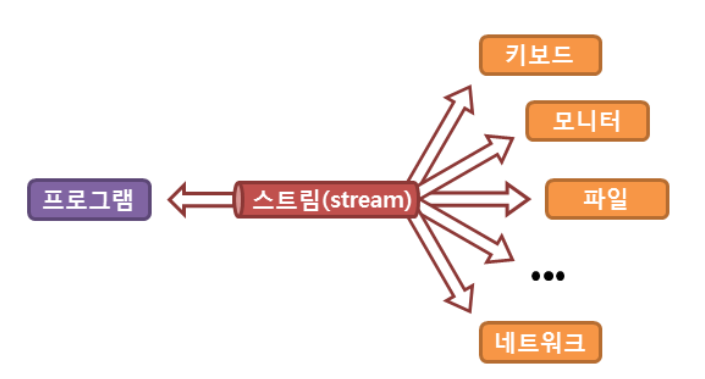
**스트림**

* 자바에서는 파일이나 콘솔의 입출력을 직접 다루지 않고, 스트림이라는 흐름을 통해 다룬다.
* 실제의 입력이나 출력이 표현된 데이터의 이상화된 흐름을 의미

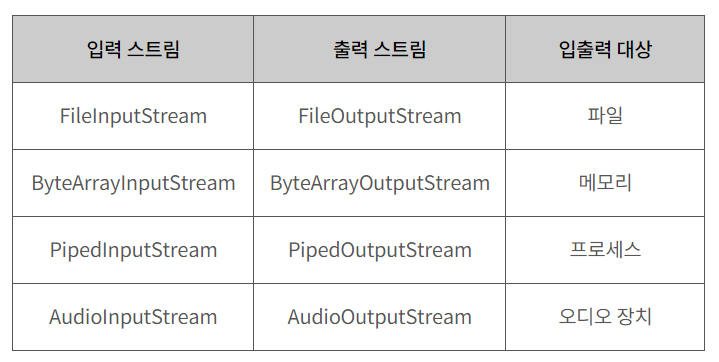


<입출력 스트림>

* 스트림은 한 방향으로만 통신할 수 있으므로, 입력과 출력을 동시에 처리할 수는 없다.
* 사용 목적에 따라 입력 스트림과 출력 스트림으로 구분됨
* InputStream 클래스에는 read() 메소드, OutputStream 클래스에는 write() 메소드가 각각 추상 메소드로 포함되어 있습니다.

<바이트 기반 스트림>

* 자바에서 스트림은 기본적으로 바이트 단위로 데이터를 전송



<문자 기반 보조 스트림>



<BufferReader vs Scanner>

1. Scanner의 버퍼 크기는 1024 chars, 반면 BufferReader의 버퍼 크기는 8192 chars이다.

(큰 파일을 읽을 때는 BufferReader이 좋다.)

1. BufferReader는 문자열을 단순히 읽고 저장, Scanner는 문자열을 구분하여 분석할 수 있다.
2. BufferReader는 동기화 된다. 반면 Scanner는 동기화가 되지 않는다.

(멀티 쓰레드의 경우 여러 쓰레드가 같은 프로세스 내의 자원을 공유하기 때문에 서로의 작업에 영향을 줄 수 있다.)

