

응용소프트웨어실습

소켓 프로그래밍

- ☐ 네트워크
- ☐ 소켓 프로그래밍
- ☐ 채팅 프로그램 실습
- ☐ 패킷 통신

네트워크

☐ 컴퓨터 네트워크

- 대부분의 인터넷 망은 TCP/IP 프로토콜로 구성되어 있음.
- TCP/IP 프로토콜을 이용하여 많은 네트워크 프로그램이 개발되고, 사용되고 있음.
 - ☐ TCP (Transmission Control Protocol)
 - ☐ IP (Internet Protocol)
 - ☐ 패킷 (Packet)
 - 데이터의 형식화 된 블록이며, 네트워크 계층에서 사용되는 데이터의 단위

☐ TCP (Transmission Control Protocol)

- 연결형
- 패킷 단위로 데이터를 전송함.
- 손상 되었거나 수신되지 않은 데이터를 검사하여 재전송 함.
- 전송한 순서대로 수신함.
- 신뢰성이 높음.
- 네트워크 부하가 상대적으로 많음.

☐ UDP (User Datagram Protocol)

- 비연결형
- 패킷 단위로 데이터를 전송함.
- 패킷 손상만 검사하고, 패킷의 순서 또는 손실 여부는 검사하지 않음.
- 데이터를 전송 시에, 데이터가 수신되었는지 확인할 수 없음.
- 신뢰성이 낮음.
- 네트워크 부하가 상대적으로 적음.

☐ IP 주소 (IP Address)

■ 네트워크 상의 장치를 구분하는 식별자

구분	IPv4	IPv6
주소길이	32bit	128bit
표시방법	1byte 4부분으로 표시 (10진수 표기) 예) 202.30.64.139	2byte 8부분으로 표시 (16진수 표기) 예) 2001:0230:avcd:ffff:0000:0000:ffff:1234
주소개수	$2^{32} = 4,294,967,296$ (약 43억)	$2^{128} \approx 3.4 \times 10^{38}$ (무한대?)
주소할당	A,B,C,D 등 클래스 단위의 비순차적 할당	네트워크 규모 및 단말기수에 따른 순차적 할당

☐ 포트 번호 (Port Number)

■ 하나의 장치에서 각 서비스를 구분하는 번호

- ☐ 0과 65535 사이의 정수
- ☐ 포트 번호는 전송 계층에서 사용하는 번호

소켓 프로그래밍

□ 소켓 관련 클래스

- 네트워크에 존재하는 다른 장치에 연결하고 통신을 가능하게 함.

클래스	설명
Socket	실제 네트워크 연결 제어 클래스
TcpClient	TCP 클라이언트
UdpClient	UDP 클라이언트
TcpListener	서버 프로그램에서 특정 포트를 열어놓고 클라이언트 요구를 대기

- ☐ 서버와 클라이언트

- 서버 (Server)

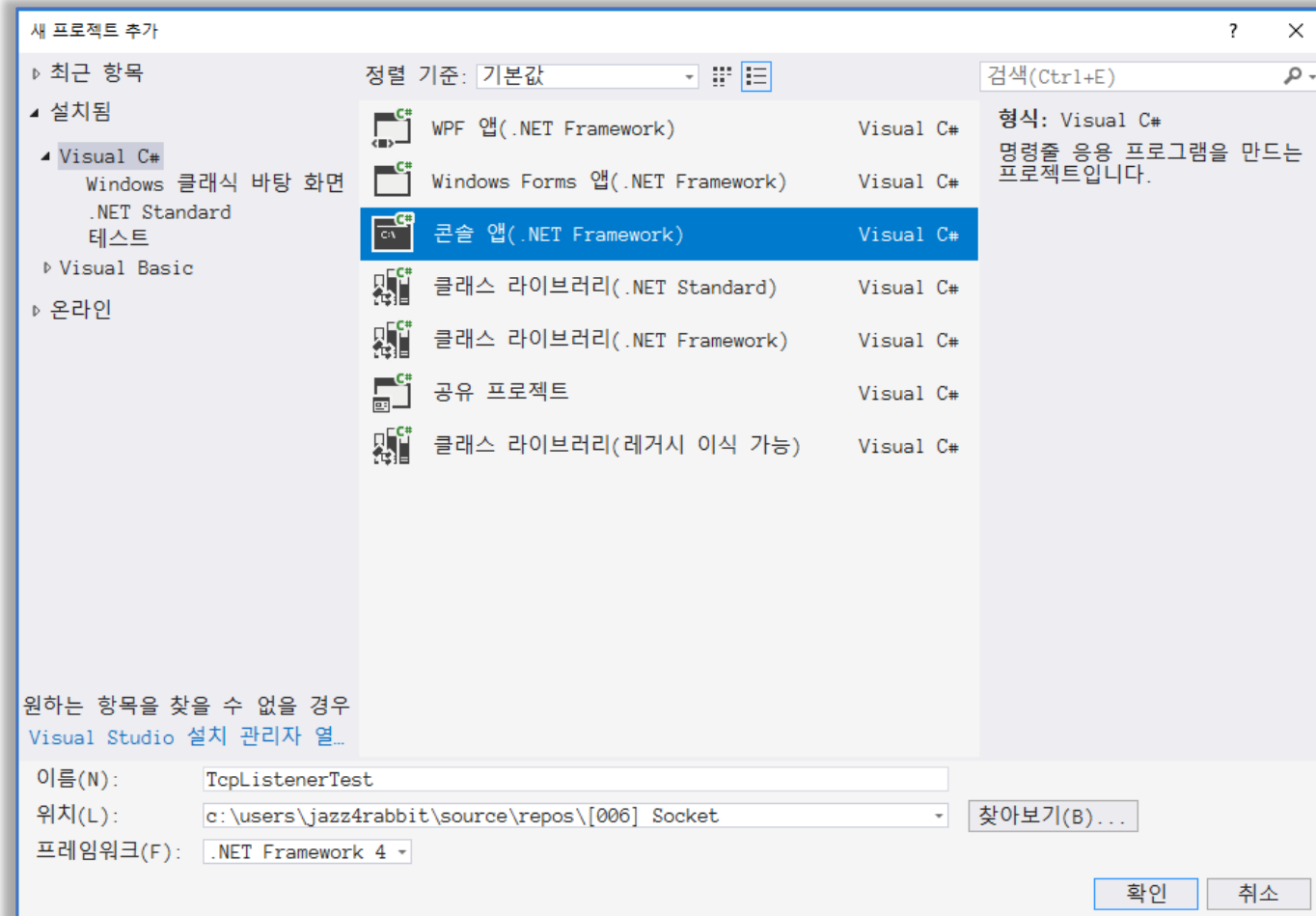
- ☐ 네트워크를 통해 클라이언트와 연결하고, 정보 또는 서비스를 제공하는 장치 및 프로그램

- 클라이언트 (Client)

- ☐ 서버에 접속하고, 정보 또는 서비스를 요청하는 장치 및 프로그램

소켓 프로그래밍 실습 예시 - 서버

□ 새 프로젝트 생성



소켓 프로그래밍 실습 예시 - 서버 (cont`d)

☐ 네임스페이스 추가

```
1 using System;  
2 using System.Net;  
3 using System.Net.Sockets;  
4 using System.Text;
```

■ System

- ☐ Console, Datetime 클래스 등

■ System.Net

- ☐ IPAddress 클래스

■ System.Net.Sockets

- ☐ TcpListener, TcpClient 클래스

■ System.Text

- ☐ Encoding 클래스

□ TcpListener 생성 및 시작

```
9      TcpListener server = null;
10     IPAddress localAddr = IPAddress.Parse("127.0.0.1"); int port = 13000;
11     try {
12         server = new TcpListener(localAddr, port);
13         server.Start();
14
15         // listening loop
```

■ listening loop는 다음 페이지에 계속

□ catch, finally, 코드 종료

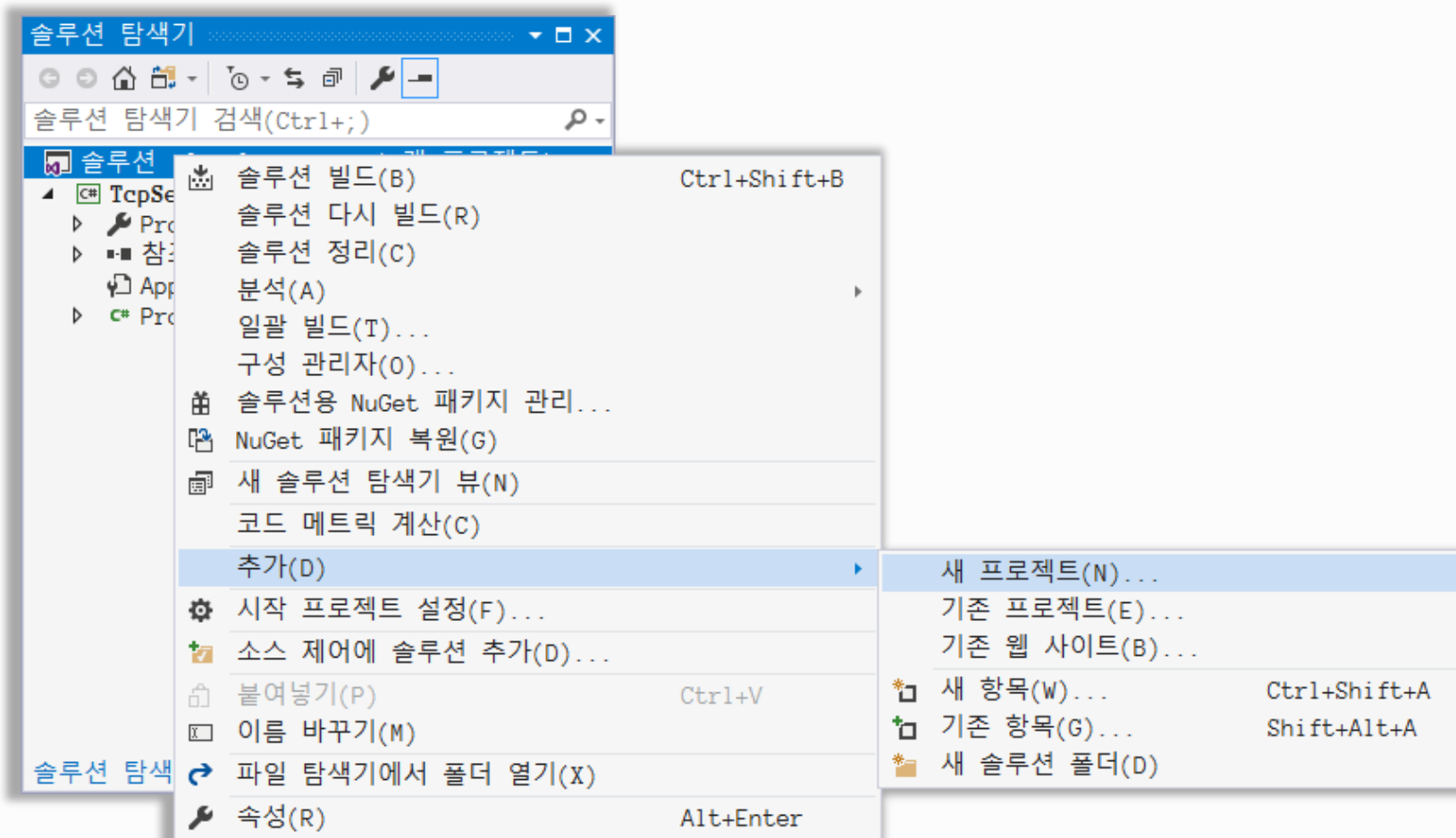
```
43     } catch (SocketException e) {
44         Console.WriteLine("SocketException:{0}", e);
45     } finally {
46         server.Stop();
47     }
48     Console.WriteLine("\n서버가 종료됩니다.");
49 }
```

