

计算机网络第二次实验

朱浩泽 1911530 计算机科学与技术

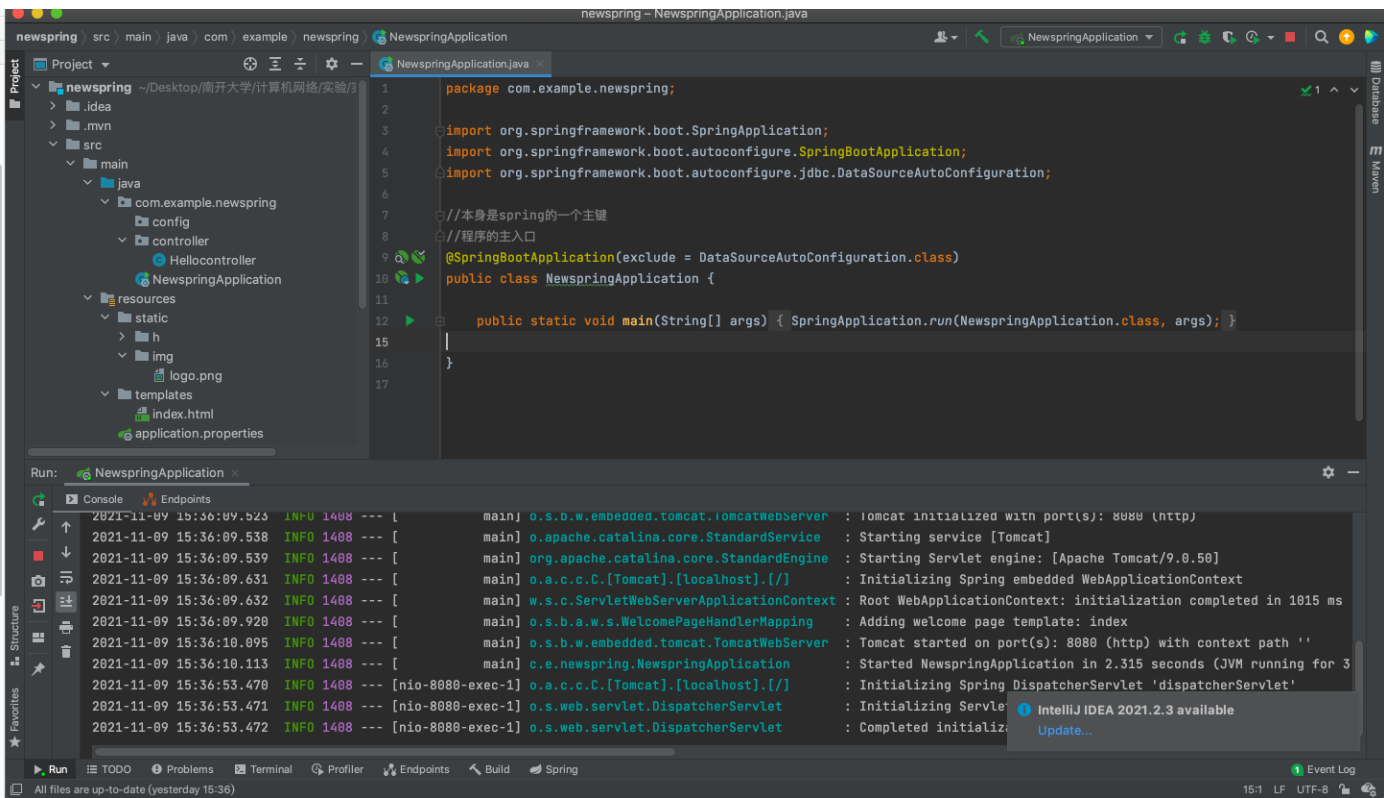
实验要求

1. 搭建Web服务器（自由选择系统），并制作简单的Web页面，包含简单文本信息（至少包含专业、学号、姓名）和自己的LOGO。
2. 通过浏览器获取自己编写的Web页面，使用Wireshark捕获浏览器与Web服务器的交互过程，并进行简单的分析说明。
3. 提交实验报告。

