

Mar 19, 14 4:17

server.c

Page 1/1

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <strings.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#define SERVER_PORT 10101 /* modified to avoid well-known ports restriction */
#define MAX_PENDING 5
#define MAX_LINE 256

int main()
{
    struct sockaddr_in sin;
    char buf[MAX_LINE];
    int len;
    int s, new_s;
    /* build address data structure */
    bzero((char *)&sin, sizeof(sin));
    sin.sin_family = AF_INET;
    sin.sin_addr.s_addr = INADDR_ANY;
    sin.sin_port = htons(SERVER_PORT);

    /* setup passive open */
    if ((s = socket(PF_INET, SOCK_STREAM, 0)) < 0) {
        perror("simplex-talk: socket");
        exit(1);
    }

    if ((bind(s, (struct sockaddr *)&sin, sizeof(sin))) < 0) {
        perror("simplex-talk: bind");
        exit(1);
    }
    listen(s, MAX_PENDING);
    /* wait for connection, then receive and print text */
    while(1) {
        if ((new_s = accept(s, (struct sockaddr *)&sin, &len)) < 0) {
            perror("simplex-talk: accept");
            exit(1);
        }
        while (len = recv(new_s, buf, sizeof(buf), 0)) {
            fputs(buf, stdout);
            send(new_s, buf, len, 0);      /* added line to send th
e received
line back to the client */
        }
        close(new_s);
    }
}

```

Mar 19, 14 4:17

client.c

Page 1/1

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <strings.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#define SERVER_PORT 10101 /* modified to avoid well-known ports restriction */
#define MAX_LINE 256

int main(int argc, char * argv[])
{
    FILE *fp;
    struct hostent *hp;
    struct sockaddr_in sin;
    char *host;
    char buf[MAX_LINE];
    int s;
    int len;
    if (argc==2) {
        host = argv[1];
    }
    else {
        fprintf(stderr, "usage: ./client host\n");
        exit(1);
    }

    /* translate host name into peerâM-^@M-^Ys IP address */
    hp = gethostbyname(host);
    if (!hp) {
        fprintf(stderr, "simplex-talk: unknown host: %s\n", host);
        exit(1);
    }
    /* build address data structure */
    bzero((char *)&sin, sizeof(sin));
    sin.sin_family = AF_INET;
    bcopy(hp->h_addr, (char *)&sin.sin_addr, hp->h_length);
    sin.sin_port = htons(SERVER_PORT);
    /* active open */
    if ((s = socket(PF_INET, SOCK_STREAM, 0)) < 0) {
        perror("simplex-talk: socket");
        exit(1);
    }
    if (connect(s, (struct sockaddr *)&sin, sizeof(sin)) < 0) {
        perror("simplex-talk: connect");
        close(s);
        exit(1);
    }
    /* main loop: get and send lines of text */
    while (fgets(buf, sizeof(buf), stdin)) {
        buf[MAX_LINE-1] = '\0';
        len = strlen(buf) + 1;
        send(s, buf, len, 0);
        /* loop added to wait for the echo of the line and print it */
        while (len = recv(s, buf, sizeof(buf), 0)) {
            fputs(buf, stdout);
            if (len) break;
        }
    }
}

```