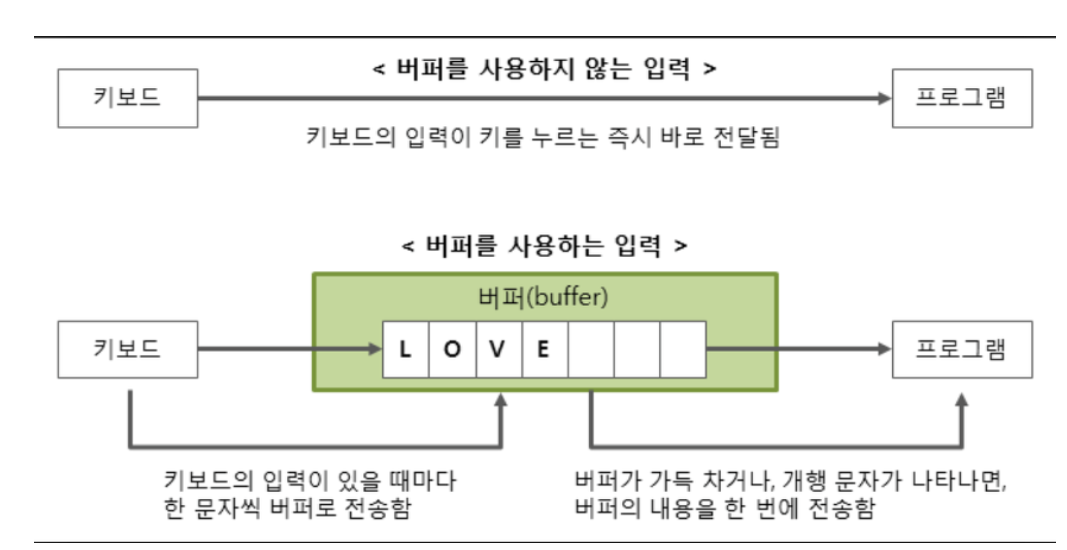
1. BufferReader는 즉시 IOException 처리를 던지지만, Scanner는 숨긴다.

세마포어(Semaphore) : 공유된 자원의 데이터를 여러 프로세스가 접근하는 것을 막는 것

현재 공유자원에 접근할 수 있는 쓰레드, 프로세스의 수를 나타내는 값을 두어 상호배제를 달성하는 기법

뮤텍스(Mutex) : 공유된 자원의 데이터를 여러 쓰레드가 접근하는 것을 막는 것,

한 쓰레드, 프로세스에 의해 소유 될 수 있는 key를 기반으로 한 상호배제기법



Buffer : 데이터를 한 곳에서 다른 한 곳으로 전송하는 동안 일시적으로 그 데이터를 보관하는 임시 메모리 영역, 입출력 속도 향상을 위해 버퍼 사용

**<process와 Thread>**

프로세스

1. 컴퓨터에서 연속적으로 실행되고 있는 컴퓨터 프로그램
2. 메모리에 올라와 실행되고 있는 프로그램의 인스턴스(독립적인 개체)
3. 운영체제로부터 시스템 자원을 할당받는 작업의 단위
4. 동적인 개념으로는 실행된 프로그램을 의미

쓰레드

1. 프로세스내에서 실행되는 여러 흐름의 단위
2. 프로세스의 특정한 수행 경로
3. 프로세스가 할당 받은 자원을 이용하는 실행의 단위

* 프로세스는 실행될 때 운영체제로부터 각각 독립된 메모리 영역(code, Data, Stack, Heap)을 할당받는다.
* 기본적으로 하나의 프로세스가 생성되면 하나의 쓰레드(메인 쓰레드)가 같이 생성됨
* 한 프로세스가 다른 프로세스의 자원에 접근하려면 프로세스 간 통신(IPC, inter-process communication)을 사용해야 한다.(ex. 파이프, 파일, 소켓 등이 이에 해당)
* 쓰레드는 한 프로세스 내에서 동작되는 여러 실행의 흐름으로 프로세스 내에서 각각 Stack만 따로 할당 받고 Code, Data, Heap 영역은 공유한다.

**<Thread vs Runnable>**

* Thread를 상속(extends)받게 되면 상속받은 클래스는 다른 클래스를 상속 받을 수 가 없다
* Thread를 한번 사용하면 재사용이 불가능
* Runnable은 인터페이스이기 때문에 구현(implements)만 하면 되고 다른 필요한 클래스를 상속 받을 수 있다는 것
* Runnable를 이용하여 Thread를 구현하면 재사용할 수 있다.

