

# 파이썬 101

무한한 라이브러리, 저 너머로!

September 18, 2019

51일 전: 왜 파이썬?

WHY Python?

Python lets you

- ▶ care less about the details.
- ▶ think at the high level.

공감 되시나요?

오늘: 왜 파이썬?

## WHY Python?

Python lets you

- ▶ care less about the details.
- ▶ think at the high level.

그리고 다양한 라이브러리들

파이선이 슈퍼히어로라면...

edufy.me

# Python

Has a superpower for everything.

```
import lasers  
import x_ray_vision  
import flight  
import super strength  
import super hearing  
import time travel  
... and the list goes on.
```



근데 진짜임

```
import antigravity
```

## 유용한 모듈, 라이브러리와 프로그램들 일부

- ▶ pdb: 디버깅
- ▶ PyPy: 빠른 파이썬
- ▶ pip: 파이썬 패키지 관리자
- ▶ Numpy: 빠른 계산
- ▶ Pillow: 이미지 처리
- ▶ OpenCV: 컴퓨터 비전
- ▶ Request: HTTP 요청 주고받기
- ▶ Scrapy: 웹크롤링
- ▶ Selenium: 웹브라우저 자동화
- ▶ KoNLPY: 한국어 처리
- ▶ tossi: 한국어 조사처리
- ▶ Ren'Py: 비주얼 노벨 파연사 만들기
- ▶ Pygame: 2D 게임 만들기

pdb

## The Python Debugger

# 지금까지의 디버깅

```
def wrong_function(n):  
    if n == 2 and n == 3:  
        print("n is: ", n)  
        return True  
    if n % 2 == 0 or n < 2:  
        print("n is now: ", n)  
        return False  
    for i in range(3, int(n**0.5+1), 2):  
        print(n, i)  
        if n % i == 0:  
            return False  
        print("should not print if n is prime", i)  
    if n == 3:  
        print("if I can read this, I am XXXXed")  
    return True
```



# 좋은 디버깅

```
~/wor../kim code > pdb gbach.py
> /home/indiofish/workspace/kim_code/gbach.py(1)<module>()
-> n = int(input(""))
(Pdb) list
1  -> n = int(input(""))
2
3      import math
4
5      def is_prime(n):
6          if n == 2 or n == 3:
7              return True
8          if n % 2 == 0 or n < 2:
9              return False
10         for i in range(3,int(n**0.5+1),2):
11             if n % i == 0:
(Pdb) next
10
> /home/indiofish/workspace/kim_code/gbach.py(3)<module>()
-> import math
(Pdb) next
> /home/indiofish/workspace/kim_code/gbach.py(5)<module>()
-> def is_prime(n):
(Pdb) print(n)
10
(Pdb) step
> /home/indiofish/workspace/kim_code/gbach.py(16)<module>()
-> for k in range(1, n//2+1):
(Pdb) step
> /home/indiofish/workspace/kim_code/gbach.py(24)<module>()
-> if is_prime(k) and is_prime(n-k):
(Pdb) print(k)
1
(Pdb) !n=30
(Pdb) print(n)
30
(Pdb) is_prime(100)
False
```

## 좋은 디버깅

- ▶ `help`  
가능한 명령어를 보여준다.
- ▶ `break` (줄번호|함수이름)  
n번째 줄이나 함수에 도달하면 일시정지
- ▶ `print()`  
우리가 아는 출력함수
- ▶ `!x = 3`  
값 바꾸기
- ▶ `next`  
현재 함수의 다음줄로 넘어간다.(중간의 함수는 그냥 실행)
- ▶ `step`  
`step into`: 함수 안으로 들어간다.

# 좋은 디버깅

## 장점들

- ▶ 프로그램을 켜다켰다하지 않고도 테스트 가능
- ▶ 프로그램을 바꾸지 않고도 테스트 가능
- ▶ 멋있음

PyPy



PyPy

Python으로 실행하는 Python

```
python program.py == pypy program.py
```

JIT 컴파일과 파이썬 코드를 C로 번역하는 과정 등을 거쳐 표준 python 구현체보다 2-10배 정도 빠를 때도 있고, 조금 느릴 때도 있습니다.

*pip install you-name-it*

파이썬에 기본적으로 포함되어 있는, 패키지 관리자. 일일이 홈페이지에 들어가서 설치파일을 받지 않아도 되는 장점이 있습니다.

# Numpy

*pip install numpy*

Anaconda 등에 포함되어 있어 써보셨을 것 같은 Numpy

## 왜 Numpy?

딥러닝 등의 근간이 되는 행렬<sup>1</sup>을 빠르게 계산 해줍니다. 물론 그외에도 많은 수학기능이 있습니다.