□ TcpListener의 listening loop

```
15 // listening loop
16 while (true) {
17     Console.WriteLine("Waiting for a connection...");
18     TcpClient client = server.AcceptTcpClient();
19     Console.WriteLine("Connected!");
20
21     DateTime t = DateTime.Now;
22     // string to byte
23     string message = string.Format("서버에서 보내는 메세지 {0}", t.ToString("yyyy-MM-dd hh:mm:ss"));
24     byte[] writeBuffer = Encoding.UTF8.GetBytes(message);
25
26     // int to byte
27     int bytes = writeBuffer.Length;
28     byte[] writeBufferSize = BitConverter.GetBytes(bytes);
29
30     // send to client
31     NetworkStream stream = client.GetStream();
32     // send BufferSize
33     stream.Write(writeBufferSize, 0, writeBufferSize.Length);
34     Console.WriteLine("Sent: {0}", bytes);
35     // send Buffer
36     stream.Write(writeBuffer, 0, writeBuffer.Length);
37     Console.WriteLine("Sent: {0}", message);
38
39     stream.Close();
40     client.Close();
41     Console.WriteLine();
42 }
```

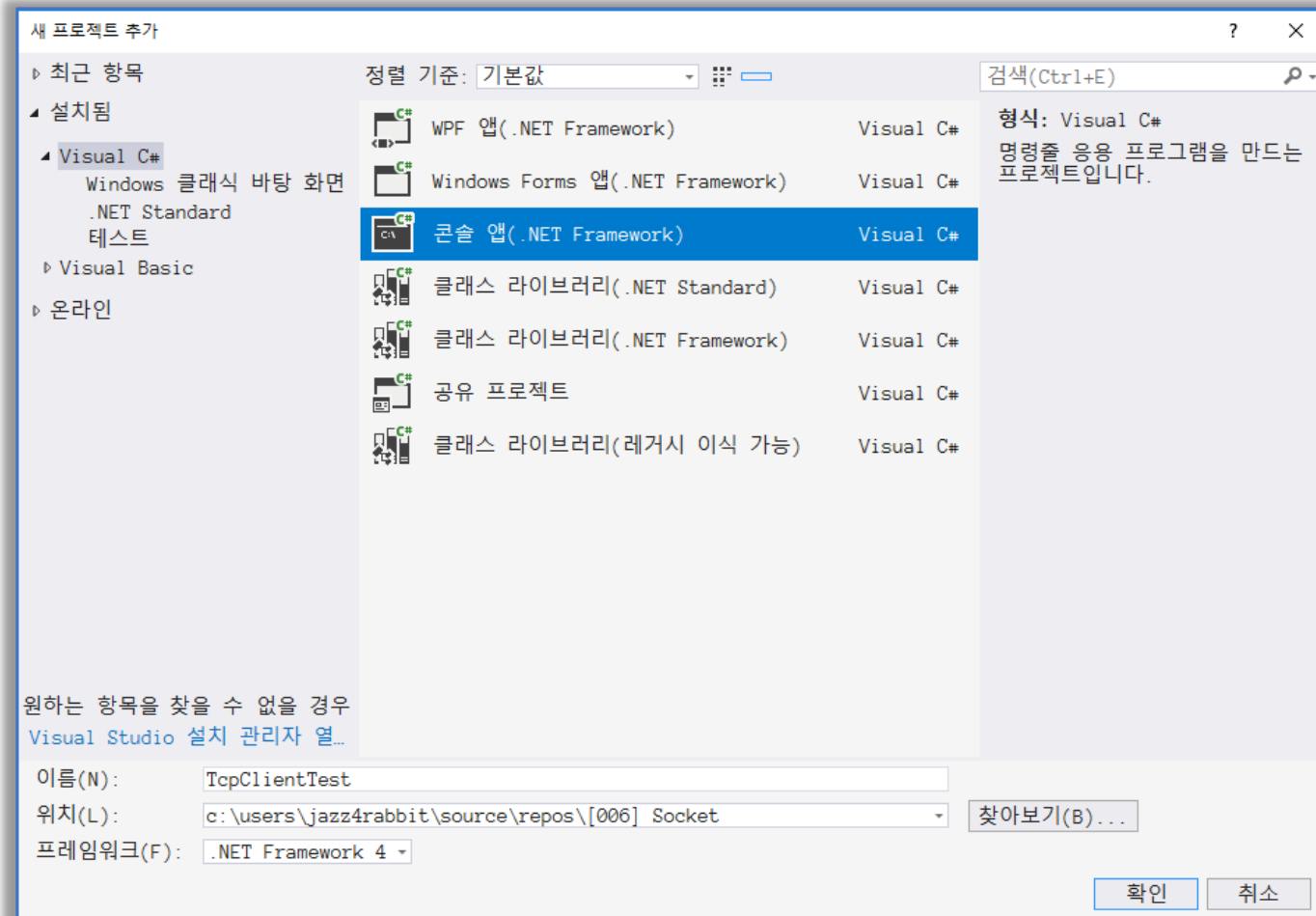
소켓 프로그래밍 실습 예시 - 클라이언트

□ 추가 > 새 프로젝트



소켓 프로그래밍 실습 예시 - 클라이언트 (cont`d)

□ 새 프로젝트 생성



소켓 프로그래밍 실습 예시 - 클라이언트 (cont`d)

□ 네임스페이스 추가

```
1 using System;  
2 using System.Net;  
3 using System.Net.Sockets;  
4 using System.Text;
```

□ TcpClient를 통한 서버와 연결

```
9 TcpClient client = null;  
10 try {  
11     // client = new TcpClient("127.0.0.1",port)와 동일  
12     client = new TcpClient();  
13     IPAddress localAddr = IPAddress.Parse("127.0.0.1"); int port = 13000;  
14     client.Connect(localAddr, port);  
15 }
```


소켓 프로그래밍 실습 예시 - 클라이언트 (cont`d)

□ 연결된 서버에서 메시지 수신

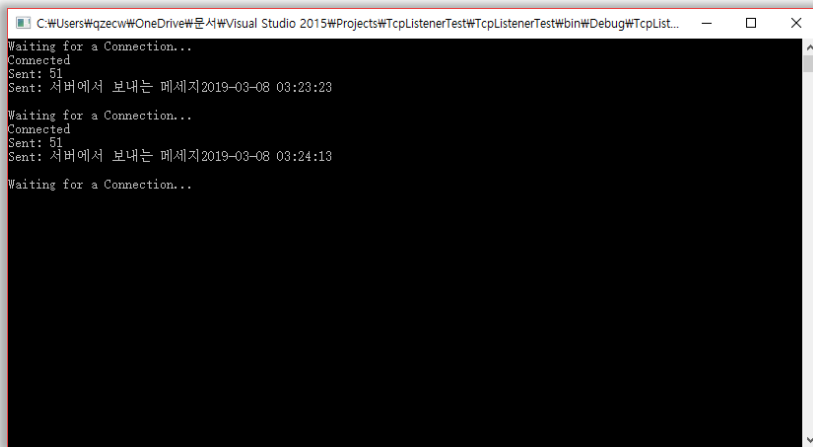
```
16 NetworkStream stream = client.GetStream();
17 byte[] readBuffer = new byte[sizeof(int)];
18
19 // read bufferSize
20 stream.Read(readBuffer, 0, readBuffer.Length);
21 int bufferSize = BitConverter.ToInt32(readBuffer, 0);
22 Console.WriteLine("Received: {0}", bufferSize);
23
24 // read buffer
25 readBuffer = new byte[bufferSize];
26 int bytes = stream.Read(readBuffer, 0, readBuffer.Length);
27 string message = Encoding.UTF8.GetString(readBuffer, 0, bytes);
28 Console.WriteLine("Received: {0}", message);
29
30 stream.Close();
31 client.Close();
32 } catch (SocketException e) {
33     Console.WriteLine("SocketException: {0}", e);
34 } finally {
35     client.Close();
36 }
37 Console.WriteLine("Client Exit");
38 }
```

코드 추가 :

```
Console.WriteLine("계속 하시려면 아무키나 누르세요.");
Console.ReadKey(); // or Console.ReadLine();
```

