

Jan 29, 16 9:17

udp-client.c

Page 1/2

```

1  /* CS480, Spring 2016
2  *
3  *   Simple UDP client, to demonstrate use of the sockets API
4  *   Compile with:
5  *       gcc -Wall -o udp-client udp-client.c
6  *   or
7  *       gcc -g -Wall -o udp-client udp-client.c # to support debugging
8  */
9
10 #include <stdio.h>
11 #include <stdlib.h>
12 #include <sys/socket.h>
13 #include <netinet/in.h>
14 #include <string.h>
15 #include <assert.h>
16 #include <arpa/inet.h>
17 #include <netdb.h>
18 #include <limits.h>
19
20 void handle_error(const char* s)
21 {
22     perror(s);
23     exit(1);
24 }
25
26 int main(int argc, char** argv)
27 {
28     int sock_fd;
29     struct sockaddr_in addr;
30     struct hostent* hostentp;
31     char* endptr;
32     unsigned int portl;
33     unsigned short port;
34     size_t num_to_send;
35     size_t num_sent;
36
37     if (argc != 4) {
38         fprintf(stderr, "usage: %s <hostname> <port> <message>\n", argv[0]);
39         exit(1);
40     }
41
42     /* convert from string to int */
43     portl = strtoul(argv[2], &endptr, 10);
44     if (endptr == NULL || portl == 0)
45         handle_error("strtoul");
46
47     assert(portl < USHRT_MAX);
48     port = (unsigned short)portl;
49
50     /*
51     * Below, we use the C idiom for "assign to a variable
52     * and then check its value"
53     */
54     if (!(hostentp = gethostbyname(argv[1]))) {
55         perror("gethostbyname");
56         exit(1);
57     }
58
59     /*
60     * Note! This code is not the same as what you need to do in lab1.
61     */
62     if ((sock_fd = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
63         handle_error("socket");
64
65     memset(&addr, 0, sizeof(addr));
66     addr.sin_family = AF_INET;
67     memcpy(&addr.sin_addr.s_addr,
68          hostentp->h_addr_list[0],
69          sizeof(struct in_addr));
70     addr.sin_port = htons(port);
71
72     printf("I am about to send %s to IP address %s and port %d\n",
73          argv[3], inet_ntoa(addr.sin_addr), atoi(argv[2]));

```

Jan 29, 16 9:17

udp-client.c

Page 2/2

```

74     num_to_send = strlen(argv[3]);
75
76     num_sent = sendto(sock_fd, /* socket */
77                    argv[3], /* buffer to send */
78                    num_to_send, /* number of bytes to send */
79                    0, /* flags=0: bare-bones use case*/
80                    (const struct sockaddr*)&addr, /* the destination */
81                    sizeof(addr)); /* size of the destination struct */
82
83
84     /*
85     * this is for education/demo purposes (there's a simpler way to
86     * write this error-checking code.) sendto() "commits" to sending
87     * the whole packet or nothing, so if less than the whole thing was
88     * sent, it better have been because there was an error (indicated
89     * by returning < 0).
90     */
91     if (num_sent != num_to_send) {
92         assert(num_sent < 0);
93         handle_error("sendto");
94     }
95
96     printf("I just sent the bytes!\n");
97
98     exit(0);
99 }

```

Jan 29, 16 9:17

udp-server.c

Page 1/2

```

1  /* CS480, Spring 2016
2  *
3  *   Simple UDP server, to demonstrate use of the sockets API
4  *   Compile with:
5  *       gcc -Wall -o udp-server udp-server.c
6  *   or
7  *       gcc -g -Wall -o udp-server udp-server.c # for debugging support
8  */
9
10 #include <stdio.h>
11 #include <stdlib.h>
12 #include <sys/types.h>
13 #include <sys/socket.h>
14 #include <netinet/in.h>
15 #include <string.h>
16 #include <assert.h>
17 #include <arpa/inet.h>
18 #include <netdb.h>
19 #include <limits.h>
20
21 void handle_error(const char* s)
22 {
23     perror(s);
24     exit(1);
25 }
26
27 int main(int argc, char** argv)
28 {
29     int sock_fd;
30     struct sockaddr_in my_addr, my_peer_addr;
31     char* endptr;
32     unsigned int portl;
33     unsigned short port;
34     int num_received;
35     char msg[1024];
36     socklen_t addrlen;
37
38     if (argc != 2) {
39         fprintf(stderr, "usage: %s <port>\n", argv[0]);
40         exit(1);
41     }
42
43     /* convert from string to int */
44     portl = strtoul(argv[1], &endptr, 10);
45     if (endptr == NULL || portl == 0)
46         handle_error("strtoul");
47
48     assert(portl < USHRT_MAX);
49     port = (unsigned short)portl;
50
51     /*
52     * Note! This code is not the same as what you need to do in lab1.
53     */
54     if ((sock_fd = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
55         handle_error("socket");
56
57     memset(&my_addr, 0, sizeof(my_addr));
58     my_addr.sin_family = AF_INET;
59     my_addr.sin_addr.s_addr = INADDR_ANY;
60     my_addr.sin_port = htons(port);
61
62     if (bind(sock_fd,
63             (struct sockaddr*)&my_addr,
64             sizeof(struct sockaddr_in)) < 0)
65         handle_error("bind");
66
67     while (1) {
68
69         addrlen = sizeof(struct sockaddr_in);
70
71         if ((num_received = recvfrom(sock_fd,
72                                    msg,
73                                    sizeof(msg),

```

Jan 29, 16 9:17

udp-server.c

Page 2/2

```

74         0,
75         /* who's sending */
76         (struct sockaddr*)&my_peer_addr,
77         /* length of buffer to receive peer info */
78         &addrlen) < 0)
79
80         handle_error("recvfrom");
81
82         assert(addrlen == sizeof(struct sockaddr_in));
83
84         printf("I got message %.*s from host %s, src port %d\n",
85               num_received, msg,
86               inet_ntoa(my_peer_addr.sin_addr),
87               ntohs(my_peer_addr.sin_port));
88
89     }
90
91     exit(0);
92 }

```