네트워크 프로그래밍

이것이 C#이다

Contents

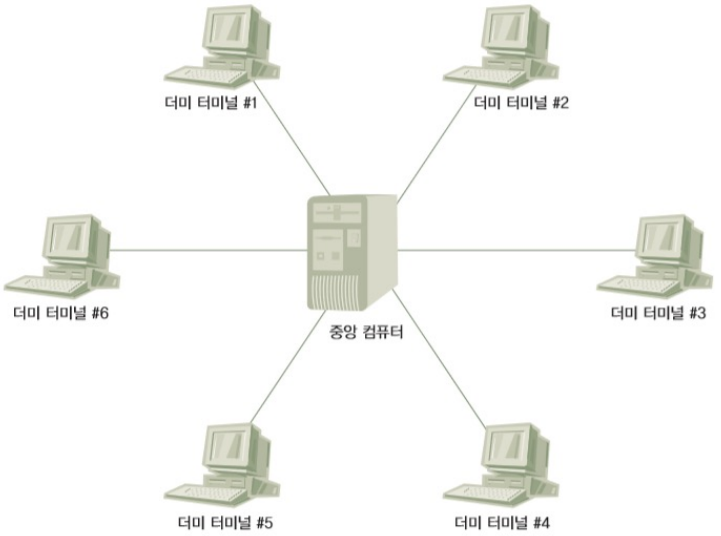
- 네트워크 프로그래밍에 앞서 알아 둬야 할 기초
- TcpListener와 TcpClient
- 흐르는 패킷
- 형식 매개 변수 제약시키기
- 일반화 컬렉션
- foreach를 사용할 수 있는 일반화 클래스

