

1강

네트워크와 소켓 이해

1. 네트워크와 네트워킹

- 1. 네트워크와 네트워킹
- 2. Internet
- 3. 소켓이란?
- 4. 포트(Port) 번호
- 5. InetAddress 클래스

“유/무선 으로 연결되어 있는 Device들의 집합 ”

□유 / 무선 으로 연결

1. Bluetooth, Wi-Fi, RFID, 적외선 통신(IrDA) 등 근거리 무선 통신
2. WCDMA, LTE 등과 같은 이동 통신 기술
3. Ethernet, Serial 통신 등과 같은 유선통신
4. GPS

1. 네트워크

□ Device(디바이스)

1. 보통, 네트워크에 연결된 컴퓨터
2. 컴퓨터가 아닌 다른 **Devices**

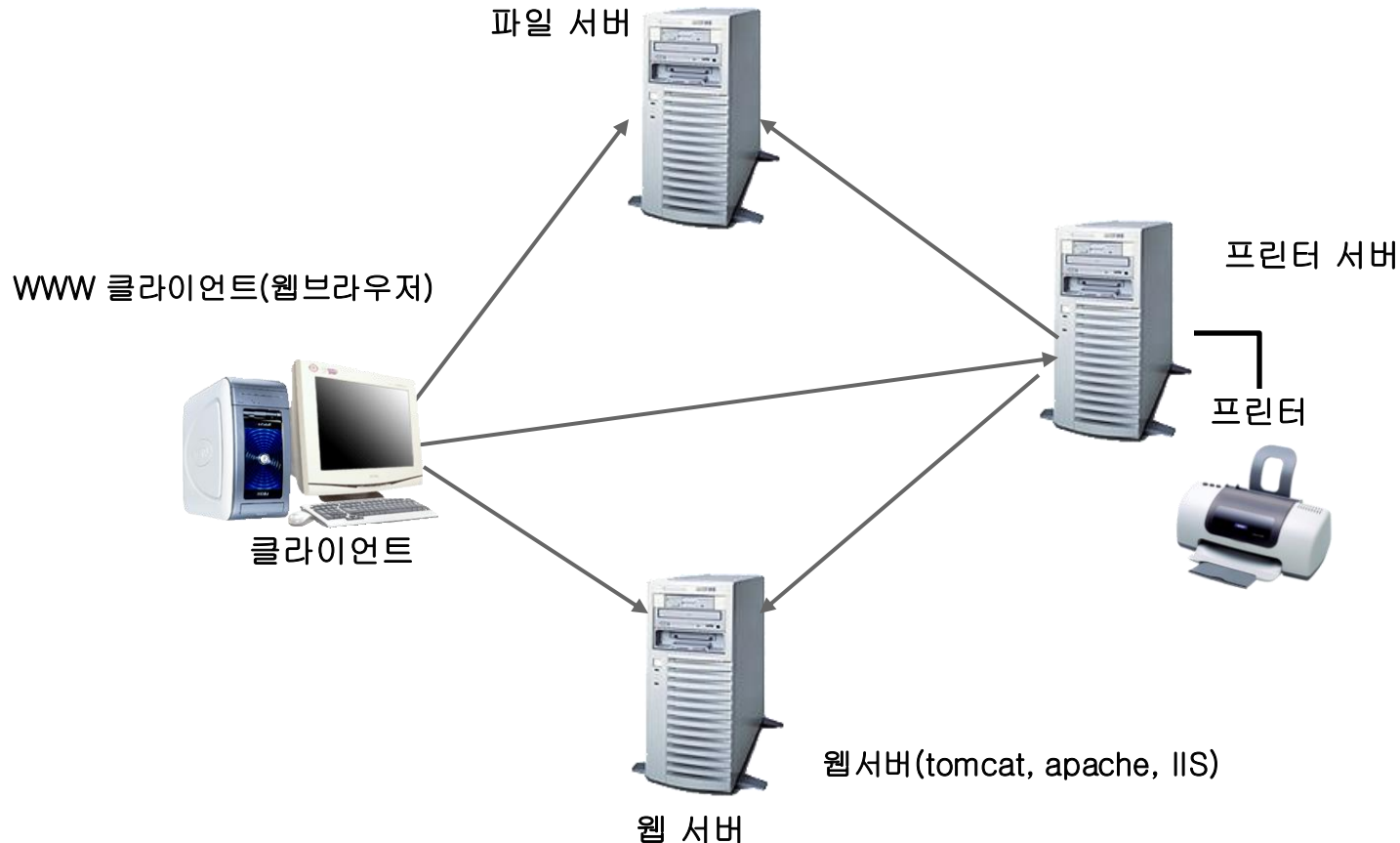
ex) 프린터, 모바일 디바이스, 가전제품, 웨어러블 컴퓨터 등 다양한 임베디드 제품들

3. 포괄적 개념으로 네트워크에 연결되어 있는 것들을 총칭해서 디바이스라 할 수 있다.



