NETWORKING BASICS – EX3 – MAYA RASKIN

Introduction

This exercise entails the implementation of RFC 2616 in C++11 utilizing sockets. The web-server executes asynchronous tasks using C++ selectors.

Testing

Postman was employed for conducting tests, with each request accompanied by a corresponding test within the collection provided alongside this submission.

Code

The primary objective was to eliminate strings. Consequently, upon receiving each message, it is promptly parsed into a self-implemented *HttpRequest* object.
Subsequently, the request is processed, generating an *HttpResponse* object. Prior to transmission via the socket, the *HttpResponse* object undergoes parsing into a valid HTTP response string.

The object-oriented perspective of this implementation involves several key classes designed to facilitate its functionality:

1. Uri: This class defines a URI, featuring a single string property named path.
2. SingletonHtmlPlaceholder: Implementing the Singleton design pattern, this class ensures the creation of a single instance of a string.
3. HttpMessage:
   • Enumerations for *HttpVersion*, *HttpMethod*, *HttpStatusCode*, *QLanguage* (representing the "lang" query parameter), and utility functions for converting strings to enums/class instances and vice versa.
   • QueryParams: A container for a map<string, string> of query parameters.
   • HttpMessageInterface: Comprising class members such as content, headers and version. The interface automatically updates the *Content-Length* header when content is set, initially set to 0.
   • HttpRequest: Implementing the *HttpMessageInterface*, this class also includes class members such as method, params, and URI.
   • HttpResponse: Also implementing the *HttpMessageInterface*, this class adds the status code class member.
4. HttpServer:
   - Defined types for a generic HTTP request handler: *HttpRequestHandler_t = std::function<HttpResponse(const HttpRequest&)>*.
   - A collection is established to store all server request handlers: *map<Uri, map<HttpMethod, HttpRequestHandler_t>>,* facilitating the *HandleHttpRequest* function's determination of when to return *NotFound* or *MethodNotAllowed*
5. SocketService: This class encompasses all functions relating to the sockets utilized by the *HttpServer*.

API

The web-server provides the following endpoints:

1. **GET /health:**
   Query parameters: None
   Response: HTTP Response with a 200 OK status code and text/plain content, containing the string "OK"
2. **HEAD /health:**
   Query parameters: None
   Response: Similar to **GET /health**, but without content
3. **TRACE /health:**
   Query parameters: None
   Response: HTTP Response with a 200 OK status code and message/http content, reflecting the sent request
4. **OPTIONS /health:**
   Query parameters: None
   Response: HTTP Response with a 204 No Content status code. The *Allow* header lists the available REST methods for the **/health** route, which are *OPTIONS*, *TRACE*, *HEAD*, and *GET*
5. **GET /index.html:**
   Query parameters: An **optional** parameter "lang", which can contain one of the strings "he", "en", or "fr"
   Response: HTTP Response with a 200 OK status code. If no query parameters are provided or an invalid string other than "en", "he", or "fr" is supplied, the server defaults to returning the English version of the HTML. Otherwise, it returns the HTML corresponding to the specified language
6. **HEAD /index.html:**
   Query parameters: An **optional** parameter "lang", which can contain one of the strings "he", "en", or "fr"
   Response: Similar to **GET /index.html**, but without content
7. **TRACE /index.html:**
   Query parameters: None
   Response: HTTP Response with a 200 OK status code and message/http content, reflecting the sent request
8. **POST /index.html:**
   Query parameters: None
   Payload: a plain text
   Response: HTTP Response with a 200 OK status code
9. **PUT /index.html:**
   Query parameters: None
   Payload: a plain text
   Response: HTTP Response with a 200 OK status code and text/plain content, containing the string "OK".
10. **DELTE /index.html:**
    Query parameters: None
    Response: HTTP Response with a 200 OK status code and text/plain content, containing the string "OK".

