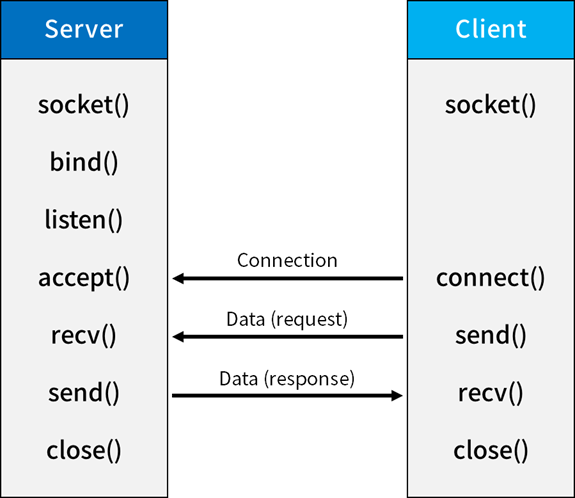
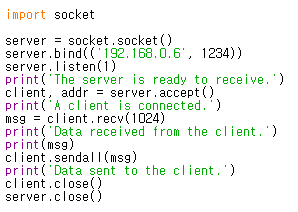
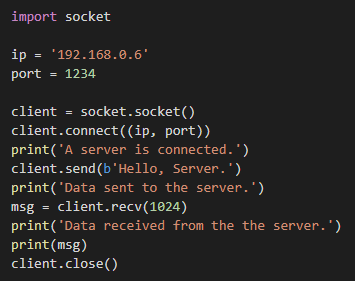
**소켓 (socket)**

회로의 [고정 부분](https://terms.naver.com/entry.nhn?docId=845142&ref=y) 끝에 있는 장치로서 이 소켓에 [플러그](https://terms.naver.com/entry.nhn?docId=832901&ref=y)(plug)를 삽입하면 회로가 연결된다.  
(1) 전기 기기용 소켓.  
(2) 마이크로 [컴퓨터](https://terms.naver.com/entry.nhn?docId=859996&ref=y)에서 IC 또는 LSI는 프린트 기판에 납땜하는 것과 ROM, PROM, CPU 등은 용도에 따라 기판에 소켓을 붙여서 바꾸어 꽂을 수 있게 한다.  
(3) [인터넷](https://terms.naver.com/entry.nhn?docId=856454&ref=y)은 전용선과 TCP/IP 통신 프로토콜을 이용하여 사용하고 있고, 우리가 쉽게 하는 [컴퓨터 통신](https://terms.naver.com/entry.nhn?docId=817099&ref=y)은 전화망, 인터넷 등 다양한 수단을 이용하여 데이터를 주고받는 통신 방식이다. 이와 같이 서로 다른 통신 환경을 이어주는 것이 소켓(socket)이다. 이를 테면 전기선과 전구를 전구 소켓이 이어주는 것과 같은 역할을 한다고 보면 된다.

[](http://blog.naver.com/PostView.nhn?blogId=hyungjungkim&logNo=221404724300&from=search&redirect=Log&widgetTypeCall=true&topReferer=http://search.naver.com/search.naver?sm%3Dtab_hty.top%26where%3Dnexearch%26query%3Dpython%2Bsocket%26oquery%3Dsocket%26tqi%3DUv7vedpySDwssaTSLLNssssst30-101029%26url%3Dhttp%3A%2F%2Fengineerk.com%2F221404724300%26ucs%3DPob63e4Wc0ms&directAccess=false)

[](http://blog.naver.com/PostView.nhn?blogId=hyungjungkim&logNo=221404724300&from=search&redirect=Log&widgetTypeCall=true&topReferer=http://search.naver.com/search.naver?sm%3Dtab_hty.top%26where%3Dnexearch%26query%3Dpython%2Bsocket%26oquery%3Dsocket%26tqi%3DUv7vedpySDwssaTSLLNssssst30-101029%26url%3Dhttp%3A%2F%2Fengineerk.com%2F221404724300%26ucs%3DPob63e4Wc0ms&directAccess=false)server.py