2. 네트워킹

**“ 네트워크에 연결된 디바이스들 간에 미리 약속된
프로토콜 을 사용하여 데이터를 교환 하는 것”**



1. 네트워크

□ Protocol(프로토콜)

1. 약속

2. 디바이스 상호간 데이터 통신을 위해 필요한 규약

3. 통신 규약

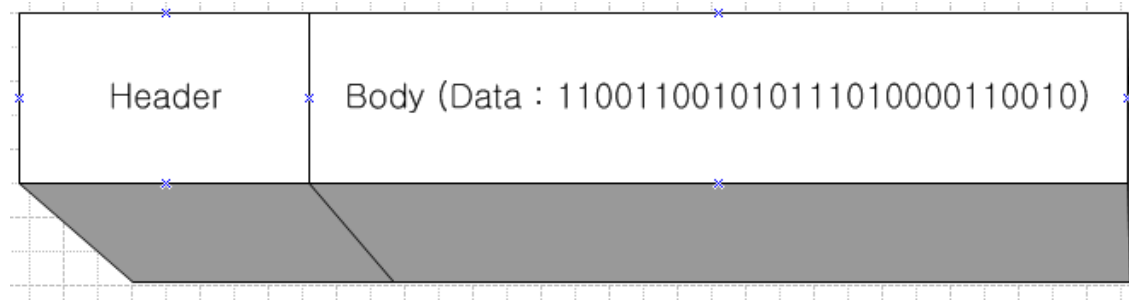
2. 네트워킹

□ 데이터 통신 방법

- 두 개의 상이한 두 디바이스가 물리적으로 떨어져 있는 경우, 유/무선을 통해 네트워크에 연결되어야 한다.
- 연결된 두 디바이스는 전류나 전파, 빛 등의 방식으로 데이터 통신을 한다.
- 이 데이터는 0/1 또는 on/off의 1bit로 표현되게 된다. (byte 단위 데이터 통신)
- 주소(address)
 - (1) 두 디바이스가 데이터 통신을 하기 위해서는 서로의 위치를 알아야 한다.
 - (2) 이 위치를 네트워크에서는 node 라고 부른다.
 - (3) 각 node 마다 고유의 주소를 가지고 있어야 한다.
- 데이터 통신을 할 때는 데이터 외에 어디로 보내야 하는 가 또는 누가 보내는 가 등의 정보를 담고 있어야 한다.

2. 네트워킹

- 실제, 네트워크를 통한 데이터 통신을 할 때는 **Packet** 을 사용하게 된다.



- (1) **Header** : 송신자/수신자 의 주소, 체크섬(checksum) 그리고 여러 제어 정보
- (2) **Body** : 전송할 데이터를 **byte** 단위로 포함한다.

1강

네트워크와 소켓 이해

2. Internet

1. 네트워크와 네트워킹
2. Internet
3. 소켓이란?
4. 포트(Port) 번호
5. InetAddress 클래스

1. 인터넷(Internet)의 이해

□ 인터넷 != WWW(World Wide Web)

인터넷 기반의 서비스

이름	프로토콜	포트	기능
WWW	HTTP	80	웹서비스
Email	SMTP/POP3/IMAP	25/110/114	이메일 서비스
FTP	FTP	21	파일 전송 서비스
DNS	DNS	23	네임서비스
NEWS	NNTP	119	인터넷 뉴스 서비스

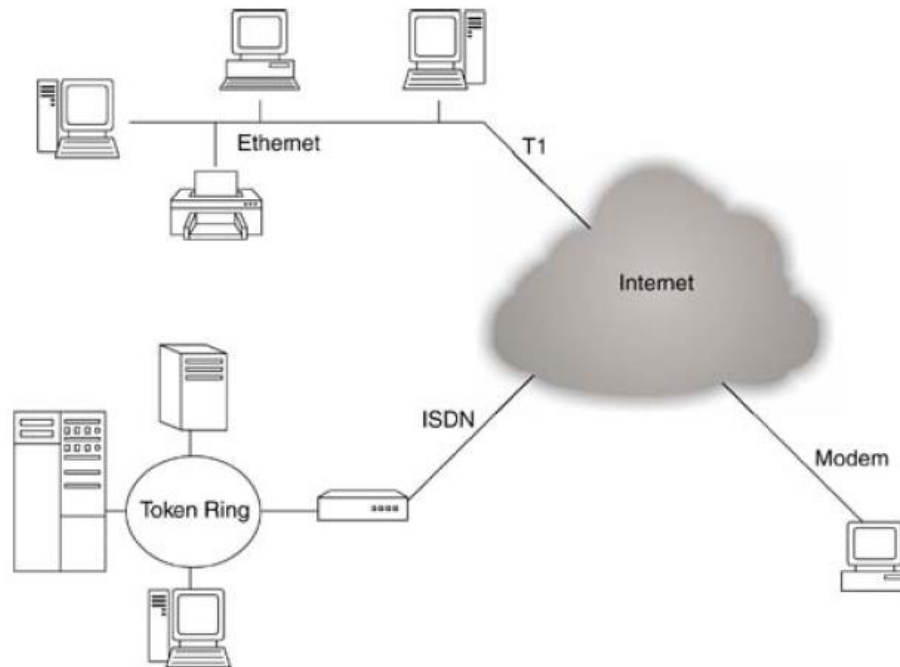
□ 인터넷 (Internet)

