

Nov 30, 20 22:15

echo_server_multi.c

Page 1/3

```

#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>

#define MAXLINE 1024
#define PORTNUM 3600
#define SOCK_SETSIZE 1021

int client_index = 0;
int stack[3];
int top = -1;

struct data
{
    char buf[MAXLINE];
    int num;
};

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;

    fd_set readfds, allfds;

    struct data rdata;
    int num = 0;
    char buf[MAXLINE] = "";

    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);

    if(bind(listen_fd, (struct sockaddr *)&server_addr, sizeof(server_addr))
    == -1)
    {
        perror("bind error");
        return 1;
    }
    if(listen(listen_fd, 5) == -1)
    {
        perror("listen error");
        return 1;
    }

    FD_ZERO(&readfds);
    FD_SET(listen_fd, &readfds);

```

Nov 30, 20 22:15

echo_server_multi.c

Page 2/3

```

    maxfd = listen_fd;
    while(1)
    {
        allfds = readfds;
        printf("Select Wait %d\n", maxfd);
        fd_num = select(maxfd + 1, &allfds, (fd_set *)0, (fd_set *)0, N
ULL);

        if (FD_ISSET(listen_fd, &allfds))
        {
            addrlen = sizeof(client_addr);
            client_fd = accept(listen_fd,
                               (struct sockaddr *)&client_addr, &addrlen);

            FD_SET(client_fd, &readfds);

            if (client_fd > maxfd)
                maxfd = client_fd;
            printf("Accept OK\n");
            continue;
        }

        if (client_index == 3)
        {
            close(client_fd);
            break;
        }

        for (i = 0; i <= maxfd; i++)
        {
            sockfd = i;
            if (FD_ISSET(sockfd, &allfds))
            {
                stack[++top] = sockfd;

                if (read(sockfd, (char*)&rdata, sizeof(rdata)) <
= 0)
                {
                    close(sockfd);
                    FD_CLR(sockfd, &readfds);
                }
                else
                {
                    if (strcmp(rdata.buf, "quit\n", 5) == 0)
                    {
                        close(sockfd);
                        FD_CLR(sockfd, &readfds);
                    }
                    else
                    {
                        printf("Read: %s and %d\n", rdata.b
uf, ntohl(rdata.num));

                        num += ntohl(rdata.num);
                        strcat(buf, rdata.buf);
                        client_index++;
                    }
                }

                if (--fd_num <= 0)
                    break;
            }
        }

        for( int j = 0; j < 3; j++)
        {

```

Nov 30, 20 22:15

echo_server_multi.c

Page 3/3

```
        rdata.num = htonl(num);  
        strcpy(rdata.buf, buf);  
        write(stack[top--], (char*)&rdata, sizeof(rdata));  
    }  
}
```