Lucie Michelet ELSS

IN423 - PW 02 – HTTP part2

1. HTTP

In the first part we will enhance the http server with two objectives:

- be able to distinguish different requests made by the client,
- send different type of data (various encoding) to the client.
- Send error message when necessary

Preliminary analysis:

1. What kind of software are used to make HTTP requests?

We are using a web browser to make HTTP request.

Open the minimal dedicated HTTP server from last practical work (or the solution provided by your teacher) and execute the script .

2. What should be written on the browser url field to make a request to the server?

http://127.0.0.1:55000/ which is the TCP IP : TCP PORT

Now use your favourite Internet browser to send a request on the server.

We want to analyze “HTTP request” sent by the Internet Browser (i.e. firefox, chrome etc..).

Note on HTTP requests:
An HTTP request is composed of a first line containing the HTTP method (GET POST etc) , requested resource and protocol version, followed by multiple lines of options set by the requester (i.e. the client or internet Browser : firefox, chrome, edge etc.).

Example of an HTTP request:

```
GET /index.html HTTP/1.1
Host: 127.0.0.1:5005
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.7; rv:46.0)
Gecko/20100101 Firefox/46.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: fr,fr-FR;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
DNT: 1
Connection: keep-alive
```

3. On this HTTP request, what is the HTTP method used?

The method used is the GET method.

4. On this HTTP request, which resource is requested?

Based on the first line of the HTTP request message, the requested resource is "/index.html". The client is requesting the server to provide the resource "index.html" located at the root directory of the server.

5. On this HTTP request, what means HTTP/1.1 ?

HTTP/1.1" refers to the version of the HTTP protocol that is being used.

6. What is the data encoding used in this protocol for requests messages?

The data encoding accepted is gzip and deflate.

7. What is/are the character(s) used between lines of the request? Give the code in hexadecimal? What is/are the escape character(s) used in programming languages?

HTTP RFC: https://tools.ietf.org/html/rfc2068#section-2.2 (see p. 14-15)
ASCCI table: http://www.columbia.edu/kermit/ascii.html

In HTTP, the characters used between lines of the request are carriage return (CR) and line feed (LF) characters, also known as CRLF.
The hexadecimal code for CR is 0D, and the hexadecimal code for LF is 0A.
In programming language, the escape character are “\r\n”

8. In the first line of the request, what is the character used to separate the three fields?

The character used to separate the three fields in the first line of the request is a space.

Implementation:

We want to send appropriate resource to the client depending on the resource requested.
We need to look at first line of the request to find the “http method” (GET, POST etc,) and related resource (an html page, an image etc.).

For example, if the first line contains:

```
1- GET / HTTP/1.1
```

or

```
2- GET /index.html HTTP/1.1
3- GET /status.html HTTP/1.1
4- GET /control.html HTTP/1.1
```

You should
1, 2 -> give the default index.html page
3 -> give server status : date, machine name and user name (cf. TP1 annex)
4 -> give a control form (see the control.html in annexe

0 – The Main loop:

The Http protocol is a connectionless protocol which mean that each time a client want to send a request he has to – connect to server – send the request – receive the response – then the server close the communication socket.
To make another request, the client needs to connect again and so one.

If this is not already done, adapt your script to make your main loop include accept, receive and send and close of the communication socket with the client.

1 - Analysing the request:

1.1 read the request on the socket and decode it with 'ascii' code (HTTP uses ascii)
1.2 now that your request is a big character string we are going to separate each lines using the split function and using \n (new line) as the separator:

request_list = request.split('\n').

We now obtain a list and want to focus on the first line.
1.3 split the first line using the space (' ') as the separator. You should obtain a list with on the first slot the string describing the method (i.e. "GET" or "POST" etc.). On the second slot you'll have the requested data.

2 – Response:

2.1 – If the request method is not a GET, send the "Method not allowed" response:

```
http_head = "HTTP/1.1 405 Method not allowed \r\n\r\n"
http_data = "html><body><p>405 cette méthode n'est pas prise en charge
</p></body></html>"
http_405error = http_head.encode("ascii") + http_data.encode("utf8")
```

2.2 – If the request is a GET, test if the file exists (`import os` then `os.path.isfile("filename")` in the www folder of the web site (provided by your teacher), otherwise send a "HTTP 404 not found" error message :

```
http_head = "HTTP/1.1 404 Not found\r\n\r\n"
http_data = "<html><body><p>404 not found</p></body></html>\r\n"
http_404error = http_head.encode("ascii") + http_data.encode("utf8")
```

2.3 – If the file exists, you need to find mime type of the file depending on its extension.
For example: .html -> text/html
.jpg -> image/jpeg
.ico -> image/x-icon
etc.
use the appropriate mime type to indicate the Content-type of your data in the http header of your response..

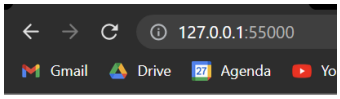
2.4 – Now read the file in binary mode (see annexe) and put the data after the HTTP 200 OK response (an empty line must separate the http header and the data.)

Example of HTTP 200 OK response:

```
reply = "HTTP/1.1 200 OK\r\n
reply += Date: Tuesday, 3 Mar 2016 16:48:13 GMT\r\n
reply += Expires: -1\r\n
reply += Cache-Control: private, max-age=0\r\n
reply += Content-Type: text/html;\r\n
reply += charset=ISO-8859-1\r\n\r\n
reply += <html><body><h1>In43 is the best course</h1>
</body></html>\r\n"
```

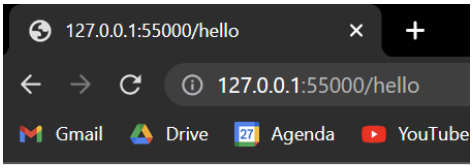
Use appropriate date and content type in your message and if necessary add charset information (let all other options like in the example) . (For help on date see Practical work 1 annexe).