TCP/IP 기반의 네트워크가 전 세계적으로 확대되어 하나로 연결된 네트워크들의
네트워크 (네트워크의 결합체)

1. 인터넷(Internet)의 이해

□ 인터넷 역사

- 미국방성의 최초의 연구 목적의 네트워크 ARPANET(1969년) 이 시초
- 미 과학재단 네트워크인 NSFNET이 연결(1986년)
- 일반 산업 목적의 네트워크들이 연결(1990년 이후)



1. 인터넷(Internet)의 이해

□ OSI 7계층과 TCP / IP 4계층

- 하드웨어, 운영체제, 접속 매체와 관계없이 동작할 수 있는 개방형 구조
- OSI 7 계층에서 4계층으로 단순화.

OSI 7계층	TCP/IP 4계층	
응용 계층 표현 계층 세션 계층	응용 계층	네트워크를 사용하는 WWW, FTP, 텔넷, SMTP 등의 응용 프로그램으로 구성.
전송 계층	전송 계층	도착지까지 데이터를 전송 각각의 시스템을 연결 TCP 프로토콜을 이용하여 데이터를 전송
네트워크 계층	인터넷 계층	데이터를 정의 및 경로 지정 정확한 라우팅을 위해 IP 프로토콜을 사용 IP 주소가 위치하는 계층
데이터 링크 계층 물리 계층	링크 계층	물리적 계층 즉 이더넷 카드와 같은 하드웨어

2. 계층(Layer)과 프로토콜

□ Link 계층(Link Layer)

- Host(호스트)간의 네트워크를 통한 데이터 통신을 위한 물리적 연결(유/무선)에 대한 표준
- LAN, WAN, MAN과 같은 네트워크 구성을 정의
- 상위 계층인 Internet 계층에서 형성된 Packet(패킷) 을 전기신호 또는 광 신호로 바꾸어 전달하는 역할을 담당한다.
- 응용(Application) 개발자가 직접 접근할 수 있는 계층이 아니고 보통 네트워크 장비나 드라이버 개발자 들이 관심을 가지는 계층이다.

2. 계층(Layer)과 프로토콜

□ Internet 계층 (Internet Layer)

- Link 계층을 통해 물리적으로 연결된 각 Host(호스트) 간의 Packet(패킷)의 전달 경로를 결정한다.
- IP(Internet Protocol)은 인터넷 계층에 존재하는 프로토콜이다.
- IP 프로토콜에는 IP 주소(Address)를 부여하는 방법과 체계를 정의하고 있다.
- IP 프로토콜은 IP 주소를 기반으로 경로를 Routing 하는 방법을 정의하고 있다.
- Transport(전송) 계층과 함께 Internet에서 중요한 계층이다.
- 데이터가 상대방에게 안전하게 전송되는 것을 보장하지 않는다.
- Transport(전송) 계층이 데이터 전달에 대한 신뢰성을 책임진다는 가정하에 목적지로 Packet(패킷)을 어떤 경로로 전송할 것인가에 대한 문제만 해결하는 계층이다.

2. 계층(Layer)과 프로토콜

□ 전송 계층(Transport Layer)

- 하위 계층 **Internet** 계층의 **IP**가 해결한 목적지까지의 네트워크상의 경로에서 실제 데이터를 전송하는 역할을 한다.
- **TCP**와 **UDP** 라는 데이터의 전달을 책임지는 프로토콜이 존재한다.
- **IP**는 하나의 **Packet**이 전달되는 과정에만 중심을 두고 설계 되었기 때문에 여러 **Packet**으로 나뉘져 전달되는 데이터의 순서와 전송 자체는 신뢰할 수 없다.
- 데이터의 순서와 신뢰할 수 있는 데이터 전송을 보장하는 역할을 전송 계층의 프로토콜이 맡는다.

2. 계층(Layer)과 프로토콜

□ 전송 계층(Transport Layer)

- TCP(Transmission Control Protocol)

- (1) 연결지향 프로토콜이라 데이터를 전송/수신하기 전에 소켓을 통해 양쪽 연결이 성립
- (2) 연결이 성립되면 **TCP**는 데이터의 손실이나 중복 없이 목적지에 확실하게 전달
- (3) 만약 이때 보낸 데이터의 일부가 유실되었다면 수신자는 발신자에게 해당 데이터의 재전송을 요청한다.
- (4) 흔히, **TCP**를 전화 통화와 비교된다.
- (5) **TCP**는 **UDP**에 비해 프로토콜이 더 복잡하고 속도도 느리다.
- (6) **UDP**에 비해 신뢰성 있는 데이터 전송이 가능하다는 장점 때문에 **HTTP, FTP, TELNET** 등 대부분 응용계층 프로토콜은 전송 계층으로 **TCP**를 이용한다.

