

REPORT



과 목 :	네트워크프로그래밍
제출일자 :	2021. 09. 27
담당교수 :	황성호
학 과 :	컴퓨터공학과
학 번 :	201720970
이 름 :	권대한

1. TCP Server의 소스코드

```
#define _WINSOCK_DEPRECATED_NO_WARNINGS

#pragma comment(lib, "ws2_32")

#include <stdio.h>
#include <stdlib.h>
#include <WinSock2.h>

#define SERVER_PORT 9000
#define BUFFER_SIZE 512

// 소켓 함수 오류 출력 후 종료
//general error message output Func
void err_quit(char* msg)
{
    // "far" + pointer + void
    LPVOID lpMsgBuf;

    FormatMessage
    (
        //flag
        FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM,
        //source
        NULL,
        //message id
        WSAGetLastError(),
        //language id
        MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
        //Long Pointer t_string, buffer location
        (LPTSTR)&lpMsgBuf,
        //size
        0,
        //arguments
        NULL
    );

    MessageBox
    (
        //handle
        NULL,
```

```

        //text
        (LPCTSTR)lpMsgBuf,
        //caption
        msg,
        //type
        MB_ICONERROR
    );

    //pointer memory free Func
    LocalFree(lpMsgBuf);

    //error code
    exit(1);
}

void err_display(char* msg)
{
    LPVOID lpMsgBuf;

    FormatMessage
    (
        //flag
        FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM,
        //source
        NULL,
        //message id
        WSAGetLastError(),
        //language id
        MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
        //Long Pointer t_string, buffer location
        (LPTSTR)&lpMsgBuf,
        //size
        0,
        //arguments
        NULL
    );

    printf("[%s] %s", msg, (char*)lpMsgBuf);

    LocalFree(lpMsgBuf);
}

```

```
}
```

```
int main(int argc, char* argv[])
```

```
{
```

```
    int retval;
```

```
    //winsock init
```

```
    WSADATA wsa;
```

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

```
    {
```

```
        return 1;
```

```
    }
```

```
    //socket()
```

```
    SOCKET listen_sock = socket(AF_INET, SOCK_STREAM, 0);
```

```
    if (listen_sock == INVALID_SOCKET)
```

```
    {
```

```
        err_quit("socket()");
```

```
    }
```

```
    //bind()
```

```
    SOCKADDR_IN serveraddr;
```

```
    ZeroMemory(&serveraddr, sizeof(serveraddr));
```

```
    serveraddr.sin_family = AF_INET;
```

```
    serveraddr.sin_addr.s_addr = htonl(INADDR_ANY);
```

```
    //serveraddr.sin_addr.s_addr = inet_addr(INADDR_ANY);
```

```
    serveraddr.sin_port = htons(SERVER_PORT);
```

```
    retval = bind(listen_sock, (SOCKADDR*)&serveraddr, sizeof(serveraddr));
```

```
    if (retval == SOCKET_ERROR)
```

```
    {
```

```
        err_quit("bind()");
```

```
    }
```

```
    //listen()
```

```
    retval = listen(listen_sock, SOMAXCONN);
```

```
    if (retval == SOCKET_ERROR)
```

```
    {
```

```
        err_quit("listen()");
```

```
    }
```

```
    //데이터 통신 변수
```

```

SOCKET client_sock;
SOCKADDR_IN clientaddr;

int addrlen;
char buf[BUFFER_SIZE + 1];

while (1)
{
    //accept()
    addrlen = sizeof(clientaddr);
    client_sock = accept(listen_sock, (SOCKADDR*)&clientaddr, &addrlen);
    if (client_sock == INVALID_SOCKET)
    {
        err_display("accept()");
        break;
    }

    printf("\n[TCP 서버] 클라이언트 접속: IP 주소 = %s, 포트 번호 = %d\n",
        inet_ntoa(clientaddr.sin_addr), ntohs(clientaddr.sin_port));

    while (1)
    {
        retval = recv(client_sock, buf, BUFFER_SIZE, 0);
        if (retval == SOCKET_ERROR)
        {
            err_display("recv()");
            break;
        }
        else if (retval == 0)
        {
            break;
        }

        //받은 데이터 출력
        buf[retval] = 0;
        printf("[TCP/%s : %d] %s\n",
            inet_ntoa(clientaddr.sin_addr), ntohs(clientaddr.sin_port), buf);

        //데이터 보내기
        retval = send(client_sock, buf, retval, 0);
    }
}

