네트워크

1. 클라이언트/서버(client/server)

- 컴퓨터간의 관계를 역할(role)로 구분하는 개념
- 서비스를 제공하는 쪽이 서버, 제공받는 쪽이 클라이언트가 된다.
- 제공하는 서비스의 종류에 따라 메일서버(email server), 파일서버(file server), 웹서버(web server) 등이 있다.
- 전용서버를 두는 것을 '서버기반 모델', 전용서버없이 각 클라이언트가 서버역할까지 동시에 수행하는 것을 'P2P 모델'이라고 한다.

서버기반 모델(server-based model)	P2P 모델(peer-to-peer model)
- 안정적인 서비스의 제공이 가능하다. - 공유 데이터의 관리와 보안이 용이하다. - 서버구축비용과 관리비용이 든다.	- 서버구축 및 운용비용을 절감할 수 있다. - 자원의 활용을 극대화 할 수 있다. - 자원의 관리가 어렵다. - 보안이 취약하다.

2. TCP와 UDP

▶ 소켓 프로그래밍이란?

- 소켓을 이용한 통신 프로그래밍을 뜻한다.
- 소켓(socket)이란, 프로세스간의 통신에 사용되는 양쪽 끝단(end point)
- 전화할 때 양쪽에 전화기가 필요한 것처럼, 프로세스간의 통신에서도 양쪽에 소켓이 필요하다.

▶ TCP와 UDP

- TCP/IP프로토콜에 포함된 프로토콜. OSI 7계층의 전송계층에 해당

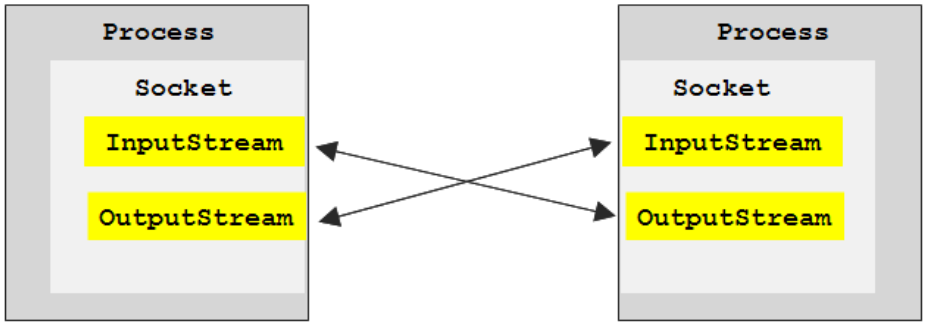
항목	TCP	UDP
연결방식	.연결기반(connection-oriented) - 연결 후 통신(전화기) - 1:1 통신방식	.비연결기반(connectionless-oriented) - 연결없이 통신(소포) - 1:1, 1:n, n:n 통신방식
특징	.데이터의 경계를 구분안함 (byte-stream) .신뢰성 있는 데이터 전송 - 데이터의 전송순서가 보장됨 - 데이터의 수신여부를 확인함 (데이터가 손실되면 재전송됨) - 패킷을 관리할 필요가 없음 .UDP보다 전송속도가 느림	.데이터의 경계를 구분함.(datagram) .신뢰성 없는 데이터 전송 - 데이터의 전송순서가 바뀔 수 있음 - 데이터의 수신여부를 확인안함 (데이터가 손실되어도 알 수 없음) - 패킷을 관리해주어야 함 .TCP보다 전송속도가 빠름
관련 클래스	.Socket .ServerSocket	.DatagramSocket .DatagramPacket .MulticastSocket

2-1. TCP소켓 프로그래밍

- 클라이언트와 서버간의 1:1 소켓 통신.
- 서버가 먼저 실행되어 클라이언트의 연결요청을 기다리고 있어야 한다.

1. 서버는 서버소켓을 사용해서 서버의 특정포트에서 클라이언트의 연결요청을 처리할 준비를 한다.
2. 클라이언트는 접속할 서버의 IP주소와 포트정보로 소켓을 생성해서 서버에 연결을 요청한다.
3. 서버소켓은 클라이언트의 연결요청을 받으면 서버에 새로운 소켓을 생성해서 클라이언트의 소켓과 연결되도록 한다.
4. 이제 클라이언트의 소켓과 새로 생성된 서버의 소켓은 서버소켓과 관계없이 1:1통신을 한다.

Socket - 프로세스간의 통신을 담당하며, InputStream과 OutputStream을 가지고 있다.
이 두 스트림을 통해 프로세스간의 통신(입출력)이 이루어진다.
ServerSocket - 포트와 연결(bind)되어 외부의 연결요청을 기다리다 연결요청이 들어오면,
Socket을 생성해서 소켓과 소켓간의 통신이 이루어지도록 한다.
한 포트에 하나의 ServerSocket만 연결할 수 있다.
(프로토콜이 다르면 같은 포트를 공유할 수 있다.)

