OS/NW 과제-11

단일 서버 멀티 클라이언트 데이터 반환 (입출력 다중화)

32184074 컴퓨터공학과 정민준 - 2022년 12월 12일

본문

<echo_client_multiplexing.c 코드>

```c
#include <sys/socket.h>  /* 소켓 관련 함수 */
#include <arpa/inet.h>   /* 소켓 지원을 위한 각종 함수 */
#include <sys/stat.h>
#include <stdio.h>       /* 표준 입출력 관련 */
#include <string.h>      /* 문자열 관련 */
#include <unistd.h>      /* 각종 시스템 함수 */
#include <stdlib.h>
#include "time.h"

#define MAXLINE    1024

struct send_data{
    char str[MAXLINE];
};
struct rev_data{
    char str[MAXLINE];
};

int main(int argc, char **argv)
{
    struct sockaddr_in serveraddr;
    int server_sockfd;
    int client_len;
    char buf[MAXLINE];
    struct rev_data rev_msg;    //서버에 보내는 데이터
    struct send_data send_msg;  //서버에서 받는 데이터
    int maxfd = 0;
    char temp_buf[MAXLINE];

    fd_set temps, reads;
    int result;
    struct timeval tv;

    if ((server_sockfd = socket(AF_INET, SOCK_STREAM, 0)) == -1)
    {
        perror("error :");
        return 1;
    }

    /* 연결요청할 서버의 주소와 포트번호 프로토콜등을 지정한다. */
    server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
    serveraddr.sin_family = AF_INET;
    serveraddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    serveraddr.sin_port = htons(3600);

    client_len = sizeof(serveraddr);

```

<echo_client_multiplexing.c - 1>

```
47
48      /* 서버에 연결을 시도한다. */
49      if (connect(server_sockfd, (struct sockaddr *)&serveraddr, client_len)
50      {
51          perror("connect error :");
52          return 1;
53      }
54
55      FD_ZERO(&reads);
56      FD_SET(server_sockfd, &reads);
57      FD_SET(0,&reads);
58      maxfd = server_sockfd;
59
60      while(1)
61      {
62          temps = reads;
63          tv.tv_sec = 4;
64          tv.tv_usec = 0;
65
66          result = select(maxfd + 1, &temps, 0, 0, &tv);
67
68          if(result == -1){
69              printf("error \n");
70              return 1;
71          }else if(result ==0){
72              continue;
73          }else{
74
75              if(FD_ISSET(0,&temps)){
76
77                  memset(buf, 0x00, MAXLINE);
78                  read(0, buf, MAXLINE);     /* 키보드 입력을 기다린다. */
79                  if(strncmp(buf, "quit\n",5) == 0)
80                      break;
81                  memset(rev_msg.str, 0x00, MAXLINE);
82                  memset(&rev_msg, 0x00, sizeof(struct rev_data));
83                  char *ptr = strtok(buf," ");
84                  strcpy(rev_msg.str,ptr);
85
86                  struct tm newtime;
87                  time_t ltime;
88                  char time_str[100];
89                  ltime=time(&ltime);
90                  localtime_r(&ltime, &newtime);
91                  asctime_r(&newtime, time_str);
92
```

<echo_client_multiplexing.c - 2>

```
92
93                  for(int i=0; rev_msg.str[i] != 0; i++)
94                  {
95                      if(rev_msg.str[i] == '\n')
96                      {
97                          rev_msg.str[i] = 0;
98                          break;
99                      }
100                 }
101                 strcat(rev_msg.str," ");
102                 strcat(rev_msg.str,time_str);
103
104                 if (write(server_sockfd, &rev_msg, sizeof(rev_msg)) <= 0) /
105                 {
106                     perror("write error : ");
107                     return 1;
108                 }
109                 memset(buf, 0x00, MAXLINE);
110                 FD_CLR(0,&temps);
111             }
112
113             if(FD_ISSET(server_sockfd,&temps)){
114                 /* 서버로 부터 데이터를 읽는다. */
115                 if (read(server_sockfd, &send_msg, sizeof(send_msg)) <= 0)
116                 {
117                     perror("read error : ");
118                     return 1;
119                 }
120                 printf("read : %s\n", send_msg.str);
121                 sleep(2);
122             }
123         }
124     }
125     close(server_sockfd);
126     return 0;
127 }
128
```

