

Technologie Sieciowe

Lista 5

Mateusz Kościelniak

1. Omówienie skryptu.

Najpierw definiujemy adres połączenia oraz port na jakim będziemy odbierać żądania. Następnie zostaje odpalona pętla która warunkuje ciągłość połączenia, dalej mamy pętlę z warunkiem otrzymania żądania „get_request”. Jeśli klient wysyła żądanie „GET” to w odpowiedzi zostaje załadowany plik „index.html” i wysłany do klienta, w przeciwnym wypadku klient dostaje kod błędu 403 - „RC_FORBIDDEN” .

```
1  use HTTP::Daemon;
2  use HTTP::Status;
3  #use IO::File;
4
5  my $d = HTTP::Daemon->new(
6      LocalAddr => 'localhost',      #ustawiamy nazwe naszego hosta
7      LocalPort => 4321,             #port na którym będzie odbierał nasz serwer
8  ) || die;
9
10 print "Please contact me at: <URL:", $d->url, ">\n";
11
12
13 while (my $c = $d->accept) {       #zmienna $c przechowuje wskaznik do połączenia z klientem
14     while (my $r = $c->get_request) { #$r przechowuje wskaznik do żądania
15         if ($r->method eq 'GET') {    #jeśli klient dostanie wyśle żądanie 'GET' w odpowiedzi w oknie przeglądarki
16             $file_s = "./index.html"; #zostanie mu wyświetlona treść pliku index.html
17             $c->send_file_response($file_s); #index.html - jakis istniejący plik
18         }
19     }
20     else {
21         $c->send_error(RC_FORBIDDEN)
22     }
23 }
24
25 }
26 $c->close;
27 undef($c);
28 }
```

2. Połączenie za pomocą przeglądarki.

Wszedłem na adres który wyświetlił mi się podczas uruchomienia skryptu „server3.pl”, w ten sposób nawiązałem połączenie z serwerem. W oknie przeglądarki wyświetliła mi się zawartość pliku „index.html”.

3. Mój własny serwer

Serwer napisałem w języku programowania „JAVA”. Aby połączyć się z serwerem, należy w przeglądarce wpisać adres: „http://localhost:1235/myserver?home”. Serwer obsługuje żądania klienta i dodatkowo wyświetla nagłówki jego żądania. W katalogu „my server” zostały umieszczone trzy pliki „html”. Po otwarciu jednej ze stron możliwe jest przechodzenie pomiędzy stronami poprzez umieszczone na nich odnośniki.

← → ↻ ⓘ localhost:1235/myserver?first

This is the first page.

[Visit second page](#)

Your header request:

HEADERS:

NAME: Accept-encoding VALUE: gzip, deflate, br

NAME: Accept VALUE: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3

NAME: Connection VALUE: keep-alive

NAME: Referer VALUE: http://localhost:1235/myserver?home

NAME: Host VALUE: localhost:1235

NAME: User-agent VALUE: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.157 Safari/537.36

NAME: Accept-language VALUE: pl-PL,pl;q=0.9,en-US;q=0.8,en;q=0.7

NAME: Upgrade-insecure-requests VALUE: 1

HTTP METHOD:

GET

QUERY:

/myserver?first

- Accept encoding – określa rodzaj kodowania za pomocą którego zostanie przesłana zawartość
- Accept - zawiera akceptowane przez przeglądarkę treści
- Connection - informuje o rodzaju nawiązanego połączenia
- Refer – zawiera adres strony z której nastąpiło przekierowanie
- Host – zawiera domenę do której wysyłane jest żądanie
- User agent – zawiera informacje o programie za pomocą którego dokonano zapytania
- Accept language – określa w jakim języku klient chce czytać strony

4. Użycie ‘sniffera’

Do przechwycenia komunikatu użyłem programu „wireshark”, używając filtra HTTP.

Komunikacja klienta i serwera:

32	4.103380387	ip6-localhost	ip6-localhost	HTTP	580	GET /myserver?home	HTTP/1.1
36	4.106037103	ip6-localhost	ip6-localhost	HTTP	1177	HTTP/1.1 200 OK	
49	6.887540186	ip6-localhost	ip6-localhost	HTTP	579	GET /myserver?first	HTTP/1.1
52	6.889185976	ip6-localhost	ip6-localhost	HTTP	1179	HTTP/1.1 200 OK	

Żądanie klienta :

```
▶ Frame 49: 579 bytes on wire (4632 bits), 579 bytes captured (4632 bits) on interface 0
▶ Linux cooked capture
▶ Internet Protocol Version 6, Src: ip6-localhost (::1), Dst: ip6-localhost (::1)
▶ Transmission Control Protocol, Src Port: 36390, Dst Port: 1235, Seq: 493, Ack: 1168, Len: 491
▼ Hypertext Transfer Protocol
  GET /myserver?first HTTP/1.1\r\n
    ▶ [Expert Info (Chat/Sequence): GET /myserver?first HTTP/1.1\r\n]
      Request Method: GET
    ▶ Request URI: /myserver?first
      Request Version: HTTP/1.1
    Host: localhost:1235\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.157 Safari/537.36\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3\r\n
    Referer: http://localhost:1235/myserver?home\r\n
    Accept-Encoding: gzip, deflate, br\r\n
    Accept-Language: pl-PL,pl;q=0.9,en-US;q=0.8,en;q=0.7\r\n
    \r\n
    [Full request URI: http://localhost:1235/myserver?first]
    [HTTP request 2/2]
    [Prev request in frame: 32]
    [Response in frame: 52]
```

Odpowiedź serwera:

```
▶ Frame 52: 1179 bytes on wire (9432 bits), 1179 bytes captured (9432 bits) on interface 0
▶ Linux cooked capture
▶ Internet Protocol Version 6, Src: ip6-localhost (:::1), Dst: ip6-localhost (:::1)
▶ Transmission Control Protocol, Src Port: 1235, Dst Port: 36390, Seq: 1246, Ack: 984, Len: 1091
▶ [2 Reassembled TCP Segments (1169 bytes): #50(78), #52(1091)]
▼ Hypertext Transfer Protocol
  ▼ HTTP/1.1 200 OK\r\n
    ▼ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      [HTTP/1.1 200 OK\r\n]
      [Severity level: Chat]
      [Group: Sequence]
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
      Date: Fri, 14 Jun 2019 09:10:30 GMT\r\n
    ▼ Content-length: 1091\r\n
      [Content length: 1091]
      \r\n
      [HTTP response 2/2]
      [Time since request: 0.001645790 seconds]
      [Prev request in frame: 32]
      [Prev response in frame: 36]
      [Request in frame: 49]
      [Request URI: http://localhost:1235/myserver?first]
      File Data: 1091 bytes
    ▼ Data (1091 bytes)
      Data: 3c21444f43545950452068746d6c3e0a3c68746d6c3e0a20...
      [Length: 1091]
```