
6     with open(filename) as f:
7         for row in csv.reader(f):
8             for cell in row:
9                 print(cell)
10
11 if __name__ == '__main__':
12     main(sys.argv[1])
```

結果

```
16-06-15.13:01:09->python loadcsv.py sample.csv
Name
Height
Weight
Hoge Taro
155
40
Fuga Hanako
165
50
Hoga Ta
175
60
16-06-15.13:03:00-> [...document]
```

標準以外のライブラリの場合

pandas を使えば一行

調べればすぐわかるので省略

余談: C++による csv の読み込み

```
1  #include <iostream>
2  #include <fstream>
3  #include <string>
4  #include <sstream>
5
6  int main(int argc, char** argv){
7      std::string filename = argv[1];
8      std::ifstream f(filename);
9      std::string row;
10     while(std::getline(f, row)){
11         std::istringstream ss(row);
12         std::string cell;
13         while(std::getline(ss, cell, ',')){
14             std::cout << cell << std::endl;
15         }
16     }
17 }
```

余談: C++による csv の読み込み

```
1  #include <iostream>
2  #include <fstream>
3  #include <string>
4  #include <sstream>
5
6  int main(int argc, char** argv){
7      std::string filename = argv[1];
8      std::ifstream f(filename);
9      std::string row;
10     while(std::getline(f, row)){
11         std::istringstream ss(row);
12         std::string cell;
13         while(std::getline(ss, cell, ',')){
14             std::cout << cell << std::endl;
15         }
16     }
17 }
```

Python の方がはるかに楽

課題: csv ファイルの読み込み&プロット

- dummy_data.csv の中身をプロットしよう
- 1 行目が x, 2 行目が y

ヒント

- プロットは matplotlib の scatter 関数
- scatter 関数: scatter(x,y)
- scatter 関数には numpy.array 型を渡すとプロット

json ファイルとは?

- JavaScript Object Notation
- Python のようなデータ構造
 - List
 - Dictionary
- Web 界隈で広く使われる形式

sample.json

```
{  
  "Hoge Taro": {  
    "Height": 155,  
    "Weigth": 40  
  },  
  "Fuga Hanako": {  
    "Height": 165,  
    "Weigth": 50  
  },  
  "Hoga Ta": {  
    "Height": 175,  
    "Weigth": 50  
  }  
}
```

Python による json の読み込み

コード

```
1 import sys
2 import json
3
4 def main(filename):
5     with open(filename) as f:
6         data = json.load(f)
7         print(data)
8
9 if __name__ == '__main__':
10     main(sys.argv[1])
```

結果

```
16-06-15.13:03:00->python loadjson.py sample.json
{'u'Hoge Taro': {'u'Weight': 40, 'u'Height': 155}, 'u'
Fuga Hanako': {'u'Weight': 50, 'u'Height': 165}, 'u'H
oga Ta': {'u'Weight': 50, 'u'Height': 175}}
16-06-15.13:52:11->
```

Questions?