

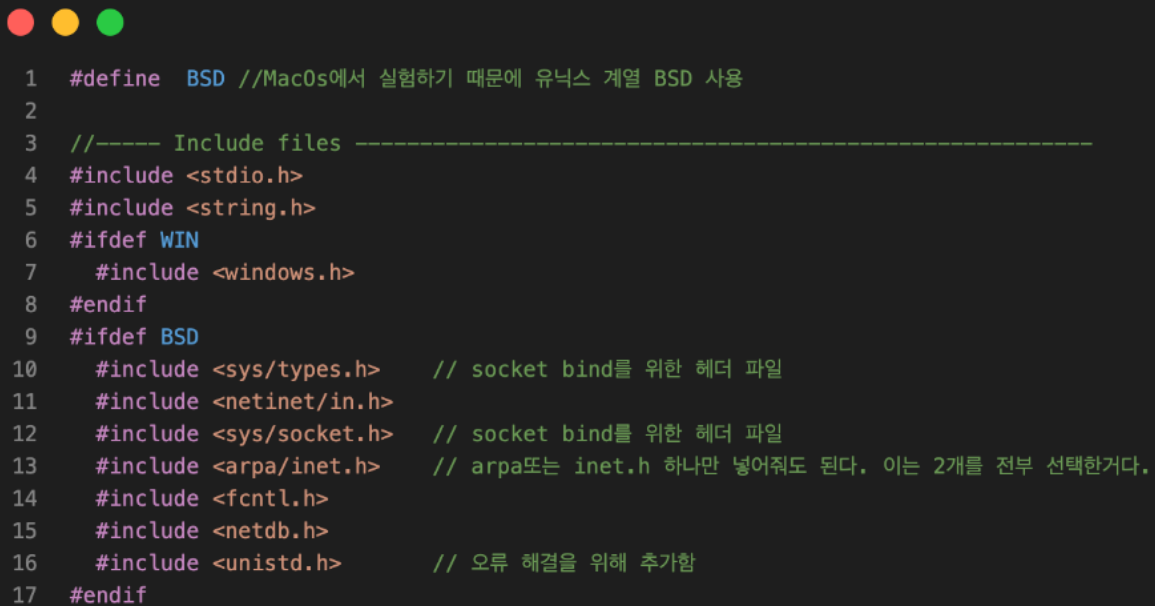
이름 : 이정환
학과 : 컴퓨터공학부
학번 : 201812163

컴퓨터 네트워크 Socket 통신

테스트 환경
OS : MacOS Monterey 12.6
Ram : 16G

1. 매크로 설명

제일 먼저 클라이언트 소켓을 살펴보면 define BSD 또는 define WIN을 설정할 수 있습니다. 이는 각 실행 환경에 따른 매크로를 정의해논 것으로 BSD와 WIN 환경에 따라 include 하는 h 파일이 달라집니다. 이외에도 각 단계별로 약간씩 다른 WIN 과 BSD 다른 점을 확인할 수 있습니다.



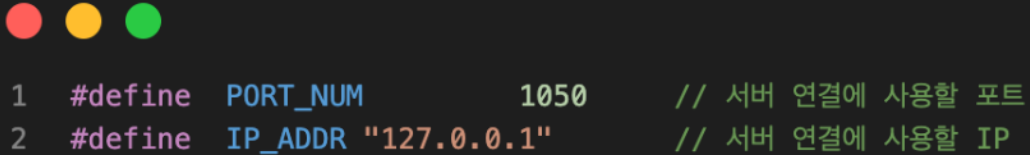
```
1  #define BSD //MacOs에서 실행하기 때문에 유닉스 계열 BSD 사용
2
3  //----- Include files -----
4  #include <stdio.h>
5  #include <string.h>
6  #ifdef WIN
7      #include <windows.h>
8  #endif
9  #ifdef BSD
10     #include <sys/types.h>    // socket bind를 위한 헤더 파일
11     #include <netinet/in.h>
12     #include <sys/socket.h>   // socket bind를 위한 헤더 파일
13     #include <arpa/inet.h>    // arpa또는 inet.h 하나만 넣어줘도 된다. 이는 2개를 전부 선택한거다.
14     #include <fcntl.h>
15     #include <netdb.h>
16     #include <unistd.h>      // 오류 해결을 위해 추가함
17 #endif
```

그림 1 BSD 매크로를 정의하고 작성하는 이미지

작성자의 PC는 MacOS 환경이기 때문에 BSD를 기준으로 작성하였습니다.
추가적으로 설명은 각 코드의 주석에 자세하게 첨부하였습니다.

* 소스 코드를 꼭 확인해주세요.

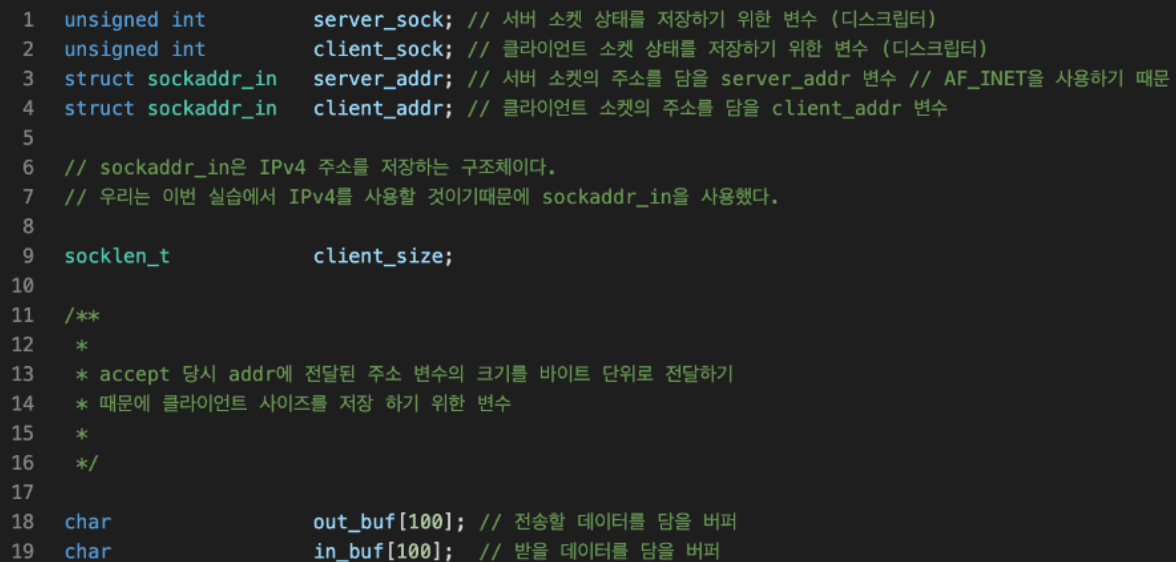
2. 함수 및 변수 설명



```
1  #define  PORT_NUM      1050      // 서버 연결에 사용할 포트
2  #define  IP_ADDR "127.0.0.1"    // 서버 연결에 사용할 IP
```

그림 2 서버 연결을 구성하기 위한 PORT_NUM / IP_ADDR 변수

그림 2은 각 접속에 사용될 포트번호와 ip 입니다. 이는 클라이언트가 차후 서버에 접속할때 입력해 줘야 하는 값 이기도 합니다.



```
1  unsigned int      server_sock; // 서버 소켓 상태를 저장하기 위한 변수 (디스크립터)
2  unsigned int      client_sock; // 클라이언트 소켓 상태를 저장하기 위한 변수 (디스크립터)
3  struct sockaddr_in server_addr; // 서버 소켓의 주소를 담을 server_addr 변수 // AF_INET을 사용하기 때문
4  struct sockaddr_in client_addr; // 클라이언트 소켓의 주소를 담을 client_addr 변수
5
6  // sockaddr_in은 IPv4 주소를 저장하는 구조체이다.
7  // 우리는 이번 실습에서 IPv4를 사용할 것이기때문에 sockaddr_in을 사용했다.
8
9  socklen_t         client_size;
10
11  /**
12   *
13   * accept 당시 addr에 전달된 주소 변수의 크기를 바이트 단위로 전달하기
14   * 때문에 클라이언트 사이즈를 저장 하기 위한 변수
15   *
16   */
17
18  char              out_buf[100]; // 전송할 데이터를 담을 버퍼
19  char              in_buf[100];  // 받을 데이터를 담을 버퍼
```

그림 3 변수 설명 이미지

그림 3은 server.c 파일안에 있는 변수를 설명해주는 주석 이미지 입니다.

서버소켓 통신 과정

socket() -> bind () -> listen () -> accept() > recv()/send() -> close()

클라이언트 소켓 통신 과정

socket() -> connect() -> send() / recv() -> close()

socket() 함수?

int socket(int domain, int type, int protocol);

소켓 함수는 도메인 , 타입 , 프로토콜 순으로 기록한다.