Web服务器的搭建

平台：springboot

在springboot中搭建一个简单的Web服务器，其框架如下



启动该服务器后，在浏览器中输入localhost:8080即可进入该网站，该网站包括了姓名、学号、专业，以及自己设计的一张图片，其截图如下：

专业： 计算机科学与技术

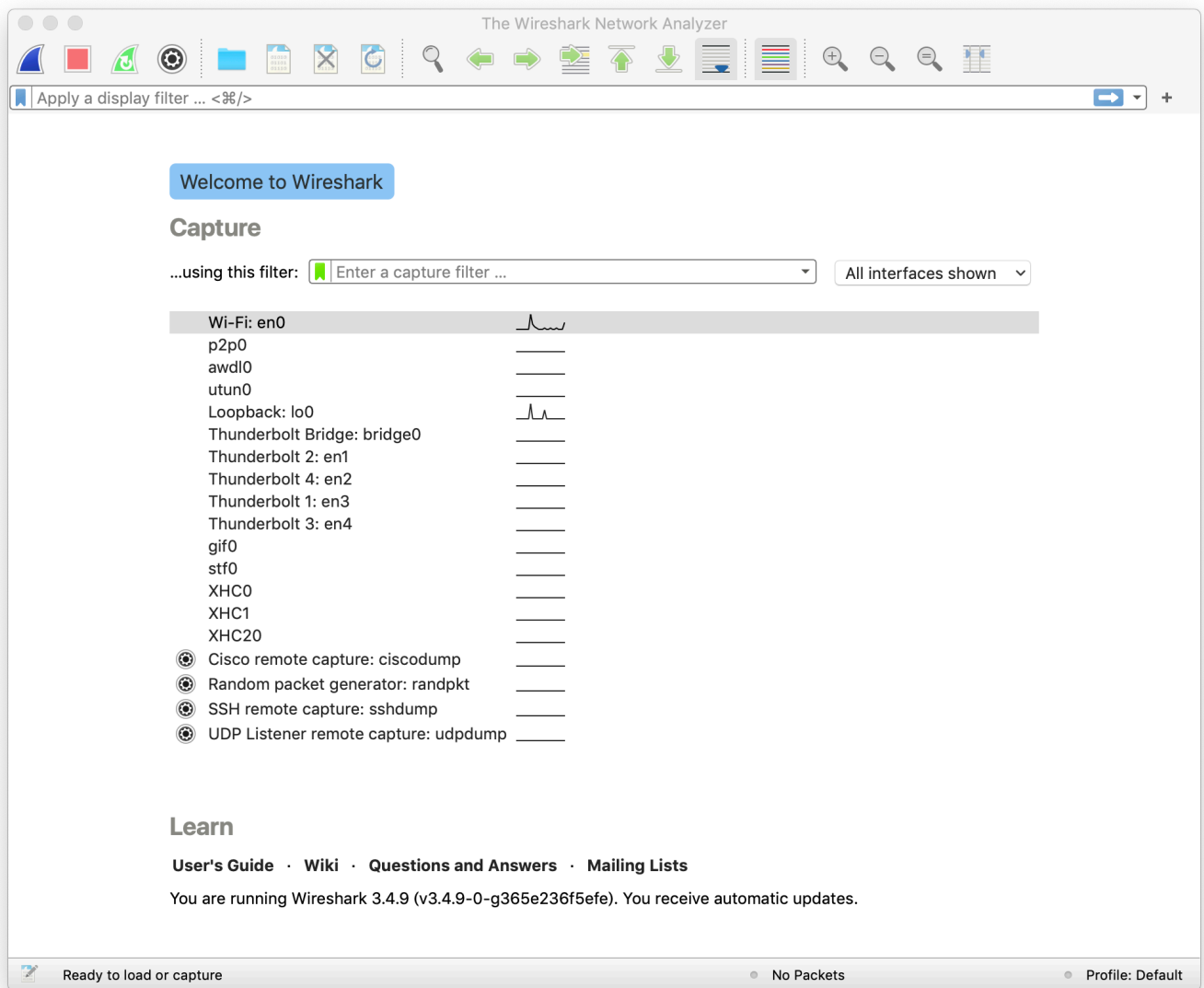
姓名： 朱浩泽

学号： 1911530



捕获交互过程

我们在实验中首先开启Wireshark，并开启抓包，在浏览器中输入自己设计的网页的网址，观察Wireshark界面如下图所示：



可以看出，由于客户端和服务端都是本机，所以我们选择Loopback:lo0进入。进入后我们使用过滤器，取出本机端口与本机交互的数据包显示，以分析浏览器与Web服务器的交互过程。

