

みんなのPython勉強会 #23

2017.4.12

春なのでPythonなど いかがでしょうか？

辻 真吾 (@tsjshg)

shingo.tsuji@gmail.com

自己紹介

- ❖ 1975年 東京都足立区出身
- ❖ 大学時代のC++やITベンチャー時代のJavaを使ったWebアプリ開発を経て、10年ほど前からほとんどPython
- ❖ 都内の某大学でライフサイエンス分野の研究員
 - ❖ 実際の作業は、Pythonでデータ解析
- ❖ 3月で大学辞めて、会社やろうと思ってましたが・・・

今日の構成

- ❖ Pythonの現状と魅力
- ❖ Pythonの環境構築方法
- ❖ きっとPythonをはじめたくなる
- ❖ Python、SQL、Excel、PowerPoint、PDFを結ぶ線

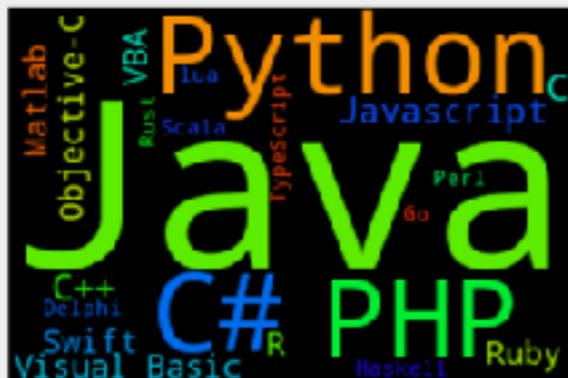
遂に2位！

PYPL Popularity of Programming Language

The PYPL Popularity of Programming Language Index is created by analyzing how often language tutorials are searched on Google.

The more a language tutorial is searched, the more popular the language is assumed to be. It is a leading indicator. The raw data comes from Google Trends.

If you believe in collective wisdom, the PYPL Popularity of Programming Language index can help you decide which language to study, or which one to use in a new software project.



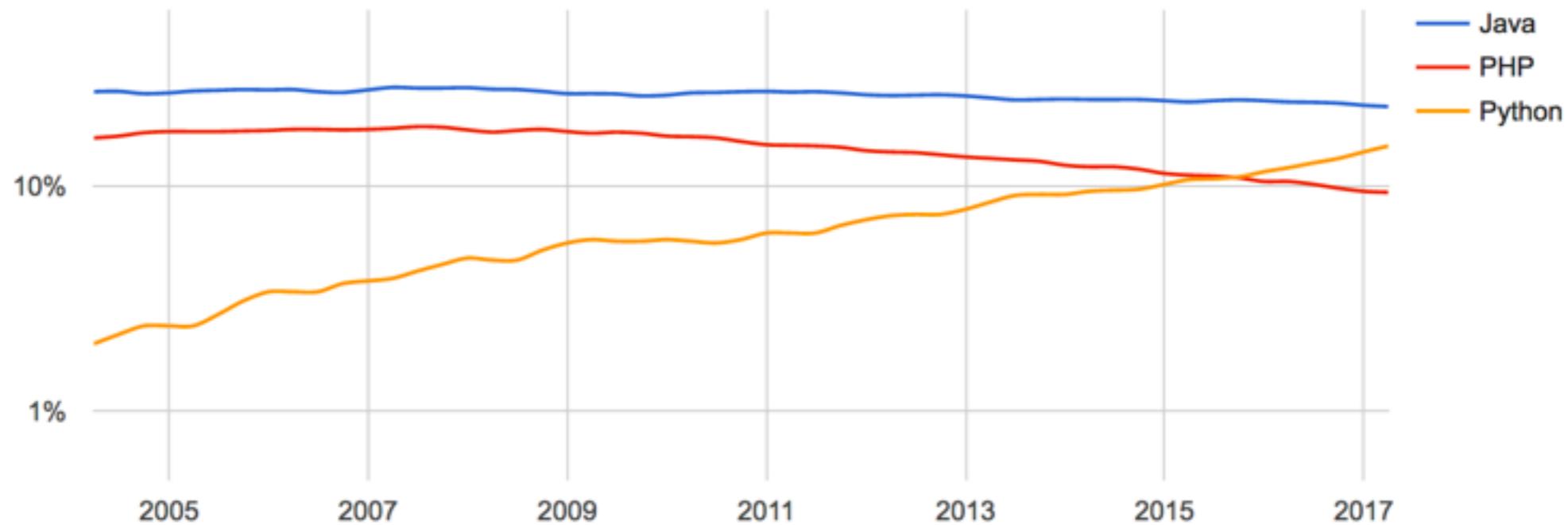
Worldwide, Apr 2017 compared to a year ago:

| Rank | Change | Language | Share | Trend |
|------|--------|--------------|--------|--------|
| 1 | | Java | 22.6 % | -1.4 % |
| 2 | | Python | 15.1 % | +3.1 % |
| 3 | | PHP | 9.4 % | -1.0 % |
| 4 | | C# | 8.3 % | -0.5 % |
| 5 | ↑ | Javascript | 7.8 % | +0.4 % |
| 6 | ↓ | C++ | 7.0 % | -0.3 % |
| 7 | | C | 6.9 % | -0.0 % |
| 8 | | Objective-C | 3.9 % | -0.7 % |
| 9 | | R | 3.5 % | +0.3 % |
| 10 | | Swift | 2.8 % | -0.1 % |
| 11 | | Matlab | 2.7 % | -0.1 % |
| 12 | | Ruby | 1.9 % | -0.3 % |
| 13 | ↑ | VBA | 1.4 % | -0.0 % |
| 14 | ↓ | Visual Basic | 1.4 % | -0.2 % |
| 15 | ↑↑↑ | TypeScript | 1.2 % | +0.8 % |
| 16 | | Scala | 1.2 % | +0.2 % |
| 17 | ↓↓ | Perl | 0.9 % | -0.2 % |

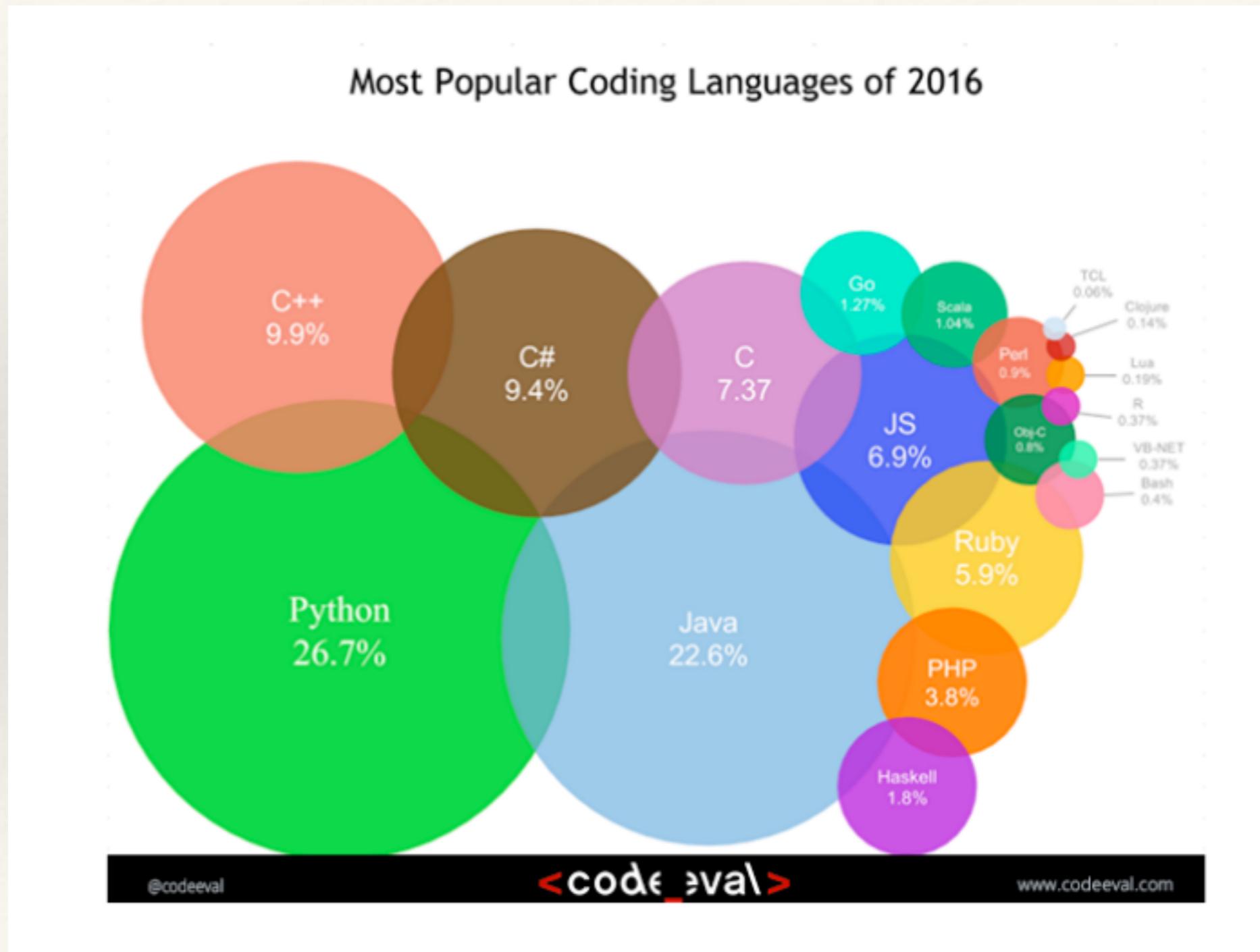
王者の背中も見えて来た！

Worldwide, Java is the most popular language, Python grew the most in the last 5 years (7.7%) and PHP lost the most (-4.8%)

PYPL PopularitY of Programming Language



実は1番人気



Pythonの魅力

- ❖ 流行りのAI（機械学習）やデータ解析の分野で中心的な役割
- ❖ 特にDeep Learningのライブラリは、現状では独壇場
- ❖ もちろん、言語そのものの魅力も
 - ❖ 充実した標準ライブラリ（Battery Included）
 - ❖ 豊富な外部ライブラリ

Pythonのインストール

- ❖ 2系と3系がある
 - ❖ 後方互換性がありません
- ❖ 是非3を！
 - ❖ 2系は2020年にメンテナンス終了の予定
- ❖ MacOSXやLinuxのOSにはじめてからインストールされている
Pythonは2系がメイン



www.python.org

豊富な外部ライブラリ

The screenshot shows the PyPI website interface. At the top left is the Python logo and the text "python™". To the right is a search bar with the word "search" in a button. Below the logo is the text "» Package Index".

On the left side, there is a navigation menu with the following items: PACKAGE INDEX (selected), Browse packages, Package submission, List trove classifiers, List packages, RSS (latest 40 updates), RSS (newest 40 packages), Python 3 Packages, PyPI Tutorial, PyPI Security, PyPI Support, PyPI Bug Reports, PyPI Discussion, and PyPI Developer info. Below this are sections for ABOUT, NEWS, DOCUMENTATION, DOWNLOAD, COMMUNITY, FOUNDATION, and CORE DEVELOPMENT, each with a right-pointing arrow.

The main content area is titled "PyPI - the Python Package Index". Below the title is a paragraph: "The Python Package Index is a repository of software for the Python programming language. There are currently 78280 packages here. To contact the PyPI admins, please use the [Support](#) or [Bug reports](#) links."

On the right side, there is a "Not Logged In" box with links for Login, Register, Lost Login?, Use OpenID, and Login with Google. Below this is a "Status" box with the text "Nothing to report".

Below the main text are three boxes: "Get Packages" (describing how to use packages), "Package Authors" (describing how to submit packages), and "Infrastructure" (describing interoperability options).

At the bottom is a table of updated packages:

| Updated | Package | Description |
|------------|-------------------------|---|
| 2016-04-10 | s2scm 0.2.7 | Compile Scheme to MIT Scratch |
| 2016-04-10 | django-svgselect 0.9.1 | A Django form widget that uses SVG files in place of Select widgets. |
| 2016-04-10 | ongair-yowsup2 2.4.48.9 | A WhatsApp python library |
| 2016-04-10 | psyplo_t_gul 0.0.1.dev1 | Graphical user interface for the psyplot package |
| 2016-04-10 | zChainer 0.3.0 | scikit-learn like interface and stacked autoencoder for chainer |
| 2016-04-10 | rowingdata 0.76.4 | The rowingdata library to create colorful plots from CrewNerd, Painsled and other rowing data tools |
| 2016-04-10 | easy_tmpl 0.1.5 | A CLI program for operation with text templates |
| 2016-04-10 | media-manager 0.9.2 | A personal media manager program |
| 2016-04-10 | gherkin-official 4.0.0 | Gherkin parser (official, by Cucumber team) |
| 2016-04-10 | ravello-sdk 1.26 | Python SDK for the Ravello API |
| 2016-04-10 | uncertainty_wrapper 0.2 | Uncertainty wrapper using estimate Jacobian |
| 2016-04-10 | kpm 0.12.1 | KPM cli |

外部モジュールが豊富 102,806個 (2017年4月10日)

ライブラリの追加方法

- ❖ PyPIからアーカイブをダウンロード
 - ❖ シェルから、`python setup.py install`
- ❖ 実際は、`pip`コマンドが便利
 - ❖ Python3.4から標準装備
 - ❖ シェルから、`pip install django`

Anacondaがおすすめ！

- ❖ Continuum Analytics社が配布するPython（無料）
- ❖ 普通のアプリケーションのようにインストール可能
- ❖ 標準のPythonに多くの外部ライブラリ（データ解析用が中心）を同梱
- ❖ condaが便利
 - ❖ `conda install django`



