**인터넷 프로토콜 HW3**

12161756 윤성호

**< 1. Server.cpp >**

#pragma comment(lib, "ws2\_32")

#include <winsock2.h>

#include <stdlib.h>

#include <stdio.h>

#define SERVERPORT 9000

#define BUFSIZE 512

void ErrorHandling(char\* message);

int main(int argc, char\* argv[]) {

int retval;

// 윈속 초기화

WSADATA wsa;

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

ErrorHandling((char\*)"WSAStartup() error!");

// socket()

SOCKET listen\_sock = socket(AF\_INET, SOCK\_STREAM, 0);

if (listen\_sock == INVALID\_SOCKET)

ErrorHandling((char\*)"socket() error");

// bind()

SOCKADDR\_IN serveraddr;

ZeroMemory(&serveraddr, sizeof(serveraddr));

serveraddr.sin\_family = AF\_INET;

serveraddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

serveraddr.sin\_port = htons(SERVERPORT);

retval = bind(listen\_sock, (SOCKADDR\*)&serveraddr, sizeof(serveraddr));

if (retval == SOCKET\_ERROR)

ErrorHandling((char\*)"bind() error");

// listen()

retval = listen(listen\_sock, SOMAXCONN);

if (retval == SOCKET\_ERROR)

ErrorHandling((char\*)"listen() error");

// 데이터 통신에 사용할 변수

SOCKET client\_sock;

SOCKADDR\_IN clientaddr;

int addrlen, len;

char buf[BUFSIZE + 1], yn;

while (1) {

// accept()

addrlen = sizeof(clientaddr);

client\_sock = accept(listen\_sock, (SOCKADDR\*)&clientaddr, &addrlen);

if (client\_sock == INVALID\_SOCKET)

ErrorHandling((char\*)"accept() error");

// 접속한 클라이언트 정보 출력

printf("\n[TCP 서버] 클라이언트 접속: IP 주소=%s, 포트 번호=%d\n",

inet\_ntoa(clientaddr.sin\_addr), ntohs(clientaddr.sin\_port));

// 클라이언트와 데이터 통신

while (1) {

// 데이터 받기

retval = recv(client\_sock, buf, BUFSIZE, 0);

if (retval == 0)

break;

// 받은 데이터 출력

buf[retval] = '\0';

printf(" [from Client] >> %s\n", buf);

// 데이터 입력

printf("[보낼 메시지] << ");

if (fgets(buf, BUFSIZE + 1, stdin) == NULL)

break;

// '\n' 문자 제거

len = strlen(buf);

if (buf[len - 1] == '\n')

buf[len - 1] = '\0';

if (strlen(buf) == 0)

break;

// 데이터 보내기

retval = send(client\_sock, buf, strlen(buf), 0);

}

// closesocket()

closesocket(client\_sock);

printf("\n[TCP 서버] 클라이언트 종료: IP 주소=%s, 포트 번호=%d\n",

inet\_ntoa(clientaddr.sin\_addr), ntohs(clientaddr.sin\_port));

printf("서버도 종료할까요? (y or n)\n");

scanf\_s("%c", &yn, sizeof(yn));

if (yn == 'y') break;

else continue;

}

// closesocket()

closesocket(listen\_sock);

// 윈속 종료

WSACleanup();

return 0;

}

void ErrorHandling(char\* message) {

fputs(message, stderr);

fputc('\n', stderr);

exit(1);

}

**< 2. Client.cpp >**

#pragma comment(lib, "ws2\_32")

#include <winsock2.h>

#include <stdlib.h>

#include <stdio.h>

#define SERVERIP "127.0.0.1"

#define SERVERPORT 9000

#define BUFSIZE 512

void ErrorHandling(char\* message);

int main(int argc, char\* argv[]) {

int retval;

// 윈속 초기화

WSADATA wsa;

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

ErrorHandling((char\*)"WSAStartup() error!");

// socket()

SOCKET sock = socket(AF\_INET, SOCK\_STREAM, 0);

if (sock == INVALID\_SOCKET)

ErrorHandling((char\*)"socket() error");

// connect()

SOCKADDR\_IN serveraddr;

ZeroMemory(&serveraddr, sizeof(serveraddr));

serveraddr.sin\_family = AF\_INET;

serveraddr.sin\_addr.s\_addr = inet\_addr(SERVERIP);

serveraddr.sin\_port = htons(SERVERPORT);

retval = connect(sock, (SOCKADDR\*)&serveraddr, sizeof(serveraddr));

if (retval == SOCKET\_ERROR)

ErrorHandling((char\*)"connect() error!");

// 데이터 통신에 사용할 변수

char buf[BUFSIZE + 1];

int len;

// 서버와 데이터 통신

while (1) {

// 데이터 입력

printf("\n[보낼 메시지] << ");

if (fgets(buf, BUFSIZE + 1, stdin) == NULL)

break;

// '\n' 문자 제거

len = strlen(buf);

if (buf[len - 1] == '\n')

buf[len - 1] = '\0';

if (strlen(buf) == 0)

break;

// 데이터 보내기

retval = send(sock, buf, strlen(buf), 0);

// 데이터 받기

retval = recv(sock, buf, BUFSIZE, 0);

if (retval == 0)

break;