<echo_client_multiplexing.c - 3>

<echo_server_multiplexing.c 코드>

```c
#include <sys/time.h>
#include <sys/types.h>
#include <unistd.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <stdio.h>
#include <string.h>
#include <stdbool.h>
#define MAXLINE 1024
#define PORTNUM 3600

struct send_data{
    char str[MAXLINE];
};
struct rev_data{
    char str[MAXLINE];
};

int main(int argc, char **argv)
{
    int listen_fd, client_fd;
    socklen_t addrlen;
    int fd_num;
    int maxfd = 0;
    int sockfd;
    int i= 0;
    char buf[MAXLINE];
    fd_set readfds, allfds;

    struct rev_data rev_msg;
    //char send_msg[MAXLINE];
    struct send_data send_msg;
    struct timeval tv;

    bool flag = false;

    struct sockaddr_in server_addr, client_addr;

    if((listen_fd = socket(AF_INET, SOCK_STREAM, 0)) == -1)
    {
        perror("socket error");
        return 1;
    }
    memset((void *)&server_addr, 0x00, sizeof(server_addr));
    server_addr.sin_family = AF_INET;
    server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
    server_addr.sin_port = htons(PORTNUM);

```

<echo_server_multiplexing.c - 1>

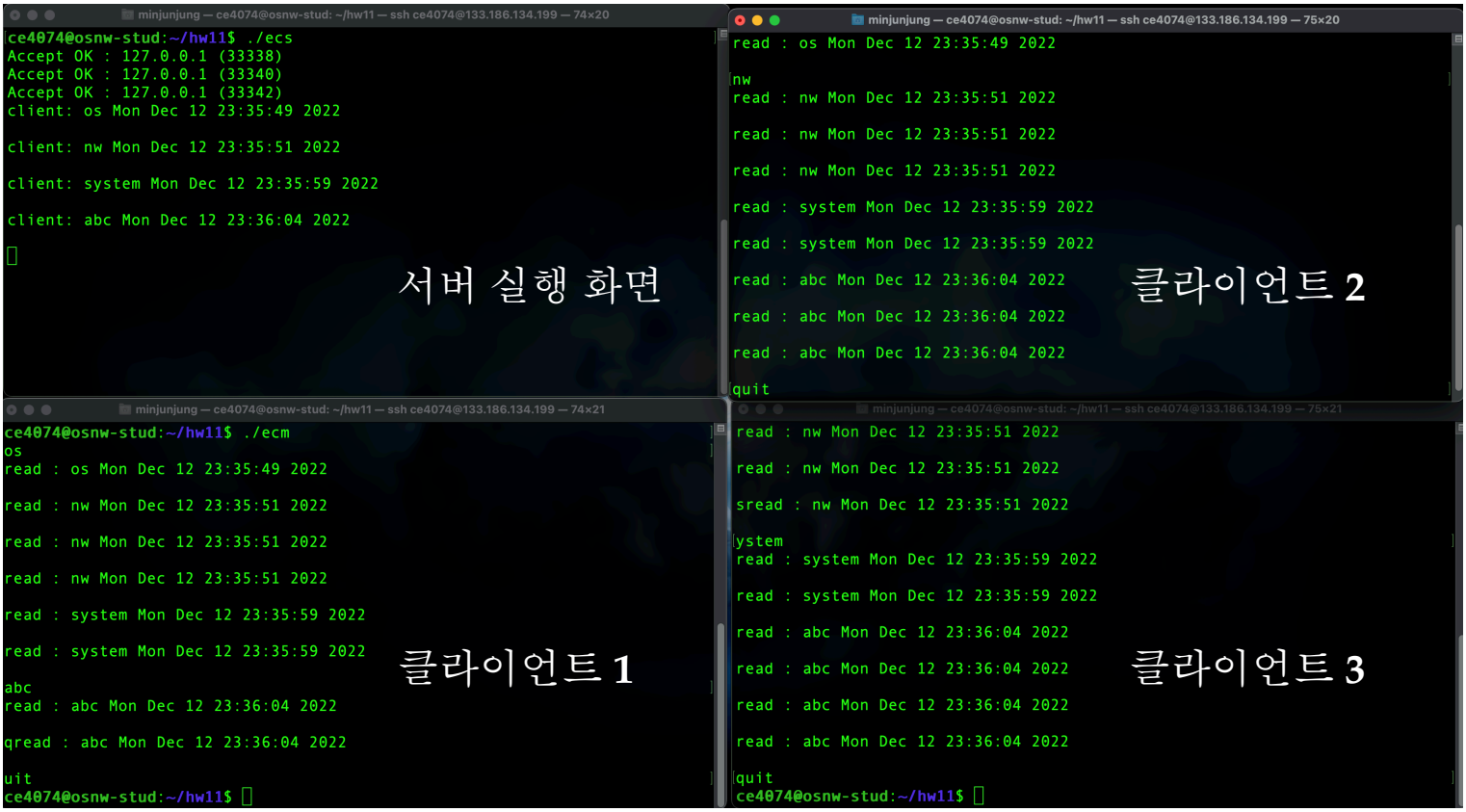
```
50
51      if(bind(listen_fd, (struct sockaddr *)&server_addr, sizeof(server_addr)
52      {
53          perror("bind error");
54          return 1;
55      }
56      if(listen(listen_fd, 5) == -1)
57      {
58          perror("listen error");
59          return 1;
60      }
61
62      FD_ZERO(&readfds);
63      FD_SET(listen_fd, &readfds);
64
65      maxfd = listen_fd;
66      while(1)
67      {
68          allfds = readfds;
69          if(flag){
70              for(i = 4 ; i <= maxfd ; i++){
71                  if(FD_ISSET(i, &allfds)){
72                      write(i,&send_msg,sizeof(send_msg));
73                  }
74              }
75          }
76          tv.tv_sec = 3;
77          tv.tv_usec = 0;
78
79          fd_num = select(maxfd + 1 , &allfds, (fd_set *)0,
80                          (fd_set *)0, &tv);
81
82          if (FD_ISSET(listen_fd, &allfds))
83          {
84              addrlen = sizeof(client_addr);
85              client_fd = accept(listen_fd,
86                                 (struct sockaddr *)&client_addr, &addrlen);
87
88              FD_SET(client_fd,&readfds);
89              if (client_fd > maxfd)
90                  maxfd = client_fd;
91              printf("Accept OK : %s (%d) \n",inet_ntoa(client_addr.sin_addr)
92              continue;
93          }
94
95          for (i = 0; i <= maxfd; i++)
96          {
97              sockfd = i;
98              if (FD_ISSET(sockfd, &allfds))
99              {
```