2. 계층(Layer)과 프로토콜

□ 전송 계층(Transport Layer)

- UDP(User Datagram Protocol)

- (1) UDP는 비 연결지향 프로토콜이다.
- (2) 전송한 데이터가 잘 전달되었는지 확인하지 않고 단지 데이터를 보내는 것으로 자신의 임무를 다한 것으로 생각한다.
- (3) UDP는 TCP에 비해 신뢰성이 떨어지는 프로토콜이다.
- (4) 흔히, 편지를 보내는 것에 비유
- (5) 음악이나 동영상의 **Streaming** 서비스 같은 것에 적당한 프로토콜이다.

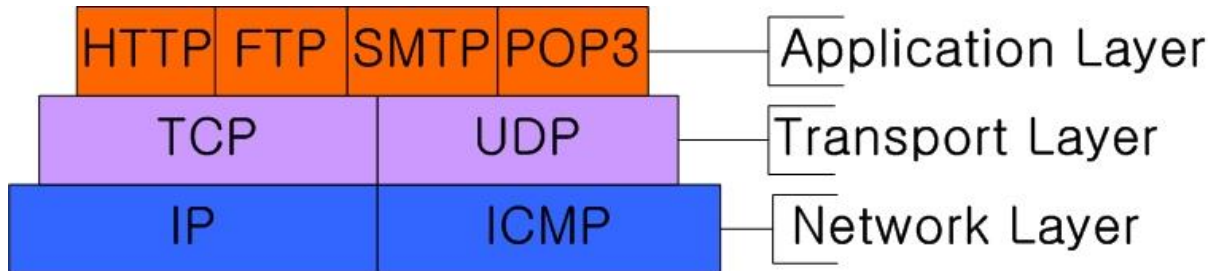
2. 계층(Layer)과 프로토콜

□ 응용 계층 (Application Layer)

- 하위 계층이 목적지가 되는 호스트에 데이터를 안전하게 전달하는 신뢰에서 응용계층에서는 응용 프로그램(프로세스) 들이 데이터 통신을 하게 된다.
- 응용 계층의 응용 프로그램(프로세스) 들의 데이터 통신에는 매우 다양하다.
ex) 메일을 보내기(SMTP), 파일을 전송하기(FTP), 웹사이트에 접속하기(HTTP)
각 각의 목적에 맞는 데이터 통신을 위한 응용 프로토콜이 이미 정의되어 있기도 하고 사용자 프로토콜을 설계하여 인터넷 기반의 서비스를 개발할 수 있다.
- **Socket(소켓)**은 응용 계층에서 개발되는 응용 프로그램에서 하위 계층의 **TCP/IP**의 역할을 감추어 준다(투명성)
- **Socket(소켓)**이라는 도구를 사용하면 응용 프로그램 간의 성격에 따라 기 설계된 응용 프로토콜을 구현한 프로그램을 개발하거나 프로토콜을 설계하고 구현하면 된다.
- 대부분의 네트워크 프로그래밍은 **socket**을 사용하여 위와 같은 작업을 하는 것이라 할 수 있다.

