Aticleworld (https://aticleworld.com/)

(https://www.facebook.com/Aticleworld-602608163238897/)

(https://www.linkedin.com/in/amlendra-kumar-bb3b2096/)

(https://www.youtube.com/channel/UCAcrc6X4yzEjza9QXM7vg2A?sub_confirmation=1)

(https://twitter.com/aticleworld)

Home (https://aticleworld.com/)

C Tutorial (https://aticleworld.com/c-tutorial/)

C Programming Examples (https://aticleworld.com/c-programming/)

Interview Questions Video Courses

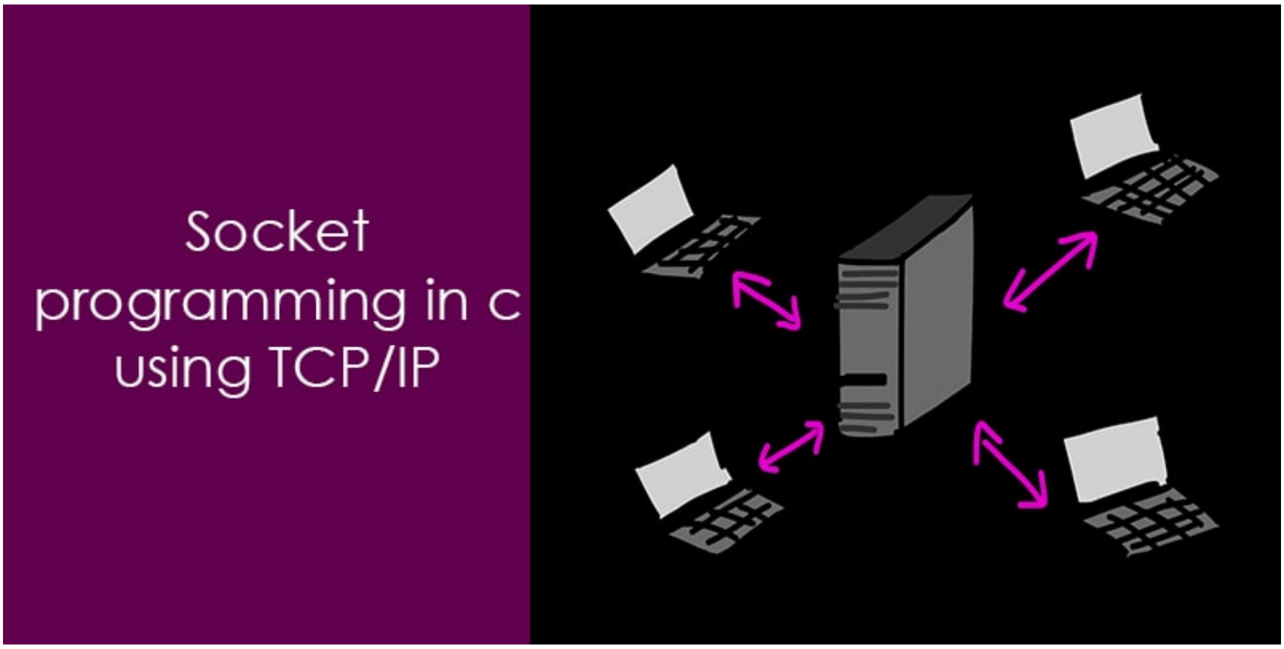
Tools (https://aticleworld.com/resources/)

Blog (https://aticleworld.com/blog-post/)

MCQ’s (https://aticleworld.com/mcq/)

,

Socket programming in c using TCP/IP



Before you start learning socket programming in c, you should basic knowledge of IP address, TCP, UDP. In this article, I shall describe TCP/IP and write a socket program using the TCP/IP API.

TCP (Transmission control protocol)

A TCP (transmission control protocol) is a connection-oriented communication. It is an intermediate layer of the application layer and internet protocol layer in the OSI model. TCP is designed to send the data packets over the network. It ensures that data is delivered to the correct destination.

