Fastcampus

Computer Science SCHOOL

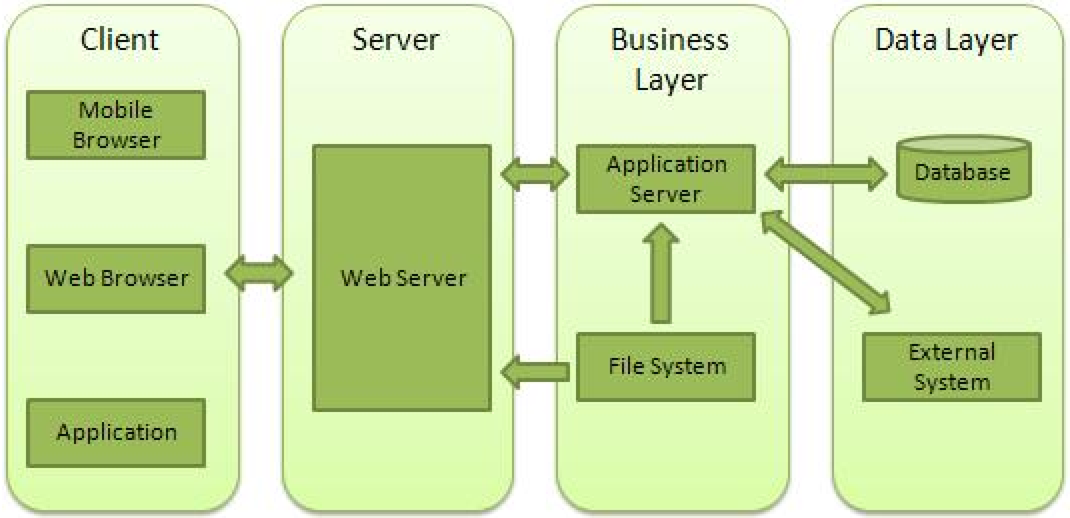
Network Basic (2)

2017.2.15

Civilization

Web Programming

Web architecture

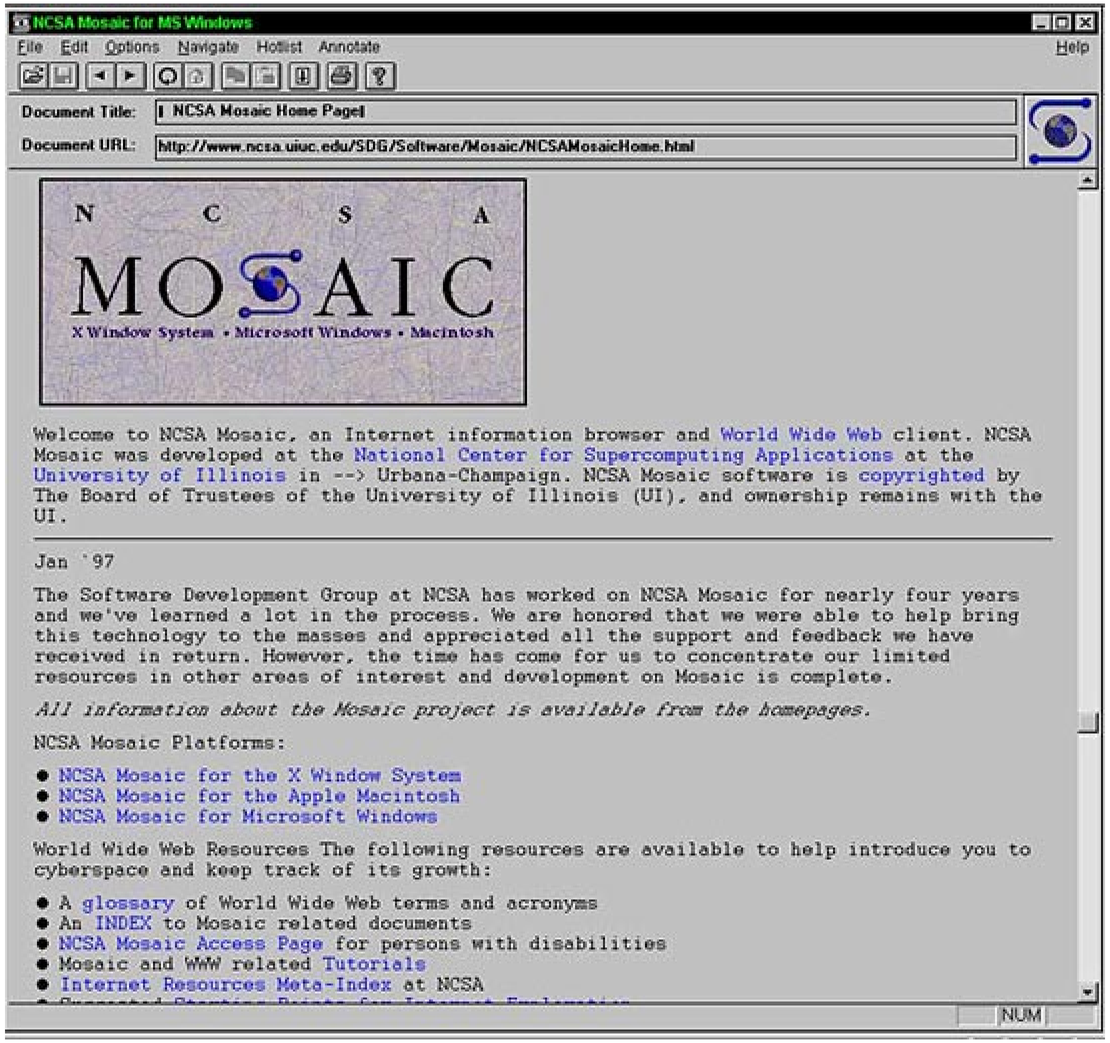


웹 개발 패턴의 변화

- 1991 ~ 1999: Sir Timothy John "Tim" Berners-Lee가 하이퍼텍스트 기반의 프로젝트를 제안한 이후 정적인 컨텐츠들을 중심으로 한 웹 기술이 발달
- 1999 ~ 2009: Linux, Apache, Mysql, Php 중심의 동적인 서버, 정적인 클라이언트 모델이 지속됨
- 2010 ~ 현재: javaScript!! (Dynamic Web Client)