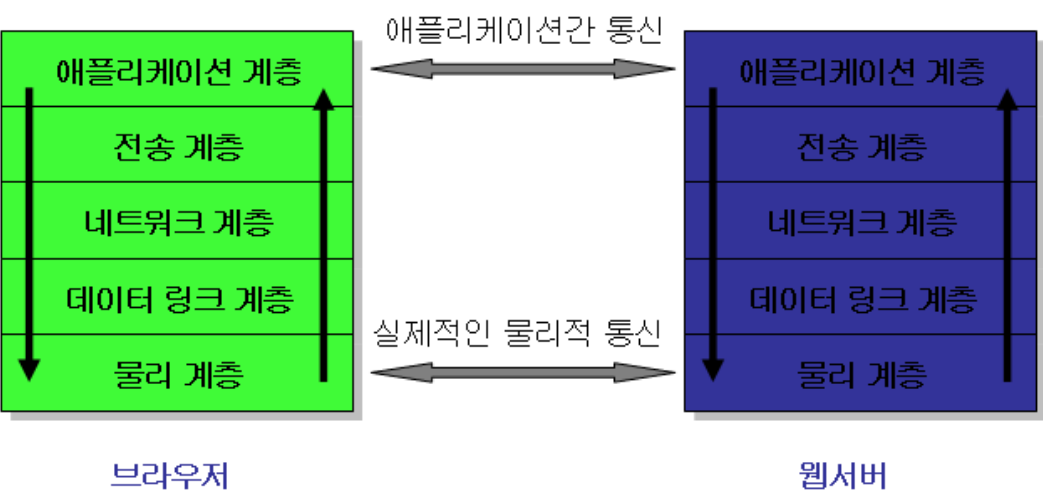
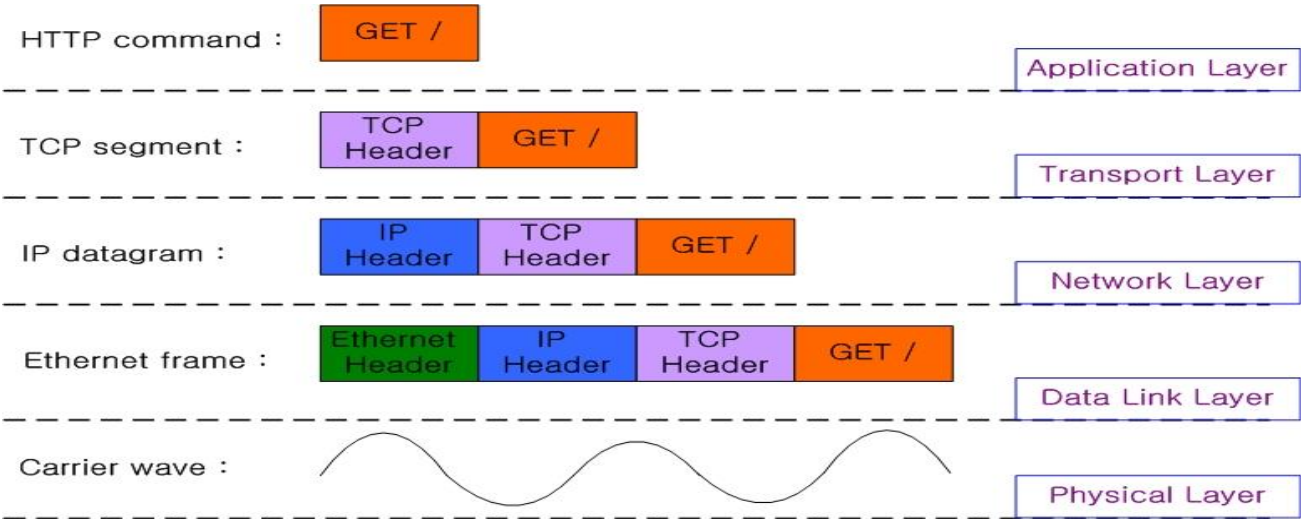
3. TCP/IP 프로토콜 스택(stack, 계층)

- ❑ TCP/IP 프로토콜 스택은 총 4개 부분으로 나누어 진다.
- ❑ 네트워크상의 데이터 통신 과정을 4개의 영역으로 계층화 한 것
- ❑ 데이터 통신 과정을 하나의 덩치 큰 프로토콜로 해결한 것이 아니고 작게 나눠서 계층화하려는 노력의 결과이다.



- ❑ “왜 **OSI 7계층**을 모두 사용하지 않는가?”
 - **OSI 7계층**은 이론적인 면과 하드웨어 장비적인 표준을 위해 제정한 것이다.
 - 실무에서 네트워크 프로그래밍을 할 경우 **90%** 이상의 위의 프로토콜 스택을 기반으로 작업하게 될 것이다.

4. 프로토콜 계층간의 데이터 캡슐화



1강

네트워크와 소켓 이해

3. 소켓이란?

1. 네트워크와 네트워킹
2. Internet
3. 소켓이란?
4. 포트(Port) 번호
5. InetAddress 클래스

1. 소켓이란?

- ❑ **TCP/IP** 프로토콜의 프로그래머 인터페이스
- ❑ 네트워크 프로그래밍에서 개발자에게 네트워크에 접근할 수 있는 인터페이스 제공
- ❑ **1986년 BSD Unix4.3** 개정된 소켓 사용
- ❑ 프로세스 간의 통신 방식(클라이언트 / 서버모델)
- ❑ 3가지 과정으로 사용
 1. 소켓 생성(소켓 열기)
 2. 소켓을 통한 송/수신
 3. 소켓 소멸(소켓 닫기)

2. 소켓생성

❑ Unix 계열 socket 프로그래밍

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

```
int sockfd = socket( PF_INET, SOCK_STREAM, IPPROTO_TCP );
```

❑ domain : 소켓이 사용할 프로토콜 체계(Protocol Family)

- **PF_INET** : IPv4 인터넷 프로토콜 체계
- **PF_INET6** : IPv6 인터넷 프로토콜 체계
- **PF_LOCAL** : 로컬 통신을 위한 **UNIX** 프로토콜 체계
- **PF_PACKET** : **Low Level** 소켓을 위한 프로토콜 체계
- **PF_IPX** : IPX 노벨 프로토콜 체계

