

네트워크 프로그래밍

소켓 프로그래밍

김성익(noerror@hitel.net)
2005.03.17

개요

- 네트워크 프로그래밍
- 소켓 통신이란 ?
- 네트워크 주소
IP 어드레스, 포트
- NAT *Network Address Translation*

분류

- TCP *Transmission Control Protocol*
UDP *User Datagram Protocol*
- 동기 / 비동기
- 서버 / 클라이언트

TCP

- TCP *Transmission Control Protocol*
Reliable Stream
연결 지향적 connection
데이터의 내용이 변하지 않음
보낸 크기대로 도착하지 않음
- 서버 / 클라이언트

UDP

- UDP *User Datagram Protocol*
Unreliable
비 연결 지향적 *Connectionless*
보낸 데이터 패킷 단위로 전송됨
크기, 내용이 변할 수 있음
도착하지 않을 수 있음
순서 바뀔 수 있음
중복해서 도착할 수 있음
- Peer2Peer

TCP 함수

server		client
socket bind listen		socket bind
accept		connect
	send recv	send recv
closesocket	closesocket	closesocket

TCP 서버 예제(1)

- 1. Winsock 초기화 (소켓공통)

```
WSADATA WSAData;  
  
if (WSAStartup(MAKEWORD(1,1), &WSAData) != 0)  
    return;
```

- 2. socket 생성 (소켓공통)

```
SOCKET server;  
  
server = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);  
  
if (server == INVALID_SOCKET)  
    return ;
```

TCP 서버 예제(2)

- 3. 특정 포트에 바인딩 (소켓공통)

```
SOCKADDR_IN addr;  
const int port = 1234;  
  
addr.sin_family = AF_INET;  
addr.sin_port = htons(port);  
addr.sin_addr.S_un.S_addr = htonl(INADDR_ANY);  
  
if (bind(server, (struct sockaddr *)&addr, sizeof(addr)) == INVALID_SOCKET)  
    return ;
```

htons : host byte order => tcp/ip byte order

ntohs : tcp/ip byte order => host byte order

TCP 서버 예제(3)

- 4. Listen 설정

```
if (listen(server, SOMAXCONN) == INVALID_SOCKET)
    return ;
```

- 5. 연결대기 / 연결

```
SOCKADDR_IN caddr;
int caddrlen;

caddrlen = sizeof(caddr);
client = accept(server, (struct sockaddr*)&caddr, &caddrlen);
```

TCP 서버 예제(4)

- 6. 받기 / 보내기 (echo)

```
while(1)
{
    char buffer[1024];
    int readbytes;

    readbytes = recv(client, buffer, sizeof(buffer), 0);

    if (readbytes > 0)
    {
        send(client, buffer, readbytes, 0);
    }
    else
    {
        printf("error detected (%d)\n", WSAGetLastError());
        break;
    }
}
```

- 7. 소켓닫기 (소켓공통)

```
closesocket(client);
closesocket(server);
```

TCP 클라이언트 예제(1)

- 1. Winsock 초기화
- 2. 소켓 생성
- 3. 임의의 port에 bind

```
SOCKADDR_IN baddr;  
  
baddr.sin_family = AF_INET;  
baddr.sin_port = htons(0);  
baddr.sin_addr.S_un.S_addr = htonl(INADDR_ANY);  
  
if (bind(s, (struct sockaddr *)&baddr, sizeof(baddr)) == INVALID_SOCKET)  
{  
    closesocket(s);  
    return;  
}
```

TCP 클라이언트 예제(2)

- 4. 서버에 연결하기

```
SOCKADDR_IN addr;
int readbytes;

const int _port = 1234;
const char * _ip = "127.0.0.1";

addr.sin_family = AF_INET;
addr.sin_port = htons(_port);
addr.sin_addr.S_un.S_addr = inet_addr(_ip);

if (connect(s, (struct sockaddr *)&addr, sizeof(addr)) == INVALID_SOCKET)
{
    closesocket(s);
    return;
}
```

inet_addr : 문자열을 주소 구조체로 변환

TCP 클라이언트 예제(3)

- 5. 보내기 / 받기

```
char buffer[1024];
int len;

int sendbytes;

sendbytes = send(s, buffer, len, 0);
if (sendbytes <= 0)
    printf("error detected (%d)\n", WSAGetLastError());
```

```
char buffer[1024];
int readbytes;

readbytes = recv(s, buffer, sizeof(buffer), 0);
```

- 6. 소켓 종료하기

```
closesocket(s);
```

비동기모드

- 비동기모드 설정 (소켓공통)

```
unsigned long m = 1;
ioctlsocket(s, FIONBIO, &m);
```

- 보내기 / 받기 특징

```
char buffer[1024];
int len;

int sendbytes;
for(i=0; i < len;)
{
    sendbytes = send(s, buffer+i, len-i, 0);
    if (sendbytes <= 0)
    {
        printf("error detected (%d)\n", WSAGetLastError());
        break;
    }
    i += sendbytes;
}
```

```
char buffer[1024];
int len;

int sendbytes;

sendbytes = send(s, buffer, len, 0);
if (sendbytes <= 0)
    printf("error detected (%d)\n", WSAGetLastError());
```

UDP 함수

server	client
socket bind	socket bind
sendto recvfrom	sendto recvfrom
closesocket	closesocket

