Lab: 6

27/2/2020

CLIENT - SERVER USING UDP

Aim

To implement client-server communication using UDP.

Program

Server

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define PORT 8080
#define MAXLINE 1024
void main(){
   int sockfd;
   char buffer[MAXLINE];
   char hello[100];
   struct sockaddr_in servaddr, cliaddr;
   if((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
       perror("Socket Creation Failed!!");
       exit(EXIT_FAILURE);
   }
   memset(&servaddr, 0, sizeof(servaddr));
   memset(&cliaddr, 0, sizeof(cliaddr));
   servaddr.sin_family = AF_INET;
   servaddr.sin_addr.s_addr = INADDR_ANY;
   servaddr.sin_port = htons(PORT);
   if(bind(sockfd,(const struct sockaddr *)&servaddr, sizeof(
               servaddr))<0)
   {  perror("Bind Failed!!");
```

Scanned with CamScanner

```
        exit(EXIT_FAILURE);
    }
    while(1) {
        int len, n;
        len = sizeof(cliaddr);
        n = recvfrom(sockfd, (char *)buffer, MAXLINE, MSG_WAITALL,
            (struct sockaddr *) &cliaddr, &len);
        buffer[n] = '\0';
        if (strcmp(buffer, "exit", 4) == 0) {
            printf("Client Exiting...\n");
            close(sockfd);
            exit(0);
        }
        printf("From Client: \t %s \n", buffer);
        printf("To Client: \t");
        scanf("%[^\n]s", hello);
        getchar();
        if (strcmp(hello, "exit", 4) == 0) {
            printf("Server Exiting...\n");
            sendto(sockfd, (const char *)hello, strlen(hello), MSG_CONFIRM,
                (const struct sockaddr *) &cliaddr, len);
            close(sockfd);
            exit(0);
        }
        sendto(sockfd, (const char *) hello, strlen(hello), MSG_CONFIRM,
            (const struct sockaddr *) &cliaddr, len);
    }
}
```

Client

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define PORT 8080
#define MAXLINE 1024
```

Scanned with CamScanner

```
void main() {
    int sockfd;
    char buffer[MAXLINE];
    char hello[100];
    struct sockaddr_in servaddr;
    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror("Socket Creation Failed!!");
        exit(EXIT_FAILURE);
    }
    memset(&servaddr, 0, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_port = htons(PORT);
    servaddr.sin_addr.s_addr = INADDR_ANY;
    while(1) {
        int n, len;
        printf("To Server: \t");
        scanf("%[^\n]s", hello);
        getchar();
        if(strncmp(hello, "exit", 4)==0){
            printf("Client Exiting...\n");
            sendto(sockfd, (const char*) hello, strlen(hello), MSG_CONFIRM
              (const struct sockaddr *) &servaddr, sizeof(servaddr));
            close(sockfd);
            exit(0);
        }
        sendto(sockfd, (const char *) hello, strlen(hello), MSG_CONFIRM,
          (const struct sockaddr *) &servaddr, sizeof(servaddr));
        n=recvfrom(sockfd, (char *) buffer, MAXLINE, MSG_WAITALL,
          (struct sockaddr *) &servaddr, &len);
        buffer[n] = '\0';
        if (strncmp(buffer, "exit", 4) == 0) {
            printf("Server Exiting...\n");
            close(sockfd);
            exit(0);
        }
        printf("From Server:\t%s\n", buffer);
    }
```

Scanned with CamScanner

}

Result

Implemented client-server communication using UDP.

Scanned with CamScanner

Output

Server

From Client: Hi Server
To Client: Hi Client
Client Exiting...

Client

To Server: Hi Server
From Server: Hi Client
To Server: exit
Client Exiting...

Scanned with CamScanner