2. 소켓생성

□ type : 소켓의 유형

- SOCK_STREAM

- 1) **TCP** 통신 소켓이다.
- 2) **stream** 방식의 연결지향 소켓을 만든다.(전화와 비슷한 연결)
- 3) 양방향 통신
- 4) 바이트를 주고 받을 수 있는 가변 길이 **stream**
- 5) 전달된 모든 데이터는 에러 없이 원격지에 도달
- 6) 전달된 순서대로 수신됨

- SOCK_DGRAM

- 1) **UDP** 통신 소켓이다.
- 2) **datagram** 방식의 비 연결성 소켓을 만든다.(정보와 비슷한 비 연결)
- 3) 양방향 통신
- 4) 고정길이의 메시지를 사용
- 5) 신뢰성이 없다.
- 6) 전달된 순서대로 수신되지 않음

2. 소켓생성

□ 생성 소켓의 유형

- 1) IPv4 인터넷 프로토콜 체계에서 연결지향형 데이터 송수신 소켓
- 2) IPv4 인터넷 프로토콜 체계에서 비 연결 지향형 데이터 송수신 소켓

□ Java에서는 다음의 Class들을 사용한다.

ServerSocket } TCP 소켓
Socket }

DatagramSocket UDP 소켓

1강

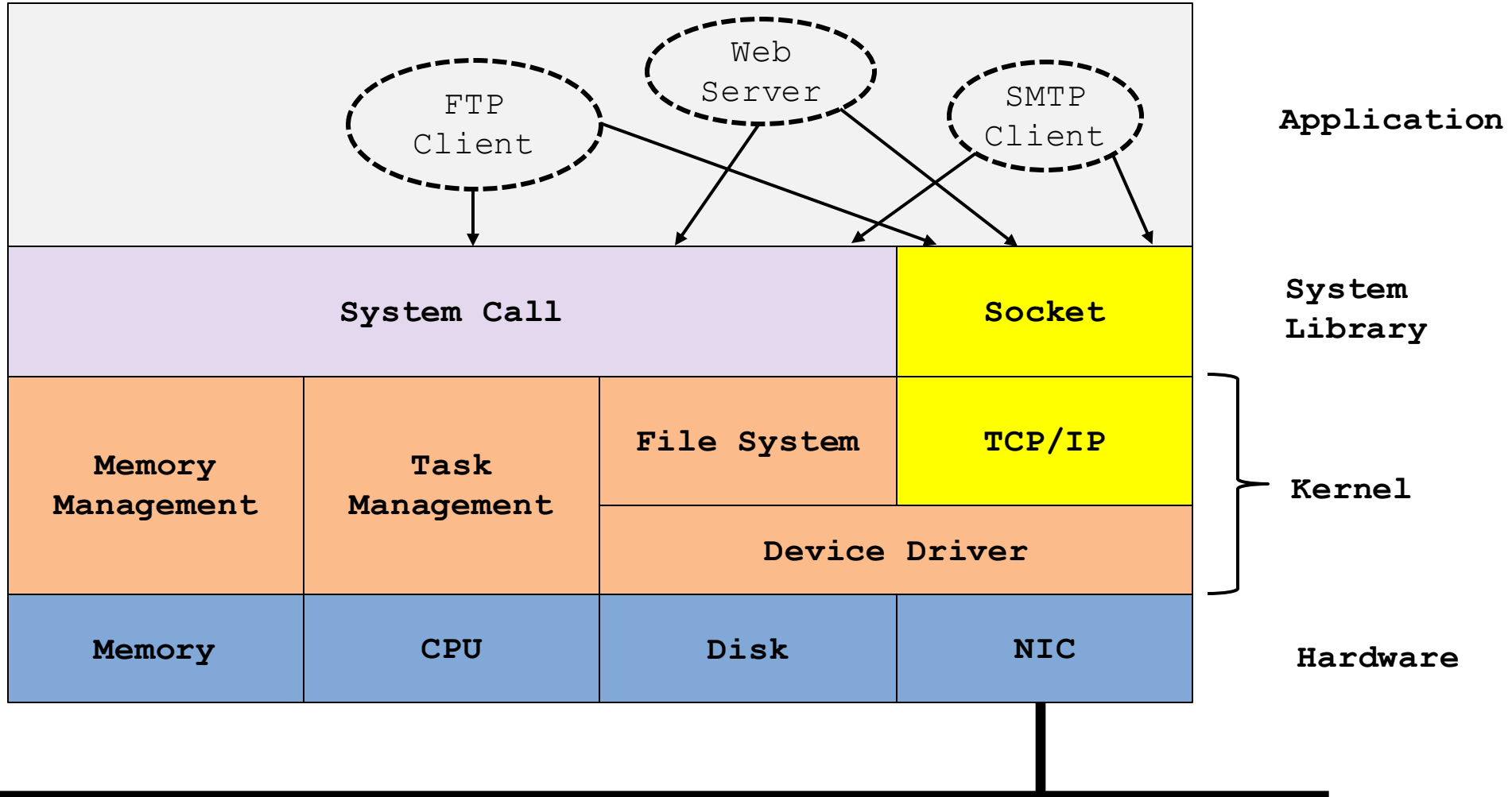
네트워크와 소켓 이해

4. 포트번호

- 1. 네트워크와 네트워킹
- 2. Internet
- 3. 소켓이란?
- 4. 포트(Port)번호
- 5. InetAddress 클래스

1. TCP/IP 그리고 Socket의 위치

□ OS의 대략적인 구조 (Unix 계열)



2. 포트(Port)의 필요성

- ❑ 실제적인 데이터 통신은 연결된 두 **Host**(컴퓨터)의 **Process**(프로그램) 사이에서 이루어진다.
- ❑ 여러 계층을 통해 애플리케이션 계층으로 들어온 데이터를 해당 **Process**에만 정확하게 전달할 필요가 있다.
- ❑ 하나의 **Host**(컴퓨터)에는 여러 개의 **Process**(프로그램)가 각각의 소켓을 사용하여 데이터 통신을 하고 있기 때문에 **TCP**에서는 각 소켓을 구분할 필요가 생긴다.
- ❑ 이 때 각각의 소켓을 구분 할 때 사용하는 것이 포트 이다.
- ❑ 쉬운 예시
 - “ 아파트(**Host**)에 사는 사람(**Process**)에게 편지(**Data**)를 보낼 때, 동(**IP Address**)과 호(**Port**)를 봉투(**Packet**)에 기입해야 한다.”

3. 포트(Port)

❑ 포트번호는 16비트 정수를 사용한다. (0 - 65535)

❑ 포트의 종류

- (1) 1 - 255 : 잘 알려진 인터넷 서비스 포트(Well-Known Port)
- (2) 256 - 1023 : 그 밖의 인터넷 서비스
- (3) 1024 - 4999 : 시스템 예약
- (4) 5000 - 65535 : 사용자 사용

[참고]

<http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>

❑ 포트의 중복은 불가능하다.

❑ 하지만, 같은 **UDP** 포트와 **TCP** 포트는 중복하여 사용할 수 있다.

[정리]

데이터 통신을 위해 목적지 주소에는 IP 주소뿐만 아니라 Port 번호로 포함된다.
그래야 최종 목적지의 해당 Process(프로그램)의 Socket 까지 데이터를 전달할 수 있다.

1강

네트워크와 소켓 이해

5. InetAddress 클래스

1. 네트워크와 네트워킹
2. Internet
3. 소켓이란?
4. 포트(port)번호
5. InetAddress 클래스

1. DNS (도메인 네임 서비스)

- ❑ 도메인 주소를 IP로 반환한다.
- ❑ 반대로 IP주소를 도메인 주소로 반환한다.
- ❑ IP주소에 대한 정보를 얻을 수 있다.
- ❑ 컴퓨터 이름을 구할 수 있다.

[실습1-1] nslookup 사용해 보기

CMD 창에서 nslookup 을 실행하고

www.bitacademy.com, www.naver.com 등
여러 도메인 주소를 가지고 있는 서버의 IP 주소를
알아본다.

```
[kickscar@localhost ~]$ nslookup
> www.bitacademy.com
Server:          115.68.102.211
Address: 115.68.102.211#53

Non-authoritative answer:
Name:   www.bitacademy.com
Address: 122.199.225.42
> www.naver.com
Server:          115.68.102.211
Address: 115.68.102.211#53

Non-authoritative answer:
www.naver.com      canonical name =
www.naver.com.nheos.com.
Name:   www.naver.com.nheos.com
Address: 125.209.222.141
Name:   www.naver.com.nheos.com
Address: 202.179.177.22
>
```

2. InetAddress 클래스

[실습1-2] Local Host 이름 및 IP 주소 구하기

다음 메소드를 사용한다.

public static InetAddress getLocalHost()

public String getHostName()

public String getHostAddress()

public byte[] getAddress()

