
OS/NW 과제-8

단일 서버, 멀티 클라이언트(fork) & 문자열 합치기(pipe)

32184074 컴퓨터공학과 정민준 - 2022년 11월 14일



본문

<hw_server.c 코드>

```
lab06 hw_server.c
hw_server.c x hw_client.c x
1 #include ...
10
11 #define MAXLINE 1024
12 #define PORTNUM 3600
13
14 int main(int argc, char **argv)
15 {
16     int listen_fd;
17     int client_fd[3]; //클라이언트가 담길 배열 3개
18     pid_t pid;
19     socklen_t addrlen;
20     int readn;
21     int p_readn;
22     char buf[MAXLINE] = {};
23     struct sockaddr_in client_addr, server_addr;
24
25     char arr[MAXLINE] = {}; //합쳐질 문자가 담길 문자열 초기화
26     int cnt = 0; //클라이언트의 순서를 나타냄
27
28     int pp[2]; //pipe 통신
29     pipe(pp);
30
31
32     if( (listen_fd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
33     {
34         return 1;
35     }
36     memset((void *)&server_addr, 0x00, sizeof(server_addr));
37     server_addr.sin_family = AF_INET;
38     server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
39     server_addr.sin_port = htons(PORTNUM);
40
41     if(bind(listen_fd, (struct sockaddr *)&server_addr, sizeof(server_addr))
42     {
43         perror("bind error");
44         return 1;
45     }
46     if(listen(listen_fd, 5) == -1)
47     {
48         perror("listen error");
49         return 1;
50     }
51
52     signal(SIGCHLD, SIG_IGN);
53     while(1)
54     {
55         addrlen = sizeof(client_addr);
56         client_fd[cnt] = accept(listen_fd,
57                                (struct sockaddr *)&client_addr, &addrlen);
58         if(client_fd[cnt] == -1)
59         {
60             return 1;
61         }
62     }
63 }
```

<hw_server.c - 1>

```
lab06 hw_server.c
hw_server.c hw_client.c
54 {
55     addrlen = sizeof(client_addr);
56     client_fd[cnt] = accept(listen_fd,
57                             (struct sockaddr *)&client_addr, &addrlen);
58     if(client_fd[cnt] == -1)
59     {
60         printf("accept error\n");
61         break;
62     }
63     pid = fork();
64
65     //자식 프로세스
66     if(pid == 0)
67     {
68         close(listen_fd);
69         memset(buf, 0x00, MAXLINE);
70
71         //클라이언트로 부터 문자열(buf) 수신
72         while((readn = read(client_fd[cnt], buf, MAXLINE)) > 0)
73         {
74             printf("Read Data %s(%d) : %s",
75                   inet_ntoa(client_addr.sin_addr),
76                   client_addr.sin_port,
77                   buf);
78
79             //부모 프로세스에 문자열(buf) 전송
80             write(fd: pp[1], buf, nbytes: strlen(s: buf));
81             memset(buf, 0x00, MAXLINE);
82         }
83         return 0;
84     }
85
86     //부모 프로세스
87     else if(pid > 0)
88     {
89         //자식 프로세스로 부터 전송받은 값 수신
90         p_readn = read(pp[0], buf, MAXLINE);
91
92         //개행문자 제거
93         for(int i = 0 ; buf[i] != 0 ; i++){
94             if(buf[i] == '\n')
95             {
96                 buf[i] = 0;
97                 break;
98             }
99         }
100
101     }
102     //띄어쓰기 삽입 & 문자열 합치기
103     strcat(arr, " ");
104 }
```

