Οδηγός BeautifulSoup για Web Scraping

BeautifulSoup Web Scraping Guide

Εισαγωγή στο DOM και τη Δομή Δέντρου / Introduction to DOM and Tree Structure

Το DOM (Document Object Model) είναι μια προγραμματιστική διεπαφή για έγγραφα HTML/XML. Αναπαριστά το έγγραφο ως ένα δέντρο όπου κάθε στοιχείο είναι ένας κόμβος.

The DOM (Document Object Model) is a programming interface for HTML/XML documents. It represents the document as a tree where each element is a node.

Παράδειγμα δομής δέντρου / Tree structure example:

```
html
├── head
│   ├── title
│   └── meta
└── body
    ├── div (class="header")
    │   └── h1
    ├── div (class="content")
    │   ├── p
    │   └── a
    └── div (class="footer")
```

Εγκατάσταση και Εισαγωγή / Installation and Import

```python
# Εγκατάσταση / Installation
# pip install beautifulsoup4

from bs4 import BeautifulSoup
import requests
```

Βασική Χρήση / Basic Usage

```python
# Δημιουργία BeautifulSoup object από HTML / Creating BeautifulSoup object from HTML
html_doc = """
<html>
    <body>
        <div class="content">
            <h1>Τίτλος / Title</h1>
            <p class="text">Κείμενο / Text</p>
        </div>
    </body>
</html>
"""
soup = BeautifulSoup(html_doc, 'html.parser')
```

Μέθοδοι Εύρεσης / Finding Methods

find() - Εύρεση Πρώτου Στοιχείου / Finding First Element

```
# Εύρεση πρώτου στοιχείου p / Find first p element
first_p = soup.find('p')
print(first_p.text)  # Εμφανίζει το κείμενο / Shows the text

# Εύρεση με κλάση / Find with class
content_div = soup.find('div', class_='content')

# Εύρεση με πολλαπλά κριτήρια / Find with multiple criteria
element = soup.find('div', {
    'class': 'content',
    'id': 'main'
})
```

find_all() - Εύρεση Όλων των Στοιχείων / Finding All Elements

```
# Εύρεση όλων των παραγράφων / Find all paragraphs
all_paragraphs = soup.find_all('p')
for p in all_paragraphs:
    print(p.text)

# Εύρεση με λίστα ετικετών / Find with list of tags
elements = soup.find_all(['h1', 'h2', 'h3'])

# Εύρεση με όριο / Find with limit
first_three = soup.find_all('p', limit=3)
```

CSS Selectors

```
# Βασικοί επιλογείς / Basic selectors
elements = soup.select('div.content')  # Κλάση / Class
element = soup.select_one('#main')     # ID
links = soup.select('a[href]')         # Ιδιότητα / Attribute

# Σύνθετοι επιλογείς / Complex selectors
nested = soup.select('div.content > p')  # Άμεσο παιδί / Direct child
descendants = soup.select('div p')       # Όλοι οι απόγονοι / All
descendants
siblings = soup.select('h1 ~ p')         # Αδέρφια / Siblings
```

Χειρισμός Ιδιοτήτων / Handling Attributes

```
# Πρόσβαση σε ιδιότητες / Accessing attributes
element = soup.find('a')
link = element['href']        # Απευθείας πρόσβαση / Direct access
link = element.get('href')    # Ασφαλής πρόσβαση / Safe access
classes = element.get('class', [])  # Με προεπιλογή / With default

# Έλεγχος ύπαρξης ιδιοτήτων / Checking attribute existence
has_class = element.has_attr('class')

# Πρόσβαση σε όλες τις ιδιότητες / Accessing all attributes
attrs = element.attrs  # Επιστρέφει λεξικό / Returns dictionary
```

Πλοήγηση στο Δέντρο / Tree Navigation

```
element = soup.find('p')

# Πρόσβαση σε γονείς / Accessing parents
parent = element.parent
parents = element.parents  # Όλοι οι πρόγονοι / All ancestors

# Πρόσβαση σε παιδιά / Accessing children
children = element.contents  # Λίστα παιδιών / List of children
for child in element.children:
    print(child)

# Πρόσβαση σε αδέρφια / Accessing siblings
next_sib = element.next_sibling
prev_sib = element.previous_sibling
```

Πρακτικό Παράδειγμα / Practical Example

```
# Scraping ενός blog post / Scraping a blog post
html = """
<article class="post">
    <h1 class="title">Τίτλος Blog / Blog Title</h1>
    <div class="meta">
        <span class="author">Συγγραφέας / Author</span>
        <span class="date">2024-01-01</span>
    </div>
    <div class="content">
        <p>Παράγραφος 1 / Paragraph 1</p>
        <p>Παράγραφος 2 / Paragraph 2</p>
    </div>
    <div class="tags">
        <a href="/tag1">Python</a>
        <a href="/tag2">WebScraping</a>
    </div>
</article>
"""

soup = BeautifulSoup(html, 'html.parser')

# Εξαγωγή δεδομένων / Extracting data
title = soup.select_one('.title').text
author = soup.select_one('.author').text
date = soup.select_one('.date').text
paragraphs = [p.text for p in soup.select('.content p')]
tags = [a.text for a in soup.select('.tags a')]

# Εκτύπωση αποτελεσμάτων / Printing results
print(f"Τίτλος / Title: {title}")
print(f"Συγγραφέας / Author: {author}")
print(f"Ημερομηνία / Date: {date}")
print(f"Παράγραφοι / Paragraphs: {paragraphs}")
print(f"Ετικέτες / Tags: {tags}")
```

Καλές Πρακτικές / Best Practices

1. Χρησιμοποιείτε CSS selectors για πολύπλοκες αναζητήσεις / Use CSS selectors for complex searches
2. Προτιμήστε `select_one()` αντί για `find()` για καλύτερη αναγνωσιμότητα / Prefer `select_one()` over `find()` for better readability

3. Χρησιμοποιείτε το `get()` για ασφαλή πρόσβαση σε ιδιότητες / Use `get()` for safe attribute access
4. Ελέγχετε πάντα αν υπάρχουν τα στοιχεία πριν την πρόσβαση / Always check if elements exist before accessing

In [ ]: