- **연결 방식**
  서버와 클라의 연결
  클라는 서버에 직접 연결되지 않음
  서버는 클라를 접속 수락할 때 핸들러를 만듦
  실질적으로 핸들러와 클라와의 1대1 연결
  서버는 여러 핸들러를 가질 수 있음

  (C <=> H) <= S => (H <=> C)
  이 방식으로 연결

  C <= S => C
  이 방식 아님

- **Socket**
  Disconnected
  수신 작업이 종료될 때 발생
  소켓 닫히면 수신 작업도 종료되는데 이 경우 연결이 끊긴 게 아니라 스스로 종료한 것이므로 이벤트 발생 안 함 (소켓 null인지 확인)

  ErrorOccured
  에러 발생 가능한 작업에서 try-catch 문으로 검사 후 에러 발생 시 소켓 에러를 넘김

Close는 프로그램 종료시 자동 호출
소켓 닫으면 다시 못 씀

- Client
  기본적으로 연결 상태는 None 상태
  연결을 호출하면 대기 상태로 설정

  시간 초과 딜레이 후 연결 되었는지 확인
  시간 초과 전에 연결되면 성공으로 변경

  시간 초과 후 연결 되었는지 확인할 때
  대기 상태일 때만 None으로 변경하고 연결 실패 이벤트 발생
  시간 초과 전 연결되면 성공 상태니까 아무 작업 없이 바로 리턴
  시간 초과 전 연결 성공 후 서버 닫혀서 연결 끊기거나 소켓 닫으면 다시 None으로 변경되므로
  확인할 때 아무 작업 없이 리턴

  - 연결 성공하는 경우
    None => 대기 => 성공
    시간 초과 확인 시 대기 상태 아니므로 리턴

- 연결 성공 후 닫히거나 연결 끊기는 경우
  None => 대기 => 성공 => None
  확인 시 대기 상태 아니므로 리턴

- 시간 초과
  None => 대기
  확인 시 대기 상태므로 연결 실패 이벤트 발생
  (이벤트 발생 전 None 상태로 변경)

연결 실패 시 연결을 취소하는데
이 때 소켓이 dispose 됨
그래서 소켓 재 할당해서 다시 연결 시도할 수 있도록 함

연결 끊길 때도 소켓 재할당해서 다시 사용 가능하도록 함

- Server
  전체 메시지 보낼려면 브로드캐스트로 보내면 되고
  특정 클라한테 보낼려면 Handlers에서 특정 핸들러 선택해서 Send 하면 됨

```
Handlers[index].Send("message");
```

Close 할 때 핸들러들 먼저 다 닫고 서버 닫음
핸들러를 닫아야 클라가 끊겼다는 걸 알 수 있음