소켓 프로그래밍 실습 예시

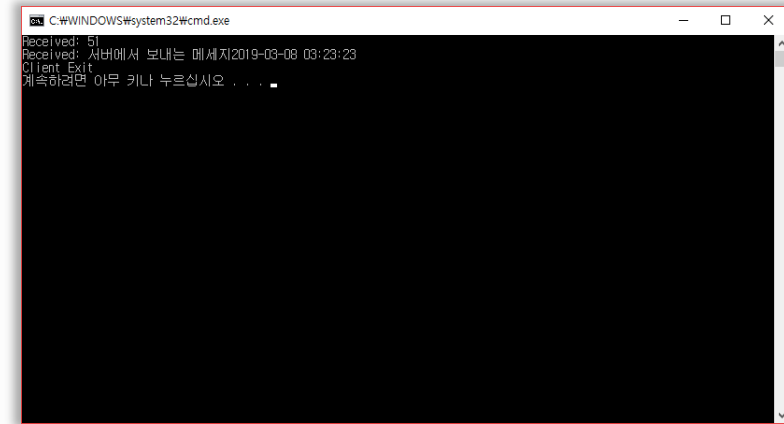
□ 결과 화면



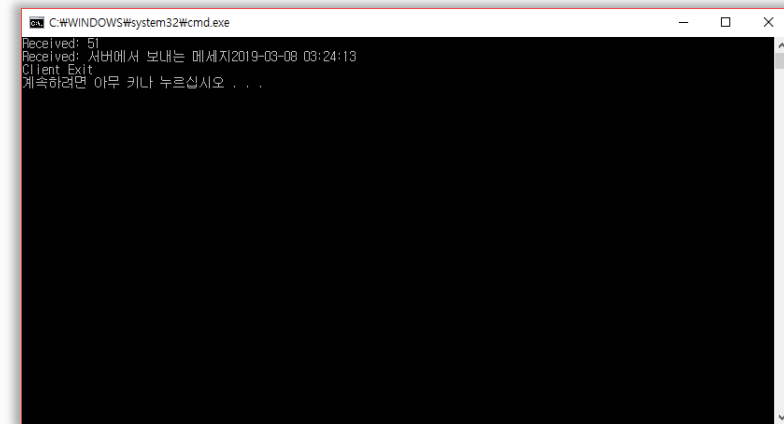
```
C:\Users#gqecw\OneDrive\문서\Visual Studio 2015\Projects\TcpListenerTest\TcpListenerTest\bin\Debug\TcpList...  
Waiting for a Connection...  
Connected  
Sent: 51  
Sent: 서버에서 보내는 메시지 2019-03-08 03:23:23  
Waiting for a Connection...  
Connected  
Sent: 51  
Sent: 서버에서 보내는 메시지 2019-03-08 03:24:13  
Waiting for a Connection...
```

서버

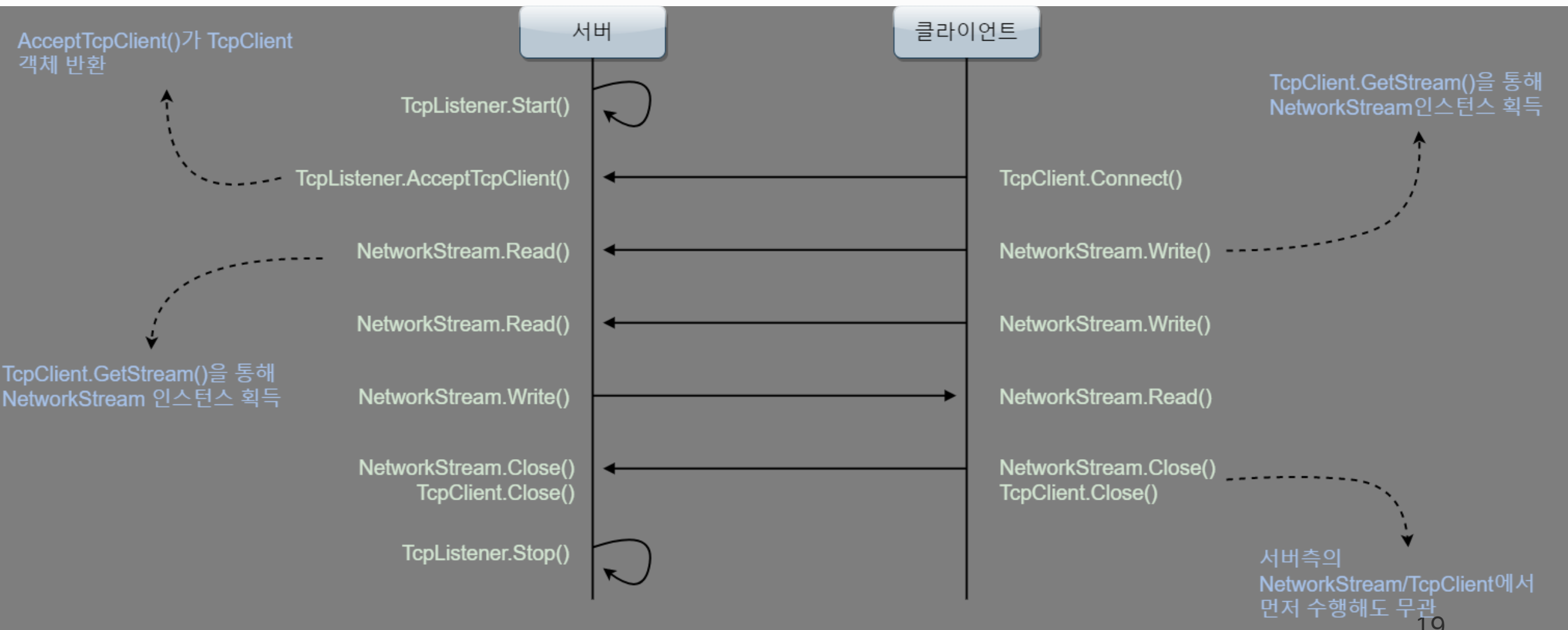
클라이언트



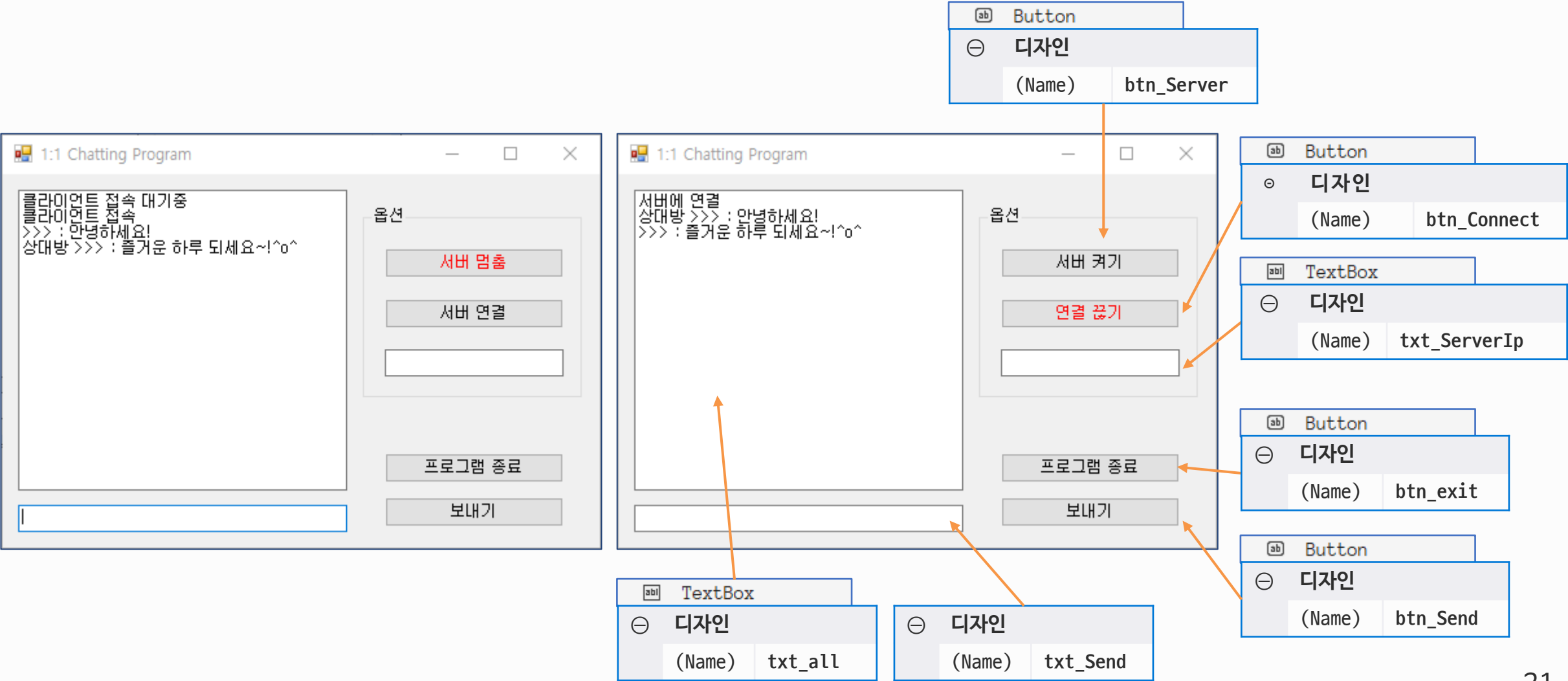
```
C:\WINDOWS\system32\cmd.exe  
Received: 51  
Received: 서버에서 보내는 메시지 2019-03-08 03:23:23  
Client Exit  
계속하려면 아무 키나 누르십시오 . . .
```



```
C:\WINDOWS\system32\cmd.exe  
Received: 51  
Received: 서버에서 보내는 메시지 2019-03-08 03:24:13  
Client Exit  
계속하려면 아무 키나 누르십시오 . . .
```



채팅 프로그램 실습



Multi Line : True

□ 네임스페이스 추가

```
using System;
using System.Drawing;
using System.IO;
using System.Net.Sockets;
using System.Threading;
using System.Windows.Forms;
```

□ 멤버 변수 추가

```
//server client members
public NetworkStream m_Stream;      //네트워크 스트림
public StreamReader m_Read;         //읽기
public StreamWriter m_Write;        //쓰기
const int PORT = 2002;              //포트번호
private Thread m_ThReader;          //읽기 스레드

//server members
public bool m_bStop = false;        //서버 시작&중단 플래그

private TcpListener m_listener;     //서버 작동 리스너
private Thread m_thServer;          //서버 스레드

//client members
public bool m_bConnect = false;     //서버 접속 플래그
TcpClient m_Client;
```

□ FormClosing 이벤트 핸들러

```
private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    ServerStop();    // 서버 중지
    Disconnect();    // 연결 종료
}
```