<https://www.continuum.io>

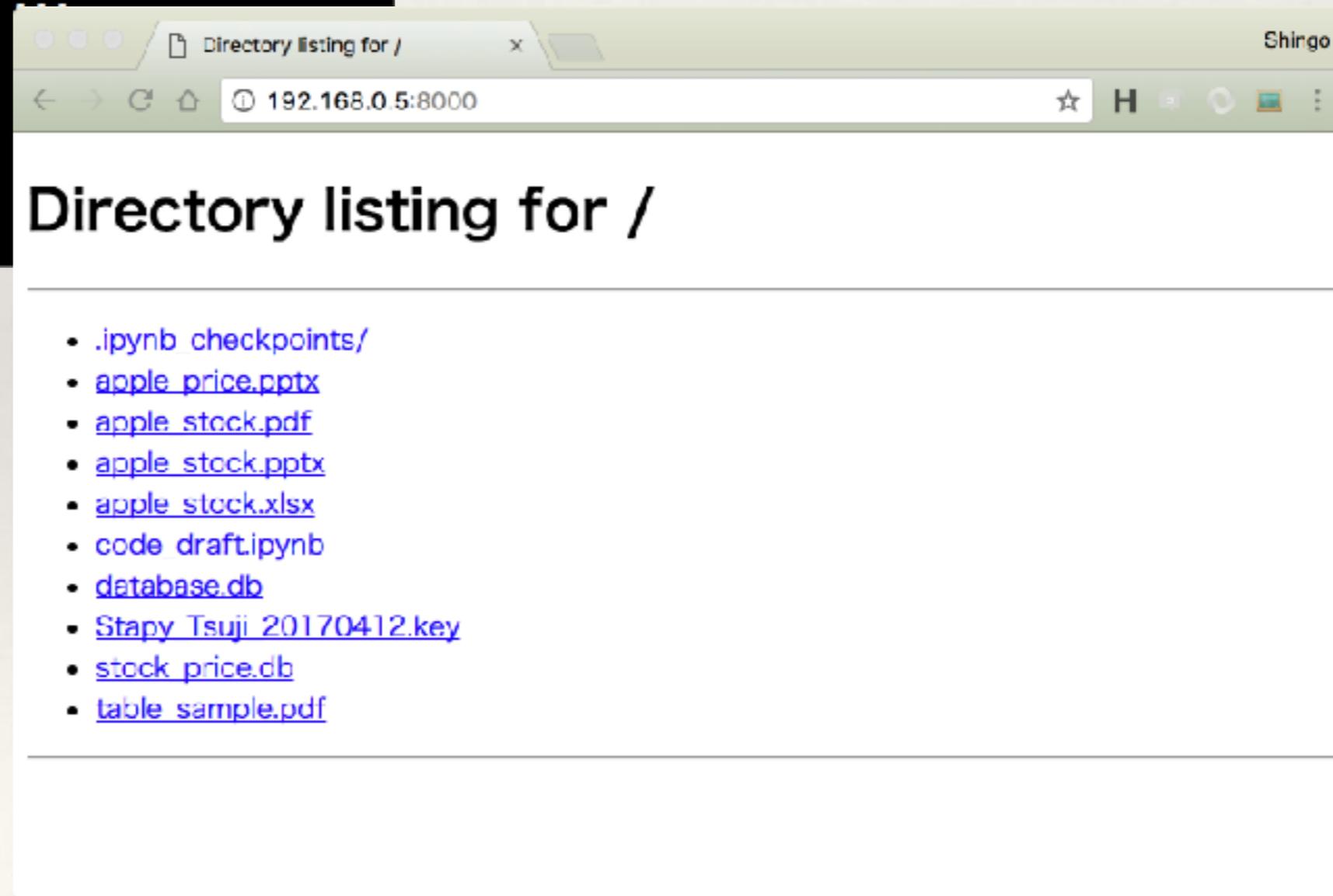
Pythonで何ができるか？

Webサーバが起動できる

```
tsujishingo — python3.5 — 52x11
~$
~$
~$ python -m http.server
Serving HTTP on 0.0.0.0 port 8000
```

しかも、標準ライブラリ

LAN内お手軽
ファイル共有

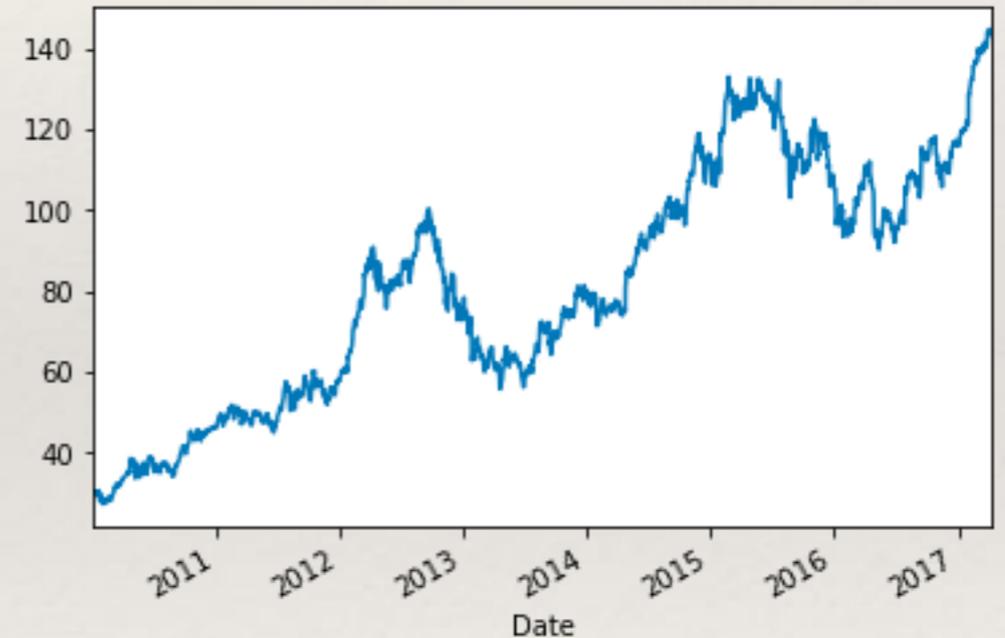


株価の情報を取得

```
import pandas_datareader as pdr  
data = pdr.get_data_google('AAPL')
```

```
%matplotlib inline  
data['Close'].plot()
```