---

<sup>1</sup>김모 교수님 "미적분학은 한물갔고 이젠 선형대수학의 시대입니다 음하하!"



# Numpy

```
import numpy as np
SIZE = 500
a = np.array([[randint(-100,100) for _ in range(SIZE)] for _ in range(SIZE)])
b = np.array([[randint(-100,100) for _ in range(SIZE)] for _ in range(SIZE)])
c = np.dot(a, b)
```

500 x 500 행렬의 곱셈

## Numpy vs 저

```
singlecore took: 19.164047956466675  
multicore took: 6.122005462646484  
numpy took: 0.12259554862976074
```

코드는 깃 저장소의 `mat_mul.py`에서 확인 가능합니다.

# Pillow

*pip install Pillow*

Python Image Library (PIL)의 변형판  
사진을 늘리고, 줄이고, 변형하는데 유용합니다.  
<https://pillow.readthedocs.io/en/stable/>

# Pillow

```
from PIL import Image, ImageFilter
import numpy as np

im = Image.open("homer.png")
im.show()
im = im.convert('RGB')
out = im.filter(ImageFilter.FIND_EDGES)
out.show("edges only")
out = im.transpose(Image.FLIP_LEFT_RIGHT)
out.show("homer in the mirror")
out = im.convert('LA')
out.show("noire homer")

# images are actually arrays!
pix = np.array(im)
print(pix)
```

# OpenCV

*pip install opencv-python*

[https://docs.opencv.org/master/d6/d00/tutorial\\_py\\_root.html](https://docs.opencv.org/master/d6/d00/tutorial_py_root.html)

이미지처리 + 컴퓨터비전을 위한 라이브러리.

Pillow보다 강력하지만, 복잡하네요

컴퓨터비전 이론을 몰라도 Ctrl CV로 사용할 수 있으나, 왜 결과가 나오는 지 모를 수가 있음.

심슨



누구 눈일까요? 5초



# OpenCV - Template Matching Ctrl CVed

```
import cv2 as cv
import numpy as np

"""copied from opencv docs & modified to use the simpsons"""
img_rgb = cv.imread('simpsons.jpeg')
img_gray = cv.cvtColor(img_rgb, cv.COLOR_BGR2GRAY)
template = cv.imread('eyes3.jpeg',0)
w, h = template.shape[::-1]

"""cv2 has this match function already implemented"""
res = cv.matchTemplate(img_gray,template,cv.TM_CCOEFF_NORMED)
threshold = 0.8
loc = np.where( res >= threshold)
# draw a rectangle around the result
for pt in zip(*loc[::-1]):
    cv.rectangle(img_rgb, pt, (pt[0] + w, pt[1] + h), (0,0,255), 2)

cv.imwrite('res.png', img_rgb) # save
cv.imshow('res.png',img_rgb) # show result
cv.waitKey(0) # if no wait, result window closes right away
cv.destroyAllWindows()
```



아하.



# Requests

```
import cv2 as cv
import numpy as np

"""copied from opencv docs & modified to use the simpsons"""
img_rgb = cv.imread('simpsons.jpeg')
img_gray = cv.cvtColor(img_rgb, cv.COLOR_BGR2GRAY)
template = cv.imread('eyes3.jpeg',0)
w, h = template.shape[::-1]

"""cv2 has this match function already implemented"""
res = cv.matchTemplate(img_gray,template,cv.TM_CCOEFF_NORMED)
threshold = 0.8
loc = np.where( res >= threshold)
# draw a rectangle around the result
for pt in zip(*loc[::-1]):
    cv.rectangle(img_rgb, pt, (pt[0] + w, pt[1] + h), (0,0,255), 2)

cv.imwrite('res.png', img_rgb) # save
cv.imshow('res.png',img_rgb) # show result
cv.waitKey(0) # if no wait, result window closes right away
cv.destroyAllWindows()
```

# Requests

*pip install requests*

HTTP 요청<sup>1</sup>을 주고받는 모듈

<https://2.python-requests.org/en/master/>

---

<sup>1</sup>비약하면 인터넷에서 보는 것

# Requests

```
import requests

BASE_ADDRESS = 'https://namu.wiki/w/'

with requests.Session() as s:
    keyword = input()

    ret = s.get(BASE_ADDRESS + keyword)

    print(ret.text)
```

