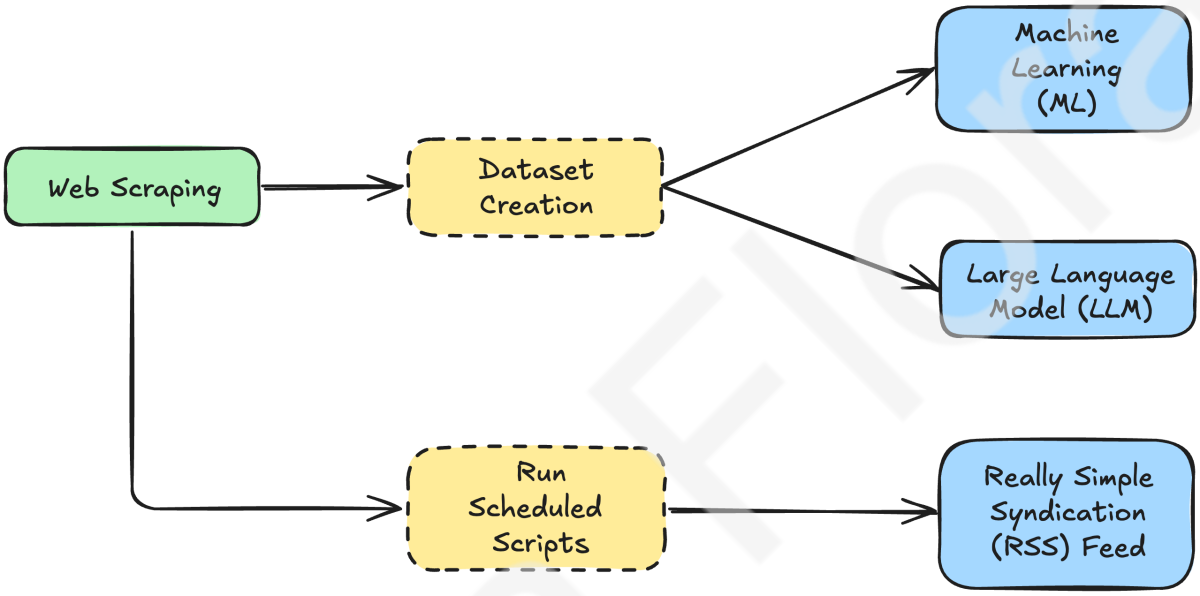
Web Scraping

Η BeautifulSoup είναι μια βιβλιοθήκη που χρησιμοποιείται ευρέως για την εξαγωγή πληροφοριών από HTML.

Είναι ιδανική για web scraping, δηλαδή την άντληση πληροφοριών από ιστοσελίδες, καθώς διευκολύνει τον εντοπισμό συγκεκριμένων στοιχείων (tags) που θέλουμε να αντλήσουμε, όπως τίτλους, παραγράφους, υπερσυνδέσμους, λίστες, πίνακες, κλπ.



Σύντομη περιγραφή βασικών όρων της HTML

Όρος	Περιγραφή
Tag	Ένα στοιχείο όπως ένας τίτλος, μια παράγραφος, ένας υπερσύνδεσμος, μια εικόνα κλπ.
Attribute	Πληροφορίες μέσα σε κλάση, IDs, σύνδεσμοι (href), αρχεία εικόνων
Element	tag + text + attributes
Container	Στοιχείο στο οποίο περιέχονται άλλα στοιχεία

Παραδείγματα από συνήθη tags

```
1. Βάσει του τύπου:
<h1> ... <h6> Τίτλοι (h1 -> η μεγαλύτερη γραμματοσειρά)
<p> Παράγραφος
<a> Υπερσύνδεσμος (text + href)
<li> Λίστα
<img> Εικόνα
```

```
<table> Πίνακας

2. Βάσει της μορφοποίησης του κειμένου (κυρίως για scraping από PDF):
<b> / <strong> Bold text
<i> Italic text
<u> Underlined text
```

Παραδείγματα από συνήθη attributes

```
href:
<a href="https://bme.uniwa.gr">Τμήμα Μηχανικών Βιοϊατρικής</a>
src, width, height:
<img src="image.jpg" width="1000" height="1000">
```