UDP 서버 예제(1)

- 1. winsock 초기화
- 2. UDP socket 생성

```
SOCKET server;  
server = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);  
if (server == INVALID_SOCKET)  
    return ;
```

```
SOCKADDR_IN addr;  
const int _port = 1234;  
addr.sin_family = AF_INET;  
addr.sin_port = htons(_port);  
addr.sin_addr.S_un.S_addr = htonl(INADDR_ANY);  
if (bind(server, (struct sockaddr *)&addr, sizeof(addr)) == INVALID_SOCKET)  
    return ;
```

UDP 서버 예제 (2)

- 3. 특정 포트에 bind

```
SOCKADDR_IN addr;  
const int _port = 1234;  
  
addr.sin_family = AF_INET;  
addr.sin_port = htons(_port);  
addr.sin_addr.S_un.S_addr = htonl(INADDR_ANY);  
  
if (bind(server, (struct sockaddr *)&addr, sizeof(addr)) == INVALID_SOCKET)  
    return ;
```

- 4. 받기 / 보내기 (echo)

```
while(1)  
{  
    char buffer[1024];  
    int readbytes, addrlen;  
    SOCKADDR_IN addr;  
  
    addrlen = sizeof(addr);  
  
    readbytes = recvfrom(server, buffer, sizeof(buffer), 0, (struct sockaddr*) &addr, &addrlen);  
  
    sendto(server, buffer, readbytes, 0, (struct sockaddr*) &addr, addrlen);  
}
```

UDP 서버 예제 (3)

- 5. 소켓 닫기

```
closesocket(server);
```

UDP 클라이언트 예제(1)

- 1. winsock 초기화
- 2. UDP socket 생성
- 3. 임의의 port 에 bind

```
SOCKET s;  
SOCKADDR_IN addr;  
  
s = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);  
  
if (s == INVALID_SOCKET)  
    return;  
  
addr.sin_family = AF_INET;  
addr.sin_port = 0;  
addr.sin_addr.S_un.S_addr = htonl(INADDR_ANY);  
  
if (bind(s, (struct sockaddr *)&addr, sizeof(addr)) == SOCKET_ERROR)  
{  
    closesocket(s);  
    return;  
}
```

UDP 클라이언트 예제 (2)

- 4. 특정 서버에 보내기

```
char buffer[1024];
int len;

SOCKADDR_IN sendaddr;

memset(&sendaddr, 0, sizeof(sendaddr));
sendaddr.sin_family = AF_INET;
sendaddr.sin_port = htons(PORT);
sendaddr.sin_addr.S_un.S_addr = inet_addr(IP);

sendto(s, buffer, len, 0, (struct sockaddr *)&sendaddr, sizeof(sendaddr));
```

- 5. 받기

```
SOCKADDR_IN recvaddr;
int readbytes;

addrlen = sizeof(recvaddr);
readbytes = recvfrom(s, buffer, sizeof(buffer), 0, (struct sockaddr *)&recvaddr, &addrlen);
```

- 6. 소켓 닫기

Multiplex IO

- 필요성
- Select
- IOCP *IO Completion Port*

select

- 1. 체크할 소켓리스트 세팅
FD_SET, FD_ZERO
- 2. Select
소켓 이벤트 있을 때 까지 블록
함수 리턴되면 결과처리

```
fd_set fdset;
SOCKET s;
timeval tm;

FD_ZERO(&fdset);
FD_SET(server, &fdset);

tm.tv_sec = 0;
tm.tv_usec = 999*1000;

if (select(FD_SETSIZE, (fd_set*)&fdset, (fd_set*)0, (fd_set*)0, &tm) != SOCKET_ERROR)
{
    for(unsigned int i=0; i<fdset.fd_count; i++)
    {
        s = fdset.fd_array[i];
```

IOCP (1)

- 1. 기본 IOCP 핸들 생성

```
HANDLE g_hlocp = 0;  
g_hlocp = CreateIoCompletionPort(INVALID_HANDLE_VALUE, NULL, 0, 0);
```

- 2. IOCP 핸들에 소켓과 key 추가

```
SOCKET client;  
DWORD dwKey = (DWORD)client;  
g_hlocp = CreateIoCompletionPort((HANDLE)client, g_hlocp, dwKey, 0);
```

- 3. 읽기 operation 수행 (혹은 쓰기)

```
unsigned long readbytes;  
OVERLAPPED op;  
char buffer[1024];  
ReadFile(client, buffer, sizeof(buffer), &readbytes, &op);
```

IOCP(2)

- 4. IOCP에 쌓인 소켓 이벤트 얻기

```
unsigned long readbytes;  
unsigned long dwCompKey;  
OVERLAPPED * pOverlap;
```

```
GetQueuedCompletionStatus(g_hIoCp, &readbytes, &dwCompKey, (LPOVERLAPPED *)&pOverlap, INFINITE);
```

- 5. 가장 앞에 있는 큐가 리턴, key 값과 recv (send)된 버퍼 이용
다시 사용하기 위해선 3번 반복
- 6. 소켓 닫으면 IOCP에서 삭제

IOCP(3)

- 특징

쓰레드 풀과 응용하기 적합

NT 이상의 WIN2000에서만 작동

소켓이외의 IO 처리에 사용가능

- 기본룰

- 주의사항