3 – Test your code with your internet browser with the url : 127.0.0.1:50080/index.html and 127.0.0.1:50080/ etc.

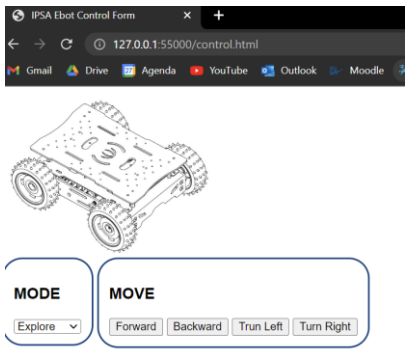


IPSA EBOT

control
status



404 not found



Sniffing:

Wireshark is a popular tool to listen the networks you are connected in. It enables you to see and analyze network messages on your different network interfaces.

Got to the wireshark software web page, download and install wireshark for your system.
https://www.wireshark.org/
https://www.wireshark.org/#download

It is a good example of software using its own notation and concepts that you should be able to learn by your own.

/!\ IMPORTANT /! : for this part you 'll have to generate some data transfer over the network for example by using a web browser to make request to your micro-HTTP server. Anyway, if the web browser and the micro-HTTP server are on the same machine, the OS will use the 127.0.0.1 interface (loopback) instead of the network interface where your server is bound. You have two solution :

- use another device on the wifi network to force the use of the wifi interface,
- or bind your server to 127.0.0.1 and use wireshark to sniff the 127.0.0.1 (loopback) interface.

Search the web or read the doc to find how to make those different filters and write them in this document.

9. Give the wireshark "display filter" to filter messages coming from your IP address

ip.**src** == 10.9.127.207 for the LAN or ip.src == 127.0.0.1 for the PAN

10. filter messages going to your IP address

ip.**dst** == 10.9.127.207

11. filter messages going to your micro-HTTP server port number

(ip.**dst** == 127.0.0.1) && (tcp.dstport==55000)

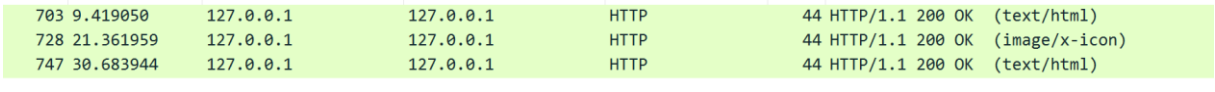
12. filter messages coming from your micro-HTTP server port number

(ip.**src** == 127.0.0.1) && (tcp.dstport==55000)

13. filter the HTTP response code 200 OK messages

http.response.code == 200

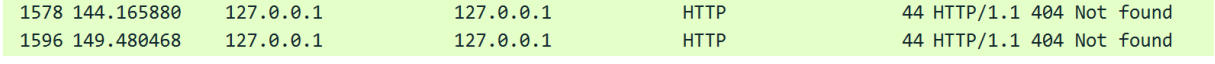
14. Give a Wireshark screenshot of you captured HTTP 200 message here



15. Give the display filter used to filter the HTTP response code 404 not found messages

http.response.code == 404

16. Give a Wireshark screenshot of you captured HTTP 404 message here



2. ANNEX – python strings

Objet type str (class str)

https://docs.python.org/3/library/stdtypes.html#textseq

Example: S = ‘'BOUUU"

Search:

S.count(str) → count the occurrence number of str in S
S.find(str) → give the index of the first occurrence of str
S.rfind(str) → give the index of the last occurrence str

Cut / past:

S.split(str) → cut the characters according to str separator return a list.
L.join(list) → concatenate characters that are contained in a list L

To split a string in python you can use the split() method, for example :

```
a = "hello you"
b = a.split(" ")
print(b)
```

will give you a list in the b variable, so print(b) will display ["hello", "you"], you can also use the first or the second element of b with b[0] or b[1].

You can make a for loop on a list like that :

```
for v in b:
    print(v)
```

for each iteration, the v variable will have one value of b.

3. ANNEX – Files in python

Open()

https://docs.python.org/3/library/functions.html#open

The function **open()** opens a file to read or write.

If the file cannot be opened, an OSError is raised

Prototype : F = open(file, mode='r', encoding=None)

file : The files path, ex: "L:\IN14\TP05\test.txt'’ "../test.txt"

mode : opening mode (read /write / both)

'r' : open for reading (default)
'w': open for writing, truncating the file first
'x' : open for exclusive creation, failing if the file already exists
'a' : open for writing, appending to the end of the file if it exists
'b' : binary mode
’t' : text mode (default)
'+' : open a disk file for updating (reading and writing)
'U' : universal newlines mode (deprecated)

'r+' ou 'w+' open for read and write

encoding : such as : ascii, iso-Latin-9, utf8 etc.

you can find the list of encoding types here:
https://docs.python.org/3/library/codecs.html#standard-encodings

Other used methods:

F.close() → close a file

F.readline() → read one line (str) (and move the reading head)

F.readlines() → list of lines (and move the reading head)

F.write(str) → write str (and move the reading head)

F.writelines(list)→ write a list of char (and move the reading head)

F.seek(int) → move the reading head by int

F.tell() → (int) position of the reading head

Examples of read/write :

```
# Read
fname = "demo.txt"
Fichier = open(fname,'rb')      # open for read 'r' as binary 'b'
data = Fichier.read()
Fichier.close()
```

```
# Write (adding)
fout = "out.txt"
Fout = open(fout,'a')
Fout.write(" - pi : " + str(math.pi)+"\n")
Fout.close()
```