Βασική χρήση της BeautifulSoup

Κατέβασμα βιβλιοθηκών

```
# Με pip:
pip install beautifulsoup4
pip install requests

# Με anaconda:
conda install beautifulsoup4
conda install requests
```

Εισαγωγή των βιβλιοθηκών

```
from bs4 import BeautifulSoup # για εύκολη εξαγωγή δεδομένων από του HTML
import requests # για αίτημα πρόσβασης σε ιστοσελίδα και εξαγωγή όλου του
HTML
```

Πρόσβαση και εξαγωγή δεδομένων από ιστοσελίδα

```
url = "https://bme.uniwa.gr/"

data = requests.get(url) # απάντηση διακομιστή (όλο το HTML της σελίδας)
html = data.text # μετατροπή του HTML σε string

# μετατροπή του HTML σε αντικείμενο της BeautifulSoup:
soup = BeautifulSoup(html, 'html.parser')
```

Εισαγωγή σε εντολές της BeautifulSoup

1. Εξαγωγή δεδομένων

Προκειμένου να εξάγουμε δεδομένα από διάφορα containers χρησιμοποιούμε την κλάση ή το tag με τη βοήθεια κάποιων βασικών εντολών. Οι δύο βασικές εντολές που χρησιμοποιούμε είναι η εντολή `select()` και η `find_all()`.

Εντολές `select()` και `find_all()`

Οι δύο αυτές εντολές είναι παρόμοιες. Και οι δύο επιστρέφουν όλα τα στοιχεία που είτε:

1. ανήκουν σε ένα συγκεκριμένο tag
2. περιέχονται σε μια συγκεκριμένη κλάση
3. ή και τα δύο

Σε κάθε περίπτωση, εάν δε βρεθούν στοιχεία στο σημείο που ψάχνουμε, θα επιστραφεί κενή λίστα.

Επιλογή στοιχείων μόνο από το tag

```
paragraphs = soup.find_all("p")
articles = soup.find_all("a")
lists = soup.find_all("li")

paragraphs = soup.select("p")
articles = soup.select("a")
lists = soup.select("li")
```

Επιλογή στοιχείων μόνο από την κλάση

Η εντολή `select()` αποτελεί νέα προσθήκη της BeautifulSoup4 και το βασικό πλεονέκτημα της σε σχέση με την `find_all()` είναι η δυνατότητα αναζήτησης σε περισσότερες από μια κλάσεις στις οποίες βρίσκεται ταυτόχρονα το στοιχείο που ψάχνουμε (για να κάνει πιο ειδική την αναζήτηση).

```
# Στοιχεία που ανήκουν σε 2 κλάσεις ταυτόχρονα:
containers = soup.select(".class1.class2") # δε γίνεται με τη find_all()

# Στοιχεία που ανήκουν σε μια από τις 2 κλάσεις:
containers = soup.select(".class1, .class2") # χωρίζονται με κόμμα
# Παραλλαγή με find_all():
containers = soup.find_all(class_=["class1", "class2"])
```

⚠ **Προσοχή!**

Όπως βλέπουμε στο παράδειγμα, η κλάση του HTML στην BeautifulSoup αναγνωρίζεται μόνο ως `class_` καθώς η λέξη `class` είναι δεσμευμένη.

Επιλογή στοιχείων από συγκεκριμένο tag και κλάση

Η λειτουργία των εντολών στην προκειμένη περίπτωση είναι η ίδια. Αλλάζει μόνο η σύνταξη.

```
containers = soup.find("tag", class_="class1")

containers = soup.select("tag.class1")
```

Εντολές `select_one()` και `find()`

Η εντολές `find()` όπως και η `select_one()` συντάσσονται όπως και οι προηγούμενες 2 εντολές, απλά επιστρέφουν μόνο το πρώτο στοιχείο του container στο οποίο ψάχνουμε.

```
# Επιστρέφει όλα τα στοιχεία από αυτήν την κλάση:
containers = soup.select('.latest_post_title')

# Επιστρέφει μόνο το πρώτο στοιχείο της κλάσης:
container = soup.select_one('.latest_post_title')
```

