(/)

Curriculum
SE Foundations ∧
Average: **123.57%** ∨

You have a captain's log due before 2024-04-28 (in 1 day)! Log it now!
(/captain_logs/5624270/edit)

0x10. Python - Network #0

Bash   Python   Scripting   Back-end   API

⚙ Weight: 1

📅 Project over - took place from Mar 27, 2024 4:00 AM to Mar 28, 2024 4:00 AM

☑ An auto review will be launched at the deadline

In a nutshell…

- **Auto QA review:** 0.0/47 mandatory & 0.0/23 optional
- **Altogether: 0.0%**
    - Mandatory: 0.0%
    - Optional: 0.0%
    - Calculation: 0.0% + (0.0% * 0.0%) == **0.0%**

Resources

Read or watch:

- HTTP (HyperText Transfer Protocol) (/rltoken/rAon_EpQ6PGl8N0plySn4A) (*except: "TRACE" Request Method, "CONNECT" Request Method, Language Negotiation and "Options MultiView" and Character Set Negotiation*)
- HTTP Cookies (/rltoken/MhVCl_0oviQIdWPn5oX-NQ)

Learning Objectives

At the end of this project, you are expected to be able to explain to anyone (/rltoken/6HRdeOrrKTW2ih43ObB8tQ), **without the help of Google**:

General

- What a URL is
- What HTTP is
- How to read a URL
- The scheme for a HTTP URL
- What a domain name is
- What a sub-domain is
- How to define a port number in a URL
- What a query string is
- What an HTTP request is
- What an HTTP response is
- What HTTP headers are
- What the HTTP message body is
- What an HTTP request method is
- What an HTTP response status code is
- What an HTTP Cookie is
- How to make a request with cURL
- What happens when you type google.com in your browser (Application level)

Copyright - Plagiarism

- You are tasked to come up with solutions for the tasks below yourself to meet with the above learning objectives.
- You will not be able to meet the objectives of this or any following project by copying and pasting someone else's work.
- You are not allowed to publish any content of this project.
- Any form of plagiarism is strictly forbidden and will result in removal from the program.

Requirements

General

- Allowed editors: `vi`, `vim`, `emacs`
- - A `README.md` file, at the root of the folder of the project, is mandatory
- All your scripts will be tested on Ubuntu 20.04 LTS
- All your Bash scripts should be exactly 3 lines long (`wc -l file` should print 3)
- All your files should end with a new line
- All your files must be executable
- The first line of all your bash files should be exactly `#!/bin/bash`
- The second line of all your Bash scripts should be a comment explaining what is the script doing
- All `curl` commands must have the option `-s` (silent mode)
- All your files will be interpreted/compiled on Ubuntu 20.04 LTS using `python3` (version 3.8.5)
- The first line of all your Python files should be exactly `#!/usr/bin/python3`
- Your code should use the pycodestyle (version `2.8.*`)
- All your modules should be documented: `python3 -c 'print(__import__("my_module").__doc__)'`
- All your classes should be documented: `python3 -c 'print(__import__("my_module").MyClass.__doc__)'`
- All your functions (inside and outside a class) should be documented: `python3 -c 'print(__import__("my_module").my_function.__doc__)'` and `python3 -c`

```
'print(__import__("my_module").MyClass.my_function.__doc__)'
```

(/). A documentation is not a simple word, it's a real sentence explaining what's the purpose of the
~~module, class or method (the length of it will be verified)~~

Quiz questions

> **Great!** You've completed the quiz successfully! Keep going! (Show quiz)

Tasks

0. cURL body size   mandatory

> Score: 0.0% (*Checks completed: 0.0%*)

Write a Bash script that takes in a URL, sends a request to that URL, and displays the size of the body of
the response