뒤에서 작성함

□ 프로그램 종료 버튼의 클릭 이벤트 핸들러

```
private void btn_exit_Click(object sender, System.EventArgs e)
{
    this.Close();
}
```

□ 문자열을 왼쪽 위 텍스트 박스에 추가하는 메소드

```
public void Message(string msg)
{
    txt_all.AppendText(msg + "\r\n");    //채팅 목록 창에 문자열 추가

    txt_all.Focus();
    txt_all.ScrollToCaret();              //현재 캐럿의 위치로 윈도우 스크롤 이동

    txt_send.Focus();
}
```

```
public void Message(string msg)
{
    this.Invoke(new MethodInvoker(delegate ()
    {
        txt_all.AppendText(msg + "\n");
        txt_all.Focus();
        txt_all.ScrollToCaret();
        txt_send.Focus();
    }));
}
```


□ 서버 시작 메소드

```
public void ServerStart()
{
    try
    {
        m_listener = new TcpListener(PORT);
        m_listener.Start();

        m_bStop = true;
        Message("클라이언트 접속 대기중");

        while (m_bStop)
        {
            TcpClient hClient = m_listener.AcceptTcpClient();

            if (hClient.Connected)
            {
                m_bConnect = true;
                Message("클라이언트 접속");

                m_Stream = hClient.GetStream();
                m_Read = new StreamReader(m_Stream);
                m_Write = new StreamWriter(m_Stream);

                m_ThReader = new Thread(new ThreadStart(Receive));
                m_ThReader.Start();
            }
        }
    }
    catch
    {
        Message("시작 도중에 오류 발생");
        return;
    }
}
```

뒤에서 작성함

□ 서버 중단 메소드

```
public void ServerStop()
{
    if (!m_bStop)
        return;

    m_listener.Stop(); //서버 소켓 작동 중지

    m_Read.Close();
    m_Write.Close();

    m_Stream.Close();

    m_ThReader.Abort(); //서버 소켓 스레드 종료
    m_thServer.Abort(); //스레드 종료

    Message("서비스 종료");
}
```

□ 서버 연결 해제 메소드

```
public void Disconnect()
{
    if (!m_bConnect)
        return;

    m_bConnect = false;

    m_Read.Close();
    m_Write.Close();

    m_Stream.Close();
    m_ThReader.Abort();

    Message("상대방과 연결 중단");
}
```

□ 서버 연결 메소드

```
public void Connect()
{
    m_Client = new TcpClient();

    try
    {
        m_Client.Connect(txt_ServerIp.Text, PORT);
    }
    catch
    {
        m_bConnect = false;
        return;
    }

    m_bConnect = true;
    Message("서버에 연결");

    m_Stream = m_Client.GetStream();

    m_Read = new StreamReader(m_Stream);
    m_Write = new StreamWriter(m_Stream);

    m_ThReader = new Thread(new ThreadStart(Receive));
    m_ThReader.Start();
}
```

□ 메시지 수신 메소드

```
public void Receive()
{
    try
    {
        while (m_bConnect)
        {
            string szMessage = m_Read.ReadLine();

            if (szMessage != null)
                Message("상대방 >>> : " + szMessage);
        }
    }
    catch
    {
        Message("데이터를 읽는 과정에서 오류가 발생");
    }
    Disconnect();
}
```

□ 메시지 송신 메소드

```
public void Send()
{
    try
    {
        m_Write.WriteLine(txt_send.Text);
        m_Write.Flush();

        Message(">>> : " + txt_send.Text);
        txt_send.Text = "";
    }
    catch
    {
        Message("데이터 전송 실패");
    }
}
```

□ 서버 켜기 버튼 클릭 이벤트 핸들러

```
private void btn_Server_Click(object sender, EventArgs e)
{
    if (btn_Server.Text == "서버 켜기")
    {
        m_thServer = new Thread(new ThreadStart(ServerStart));
        m_thServer.Start();

        btn_Server.Text = "서버 멈춤";
        btn_Server.ForeColor = Color.Red;
    }
    else
    {
        ServerStop();
        btn_Server.Text = "서버 켜기";
        btn_Server.ForeColor = Color.Black;
    }
}
```

□ 서버 연결 버튼 클릭 이벤트 핸들러

```
private void btn_Connect_Click(object sender, EventArgs e)
{
    if (btn_Connect.Text == "서버 연결")
    {
        Connect();
        if (m_bConnect)
        {
            btn_Connect.Text = "연결 끊기";
            btn_Connect.ForeColor = Color.Red;
        }
    }
    else
    {
        Disconnect();
        btn_Connect.Text = "서버 연결";
        btn_Connect.ForeColor = Color.Black;
    }
}
```

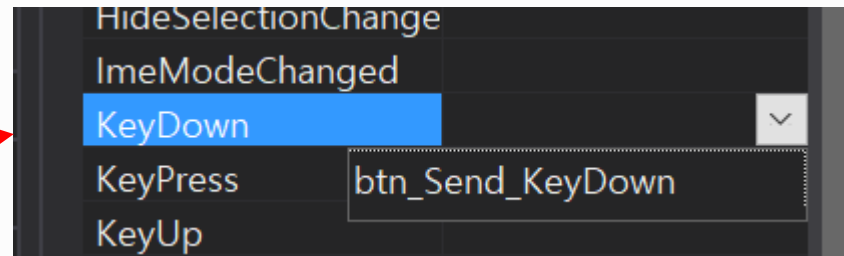
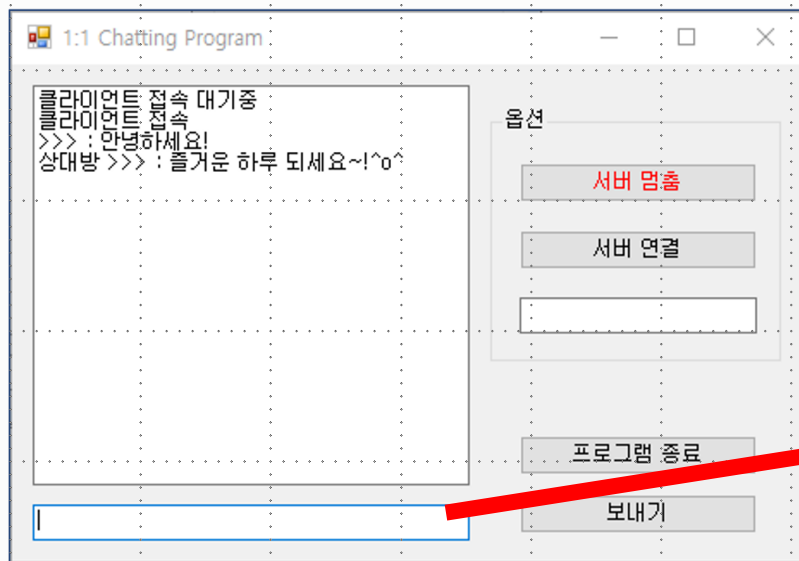
채팅 프로그램 실습 (cont`d)

□ 보내기 버튼 클릭 이벤트 핸들러

```
private void btn_Send_Click(object sender, EventArgs e)
{
    Send();
}
```

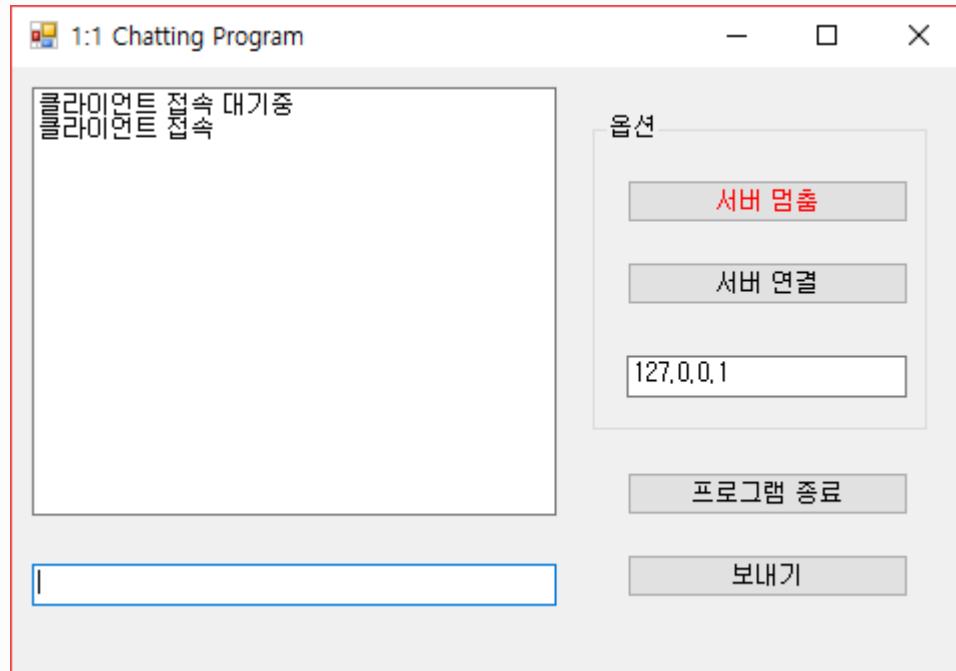
□ 문자열 송신 키 입력 이벤트 핸들러

```
private void txt_send_KeyDown(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Enter)
        Send();
}
```