2. Εξαγωγή κειμένου από το HTML

Για αυτόν τον σκοπό μπορούμε να χρησιμοποιήσουμε 2 τρόπους. Συνήθως χρησιμοποιείται η `get_text()`

```
# 1ος τρόπος
element_txt = element.text

# 2ος τρόπος
element_txt = element.get_text(strip=True)

# 3ος τρόπος - ίδιο αποτέλεσμα με το 2ο
element_txt = element.get_text().strip() # γενική εντολή της python
```

Βασική διαφορά

Η `get_text()` με `strip=True` μας δίνει τη δυνατότητα να "αγνοήσουμε" επιπλέον κενά ή αλλαγές γραμμής που μπορεί να υπάρχουν στην HTML στην αρχή ή στο τέλος του string, αλλά όχι ενδιάμεσα από άλλους χαρακτήρες.

3. Επιλογή attribute από tag

Εντολή `get()`

```
# Εξαγωγή href από υπερσύνδεσμο (article)
link = soup.find('a')
href = link.get('href')

# Output: https://bme.uniwa.gr

# Εξαγωγή attribute "src" από εικόνα
image = soup.find('img')
src = image.get('src')

# Output: image.jpg
```

4. Φιλτράρισμα των δεδομένων

Για τον σκοπό αυτό μπορούμε να χρησιμοποιήσουμε γενικότερα εντολές της Python για αναζήτηση λέξεων ή άλλων χαρακτηριστικών σε string. Το φιλτράρισμα είναι πολύ σημαντικό για τη δημιουργία ενός καλού dataset.

```
phrase1 = "Magnetic Resonance Imaging"
phrase2 = "Positron Emission Tomography"
phrases = [phrase1, phrase2]

filtered_list1 = []

for phrase in phrases:
    if "emission" not in phrase.lower(): # αναζήτηση λέξης σε string
        filtered_list1.append(phrase)

# Output: ['Magnetic Resonance Imaging']

filtered_list2 = []

for phrase in phrases:
    if not phrase.lower().startswith('magnetic'): # αν ξεκινάει από μια
λέξη
        filtered_list2.append(phrase)

# Output: ['Positron Emission Tomography']
```

Αντίστοιχα, υπάρχει και η εντολή `endswith()` και πολλές άλλες εντολές και βιβλιοθήκες της Python που επικεντρώνονται στο φιλτράρισμα ή την απλοποίηση των string, π.χ. Textblob.

5. Αποθήκευση των δεδομένων σε αρχεία

Σε ένα πρότζεκτ το πιο πιθανό είναι ότι θα αποθηκεύουμε τα δεδομένα μας σε ένα ή περισσότερα αρχεία, π.χ. JSON, CSV.

Μορφή αρχείου JSON

```
"bme": {
    "grammateia": "",
    "mathimaton": "",
    "eksetaseon": ""
    }
```

Αποθήκευση αρχείου JSON

```
import json

filename = "data.json"

# Διάβασμα του αρχείου
with open(filename, 'r') as f:
    data = json.load(f)

new_link =
"https://bme.uniwa.gr/announcements/undergraduate/announcement1/"

# Ανανέωση του αρχείου
if new_link:
    data['bme']['grammateia'] = new_link
    with open(filename, 'w') as f:
        json.dump(data, f, indent=4)
```

Μορφή αρχείου CSV

```
Question, Answer

What is Magnetic Resonance Angiography (MRA)?, Angiography using MRI

What are the indications for brain MRV?, "Tumour of the cerebral venous
sinus, drowsiness and confusion accompanying a headache"
```

Παράδειγμα αποθήκευσης σε αρχείο CSV

Το παρακάτω απόσπασμα κώδικα αποτελεί ένα σύντομο παράδειγμα, παρόμοιο με τον κώδικα που έχουμε γράψει για το πρότζεκτ **MRI Assistant** της ομάδας του **Biomed**.