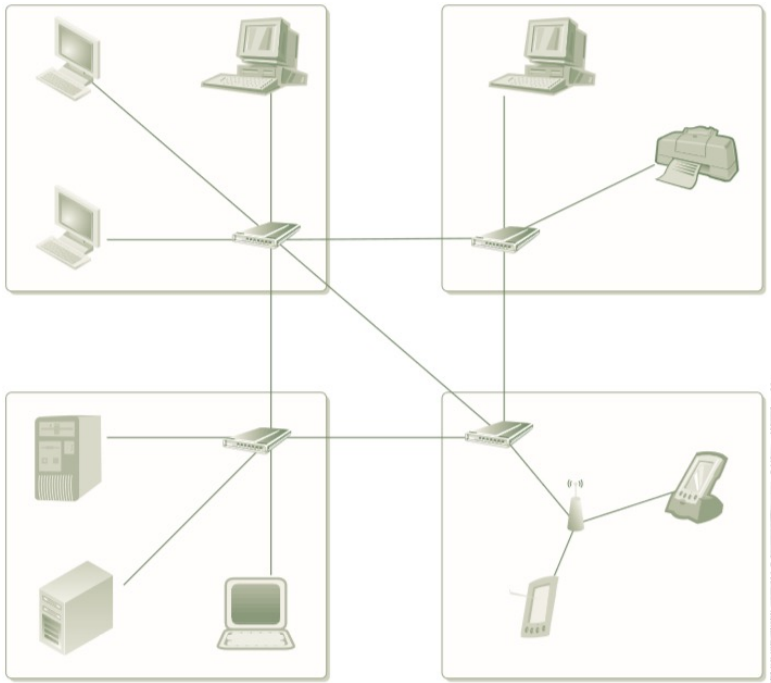
21.1 네트워크 프로그래밍에 앞서 알아 둬야 할 기초

- 인터넷의 유래
- TCP/IP 스택
- TCP/IP의 주소 체계: IP 주소
- 포트
- TCP/IP의 동작 과정

21.1.1 인터넷의 유래

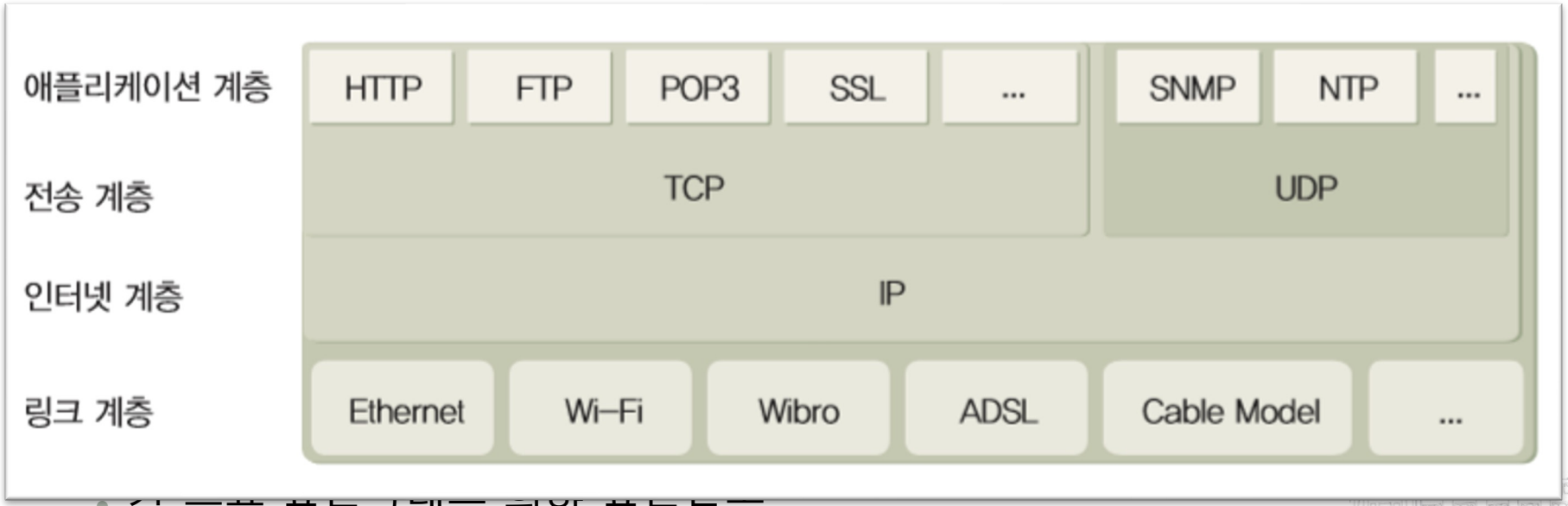
- **컴퓨터의 발전**
  - 천공카드 → 키보드와 모니터 → 멀티 프로세싱
- **중앙 컴퓨터와 더미 터미널**
  - 입력(키보드)과 출력(모니터), 중앙 컴퓨터와 통신
- **소련의 스푸트니크 위성과 DARPA 설립**
- **연구 자료의 안전하고 빠른 공유**
  - 네트워크 → DARPANET
  - 1980년대 말, '인터넷'망 형성





21.1.2 TCP/IP 스택

- **프로토콜 – 통신 규약**
- **인터넷의 표준 통신 프로토콜 - TCP/IP**
- **TCP/IP 스택 구조**
  - 링크 계층
    - 네트워크의 물리적인 연결 매체를 통해 패킷을 주고받는 작업 담당



    - 각 응용 프로그램을 위한 프로토콜
    - HTTP, FTP, SNMP 등

21.1.3 TCP/IP의 주소 체계: IP 주소

- **패킷을 보낼 곳에 대한 정보 – IP 주소**
- **IPv4**
  - 8비트 정수 4개와 (.)으로 구성(32비트)
  - 256×256×256×256=4,294,967,296 (약 40억 개)
  - 1983년에 할당 시작 → 2011년에 고갈 상태에 이름
- **IPv6**
  - 주소 길이 128비트
  - 16비트의 수 8개를 콜론(:)으로 연결
  - 약 43억×43억×43억×43억 개

21.1.4 포트

- 네트워크 패킷이 드나드는 출입구
- 부호 없는 16비트 정수 0~65535 사이의 값
- 표준 프로토콜포트 포트 번호
  - Well Known Port Number: 1~1023
  - HTTP : 80
  - HTTPS : 443
  - FTP : 21
  - Telnet : 23
  - SMTP : 25
  - IRC : 194
  - IIOP : 535