TCP creates a connection between the source and destination node before transmitting the data and keeps the connection alive until the communication is active.

In TCP before sending the data it breaks the large data into smaller packets and cares the integrity of the data at the time of reassembling at the destination node. Major Internet applications such as the World Wide Web, email, remote administration, and file transfer rely on TCP.

TCP also offers the facility of retransmission, when a TCP client sends data to the server, it requires an acknowledgment in return. If an acknowledgment is not received, after a certain amount of time transmitted data will be loss and TCP automatically retransmits the data.

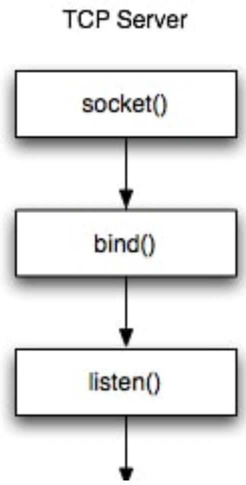
The communication over the network in TCP/IP model takes place in form of a client-server architecture. ie, the client begins the communication and establishes a connection with a server.

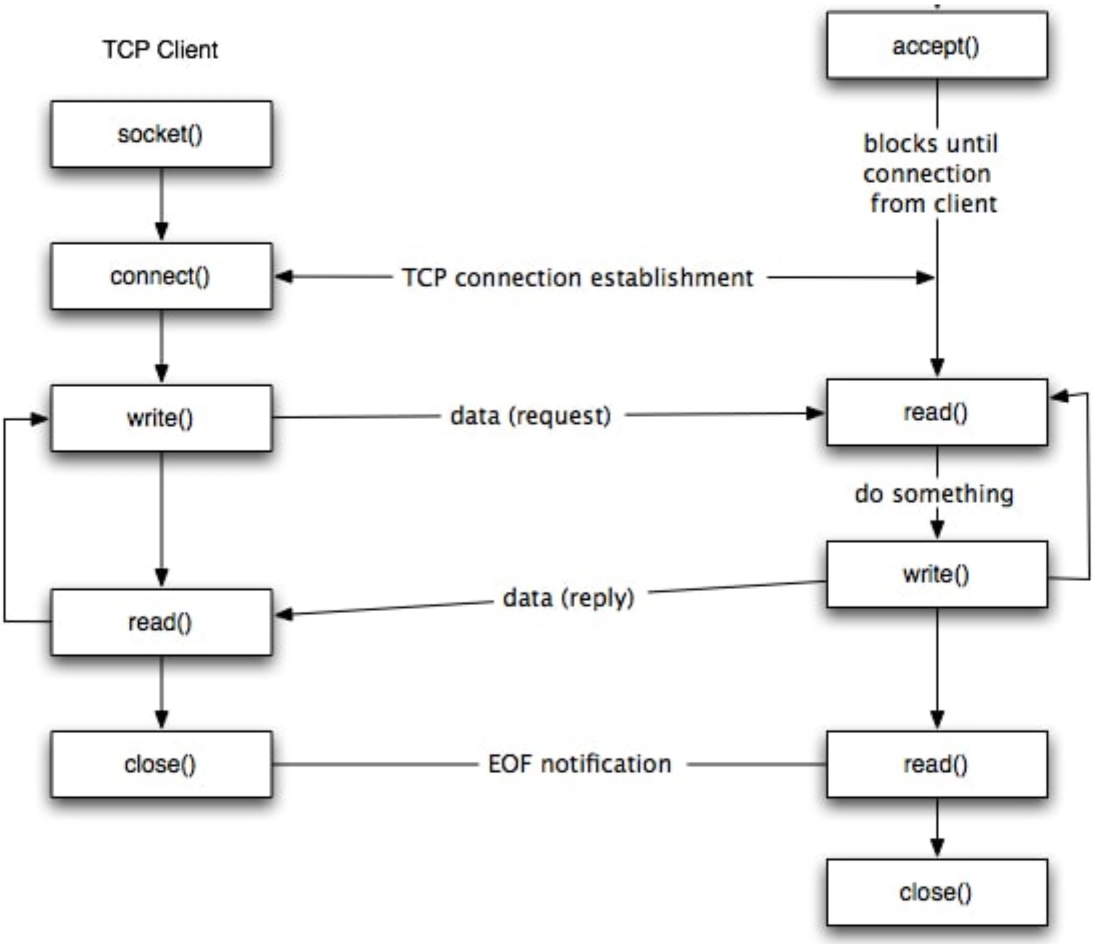
For more understanding let's create a server that continuously runs and establish the connection after getting a request from the client.

