WYŻSZA SZKOŁA
INFORMATYKI i ZARZĄDZANIA
z siedzibą w Rzeszowie

Rzeszów, 2025

ZARZĄDZANIE DANYMI

LABORATORIUM nr 1

Temat: Wprowadzenie do formatu ***JSON*** (ang. ***J**ava**S**cript **O**bject **N**otation*) z zastosowaniem języka programowania ***Python***

Laboratorium obejmuje m.in.:

1. omówienie podstawowej składni formatu ***JSON***, w tym porównanie do formatu ***XML***,
2. podstawowych funkcji do obsługi ***JSON***'a z wykorzystaniem języka ***Python***, oraz
3. wykonanie prostej aplikacji ***tezaurusa (słownika) do języka angielskiego***, w tym wykorzystującego dane zapisane w formacie ***JSON***.

Samodzielne wykonanie zadań z laboratorium będzie wymagane z zastosowaniem:

- dowolnego edytora ***IDE*** – np. ***Microsoft Visual Studio Code***, edytora ***Atom***, itp. (dla plików ****.py***),

oraz

- środowiska ***Jupyter Notebook / LAB*** (dla plików ****.ipynb***) lub (ewentualnie) ***Google COLAB***.

Wprowadzenie

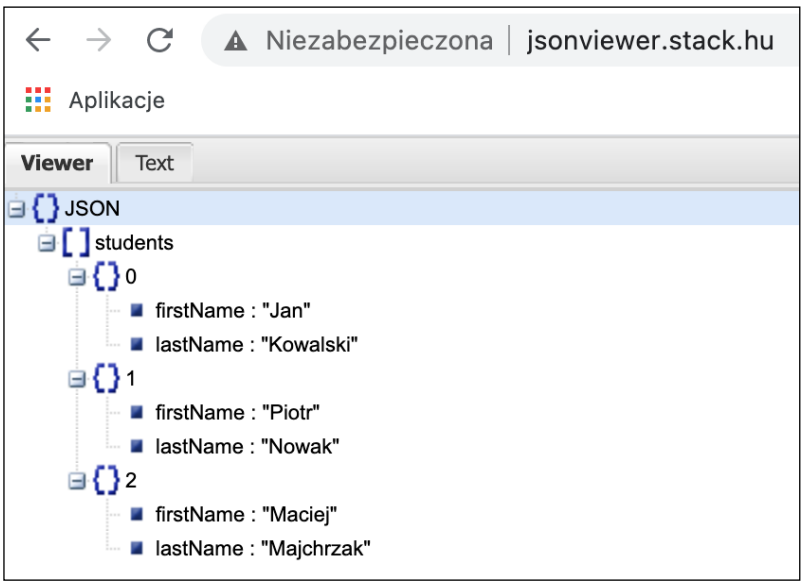
JSON (ang., ***JavaScript Object Notation***) to format przechowywania oraz wymiany informacji tekstowych, przy czym:

- zapis w ***JSON*** jest bardzo podobny do języka ***XML*** (ang., ***eXtensible Markup Language***). Jednak ***JSON*** jest lżejszy niż ***XML*** oraz szybszy (i łatwiejszy) do analizy, oraz
- ***JSON*** jest niezależny od języka oraz samo-opisujący się.

Ponadto, format ***JSON*** jest syntaktycznie identyczny z kodem do tworzenia obiektów ***JavaScript***. Z powodu tego podobieństwa, zamiast wykorzystywać parser, programy ***JavaScript*** mogą korzystać z wbudowanej funkcji ***eval()*** do przetworzenia danych w formacie ***JSON*** i utworzenia obiektów ***JavaScript***.

```
{
"students":[
    {"firstName":"Jan", "lastName":"Kowalski"},
    {"firstName":"Piotr", "lastName":"Nowak"},
    {"firstName":"Maciej", "lastName":"Majchrzak"}
]}
```

W celu zrozumienia struktury obiektów wklej powyższy kod na stronie http://jsonviewer.stack.hu/
Wynik powinien być podobny do zamieszczonego poniżej:



Rys.1. *Przykładowy wynik dla* ***JSON***

Postać danych w formacie XML

```
<students>
    <student>
        <firstName>Jan</firstName> <lastName>Kowalski</lastName>
    </student>
    <student>
```

```
        <firstName>Piotr</firstName> <lastName>Nowak</lastName>
    </student>
    <student>
        <firstName>Maciej</firstName> <lastName>Majchrzak</lastName>
    </student>
</students>
```