No.	Time	Source	Destination	Protocol	Length	Info
19	3.536277	:::1	:::1	TCP	88	51912 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=16324 WS=64 TSval=722240109 TSecr=0 SACK_PERM=1
20	3.536345	:::1	:::1	TCP	88	8080 → 51912 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=16324 WS=64 TSval=722240109 TSecr=722240109 SACK_PERM=1
21	3.536360	:::1	:::1	TCP	76	51912 → 8080 [ACK] Seq=1 Ack=1 Win=407744 Len=0 TSval=722240109 TSecr=722240109
22	3.536371	:::1	:::1	TCP	76	[TCP Window Update] 8080 → 51912 [ACK] Seq=1 Ack=1 Win=407744 Len=0 TSval=722240109 TSecr=722240109
23	3.549123	:::1	:::1	HTTP	432	GET / HTTP/1.1
24	3.549151	:::1	:::1	TCP	76	8080 → 51912 [ACK] Seq=1 Ack=357 Win=407424 Len=0 TSval=722240122 TSecr=722240122
25	3.551212	:::1	:::1	TCP	88	51913 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=16324 WS=64 TSval=722240124 TSecr=0 SACK_PERM=1
26	3.551273	:::1	:::1	TCP	88	8080 → 51913 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=16324 WS=64 TSval=722240124 TSecr=722240124 SACK_PERM=1
27	3.551280	:::1	:::1	TCP	76	51913 → 8080 [ACK] Seq=1 Ack=1 Win=407744 Len=0 TSval=722240124 TSecr=722240124
28	3.551288	:::1	:::1	TCP	76	[TCP Window Update] 8080 → 51913 [ACK] Seq=1 Ack=1 Win=407744 Len=0 TSval=722240124 TSecr=722240124
29	3.552757	:::1	:::1	TCP	549	8080 → 51912 [PSH, ACK] Seq=1 Ack=357 Win=407424 Len=473 TSval=722240125 TSecr=722240122 [TCP segment of a reassembled
30	3.552788	:::1	:::1	TCP	76	51912 → 8080 [ACK] Seq=357 Ack=474 Win=407296 Len=0 TSval=722240125 TSecr=722240125
31	3.552969	:::1	:::1	HTTP	81	HTTP/1.1 200 (text/html)
32	3.552981	:::1	:::1	TCP	76	51912 → 8080 [ACK] Seq=357 Ack=479 Win=407296 Len=0 TSval=722240125 TSecr=722240125
33	3.554034	:::1	:::1	HTTP	495	GET /img/logo.png HTTP/1.1
34	3.554063	:::1	:::1	TCP	76	8080 → 51913 [ACK] Seq=1 Ack=420 Win=407360 Len=0 TSval=722240126 TSecr=722240126
35	3.563871	:::1	:::1	HTTP	313	HTTP/1.1 304
36	3.563898	:::1	:::1	TCP	76	51913 → 8080 [ACK] Seq=420 Ack=238 Win=407552 Len=0 TSval=722240135 TSecr=722240135
37	3.566844	:::1	:::1	HTTP	495	GET /img/logo.png HTTP/1.1
38	3.566875	:::1	:::1	TCP	76	8080 → 51912 [ACK] Seq=479 Ack=776 Win=406976 Len=0 TSval=722240137 TSecr=722240137
39	3.569507	:::1	:::1	HTTP	313	HTTP/1.1 304
40	3.569540	:::1	:::1	TCP	76	51912 → 8080 [ACK] Seq=776 Ack=716 Win=407040 Len=0 TSval=722240139 TSecr=722240139
41	5.490469	:::1	:::1	HTTP	432	GET / HTTP/1.1
42	5.490504	:::1	:::1	TCP	76	8080 → 51912 [ACK] Seq=716 Ack=1132 Win=406656 Len=0 TSval=722242040 TSecr=722242040
43	5.493415	:::1	:::1	TCP	549	8080 → 51912 [PSH, ACK] Seq=716 Ack=1132 Win=406656 Len=473 TSval=722242043 TSecr=722242040 [TCP segment of a reassembled
44	5.493445	:::1	:::1	TCP	76	51912 → 8080 [ACK] Seq=1132 Ack=1189 Win=406592 Len=0 TSval=722242043 TSecr=722242043
45	5.493637	:::1	:::1	HTTP	81	HTTP/1.1 200 (text/html)
46	5.493655	:::1	:::1	TCP	76	51912 → 8080 [ACK] Seq=1132 Ack=1194 Win=406592 Len=0 TSval=722242043 TSecr=722242043
109	34.532624	:::1	:::1	TCP	76	51913 → 8080 [FIN, ACK] Seq=420 Ack=238 Win=407552 Len=0 TSval=722271005 TSecr=722240135
110	34.532689	:::1	:::1	TCP	76	8080 → 51913 [ACK] Seq=238 Ack=421 Win=407360 Len=0 TSval=722271005 TSecr=722271005
111	34.533143	:::1	:::1	TCP	76	8080 → 51913 [FIN, ACK] Seq=238 Ack=421 Win=407360 Len=0 TSval=722271006 TSecr=722271005
112	34.533200	:::1	:::1	TCP	76	51913 → 8080 [ACK] Seq=421 Ack=239 Win=407552 Len=0 TSval=722271006 TSecr=722271006
175	64.533607	:::1	:::1	TCP	76	51912 → 8080 [FIN, ACK] Seq=1132 Ack=1194 Win=406592 Len=0 TSval=722300968 TSecr=722242043
176	64.533691	:::1	:::1	TCP	76	8080 → 51912 [ACK] Seq=1194 Ack=1133 Win=406656 Len=0 TSval=722300968 TSecr=722300968
177	64.534144	:::1	:::1	TCP	76	8080 → 51912 [FIN, ACK] Seq=1194 Ack=1133 Win=406656 Len=0 TSval=722300969 TSecr=722300968
178	64.534200	:::1	:::1	TCP	76	51912 → 8080 [ACK] Seq=1133 Ack=1195 Win=406592 Len=0 TSval=722300969 TSecr=722300969