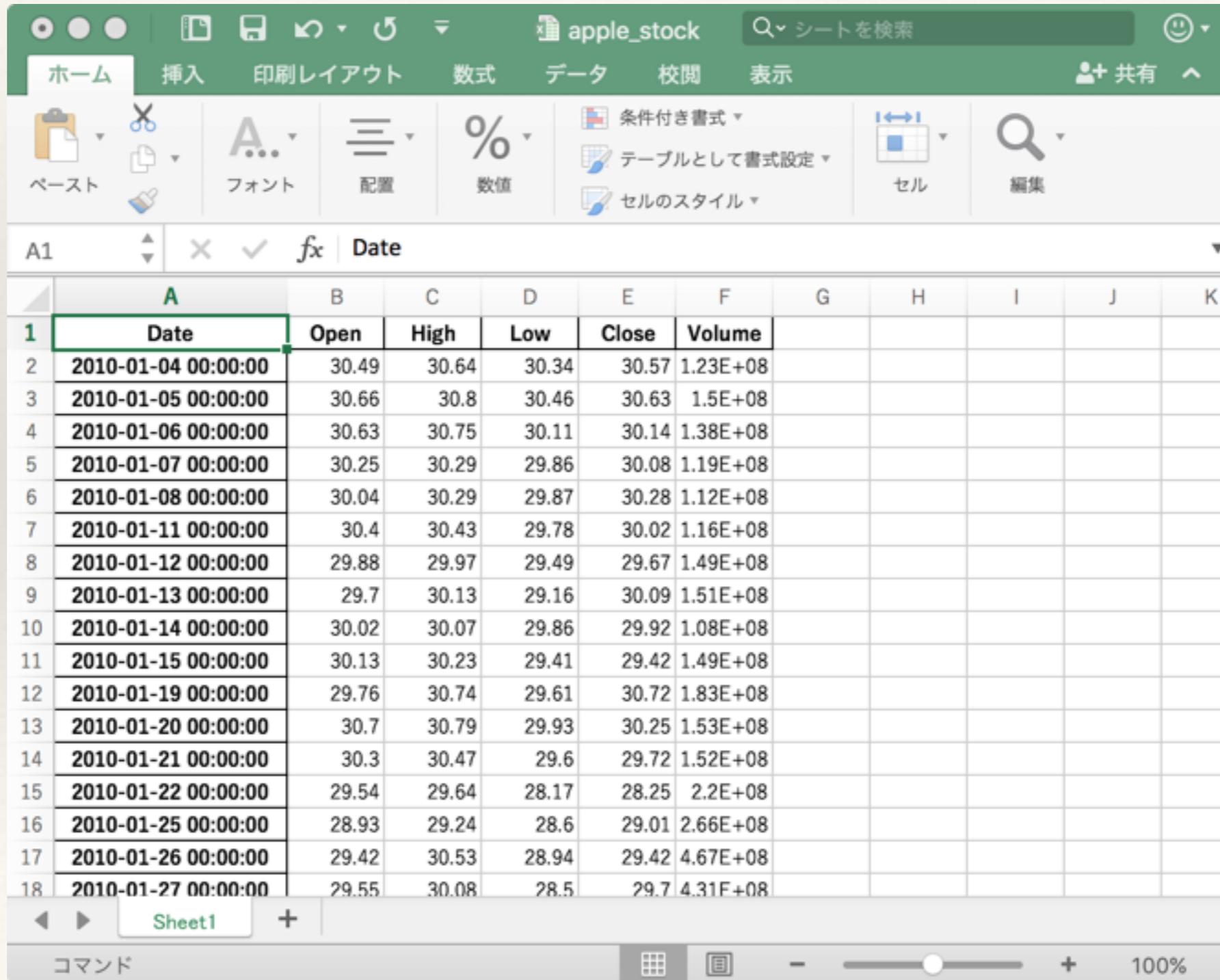
| | Open | High | Low | Close | Volume |
|------------|-------|-------|-------|-------|-----------|
| Date | | | | | |
| 2010-01-04 | 30.49 | 30.64 | 30.34 | 30.57 | 123432050 |
| 2010-01-05 | 30.66 | 30.80 | 30.46 | 30.63 | 150476004 |
| 2010-01-06 | 30.63 | 30.75 | 30.11 | 30.14 | 138039594 |
| 2010-01-07 | 30.25 | 30.29 | 29.86 | 30.08 | 119282324 |
| 2010-01-08 | 30.04 | 30.29 | 29.87 | 30.28 | 111969081 |