<hw_server.c - 2>

```
lab06 / hw_server.c
hw_server.c x hw_client.c x

71 //클라이언트로 부터 문자열(buf) 수신
72 while((readn = read(client_fd[cnt], buf, MAXLINE)) > 0)
73 {
74     printf("Read Data %s(%d) : %s",
75           inet_ntoa(client_addr.sin_addr),
76           client_addr.sin_port,
77           buf);
78
79     //부모 프로세스에 문자열(buf) 전송
80     write(fd: pp[1], buf, nbyte: strlen(s: buf));
81     memset(buf, 0x00, MAXLINE);
82 }
83 return 0;
84 }
85
86 //부모 프로세스
87 else if( pid > 0)
88 {
89     //자식 프로세스로 부터 전송받은 값 수신
90     p_readn = read(pp[0], buf, MAXLINE);
91
92     //개행문자 제거
93     for(int i = 0 ; buf[i] != 0 ; i++){
94         if(buf[i] == '\n')
95         {
96             buf[i] = 0;
97             break;
98         }
99     }
100
101 }
102 //띄어쓰기 삽입 & 문자열 합치기
103 strcat(arr, " ");
104 strcat(arr, buf);
105
106 //클라이언트에게 전송
107 write(fd: client_fd[cnt], buf: arr, nbyte: sizeof(arr));
108 close(client_fd[cnt]);
109
110 //다음 클라이언트로 순서로 넘어가기 위해 cnt값 증가
111 cnt++;
112 }
113 close(listen_fd);
114 return 0;
115 }
```

