

Lab 3.6.6 – Parsing Type Data di Python

Tujuan :

- Part 1 : Jalankan DEVASC VM
- Part 2 : Parsing XML di Python
- Part 3 : Parsing JSON di Python
- Part 4 : Parsing YAML di Python

Kebutuhan Resource :

- PC/Laptop
- Virtual Box/VMWare
- DEVASC Virtual Machine

Langkah-langkah :

Part 1 : Jalankan DEVASC VM

Part 2 : Parsing XML di Python

Langkah 1 : Jalankan script untuk parse data XML

- a. Open file **parsexml.py** yang ada di directory **~/labs/devnet-src/parsing**
- b. Import module **ElementTree** , yang digunakan untuk memarsing.

```
import xml.etree.ElementTree as ET
import re
```
- c. Kemudian, gunakan function **parse** dari ET (**ElementTree**) untuk memarsing file **myfile.xml**.
Selanjutnya, gunakan fungsi **parse** dari ET (**ElementTree**) untuk mengurai file **.xml** dan menetapkannya ke variabel (**xml**). Kemudian, dapatkan elemen root dengan fungsi **getroot** dan tetapkan ke variabel (**root**).

```
xml = ET.parse("myfile.xml")
root = xml.getroot()
```
- d. Sekarang tingkat teratas dari tree dapat dicari untuk tag yang mengandung **<edit-config>**, dan ketika ditemukan, itu blok yang diberi tag dapat dicari untuk dua nilai bernama yang dikandungnya: **<default-operation>** dan **<test-option>**.

Buat ekspresi reguler untuk mendapatkan konten konten root XML di tag **<rpc>** lalu tambahkan ekspresi reguler tambahan untuk menelusuri konten untuk menemukan nilai dari element **<edit-config>**, **<default-operation>**, dan **<test-option>**.

```
ns= re.match('{.*}', root.tag).group(0)
editconf = root.find("{}edit-config".format(ns))
```

```
defop = editconf.find("{}default-operation".format(ns))
```

```
testop = editconf.find("{}test-option".format(ns))
```

- e. Tambahkan statement untuk print nilai element **<default-operation>** dan **<test-option>** .

```
print("The default-operation contains: {}".format(defop.text))
```

```
print("The test-option contains:{}".format(testop.text))
```

Langkah 2 :Run script.

Simpan dan run file **parsexml.py**, dan perhatikan outputnya seperti dibawah ini :

```
devasc@labvm:~/labs/devnet-src/parsing$ python3 parsexml.py
```

```
The default-operation contains: merge
```

```
The test-option contains:set
```

```
devasc@labvm:~/labs/devnet-src/parsing$
```

Part 3 : Parsing JSON di Python

Parsing JavaScript Object Notation (JSON) adalah persyaratan yang sering digunakan untuk berinteraksi dengan REST API. Langkah-langkahnya biasanya sebagai berikut:

- 1) Otentikasi menggunakan kombinasi user / password untuk mengambil token yang akan kedaluwarsa setelah jumlah yang ditentukan waktu. Token ini digunakan untuk mengautentikasi permintaan berikutnya.
- 2) Jalankan GET request ke REST API, mengautentikasi sesuai kebutuhan, untuk mengambil status sebuah resource , merequest JSON sebagai format output.
- 3) Ubah JSON yang dikembalikan, sesuai kebutuhan.
- 4) Jalankan POST (atau PUT) ke REST API yang sama (sekali lagi, autentikasi sesuai kebutuhan) untuk mengubah status resource, sekali lagi meminta JSON sebagai format keluaran dan menafsirkannya sesuai kebutuhan untuk menentukan apakah operasi berhasil.

Contoh script untuk melakukan parsing adalah :

```
{
"access_token":"ZDI3MGEyYzQtNmFlNS00NDNhLWF1NzAtZGVjNjE0MGU1OGZmZWNmZD
EwN2ItY
TU3",
"expires_in":1209600,
"refresh_token":"MDEyMzQ1Njc4OTAxMjMONTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4O
TEyMzQ1
Njc4",
```

```
"refreshtokenexpires_in":7776000
}
```

Langkah 1 : Buat script untuk parsing data JSON

- Open file **parsejson.py** yang ada di dalam directory **~/labs/devnet-src/parsing**

- Import library json dan yaml

```
import json
import yaml
```

- Gunakan statement Python **with** untuk membuka **myfile.json** dan buat nama variabel **json_file**. Kemudian gunakan method **json.load** untuk memuat file JSON ke dalam string yang diset ke nama variabel **ourjson**.

```
with open('myfile.json','r') as json_file:
    ourjson = json.load(json_file)
```

- Tambahkan statement print.

```
print(ourjson)
```

Langkah 2 : run script dan print data JSON

- Simpan dan Jalankan script

```
devasc@labvm:~/labs/devnet-src/parsing$ python3 parsejson.py
{'access_token':
'ZDI3MGeyYzQtNmFlNS00NDNhLWF1NzAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
',
    'expires_in': 1209600,
    'refresh_token':
'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
', 'refreshtokenexpires_in': 7776000}
devasc@labvm:~/labs/devnet-src/parsing$
```