Note: here I am creating the server and client for Linux.

In this example, After the connection with a client, the server will wait for a message from the client. After getting the message server will check the received message and send a proper response as per the received message.

Sequence of function for client-server communication





Steps to create a client using TCP/IP API

- Create a socket using the socket() **function (https://aticleworld.com/function-pointer-in-c/)**in c.
- Initialize the socket address structure as per the server and connect the socket to the address of the server using the connect();
- Receive and send the data using the recv() and send() functions.
  Close the connection by calling the close() function.

Steps to create a server using TCP/IP API

- Create a socket using the socket() function in c.
- Initialize the socket address structure and bind the socket to an address using the bind() function.
- Listen for connections with the listen() function.
- Accept a connection with the accept() function system call. This call typically

blocks until a client connects to the server.

- Receive and send data by using the recv() and send() function in c.
- Close the connection by using the close() function.

If you are a beginner and want to learn TCP/IP, then you can check this course "TCP/IP Networking for Developers (https://pluralsight.pxf.io/c/1192901/424552/7490?u=https%3A%2F%2Fwww.pluralsight.com%2Fcourses%2Ftcp-ip-networking-for-devs)" that created by Steve Evans (https://www.linkedin.com/in/loudsteve/), and the rating of this course is around 4.7. The good thing is that FREE TRIAL is available and you can also access thousands of courses that are created by industry experts.

Today Grab your free Trial (https://pluralsight.pxf.io/c/1192901/424552/7490?u=https%3A%2F%2Fwww.pluralsight.com%2Fcourses%2Ftcp-ip-networking-for-devs)

Example of Socket programming in C using TCP/IP:

As we know in socket programming network nodes (sockets) are communicating with each other over the network. One socket(node) listens on a particular port at an IP, while the other socket reaches out to the other to form a connection. In this example code, we will create two-node, one node for the server and the other for the client. So let's see the example code for client-server socket programming in C.

Example Source code for TCP/IP client in C Linux:

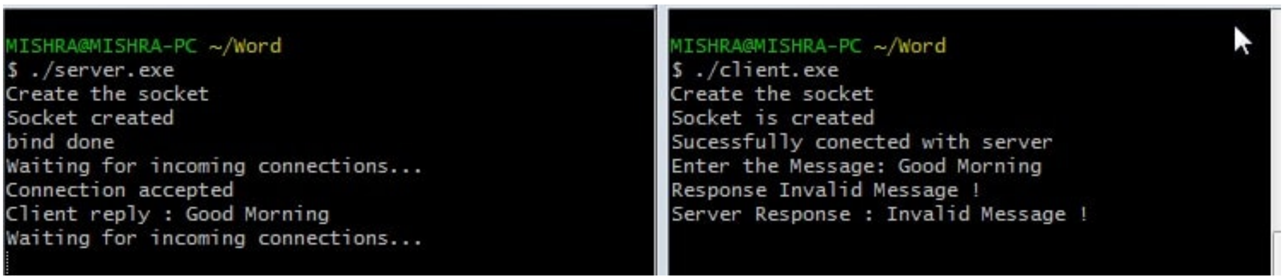
```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<sys/socket.h>
#include<arpa/inet.h>
#include<unistd.h>
```

```
//Create a Socket for server communication
short SocketCreate(void)
{
    short hSocket;
    printf("Create the socket\n");
    hSocket = socket(AF_INET, SOCK_STREAM, 0);
    return hSocket;
}

//try to connect with server
int SocketConnect(int hSocket)
{
    int iRetval=-1;
    int ServerPort = 90190;
    struct sockaddr_in remote= {0};

    remote.sin_addr.s_addr = inet_addr("127.0.0.1"); //Local Host
    remote.sin_family = AF_INET;
    remote.sin_port = htons(ServerPort);

    iRetval = connect(hSocket,(struct sockaddr *)&remote,sizeof(struct sockaddr_

    return iRetval;
}


// Send the data to the server and set the timeout of 20 seconds
int SocketSend(int hSocket,char* Rqst,short lenRqst)

{

    int shortRetval = -1;
    struct timeval tv;
    tv.tv_sec = 20;  /* 20 Secs Timeout */
    tv.tv_usec = 0;

    if(setsockopt(hSocket,SOL_SOCKET,SO_SNDTIMEO,(char *)&tv,sizeof(tv)) < 0)
    {
        printf("Time Out\n");
        return -1;
    }
    shortRetval = send(hSocket, Rqst, lenRqst, 0);

    return shortRetval;
}


//receive the data from the server
int SocketReceive(int hSocket,char* Rsp,short RvcSize)
{

    int shortRetval = -1;
    struct timeval tv;
    tv.tv_sec = 20;  /* 20 Secs Timeout */
    tv.tv_usec = 0;

    if(setsockopt(hSocket, SOL_SOCKET, SO_RCVTIMEO,(char *)&tv,sizeof(tv)) < 0)
    {
        printf("Time Out\n");
        return -1;
    }
    shortRetval = recv(hSocket, Rsp, RvcSize, 0);

    printf("Response %s\n",Rsp);

    return shortRetval;
}


//main driver program
int main(int argc, char *argv[])
{
    int hSocket, read_size;
    struct sockaddr_in server;
    char SendToServer[100] = {0};
    char server_reply[200] = {0};

    //Create socket
    hSocket = SocketCreate();
    if(hSocket == -1)
    {
        printf("Could not create socket\n");
        return 1;
    }

    printf("Socket is created\n");

    //Connect to remote server
    if (SocketConnect(hSocket) < 0)
    {
        perror("connect failed.\n");
        return 1;
    }

    printf("Sucessfully conected with server\n");

    printf("Enter the Message: ");
    gets(SendToServer);

    //Send data to the server
    SocketSend(hSocket, SendToServer, strlen(SendToServer));


    //Received the data from the server
    read_size = SocketReceive(hSocket, server_reply, 200);

    printf("Server Response : %s\n\n",server_reply);


    close(hSocket);
    shutdown(hSocket,0);
    shutdown(hSocket,1);
    shutdown(hSocket,2);
    return 0;
}
```

Example Source code for TCP/IP server in C Linux

```c
#include<stdio.h>
#include<string.h>
#include<sys/socket.h>
#include<arpa/inet.h>
#include<unistd.h>


short SocketCreate(void)
{
    short hSocket;
    printf("Create the socket\n");

    hSocket = socket(AF_INET, SOCK_STREAM, 0);
    return hSocket;
}


int BindCreatedSocket(int hSocket)
{
    int iRetval=-1;
    int ClientPort = 90190;
    struct sockaddr_in  remote= {0};

    /* Internet address family */
    remote.sin_family = AF_INET;
    /* Any incoming interface */
    remote.sin_addr.s_addr = htonl(INADDR_ANY);
    remote.sin_port = htons(ClientPort); /* Local port */
    iRetval = bind(hSocket,(struct sockaddr *)&remote,sizeof(remote));

    return iRetval;
}


int main(int argc, char *argv[])
{
    int socket_desc, sock, clientLen, read_size;
    struct sockaddr_in server, client;
    char client_message[200]= {0};
    char message[100] = {0};
    const char *pMessage = "hello aticleworld.com";

    //Create socket
    socket_desc = SocketCreate();
    if (socket_desc == -1)
    {
        printf("Could not create socket");
        return 1;
    }
    printf("Socket created\n");

    //Bind
    if( BindCreatedSocket(socket_desc) < 0)
    {
        //print the error message
        perror("bind failed.");
        return 1;
    }
    printf("bind done\n");

    //Listen
    listen(socket_desc, 3);

    //Accept and incoming connection

    while(1)
    {
        printf("Waiting for incoming connections...\n");
        clientLen = sizeof(struct sockaddr_in);

        //accept connection from an incoming client
        sock = accept(socket_desc,(struct sockaddr *)&client,(socklen_t*)&client
        if (sock < 0)
        {
            perror("accept failed");
            return 1;
        }
        printf("Connection accepted\n");

        memset(client_message, '\0', sizeof client_message);
        memset(message, '\0', sizeof message);
        //Receive a reply from the client
        if( recv(sock, client_message, 200, 0) < 0)
        {
            printf("recv failed");
            break;
        }

        printf("Client reply : %s\n",client_message);

        if(strcmp(pMessage,client_message)==0)
        {
            strcpy(message,"Hi there !");
        }
        else
        {
            strcpy(message,"Invalid Message !");
        }

        // Send some data
        if( send(sock, message, strlen(message), 0) < 0)
        {
            printf("Send failed");
            return 1;
        }

        close(sock);
        sleep(1);
    }
    return 0;
}
```