Web Browser

Mosaic(1993)


NCSA Mosaic for MS Windows
File Edit Options Navigate Hotlist Annotate Help
Document Title: NCSA Mosaic Home Page
Document URL: http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/NCSAMosaicHome.html
N C S A MOSAIC
X Window System • Microsoft Windows • Macintosh
NUM

Welcome to NCSA Mosaic, an Internet information browser and World Wide Web client. NCSA Mosaic was developed at the National Center for Supercomputing Applications at the University of Illinois in --> Urbana-Champaign. NCSA Mosaic software is copyrighted by The Board of Trustees of the University of Illinois (UI), and ownership remains with the UI.

Jan '97

The Software Development Group at NCSA has worked on NCSA Mosaic for nearly four years and we've learned a lot in the process. We are honored that we were able to help bring this technology to the masses and appreciated all the support and feedback we have received in return. However, the time has come for us to concentrate our limited resources in other areas of interest and development on Mosaic is complete.

All information about the Mosaic project is available from the homepages.

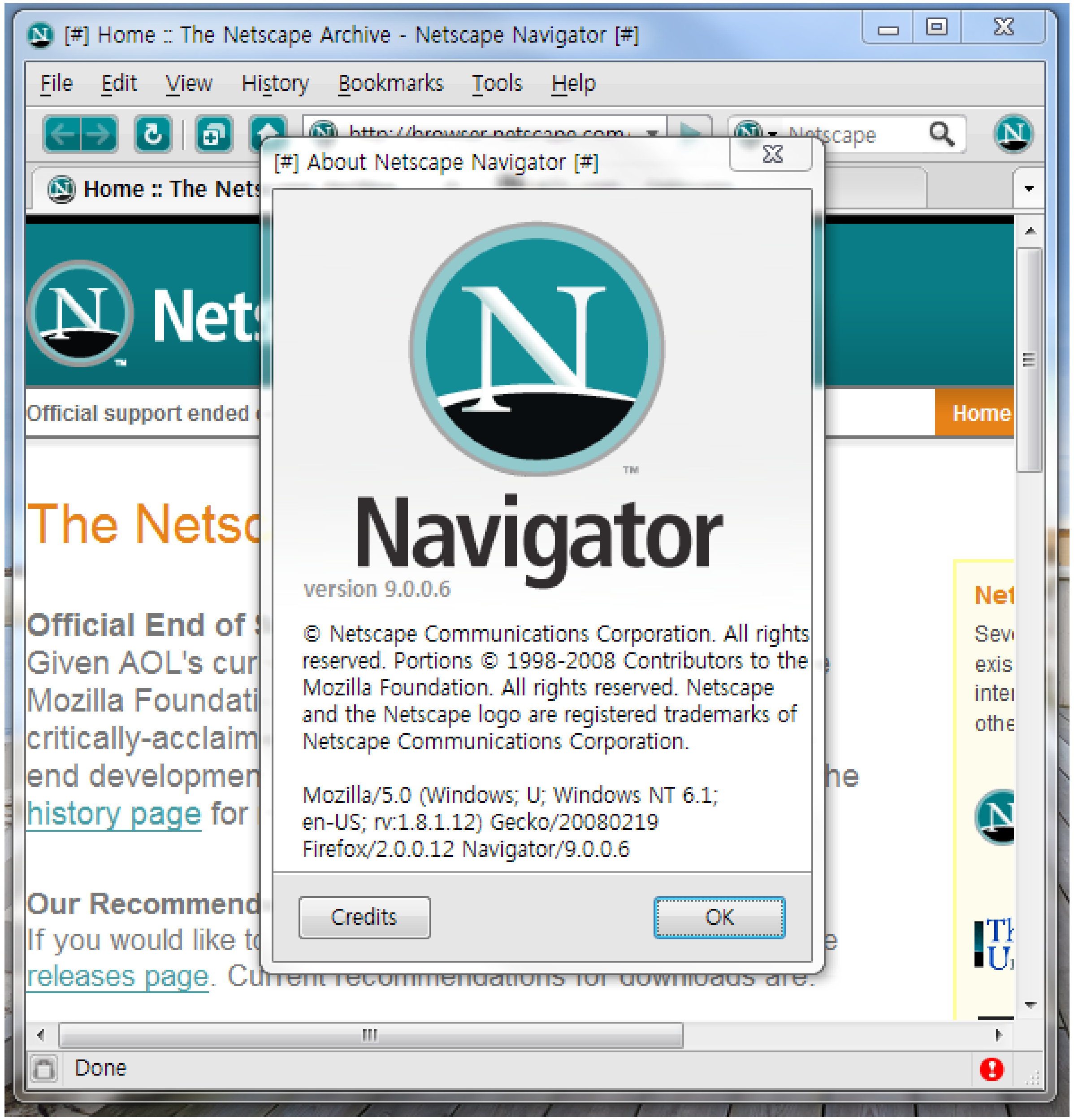
NCSA Mosaic Platforms:

- NCSA Mosaic for the X Window System
- NCSA Mosaic for the Apple Macintosh
- NCSA Mosaic for Microsoft Windows

World Wide Web Resources The following resources are available to help introduce you to cyberspace and keep track of its growth:

- A glossary of World Wide Web terms and acronyms
- An INDEX to Mosaic related documents
- NCSA Mosaic Access Page for persons with disabilities
- Mosaic and WWW related Tutorials
- Internet Resources Meta-Index at NCSA