```

```
        if (retval == SOCKET_ERROR)
        {
            err_display("send()");
            break;
        }
    }

    closesocket(client_sock);
    printf("[TCP 서버] 클라이언트 종료: IP 주소 = %s, 포트 번호 = %d\n",
        inet_ntoa(clientaddr.sin_addr), ntohs(clientaddr.sin_port));
}

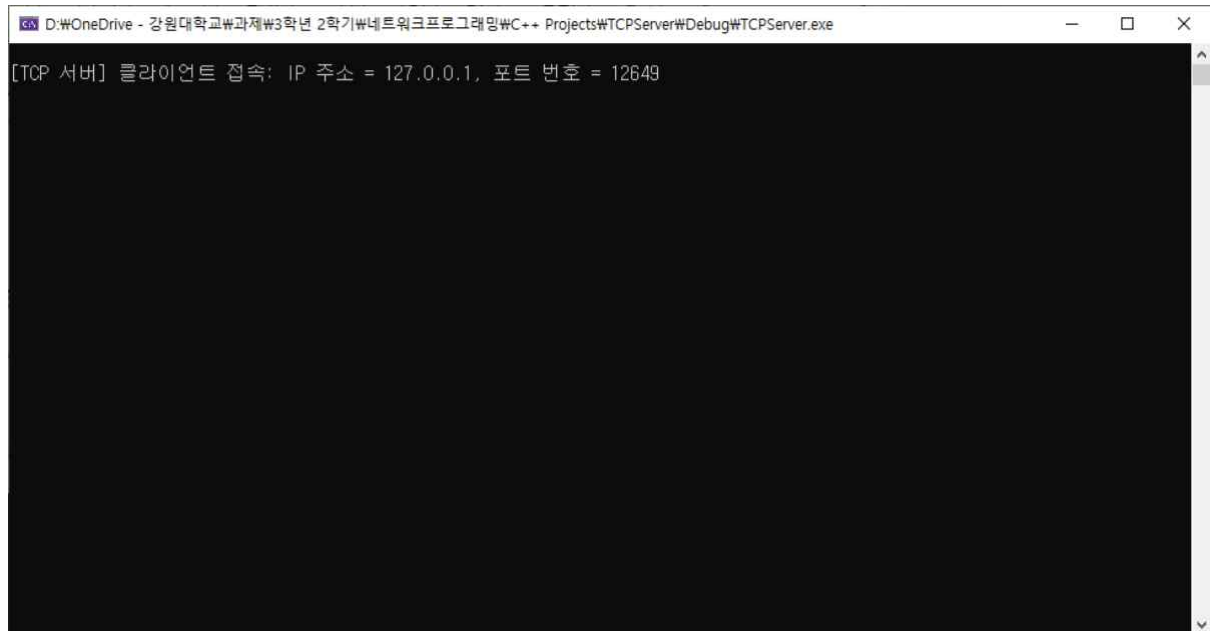
//closesocket()
closesocket(listen_sock);

//winsock 종료
WSACleanup();

return 0;
}
```

-> 출력결과

Case 1. TCPClient와 같이 실행하였을 경우.



The screenshot shows a Windows command prompt window with the title bar "D:\OneDrive - 강원대학교\학과\3학년 2학기\네트워크프로그래밍\C++ Projects\TCPServer\Debug\TCPServer.exe". The window contains the following text: "[TCP 서버] 클라이언트 접속: IP 주소 = 127.0.0.1, 포트 번호 = 12649". The rest of the window is black, indicating no further output.

Case 2. TCPClient에서 데이터를 수신하였을 경우.



The screenshot shows a Windows command prompt window with the title bar "D:\OneDrive - 강원대학교\학과\3학년 2학기\네트워크프로그래밍\C++ Projects\TCPServer\Debug\TCPServer.exe". The window contains the following text: "[TCP 서버] 클라이언트 접속: IP 주소 = 127.0.0.1, 포트 번호 = 12649" and "[TCP/127.0.0.1 : 12649] Submission!". The rest of the window is black, indicating no further output.

2. TCP Client의 소스코드

```
#define _WINSOCK_DEPRECATED_NO_WARNINGS

#pragma comment(lib, "ws2_32")

#include <stdio.h>
#include <stdlib.h>
#include <WinSock2.h>

#define SERVERIP "127.0.0.1" //loop-back
#define SERVER_PORT 9000
#define BUFFER_SIZE 512

// 소켓 함수 오류 출력 후 종료
//general error message output Func
void err_quit(char* msg)
{
    // "far" + pointer + void
    LPVOID lpMsgBuf;

    FormatMessage
    (
        //flag
        FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM,
        //source
        NULL,
        //message id
        WSAGetLastError(),
        //language id
        MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
        //Long Pointer t_string, buffer location
        (LPTSTR)&lpMsgBuf,
        //size
        0,
        //arguments
        NULL
    );

    MessageBox
    (
        //handle
```



```

        NULL,
        //text
        (LPCTSTR)lpMsgBuf,
        //caption
        msg,
        //type
        MB_ICONERROR
    );

    //pointer memory free Func
    LocalFree(lpMsgBuf);

    //error code
    exit(1);
}

//소켓 함수 오류 출력
void err_display(char* msg)
{
    LPVOID lpMsgBuf;

    FormatMessage
    (
        //flag
        FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM,
        //source
        NULL,
        //message id
        WSAGetLastError(),
        //language id
        MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
        //Long Pointer t_string, buffer location
        (LPTSTR)&lpMsgBuf,
        //size
        0,
        //arguments
        NULL
    );

    printf("[%s] %s", msg, (char*)lpMsgBuf);