▶ Frame 19: 88 bytes on wire (704 bits), 88 bytes captured (704 bits) on interface lo0, id 0
 ▶ Null/Loopback
 ▶ Internet Protocol Version 6, Src: ::1, Dst: ::1
 ▶ Transmission Control Protocol, Src Port: 51912, Dst Port: 8080, Seq: 0, Len: 0

HTTP相关数据包

23	3.549123	:::1	:::1	HTTP	432	GET / HTTP/1.1
31	3.552969	:::1	:::1	HTTP	81	HTTP/1.1 200 (text/html)
33	3.554034	:::1	:::1	HTTP	495	GET /img/logo.png HTTP/1.1
35	3.563871	:::1	:::1	HTTP	313	HTTP/1.1 304

首先我们可以看出，springboot使用的是http1.1协议首先发送一个访问页面的get请求，客户端收到get请求后将所请求内容发送给客户端

- 访问页面的请求

```
▶ Frame 23: 432 bytes on wire (3456 bits), 432 bytes captured (3456 bits) on interface lo0, id 0
▶ Null/Loopback
▶ Internet Protocol Version 6, Src: ::1, Dst: ::1
▶ Transmission Control Protocol, Src Port: 51912, Dst Port: 8080, Seq: 1, Ack: 1, Len: 356
▼ Hypertext Transfer Protocol
  ▶ GET / HTTP/1.1\r\n
    Host: localhost:8080\r\n
    Upgrade-Insecure-Requests: 1\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
    User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/14.1.2 Safari/605.1.15\r\n
    Accept-Language: zh-cn\r\n
    Accept-Encoding: gzip, deflate\r\n
    Connection: keep-alive\r\n
    \r\n
    [Full request URI: http://localhost:8080/]
    [HTTP request 1/3]
    [Response in frame: 31]
    [Next request in frame: 37]
```

0020	00 00 00 00 00 00 00 00	00 00 00 01 ca c8 1f 90C...@...
0030	20 76 84 7e 96 d2 c8 74	80 18 18 e3 01 8c 00 00t.....
0040	01 01 08 0a 2b 0c 82 7a	2b 0c 82 6d 47 45 54 20z...mGET
0050	2f 20 48 54 54 50 2f 31	2e 31 0d 0a 48 6f 73 74	/ HTTP/1.1..Host
0060	3a 20 6c 6f 63 61 6c 68	6f 73 74 3a 38 30 38 30	: localh ost:8080
0070	0d 0a 55 70 67 72 61 64	65 2d 49 6e 73 65 63 75	..Upgrad e-Insecu
0080	72 65 2d 52 65 71 75 65	73 74 73 3a 20 31 0d 0a	re-Reqe sts: 1..

No.: 23 · Time: 3.549123 · Source: ::1 · Destination: ::1 · Protocol: HTTP · Length: 432 · Info: GET / HTTP/1.1

Help Close

```
▶ Frame 33: 495 bytes on wire (3960 bits), 495 bytes captured (3960 bits) on interface lo0, id 0
▶ Null/Loopback
▶ Internet Protocol Version 6, Src: ::1, Dst: ::1
▶ Transmission Control Protocol, Src Port: 51913, Dst Port: 8080, Seq: 1, Ack: 1, Len: 419
▼ Hypertext Transfer Protocol
  ▶ GET /img/logo.png HTTP/1.1\r\n
    Host: localhost:8080\r\n
    Connection: keep-alive\r\n
    Accept: image/png,image/svg+xml,image/*;q=0.8,video/*;q=0.8,*/*;q=0.5\r\n
    If-Modified-Since: Wed, 10 Nov 2021 10:37:51 GMT\r\n
    User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/14.1.2 Safari/605.1.15\r\n
    Referer: http://localhost:8080/\r\n
    Accept-Encoding: gzip, deflate\r\n
    \r\n
    [Full request URI: http://localhost:8080/img/logo.png]
    [HTTP request 1/1]
    [Response in frame: 35]
```