Netscape(1994)


[#] Home :: The Netscape Archive - Netscape Navigator [#]
File Edit View History Bookmarks Tools Help
http://browser.netscape.com/
Netscape
Home :: The Nets
Home
Done

[#] About Netscape Navigator [#]

Navigator

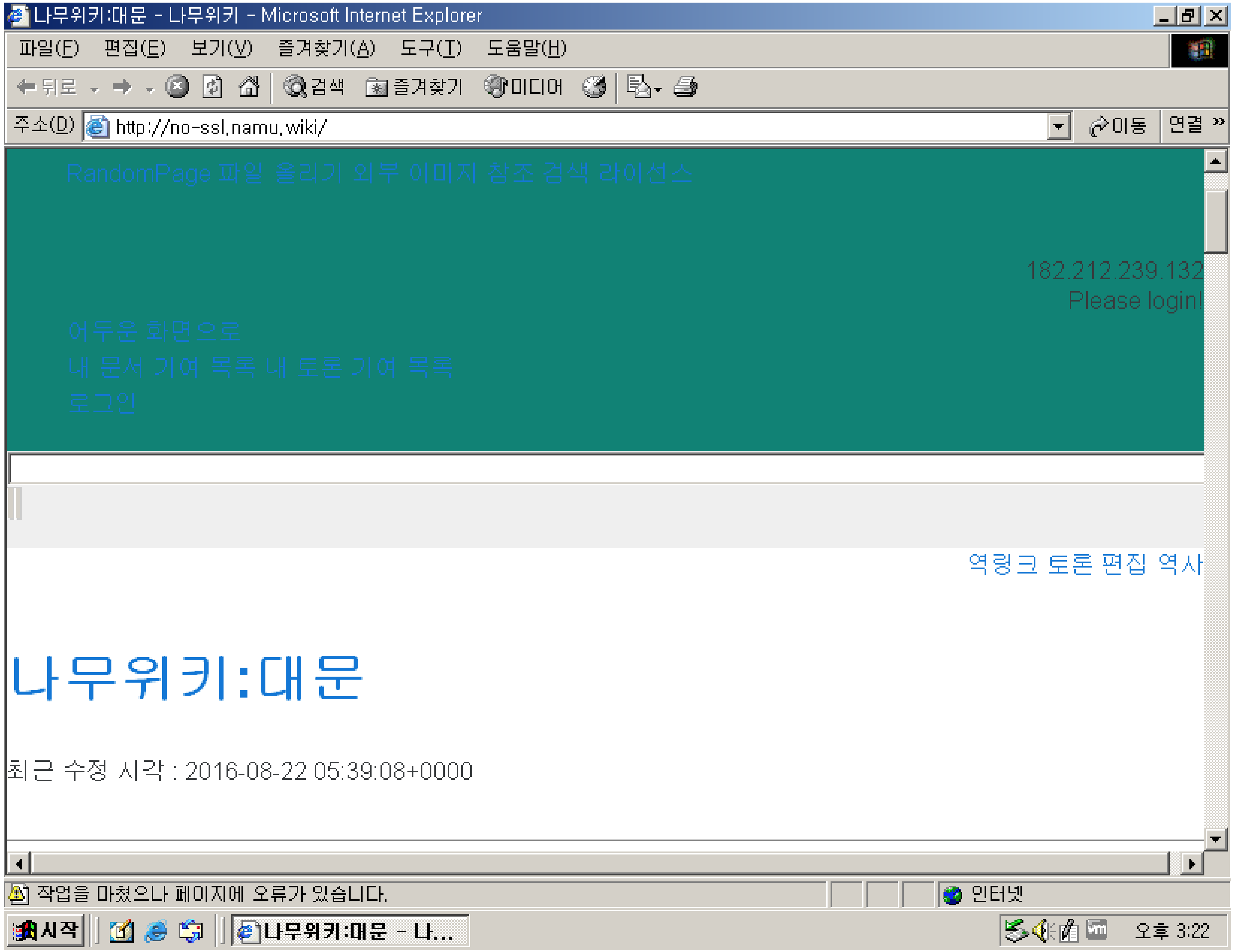
version 9.0.0.6

© Netscape Communications Corporation. All rights reserved. Portions © 1998-2008 Contributors to the Mozilla Foundation. All rights reserved. Netscape and the Netscape logo are registered trademarks of Netscape Communications Corporation.

Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.8.1.12) Gecko/20080219 Firefox/2.0.0.12 Navigator/9.0.0.6

Credits OK

Internet Explorer(1995)



나무위키:대문 - 나무위키 - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
뒤로 검색 즐겨찾기 미디어
주소(D) http://no-ssl.namu.wiki/
이동 연결