NETWORKING BASICS – EX3 – MAYA RASKIN

Wireshark screenshots

A consolidated .pcap file comprising all the requests has been included in the MAMA submission as well.

	Time	Source	Destination	Protocol	Length	Info	Data
1	0.000000	127.0.0.1	127.0.0.1	TCP	44	8080 → 50758 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
2	7.252192	::1	::1	TCP	76	50838 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=65475 WS=256 SACK_P…	
3	7.252211	::1	::1	TCP	64	8080 → 50838 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
4	7.253163	127.0.0.1	127.0.0.1	TCP	56	50839 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_P…	
5	7.253218	127.0.0.1	127.0.0.1	TCP	56	8080 → 50839 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS…	
6	7.253246	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=1 Ack=1 Win=2619648 Len=0	
7	7.253550	127.0.0.1	127.0.0.1	HTTP	255	OPTIONS /health HTTP/1.1	
8	7.253566	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=1 Ack=212 Win=2619648 Len=0	
9	7.254887	127.0.0.1	127.0.0.1	HTTP	181	HTTP/1.1 204	
10	7.254907	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=212 Ack=138 Win=2619648 Len=0	
11	7.345601	127.0.0.1	127.0.0.1	HTTP	253	TRACE /health HTTP/1.1	
12	7.345633	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=138 Ack=421 Win=2619392 Len=0	
13	7.347103	127.0.0.1	127.0.0.1	HTTP	405	HTTP/1.1 200 OK (message/http)	
14	7.347146	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=421 Ack=499 Win=2619136 Len=0	
15	7.422265	127.0.0.1	127.0.0.1	HTTP	252	HEAD /health HTTP/1.1	
16	7.422295	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=499 Ack=629 Win=2619136 Len=0	
17	7.423963	127.0.0.1	127.0.0.1	HTTP	175	HTTP/1.1 200 OK	
18	7.424003	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=629 Ack=630 Win=2619136 Len=0	
19	7.516517	127.0.0.1	127.0.0.1	HTTP	251	GET /health HTTP/1.1	
20	7.516565	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=630 Ack=836 Win=2619136 Len=0	
21	7.518500	127.0.0.1	127.0.0.1	HTTP	177	HTTP/1.1 200 OK (text/plain)	
22	7.518528	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=836 Ack=763 Win=2618880 Len=0	
23	7.594268	127.0.0.1	127.0.0.1	HTTP	255	GET /some_route HTTP/1.1	
24	7.594309	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=763 Ack=1047 Win=2618880 Len=0	
25	7.596690	127.0.0.1	127.0.0.1	HTTP	156	HTTP/1.1 404 Not Found	
26	7.596728	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=1047 Ack=875 Win=2618880 Len=0	
27	7.656523	127.0.0.1	127.0.0.1	HTTP	254	DELETE /health HTTP/1.1	
28	7.656555	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=875 Ack=1257 Win=2618624 Len=0	
29	7.658434	127.0.0.1	127.0.0.1	HTTP	165	HTTP/1.1 405 Method Not Allowed	
30	7.658472	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=1257 Ack=996 Win=2618624 Len=0	
31	7.733126	127.0.0.1	127.0.0.1	HTTP	259	OPTIONS /index.html HTTP/1.1	
32	7.733164	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=996 Ack=1472 Win=2618368 Len=0	
33	7.735240	127.0.0.1	127.0.0.1	HTTP	200	HTTP/1.1 204	
34	7.735269	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=1472 Ack=1152 Win=2618624 Len=0	
35	7.811371	127.0.0.1	127.0.0.1	HTTP	257	TRACE /index.html HTTP/1.1	
36	7.811407	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=1152 Ack=1685 Win=2618112 Len=0	
