파이썬(2) - 12주차 / 1905096(진태양)

파이썬을 이용한 웹 데이터 처리

- 공공 데이터, 민간 데이터 / 데이터의 활용
- 이러한 데이터는 Local에 존재하는 것이 아닌, Web 상에 존재한다.
- 이에 이를 Access하고 Fetch 하는 일련의 과정이 요구된다. 이를 Python으로 해결할 수 있다.

웹 상의 데이터

- HTTP 요청과 응답에 대한 이해와 자원을 바탕으로, 이 프로토콜을 이용한 프로그램 간의 정보 교환의 추세
- 네트워크와 응용프로그램 간의 데이터 표현 방식에 있어서 합의가 필요
- 가장 널리 사용되는 두 가지 포맷: `XML`, `JSON`

네트워크를 통한 정보 전송

- Python, PHP, JavaScript, HashMap 등 언어에 따라 데이터 표현 방식이 상이하다.
- 서로 다른 표현 방식의 데이터를 공유하려면 포맷을 통일 시킬 필요가 있다.

> 와이어 프로토콜: 우리가 와이어 상에 보내는 것

와이어 포맷

- 두 프로그램의 데이터 표현 방식이 상이할 경우 공통 포맷을 정의해야 한다.
- 대표적으로 사용되는 것이 `XML`과 `JSON` 이다.
- [파이썬 딕셔너리] <=> [JSON/XML] <=> [자바 HashMap]

> 직렬화(Serialize)와 역직렬화(De-Serialize)를 거쳐 통신 수행

XML(eXtensible Markup Language)

개요

- 정보 시스템이 구조화된 데이터를 공유하는 것이 초기 목적
- 표준 범용 교정 용어 (SGML)의 간소화된 버전으로 시작하였고, 조금 더 인간에게 친숙한 방향으로 디자인

> http://en.wikipedia.org/wiki/XML

- 시작 태그(start tag), 끝 태그(end tag), 문자 정보(text element), 속성(attributes), 스스로 닫는 태그(self closing tag)로 이루어짐

```
<person>
    <name>Chuck</name>
    <phone type="intl">+1 734 303 4456</phone>
    <email hide="yes" />
</person>
```

XML 스키마

- XML 문서의 올바른 형식에 대한 설명
- 문서의 구조와 내용에 대한 제한의 형식으로 표현됨
- 시스템 간의 '약속'을 표현할 때 주로 사용됨 - '내 시스템은 이 스키마에 맞는 XML만 수용할 것이다.'
- 특정 XML이 스키마의 사항들을 모두 만족할 때, 우리는 그것을 '타당하다(`validate`)'라고 한다

> http://en.wikipedia.org/wiki/Xml_schema

XML 스키마 계약서 - XML 스키마 W3C(XSD)

- xs:element:
  - minOccurs: 최소 빈도
  - maxOccrurs: 최대 빈도
- xs:complexType: 여러 개의 태그로 구성되어 있다.(name="person")
- xs:sequence: 해당 태그는 순서대로 이러한 태그를 가진다.
- xs:simpleType / xs:restriction / xs:enumeration

```
<xs:complexType name="person">
    <xs:sequence>
        <xs:element name="lastname" type="xs:string"/>
        <xs:element name="age" type="xs:integer"/>
        <xs:element name="dateborn" type="xs:date"/>
    </xs:sequence>
</xs:complexType>
```

XML의 사용

```
import xml.etree.ElementTree as ET
data = '''
<person>
    <name>Chuck</name>
    <phone type="intl">+1 734 303 4456</phone>
    <email hide="yes" />
</person>
'''

tree = ET.fromstring(data)
print('Name:', tree.find('name').text) # text of the tag
print('Attr:', tree.find('email').get('hide')) # attribute value of the tag
```

JSON(JavaScript Object Notation)

데이터를 중첩된 `리스트`와 `딕셔너리`로 표현

```
import json

data = '''
{
  "name" : "Chuck",
  "phone" : {
```

```
    "type" : "intl",
    "number" : "+1 734 303 4456"
   },
   "email" : {
     "hide" : "yes"
   }
}'''

info = json.loads(data)
print('Name:', info["name"])
print('Hide:', info["email"]["hide"])
```

서비스 지향적 접근(SOA; Service Oriented Access)

- 대부분의 웹 애플리케이션은 여러 서비스를 이용
- 다른 애플리케이션의 서비스를 가져와 사용
  - 신용카드 청구
  - 호텔 예약 시스템
- 서비스는 애플리케이션이 서비스를 이용하기 위해 따라야 하는 `규칙`을 만듦 (API; Application Programming Interface)

API(Application Programming Interface)

- API는 인터페이스를 지정하고, 그 인터페이스의 행동을 제어한다는 점에서 매우 추상적
- API에 명시된 기능을 제공하는 소프트웨어를 API의 `실행`이라고 함
- API는 대체로 애플리케이션을 구성하게 되는 언어로 정의됨

> http://en.wikipedia.org/wiki/API

Geo API

- 위치 정보와 관련된 API
- 구글, 네이버, 카카오 등 지도 서비스 제공자가 제공한다.

마무리

- 데이터는 곳곳에 존재한다. 이러한 데이터를 굳이 우리가 가질 필요가 없다.
- 그들이 제공해주는 수단으로, 받아와 사용하면 충분하다.
- 그렇기에 이를 받아와 사용하는 방법을 익히는 것이 요구된다.