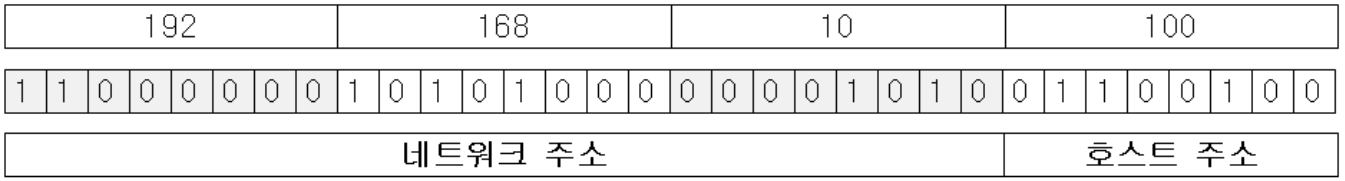


2-2. UDP소켓 프로그래밍

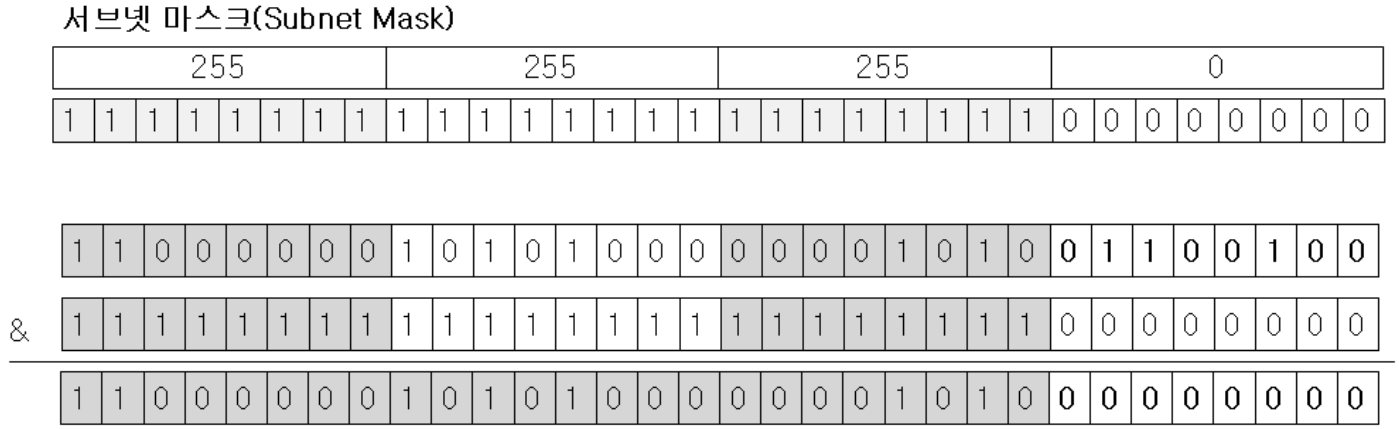
- TCP소켓 프로그래밍에서는 Socket과 ServerSocket을 사용하지만,
  UDP소켓 프로그래밍에서는 DatagramSocket과 DatagramPacket을 사용.
- UDP는 연결지향적이지 않으므로 연결요청을 받아줄 서버소켓이 필요없다.
- DatagramSocket간에 데이터(DatagramPacket)를 주고 받는다.

3. IP주소(IP address)

- 컴퓨터(host, 호스트)를 구별하는데 사용되는 고유한 주소값
- 4 byte의 정수로 'a.b.c.d'와 같은 형식으로 표현.(a,b,c,d는 0~255의 정수)
- IP주소는 네트워크주소와 호스트주소로 구성되어 있다.



- 네트워크주소가 같은 두 호스트는 같은 네트워크에 존재한다.
- IP주소와 서브넷마스크를 '&'연산하면 네트워크주소를 얻는다.



3-1. InetAddress

- IP주소를 다루기 위한 클래스

메서드	설명
byte[] getAddress()	IP주소를 byte배열로 반환한다.
static InetAddress[] getAllByName(String host)	도메인명(host)에 지정된 모든 호스트의 IP주소를 배열에 담아 반환한다.
static InetAddress getByAddress(byte[] addr)	byte배열을 통해 IP주소를 얻는다.
static InetAddress getByName(String host)	도메인명(host)을 통해 IP주소를 얻는다.
String getCanonicalHostName()	FQDN(fully qualified domain name)을 반환한다.
String getHostAddress()	호스트의 IP주소를 반환한다.
String getHostName()	호스트의 이름을 반환한다.
static InetAddress getLocalHost()	지역호스트의 IP주소를 반환한다.
boolean isMulticastAddress()	IP주소가 멀티캐스트 주소인지 알려준다.
boolean isLoopbackAddress()	IP주소가 loopback 주소(127.0.0.1)인지 알려준다.

4. URL(Uniform Resource Location)

- 인터넷에 존재하는 서버들의 자원에 접근할 수 있는 주소.

  http://www.naver.com:80/

5. URLConnection

- 어플리케이션과 URL간의 통신연결을 위한 추상클래스