RandomPage 파일 올리기 외부 이미지 참조 검색 라이선스

182.212.239.132
Please login!

어두운 화면으로
내 문서 기여 목록 내 토론 기여 목록
로그인

역링크 토론 편집 역사

나무위키:대문

최근 수정 시각 : 2016-08-22 05:39:08+0000

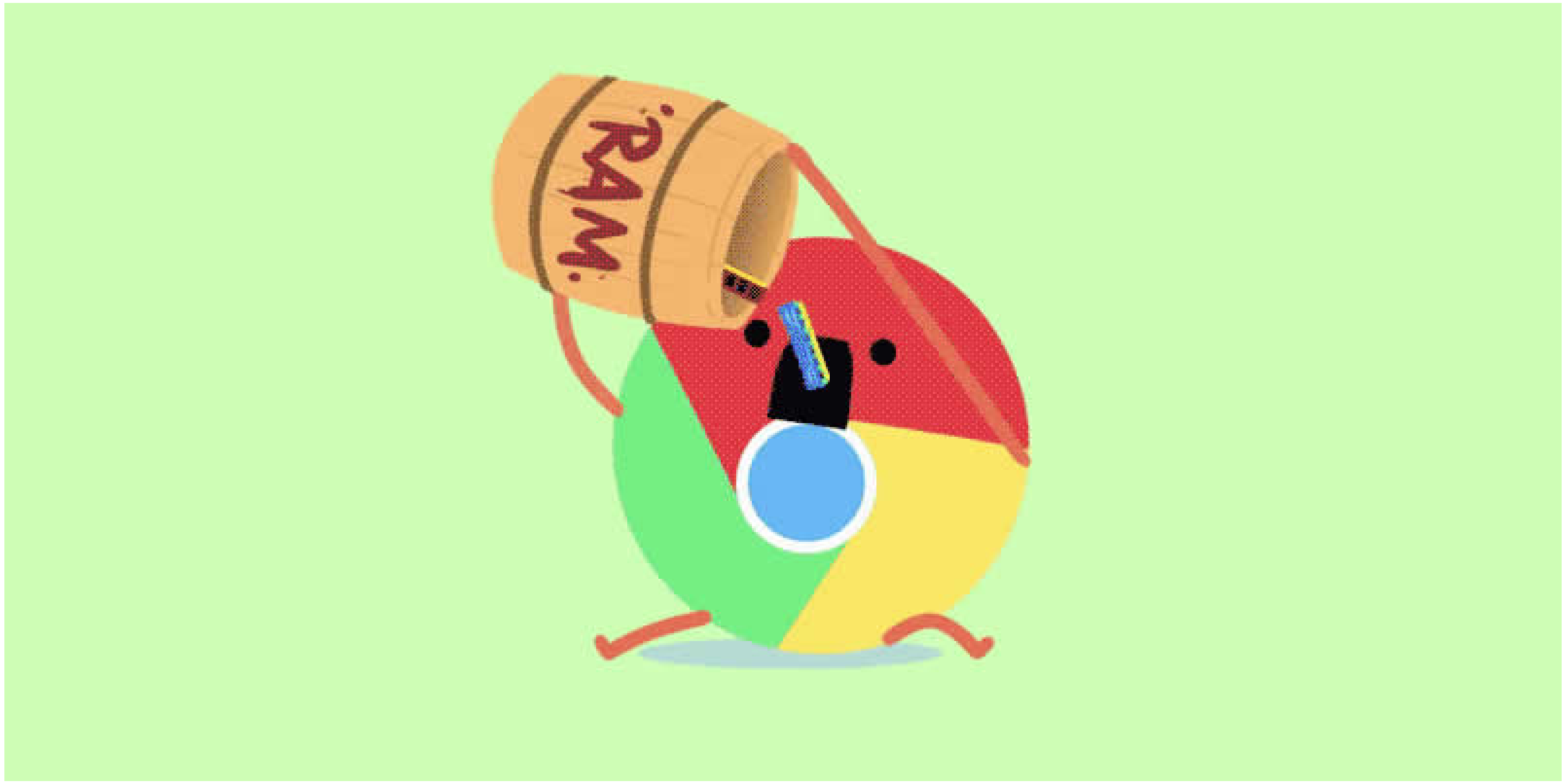
작업을 마쳤으나 페이지에 오류가 있습니다.

인터넷
시작
나무위키:대문 - 나...
오후 3:22

FireFox(2004)

4 (월)
08:00
불여우 (5) / 인터넷 브라우저
"어..크롬까는구나.. 아니 눈치보지 말고 설치해도
돼..난 괜찮아.."

Chrome(2008)



웹 개발의 현재

javaScript

Client-side

- HTML/CSS, javaScript
- jQuery, AJAX
- Front-end Web Framework
    - AngularJS
    - React.js
    - Vue.js
- CSS Framework
    - Bootstrap
    - Foundation

Server-side

- Depends on Language
    - PHP: Laravel
    - javaScript: Node.js(Express.js)
    - Java: Spring
    - C++, C#: ASP.net
    - Python: Django, Flask
    - Golang: itself
    - Ruby: Ruby on Rails

Database

- RDBMS
    - MySQL
    - PostgreSQL
    - MariaDB
- noSQL
    - MongoDB
    - CouchDB
    - Redis

etc

- celery (for Distributed Task Queue)
- github, Bitbucket, gitlab (for SCM)
- travis CI or jenkins (for Continuous Integration)
- slack, trello

URI, URL, URN

URI

- Uniform Resource Information
- `https://www.example.com/post/how-to-make-url`

URL

- Uniform Resource Locator
- `https://www.example.com/post/`

URN

- Uniform Resource Name
- `www.example.com/post/how-to-make-url`

REST API

`RE` presentational `S` tate `T` ransfer
`A` pplication `P` rogramming `I` nterface

`Resource` - URI
`Verb` - HTTP method
`Representations` - 표현

CRUD

Create

Read

Update

Delete

REST API **설계시 주의할 점**

- 버전관리 https://api.foo.com/v1/bar
- 명사형 사용 https://foo.com/showid/ --> https://foo.com/user/
- 반응형 https://foo.com/m/user/, https://m.foo.com/user/ (x)
- 언어코드 https://foo.com/kr/, https://kr.foo.com/ (x)
- 응답상태 코드 (200, 400, 500)