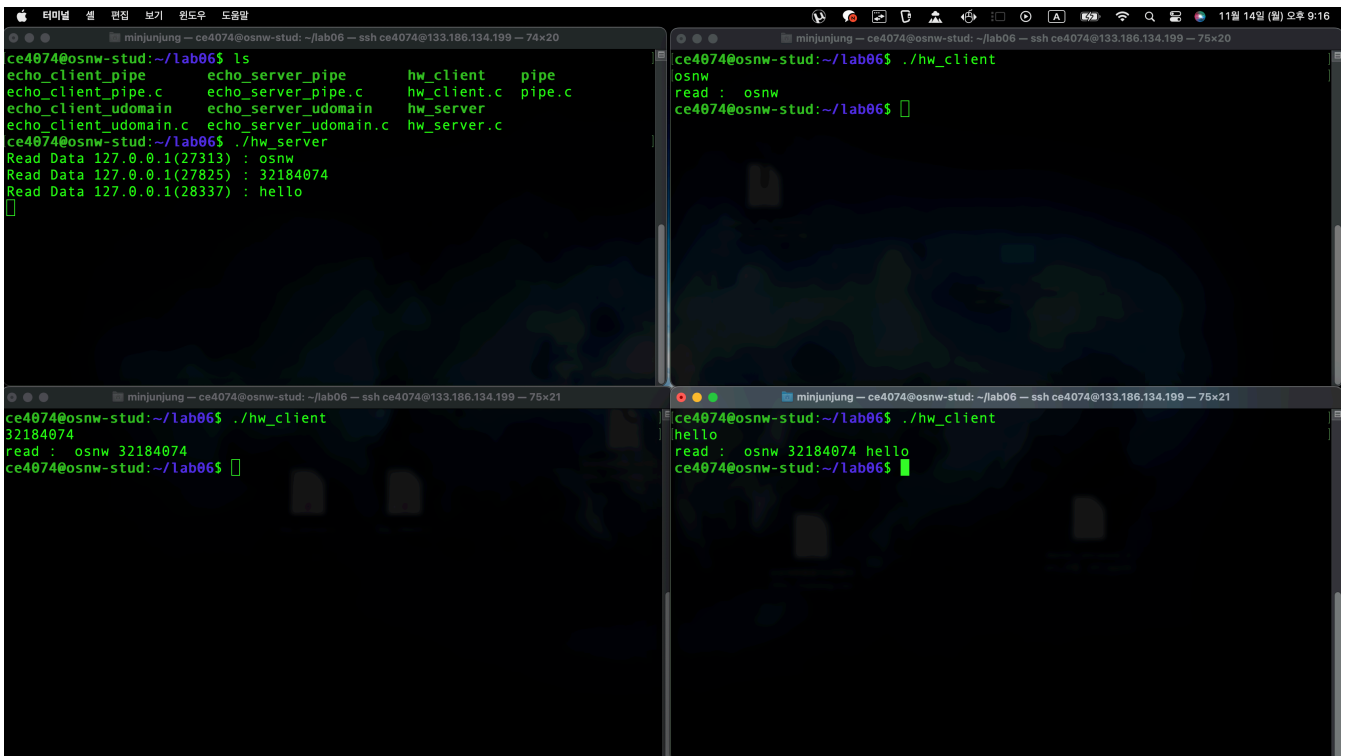
<hw_server.c - 3>

<hw_client.c 코드>

```
lab06 hw_client.c
hw_server.c x hw_client.c x
1 #include ...
7
8 #define MAXLINE 1024
9
10 int main(int argc, char **argv)
11 {
12     struct sockaddr_in serveraddr;
13     int server_sockfd;
14     int client_len;
15     char buf[MAXLINE];
16
17     if ((server_sockfd = socket(AF_INET, SOCK_STREAM, 0)) == -1)
18     {
19         perror("error :");
20         return 1;
21     }
22     /* 연결요청할 서버의 주소와 포트번호 프로토콜등을 지정한다. */
23     server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
24     serveraddr.sin_family = AF_INET;
25     serveraddr.sin_addr.s_addr = inet_addr("127.0.0.1");
26     serveraddr.sin_port = htons(3600);
27     client_len = sizeof(serveraddr);
28
29     /* 서버에 연결을 시도한다. */
30     if (connect(server_sockfd, (struct sockaddr *)&serveraddr,
31         client_len) == -1)
32     {
33         perror("connect error :");
34         return 1;
35     }
36     memset(buf, 0x00, MAXLINE);
37     read(0, buf, MAXLINE); /* 키보드 입력을 기다린다. */
38
39     /* 입력 받은 데이터를 서버로 전송한다. */
40     if (write(fd: server_sockfd, buf, nbyte: MAXLINE) <= 0)
41     {
42         perror("write error :");
43         return 1;
44     }
45     memset(buf, 0x00, MAXLINE);
46     /* 서버로 부터 데이터를 읽는다. */
47     if (read(server_sockfd, buf, MAXLINE) <= 0)
48     {
49         perror("read error :");
50         return 1;
51     }
52     printf("read : %s", buf);
53     printf("\n");
54     close(server_sockfd);
55     return 0;
56 }
```

<hw_client.c>

<결과 화면>



```
ce4074@osnw-stud:~/lab06$ ls
echo_client_pipe.c  echo_server_pipe.c  hw_client  pipe
echo_client_pipe.c  echo_server_pipe.c  hw_client.c  pipe.c
echo_client_udomain.c  echo_server_udomain.c  hw_server
ce4074@osnw-stud:~/lab06$ ./hw_server
Read Data 127.0.0.1(27313) : osnw
Read Data 127.0.0.1(27825) : 32184074
Read Data 127.0.0.1(28337) : hello
ce4074@osnw-stud:~/lab06$

ce4074@osnw-stud:~/lab06$ ./hw_client
osnw
read : osnw
ce4074@osnw-stud:~/lab06$

ce4074@osnw-stud:~/lab06$ ./hw_client
32184074
read : osnw 32184074
ce4074@osnw-stud:~/lab06$

ce4074@osnw-stud:~/lab06$ ./hw_client
hello
read : osnw 32184074 hello
ce4074@osnw-stud:~/lab06$
```

1. 3명의 클라이언트를 받아야 하기 때문에 정수형 크기가 3인 client_fd 배열을 선언합니다.
2. 클라이언트로 부터 입력받은 값을 자식 프로세스에서 부모 프로세스로 pipe 통신으로 보내줍니다.
3. 부모 프로세스는 strcat을 통해서 문자열 arr을 이어 붙이면서 반환할 새로운 문자열을 만듭니다.
4. 클라이언트는 전달받은 문자열 arr을 출력합니다.