- Tambahkan statement untuk menampilkan nilai token dan berapa detik masaexpired token

```
print("The access token is : {}".format(ourjson['access_token']))
print("The token expires in {} seconds.".format(ourjson['expires_in']))
```

- Simpan dan run script, dan perhatikan outputnya .

```
devasc@labvm:~/labs/devnet-src/parsing$ python3 parsejson.py
{'access_token':
'ZDI3MGeyYzQtNmFlNS00NDNhLWF1NzAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
',
    'expires_in': 1209600,
    'refresh_token':
'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
', 'refreshtokenexpires_in': 7776000}
```

```
The          access          token          is          :
ZDI3MGEyYzQtNmFlNS00NDNhLWFlnZAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
The token expires in 1209600 seconds.
devasc@labvm:~/labs/devnet-src/parsing$
```

Langkah 3 : output data Json dalam format yaml

- a. Tambahkan statement print dalam script, dan akan digunakan method dump().

```
print("\n\n---")
print(yaml.dump(ourjson))
```

- b. Save dan run

```
devasc@labvm:~/labs/devnet-src/parsing$ python3 parsejson.py
{'access_token':
'ZDI3MGEyYzQtNmFlNS00NDNhLWFlnZAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
',
      'expires_in':      1209600,
      'refresh_token':
'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
', 'refreshtokenexpires_in': 7776000}
The          access          token          is          :
ZDI3MGEyYzQtNmFlNS00NDNhLWFlnZAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
The token expires in 1209600 seconds.
```

```
---
access_token:
ZDI3MGEyYzQtNmFlNS00NDNhLWFlnZAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
expires_in: 1209600
refresh_token:
MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
refreshtokenexpires_in: 7776000
```

```
devasc@labvm:~/labs/devnet-src/parsing$
```

Part 4 : Parse YAML di Python

Langkah 1 : Build script untuk parsing data YAML

- a. Open file **parseyaml.py** yang ada di **directory : ~/labs/devnet-src/parsing**
- b. Import library json dan yaml

```
import json
```

```
import yaml
```

- c. Gunakan statement untuk membuka myfile.yaml dan men set variable yaml_file. Kemudian gunakan method **yaml.safe_load** untuk mengambil file yaml ke dalam tipe data string yang disimpan dalam variable ouryaml.

```
with open('myfile.yaml','r') as yaml_file:
    ouryaml = yaml.safe_load(yaml_file)
```

- d. Tambahkan statement print untuk melihat output dari ouryaml.

```
print(ouryaml)
```

Langkah 2 : Run script untuk menampilkan data YAML dan modifikasi datanya

- a. Save dan run scriptnya

```
devasc@labvm:~/labs/devnet-src/parsing$ python3 parseyaml.py
{'access_token':
'ZDI3MGEyYzQtNmFlNS00NDNhLWFlnZAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
',
    'expires_in': 1209600,
    'refresh_token':
'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
', 'refreshtokenexpires_in': 7776000}
devasc@labvm:~/labs/devnet-src/parsing$
```

- b. Tambahkan statement print untuk mendisplay token dan waktu expired

```
print("The access token is {}".format(ouryaml['access_token']))
print      ("The      token      expires      in      {}
seconds.".format(ouryaml['expires_in']))
```

- c. Save dan run

```
devasc@labvm:~/labs/devnet-src/parsing$ python3 parseyaml.py
{'access_token':
'ZDI3MGEyYzQtNmFlNS00NDNhLWFlnZAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
',
    'expires_in': 1209600,
    'refresh_token':
'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
', 'refreshtokenexpires_in': 7776000}
The      access      token      is
ZDI3MGEyYzQtNmFlNS00NDNhLWFlnZAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
The token expires in 1209600 seconds.
devasc@labvm:~/labs/devnet-src/parsing$
```

Langkah 3: Output parse data YAML dalam format data JSON

- a. Tambahkan statement print dan gunakan method **dumps()**.

```
print ("\n\n")
print(json.dumps(ouryaml, indent=4))
```

b. Save dan run

```
devasc@labvm:~/labs/devnet-src/parsing$ python3 parseyaml.py
{'access_token':
'ZDI3MGEyYzQtNmFlNS00NDNhLWF1NzAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
',
    'expires_in': 1209600,
    'refresh_token':
'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
', 'refreshtokenexpires_in': 7776000}
The          access          token          is
ZDI3MGEyYzQtNmFlNS00NDNhLWF1NzAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
The token expires in 1209600 seconds.
```

```
{
    "access_token":
"ZDI3MGEyYzQtNmFlNS00NDNhLWF1NzAtZGVjNjE0MGU1OGZmZWNmZDEwN2ItYTU3
",
    "expires_in": 1209600,
    "refresh_token":
"MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
",
    "refreshtokenexpires_in": 7776000
}
devasc@labvm:~/labs/devnet-src/parsing$
```