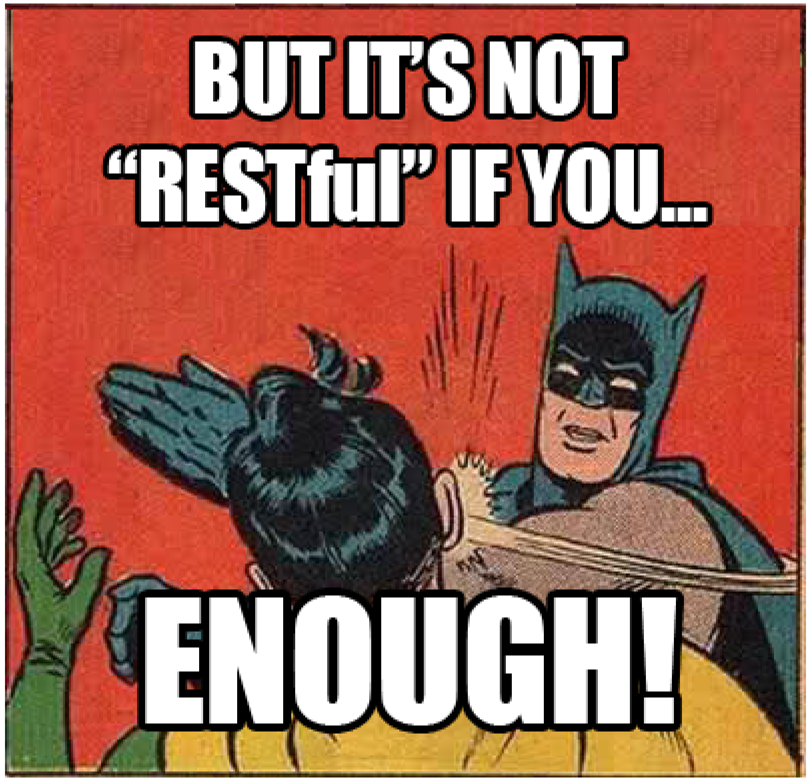
HTTP Response code

wikipedia

200, 201 - Success

400, 404 - Not found

500 - Server error


BUT IT'S NOT "RESTful" IF YOU...
ENOUGH!

Simple Server Framework: Flask

```
$ pip install flask
```

```
from flask import Flask


app = Flask(__name__)

@app.route('/')
def index():
        return 'hello world!'

if __name__ == '__main__':
        app.run(host='0.0.0.0')
```

c9.io

```
from flask import Flask
import os

app = Flask(__name__)

@app.route('/')
def index():
    return 'hello world!'

app.run(host=os.getenv('IP', '0.0.0.0'),port=int(os.getenv('PORT
```

Simple Server Framework: Flask

```python
from flask import Flask, render_template


app = Flask(__name__)

@app.route('/')
def index(name=None):
        return render_template('index.html', name=name)

@app.route('/about')
def about(name=None):
        return render_template('about.html', name=name)

if __name__ == '__main__':
        app.run(host='0.0.0.0')
```

Simple Server Framework: Flask

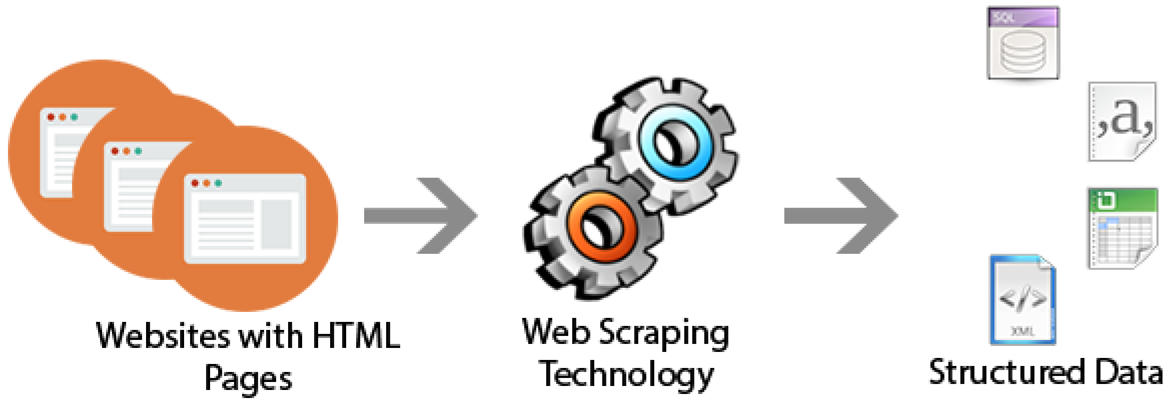
```
/
  simpleserver.py
  /templates
    index.html
    about.html
```