// 받은 데이터 출력

buf[retval] = '\0';

printf(" [from Server] >> %s", buf);

}

// closesocket()

closesocket(sock);

// 윈속 종료

WSACleanup();

return 0;

}

void ErrorHandling(char\* message) {

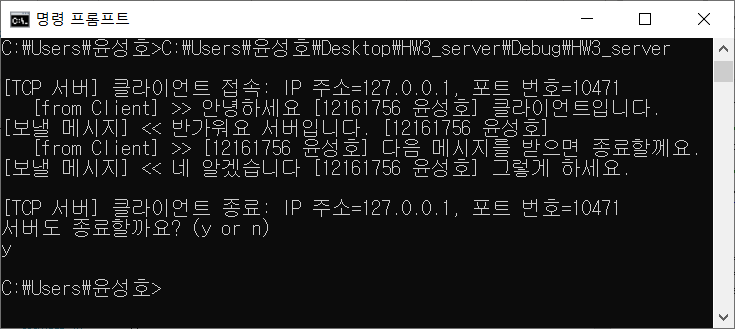
fputs(message, stderr);

fputc('\n', stderr);

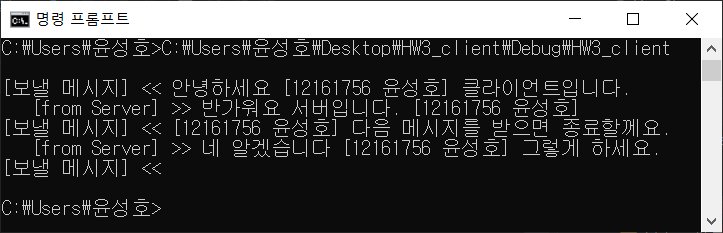
exit(1);

}

**< 3. 실행 화면 >**



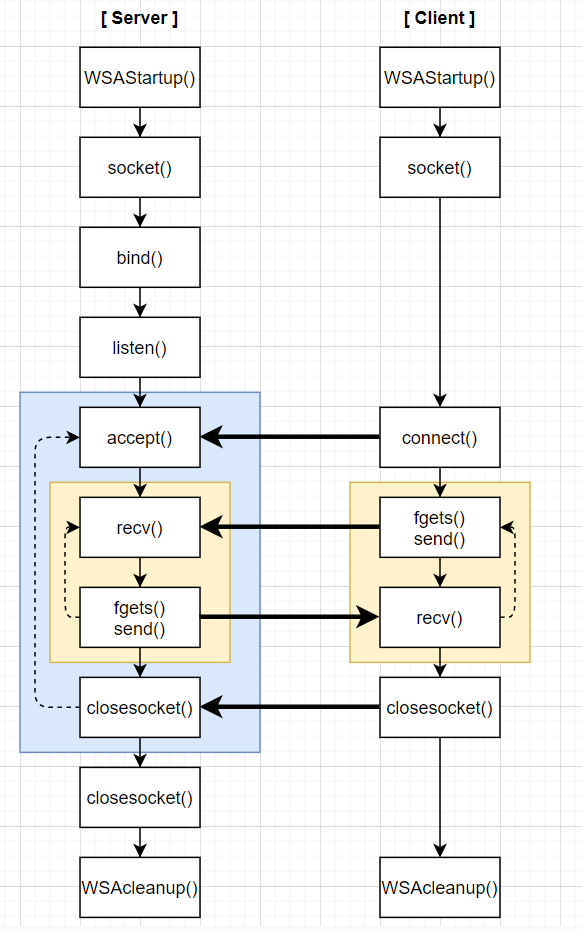
[그림 1. Server 실행 화면]

****

[그림 2. Client 실행화면]

< 4. 내용 정리 >

1) 작성한 채팅 프로그램을 주요 함수 위주로 흐름도를 그려보면 다음과 같다.



[그림 3. 흐름도]

2) 실행화면의 과정을 설명하면 맨 처음 server.exe를 실행한다. 이 상태에서 server 프로그램은 accept() 함수까지 호출한 후 client의 connection 요청이 오기까지 기다린다. 그리고 새 cmd창에서 client.exe를 실행하면 client 프로그램은 server로 connection 요청을 보낸다.

3) 여기까지 완료되면 server는 recv()로 client의 메시지를 기다리는 상태가 되며, client는 fgets로 보낼 메시지를 입력하고 send로 전송할 수 있게 된다. client가 메시지를 보내면 client는 recv() 함수를 호출한 후 server가 보낼 메시지를 기다린다. server는 받은 메시지를 출력한 후 client에게 메시지를 보낼 수 있게 된다. 이 과정을 server와 client 어느 한쪽이 아무 내용 없는 메시지를 전송할 때까지 반복한다.

4) client 실행화면 캡처의 마지막에서 보면 client의 메시지 입력 상황에서 그냥 enter를 입력하자 server쪽 창에서 클라이언트 종료 메시지가 출력된다. 이는 client의 흐름도에서 노랑색 루프를 탈출한 후 closesocket()으로 client 소켓이 종료된 상황을 의미한다. 이 때 server쪽에서는 해당 client 소켓을 close한 후 다른 client의 connection 요청을 기다리는 상태로 돌아갈지, 아니면 서버 소켓을 닫을지 선택하게 된다. 실행화면에서는 y를 입력해 server를 종료한 모습이다.