[](http://blog.naver.com/PostView.nhn?blogId=hyungjungkim&logNo=221404724300&from=search&redirect=Log&widgetTypeCall=true&topReferer=http://search.naver.com/search.naver?sm%3Dtab_hty.top%26where%3Dnexearch%26query%3Dpython%2Bsocket%26oquery%3Dsocket%26tqi%3DUv7vedpySDwssaTSLLNssssst30-101029%26url%3Dhttp%3A%2F%2Fengineerk.com%2F221404724300%26ucs%3DPob63e4Wc0ms&directAccess=false)client.py

소켓 통신의 기본은 통신의 주체가 되는 서버가 먼저 만들어져야 해요. 하지만, 여기에서 사용되는 **socket() 함수**는 서버나 클라이언트나 모두 같은 방식이예요.

그리고 서버는 접속에 필요한 서버 정보를 가지고 **bind() 함수**로 클라이언트에서 접속이 가능하도록 하고, **listen() 함수**로 접속 대기 상태를 유지해요.

클라이언트는 서버의 접속 정보를 가지고 **connect() 함수**로 서버에 접속하고요.

클라이언트가 서버에 접속하면, 서버의 소켓은 **accept() 함수**로 연결을 확인해요.

**socket( )** 함수는 데이터 전송에 사용되는 소켓을 생성할 때 호출하며, 소켓을 성공적으로 생성하면 소켓 디스크립터를 반환한다.

### socket( ) 함수 사용법

첫 번째 매개변수 domain은 AF\_UNIX, AF\_INET 등과 같이 사용할 [프로토콜](https://terms.naver.com/entry.nhn?docId=2271799&ref=y)의 도메인이고, 두 번째 매개변수 type은 SOCK\_STREAM, SOCK\_DGRAM 등의 서비스 유형이다. 마지막의 protocol 매개변수는 시스템에서 적절한 프로토콜을 선택해주도록 대개 0으로 지정한다. 인터넷 도메인을 의미하는 AF\_INET을 예로 들면 [연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271961&ref=y)는 TCP를 선택하고, [비연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271962&ref=y)는 UDP를 선택한다.

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

함수 앞에 표시된 헤더 파일은 socket( ) 함수를 사용하기 위한 관련 정보므로, 프로그램에 반드시 추가해야 한다. 이는 앞으로 설명할 다른 소켓 함수도 마찬가지다.

**bind() 함수**

[socket( ) 함수](https://terms.naver.com/entry.nhn?docId=2271953&ref=y)로 소켓을 생성하면 소켓 디스크립터가 반환된다. 이 디스크립터를 이용해 상대 프로세스와 통신하려면 먼저 생성된 소켓에 주소를 부여해야 한다. **bind( )** 함수가 소켓에 주소를 부여하는 기능을 수행한다. 그러면 이후에 connect( )와 [accept( ) 함수](https://terms.naver.com/entry.nhn?docId=2271956&ref=y)는 연결 설정하는 과정에서 바인드된 [소켓 주소](https://terms.naver.com/entry.nhn?docId=2271950&ref=y)를 사용한다. 전화 시스템에서 전화기 자체는 소켓으로 볼 수 있으며, 전화기에 전화번호를 부여하는 기능을 bind( ) 함수가 담당한다.  
  
AF\_UNIX 도메인에서는 파일 경로명을 이용하여 주소를 부여하고, AF\_INET 도메인에서는 호스트의 IP 주소와 [포트 번호](https://terms.naver.com/entry.nhn?docId=2271915&ref=y) 조합으로 주소를 부여한다.

### bind( ) 함수 사용법

bind( ) 함수의 첫 번째 매개변수 s는 [socket( ) 함수](https://terms.naver.com/entry.nhn?docId=2271953&ref=y)의 반환값인 소켓 디스크립터 번호고, 두 번째 매개변수 name은 바인드할 [소켓 주소](https://terms.naver.com/entry.nhn?docId=2271950&ref=y)가 표기된다. 마지막 매개변수 namelen은 name에 보관된 주소의 크기다.