Flask with `socket.io`

Web Crawling with Python

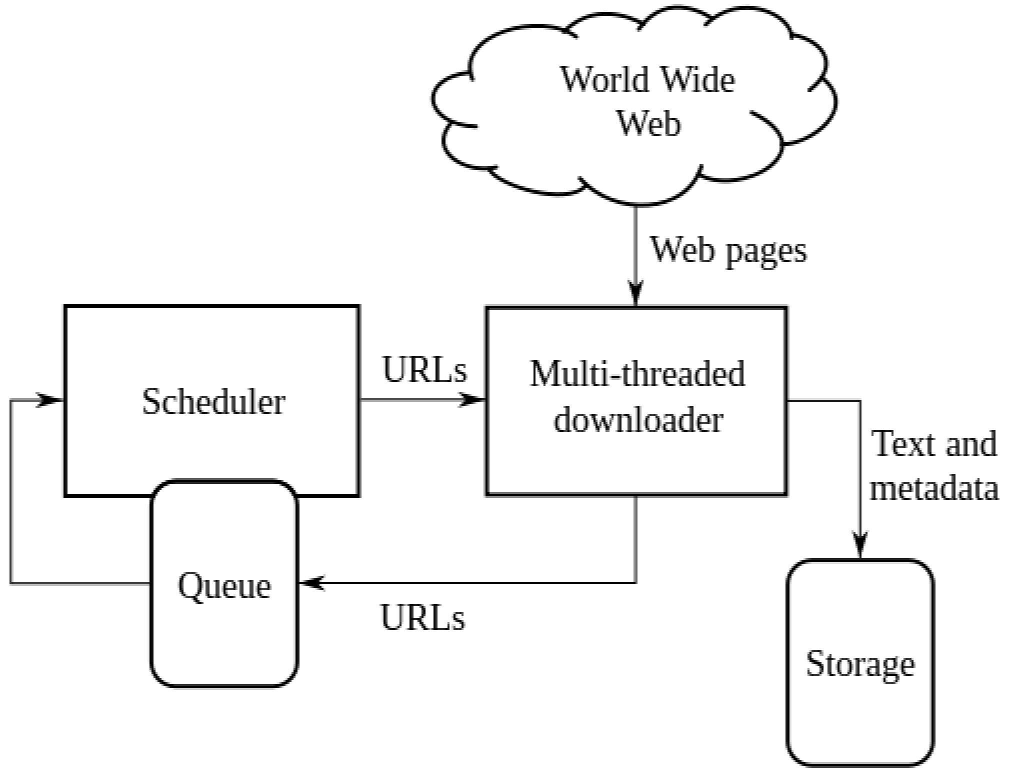
Scraping vs Crawling vs Parsing

Scraping: 데이터를 수집하는 행위



Scraping vs Crawling vs Parsing

Crawling: 조직적 자동화된 방법으로 월드 와이드 웹을 탐색하는 것



Scraping vs Crawling vs Parsing

Parsing: 문장 혹은 문서를 구성 성분으로 분해하고 위계관계를 분석하여 문장의 구조를 결정하는 것

iS THE MAN WHO iS TALL HAPPY?
FROM THE DIRECTOR OF "ETERNAL SUNSHINE OF THE SPOTLESS MIND"
A FILM BY MICHEL GONDRY
AN ANIMATED CONVERSATION WITH NOAM CHOMSKY
IN THEATERS 11/22 AND iTUNES 11/25

Caution!!

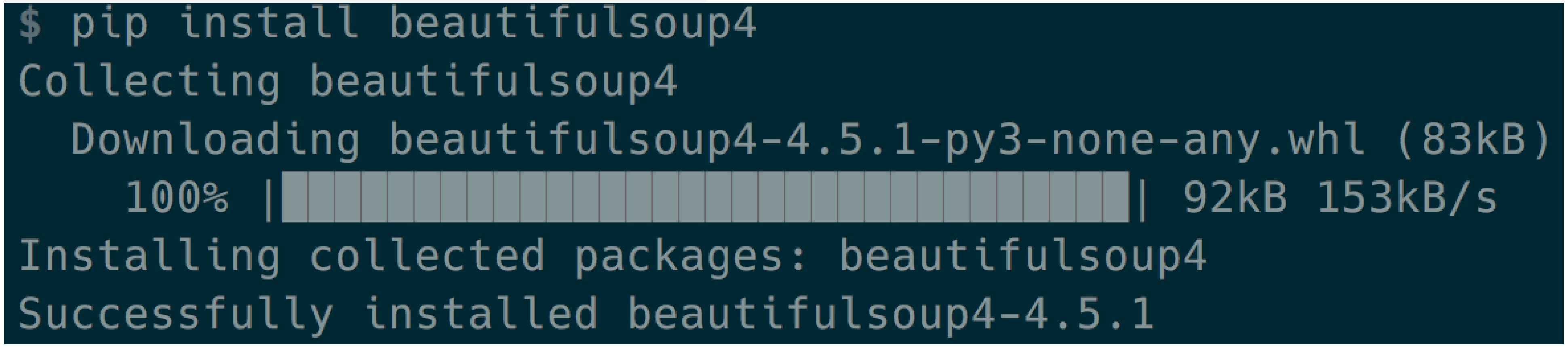
저작권 침해 위반 소지

- 웹사이트 운영자의 크롤링 금지 룰을 어길경우
- 월권하여 데이터베이스에 접근
- 타인의 경제적 이익을 침해할 경우
- 개인정보를 수집할 경우(전화번호, 주소, ..)

Beautiful Soup

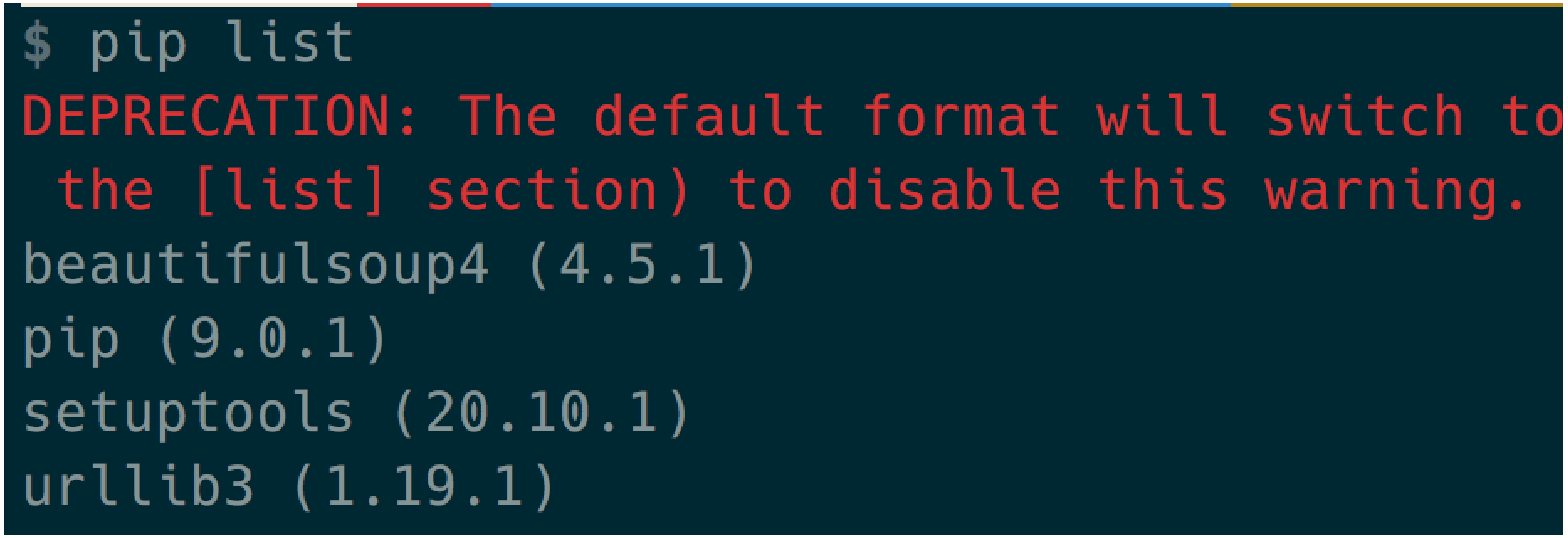
Web Scraping with Beautiful Soup

```
$ pip install beautifulsoup4
```



```
$ pip install beautifulsoup4
Collecting beautifulsoup4
  Downloading beautifulsoup4-4.5.1-py3-none-any.whl (83kB)
    100% |████████████████████████████████| 92kB 153kB/s
Installing collected packages: beautifulsoup4
Successfully installed beautifulsoup4-4.5.1
```

