[TCP / IP 강의] 4. 소켓 네트워크 프로그램 개발

네트워크 프로그램의 흐름

1. 기존의 망에 노드를 추가한다
   - `소켓 (ip주소, port 번호)` – 인터넷
2. 노드를 링크에 연결
   - 소켓 (ip주소, port 번호) ⬅ ➡ 인터넷
3. 통신 상태 찾아가기
   - 소켓 (ip주소, port 번호) ⬅ ➡ 인터넷 ⬅ ➡ 소켓 (ip주소, port 번호)

송신측

소켓 생성 ➡ 포트 부여 ➡ 상대편 IP/Port 주소로 연결 ➡ 통신 ➡ 종료

수신측

소켓 생성 ➡ 포트 부여 ➡ 상대편 연결 기다리기 ➡ 통신 ➡ 종료

1) 소켓 생성

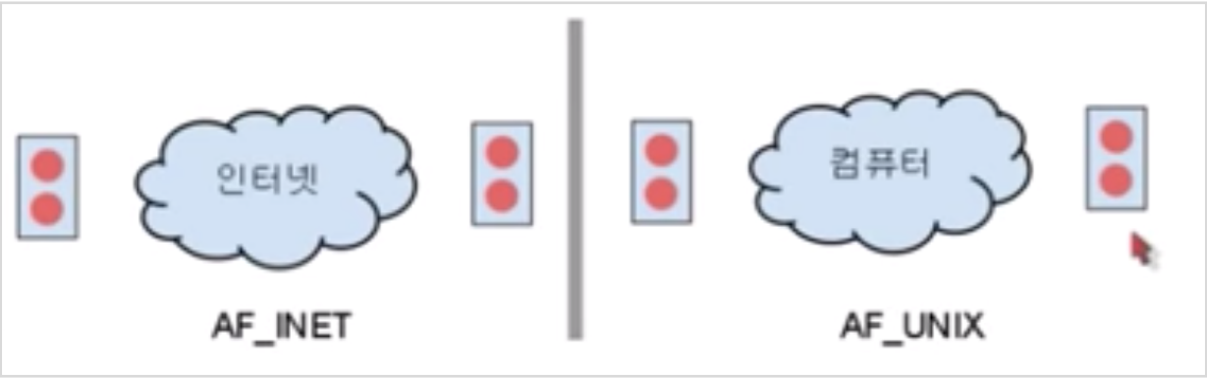
- 인터넷과 연결하기 위한 접점 소켓을 생성

```
#include <sys/types.h>
#inclide <sys/socket.h>
int soket(int domain, int type, int protocol);
```

- 소켓 init 시 받는 인자 - domain (소켓 사용 영역), type(소켓 유형), protocol (소켓 사용 프로토콜)