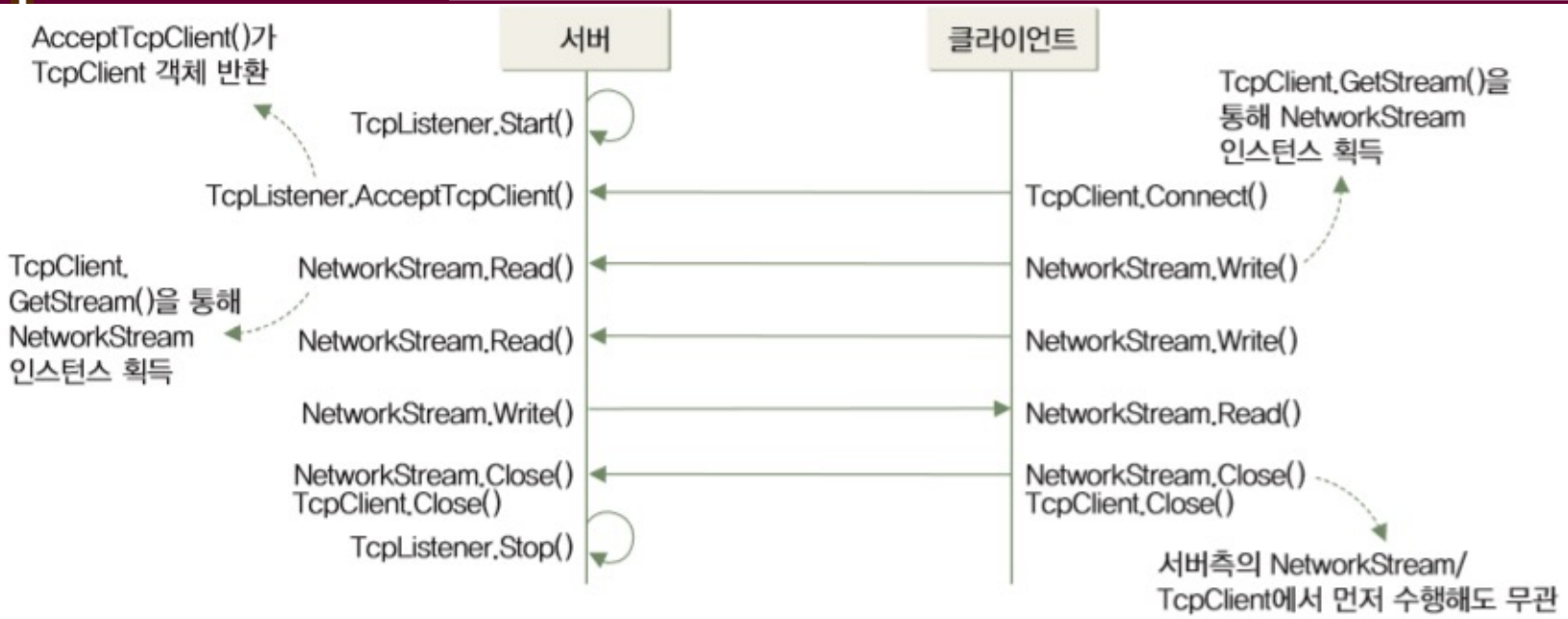
21.1.5 TCP/IP의 동작 과정

❖ 서버/클라이언트 방식의 동작

- 웹 서버와 웹 브라우저, FTP 서버와 FTP 클라이언트, SMTP 메일 서버와 메일 클라이언트

❖ 동작 과정

21.2 TcpListener와 TcpClient (1)



클래스	메소드	설명
TcpListener	Start( )	연결 요청 수신 대기를 시작합니다.
	AcceptTcpClient( )	클라이언트의 연결 요청을 수락합니다. 이 메소드는 TcpClient 객체를 반환합니다.
	Stop( )	연결 요청 수신 대기를 종료합니다.
TcpClient	Connect( )	서버에 연결을 요청합니다.
	GetStream( )	데이터를 주고받는데 사용하는 매개체인 NetworkStream을 가져옵니다.
	Close( )	연결을 닫습니다.

