

Laboratorio 3b

Analizar diferentes tipos de datos con Python

Sergio Sebastian Pezo Jimenez - 20224087G

Parte 1: Inicializamos la VM de DEVASC.

Parte 2: Analicemos un XML

Primero ejecutamos nuestro código inicial

The screenshot shows a code editor with two files: `myfile.xml` and `parsexml.py`. The `myfile.xml` file contains an XML document with a root element `<rpc message-id="1">` and a child element `<edit-config>`. The `parsexml.py` file contains Python code that uses the `xml.etree.ElementTree` module to parse the XML file and extract specific values. A terminal window at the bottom shows the command `python3 parsexml.py` being executed, and the output of the program is displayed.

```
myfile.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <rpc message-id="1">
3   xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
4     <edit-config>
5       <target>
6         <candidate/>
7       </target>
8       <default-operation>merge</default-operation>
9       <test-option>set</test-option>
10      <config>
11        <int8.1>
12          xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
13          nc:operation="create"
14          xmlns="http://netconfcentral.org/ns/test">9</int8.1>
15        </config>
16      </edit-config>
17    </rpc>
18  </xml>
```

```
parsexml.py
1 # Fill in this file with the code from parsing XML exercise
2 import xml.etree.ElementTree as ET
3 import re
4
5 xml = ET.parse("myfile.xml")
6 root = xml.getroot()
7
8 ns = re.match('(.*)', root.tag).group(0)
9 editconf = root.find("{}edit-config".format(ns))
10 defop = editconf.find("{}default-operation".format(ns))
11 testop = editconf.find("{}test-option".format(ns))
12
13 print("The default-operation contains: {}".format(defop.text))
14 print("The test-option contains: {}".format(testop.text))
```

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

Parte 3: Analicemos un JSON

Procedamos a parsear nuestro JSON para obtener las llaves correspondientes para imprimirlas

The screenshot shows a VS Code editor with three panels. The left panel displays a JSON file named `myfile.json` with the following content:

```
1 {
2   "access_token": "ZDI3MGEyYzQtNmFlNS00NDNhLWFLNzAtZGVjNjE0MGU1OGZmZWlnZDEwN2ItYTU3",
3   "expires_in": 1209600,
4   "refresh_token": "MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4OTEyMzQ1Njc4",
5   "refreshtokenexpires_in": 7776000
6 }
```

The right panel shows a Python script named `parsejson.py` with the following code:

```
1 # Fill in this file with the code from parsing JSON exercise
2 import json
3 import yaml
4
5 with open('myfile.json', 'r') as json_file:
6     ourjson = json.load(json_file)
7
8 print(ourjson)
9
10 print("The access token is: {}".format(ourjson['access_token']))
11 print("The token expires in {} seconds.".format(ourjson['expires_in']))
```

The bottom panel is a terminal window showing the execution of the script:

```
devasc@labvm: ~/labs/devnet-src/parsing
File Edit View Search Terminal Help
The default-operation contains: merge
The test-option contains: set
devasc@labvm:~/labs/devnet-src/parsing$ python3 parsejson.py
{'access_token': 'ZDI3MGEyYzQtNmFlNS00NDNhLWFLNzAtZGVjNjE0MGU1OGZmZWlnZDEwN2ItYTU3',
 'expires_in': 1209600, 'refresh_token': 'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4OTEyMzQ1Njc4', 'refreshtokenexpires_in': 7776000}
devasc@labvm:~/labs/devnet-src/parsing$ python3 parsejson.py
{'access_token': 'ZDI3MGEyYzQtNmFlNS00NDNhLWFLNzAtZGVjNjE0MGU1OGZmZWlnZDEwN2ItYTU3',
 'expires_in': 1209600, 'refresh_token': 'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4OTEyMzQ1Njc4', 'refreshtokenexpires_in': 7776000}
The access token is: ZDI3MGEyYzQtNmFlNS00NDNhLWFLNzAtZGVjNjE0MGU1OGZmZWlnZDEwN2ItYTU3
The token expires in 1209600 seconds.
```

Agregamos un par de líneas para formatear nuestro JSON a YAML.

The screenshot shows the same VS Code editor setup as before, but with additional code in the `parsejson.py` script and the terminal output.

The `parsejson.py` script now includes the following lines:

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

The terminal output now shows the JSON data formatted as YAML:

```
devasc@labvm:~/labs/devnet-src/parsing
File Edit View Search Terminal Help
The default-operation contains: merge
The test-option contains: set
devasc@labvm:~/labs/devnet-src/parsing$ python3 parsejson.py
{'access_token': 'ZDI3MGEyYzQtNmFlNS00NDNhLWFLNzAtZGVjNjE0MGU1OGZmZWlnZDEwN2ItYTU3',
 'expires_in': 1209600, 'refresh_token': 'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4OTEyMzQ1Njc4', 'refreshtokenexpires_in': 7776000}
devasc@labvm:~/labs/devnet-src/parsing$ python3 parsejson.py
{'access_token': 'ZDI3MGEyYzQtNmFlNS00NDNhLWFLNzAtZGVjNjE0MGU1OGZmZWlnZDEwN2ItYTU3',
 'expires_in': 1209600, 'refresh_token': 'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4OTEyMzQ1Njc4', 'refreshtokenexpires_in': 7776000}
The access token is: ZDI3MGEyYzQtNmFlNS00NDNhLWFLNzAtZGVjNjE0MGU1OGZmZWlnZDEwN2ItYTU3
The token expires in 1209600 seconds.

---
access_token: ZDI3MGEyYzQtNmFlNS00NDNhLWFLNzAtZGVjNjE0MGU1OGZmZWlnZDEwN2ItYTU3
expires_in: 1209600
refresh_token: MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4OTEyMzQ1Njc4
refreshtokenexpires_in: 7776000
```

Parte 4: Analicemos un YAML

Procedamos a parsear nuestro YAML.

```
! myfile.yaml x
! myfile.yaml
1 ---
2 access_token: ZDI3MGEyYzQtNmFlnS00NDNhLWFlNzAtZGVjNjE0MGU1OGZnZWlnZDEwN2ItYTU3
3 expires_in: 1209600
4 refresh_token: MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
5 refreshtokenexpires_in: 7776000

devasc@labvm: ~/labs/devnet-src/parsing
File Edit View Search Terminal Help
Ext ---
access_token: ZDI3MGEyYzQtNmFlnS00NDNhLWFlNzAtZGVjNjE0MGU1OGZnZWlnZDEwN2ItYTU3
expires_in: 1209600
refresh_token: MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
refreshtokenexpires_in: 7776000

devasc@labvm:~/labs/devnet-src/parsing$ python3 parseyaml.py
{'access_token': 'ZDI3MGEyYzQtNmFlnS00NDNhLWFlNzAtZGVjNjE0MGU1OGZnZWlnZDEwN2ItYTU3',
 'expires_in': 1209600, 'refresh_token': 'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4',
 'refreshtokenexpires_in': 7776000}
The access token is ZDI3MGEyYzQtNmFlnS00NDNhLWFlNzAtZGVjNjE0MGU1OGZnZWlnZDEwN2ItYTU3
The token expires in 1209600 seconds.
devasc@labvm:~/labs/devnet-src/parsing$
```

```
parseyaml.py x
parseyaml.py > ...
1 # Fill in this file with the code from parsing YAML exercise
2 import json
3 import yaml
4
5 with open('myfile.yaml','r') as yaml_file:
6     ouryaml = yaml.safe_load(yaml_file)
7
8 print(ouryaml)
9
10 print("The access token is {}".format(ouryaml['access_token']
11 print("The token expires in {} seconds.".format(ouryaml['exp...
```

De igual manera parseemos nuestro contenido de YAML a formato JSON.

```
! myfile.yaml x
! myfile.yaml
1 ---
2 access_token: ZDI3MGEyYzQtNmFlnS00NDNhLWFlNzAtZGVjNjE0MGU1OGZnZWlnZDEwN2ItYTU3
3 expires_in: 1209600
4 refresh_token: MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
5 refreshtokenexpires_in: 7776000
6

devasc@labvm:~/labs/devnet-src/parsing
File Edit View Search Terminal Help
Ext ---
access_token: ZDI3MGEyYzQtNmFlnS00NDNhLWFlNzAtZGVjNjE0MGU1OGZnZWlnZDEwN2ItYTU3
expires_in: 1209600
refresh_token: MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
refreshtokenexpires_in: 7776000

devasc@labvm:~/labs/devnet-src/parsing$ python3 parseyaml.py
{'access_token': 'ZDI3MGEyYzQtNmFlnS00NDNhLWFlNzAtZGVjNjE0MGU1OGZnZWlnZDEwN2ItYTU3', 'exp
pires_in': 1209600, 'refresh_token': 'MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Nj
c4OTEyMzQ1Njc4', 'refreshtokenexpires_in': 7776000}
The access token is ZDI3MGEyYzQtNmFlnS00NDNhLWFlNzAtZGVjNjE0MGU1OGZnZWlnZDEwN2ItYTU3
The token expires in 1209600 seconds.

{
  "access_token": "ZDI3MGEyYzQtNmFlnS00NDNhLWFlNzAtZGVjNjE0MGU1OGZnZWlnZDEwN2ItYTU3",
  "expires_in": 1209600,
  "refresh_token": "MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4",
  "refreshtokenexpires_in": 7776000
}
devasc@labvm:~/labs/devnet-src/parsing$
```

```
parseyaml.py x
parseyaml.py > ...
1 # Fill in this file with the code from parsing YAML exercise
2 import json
3 import yaml
4
5 with open('myfile.yaml','r') as yaml_file:
6     ouryaml = yaml.safe_load(yaml_file)
7
8 print(ouryaml)
9
10 print("The access token is {}".format(ouryaml['access_token']
11 print("The token expires in {} seconds.".format(ouryaml['exp
12
13 print("\n\n")
14 print(json.dumps(ouryaml, indent=4))
```