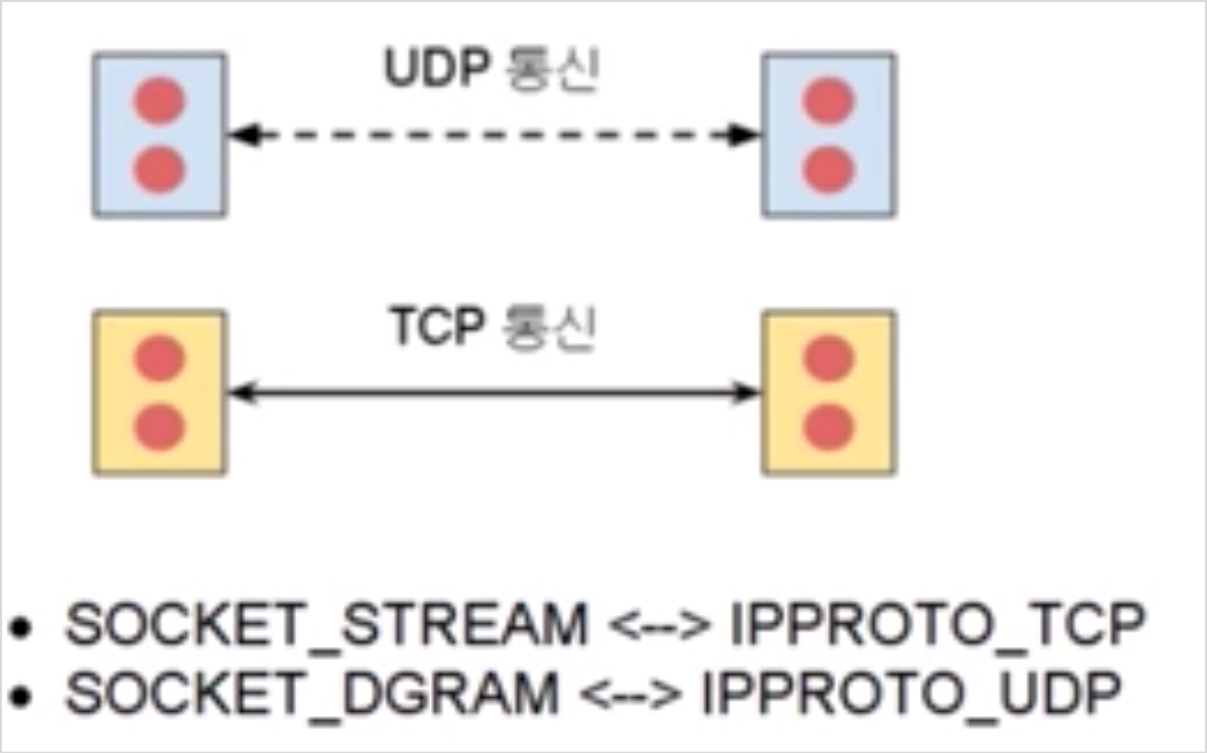
domain

- 소켓의 사용영역 정의



- 인터넷 영역의 소켓 사용 영역은 `AF_INET`

Type과 Protocol



- Type과 Protocol은 한쌍으로 움직인다.
- TCP 통신은 `SOKET_STREAM <-> IPPROTO_TCP`
- UDP 통신은 `SOKET_DGRAM <-> IPPROTO_UDP`

소켓 함수 사용의 예

```
// TCP
int fd = soket(AF_INET, SOKET_STREAM, IPPROTO_TCP)

// UDP
int fd = soket(AF_INET, SOKET_DGRAM, IPPROTO_UDP)
```

- `int fd`는 소켓을 가리키는 값 = 소켓 지정 번호

2) 소켓에 연결하기

2-1 인터넷 주소와 포트번호를 이용해서 원격 소켓에 연결

```
#include <sys/types.h>
#inclide <sys/socket.h>
int connect(int sockfd, const struct sockaddr *serv_addr, socklen_t addrlen);
```

- `sockfd` : 1로 만든 소켓 지정 번호
- `sockaddr` : IP 주소 / 포트 번호 지정

2-2 원격 소켓의 인터넷 주소 지정하기

```
struct sockaddr_in serveraddr;

server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
serveraddr.sin_family = AF_INET; // soket domain
serveraddr.sin_addr.s_addr = inet_addr("218.234.19.87") // IP 주소 지정
serveraddr.sin_port = htons(8080); // Port 번호 지정
```

2-3 원격 소켓으로 연결 요청하기

```
client_len = sizeof(serveraddr);
connect(server_sockfd, (struct sockaddr*)&serveraddr, client_len);
```

- connect 함수 호출
  - 위에 지정한 IP주소의 해당 포트 번호로 요청

3. 데이터 통신하기

- 데이터 입출력
  - 소켓 함수 : send, recv, sendto, recvfrom
  - 파일 함수 : read, write
    - 파일 입출력 함수는 윈도우 외의 OS에서 사용 가능. 윈도우는 소켓을 파일로 보지 않아서 사용 불가능

```
sszie_t write(int fd, const void *buf, size_t count);
int send(int fd, const void *msg, size_t len, int flags);

sszie_t read(int fd, const void *buf, size_t count);
int read(int fd, const void *msg, size_t len, int flags);
```