21.2 TcpListener와 TcpClient (2)

❖ TcpListener와 TcpClient 클래스의 사용 예제 분석

- 서버의 TcpListener를 시작 부분
- 클라이언트에서 TcpClient 객체를 생성하고 서버에 연결을 요청하는 코드
- 서버에서 AcceptTcpClient()를 호출해 클라이언트의 연결 요청 대기
- 연결 후 NetworkStream 객체를 이용하여 데이터를 읽고 쓰기

```
NetworkStream stream = client.GetStream();

int length;
string data = null;
byte[] bytes = new byte[256];

while ((length = stream.Read(bytes, 0, bytes.Length)) != 0)
{
    data = Encoding.Default.GetString(bytes, 0, length);
    Console.WriteLine(String.Format("수신: {0}", data));
    byte[] msg = Encoding.Default.GetBytes(data);

    stream.Write(msg, 0, msg.Length);
    Console.WriteLine(String.Format("송신: {0}", data));
}
```

TcpClient를 통해 NetworkStream 객체를 얻습니다.

NetworkStream.Read() 메소드는 상대방이 보내온 데이터를 읽어 들입니다. 한편, 상대와의 연결이 끊어지면 이 메소드는 0을 반환합니다. 즉, 이 루프는 연결이 끊어지기 전까지는 계속됩니다.

NetworkStream.Write() 메소드를 통해 상대방에게 메시지를 전송합니다.

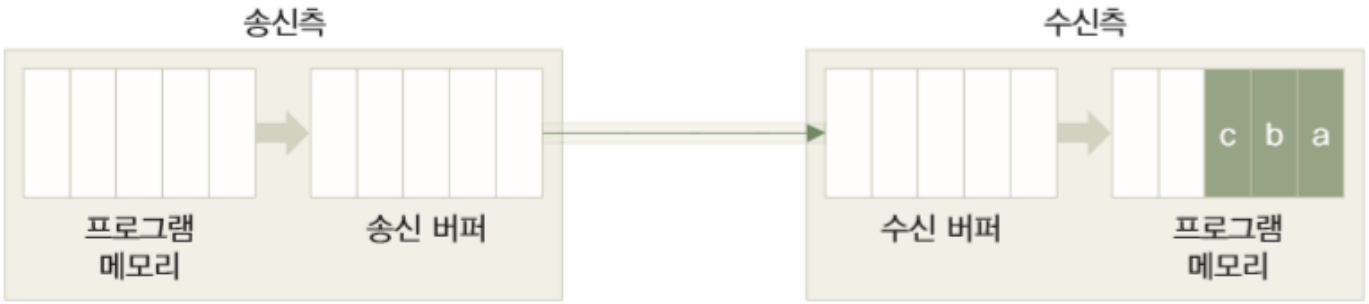
❖ 데모 예제 – EchoServer, EchoClient

21.3 흐르는 패킷

❖ TCP는 연결 지향, 흐름 지향 프로토콜

- 전기처럼 양쪽을 연결해야 하고 보내는 쪽에서 받는 쪽으로 패킷을 흘려 보냄
- 그러나 개별 패킷은 경계를 구분함
- 패킷의 일정한 흐름을 관리하기 위해 버퍼 사용

❖ 애플리케이션 간 데이터 전송 과정



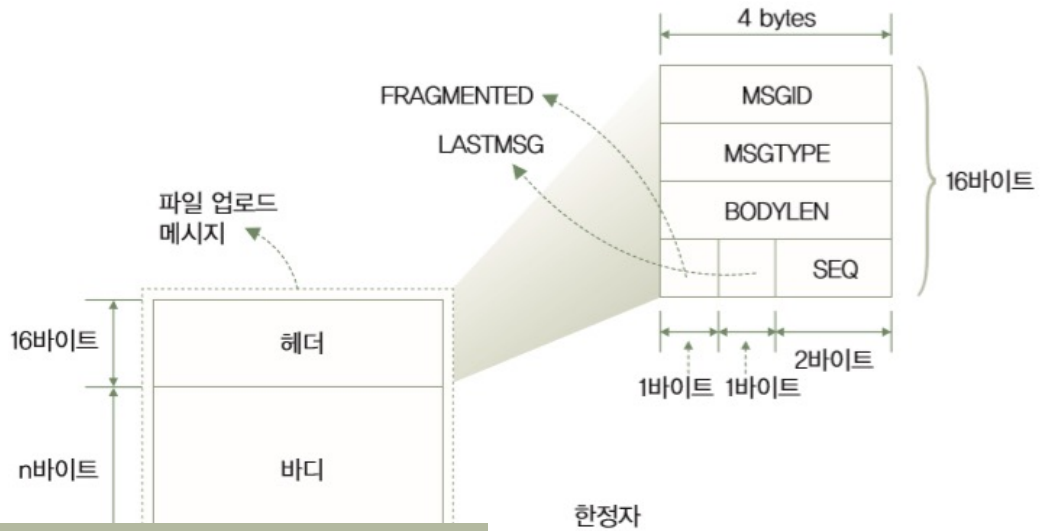
21.3.1 프로토콜 설계와 네트워크 애플리케이션 예제 (1)