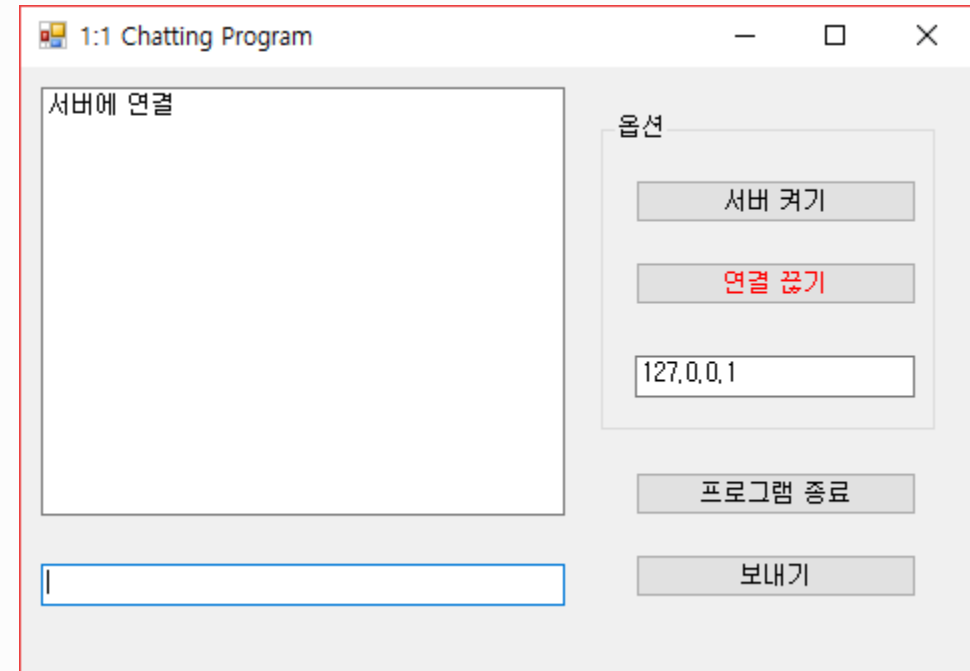


채팅 프로그램 실습 (cont`d)

□ 결과 화면



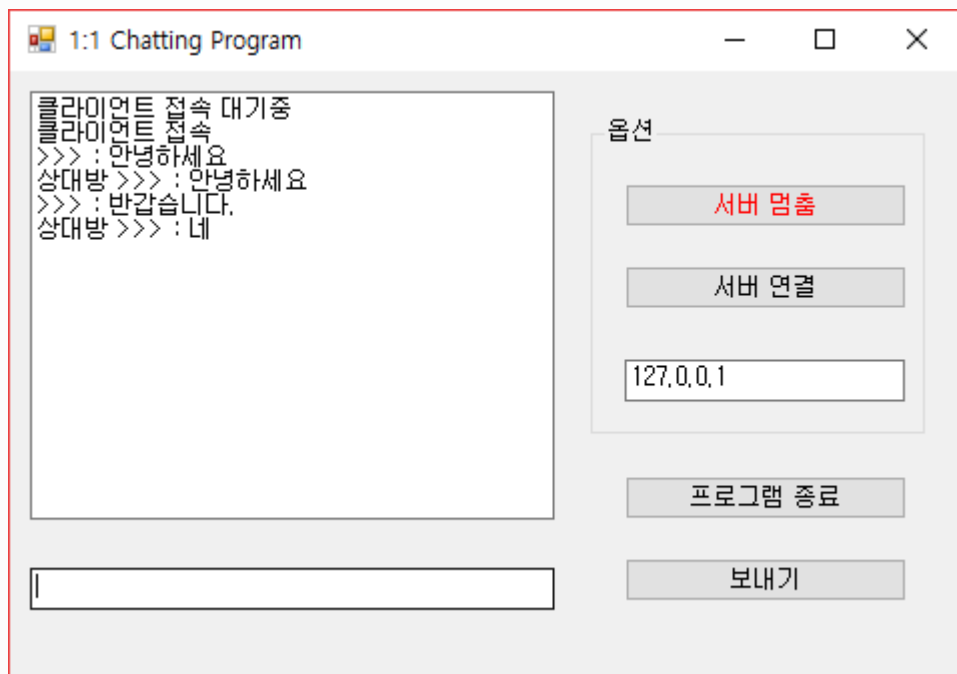
서버



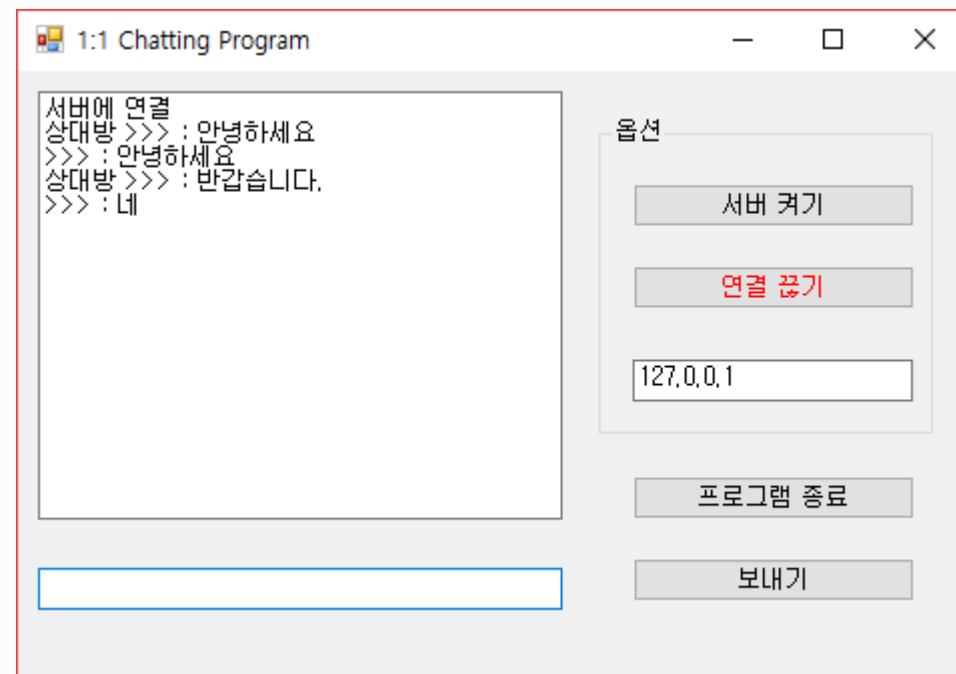
클라이언트

채팅 프로그램 실습 (cont`d)

□ 결과 화면 (cont`d)



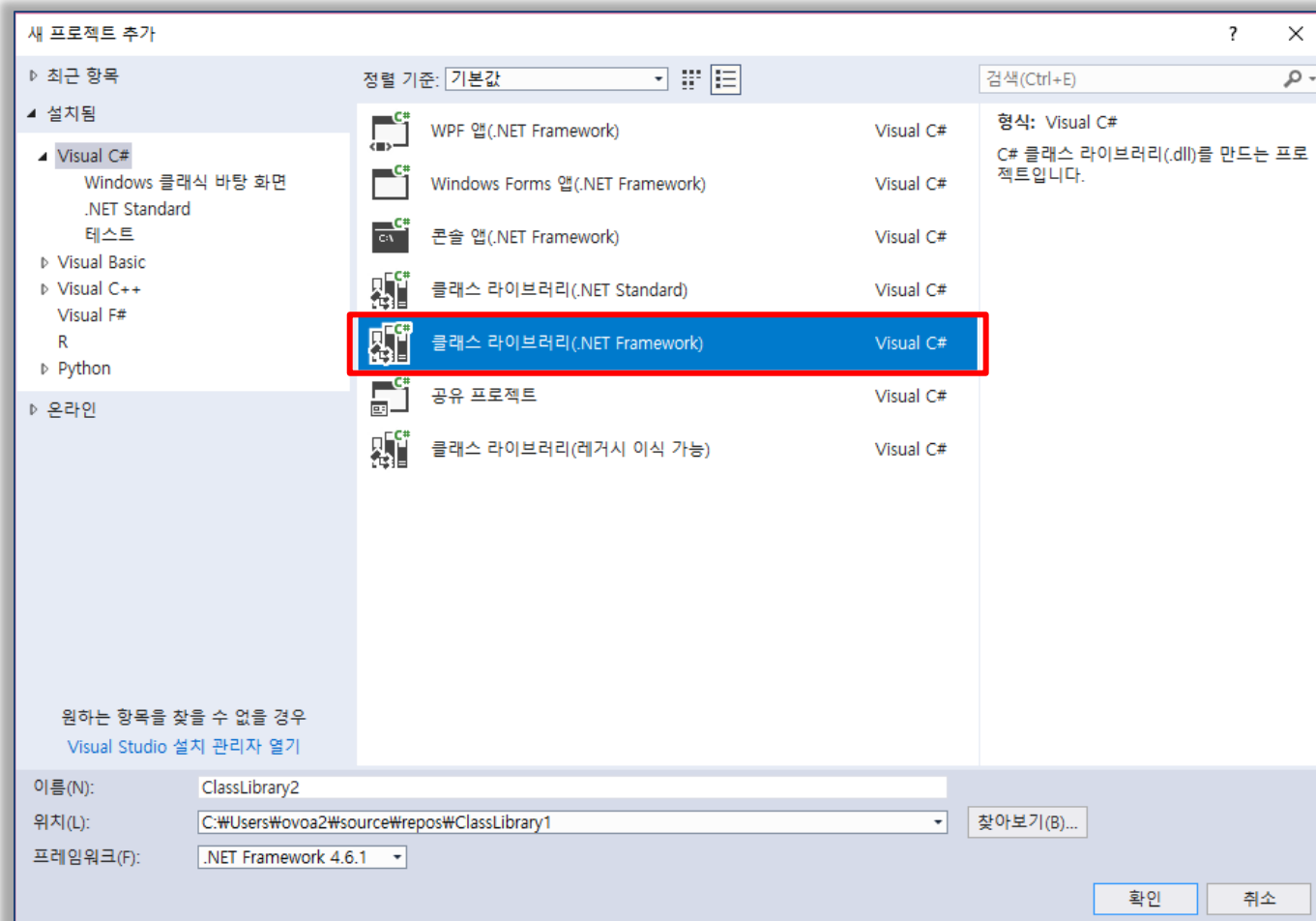
서버



클라이언트

패킷 통신

□ 프로젝트 생성



☐ 네임스페이스 추가

```
using System;  
using System.IO;  
using System.Runtime.Serialization.Formatters.Binary;
```

☐ 패킷 통신을 위한 열거자 작성

```
public enum PacketType  
{  
    초기화 = 0,  
    로그인  
}  
  
public enum PacketSendERROR  
{  
    정상 = 0,  
    에러  
}
```

□ Packet 클래스 정의

```
[Serializable]
public class Packet
{
    public int Length;
    public int Type;

    public Packet()
    {
        this.Length = 0;
        this.Type = 0;
    }

    public static byte[] Serialize(Object o)
    {
        MemoryStream ms = new MemoryStream(1024 * 4);
        BinaryFormatter bf = new BinaryFormatter();
        bf.Serialize(ms, o);
        return ms.ToArray();
    }

    public static Object Dessererialize(byte[] bt)
    {
        MemoryStream ms = new MemoryStream(1024 * 4);
        foreach (byte b in bt)
        {
            ms.WriteByte(b);
        }

        ms.Position = 0;
        BinaryFormatter bf = new BinaryFormatter();
        Object obj = bf.Deserialize(ms);
        ms.Close();
        return obj;
    }
}
```