나무위키 링크에서 무언가 받아오기

# Requests

```
from bs4 import BeautifulSoup
import json, random, re, requests

BASE_URL = 'https://www.instagram.com/accounts/login/'
LOGIN_URL = BASE_URL + 'ajax/'

headers_list = [
    "Mozilla/5.0 (Windows NT 5.1; rv:41.0) Gecko/20100101\"\\
    " Firefox/41.0",
    "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_2)\"\\
    " AppleWebKit/601.3.9 (KHTML, like Gecko) Version/9.0.2\"\\
    " Safari/601.3.9",
    "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:15.0)\"\\
    " Gecko/20100101 Firefox/15.0.1",
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36\"\\
    " (KHTML, like Gecko) Chrome/42.0.2311.135 Safari/537.36\"\\
    " Edge/12.246" ]

USERNAME = 'o_taeji'
PASSWD = 'ro6sk7fl8'
USER_AGENT = headers_list[random.randrange(0,4)]

session = requests.Session()
session.headers = {'user-agent': USER_AGENT}
session.headers.update({'Referer': BASE_URL})
req = session.get(BASE_URL)
soup = BeautifulSoup(req.content, 'html.parser')
body = soup.find('body')

pattern = re.compile('window._sharedData')
script = body.find("script", text=pattern)

script = script.get_text().replace('window._sharedData = ', '')[::-1]
data = json.loads(script)
```

## Requests - 단점

사이트마다 이런 종류의 접속을 막기 위해 다양한 방법을 사용해서, 사이트의 구조를 잘 찾아봐야 짤 수 있고, 귀찮다.  
정적인 웹페이지에서는 잘 되지만, js를 많이 사용하는 동적인 웹페이지는 정보를 받아오기 어렵다.

## 그래서 Selenium

*pip install selenium*

브라우저를 직접 이용하여, 웹사이트에 접근 할 수 있다.

만든 웹앱을 테스트하는 용도의 프로그램이지만, 얼마든지 응용 가능.

# Selenium

*pip install selenium*

+

자기가 사용할 브라우저의 웹드라이버를 설치해야 한다.

크롬:

<https://sites.google.com/a/chromium.org/chromedriver/downloads>

파이어폭스: <https://github.com/mozilla/geckodriver>

익스플로러: ??

화면뜨는 것이 싫다면, headless 크롬을 이용하면 된다.



# Selenium

```
from selenium import webdriver
from getpass import getpass

snu_id = input("ID:")
pwd = getpass("PASSWORD:")
driver = webdriver.Chrome('/home/thoum/chromedriver')
driver.implicitly_wait(3)
driver.get("https://etl.snu.ac.kr")
driver.find_element_by_id("input-username").send_keys(snu_id)
driver.find_element_by_id("input-password").send_keys(pwd)
driver.find_element_by_name("loginbutton").click()
```

# Selenium

```
from selenium import webdriver
from getpass import getpass
from selenium.webdriver.common.by import By
import time

userid = input("ID:")
pwd = getpass("PASSWORD:")
driver = webdriver.Chrome('/home/thoum/chromedriver')
driver.implicitly_wait(3)
driver.get("https://www.instagram.com/accounts/login/?source=auth_switcher")
driver.find_element_by_name("username").send_keys(userid)
driver.find_element_by_name("password").send_keys(pwd)

#might have to sleep here if slow

#find it by inspection mode
path_of_loginbtn = "//*[@id=\"react-root\"]/section/main/div\
                    /article/div/div[1]/div/form/div[4]/button/div"
driver.find_element_by_xpath(path_of_loginbtn).submit()
```

# Selenium

특정 HTML 요소에 접근하는 다양한 방법이 있고, 되는 방법도, 안되는 방법도 있습니다 (복불복?)

# Selenium

인터넷에서 이것저것 긁어왔는데, 그 다음엔?

# KoNLPy

한국어 처리 라이브러리

# KoNLPy

설치하기가 조금 까다로워서 불편합니다.

<https://konlpy-ko.readthedocs.io/ko/v0.4.3/#start>

# KoNLPy

```
from konlpy.tag import Kkma  
  
k = Kkma()  
s = "치킨과 맥주를 한강에서 먹고 싶은 날씨"  
print(k.nouns(s)) # 명사들  
print(k.pos(s)) # 형태소 분석
```



이런 문제(을)를 해결하는 라이브러리입니다.

<https://github.com/what-studio/tossi>



# 게임제작툴

은 생략.

다른 라이브러리들?



🔍 How to do something in python



🔍 how to do something in python

Google 검색

I'm Feeling Lucky

부적절한 예상 검색어 신고