- The size must be displayed in bytes
- You have to use `curl`

Please test your script in the sandbox provided, using the web server running on port 5000

```
guillaume@ubuntu:~/0x10$ ./0-body_size.sh 0.0.0.0:5000
10
guillaume@ubuntu:~/0x10$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x10-python-network_0`
- File: `0-body_size.sh`

☐ Done?   Check your code   Ask for a new correction   >_ Get a sandbox   QA Review

1. cURL to the end   mandatory

🔍

> Score: 0.0% (*Checks completed: 0.0%*)

Write a Bash script that takes in a URL, sends a `GET` request to the URL, and displays the body of the
response

(/)
- Display only body of a `200` status code response
- You have to use `curl`

Please test your script in the sandbox provided, using the web server running on port 5000

```
guillaume@ubuntu:~/0x10$ ./1-body.sh 0.0.0.0:5000/route_1 ; echo ""
Route 2
guillaume@ubuntu:~/0x10$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x10-python-network_0`
- File: `1-body.sh`

☐ Done?   Check your code   Ask for a new correction   >_ Get a sandbox   QA Review

2. cURL Method   `mandatory`

Score: 0.0% (*Checks completed: 0.0%*)

Write a Bash script that sends a `DELETE` request to the URL passed as the first argument and displays the body of the response

- You have to use `curl`

Please test your script in the sandbox provided, using the web server running on port 5000

```
guillaume@ubuntu:~/0x10$ ./2-delete.sh 0.0.0.0:5000/route_3 ; echo ""
I'm a DELETE request
guillaume@ubuntu:~/0x10$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x10-python-network_0`
- File: `2-delete.sh`

☐ Done?   Check your code   Ask for a new correction   >_ Get a sandbox   QA Review

3. cURL only methods   `mandatory`

Score: 0.0% (*Checks completed: 0.0%*)

Write a Bash script that takes in a URL and displays all HTTP methods the server will accept.

- You have to use `curl`

(/)

Please test your script in the sandbox provided, using the web server running on port 5000

```
guillaume@ubuntu:~/0x10$ ./3-methods.sh 0.0.0.0:5000/route_4
OPTIONS, HEAD, PUT
guillaume@ubuntu:~/0x10$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x10-python-network_0`
- File: `3-methods.sh`

☐ Done?   Check your code   Ask for a new correction   >_ Get a sandbox   QA Review

4. cURL headers   mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a Bash script that takes in a URL as an argument, sends a `GET` request to the URL, and displays the body of the response

- A header variable `X-School-User-Id` must be sent with the value `98`
- You have to use `curl`

Please test your script in the sandbox provided, using the web server running on port 5000

```
guillaume@ubuntu:~/0x10$ ./4-header.sh 0.0.0.0:5000/route_5 ; echo ""
Hello School!
guillaume@ubuntu:~/0x10$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x10-python-network_0`
- File: `4-header.sh`

☐ Done?   Check your code   Ask for a new correction   >_ Get a sandbox   QA Review

5. cURL POST parameters   mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a Bash script that takes in a URL, sends a `POST` request to the passed URL, and displays the body
of the response

- A variable `email` must be sent with the value `test@gmail.com`
- A variable `subject` must be sent with the value `I will always be here for PLD`
- You have to use `curl`

Please test your script in the sandbox provided, using the web server running on port 5000

```
guillaume@ubuntu:~/0x10$ ./5-post_params.sh 0.0.0.0:5000/route_6 ; echo ""
POST params:
    email: test@gmail.com
    subject: I will always be here for PLD
guillaume@ubuntu:~/0x10$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x10-python-network_0`
- File: `5-post_params.sh`

[ ] Done?   Check your code   Ask for a new correction   >_ Get a sandbox   QA Review

6. Find a peak   `mandatory`

Score: 0.0% (*Checks completed: 0.0%*)

Technical interview preparation:

- You are not allowed to google anything
- Whiteboard first

Write a function that finds **a peak** in a list of unsorted integers.

