Technical University of Moldova

Faculty of Computers, Informatics and Microelectronics

Department of Software Engineering and Automation

Web Programming

Laboratory work #2

# HTTP, caching and content negotiation

*Author:*
Popa Eugeniu
std. gr. FAF-202

*Supervisor:*
Alexei Șerșun

Chișinău 2023

# 1 Task

- Write a command line program, using go2web executable as a starting point.
- The program should implement at least the following CLI:

```
go2web -u <URL> # make an HTTP request to the specified URL and print the response
go2web -s <search-term> # make an HTTP request to search the term using your favorite search engine and print top 10 results
go2web -h # show this help
```

- The responses from request should be human-readable (e.g. no HTML tags in the output).

# 2 Results

- I have written the command line program in Python. [1].
- The program implements the following CLI:

```
go2web -u <URL>
go2web -s <search-term>
go2web -h
```

These commands can be used in this way: python go2web.py [option] [argument]

- The response from the request is completely human readable.

Function to access the given URL:

```
import re
import socket
import sys
from urllib.parse import parse_qs, unquote, urlparse
```

Import the required modules, socket for creating a TCP/IP socket and connecting to the web server, and BeautifulSoup for parsing the HTML response.

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.settimeout(5)
```

This creates a new TCP/IP socket object using the socket module, with the AF_INET address family (IPv4) and SOCK_STREAM socket type. The settimeout method sets a timeout period of 5 seconds on the socket, meaning that if a connection cannot be established within that time, a socket.timeout exception will be raised.

```
start_pos = url.find("//") + 2
end_pos = url.find("/", start_pos)
if end_pos == -1:
    host = url[start_pos:]
    path = ""
else:
    host = url[start_pos:end_pos]
    end_pos_query = url.find("?")
    if end_pos_query == -1:
        end_pos_frag = url.find("#")
```

```
            if end_pos_frag == -1:
                end_pos_frag = len(url)
            path = url[end_pos:end_pos_frag]
        else:
            path = url[end_pos:end_pos_query]
```

These lines parse the given URL to extract the host and path components of the HTTP GET request. It finds the position of the double slashes // in the URL and adds 2 to get the start position of the host. It also finds the first forward slash / after the double slashes to determine the end of the host component and the beginning of the path component. If the forward slash is not found, it means that the URL does not contain a path component, so the path variable is set to an empty string. Otherwise, it extracts the path component from the URL, which may contain query parameters and/or a fragment identifier.

```
    port = 80
    try:
        s.connect((host, port))
    except socket.timeout:
        print("Connection timed out.")
        s.close()
        exit()
```

This sets the default port number for HTTP requests to 80 and establishes a connection to the web server using the connect() method of the socket object. It tries to connect to the host and port specified, and if the connection times out (due to the settimeout value), it will print an error message, close the socket, and exit the program.

```
    print("Sending request to", host, "at path", path, "\n")

    request = "GET " + path + " HTTP/1.1\r\nHost: " + host + "\r\n\r\n"
    s.sendall(request.encode())
```

This prints a message indicating the host and path of the HTTP GET request and then creates a GET request header with HTTP/1.1 protocol version, host name, and a blank line to signal the end of the header. It sends the request using the sendall() method of the socket object, which sends the entire message in one go.

```
    response = b""
    try:
        while True:
            buffer_size = 3072
            data = s.recv(buffer_size)
            if not data:
                break
            response += data
    except socket.timeout:
        pass
```

This receives the response from the server and stores it in a response variable. It uses a loop to receive data in chunks of 3072 bytes until there is no more data to receive. The recv() method of the socket object is used to receive data. If the method returns an empty byte string, it means that there is no more data to receive, and the loop breaks. If the method times out (due to the settimeout value), the loop will also break.

```
    content = response.decode()
    body_start = content.find('\r\n\r\n') + 4
    body = content[body_start:]

```

```
    soup = BeautifulSoup(body, 'html.parser')
    body = soup.get_text()
    body = ' '.join(body.split())

    print('\nContent:')
    print(body)
```

This decodes the response from bytes to a string and extracts the body of the response by finding the start of the body after the header. It then uses BeautifulSoup to parse the HTML response and extract the text content of the HTML. Finally, it cleans up the text by removing any extra whitespace and prints the resulting text to the console.

Overall, this function takes a URL as input, connects to the web server using a TCP/IP socket, sends an HTTP GET request, receives the response, extracts the body content of the response, and prints it to the console.

Output example:

```
python go2web.py -u https://quotes.toscrape.com/
Sending request to quotes.toscrape.com at path /