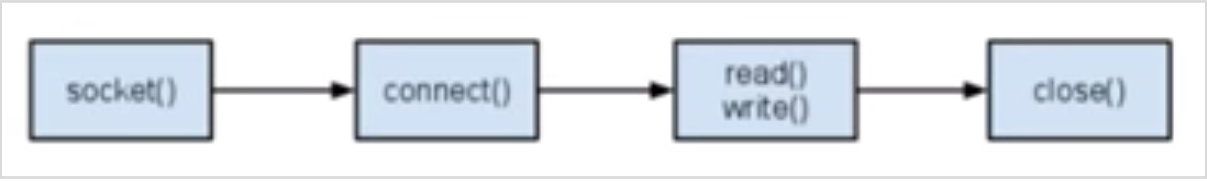
- 모두 소켓 지정 번호를 매개변수로 가짐

4. 연결 종료

```
close(int sockfd);// 그외
```

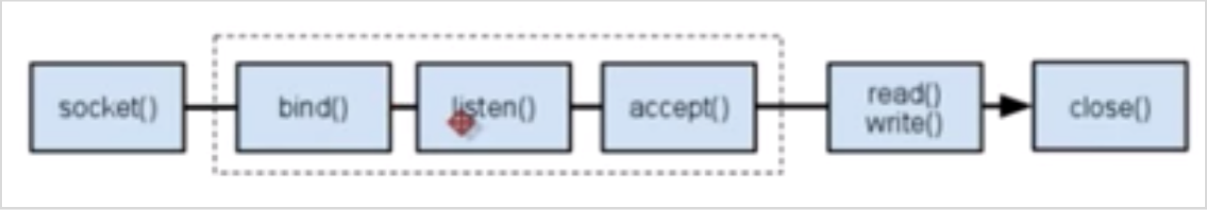
```
closesocket(SOCKET sockfd);//윈도우 전용
```

클라이언트 전체 흐름



서버 프로그램 흐름 만들기

- 클라이언트와 가장 다른 점 : 서버는 기다린다



bind 함수

```
int bind(int sockfd, struct sockaddr *myaddr, socklen)t addrlen);
```

- 소켓을 인터넷에 주소에 묶어주기 위해 사용

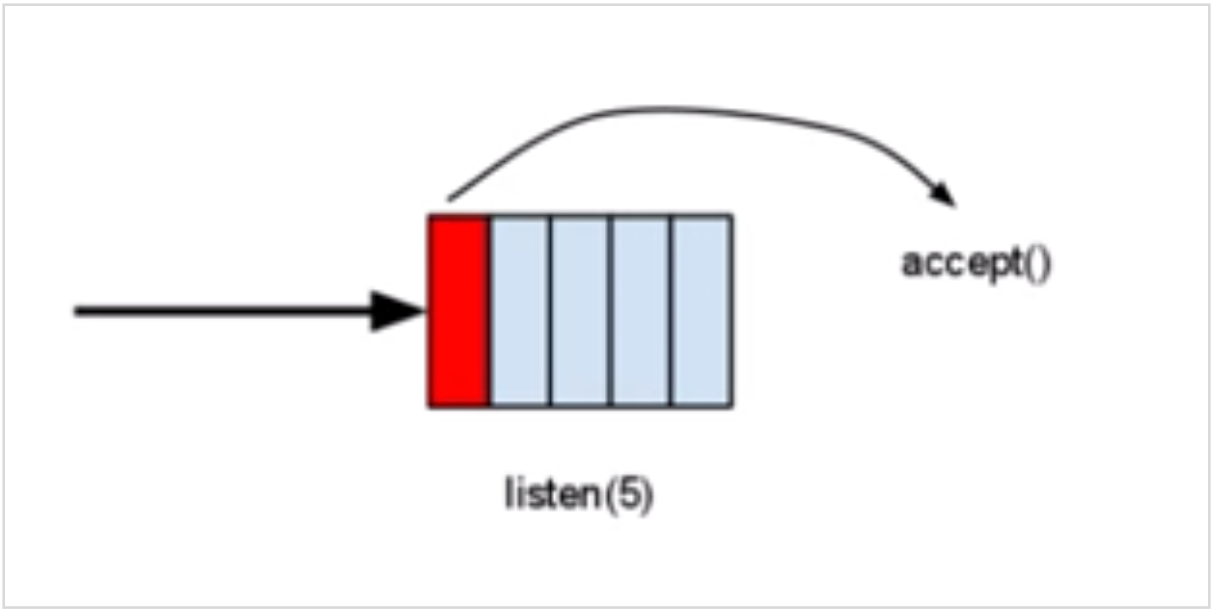
```
struct sockddr_in addr;
addr.sin_family = AF_INET;//기다릴 soket domain
addr.sin_addr.s_addr = hronl(INADDR_ANY);//기다릴 soket IP 주소 : INADDR_ANY 는 모든 주소로 기다리겠다는 뜻
addr.sin_port = htons(8080);//기다릴 포트번호

state = bind(sockfd, (struct sockaddr *)&addr, sizeof(addr));
```