- **TCP 네트워크 프로그래밍은 복잡하고 고려사항이 많다**
  - 전송 시 객체를 바이트 스트림으로 변경
  - 수신 측에서 다시 바이트 스트림을 객체로 변경
  - 수신 데이터가 정상인지 검증
  - 안정성을 위해 연결 상태도 수시로 확인
  - 서버와 클라이언트를 구현한 기능 테스트의 번거로움
- **준비 운동 – 프로토콜 설계**
  - 간단한 파일 전송 프로토콜 설계를 통한 프로토콜의 이해

21.3.1 프로토콜 설계와 네트워크 애플리케이션 예제 (2)

❖ 파일 업로드 프로토콜 (FUP)

- 헤더(메시지 속성, 고정 16바이트)와 바디(전달 데이터, 가변 길이)



❖ FUP 헤더의 속성 필드

필드 이름	크기(바이트)	설명
MSGID	4	메시지 식별 번호
MSGTYPE	4	메시지의 종류 • 0x01 : 파일 전송 요청 • 0x02 : 파일 전송 요청에 대한 응답 • 0x03 : 파일 전송 데이터 • 0x04 : 파일 수신 결과
BODYLEN	4	메시지 본문의 길이(단위: 바이트)
FRAGMENTED	1	메시지의 분할 여부 • 미분할: 0x0 • 분할: 0x1
LASTMSG	1	분할된 메시지가 마지막인지 여부 • 마지막 아님: 0x0 • 마지막: 0x1
SEQ	2	메시지의 파편 번호

한정자

21.3.1 프로토콜 설계와 네트워크 애플리케이션 예제 (3)

❖ FUP 바디 – 4가지 메시지 형식

- MSGTYPE이 파일 전송 요청(0x01)인 경우
- 파일 전송 요청에 대한 응답(0x02) 메시지의 바디 구조
- 파일 전송 데이터(0x03) 메시지
- 파일 수신 결과(0x04) 메시시

필드 이름	크기(바이트)	설명
FILESIZE	8	전송할 파일 크기(단위: 바이트)
FILENAME	BODYLEN – FILESIZE(8 byte)	전송할 파일의 이름

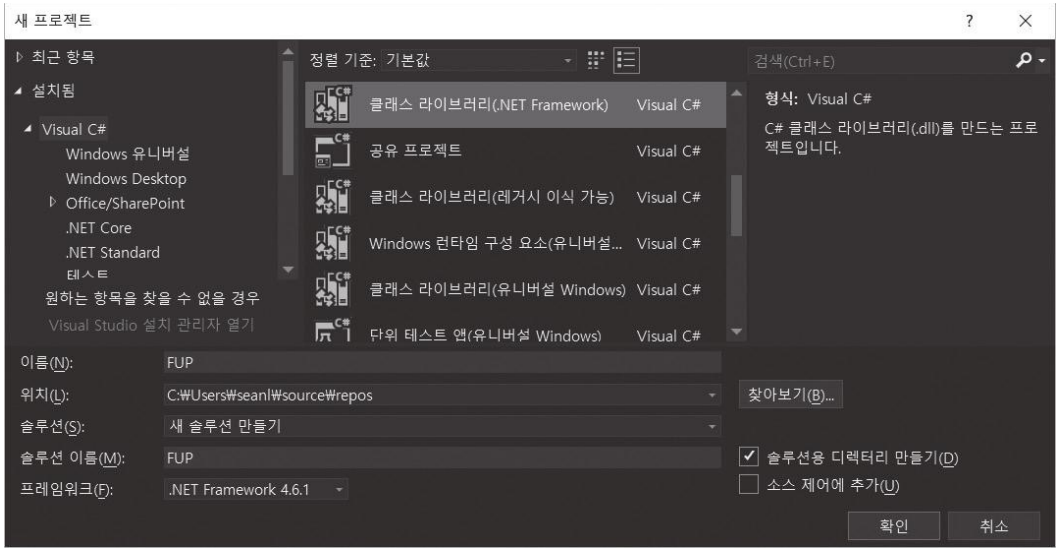
필드 이름	크기(바이트)	설명
MSGID	4	파일 전송 요청 메시지(0x01)의 메시지 식별 번호
RESPONSE	1	파일 전송 승인 여부 • 거절 : 0x0 • 승인 : 0x1