0000	1e 00 00 00 60 0d e0 43	01 c3 06 40 00 00 00 00C...@...
0010	00 00 00 00 00 00 00 00	00 00 00 01 00 00 00 00t.....
0020	00 00 00 00 00 00 00 00	00 00 00 01 ca c9 1f 90t.....
0030	b8 3d 0b ff 02 0f 9f 75	80 18 18 e3 01 cb 00 00u.....
0040	01 01 08 0a 2b 0c 82 7e	2b 0c 82 7c 47 45 54 20z...mGET
0050	2f 69 6d 67 2f 6c 6f 67	6f 2e 70 6e 67 20 48 54	/img/log o.png HT
0060	54 50 2f 31 2e 31 0d 0a	48 6f 73 74 3a 20 6c 6f	TP/1.1.. Host: lo
0070	63 61 6c 68 6f 73 74 3a	38 30 38 30 0d 0a 43 6f	calhost: 8080..Co
0080	6e 6e 65 63 74 69 6f 6e	3a 20 6b 65 65 70 2d 61	nnection : keep-a
0090	6c 69 76 65 0d 0a 41 63	63 65 70 74 3a 20 69 6d	live..Ac cept: im
00a0	61 67 65 2f 70 6e 67 2c	69 6d 61 67 65 2f 73 76	age/png, image/sv
00b0	67 2b 78 6d 6c 2c 69 6d	61 67 65 2f 2a 3b 71 3d	g+xml,im age/*;q=
00c0	30 2e 38 2c 76 69 64 65	6f 2f 2a 3b 71 3d 30 2e	0.8,vide o/*;q=0.
00d0	38 2c 2a 2f 2a 3b 71 3d	30 2e 35 0d 0a 49 6d 2d	8,*/*;q= 0.5..If-
00e0	4d 6f 64 69 66 69 65 64	2d 53 69 6e 63 65 3a 20	Modified -Since:
00f0	57 65 64 2c 20 31 30 20	4e 6f 76 20 32 30 32 31	Wed, 10 Nov 2021
0100	20 31 30 3a 33 37 3a 35	31 20 47 4d 54 0d 0a 55	10:37:5 1 GMT..U
0110	73 65 72 2d 41 67 65 6e	74 3a 20 4d 6f 7a 69 6c	ser-Agen t: Mozil
0120	6c 61 2f 35 2e 30 20 28	4d 61 63 69 6e 74 6f 73	la/5.0 (Macintos
0130	68 3b 20 49 6e 74 65 6c	20 4d 61 63 20 4f 53 20	h; Intel Mac OS
0140	58 20 31 30 5f 31 34 5f	36 29 20 41 70 70 6e 65	X 10.14. 6) Apple
0150	57 65 62 4b 69 74 2f 36	30 35 2e 31 2e 31 35 20	WebKit/6 05.1.15
0160	28 4b 48 54 4d 4c 2c 20	6c 69 6b 65 20 47 65 63	(KHTML, like Gec
0170	6b 6f 29 20 56 65 72 73	69 6f 6e 2f 31 34 2e 31	ko) Vers ion/14.1
0180	2e 32 20 53 61 66 61 72	69 2f 36 30 35 2e 31 2e	.2 Safar i/605.1.
0190	31 35 0d 0a 41 63 63 65	70 74 2d 4c 61 6e 67 75	15..Acce pt-Langu
01a0	61 67 65 3a 20 7a 68 2d	63 6e 0d 0a 52 65 66 65	age: zh- cn..Refe
01b0	72 65 72 3a 20 68 74 74	70 3a 2f 2f 6c 6f 63 61	rer: htt p://loca

Help Close

点开get请求，可以看出有关http协议的相关内容

GET / HTTP/1.1\r\n 称为请求行，由三个部分组成：请求方法、URI、HTTP协议版本，他们之间用空格分隔。该部分位于数据首行，基本格式为：该部分的请求方法字段（GET）给出了请求类型，URI给出请求的资源位置），最后HTTP协议版本给出HTTP的版本号。