残念ながら、米国企業ですが・・・

データをExcelファイルに

```
data.to_excel('apple_stock.xlsx')
```



| | A | B | C | D | E | F | G | H | I | J | K |
|----|---------------------|-------|-------|-------|-------|----------|---|---|---|---|---|
| 1 | Date | Open | High | Low | Close | Volume | | | | | |
| 2 | 2010-01-04 00:00:00 | 30.49 | 30.64 | 30.34 | 30.57 | 1.23E+08 | | | | | |
| 3 | 2010-01-05 00:00:00 | 30.66 | 30.8 | 30.46 | 30.63 | 1.5E+08 | | | | | |
| 4 | 2010-01-06 00:00:00 | 30.63 | 30.75 | 30.11 | 30.14 | 1.38E+08 | | | | | |
| 5 | 2010-01-07 00:00:00 | 30.25 | 30.29 | 29.86 | 30.08 | 1.19E+08 | | | | | |
| 6 | 2010-01-08 00:00:00 | 30.04 | 30.29 | 29.87 | 30.28 | 1.12E+08 | | | | | |
| 7 | 2010-01-11 00:00:00 | 30.4 | 30.43 | 29.78 | 30.02 | 1.16E+08 | | | | | |
| 8 | 2010-01-12 00:00:00 | 29.88 | 29.97 | 29.49 | 29.67 | 1.49E+08 | | | | | |
| 9 | 2010-01-13 00:00:00 | 29.7 | 30.13 | 29.16 | 30.09 | 1.51E+08 | | | | | |
| 10 | 2010-01-14 00:00:00 | 30.02 | 30.07 | 29.86 | 29.92 | 1.08E+08 | | | | | |
| 11 | 2010-01-15 00:00:00 | 30.13 | 30.23 | 29.41 | 29.42 | 1.49E+08 | | | | | |
| 12 | 2010-01-19 00:00:00 | 29.76 | 30.74 | 29.61 | 30.72 | 1.83E+08 | | | | | |
| 13 | 2010-01-20 00:00:00 | 30.7 | 30.79 | 29.93 | 30.25 | 1.53E+08 | | | | | |
| 14 | 2010-01-21 00:00:00 | 30.3 | 30.47 | 29.6 | 29.72 | 1.52E+08 | | | | | |
| 15 | 2010-01-22 00:00:00 | 29.54 | 29.64 | 28.17 | 28.25 | 2.2E+08 | | | | | |
| 16 | 2010-01-25 00:00:00 | 28.93 | 29.24 | 28.6 | 29.01 | 2.66E+08 | | | | | |
| 17 | 2010-01-26 00:00:00 | 29.42 | 30.53 | 28.94 | 29.42 | 4.67E+08 | | | | | |
| 18 | 2010-01-27 00:00:00 | 29.55 | 30.08 | 28.5 | 29.7 | 4.31E+08 | | | | | |

データベースに格納

```
import sqlite3
```

```
dbname = 'stock_price.db'  
conn = sqlite3.connect(dbname)  
c = conn.cursor()
```

```
_sql = 'create table apple (date char(8), price float)'  
c.execute(_sql)
```

```
close = data['Close']  
_sql = 'insert into apple values(?, ?)'  
_data = [(x.strftime('%Y%m%d'), y) for x,y in zip(close.index, close.values)]
```

```
c.executemany(_sql, _data)  
conn.commit()  
conn.close()
```

データベースからの読み込み

```
import pandas as pd
dbname = 'stock_price.db'
conn = sqlite3.connect(dbname)

_sql = "select date,price from apple where date >= '20160101'"
data = pd.read_sql_query(_sql, conn, index_col='date', parse_dates=['date'])
```

| | price |
|------------|-------|
| date | |
| 2010-01-04 | 30.57 |
| 2010-01-05 | 30.63 |
| 2010-01-06 | 30.14 |
| 2010-01-07 | 30.08 |
| 2010-01-08 | 30.28 |
| 2010-01-11 | 30.02 |
| 2010-01-12 | 29.67 |
| 2010-01-13 | 30.09 |
| 2010-01-14 | 29.92 |

indexになっているdate列が日付として保存されているので、時系列データになっている

時系列データの加工

```
data.resample('M').mean()
```

| | price |
|------------|------------|
| date | |
| 2016-01-31 | 98.428947 |
| 2016-02-29 | 95.746500 |
| 2016-03-31 | 104.267273 |
| 2016-04-30 | 106.739048 |
| 2016-05-31 | 94.974762 |
| 2016-06-30 | 96.622273 |
| 2016-07-31 | 98.556500 |
| 2016-08-31 | 107.665217 |
| 2016-09-30 | 110.857143 |

```
data.resample('W').mean()
```

| | price |
|------------|----------|
| date | |
| 2016-01-10 | 100.4340 |
| 2016-01-17 | 98.5060 |
| 2016-01-24 | 97.7925 |
| 2016-01-31 | 96.8560 |
| 2016-02-07 | 95.5760 |
| 2016-02-14 | 94.3920 |
| 2016-02-21 | 96.7650 |
| 2016-02-28 | 96.2680 |
| 2016-03-06 | 100.4960 |

