

ΠΠΣ
Εργαστηριακές Σημειώσεις - Υποδοχές

ΕΡΓΑΣΤΗΡΙΟ Λ.Σ.

Εαρινό 2017

1 Το αρχείο server.c

Listing 1: file.c

```
1  #include "stdio.h"
2  #include "stdlib.h"
3  #include "sys/types.h"
4  #include "sys/socket.h"
5  #include <netinet/in.h>
6  #include <arpa/inet.h>
7  #include <unistd.h>
8  #include "string.h"
9  #include "netinet/in.h"
10 #include "pthread.h"
11
12 #define PORT 4444 //change this
13 #define BUF_SIZE 2000
14 #define CLADDR_LEN 100
15
16 void * receiveMessage(void * socket) {
17     int sockfd, ret;
18     char buffer[BUF_SIZE];
19     sockfd = *((int*)socket);
20     memset(buffer, 0, BUF_SIZE);
21     for (;;) {
22         ret = recvfrom(sockfd, buffer, BUF_SIZE, 0, NULL, NULL);
23         if (ret < 0) {
24             printf("Error receiving data!\n");
25         } else {
26             printf("client: ");
27             fputs(buffer, stdout);
28             //printf("\n");
29         }
30     }
31 }
32
33 int main() {
34     struct sockaddr_in addr, cl_addr;
35     int sockfd, len, ret, newsockfd;
36     int yes=1;
37     char buffer[BUF_SIZE];
38     char clientAddr[CLADDR_LEN];
39     pthread_t rThread;
40
41     sockfd = socket(AF_INET, SOCK_STREAM, 0); //create tcp socket
42     if (sockfd < 0) {
43         printf("Error creating socket!\n");
44         exit(1);
45     }
46     //Permits multiple sockets to be bound to an identical socket
    address
47     if (setsockopt(sockfd, SOL_SOCKET, SO_REUSEADDR, &yes, sizeof
        (int)) == -1) {
48         perror("setsockopt");
```

```

49     exit(1);
50 }
51 printf("Socket created...\n");
52
53 memset(&addr, 0, sizeof(addr));
54 addr.sin_family = AF_INET;
55 addr.sin_addr.s_addr = INADDR_ANY;
56 addr.sin_port = PORT;
57
58 ret = bind(sockfd, (struct sockaddr *) &addr, sizeof(addr));
59 if (ret < 0) {
60     printf("Error binding!\n");
61     exit(1);
62 }
63 printf("Binding done...\n");
64
65 printf("Waiting for a connection...\n");
66 listen(sockfd, 5);
67
68
69 len = sizeof(struct sockaddr_in);
70 newsockfd = accept(sockfd, (struct sockaddr*)&cl_addr, (
71     socklen_t*)&len);
72 if (newsockfd < 0) {
73     printf("Error accepting connection!\n");
74     exit(1);
75 }
76 //converts the network address structure into a character
77     string
78 inet_ntop(AF_INET, &(cl_addr.sin_addr), clientAddr,
79     CLADDR_LEN);
80 printf("Connection accepted from %s...\n", clientAddr);
81
82 memset(buffer, 0, BUF_SIZE);
83 printf("Enter your messages one by one and press return key!\n
84     n");
85
86 //creating a new thread for receiving messages from the client
87 ret = pthread_create(&rThread, NULL, receiveMessage, &
88     newsockfd);
89 if (ret) {
90     printf("ERROR: Return Code from pthread_create() is %d\n",
91         ret);
92     exit(1);
93 }
94
95 while (fgets(buffer, BUF_SIZE, stdin) != NULL) {
96     ret = sendto(newsockfd, buffer, BUF_SIZE, 0, (struct
97         sockaddr *) &cl_addr, len);
98     if (ret < 0) {
99         printf("Error sending data!\n");
100         exit(1);
101     }
102 }

```

```
96
97     close(newsockfd);
98     close(sockfd);
99
100     pthread_exit(NULL);
101     return 0;
102 }
```

2 Το αρχείο client.c

Listing 2: file.c

```
1  #include "stdio.h"
2  #include "stdlib.h"
3  #include "sys/types.h"
4  #include <sys/socket.h>
5  #include <netinet/in.h>
6  #include <arpa/inet.h>
7  #include <unistd.h>
8  #include "string.h"
9  #include "netinet/in.h"
10 #include "netdb.h"
11 #include "pthread.h"
12
13 #define PORT 4444 //change this
14 #define BUF_SIZE 2000
15
16 void * receiveMessage(void * socket) {
17     int sockfd, ret;
18     char buffer[BUF_SIZE];
19     sockfd = *((int*)socket);
20     memset(buffer, 0, BUF_SIZE);
21     for (;;) {
22         ret = recvfrom(sockfd, buffer, BUF_SIZE, 0, NULL, NULL);
23         if (ret < 0) {
24             printf("Error receiving data!\n");
25         } else {
26             printf("server: ");
27             fputs(buffer, stdout);
28         }
29     }
30 }
31
32 int main(int argc, char**argv) {
33     struct sockaddr_in addr;
34     int sockfd, ret;
35     char buffer[BUF_SIZE];
36     char * serverAddr;
37     pthread_t rThread;
38
39     if (argc < 2) {
40         printf("usage: client < ip address >\n");
41         exit(1);
42     }
43
44     serverAddr = argv[1];
45
46     sockfd = socket(AF_INET, SOCK_STREAM, 0);
47     if (sockfd < 0) {
48         printf("Error creating socket!\n");
49         exit(1);
50     }
```

```

51  printf("Socket created...\n");
52
53  memset(&addr, 0, sizeof(addr));
54  addr.sin_family = AF_INET;
55  addr.sin_addr.s_addr = inet_addr(serverAddr);
56  addr.sin_port = PORT;
57
58  ret = connect(sockfd, (struct sockaddr *) &addr, sizeof(addr))
    ;
59  if (ret < 0) {
60      printf("Error connecting to the server!\n");
61      exit(1);
62  }
63  printf("Connected to the server...\n");
64
65  memset(buffer, 0, BUF_SIZE);
66  printf("Enter your messages one by one and press return key!\n
    ");
67
68  //creating a new thread for receiving messages from the server
69  ret = pthread_create(&rThread, NULL, receiveMessage, &sockfd);
70  if (ret) {
71      printf("ERROR: Return Code from pthread_create() is %d\n",
        ret);
72      exit(1);
73  }
74
75  while (fgets(buffer, BUF_SIZE, stdin) != NULL) {
76      ret = sendto(sockfd, buffer, BUF_SIZE, 0, (struct sockaddr *)
        &addr, sizeof(addr));
77      if (ret < 0) {
78          printf("Error sending data!\n\t-%s", buffer);
79      }
80  }
81
82  close(sockfd);
83  pthread_exit(NULL);
84
85  return 0;
86  }

```