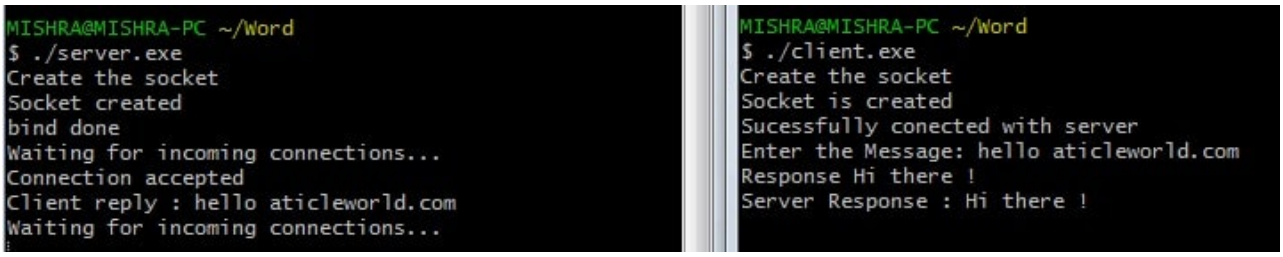
OutPut 1.



```
MISHRA@MISHRA-PC ~/Word
$ ./server.exe
Create the socket
Socket created
bind done
Waiting for incoming connections...
Connection accepted
Client reply : Good Morning
Waiting for incoming connections...
```

```
MISHRA@MISHRA-PC ~/Word
$ ./client.exe
Create the socket
Socket is created
Sucessfully conected with server
Enter the Message: Good Morning
Response Invalid Message !
Server Response : Invalid Message !
```

OutPut 2.



```
MISHRA@MISHRA-PC ~/Word
$ ./server.exe
Create the socket
Socket created
bind done
Waiting for incoming connections...
Connection accepted
Client reply : hello aticleworld.com
Waiting for incoming connections...
```

```
MISHRA@MISHRA-PC ~/Word
$ ./client.exe
Create the socket
Socket is created
Sucessfully conected with server
Enter the Message: hello aticleworld.com
Response Hi there !
Server Response : Hi there !
```

Recommended Post

- **Best Mouse for programmers. (https://aticleworld.com/best-mouse-for-programming-and-coding/)**
- **HDLC Protocol in C. (https://aticleworld.com/hdlc-protocol/)**
- **SSL programming in C. (https://aticleworld.com/ssl-server-client-using-openssl-in-c/)**
- **Socket programming in C (https://aticleworld.com/socket-programming-in-c-using-tcpip/)**.
- **Parse XML response in C without using the library. (https://aticleworld.com/parse-xml-response-in-c/)**
- **Create Http Get and Post request in C (https://aticleworld.com/http-get-and-post-methods-example-in-c/)**.
- **File handling in C (https://aticleworld.com/file-handling-in-c/)**.
- **I2C Communication protocol (https://aticleworld.com/i2c-bus-protocol-and-interface/)**.
- **Embedded C Interview Questions. (https://aticleworld.com/embedded-c-interview-questions-2/)**
- **Pointers in C. (https://aticleworld.com/pointers-in-c/)**