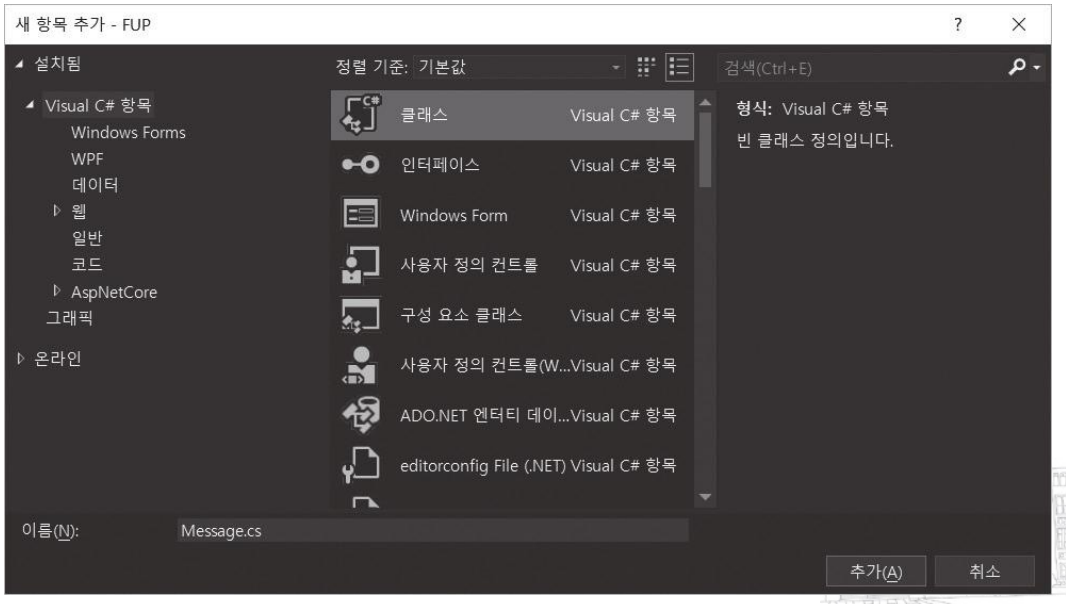
필드 이름	크기(바이트)	설명
DATA	헤더의 BODYLEN	파일 내용

필드 이름	크기(바이트)	설명
MSGID	4	파일 전송 데이터(0x03)의 식별 번호
RESULT	1	파일 전송 성공 여부 • 실패 : 0x0 • 성공 : 0x1

21.3.1 프로토콜 설계와 네트워크 애플리케이션 예제 (4)

- **서버와 클라이언트가 사용할 클래스 라이브러리 구현**
- **FUP 프로토콜 처리 라이브러리 구현 단계**
  - 프로젝트 생성
  - 클래스 추가: Message.cs, Header.cs, Body.cs, MessageUtil.cs
  - 라이브러리 빌드