直近3ヶ月の平均株価

```
m3 = data.resample('M').mean().iloc[-3:]
```

PowerPointにする

Pytho-pptxという外部パッケージがあります

```
from pptx import Presentation
from pptx.util import Cm
```

```
prs = Presentation()
title_only_slide_layout = prs.slide_layouts[5]
slide = prs.slides.add_slide(title_only_slide_layout)
shapes = slide.shapes
shapes.title.text = 'アップル直近3ヶ月の株価'
```

```
rows = 4
cols = 2
left = Cm(2.0)
top = Cm(6.0)
width = Cm(6.0)
height = Cm(0.8)
```

データの出力と書き込み

```
table = shapes.add_table(rows, cols, left, top, width, height).table

table.columns[0].width = Cm(8.0)
table.columns[1].width = Cm(4.0)

table.cell(0, 0).text = '年月'
table.cell(0, 1).text = '平均株価'

for i, idx in enumerate(m3.index):
    table.cell(i+1, 0).text = idx.strftime('%Y年%m月%d日')
    table.cell(i+1, 1).text = '{:.2f}'.format(m3.ix[idx]['price'])

prs.save('apple_stock.pptx')
```

1 アップル直近3ヶ月の株価

| 年月 | 平均株価 |
|-------------|--------|
| 2017年02月28日 | 133.71 |
| 2017年03月31日 | 140.62 |
| 2017年04月30日 | 143.78 |

アップル直近3ヶ月の株価

| 年月 | 平均株価 |
|-------------|--------|
| 2017年02月28日 | 133.71 |
| 2017年03月31日 | 140.62 |
| 2017年04月30日 | 143.78 |

クリックしてノートを入力

PDFにしたい

ReportLab

<http://www.reportlab.com/>

```
from reportlab.pdfbase import pdfmetrics
from reportlab.pdfbase.cidfonts import UnicodeCIDFont
from reportlab.lib.units import cm
from reportlab.lib import colors
from reportlab.platypus import SimpleDocTemplate, Table, TableStyle
```

```
font_name = 'HeiseiKakuGo-W5'
pdfmetrics.registerFont(UnicodeCIDFont(font_name))
doc = SimpleDocTemplate('apple_stock.pdf')
elements = []
```

データを出力

```
data = [['年月', '平均株価']]

for i,idx in enumerate(m3.index):
    _temp = []
    _temp.append(idx.strftime('%Y年%m月%d日'))
    _temp.append('{:.2f}'.format(m3.ix[idx]['price']))
    data.append(_temp)

table = Table(data, (5.0*cm), (1.0* cm))
table.setStyle(TableStyle([
    ('FONT', (0, 0), (-1, -1), font_name, 12),
    ('BOX', (0, 0), (-1, -1), 0.25, colors.black),
    ('INNERGRID', (0, 0), (-1, -1), 0.25, colors.black),
    ('ALIGN', (0, 0), (-1, 0), 'CENTER'),
    ('ALIGN', (1, 1), (-1, -1), 'RIGHT'),
    ('BACKGROUND', (0, 0), (-1, 0), colors.lightblue)
]))

elements.append(table)
doc.build(elements)
```



| 年月 | 平均株価 |
|-------------|--------|
| 2017年02月28日 | 133.71 |
| 2017年03月31日 | 140.62 |
| 2017年04月30日 | 143.78 |

まとめ

- ❖ Pythonがブームです
- ❖ Anacondaがおすすめ
- ❖ データ処理から、スライド、ドキュメントの作成まで
- ❖ 日頃のPC作業にPythonが活躍できる場所を探してみてください

ご静聴ありがとうございました