☐ Packet initializing 클래스 작성

```
[Serializable]
public class Initialize : Packet
{
    public int Data = 0;
}
```

☐ Packet 통신을 위한 클래스 작성

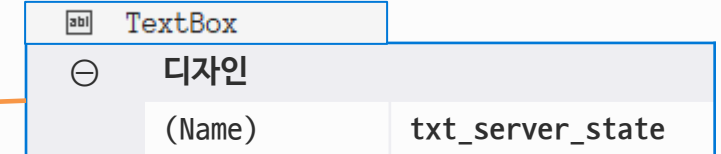
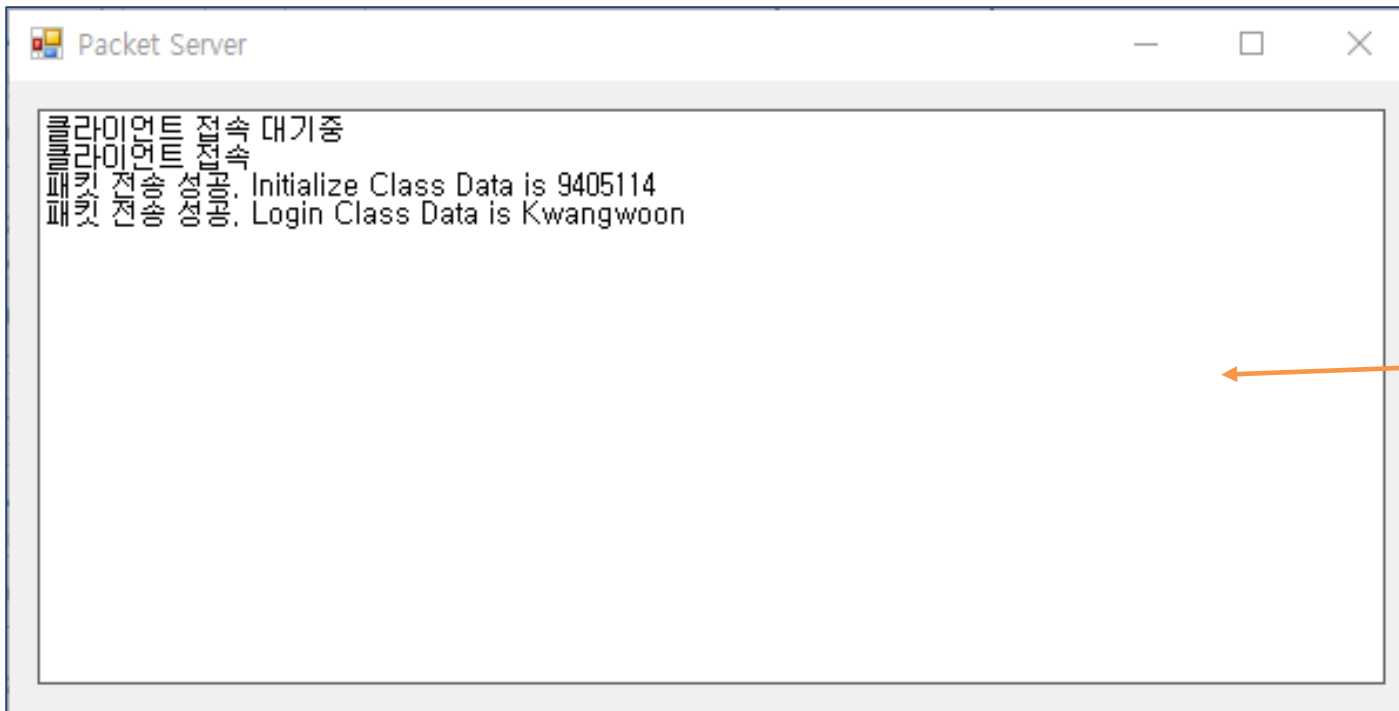
```
[Serializable]
public class Login : Packet
{
    public string m_strID;

    public Login()
    {
        this.m_strID = null;
    }
}
```

패킷 통신 실습 예시 - 서버 (cont`d)

□ 프로젝트 생성 및 폼 구성

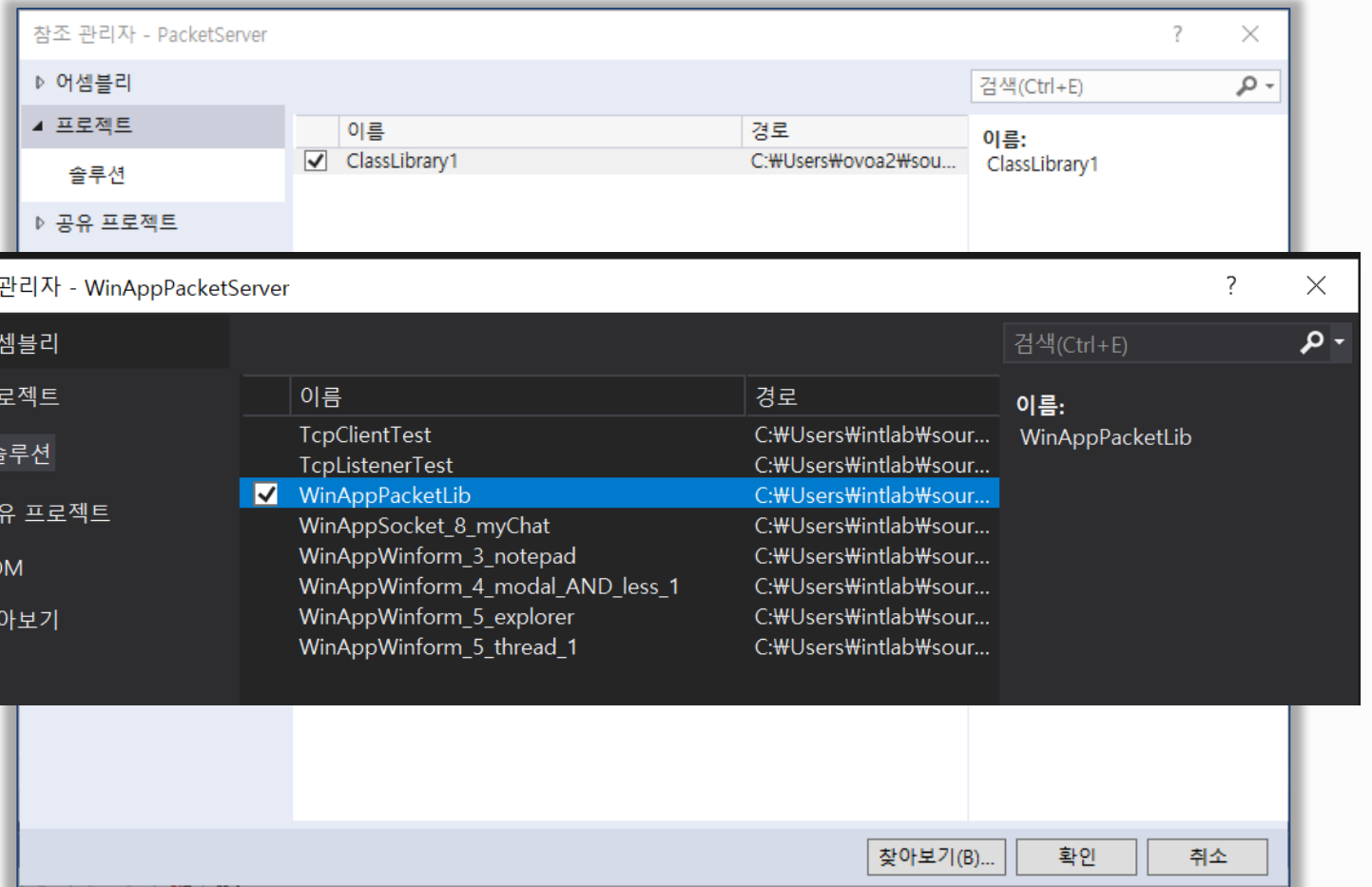
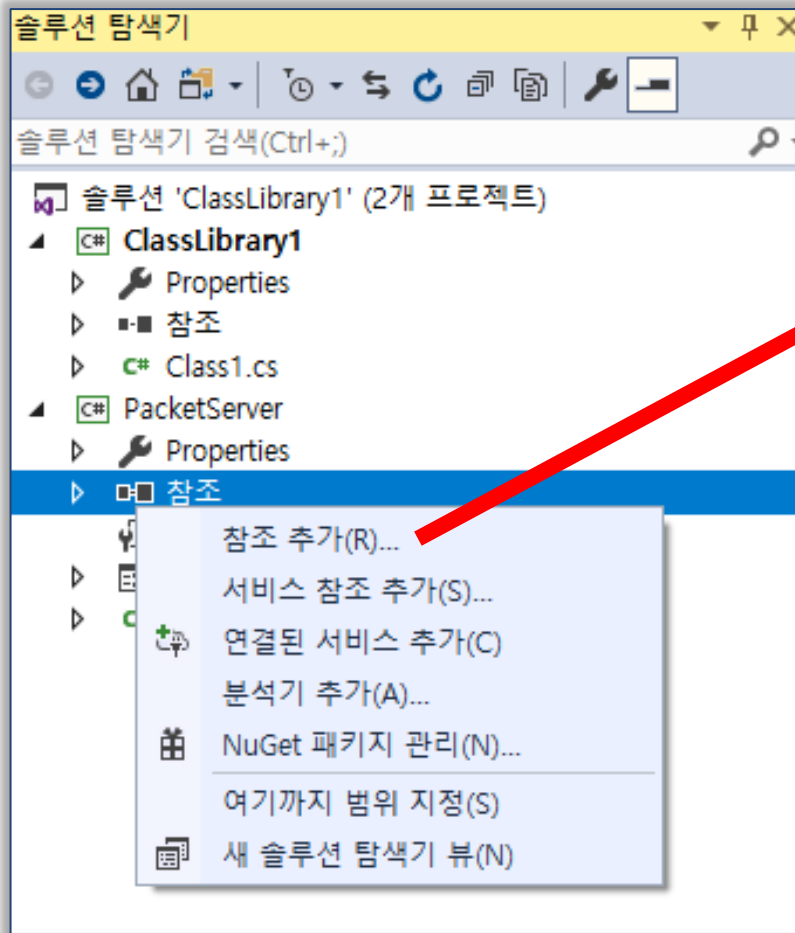
Window Forms 앱(.Net Framework)