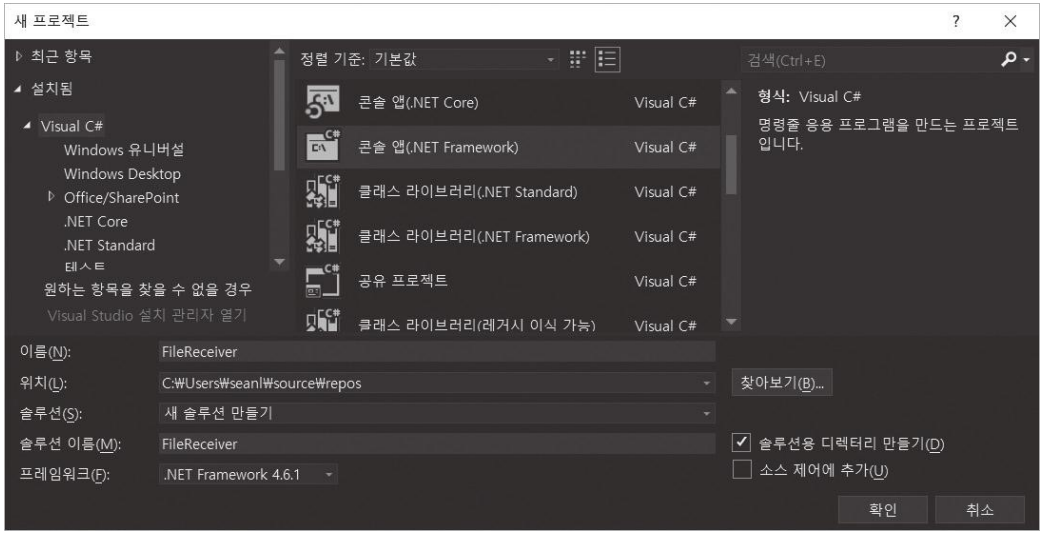




21.3.1 프로토콜 설계와 네트워크 애플리케이션 예제 (5)

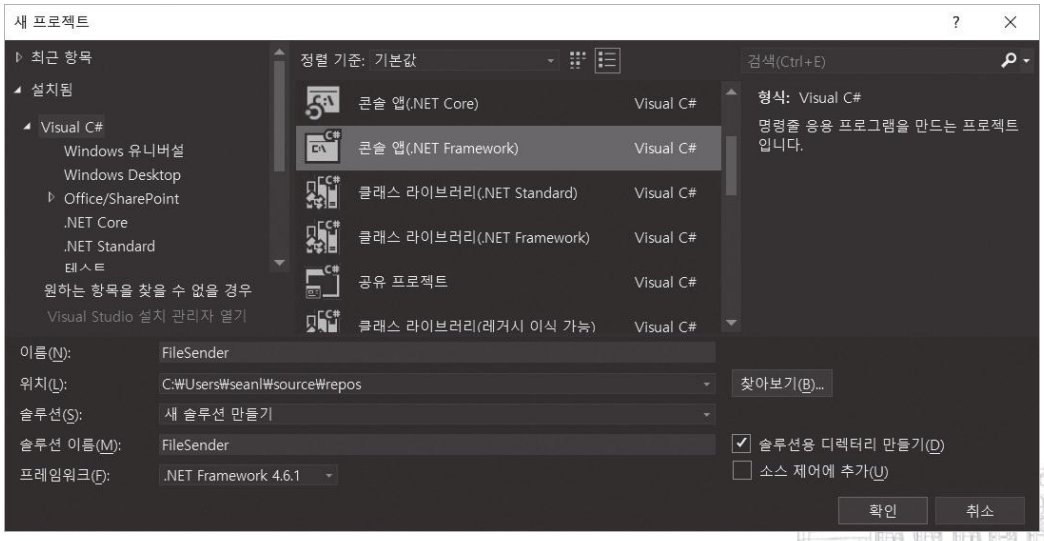
❖ 파일 업로드 서버 구현

- 프로젝트 생성
- 업로드 프로토콜 라이브러리 참조 추가
- 서버 프로그램 작성 (MainApp.cs)



❖ 클라이언트 구현

- 프로젝트 생성
- 업로드 프로토콜 라이브러리 참조 추가
- 클라이언트 프로그램 작성 (MainApp.cs)



❖ 파일 업로드 시험

Thank You !

이것이 C#이다