- Prototype: `def find_peak(list_of_integers):`
- You are not allowed to import any module
- Your algorithm must have the lowest complexity (*hint: you don't need to go through all numbers to find a peak*)
- `6-peak.py` must contain the function
- `6-peak.txt` must contain the complexity of your algorithm: `O(log(n))`, `O(n)`, `O(nlog(n))` or `O(n2)`
- **Note**: there may be more than one peak in the list

```
guillaume@ubuntu:~/0x10$ cat 6-main.py
(/)
#!/usr/bin/python3
""" Test function find_peak """
find_peak = __import__('6-peak').find_peak

print(find_peak([1, 2, 4, 6, 3]))
print(find_peak([4, 2, 1, 2, 3, 1]))
print(find_peak([2, 2, 2]))
print(find_peak([]))
print(find_peak([-2, -4, 2, 1]))
print(find_peak([4, 2, 1, 2, 2, 2, 3, 1]))

guillaume@ubuntu:~/0x10$ ./6-main.py
6
3
2
None
2
4
guillaume@ubuntu:~/0x10$ wc -l 6-peak.txt
2 6-peak.txt
guillaume@ubuntu:~/0x10$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x10-python-network_0`
- File: `6-peak.py, 6-peak.txt`

[ ] Done? | Check your code | Ask for a new correction | >_ Get a sandbox | QA Review

7. Only status code   #advanced

Score: 0.0% (*Checks completed: 0.0%*)

Write a Bash script that sends a request to a URL passed as an argument, and displays only the status code of the response.

- You are not allowed to use any pipe, redirection, etc.
- You are not allowed to use `;` and `&&`
- You have to use `curl`

Please test your script in the sandbox provided, using the web server running on port 5000

```
guillaume@ubuntu:~/0x10$ ./100-status_code.sh 0.0.0.0:5000 ; echo ""
200
guillaume@ubuntu:~/0x10$
guillaume@ubuntu:~/0x10$ ./100-status_code.sh 0.0.0.0:5000/nop ; echo ""
404
guillaume@ubuntu:~/0x10$
```

Repo:
(7)

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x10-python-network_0`
- File: `100-status_code.sh`

☐ Done?   Check your code   Ask for a new correction   >_ Get a sandbox   QA Review

8. cURL a JSON file   #advanced

Score: 0.0% (*Checks completed: 0.0%*)

Write a Bash script that sends a JSON `POST` request to a URL passed as the first argument, and displays the body of the response.

- Your script must send a `POST` request with the contents of a file, passed with the filename as the second argument of the script, in the body of the request
- You have to use `curl`

Please test your scripts in the sandbox provided, using the web server running on port 5000

```
guillaume@ubuntu:~/0x10$ cat my_json_0
{
    "name": "John Doe",
    "age": 33
}
guillaume@ubuntu:~/0x10$ ./101-post_json.sh 0.0.0.0:5000/route_json my_json_0 ; e
cho ""
Valid JSON
guillaume@ubuntu:~/0x10$
guillaume@ubuntu:~/0x10$ cat my_json_1
I'm a JSON! really!
guillaume@ubuntu:~/0x10$ ./101-post_json.sh 0.0.0.0:5000/route_json my_json_1 ; e
cho ""
Not a valid JSON
guillaume@ubuntu:~/0x10$
guillaume@ubuntu:~/0x10$ cat my_json_2
{
    "name": "John Doe",
    "age": 33,
}
guillaume@ubuntu:~/0x10$ ./101-post_json.sh 0.0.0.0:5000/route_json my_json_2 ; e
cho ""
Not a valid JSON
guillaume@ubuntu:~/0x10$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x10-python-network_0`

- File: `101-post_json.sh`

(/)

[ ] Done?   Check your code   Ask for a new correction   >_ Get a sandbox   QA Review

9. Catch me if you can!   #advanced

Score: 0.0% (*Checks completed: 0.0%*)

Write a Bash script that makes a request to `0.0.0.0:5000/catch_me` that causes the server to respond with a message containing `You got me!`, in the body of the response.

- You have to use `curl`
- You are not allow to use `echo`, `cat`, etc. to display the final result

Please test your script in the sandbox provided, using the web server running on port 5000

```
guillaume@ubuntu:~/0x10$ ./102-catch_me.sh ; echo ""
You got me!
guillaume@ubuntu:~/0x10$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x10-python-network_0`
- File: `102-catch_me.sh`

[ ] Done?   Check your code   Ask for a new correction   >_ Get a sandbox   QA Review