**30. Creating the applications using TCP chat client and chat server in java/C.**

**Server:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <winsock2.h>

#include <ws2tcpip.h>

#pragma comment(lib, "ws2\_32.lib") // Link with Winsock library

#define PORT 8080

#define BUFFER\_SIZE 1024

int main() {

WSADATA wsa;

SOCKET server\_fd, new\_socket;

struct sockaddr\_in address;

int addrlen = sizeof(address);

char buffer[BUFFER\_SIZE] = {0};

// Initialize Winsock

if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0) {

printf("WSAStartup failed. Error Code: %d\n", WSAGetLastError());

return 1;

}

// Create socket

if ((server\_fd = socket(AF\_INET, SOCK\_STREAM, 0)) == INVALID\_SOCKET) {

printf("Socket creation failed. Error Code: %d\n", WSAGetLastError());

return 1;

}

// Configure server address

address.sin\_family = AF\_INET;

address.sin\_addr.s\_addr = INADDR\_ANY;

address.sin\_port = htons(PORT);

// Bind socket

if (bind(server\_fd, (struct sockaddr\*)&address, sizeof(address)) == SOCKET\_ERROR) {

printf("Bind failed. Error Code: %d\n", WSAGetLastError());

return 1;

}

// Listen for client

if (listen(server\_fd, 3) == SOCKET\_ERROR) {

printf("Listen failed. Error Code: %d\n", WSAGetLastError());

return 1;

}

printf("Server listening on port %d...\n", PORT);

// Accept client connection

if ((new\_socket = accept(server\_fd, (struct sockaddr\*)&address, &addrlen)) == INVALID\_SOCKET) {

printf("Accept failed. Error Code: %d\n", WSAGetLastError());

return 1;

}

printf("Client connected.\n");

while (1) {

memset(buffer, 0, BUFFER\_SIZE);

int valread = recv(new\_socket, buffer, BUFFER\_SIZE, 0);

if (valread <= 0) {

printf("Client disconnected.\n");

break;

}

printf("Client: %s", buffer);

printf("Server: ");

fgets(buffer, BUFFER\_SIZE, stdin);

send(new\_socket, buffer, strlen(buffer), 0);

}

closesocket(new\_socket);

closesocket(server\_fd);

WSACleanup();

return 0;

}

**Client:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <winsock2.h>

#include <ws2tcpip.h>

#pragma comment(lib, "ws2\_32.lib") // Link with Winsock library

#define PORT 8080

#define BUFFER\_SIZE 1024

int main() {

WSADATA wsa;

SOCKET sock;

struct sockaddr\_in serv\_addr;

char buffer[BUFFER\_SIZE] = {0};

// Initialize Winsock

if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0) {

printf("WSAStartup failed. Error Code: %d\n", WSAGetLastError());

return 1;

}

// Create socket

if ((sock = socket(AF\_INET, SOCK\_STREAM, 0)) == INVALID\_SOCKET) {

printf("Socket creation failed. Error Code: %d\n", WSAGetLastError());

return 1;

}

serv\_addr.sin\_family = AF\_INET;

serv\_addr.sin\_port = htons(PORT);

serv\_addr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

// Connect to server

if (connect(sock, (struct sockaddr\*)&serv\_addr, sizeof(serv\_addr)) < 0) {

printf("Connection Failed. Error Code: %d\n", WSAGetLastError());

return 1;

}

printf("Connected to server. Start chatting!\n");

while (1) {

printf("Client: ");

fgets(buffer, BUFFER\_SIZE, stdin);

send(sock, buffer, strlen(buffer), 0);

memset(buffer, 0, BUFFER\_SIZE);

int valread = recv(sock, buffer, BUFFER\_SIZE, 0);

if (valread <= 0) {

printf("Server disconnected.\n");

break;

}

printf("Server: %s", buffer);

}

closesocket(sock);

WSACleanup();

return 0;

}

A screenshot of a computer

AI-generated content may be incorrect.