Content:
Quotes to Scrape Quotes to Scrape Login “The world as we have created it is a process
    of our thinking. It cannot be changed without changing our thinking”. by Albert
    Einstein (about) Tags: change deep-thoughts thinking world “It is our choices,
    Harry, that show what we truly are, far more than our abilities”. by J.K. Rowling
    (about) Tags: abilities choices “There are only two ways to live your life. One is
    as though nothing is a miracle. The other is as though everything is a miracle”.
    by Albert Einstein (about) Tags: inspirational life live miracle miracles “The
    person, be it gentleman or lady, who has not pleasure in a good novel, must be
    intolerably stupid”. by Jane Austen (about) Tags: aliteracy books classic humor
    “Imperfection is beauty, madness is genius and it’s better to be absolutely
    ridiculous than absolutely boring”. by Marilyn Monroe (about) Tags: be-yourself
    inspirational “Try not to become a man of success. Rather become a man of value”.
    by Albert Einstein (about) Tags: adulthood success value “It is better to be hated
    for what you are than to be loved for what you are not”. by André Gide (about)
    Tags: life love “I have not failed. I’ve just found 10,000 ways that won’t work”.
    by Thomas A. Edison (about) Tags: edison failure inspirational paraphrased “A
    woman is like a tea bag; you never know how strong it is until it’s in hot water”.
    by Eleanor Roosevelt (about) Tags: misattributed-eleanor-roosevelt “A day without
    sunshine is like, you know, night”. by Steve Martin (about) Tags: humor obvious
    simile Next → Top Ten tags love inspirational life humor books reading friendship
    friends truth simile Quotes by: GoodReads.com Made with  by Scrapinghub
```

Function to print the top 10 results of the search term:

```
    import re
    import socket
    import sys
    from urllib.parse import parse_qs, unquote, urlparsep
```

Import the required modules.

```
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.settimeout(5)

    host = "www.google.com"
    port = 80

    try:
        s.connect((host, port))
```

```
    except socket.timeout:
        print("Connection timed out.")
        s.close()
        exit()
```

This creates a TCP/IP socket and connects to the Google search page. If a connection cannot be established within 5 seconds, the function prints an error message and exits.

```
    encoded_search_term = search_term.replace(" ", "+")
    request = f'GET /search?q={encoded_search_term} HTTP/1.1\r\nHost: www.google.com\
    r\n\r\n'
    s.sendall(request.encode())
```

This encodes the search term by replacing any spaces with "+" characters and sends an HTTP GET request to the Google search page. The request includes the encoded search term as a query parameter.

```
    response = b""
    try:
        while True:
            buffer_size = 3072
            data = s.recv(buffer_size)
            if not data:
                break
            response += data
    except socket.timeout:
        pass
```

This receives the response from the server in chunks of 3072 bytes until there is no more data to receive. If the method times out (due to the settimeout value), the loop will break.

```
    print("Top 10 results from Google:")
    print()

    soup = BeautifulSoup(response, 'html.parser')
    links = soup.find_all(class_="egMi0 kCrYT")

    for i, elem in enumerate(links[:10]):
        title = elem.find('h3').get_text()
        url_elem = elem.find('a')
        url = unquote(url_elem['href'])
        url_match = re.search('(?P<url>https?://[^&\s]+)', url)
        url = url_match.group("url")
        url = unquote(url)
        url = re.sub('&sa=.*', '', url)
        url_parts = urlparse(url)
        url_params = parse_qs(url_parts.query)

        if 'url' in url_params:
            url = url_params['url'][0]

        print(f'{int(i+1)}. {title}')
        print(f'Access the link: {url}')
        print()
```

This parses the HTML response using BeautifulSoup and extracts the top 10 search results. For each result, it extracts the title and URL from the HTML, decodes the URL using unquote(), removes any unnecessary query parameters, and prints the result to the console.

Output example:

```
python go2web.py -s "hello kitty"
Top 10 results from Google:

1. The Official Home of Hello Kitty & Friends
Access the link: https://www.sanrio.com/

2. HELLO KITTY - Sanrio
Access the link: https://www.sanrio.com/collections/hello-kitty

3. Hello Kitty - Wikipedia
Access the link: https://en.wikipedia.org/wiki/Hello_Kitty

4. Produse Hello Kitty - eMAG.ro
Access the link: https://www.emag.ro/brands/brand/hello-kitty

5. Hello Kitty - Facebook
Access the link: https://www.facebook.com/hellokitty/

6. Avril Lavigne - Hello Kitty (Official Video) - YouTube
Access the link: https://www.youtube.com/watch?v=LiaYDPRedWQ

7. Hello Kitty in romana - Seria 1 - YouTube
Access the link: https://www.youtube.com/watch?v=lyNWq5oXFZY

8. Hello Kitty (@hellokitty) / Twitter
Access the link: https://twitter.com/hellokitty

9. Hello Kitty (@hellokitty) • Instagram photos and videos
Access the link: https://www.instagram.com/hellokitty/

10. Cumpara autocolante auto Hello Kitty AK47 in Chisinau - Autoshina
Access the link: https://autoshina.md/ro/autocolante-auto-hello-kitty-ak47/
```

Function to print the help menu:

```
def help():
    help_message = """Available CLI:
    -u <URL>          # make an HTTP request to the specified URL and print the response
    -s <search-term>  # make an HTTP request to search the term using Google and print top 10 results
    -h                # show this help
Usage: python go2web.py [option] [argument]"""

    print(help_message)
```

# 3 Conclusion

This laboratory work was quite challenging but also rewarding. It required a good understanding of HTTP requests and responses, as well as the ability to work with sockets and parse HTML content. One of the most challenging aspects was making HTTP requests without using any third-party libraries. This required a lot of low-level programming using the socket module in Python. However, it also helped to gain a better understanding of how HTTP works under the hood and how web browsers interact with web servers. Overall, this laboratory work provided a great opportunity to learn more about web development and HTTP, and to develop some practical skills in making HTTP requests and parsing HTML content. It also helped to understand the importance of caching and content negotiation in web development.

## References

[1] Python

# 4 GitHub

https://github.com/eugencic/utm-pw/tree/main/lab2