- CAN Protocol Interview Questions. (https://aticleworld.com/can-protocol-interview-questions-and-answers-in-detail/)
- Bit-wise interview Questions in C. (https://aticleworld.com/interview-questions-on-bitwise-operators-in-c/)
- Base64 encoding decoding online tool (https://mysuperblist.com/base64-encoding-and-decoding/)

About (https://aticleworld.com/author/pritosh/)

I am an embedded c software engineer and a corporate trainer, currently, I am working as senior software engineer in a largest Software consulting company . I have working experience of different microcontrollers (stm32, LPC, PIC AVR and 8051), drivers (USB and virtual com-port), POS device (VeriFone) and payment gateway (global and first data).

WEBSITE (HTTPS://ATICLEWORLD.COM/) TWITTER (HTTPS://TWITTER.COM/HTTPS://TWITTER.COM/ATICLEWORLD) FACEBOOK (HTTPS://WWW.FACEBOOK.COM/ATICLEWORLD-602608163238897/) LINKEDIN (HTTPS://WWW.LINKEDIN.COM/IN/AMLENDRA-KUMAR-BB3B2096/) INSTAGRAM (HTTPS://WWW.INSTAGRAM.COM/ATICLEWORLD/)

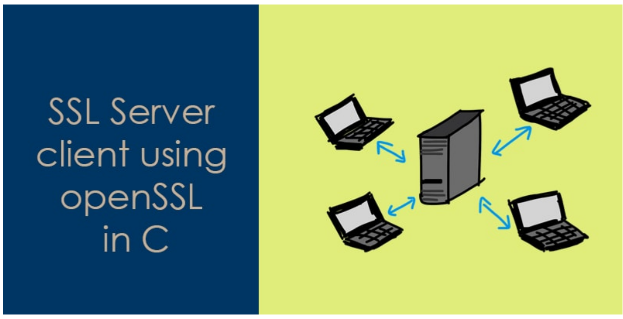
← PREVIOUS ARTICLE (HTTPS://ATICLEWORLD.COM/SEARCH-AN-ELEMENT-IN-A-LINKED-LIST-USING-ITERATIVE-AND-RECURSIVE/)

(https://aticleworld.com/search-an-element-in-a-linked-list-using-iterative-and-recursive/)

C Program to Search an element in a Linked List using Iterative and Recursive (https://aticleworld.com/search-an-element-in-a-linked-list-using-iterative-and-recursive/)

(HTTPS://ATICLEWORLD.COM/AUTHOR/PRITOSH/)/ (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/)

NEXT ARTICLE (HTTPS://ATICLEWORLD.COM/SSL-SERVER-CLIENT-USING-OPENSSL-IN-C/) →



(https://aticleworld.com/ssl-server-client-using-openssl-in-c/)

ssl server client programming using openssl in c (https://aticleworld.com/ssl-server-client-using-openssl-in-c/)

(HTTPS://ATICLEWORLD.COM/AUTHOR/PRITOSH/)/ (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/)

22 comments

Jenson

JUNE 15, 2017 AT 12:13 PM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-430)

You should not be using gets() as it can lead to buffer overflow (unsafe/security exploit). C11 has deprecated this API and replaced it with fgets().

REPLY

Amlendra

JUNE 15, 2017 AT 12:40 PM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-431)

Yes Agreed.