```

```
LocalFree(lpMsgBuf);
```

```
}
```

```
//사용자 정의 데이터 수신 함수
```

```
int recvn(SOCKET s, char* buf, int len, int flags)
```

```
{
```

```
    int received;
```

```
    char* ptr = buf;
```

```
    int left = len;
```

```
    while (left > 0)
```

```
    {
```

```
        received = recv(
```

```
            //socket struct, object
```

```
            s,
```

```
            //buffer
```

```
            ptr,
```

```
            //left length
```

```
            left,
```

```
            //flags, default == 0
```

```
            flags);
```

```
        if (received == SOCKET_ERROR)
```

```
        {
```

```
            return SOCKET_ERROR;
```

```
        }
```

```
        //Transfer end
```

```
        else if (received == 0)
```

```
        {
```

```
            break;
```

```
        }
```

```
        left -= received;
```

```
        ptr += received;
```

```
    }
```

```
    return (len - left);
```

```
}
```

```
int main(int argc, char *argv[])
```

```
{
```

```

int retval;

//winsock init
WSADATA wsa;
if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0)
{
    return 1;
}

//socket()
SOCKET sock = socket(AF_INET, SOCK_STREAM, 0);
if (sock == INVALID_SOCKET)
{
    err_quit("socket()");
}

//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(SERVER_PORT);
retval = connect(sock, (SOCKADDR*)&serveraddr, sizeof(serveraddr));
if (retval == SOCKET_ERROR)
{
    err_quit("connect()");
}

//데이터 통신에 사용할 변수
char buf[BUFFER_SIZE + 1];
int len;

//서버와 데이터 통신
while (1)
{
    //데이터 입력
    printf("\n[보낼 데이터] ");
    if (fgets(buf, BUFFER_SIZE + 1, stdin) == NULL)
    {
        break;
    }
}

```

```
// '\n' 문자 제거 <- 사용자가 전달하는 텍스트에 개행문자가 포함되어 있으므로,,!
```

```
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);
```

```
if (retval == SOCKET_ERROR)
```

```
{
```

```
    err_display("send()");
```

```
    break;
```

```
}
```

```
printf("[TCP 클라이언트] %d바이트를 보냈습니다.\n", retval);
```

```
//데이터 받기
```

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

```
if (retval == SOCKET_ERROR)
```

```
{
```

```
    err_display("recv()");
```

```
    break;
```

```
}
```

```
else if (retval == 0)
```

```
{
```

```
    break;
```

```
}
```

```
//받은 데이터 출력
```

```
buf[retval] = '\0';
```

```
printf("[TCP 클라이언트] %d바이트를 받았습니다.\n", retval);
```

```
printf("[받은 데이터] %s\n", buf);
```

```
}
```