Web Scraping with Beautiful Soup



```
$ pip list
DEPRECATION: The default format will switch to
 the [list] section) to disable this warning.
beautifulsoup4 (4.5.1)
pip (9.0.1)
setuptools (20.10.1)
urllib3 (1.19.1)
```

Web Scraping with Beautiful Soup

```
>>> import urllib
>>> from bs4 import BeautifulSoup
>>> html = """
... <html><head><title>The Dormouse's story</title></head>
... <body>
... <p class="title"><b>The Dormouse's story</b></p>
...
... <p class="story">Once upon a time there were three little sisters; and their names were
... <a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
... <a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and
... <a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>;
... and they lived at the bottom of a well.</p>
...
... <p class="story">...</p>
... """
>>> soup = BeautifulSoup(html, 'html.parser')
>>> print(soup.prettify())
```

Web Scraping with Beautiful Soup

```
import urllib
from bs4 import BeautifulSoup
html = """

uglified html code

"""
soup = BeautifulSoup(html, "html.parser")
print(soup.prettify())
```

Web Scraping with Beautiful Soup

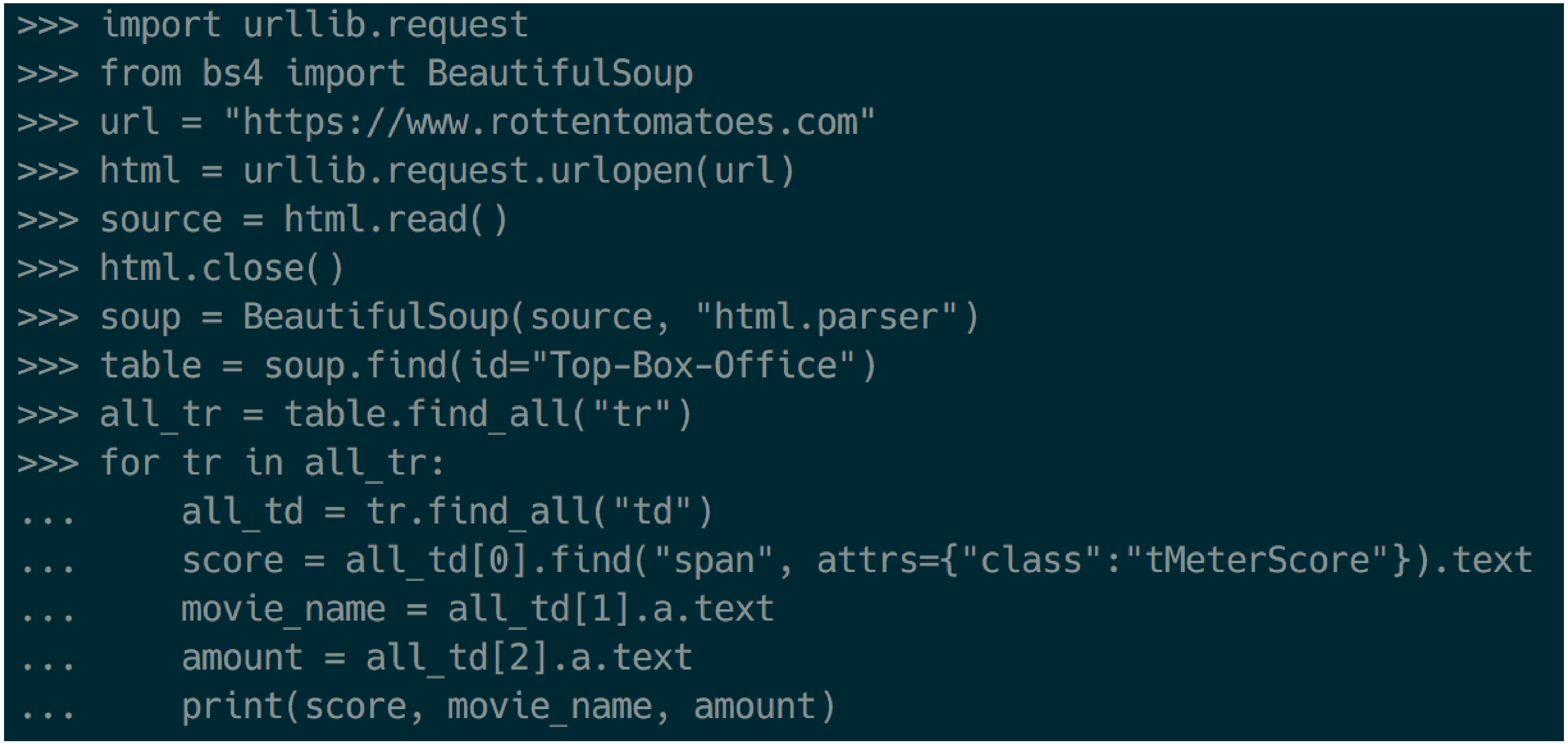
```
curl https://www.rottentomatoes.com
```

```
import urllib.request
from bs4 import BeautifulSoup


url = "https://www.rottentomatoes.com"
html = urllib.request.urlopen(url)
source = html.read()
html.close()

soup = BeautifulSoup(source, "html.parser")
print(soup)

table = soup.find(id="Top-Box-Office")
print(table)
```