接下来的部分称为请求头部，其紧跟着请求行，该部分主要是用于描述请求正文，主要是用于说明请求源、连接类型、以及一些Cookie信息等。

每一行都以\r\n结尾。

- 返回数据

```
▶ Frame 31: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface lo0, id 0
▶ Null/Loopback
▶ Internet Protocol Version 6, Src: ::1, Dst: ::1
▶ Transmission Control Protocol, Src Port: 8080, Dst Port: 51912, Seq: 474, Ack: 357, Len: 5
▶ [2 Reassembled TCP Segments (478 bytes): #29(473), #31(5)]
▼ Hypertext Transfer Protocol
  ▶ HTTP/1.1 200 \r\n
    Content-Type: text/html;charset=UTF-8\r\n
    Content-Language: zh-CN\r\n
    Transfer-Encoding: chunked\r\n
    Date: Wed, 10 Nov 2021 11:04:04 GMT\r\n
    Keep-Alive: timeout=60\r\n
    Connection: keep-alive\r\n
    \r\n
    [HTTP response 1/3]
    [Time since request: 0.003846000 seconds]
    [Request in frame: 23]
    [Next request in frame: 37]
    [Next response in frame: 39]
    [Request URI: http://localhost:8080/]
  ▶ HTTP chunked response
    File Data: 272 bytes
  ▼ Line-based text data: text/html (13 lines)
    <!DOCTYPE html>\n
    <html lang="en">\n
    <head>\n
      <meta charset="UTF-8">\n
      <title>hello</title>\n
    </head>\n
    <body>\n
      <h2>专业：计算机科学与技术</h2>\n
      <h2>姓名：朱浩泽</h2>\n
      <h2>学号：1911530</h2>\n
      \n
    </body>\n
    </html>\n

0000  3c 21 44 4f 43 54 59 50 45 20 68 74 6d 6c 3e 0a <!DOCTYPE E html>
0010  3c 68 74 6d 6c 20 6c 61 6e 67 3d 22 65 6e 22 3e <html la ng="en">
0020  0a 3c 68 65 61 64 3e 0a 20 20 20 3c 6d 65 74 <head>
0030  61 20 69 68 61 72 73 65 74 3d 22 55 54 46 2d 38 a charse t="UTF-8
0040  22 3e 0a 20 20 20 3c 74 69 74 6c 65 3e 68 65 "> < title>he

Frame (81 bytes) Reassembled TCP (478 bytes) De-chunked entity body (272 bytes)
```

```
▼ Hypertext Transfer Protocol
  ▶ HTTP/1.1 200 \r\n
    Content-Type: text/html;charset=UTF-8\r\n
    Content-Language: zh-CN\r\n
    Transfer-Encoding: chunked\r\n
    Date: Wed, 10 Nov 2021 11:04:04 GMT\r\n
    Keep-Alive: timeout=60\r\n
    Connection: keep-alive\r\n
    \r\n
    [HTTP response 1/3]
    [Time since request: 0.003846000 seconds]
    [Request in frame: 23]
    [Next request in frame: 37]
    [Next response in frame: 39]

0000  48 54 54 50 2f 31 2e 31 20 32 30 30 20 0d 0a 43 HTTP/1.1 200 ..C
0010  6f 6e 74 65 6e 74 2d 54 79 70 65 3a 20 74 65 78 ontent-T ype: tex
0020  74 2f 68 74 6d 6c 3b 63 68 61 72 73 65 74 3d 55 t/html;c harset=U
0030  54 46 2d 38 0d 0a 43 6f 6e 74 65 6e 74 2d 4c 61 TF-8 "Co ntent-La
0040  6e 67 75 61 67 65 3a 20 7a 68 2d 43 4e 0d 0a 54 nguage: zh-CN-.T
0050  72 61 6e 73 66 65 72 2d 45 6e 63 6f 64 69 6e 67 ransfer- Encoding
0060  3a 20 63 68 75 6e 6b 65 64 0d 0a 44 61 74 65 3a : chunke d .Date:
0070  20 57 65 64 2c 20 31 30 20 4e 6f 76 20 32 30 32 Wed, 10 Nov 202
0080  31 20 31 31 3a 30 34 3a 30 34 20 47 4d 54 0d 0a 1 11:04: 04 GMT..
0090  4b 65 65 70 2d 41 6c 69 76 65 3a 20 74 69 6d 68 Keep-Ali ve: time
00a0  6f 75 74 3d 30 0d 0a 43 6f 6e 6e 65 63 74 69 out=60..n Connecti
00b0  6f 6e 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a on: keep -alive..
00c0  0d 0a 31 31 30 0d 0a 3c 21 44 4f 43 54 59 50 45 ..110..< !DOCTYPE
00d0  20 68 74 6d 6c 3e 0a 3c 68 74 6d 6c 20 6c 61 6e html> < html lan
00e0  67 3d 22 65 6e 22 3e 0a 3c 68 65 61 64 3e 0a 20 g="en"> <head>
00f0  20 20 20 3c 6d 65 74 61 20 63 68 61 72 73 65 74 <meta charse
0100  3d 22 55 54 46 2d 38 22 3e 0a 20 20 20 3c 74 = "UTF-8" > <t
0110  69 74 6e 65 3e 6b 65 6c 61 6c 61 3c 2f 74 69 74 6c itle=hel loc/<titl
0120  65 3e 0a 3c 2f 68 65 61 64 3e 0a 3c 62 6f 64 79 e> </hea d> <body
0130  3e 0a 3c 68 32 3e e4 b8 93 e4 b8 9a ef bc 9a e8 > <h2> ..
0140  ae a1 e7 ae 97 e6 9c ba e7 a7 91 e5 ad a6 e4 b8 .....
0150  8e e6 8a 80 e6 9c af 3c 2f 68 32 3e 0a 3c 68 32 ..... </h2> <h2
0160  3e e5 a7 93 e5 90 8d ef bc 9a e6 9c b1 e6 b5 a9 .....
0170  e6 b3 bd 3c 2f 68 32 3e 0a 3c 68 32 3e e5 ad a6 ..... </h2> <h2>...
0180  e5 8f b7 ef bc 9c 21 39 31 31 35 33 30 3c 2f 68 ..... 19 11530</h
0190  32 3e 0a 3c 69 6d 67 20 73 72 63 3d 22 7e 2e 2f 2> <img src=
01a0  69 6d 67 2f 6c 6f 67 6f 2e 70 6e 67 22 20 61 6c img/logo .png" al
01b0  74 3d 22 e5 9b be e7 89 87 e6 9c aa e5 8a a0 e8 t=".....
01c0  bd bd 22 20 2f 3e 0a 3c 2f 62 6f 64 79 3e 0a 3c .." /> < /body> <
01d0  2f 68 74 6d 6c 3e 0a 0d 0a 30 0d 0a 0d 0a </html> .. 0....

Frame (81 bytes) Reassembled TCP (478 bytes) De-chunked entity body (272 bytes)
```

```

▶ Frame 35: 313 bytes on wire (2504 bits), 313 bytes captured (2504 bits) on interface lo0, id 0
▶ Null/Loopback
▶ Internet Protocol Version 6, Src: ::1, Dst: ::1
▶ Transmission Control Protocol, Src Port: 8080, Dst Port: 51913, Seq: 1, Ack: 420, Len: 237
▼ Hypertext Transfer Protocol
  ▶ HTTP/1.1 304 \r\n
    Vary: Origin\r\n
    Vary: Access-Control-Request-Method\r\n
    Vary: Access-Control-Request-Headers\r\n
    Last-Modified: Wed, 10 Nov 2021 10:37:51 GMT\r\n
    Date: Wed, 10 Nov 2021 11:04:04 GMT\r\n
    Keep-Alive: timeout=60\r\n
    Connection: keep-alive\r\n
    \r\n
    [HTTP response 1/1]
    [Time since request: 0.009837000 seconds]
    [Request in frame: 33]
    [Request URI: http://localhost:8080/img/logo.png]