<echo_server_multiplexing.c - 2>

```
94
95          for (i = 0; i <= maxfd; i++)
96          {
97              sockfd = i;
98              if (FD_ISSET(sockfd, &allfds))
99              {
100                 memset(&rev_msg, 0x00, sizeof(struct rev_data));
101                 if (read(sockfd, &rev_msg, sizeof(rev_msg)) <= 0)
102                 {
103                     close(sockfd);
104                     FD_CLR(sockfd, &readfds);
105                 }
106                 else
107                 {
108                     if (strncmp(rev_msg.str, "quit\n", 5) ==0)
109                     {
110                         close(sockfd);
111                         FD_CLR(sockfd, &readfds);
112                     }else if(strlen(rev_msg.str)==0){
113                         continue;
114                     }
115                     else
116                     {   flag = true;
117                         printf("client: %s \n", rev_msg.str);
118                         strcpy(send_msg.str, rev_msg.str);
119                     }
120                 }
121                 if (--fd_num <= 0)
122                     break;
123             }
124         }
125     }
126 }
```

<echo_server_multiplexing.c - 3>

<결과화면>



1. 서버 실행 화면에서 각 클라이언트의 접속여부와 입력한 문장을 확인합니다.
2. 클라이언트 1,2,3을 서버에 접속시킵니다.
3. 클라이언트1 에서 ‘os’를 입력하면 클라이언트 1,2,3에 ‘os’와 클라이언트1 에서 ‘os'를 서버로 전송한 시각이 같이 출력됩니다.
4. 클라이언트2 에서 ‘nw’를 입력하면 클라이언트 1,2,3에 ‘nw’와 클라이언트1 에서 ‘nw’를 서버로 전송한 시각이 같이 출력됩니다.
5. 클라이언트3 에서 ‘system’를 입력하면 클라이언트 1,2,3에 ‘system’와 클라이언트1 에서 ‘system’를 서버로 전송한 시각이 같이 출력됩니다.
6. 클라이언트1 에서 다시 ‘abc’를 입력하면 클라이언트 1,2,3에 ‘abc’와 클라이언트 1 에서 ‘abc'를 서버로 전송한 시각이 같이 출력됩니다.
7. quit을 입력하면 클라이언트를 종료합니다.