37	7.813391	127.0.0.1	127.0.0.1	HTTP	409	HTTP/1.1 200 OK (message/http)	
38	7.813422	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=1685 Ack=1517 Win=2618112 Len=0	
39	7.891023	127.0.0.1	127.0.0.1	HTTP	317	POST /index.html HTTP/1.1 (text/plain)	
40	7.891065	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=1517 Ack=1958 Win=2617856 Len=0	
41	7.895296	127.0.0.1	127.0.0.1	HTTP	149	HTTP/1.1 200 OK	
42	7.895358	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=1958 Ack=1622 Win=2618112 Len=0	
43	7.966663	127.0.0.1	127.0.0.1	HTTP	256	HEAD /index.html HTTP/1.1	
44	7.966713	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=1622 Ack=2170 Win=2617600 Len=0	
45	7.975527	127.0.0.1	127.0.0.1	HTTP	174	HTTP/1.1 200 OK	
46	7.975572	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=2170 Ack=1752 Win=2617856 Len=0	
47	8.030223	127.0.0.1	127.0.0.1	HTTP	255	GET /index.html HTTP/1.1	
48	8.030269	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=1752 Ack=2381 Win=2617600 Len=0	
49	8.069215	127.0.0.1	127.0.0.1	HTTP	727	HTTP/1.1 200 OK (text/html)	
50	8.069262	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=2381 Ack=2435 Win=2617344 Len=0	
51	8.153616	127.0.0.1	127.0.0.1	HTTP	304	PUT /index.html HTTP/1.1 (text/plain)	
52	8.153689	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=2435 Ack=2641 Win=2617344 Len=0	
53	8.156864	127.0.0.1	127.0.0.1	HTTP	177	HTTP/1.1 200 OK (text/plain)	
54	8.156904	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=2641 Ack=2568 Win=2617088 Len=0	
55	8.258383	127.0.0.1	127.0.0.1	HTTP	263	GET /index.html?lang=en HTTP/1.1	
56	8.258423	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=2568 Ack=2860 Win=2617088 Len=0	
57	8.261350	127.0.0.1	127.0.0.1	HTTP	755	HTTP/1.1 200 OK (text/html)	
58	8.261400	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=2860 Ack=3279 Win=2616320 Len=0	
59	8.309235	127.0.0.1	127.0.0.1	HTTP	263	GET /index.html?lang=he HTTP/1.1	
60	8.309283	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=3279 Ack=3079 Win=2616832 Len=0	
61	8.312356	127.0.0.1	127.0.0.1	HTTP	860	HTTP/1.1 200 OK (text/html)	
62	8.312397	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=3079 Ack=4095 Win=2615552 Len=0	
63	8.385517	127.0.0.1	127.0.0.1	HTTP	263	GET /index.html?lang=fr HTTP/1.1	
64	8.385567	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=4095 Ack=3298 Win=2616576 Len=0	
65	8.388688	127.0.0.1	127.0.0.1	HTTP	829	HTTP/1.1 200 OK (text/html)	
66	8.388737	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=3298 Ack=4880 Win=2614784 Len=0	
67	8.478869	127.0.0.1	127.0.0.1	HTTP	258	DELETE /index.html HTTP/1.1	
68	8.478916	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=4880 Ack=3512 Win=2616320 Len=0	
69	8.481507	127.0.0.1	127.0.0.1	HTTP	177	HTTP/1.1 200 OK (text/plain)	
70	8.481541	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=3512 Ack=5013 Win=2614784 Len=0	
71	8.540908	127.0.0.1	127.0.0.1	HTTP	255	GET /index.html HTTP/1.1	
72	8.540960	127.0.0.1	127.0.0.1	TCP	44	8080 → 50839 [ACK] Seq=5013 Ack=3723 Win=2616064 Len=0	
73	8.544725	127.0.0.1	127.0.0.1	HTTP	727	HTTP/1.1 200 OK (text/html)	
74	8.544773	127.0.0.1	127.0.0.1	TCP	44	50839 → 8080 [ACK] Seq=3723 Ack=5696 Win=2614016 Len=0	