```

```

0000 1a 00 00 00 60 0c 21 5d 01 0d 06 40 00 00 00 00  ....!|...@...
0010 00 00 00 00 00 00 00 00 00 00 01 00 00 00 00  ....
0020 00 00 00 00 00 00 00 00 00 00 00 01 1f 90 ca c9  ....
0030 02 0f 9f 75 b8 3d 0d a2 80 18 18 dd 01 15 00 00  ..u=-.....
0040 01 01 08 0a 2b 0c 82 87 2b 0c 82 7e 48 54 54 50  ....+...+...HTTP
0050 2f 31 2e 31 20 33 30 34 20 0d 0a 56 61 72 79 3a  /1.1 304 ...Vary:
0060 20 4f 72 69 67 69 6e 0d 0a 56 61 72 79 3a 20 41  Origin...Vary: A
0070 63 63 65 73 73 2d 43 6f 6e 74 72 6f 6c 2d 52 65  ccess-Co ntrol-Re
0080 71 75 65 73 74 2d 4d 65 74 68 6f 64 0d 0a 56 61  quest-Me thod-Va
0090 72 79 3a 20 41 63 63 65 73 73 2d 43 6f 6e 74 72  ry: Acce ss-Contr
00a0 6f 6c 2d 52 65 71 75 65 73 74 2d 48 65 61 64 65  ol-Reque st-Heade
00b0 72 73 0d 0a 4c 61 73 74 2d 4d 6f 64 69 66 69 65  rs..Last -Modifie
00c0 64 3a 20 57 65 64 2c 20 31 30 20 4e 6f 76 20 32  d: Wed, 10 Nov 2
00d0 30 32 31 20 31 30 3a 33 37 3a 35 31 20 47 4d 54  021 10:3 7:51 GMT
00e0 0d 0a 44 61 74 65 3a 20 57 65 64 2c 20 31 30 20  ..Date: Wed, 10
00f0 4e 6f 76 20 32 30 32 31 20 31 31 3a 30 34 3a 30  Nov 2021 11:04:0
0100 34 20 47 4d 54 0d 0a 4b 65 65 70 2d 41 6c 69 76  4 GMT. K eep-Aliv
0110 65 3a 20 74 69 6d 65 6f 75 74 3d 36 30 0d 0a 43  e: timeo ut=60..C
0120 6f 6e 6e 65 63 74 69 6f 6e 3a 20 6b 65 65 70 2d  onnectio n: keep-
0130 61 6c 69 76 65 0d 0a 0d 0a

```

响应报文由状态行、响应头部、空行和响应体4个部分构成。

首先是状态行，主要给出响应HTTP协议的版本号、响应返回状态码、响应描述；响应头部主要是返回一些服务器的基本信息，以及一些Cookie值等；响应体为所请求的数据（可以为html文档内容或图片等多种格式数据）。

TCP相关数据包

TCP协议提供的是按序、可靠的服务，是一种面向连接的传输方式，即其发送数据之前发送方和接收放需要握手，断开链接时需要四次挥手。

TCP的三次握手

- TCP三次握手过程
 1. 第一次握手： 客户端发送syn包(seq=x)到服务器，并进入SYN_SEND状态，等待服务器确认
 2. 第二次握手： 服务器收到syn包，必须确认客户的SYN(ack=x+1)，同时自己也发送一个SYN包(seq=y)，即SYN+ACK包，此时服务器进入SYN_RECV状态
 3. 第三次握手： 客户端收到服务器的SYN+ACK包，向服务器发送确认包ACK(ack=y+1)，此包发送完毕，客户端和服务端进入ESTABLISHED状态，完成三次握手