listen 함수

- 수신 대기열을 만든다
  - 서버는 여러 클라이언트의 요청을 받아야 하기 때문

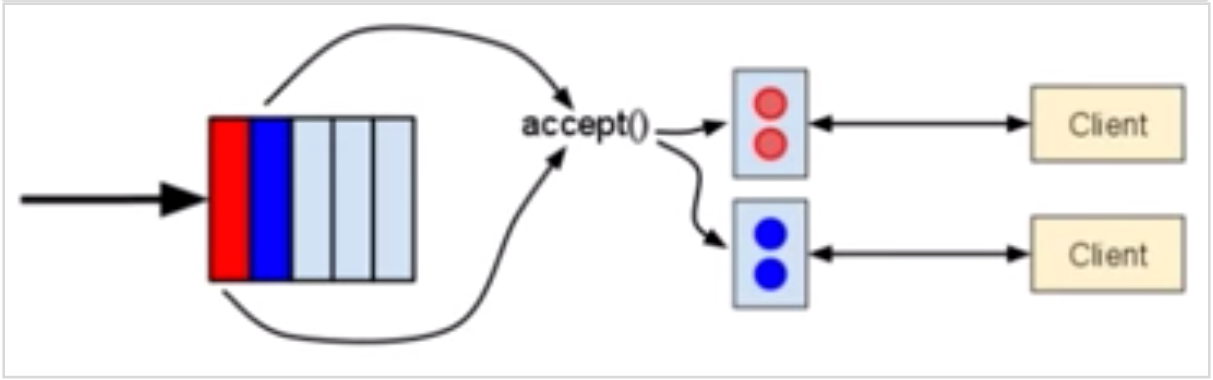
```
int listen(int queue_size);
```



accept 함수

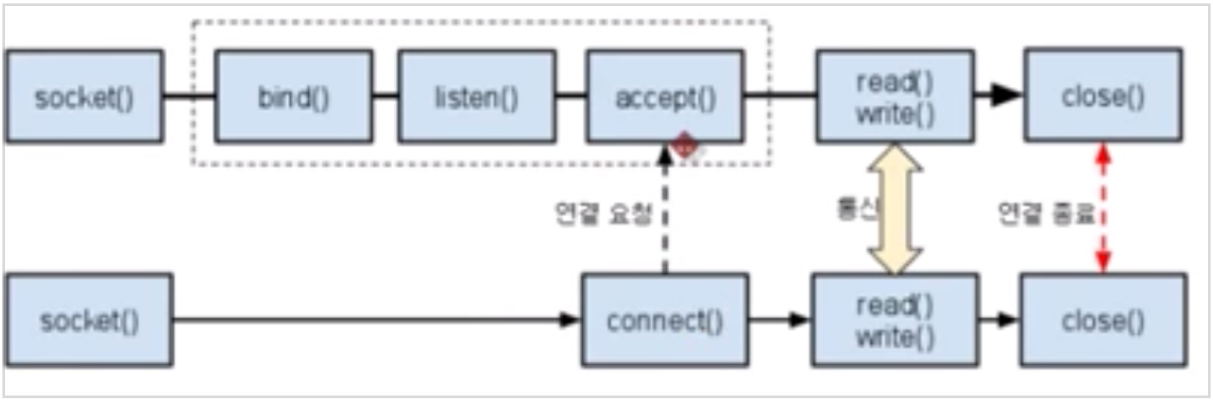
- 수신 대기열에서 연결을 가져온다

```
int accept(int s, struct sockaddr *addr, socklen_t *addrlen);
```



- accept 함수를 성공적으로 실행하면 새로운 소켓을 생성
  - 클라이언트와 직접 연결이 되는 소켓 = 연결 소켓
  - accept 의 매개변수와는 다른 소켓
    - accept 의 매개변수 = 듣기소켓
    - accept 로 생성된 소켓 = 연결 소켓
  - 동시에 두개 이상의 소켓을 다뤄야 여러 요청을 다룰 수 있음

서버 & 클라이언트 흐름의 연결



1. 서버와 클라이언트 소켓 생성
2. 서버 : `bind/listen/accept` 를 통해 데이터를 기다림
3. 클라이언트 : 서버에 `connect` 를 통해 연결 요청
4. 서버와 클라이언트 간의 연결 : `read / write`
5. 모든 통신이 끝나면 `close`

#Development/study