첫번째로 도메인은 어떤 영역에서 통신 할 건지 지정해주는 것이다.

AF_UNIX : 프로세스간 통신을 위해 사용된다

AF_INET : 물리적으로 분리된 컴퓨터간 통신에 사용되는 IPv4 통신이다.

두번째로 타입은 어떤 타입의 프로토콜을 사용할 것인지 알려준다.

SOCK_STREAM : TCP/IP를 사용한다.

SOCK_DGRAM : UDP를 사용한다.

SOCKRAW : 사용자가 직접 정의한 규약을 사용한다.

세번째는 프로토콜의 값을 결정하는 것으로 보통 0을 기록한다.

0 : 기본값 사용

IPPROTO_TCP : AF_INET과 SOCK_STREAM 유형과 사용된다.

IPPROTO_UDP : AF_UNIX와 SOCK_DGRAM 유형과 사용된다.

bind() 함수?

bind는 만들어진 빈 소켓에 IP와 포트 번호를 지정해주는 함수이다.

첫번째는 서버 소켓의 디스크립터 정보를 넘겨준다.

두번째는 할당하고자 하는 주소 정보를 가진 server_addr의 값을 넘겨준다.

세번째는 두번째 인자 값으로 전달된 server_addr의 사이즈 값을 넘겨준다.

listen() 함수?

할당 받은 서버가 수신을 대기할 대기열이다.

현재 5명으로 설정되어 있고 , 인자 값으로

첫번째는 서버의 디스크립터 정보, 두번째는 대기열 사이즈를 넘겨준다.

accept() 함수?

연결 받은 클라이언트 사이즈를 sockaddr 사이즈 만큼 할당하여 허용해준다.

첫번째 인자는 서버의 디스크립터 정보를 넘겨주고
두번째 인자 값은 연결 요청한 클라이언트의 주소 정보를 담은 주소 값을 전달한다.
세번째 값은 할당할 사이즈를 넘겨준다.

sockaddr를 사용하는 이유는 sockaddr_in은 accept시에 허용하지 않아 sockaddr로 형변환하여 넘겨줘야한다.

recv() / send() 함수?

각 recv와 send는 데이터를 받고, 전송하는 역할을 수행한다.
제일 먼저, recv는 상대방에게 데이터를 받아와 버퍼에 집어넣는 역할을 한다.

recv()의 첫번째 인자는 수신할 연결상태를 담은 소켓, 두번째 인자는 데이터를 담을 버퍼의 포인터
세번째 인자는 버퍼의 사이즈, 네번째 인자는 추가 옵션으로 필요 없을경우 0을 넣는다.

send()는 데이터를 입력한 값으로부터 받아 상대방에 전송하는 함수이다.
첫번째 인자는 데이터를 보낼 대상의 연결 상태 소켓과 두번째 값은 전송할 버퍼의 포인터, 3번째 인자는 버퍼의 사이즈

네번째 인자는 추가 옵션이고 recv와 같이 필요 없을경우 0을 넣는다.
+1이 들어간 이유는 엔터를 하면 'w0' 과 같이 버퍼에 줄바꿈이 함께 들어오기 때문이다.

클라이언트 - connect() 함수 ?

connect() ?