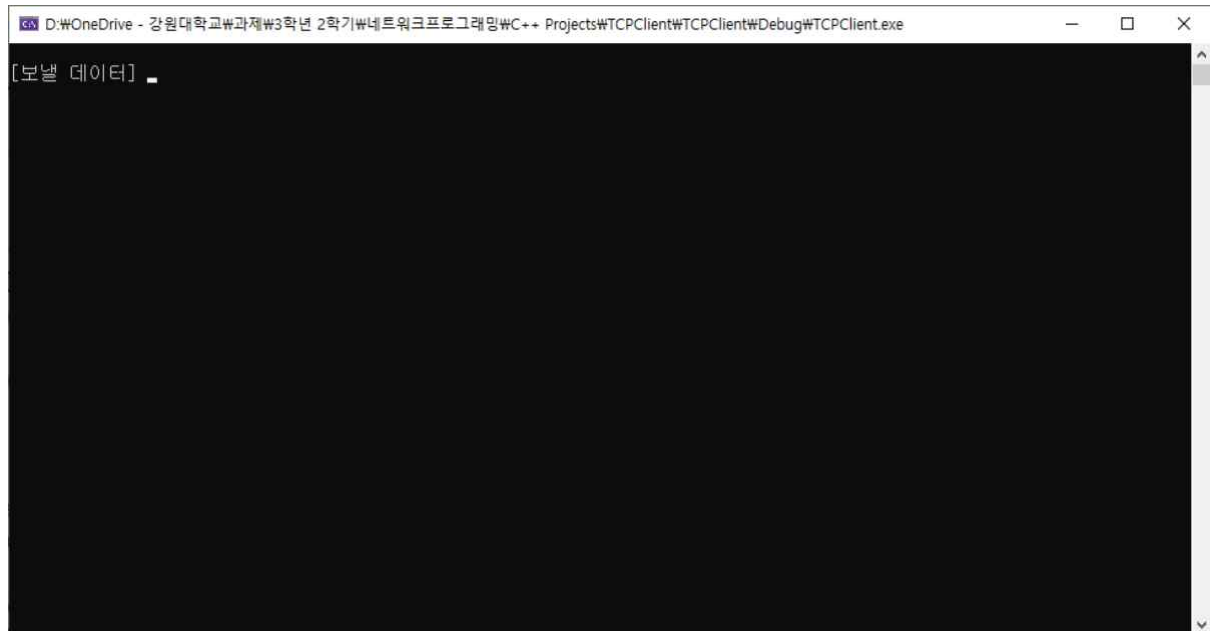
```
//closesocket()
closesocket(sock);

//winsock 종료
WSACleanup();

return 0;
}
```

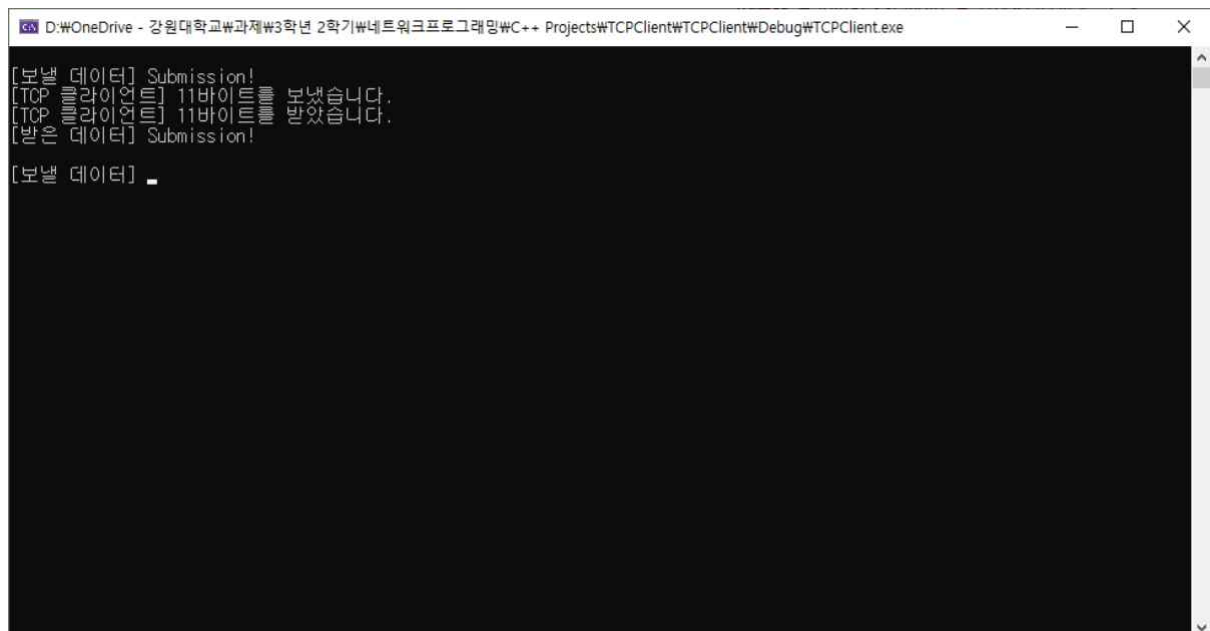
-> 출력결과

Case 1. TCPServer와 동시 실행하였을 경우.



```
D:\OneDrive - 강원대학교\학과\제3학년 2학기\네트워크프로그래밍\C++ Projects\TCPServer\TCPServer\Debug\TCPServer.exe
[보낼 데이터] _
```

Case 2. TCPServer에 데이터를 보냈을 경우.



```
D:\OneDrive - 강원대학교\학과\제3학년 2학기\네트워크프로그래밍\C++ Projects\TCPServer\TCPServer\Debug\TCPServer.exe
[보낼 데이터] Submission!
[TCP 클라이언트] 11바이트를 보냈습니다.
[TCP 클라이언트] 11바이트를 받았습니다.
[받은 데이터] Submission!
[보낼 데이터] _
```