Dock : Fill
Multiline : True

패킷 통신 실습 예시 - 서버 (cont`d)

- Packet 클래스가 정의된 클래스 라이브러리 참조 추가



패킷 통신 실습 예시 - 서버 (cont`d)

□ 네임스페이스 추가

```
using ClassLibrary1;  
using System;  
using System.Net.Sockets;  
using System.Threading;  
using System.Windows.Forms;
```

개인마다 다를 수 있음
이전에 작성한 패킷 클래스의 네임스페이스를 넣어야 함

□ 서버 멤버 변수

```
private NetworkStream m_networkstream;  
private TcpListener m_listener;  
  
private byte[] sendBuffer = new byte[1024 * 4];  
private byte[] readBuffer = new byte[1024 * 4];  
  
private bool m_bClientOn = false;  
  
private Thread m_thread;  
  
public Initialize m_initializeClass;  
public Login m_loginClass;
```

패킷 통신 실습 예시 - 서버 (con

□ RUN 메소드 작성

```
public void RUN()
{
    this.m_listener = new TcpListener(7777);
    this.m_listener.Start();

    if (!this.m_bClientOn)
    {
        this.Invoke(new MethodInvoker(delegate ()
        {
            this.txt_server_state.AppendText("클라이언트 접속 대기중\n");
        }));
    }

    TcpClient client = this.m_listener.AcceptTcpClient();

    if (client.Connected)
    {
        this.m_bClientOn = true;
        this.Invoke(new MethodInvoker(delegate ()
        {
            this.txt_server_state.AppendText("클라이언트 접속\n");
        }));
        m_networkstream = client.GetStream();

        int nRead = 0;
```

```
while (this.m_bClientOn)
{
    try
    {
        nRead = 0;
        nRead = this.m_networkstream.Read(readBuffer, 0, 1024 * 4);
    }
    catch
    {
        this.m_bClientOn = false;
        this.m_networkstream = null;
    }

    Packet packet = (Packet)Packet.Desserialize(this.readBuffer);

    switch ((int)packet.Type)
    {
        case (int)PacketType.초기화:
        {
            this.m_initializeClass =
                (Initialize)Packet.Desserialize(this.readBuffer);
            this.Invoke(new MethodInvoker(delegate ()
            {
                this.txt_server_state.AppendText("패킷 전송 성공. " +
                    "Initialize Class Data is" + this.m_initializeClass.Data + "\n");
            }));
            break;
        }
        case (int)PacketType.로그인:
        {
            this.m_loginClass =
                (Login)Packet.Desserialize(this.readBuffer);
            this.Invoke(new MethodInvoker(delegate ()
            {
                this.txt_server_state.AppendText("패킷 전송 성공. Login Class Data is"
                    + this.m_loginClass.m_strID + "\n");
            }));
            break;
        }
    }
}
}
```


□ 폼의 Load 이벤트 핸들러

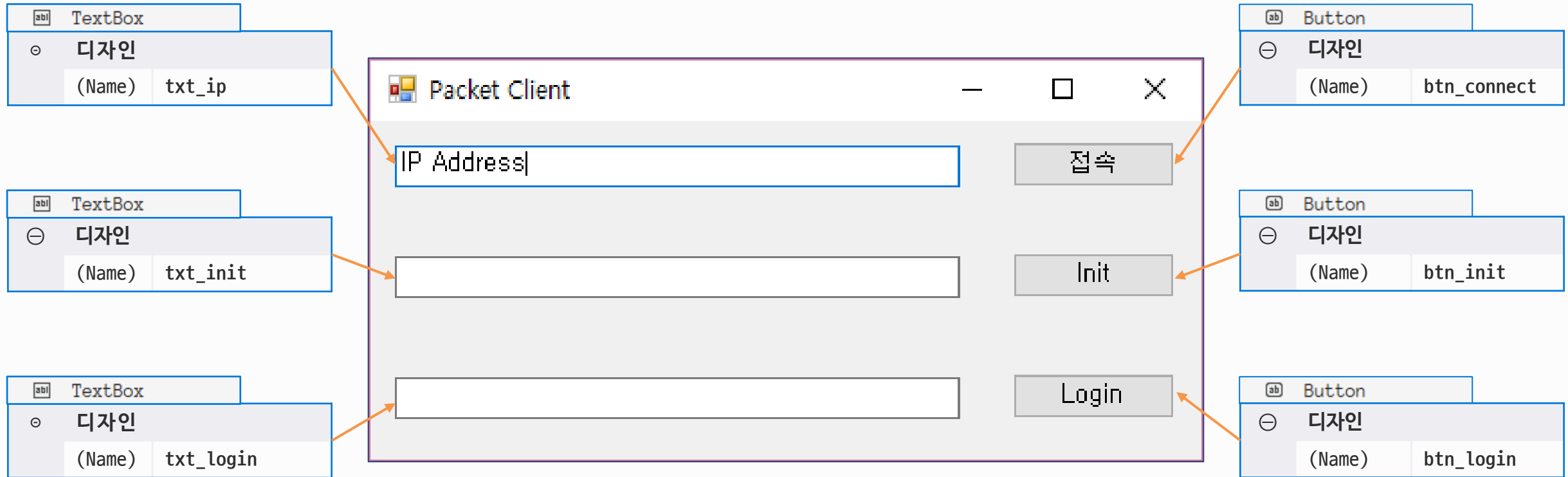
```
private void Form1_Load(object sender, EventArgs e)
{
    this.m_thread = new Thread(new ThreadStart(RUN));
    this.m_thread.Start();
}
```

□ 폼의 Closed 이벤트 핸들러

```
private void Form1_FormClosed(object sender, FormClosedEventArgs e)
{
    this.m_listener.Stop();
    this.m_networkstream.Close();
    this.m_thread.Abort();
}
```

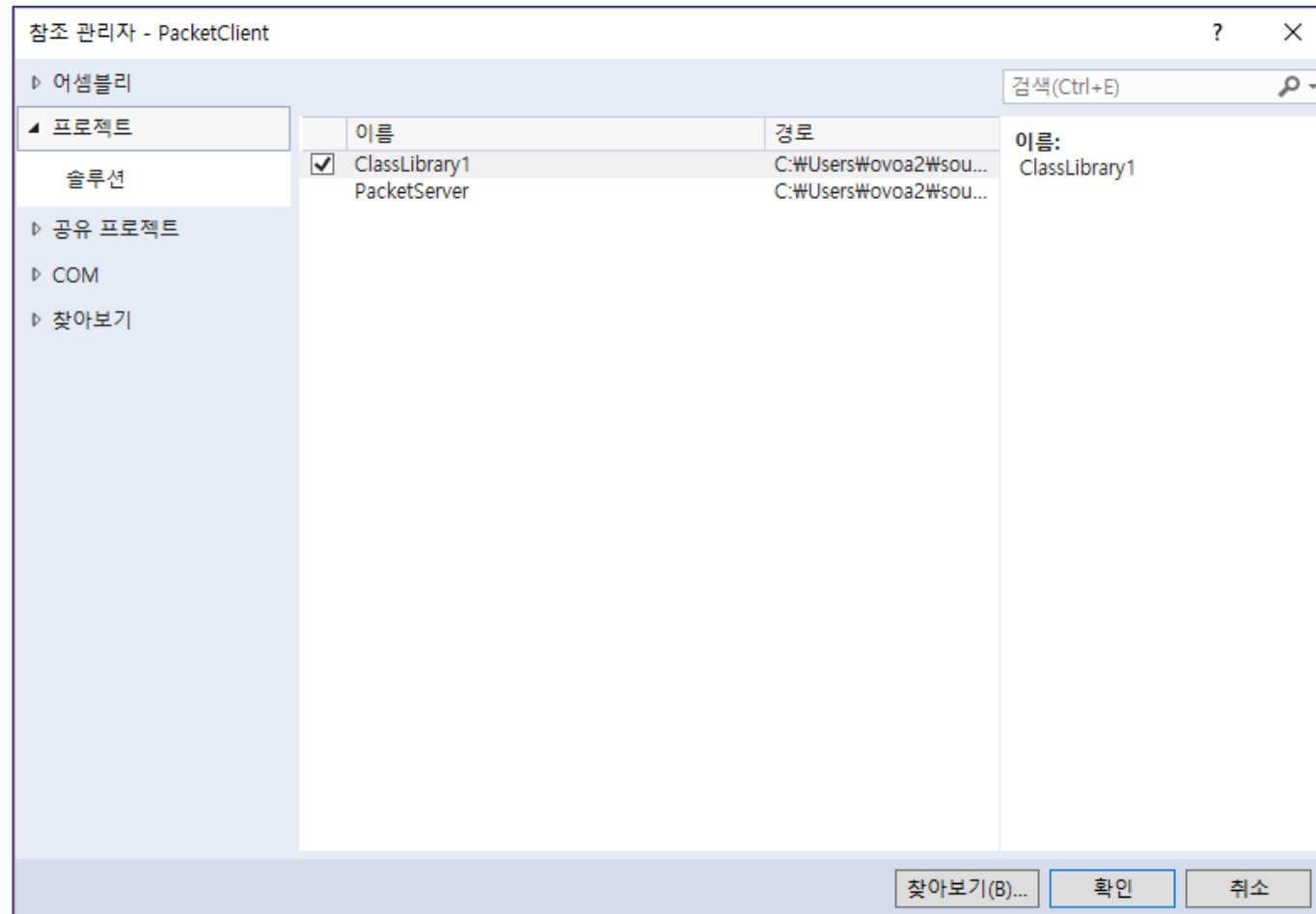
패킷 통신 실습 예시 - 클라이언트

□ 프로젝트 생성 및 폼 구성



패킷 통신 실습 예시 - 클라이언트 (cont`d)