REPLY

Brugu Nadh

NOVEMBER 22, 2017 AT 10:00 PM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-737)

Mast

REPLY

ME (https://www.privacy.com)

DECEMBER 30, 2017 AT 6:19 AM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-746)

I get Output 1 (Invalid message) on client side.
How to get Output 2 (receive/display message fràm server)

REPLY

ME (https://www.privacy.com)

DECEMBER 30, 2017 AT 6:22 AM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-

TCPIP/#COMMENT-747)

OK, got it !
Client sends "hello aticleworld.com" and server replies.
All other messages terminate client application.
OK 🙂

REPLY

resmi

MARCH 21, 2018 AT 8:06 PM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-797)

I am programming Barcode Reader with B&R Automation Studio using TCP/IP. Can i use the same logic?

REPLY

venu

SEPTEMBER 3, 2018 AT 5:51 PM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-1114)

I am running socket server and client as a web browser.using local host(127.0.0.1) able to connect the server. unable to connect the server using with different local host ip(Ex:192.168.X.X).

REPLY

Lakshmi

JUNE 20, 2019 AT 1:13 PM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-2876)

How to implement non-blocking socket for the client side ?
I am using hercules as the server over windows and running the client code on linux

REPLY

Amlendra (https://aticleworld.com/)

JUNE 21, 2019 AT 8:40 AM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-2877) you can use fcntl with O_NONBLOCK

REPLY

rider321

JULY 24, 2019 AT 12:05 PM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-3097)

yes you can use that also

REPLY

Juan Mamani

SEPTEMBER 9, 2019 AT 12:42 PM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-3341)

Hi!
what about to use epoll? What is your opinion about?

REPLY

ahmed said

MARCH 12, 2020 AT 2:44 AM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-4212)

i am very grateful for you i have searched and tried so long to do what this code do

REPLY

Pingback: Difference Between High-level Data Link Control (HDLC) and Point-to-Point Protocol (PPP) - AticleWorld (https://aticleworld.com/difference-between-hdlc-and-ppp/)

Pingback: What is the difference between HTTP and HTTPS (HTTP vs HTTPS)? - AticleWorld (https://aticleworld.com/difference-between-http-and-https-http-vs-https/)

Pingback: Framing in Data Link Layer - AticleWorld (https://aticleworld.com/framing-in-data-link-layer/)

Pingback: Understanding Linear Regression - AticleWorld (https://aticleworld.com/understanding-linear-regression/)

Pingback: http get and post methods example in c - AticleWorld (https://aticleworld.com/http-get-and-post-methods-example-in-c/)

Pingback: Active, Reactive and Apparent Power - AticleWorld (https://aticleworld.com/active-reactive-and-apparent-power/)

Pingback: Difference between SSL and TLS (SSL vs TLS) - AticleWorld (https://aticleworld.com/difference-between-secure-socket-layer-ssl-and-transport-layer-security-tls/)

Pingback: Metaboly (https://blog.barakinnovations.in/2020/08/ssl-server-client-programming-using-openssl-in-c/)

daksh

AUGUST 5, 2021 AT 2:07 PM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-5868)

one of the most irritating classes of this semester is network programming…. never understood a word of it

REPLY

Amlendra (https://aticleworld.com/)

AUGUST 5, 2021 AT 6:59 PM (HTTPS://ATICLEWORLD.COM/SOCKET-PROGRAMMING-IN-C-USING-TCPIP/#COMMENT-5869)

Let me know where I can help.

REPLY

Leave a Reply

YOUR EMAIL ADDRESS WILL NOT BE PUBLISHED. REQUIRED FIELDS ARE MARKED *

COMMENT *

NAME *

EMAIL *

WEBSITE

□ SAVE MY NAME, EMAIL, AND WEBSITE IN THIS BROWSER FOR THE NEXT TIME I COMMENT.

POST COMMENT

(https://aticleworld.com/best-c-programming-books/)

Pages

- About (https://aticleworld.com/about/)
- Guest Article (https://aticleworld.com/guest-article/)
- Blog Posts (https://aticleworld.com/blog-post/)
- affiliate-disclosure (https://aticleworld.com/affiliate-disclosure/)
- disclaimer (https://aticleworld.com/disclaimer/)

BACK TO TOP ↑