파이썬(2) - 11주차 / 1905096(진태양)

HTTP (HyperText Transfer Protocol)

브라우저가 서버로부터 인터넷을 통해 웹 문서를 받는 경우의 규칙을 정한 것.

```
[브라우저]--(request)-->[서버]
[브라우저]<-(response)--[서버]
```

- 모든 인터넷 프로토콜 기준은 한 기관에 의해 개발 / IETF: Internet Engineering Task Force
- 기준은 `RFCs(Request for Comments)`라고 부름

프로토콜이란?

- 규칙의 모음, 모두가 따르므로 서로가 서로의 행동을 예측 가능
- 서로 충돌하지 않아야 함.
    - 미국의 이차선 도로에서는 오른쪽 도로로 달려야 함.
    - 영국의 이차선 도로에서는 왼쪽 도로로 달려야 함.

서버로부터 데이터 받기

- 사용자가 `'href=값'`을 가지고 있는 앵커 태그를 클릭해 새로운 페이지로 이동할 때 마다, 브라우저는 웹 서버와 연결을 만들고 GET 요청을 실행해 페이지 URL에 나타난 값을 수신
- 서버는 문서를 포맷팅하고 유저에게 보여주는 HTML 문서를 리턴

파이썬에서의 HTTP 요청

- `socket` 라이브러리 사용: Connection을 수립하고, 이를 이용해 통신을 수행한다.

인코딩

- 문자를 표현하는 방식
- 인코딩과 디코딩은 서로 다른 표현 방식을 서로 변환해주는 행위를 의미한다.
- Server에서는 UTF-8 방식의 인코딩을 사용하고, 프로그램 상에선 unicode 방식의 인코딩을 사용하는 경우 이를 상호 변환시켜주며 사용해야 한다.

여러 바이트로 된 문자

보다 다양한 문자를 나타내기 위해서는 더 많은 바이트를 쓸 필요가 있음

- UTF-16: 길이 고정, 2바이트
- UTF-32: 길이 고정, 4바이트
- UTF-8: 1~4 bytes
    - ASCII를 포함하며 호환
    - ASCII를 자동으로 감지 가능
    - UTF-8은 시스템 간에 데이터를 교환할 때 가장 실용적으로 추천되는 인코딩 형식

urllib 라이브러리

HTTP는 굉장히 많이 쓰이기 때문에 소켓을 다루고 웹 페이지를 불러오는 라이브러리(`urllib`)존재

```
import urllib.request, urllib.parse, urllib.error

fhand = urllib.request.urlopen('http://data.pr4e.org/romeo.txt')

counts = dict()
for line in fhand:
    words = line.decode().split()
    for word in words:
        counts[word] = counts.get(word, 0) + 1
print(counts)
```

HTML 파싱(Web Scraping)

- 프로그램이나 스크립트가 브라우저처럼 행동하며 페이지를 살펴보고 정보를 추출하고 조사하는 것을 지칭
- 검색엔진은 웹 페이지를 스크래핑함 - 이를 스파이더링 또는 크롤링이라고도 함

스크래핑을 하는 이유

- 데이터 가져오기: 특히 소셜 데이터, 누가 연결되어 있는지(SNS, 기업, 정치 등)
- 외부로 내보내는 기능이 없는 시스템에서 데이터 가져오기
- 사이트를 모니터링하며 새로운 정보 감지
- 검색엔진의 데이터베이스를 구축하기 위한 스크래핑

주의 사항

- 웹 페이지 스크래핑은 웹 페이지 내용을 마음대로 빼간다는 점에서 논란의 여지가 있음
- copyright된 정보를 다시 출판하는 것은 허용되지 않음
- 이용약관을 위배하지 않도록 유의

BeautifulSoup

- HTML 파싱을 도와주는 파이썬 라이브러리

```
import urllib.request, urllib.parse, urllib.error
from bs4 import BeautifulSoup
import ssl

# Ignore SSL certificate errors
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

url = input('Enter - ')
html = urllib.request.urlopen(url, context=ctx).read()
soup = BeautifulSoup(html, 'html.parser')

tags = soup('a')
for tag in tags:
    print(tag.get('href', None))
```