#include <sys/types.h>  
#include <sys/socket.h>  
int bind(int s, const struct sockaddr \*name, socklen\_t \*namelen);

**listen( ) 함수**

**listen( )** 함수는 서버 프로그램에서 실행되며, 첫 번째 매개변수 s는 소켓에서 대기할 수 있는 클라이언트의 연결 요청 개수를 지정한다. 일반적으로 SOCK\_DGRAM을 이용한 [비연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271962&ref=y)는 이 함수를 사용하지 않고, [연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271961&ref=y)에서만 사용한다. 두 번째 매개변수 backlog 값은 보통 5로 지정한다.

#include<sys/types.h>  
#include<sys/socket.h>  
int listen(int s, int backlog);

**accept() 함수**

[연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271961&ref=y)를 지원하는 서버 프로그램이 클라이언트의 연결 요청을 받으려면 **accept( )** 함수에서 대기해야 한다. 즉, 클라이언트-서버 환경에서 서버 프로세스는 accept( ) 함수를 실행해 클라이언트의 요청을 기다리고, 클라이언트 프로세스의 connect( ) 요청이 발생하면 연결이 설정된다.

### accept( ) 함수 사용법

클라이언트로부터 연결 요청이 발생하면 둘 사이의 연결이 설정되고, 두 번째 매개변수 addr에 연결을 요청한 클라이언트의 [소켓 주소](https://terms.naver.com/entry.nhn?docId=2271950&ref=y)를 반환한다. 이 값으로 어느 클라이언트가 연결을 시도했는지 확인할 수 있고, 경우에 따라서는 클라이언트 주소를 확인해 연결 요청을 소프트웨어적으로 거부할 수도 있다.

#include<sys/types.h>  
#include<sys/socket.h>  
int accept(int s, struct sockaddr \*addr, socklen\_t \*addrlen);

성공적으로 연결되면 [socket( ) 함수](https://terms.naver.com/entry.nhn?docId=2271953&ref=y)로 생성한 원래의 소켓과는 별도의 소켓이 새로 만들어진다. 생성된 소켓의 디스크립터는 accept( ) 함수의 반환값으로 얻을 수 있으며, 클라이언트와 통신할 때는 보통 이 소켓을 사용한다.

### connect( ) 함수

[연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271961&ref=y)에서는 클라이언트 프로세스가 서버 프로세스에 연결 요청을 할 때 **connect( )** 함수를 사용한다. 두 번째 매개변수 name에 연결을 원하는 서버 프로그램의 [소켓 주소](https://terms.naver.com/entry.nhn?docId=2271950&ref=y)를 표기한다.

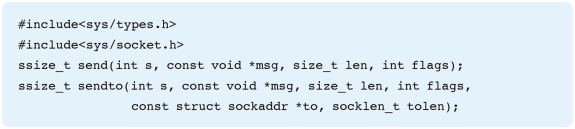
#include<sys/types.h>  
#include<sys/socket.h>  
int connect(int s, const struct sockaddr \*name, socklen\_t namelen);

**send() 함수**

소켓 연결이 이루어지면 send( ) 함수를 이용해 데이터를 전송할 수 있다. **send( )** 함수는 [연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271961&ref=y)에서의 데이터 전송에, **sendto( )** 함수는 [비연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271962&ref=y)에 사용한다.

**send( ) 함수 사용법**

send( ) 함수의 첫 번째 매개변수인 소켓 디스크립터 s는 클라이언트와 서버 사이의 연결 설정이 완료된 상태의 소켓이다. 즉, 서버의 [accept( ) 함수](https://terms.naver.com/entry.nhn?docId=2271956&ref=y)와 클라이언트의 [connect( ) 함수](https://terms.naver.com/entry.nhn?docId=2271957&ref=y)가 성공적으로 실행된 후의 소켓 디스크립터다. 전송되는 데이터는 msg에 보관된다.