Web Scraping with Beautiful Soup

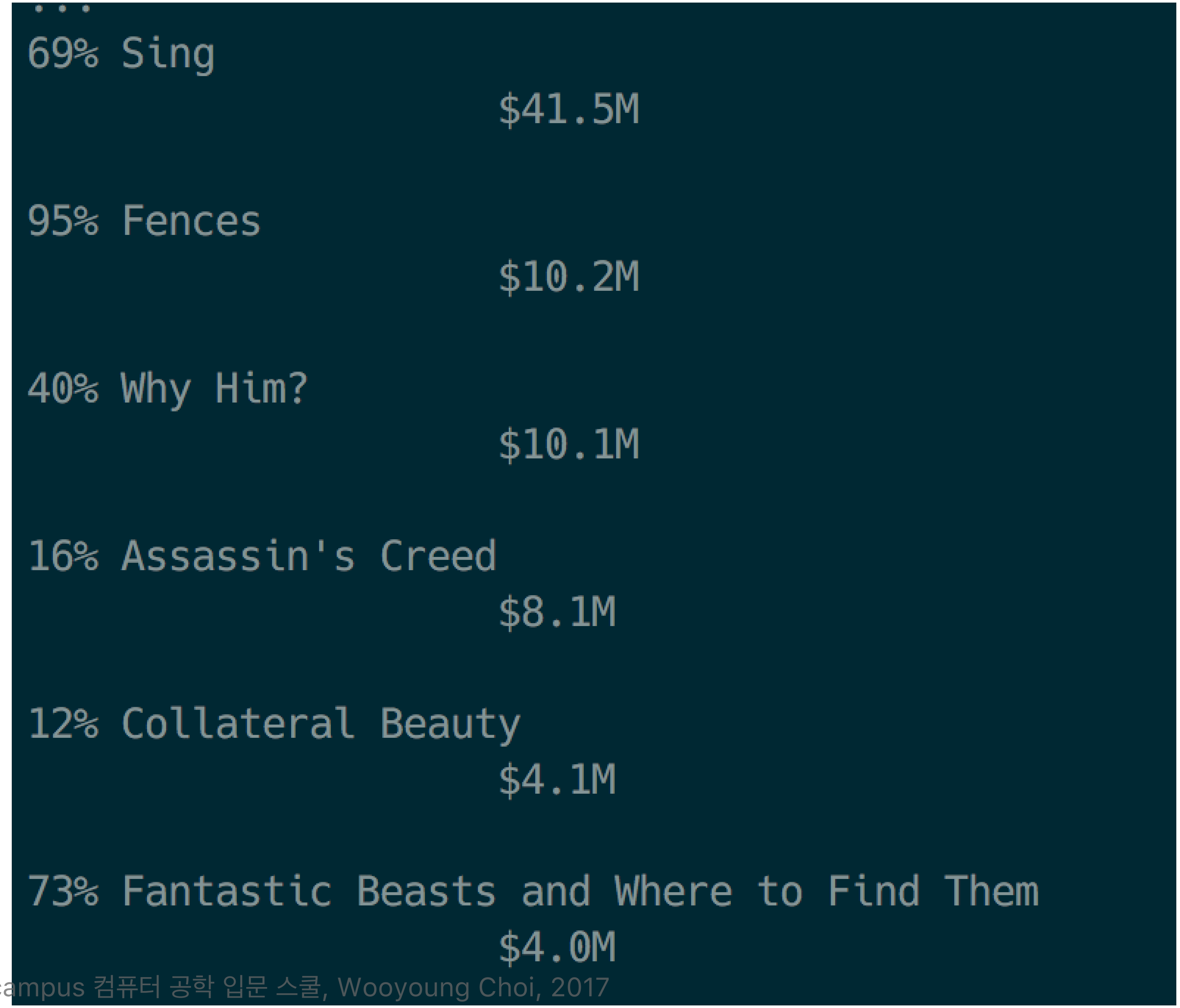
```
all_tr = table.find_all("tr")

for tr in all_tr:
     all_td = tr.find_all("td")
     score = all_td[0].find("span", attrs={"class":"tMeterScore"
     movie_name = all_td[1].a.text
     amount = all_td[2].a.text
     print(score, movie_name, amount)
```

Web Scraping with Beautiful Soup



```
>>> import urllib.request
>>> from bs4 import BeautifulSoup
>>> url = "https://www.rottentomatoes.com"
>>> html = urllib.request.urlopen(url)
>>> source = html.read()
>>> html.close()
>>> soup = BeautifulSoup(source, "html.parser")
>>> table = soup.find(id="Top-Box-Office")
>>> all_tr = table.find_all("tr")
>>> for tr in all_tr:
...     all_td = tr.find_all("td")
...     score = all_td[0].find("span", attrs={"class":"tMeterScore"}).text
...     movie_name = all_td[1].a.text
...     amount = all_td[2].a.text
...     print(score, movie_name, amount)
```

Web Scraping with Beautiful Soup



```
...
69% Sing
                    $41.5M


95% Fences
                    $10.2M


40% Why Him?
                    $10.1M


16% Assassin's Creed
                    $8.1M


12% Collateral Beauty
                    $4.1M


73% Fantastic Beasts and Where to Find Them
                    $4.0M
```

So, Let's Scrap Naver