- Packet 클래스가 정의된 클래스 라이브러리 참조 추가



패킷 통신 실습 예시 - 클라이언트 (cont`d)

□ 네임스페이스 추가

```
using ClassLibrary1;  
using System;  
using System.Net.Sockets;  
using System.Windows.Forms;
```

□ 멤버 변수 추가

```
private NetworkStream m_networkstream;  
private TcpClient m_client;  
  
private byte[] sendBuffer = new byte[1024 * 4];  
private byte[] readBuffer = new byte[1024 * 4];  
  
private bool m_bConnect = false;  
  
public Initialize m_initializeClass;  
public Login m_loginClass;
```

패킷 통신 실습 예시 - 클라이언트 (cont`d)

☐ Send() 메소드

```
public void Send()
{
    this.m_networkstream.Write(this.sendBuffer, 0, this.sendBuffer.Length);
    this.m_networkstream.Flush();

    for (int i = 0; i < 1024 * 4; i++)
    {
        this.sendBuffer[i] = 0;
    }
}
```

☐ Form Closed 이벤트 핸들러 추가

```
private void Form1_FormClosed(object sender, FormClosedEventArgs e)
{
    this.m_client.Close();
    this.m_networkstream.Close();
}
```

□ 접속 버튼 클릭 이벤트 등록

```
private void btn_connect_Click(object sender, EventArgs e)
{
    this.m_client = new TcpClient();
    try
    {
        this.m_client.Connect(this.txt_ip.Text, 7777);
    }
    catch
    {
        MessageBox.Show("접속 에러");
        return;
    }
    this.m_bConnect = true;
    this.m_networkstream = this.m_client.GetStream();
}
```

□ Init 버튼 클릭 이벤트 핸들러 추가

```
private void btn_init_Click(object sender, EventArgs e)
{
    if (!this.m_bConnect)
    {
        return;
    }
    Initialize Init = new Initialize();
    Init.Type = (int)PacketType.초기화;
    Init.Data = Int32.Parse(this.txt_init.Text);

    Packet.Serialize(Init).CopyTo(this.sendBuffer, 0);
    this.Send();
}
```

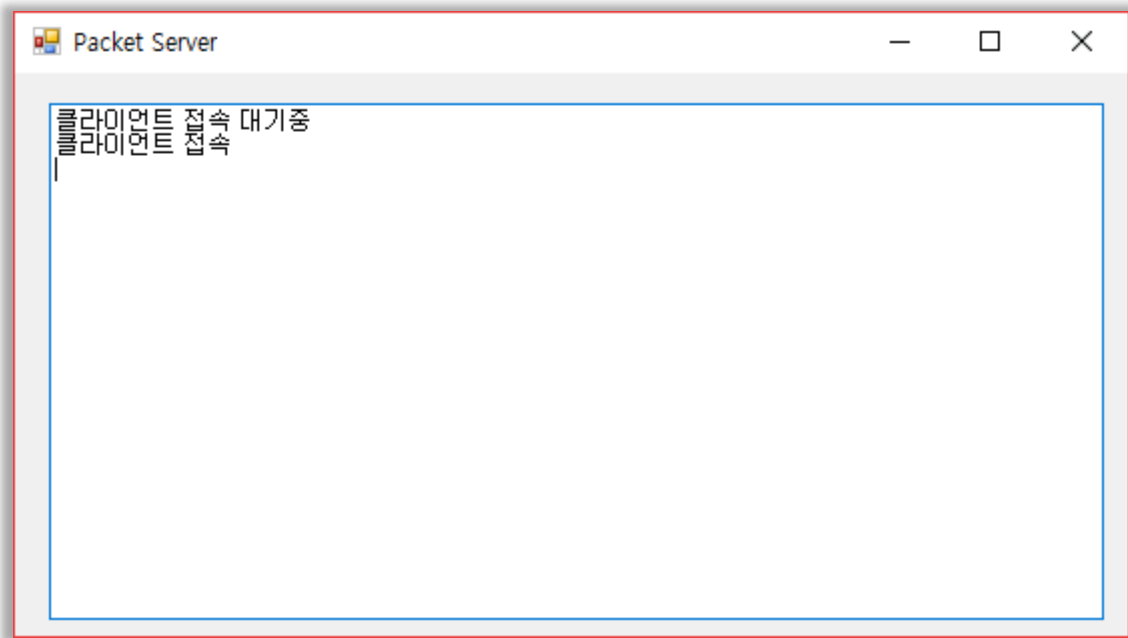
□ Login 버튼 클릭 이벤트 등록

```
private void btn_login_Click(object sender, EventArgs e)
{
    if (!this.m_bConnect)
    {
        return;
    }
    Login login = new Login();
    login.Type = (int)PacketType.로그인;
    login.m_strID = this.txt_login.Text;

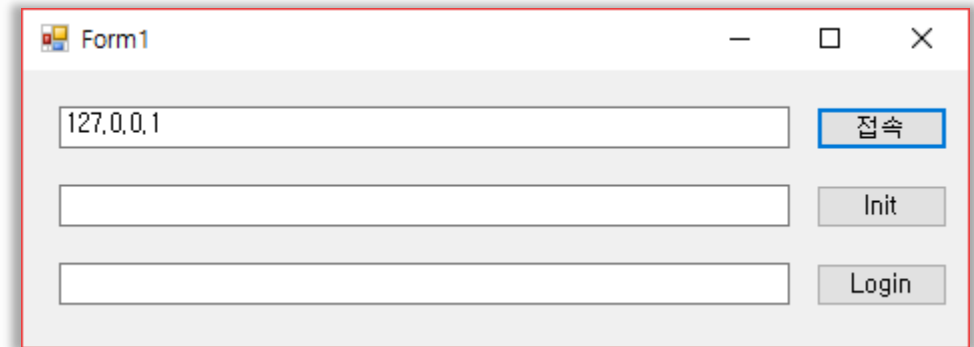
    Packet.Serialize(login).CopyTo(this.sendBuffer, 0);
    this.Send();
}
```

패킷 통신 실습 예시 (cont`d)

□ 결과 화면



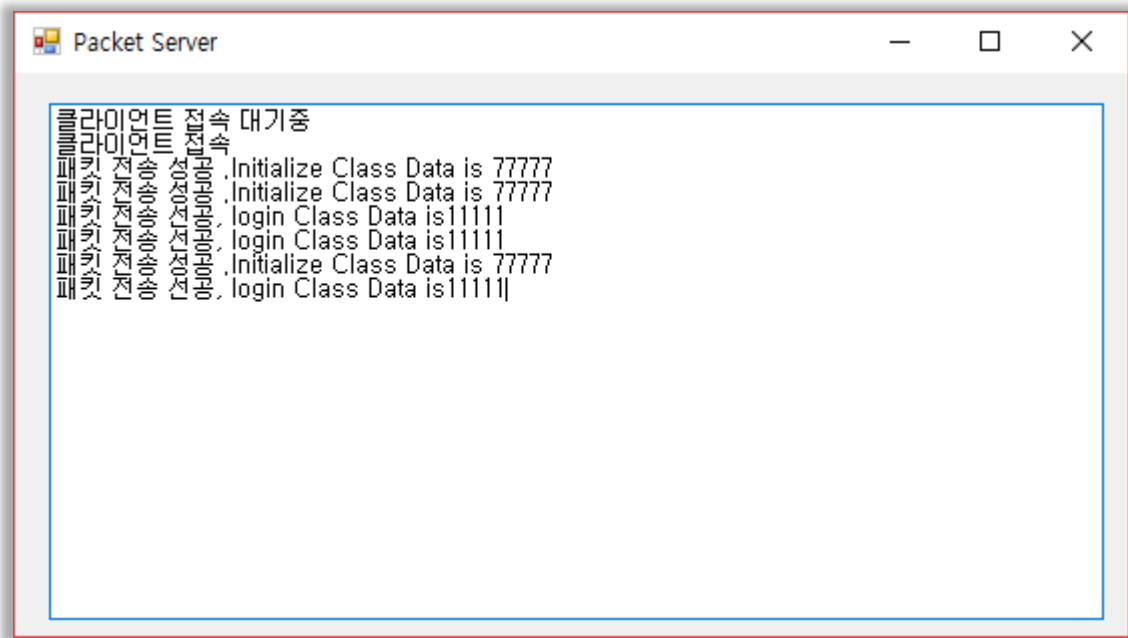
서버



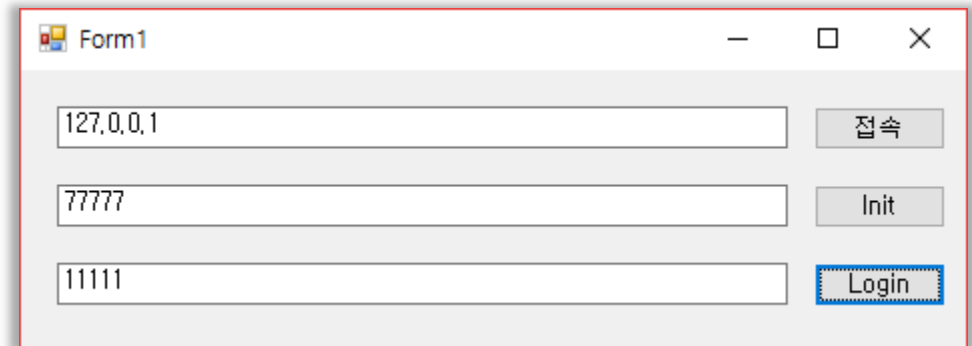
클라이언트

패킷 통신 실습 예시 (cont`d)

□ 결과 화면 (cont`d)



서버



클라이언트

관리자 권한 실행파일 만들기

- ☐ 프로젝트 속성 > 보안 > ClickOnce 속성 부여 : 체크
- ☐ 솔루션 탐색기의 Properties에서 `app.manifest` 파일이 생성되는 것을 확인 함.
- ☐ ClickOnce 속성 해제
- ☐ `app.manifest` 파일 열기
 - `<requestedExecutionLevel level="asInvoker" uiAccess="false" />`
 - `asInvoker`를 `requireAdministrator`로 수정하고, 저장함.
- ☐ 레지스트리 등록 또는 관리자 권한이 필요한 실행 파일을 만들 때에 사용함.
- ☐ 단, 디버깅을 하기 위해서는 Visual Studio를 관리자 권한으로 실행해야 함.