Podobieństwa do języka XML:

- czysty tekst
- samo-opisujący
- hierarchiczny
- może być przetwarzany przez ***JavaScript***
- może być przesyłany z wykorzystaniem ***AJAX***'a

Różnice względem języka XML:

- nie ma tagów zamykających,
- krótszy,
- szybszy w odczycie i zapisie,
- może być parsowany z wykorzystaniem funkcji ***JavaScript eval()***,
- wykorzystuje tablice,
- brak zarezerwowanych słów.

Składnia języka JSON:

Składnia języka ***JSON*** jest podzbiorem składni opisującej obiekty w języku ***JavaScript***:

- dane są w postaci pary ***klucz/wartość,***
- dane oddzielane są przecinkiem,
- nawiasy klamrowe przechowują obiekty,
- w nawiasach prostokątnych przechowywane są tablice.

Przykład danych:

```
"firstName" : "Łukasz"
```

oznacza:

```
firstName = "John"
```

Wartością może być:

- liczba,
- tekst,
- wartość logiczna (***true*** oraz/lub ***false***),
- tablica,
- obiekt w nawiasach klamrowych,
- wartość ***null***.

Przykład odwołania do obiektu JavaScript

```
var students = [
                  { "firstName":"Jan" , "lastName":"Kowalski" },
                  { "firstName":"Piotr" , "lastName":"Nowak" },
                  { "firstName":"Maciej" , "lastName":"Majchrzak" }
                ]
students[0].lastName; pobierze wartość Kowalski
```

Przykładowy wygląd (funkcjonalności) aplikacji słownika

(do wykonania na laboratorium)

```
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Zawartość pliku json = {'imię': ['Lukasz'], 'nazwisko': 'Piatek', 'stanowiska': ['Wykładowca', 'Programista AI/IT']}
```

Rys.1. Odczyt oraz wyświetlenie zawartości pliku ***JSON***

```
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: imię
['Lukasz']
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: stanowiska
['Wykładowca', 'Programista AI/IT']
```

Rys.2. Zwrot *wartości dla podanego słowa (klucza) z pliku* ***JSON***

```
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: jakis_wyraz
Wyraz = jakis_wyraz nie iestnieje w pliku. Sprawdź ponownie.
```

Rys.3. *Obsługa dla słów (kluczy) nieistniejących w danych* ***JSON***

```
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: StAnOwIsKa
['Wykładowca', 'Programista AI/IT']
```

Rys.4. Obsługa wielkości liter (mapowanie wyrazów-kluczy do małych liter)

```
>>> SequenceMatcher(None, "imię", "imi").ratio()
0.8571428571428571
```

Rys.5. Współczynnik podobieństwa pomiędzy słowami/wyrazami (podstawa)

```
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: nazwis
Czy chciałeś podać nazwisko?
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: nazwisko
Piatek
```

Rys.6. Podpowiedź/rekomendowanie najlepszego dopasowania (klucza do pliku ***JSON***)

```
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: nazwi
Czy chciałeś podać nazwisko? Podaj T lub N: T
Piatek
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$
```

Rys.7. Podpowiedź (potwierdzenie) od użytkownika (odpowiedź Tak)

```
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: nazw
Czy chciałeś podać nazwisko? Podaj T lub N: N
Wyraz = nazw nie iestnieje w pliku. Sprawdź ponownie.
```

Rys.7B. Podpowiedź (potwierdzenie) od użytkownika (odpowiedź Nie)

```
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: stanowiska
Wykładowca
Programista AI/IT
```

Rys.8. Optymalizacja kodu (1) – tablica w formie kolejnych pozycji

```
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: Delhi
The largest metropolis by area and the second-largest metropolis by population in India.
```

Rys.9. Optymalizacja kodu (2) – sprawdzanie nazw własnych (rozpoczynających się z dużej litery)

```
(base) MacBook-Pro-3:ZarzadzanieDanymi lukaszpiatek$ python3 lab_1.py
Podaj wyraz: USA
A country and federal republic in North America located north of Mexico and south of Canada, including Alaska, Hawaii and overseas territories.
```

Rys.10. Optymalizacja kodu (3) – sprawdzanie akronimów (np. USA)