Ουσιαστικά δείχνει πως μπορούμε εύκολα να εξάγουμε δεδομένα από πολλούς διαφορετικούς συνδέσμους της ίδιας ιστοσελίδας και να περάσουμε όλα τα δεδομένα με τη μορφή ερώτησης-απάντησης σε ένα αρχείο CSV.

```
brain_links = links[0:24] # εξαγωγή συνδέσμων από μια ανατομική περιοχή
final_data_list = []  # λίστα στην οποία θα αποθηκευτούν όλα τα δεδομένα

for url in brain_links:
    delay = random.uniform(3, 7) # τυχαίο delay μεταξύ αιτημάτων
    print(f"{delay:.2f} seconds before scraping {url}")
    time.sleep(delay)

    data = requests.get(url)
    html = data.text

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

    data_list = [] # απάντηση (π.χ. παράγραφοι)
    heading_list = []  # τίτλος (που θα μετατραπεί σε μορφή ερώτησης)

    containers = soup.select('.elementor-widget-container')

    for container in containers:
        heading = container.select_one('.elementor-heading-title')
        list_items = container.select('.elementor-icon-list-text')
        paragraphs = container.select('p')

        if heading:
            current_title = heading.get_text(strip=True)
            heading_list.append(current_title)

        for paragraph in paragraphs:
            paragraph_text = paragraph.get_text(strip=True)
            data_list.append({"Question": f"What is {current_title}?",
"Answer": paragraph_text})

    # προσθήκη δεδομένων του κάθε συνδέσμου στη συνολική λίστα
    final_data_list += data_list

csv_file = "brain_data.csv" # αποθήκευση δεδομένων στο CSV
with open(csv_file, mode='w', newline='', encoding='utf-8') as file:
    writer = csv.DictWriter(file, fieldnames=["Question", "Answer"])
    writer.writeheader()
    writer.writerows(final_data_list)
```

😀

Πρότζεκτ του παραρτήματος που χρησιμοποιούν web scraping

MRI Assistant

Το **MRI Assistant** αποτελεί πρότζεκτ της ομάδας του **Biomed** και αφορά τη δημιουργία ενός **chatbot το οποίο δίνει απαντήσεις σχετικά με τα πρωτόκολλα, τις ενδείξεις και αντενδείξεις και την προετοιμασία ασθενών για την εκτέλεση εξετάσεων μαγνητικής τομογραφίας (MRI)**, διευκολύνοντας την εκπαίδευση των ενδιαφερόμενων.

Το dataset του συγκεκριμένου πρότζεκτ έχει δημιουργηθεί μέσω web scraping από έμπιστες πηγές που παρέχουν πληροφορίες για MRI. Έπειτα ακολούθησε η βελτιστοποίηση ενός προεκπαιδευμένου μοντέλου LLM από τη Llama ώστε το μοντέλο να απαντάει τις "σχετικές" ερωτήσεις σύμφωνα με το dataset.

UniWA Notifier

Το **UniWA Notifier** αποτελεί πρότζεκτ της ομάδας του **Web Development**, με σκοπό τη δημιουργία ενός discord bot το οποίο **εξασφαλίζει την έγκαιρη ενημέρωση των φοιτητών για ανακοινώσεις των τμημάτων μηχανικών και του γεύματος της σίτισης του ΠΑΔΑ.**

Το discord bot τρέχει κάθε 1 ώρα κάνοντας scrape τις νέες ανακοινώσεις και προωθώντας τις σε server του παραρτήματος. Κάντε κλικ για να επισκεφτείτε τον server μας!

> ⚡ **Προσοχή!**
>
> Αν σκέφτεσαι να ασχοληθείς με το web scraping, έχε κατά νου τα παρακάτω:
>
> 1. Να χρησιμοποιείς delays μερικών δευτερολέπτων ανάμεσα στα αιτήματα που στέλνεις με την `time.sleep()` για να μην φορτώνεις τον server.
> 2. Έλεγξε τους όρους και προϋποθέσεις της σελίδας. Βεβαιώσου ότι δεν παραβιάζεις τα πνευματικά δικαιώματα του κατόχου της σελίδας.
> 3. Επικοινώνησε με τον κάτοχο της σελίδας σε περίπτωση που σε ενδιαφέρει η εκπαιδευτική ή εμπορική χρήση των δεδομένων και πάρε άδεια υπό κάποιους όρους.
> 4. Δείξε σεβασμό προς τον δημιουργό, τον κόπο και τον χρόνο που αφιέρωσε για να μαζέψει τα δεδομένα που αντλείς.

😀

😊 Καλό scraping!