握手过程中传送的包里不包含数据，三次握手完毕后，客户端与服务器才正式开始传送数据。理想状态下，TCP连接一旦建立，在通信双方中的任何一方主动关闭连接之前，TCP连接都将被一直保持下去。

No.	Time	Source	Destination	Protocol	Length	Info
19	3.536277	::1	::1	TCP	88	51912 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=16324 WS=64 TSval=722240109 TSecr=0 SACK_PERM=1
20	3.536345	::1	::1	TCP	88	8080 → 51912 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=16324 WS=64 TSval=722240109 TSecr=722240109 SACK_PERM=1
21	3.536360	::1	::1	TCP	76	51912 → 8080 [ACK] Seq=1 Ack=1 Win=407744 Len=0 TSval=722240109 TSecr=722240109

- 如上图所示，即为tcp的三次握手过程。首先由51912向8080端口发送了一条syn包，并令 $seq = x = 0$ （本机初始序列号），报文长度为0，滑动窗口为65535，最大窗口长度为163245，窗口扩大因子为64，此时浏览进入了SYN_SEND状态，这是第一次握手；然后服务器8080端口接收到了浏览器发送的syn包后，确认客户的syn，使 $ack = x + 1 = 1$ ， $seq = y = 0$ ，对序列号为1（ack）之前的报文进行确认，同时向客户发送一个（syn，ack）包，其中报文长度为0，滑动窗口为65535，最大窗口长度为163245，这是第二次握手；客户端接受到了服务器发送的（syn，ack）包后，客户端向服务器返回确认包，其中 $ack = y + 1 = 1$ ，滑动窗口为407744，报文长度为0，这是第三次握手。至此三次握手完成，服务器和客户端建立了联系。

TCP的四次挥手

- TCP四次挥手过程
 1. 第一次挥手： Client发送一个FIN，用来关闭Client到Server的数据传送，Client进入FIN_WAIT_1状态。
 2. 第二次挥手： Server收到FIN后，发送一个ACK给Client，确认序号为收到序号+1（与SYN相同，一个FIN占用一个序号），Server进入CLOSE_WAIT状态。
 3. 第三次挥手： Server发送一个FIN，用来关闭Server到Client的数据传送，Server进入LAST_ACK状态。
 4. 第四次挥手： Client收到FIN后，Client进入TIME_WAIT状态，接着发送一个ACK给Server，确认序号为收到序号+1，Server进入CLOSED状态，完成四次挥手。

109	34.532624	::1	::1	TCP	76	51913 → 8080 [FIN, ACK] Seq=420 Ack=238 Win=407552 Len=0 TSval=722271005 TSecr=722240135
110	34.532689	::1	::1	TCP	76	8080 → 51913 [ACK] Seq=238 Ack=421 Win=407360 Len=0 TSval=722271005 TSecr=722271005
111	34.533143	::1	::1	TCP	76	8080 → 51913 [FIN, ACK] Seq=238 Ack=421 Win=407360 Len=0 TSval=722271006 TSecr=722271005
112	34.533200	::1	::1	TCP	76	51913 → 8080 [ACK] Seq=421 Ack=239 Win=407552 Len=0 TSval=722271006 TSecr=722271006
175	64.533607	::1	::1	TCP	76	51912 → 8080 [FIN, ACK] Seq=1132 Ack=1194 Win=406592 Len=0 TSval=722300968 TSecr=722242043
176	64.533691	::1	::1	TCP	76	8080 → 51912 [ACK] Seq=1194 Ack=1133 Win=406656 Len=0 TSval=722300968 TSecr=722300968
177	64.534144	::1	::1	TCP	76	8080 → 51912 [FIN, ACK] Seq=1194 Ack=1133 Win=406656 Len=0 TSval=722300969 TSecr=722300968
178	64.534200	::1	::1	TCP	76	51912 → 8080 [ACK] Seq=1133 Ack=1195 Win=406592 Len=0 TSval=722300969 TSecr=722300969

- 如上图所示，即为tcp四次挥手过程（由于浏览器开启了多线程，我们只对一个线程做分析）。首先，51912向8080端口发送了一条请求结束的报文（fin，ack），报文序列号为 $seq = x = 1132$ ， $ack = y = 1194$ ，报文长度为0，这是第一次挥手；然后服务器8080端口接收到了浏览器发送的fin包后，对收到的1194号前的报文进行确认，其报文序列号为 $seq = y = 1194$ ， $ack = x + 1 = 1132 + 1 = 1133$ ，这是第二次挥手；然后服务器（8080端口）再

向浏览器（51912）发送一个fin包，其内容与前一个基本相同分，用来关闭服务器端到客户端的数据传送，这是第三次挥手；客户端对收到的fin包进行确认，其报文序列号为 $seq = y = 1133$ ， $ack = 1194 + 1 = 1195$ ，客户端进入关闭状态，这是第四次挥手。至此tcp的四次挥手结束，客户端和服务端断开连接，访问结束。