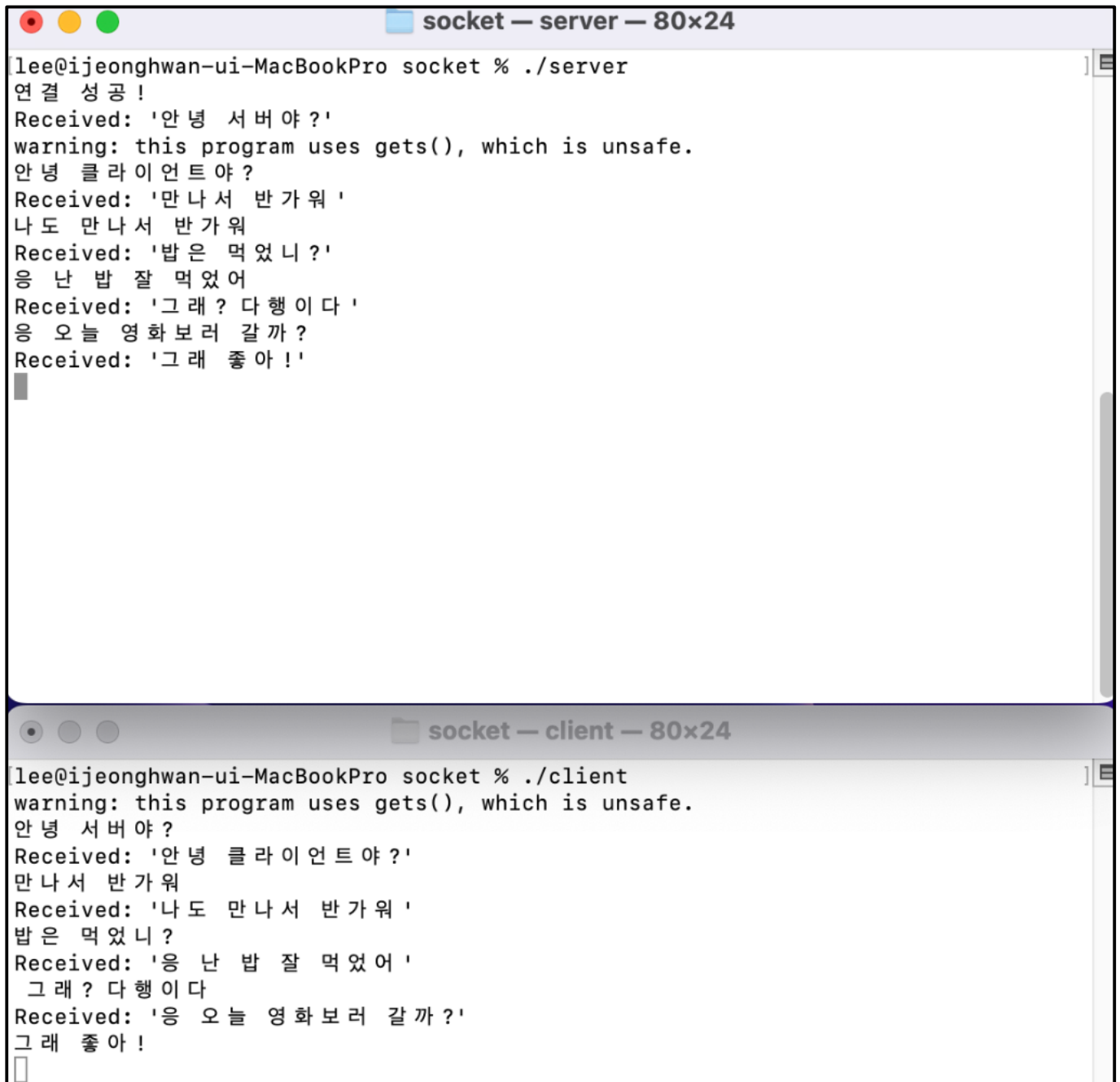
connect 함수는 일종의 서버에게 연결 요청을 하는 함수이다.
첫번째 인자로 할당 받은 소켓의 디스크립터를 넘겨주고,
두번째로 sockaddr 즉, 연결할 서버 정보가 담긴 server_addr의 값을 넘겨준다.
마지막으로 server_addr에 전달된 주소 변수 크기를 바이트 단위로 전달한다.

클라이언트에서는
connect()를 호출하면 자동적으로 ip와 포트가 할당되기 때문에
bind를 진행할 필요가 없다.

3. 소켓 통신 결과

각 통신은 클라이언트 -> 서버, 서버 -> 클라이언트와 같이 서로 차례 대로 수신/발신 하는 구조로 제작했습니다.

이미지 상단에 위치한 터미널 창은 서버, 아래 터미널 창은 클라이언트를 나타냅니다.



```
socket — server — 80x24
lee@ijeonghwan-ui-MacBookPro socket % ./server
연결 성공!
Received: '안녕 서버야?'
warning: this program uses gets(), which is unsafe.
안녕 클라이언트야?
Received: '만나서 반가워'
나도 만나서 반가워
Received: '밥은 먹었니?'
응 난 밥 잘 먹었어
Received: '그래? 다행이다'
응 오늘 영화보러 갈까?
Received: '그래 좋아!'

socket — client — 80x24
lee@ijeonghwan-ui-MacBookPro socket % ./client
warning: this program uses gets(), which is unsafe.
안녕 서버야?
Received: '안녕 클라이언트야?'
만나서 반가워
Received: '나도 만나서 반가워'
밥은 먹었니?
Received: '응 난 밥 잘 먹었어'
그래? 다행이다
Received: '응 오늘 영화보러 갈까?'
그래 좋아!
```

그림 4 서버와 클라이언트가 소켓 통신을 하는 이미지