[](https://terms.naver.com/imageDetail.nhn?docId=2271958&imageUrl=https://dbscthumb-phinf.pstatic.net/3578_000_1/20141203001030701_K121SZ5WI.jpg/ka8_171_i1.jpg?type%3Dm4500_4500_fst%26wm%3DN)

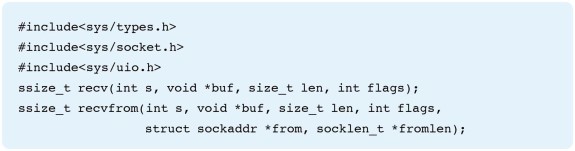
[비연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271962&ref=y)에서 사용하는 sendto( ) 함수는 연결 설정 과정을 거치지 않은 소켓 디스크립터를 사용하며, 5번째와 6번째 매개변수 to와 tolen을 이용해 전송 데이터를 수신할 [소켓 주소](https://terms.naver.com/entry.nhn?docId=2271950&ref=y)를 표기한다. 전송 데이터는 msg와 len으로 표현한다. flags는 데이터를 정상적으로 전송하면 0으로 지정하고, out-of-band 데이터면 MSG\_OOB로 지정한다.

**recv() 함수**

recv( )와 recvfrom( ) 함수를 사용하여 상대 프로세스가 전송한 데이터를 소켓을 통해 읽을 수 있다. **recv( )** 함수는 [연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271961&ref=y)에, **recvfrom( )** 함수에 [비연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271962&ref=y)에 사용한다.

### recv( ) 함수 사용법

소켓을 통해 읽은 데이터는 buf가 가리키는 공간에 저장되며, 크기는 반환값으로 얻을 수 있다. 송신자의 [소켓 주소](https://terms.naver.com/entry.nhn?docId=2271950&ref=y)가 recvfrom( ) 함수의 매개변수 from에 기록되므로 누가 전송한 데이터인지 확인할 수 있다.

[](https://terms.naver.com/imageDetail.nhn?docId=2271959&imageUrl=https://dbscthumb-phinf.pstatic.net/3578_000_1/20141203001031057_9KDSGGL5O.jpg/ka8_172_i1.jpg?type%3Dm4500_4500_fst%26wm%3DN)

**요약**

1. 프로세스 사이의 통신을 지원하는 포트 주소는 [프로토콜](https://terms.naver.com/entry.nhn?docId=2271799&ref=y)의 종류에 따라 달라진다. 대표적인 예는 호스트 내부의 프로세스끼리 통신하는 유닉스 도메인과 서로 다른 호스트 사이에서 통신하는 인터넷 도메인이다.

2. [socket( ) 함수](https://terms.naver.com/entry.nhn?docId=2271953&ref=y)로 소켓을 생성하고, [bind( ) 함수](https://terms.naver.com/entry.nhn?docId=2271954&ref=y)로 [소켓 주소](https://terms.naver.com/entry.nhn?docId=2271950&ref=y)를 부여할 수 있다

3. TCP를 사용하려면 connect( )와 [accept( ) 함수](https://terms.naver.com/entry.nhn?docId=2271956&ref=y)를 이용해 연결을 설정하고, send( )와 [recv( ) 함수](https://terms.naver.com/entry.nhn?docId=2271959&ref=y)를 이용해 데이터를 송수신한다.

4. [비연결형 서비스](https://terms.naver.com/entry.nhn?docId=2271962&ref=y)를 제공하는 UDP에서는 sendto( )와 recvfrom( ) 함수를 사용해 데이터를 송수신한다.

5. 호스트 주소 INADDR\_ANY는 서버 프로세스가 실행되는 호스트의 IP 주소를 의미한다.

6. IP 주소와 포트 [주소의 표현](https://terms.naver.com/entry.nhn?docId=2271949&ref=y)과 관련해 프로그램 개발 환경에서 개발자에게 다양한 변환 함수를 제공한다.

7. [클라이언트-서버 모델](https://terms.naver.com/entry.nhn?docId=2271945&ref=y)에서 서버 프로세스는 클라이언트보다 먼저 실행되어 accept( ) 함수에서 대기하고, 클라이언트 프로세스의 [connect( ) 함수](https://terms.naver.com/entry.nhn?docId=2271957&ref=y)와 만나 연결이 설정된다.

출처는 네이버 지식백과 ~,~