Язык Python. Часть 2

Лекция 5

Рекурсивность структуры

- На уровне документа работаем с тегами
- И с внетеговым содержимым
- Аналогичные операции можем делать на уровне тега
- С его подтегами
- И с его внетеговым содержимым

Поиск

- Навигация дает возможность перебрать все теги
- Проанализировать их аттрибуты
- И содержимое
- Но есть типовые задачи
- И их можно выразить более компактно

Напечатаем все ресурсы, на которые ссылаемся - через поиск

```
1 import requests
2 from bs4 import BeautifulSoup
3
4 response = requests.get('https://docs.python.org/3/library/i
5 soup = BeautifulSoup(response.text, 'html.parser')
6
7 for link in soup.find_all('a'):
8     if link.has_attr('href'):
9         print(link['href'])
```

Посчитаем все нумерованные и ненумерованные списки

```
1 import requests
2 from bs4 import BeautifulSoup
3
4 response = requests.get('https://docs.python.org/3/library/i
5 soup = BeautifulSoup(response.text, 'html.parser')
6
7 print(sum(1 for link in soup.find_all(('ol', 'ul'))))
```

Поиск заголовков

- В HTML есть специальные теги для заголовков
- h1, h2, h3, h4, h5, h6
- Можно списком задать
- А можно - регулярным выражением
- Подробнее о регулярных выражениях - позже

Кратко о регулярных выражениях

- Язык шаблонов
- Описываем шаблон
- Про любую строку можно понять, соответствует шаблону или нет
- Можно вызвать find_all с регулярным выражением
- Для заголовков это будет `h[0-6]`

Посчитаем все нумерованные и ненумерованные списки

```
 1 import re
 2 import requests
 3 from bs4 import BeautifulSoup
 4
 5 response = requests.get('https://docs.python.org/3/library/
 6 soup = BeautifulSoup(response.text, 'html.parser')
 7
 8 import re
 9 for tag in soup.find_all(re.compile("h[0-6]")):
10     print(tag)
```

Поиск по сложному критерию

- Хотим найти все теги
  - С определенным именем
  - Или с определенным атрибутом
  - Или с определенным значением заданного атрибута
  - Можно определить соответствующую функцию
  - И передать ее в find_all

Если не хочется искать рекурсивно

- Хотим пройтись по всем спискам
- И для каждого списка перебрать его элементы
- Не трогая подсписки
- У find_all есть специальный параметр - recursive

Если хочется просто текстовое содержимое

- Просто text
- Очищенный от всех тегов
- Есть специальный метод
- Называется get_text
- Именно метод, а не свойство

Ограничения requests

Ограничения requests

- Статичный HTML - это текст, снабженный тегами
- Но есть один интересный тег
- Он называется script
- В нем есть программный код
- На языке JavaScript

Ограничения requests

- При открытии страницы в браузере
- Этот код активизируется
- И может заполнять элементы HTML
- И даже создавать новые

Ограничения requests

- Иногда он даже делает http-запросы
- И использует ответы сервера для обновления частей исходного документа
- Это называется single-page design
- Часто используется в соцсетях
- Видно по тому, что содержимое меняется, а страница целиком не перегружается

Статическое и динамическое содержимое

- Эти термины уже употреблялись
- Но немного в другом смысле
- Тогда речь шла о том, как HTML документ порождается
- Сейчас - о том, насколько велика в нем роль JavaScript-элементов

Статическое и динамическое содержимое

- Без JavaScript - статический
- JavaScript есть, но используется для мелких красот - не совсем статический
- Но для наших задач - равносилен статическому
- А если элементы данных порождаются JavaScript-ом - это существенно динамический

Ограничения requests

Еще ограничения

- HTML - гибкий язык
- В некотором смысле - слишком гибкий
- Одну задачу можно решить многими способами
- Использование многих тегов не всегда соответствует имени
- Например, table, h1-h6

Еще ограничения

- Очень многое зависит от CSS
- В итоге
  - Сложно придумать общую схему анализа для группы сайтов
  - Даже одного типа - интернет, новости
  - Нужен индивидуальный подход
  - И смотреть глазами

Еще ограничения

- Организация HTML-документов зависит от их разработчиков
- Она может меняться
- И ее разработчики не обязаны перед нами отчитываться
- Скрипт, работающий сегодня, может перестать работать завтра

Еще ограничения

- И еще непросто отличить основное содержимое от служебного
- Не поможет ни requests, ни Selenium
- Либо индивидуальные эвристики
- Либо машинное обучение (задача классификации и т.п.)
- Либо другой тип данных - Rest API, json

REST API

- Когда мы пытаемся что-то извлечь из HTML
- Мы его используем не совсем по назначению
- Это язык разметки
- Но больше для целей визуализации
- А не семантической разметки

REST API

- Иногда ресурс умеет отвечать не HTML-документом
- А json-документом
- Он сильно проще для анализа в целом
- И конкретные json-документы сделаны в расчете на их анализ
- Единственная проблема - надо, чтобы ресурс хотел его отдавать

JSON

- JSON - JavaScript Object Notation
- Это не значит, что используется только в JavaScript
- Центральное понятие - объект
- Ограничивается фигурными скобками
- Простейший объект - {}

JSON

- В объекте могут быть поля
- У поля есть имя и значение
- Имена уникальны в рамках объекта
- Идейно это напоминает словарь в Python
- Пары имя-значение отделяются запятой
- Имя от значения отделяется двоеточием
- Имя и значение пишутся в кавычках

Объект с данными о пользователе

```
{
    "name": "Vasily Petrov",
    "id": 1234567,
    "birthday": "10.02.1998"
}
```

Другие типы значений

- Значением может быть число
- Последовательность цифр
- Без кавычек

```
{
   "name": "Vasily Petrov",
   "id": 1234567,
   "age": 25
}
```

Другие типы значений

- Значением может быть другой объект
- Обрамляется фигурными скобочками
- И там - тоже пары значений
- И дальше могут быть вложенные объекты

```
{
    "person": {
        "name": "Vasily Petrov",
        "id": 1234567,
        "age": 25
     },
    "photo": "https://image-storage.com/123234325.jpg
}
```

Другие типы значений

- Значением может быть список значений
- Элементы списка - строки, числа, объекты, списки
- Обрамляется квадратными скобками
- Элементы разделяются запятыми

```
{
    "person": {
        "name": "Vasily Petrov",
        "id": 1234567,
        "age": 25
     },
    "skills": ['java', 'python']
}
```