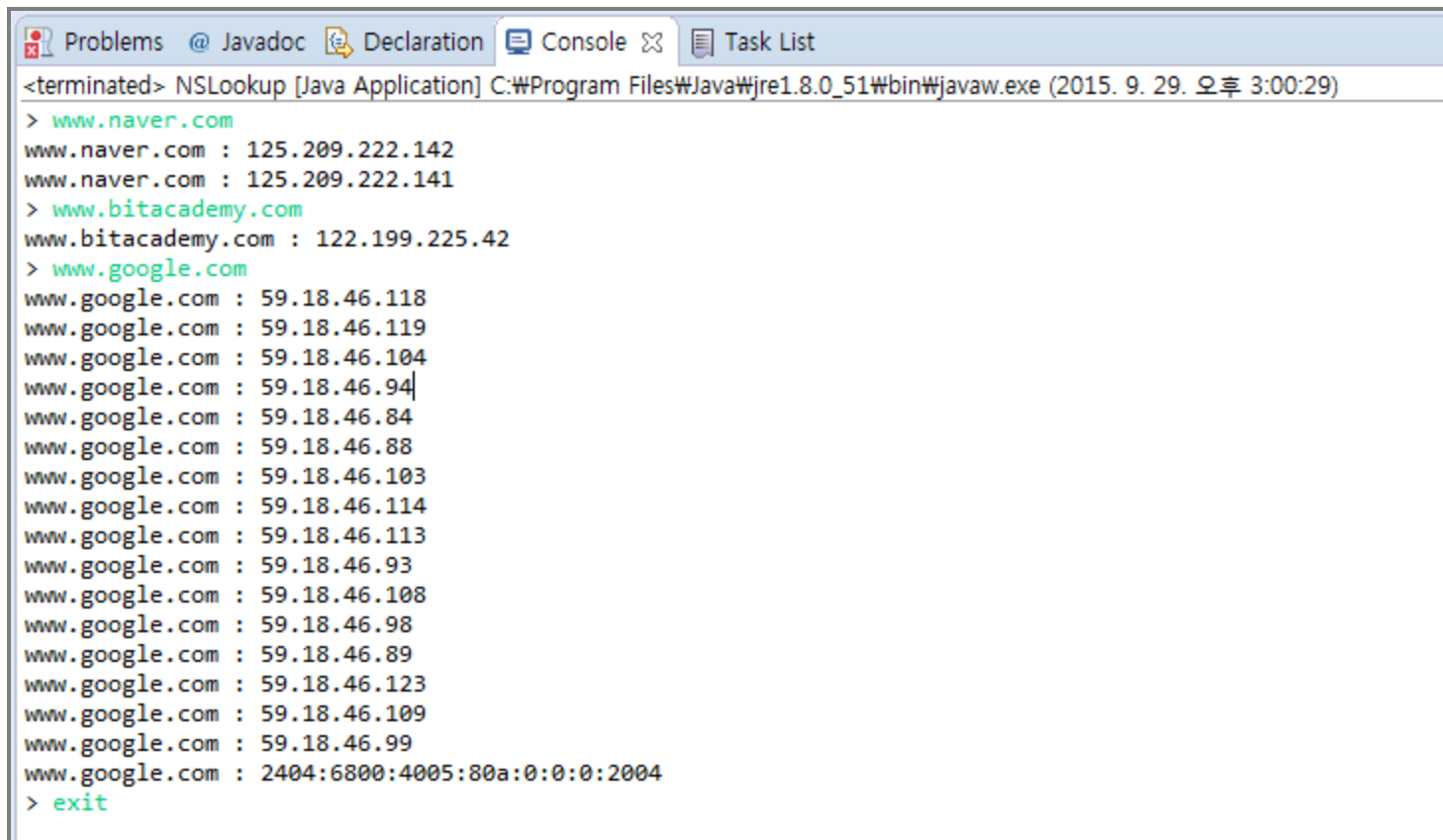
다음 코드를 참고해서 작성하고 테스트 한다.

```
01 public class LocalHost {
02     public static void main(String[] args) {
03         try {
04             InetAddress inetAddress = InetAddress.getLocalHost();
05
06             /* 여기에 정보를 출력하는 코드 추가 */
07
08
09         } catch (UnknownHostException e) {
10             e.printStackTrace();
11         }
12     }
13 }
```


3. InetAddress 클래스

[과제1-1] NSLookup 구현 하기

다음 실행화면과 같은 NSLookup 프로그램을 구현한다.



```
<terminated> NSLookup [Java Application] C:\Program Files\Java\jre1.8.0_51\bin\javaw.exe (2015. 9. 29. 오후 3:00:29)
> www.naver.com
www.naver.com : 125.209.222.142
www.naver.com : 125.209.222.141
> www.bitacademy.com
www.bitacademy.com : 122.199.225.42
> www.google.com
www.google.com : 59.18.46.118
www.google.com : 59.18.46.119
www.google.com : 59.18.46.104
www.google.com : 59.18.46.94
www.google.com : 59.18.46.84
www.google.com : 59.18.46.88
www.google.com : 59.18.46.103
www.google.com : 59.18.46.114
www.google.com : 59.18.46.113
www.google.com : 59.18.46.93
www.google.com : 59.18.46.108
www.google.com : 59.18.46.98
www.google.com : 59.18.46.89
www.google.com : 59.18.46.123
www.google.com : 59.18.46.109
www.google.com : 59.18.46.99
www.google.com : 2404:6800:4005:80a:0:0:0:2004
> exit
```

3. InetAddress 클래스

[과제1-1] NSLookup 구현 하기

1. **Scanner**와 **nextLine()** 메소드를 사용해서 계속 입력받은 도메인을 입력 받는다.
2. 입력받은 도메인의 **IP** 주소를 출력한다.
3. “**exit**” 를 입력 받아 프로그램을 종료할 때 까지 계속 프로그램을 실행한다.
4. **InetAddress**의 **static** 메서드 **getAllByName(String host)** 를 사용한다.