<채팅 만들기>

<https://gdtbgl93.tistory.com/144>

* 채팅의 원리

1. 서버-클라이언트

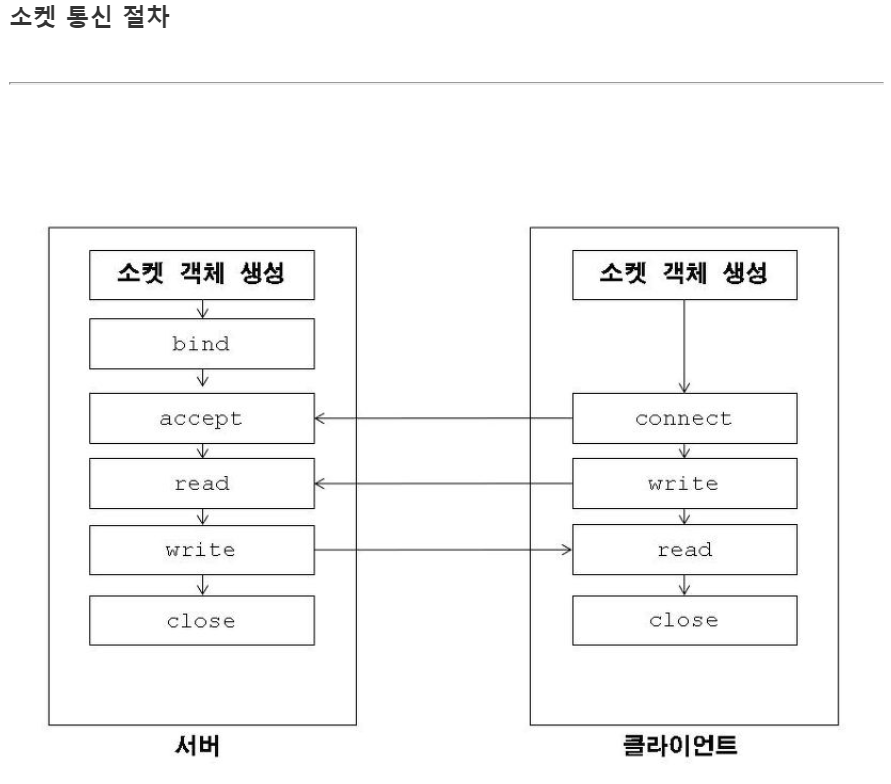
* 서버와 클라이언트 모두 네트워크에 연결되어 IP를 할당 받고 있는 디바이스이다.
* 서버는 클라이언트가 필요한 자원을 가지고 있다.
* 클라이언트는 서버에 특정 자원에 대한 요청을 하면 서버는 그 자원을 찾아서 응답을 해준다.

1. TCP/IP

* 다른 컴퓨터와 통신을 하기 위한 통신 규약 중 하나로 OSI 계층 중에 전송 계층에서 사용된다.
* 인터넷에서 서로 연결된 컴퓨터 프로그램 간에 데이터를 안정적이고 순서대로 교환할 수 있게 해주는 신뢰성이 높은 방식

1. Socket

* 컴퓨터 네트워크를 경유하는 프로세스 간 통신의 종착점(ex. 톨게이트)
* 클라이언트에서 출발한 패킷이 회선을 거쳐서 서버 컴퓨터의 맨 마지막 하단부인 소켓에 도달해서 데이터를 교환
* 소켓은 인터넷 프로토콜, 원격지와 로컬의 IP 주소와 포트에 관한 정보 등을 담고 있다.



사용되는 클래스 설명

1. InetAddress

* IP(Internet Protocol)에 대한 정보를 가지고 있는 클래스.
* Host와 Address등에 관한 정보를 담고 있다.
* 이 클래스는 static 클래스로 동적으로 생성하지 않는다.

2. Socket

* Client Socket을 구현하는 클래스
* 두 기기 사이에 종단점에 위치하여 데이터를 교환

3. Server Socket

* Client에서 들어오는 요청을 기다리는 Server Socket을 구현하는 클래스.

4. InputStreamReader

* byte 단위로 들어오는 InputStream에 대하여 char 단위로 읽고쓰는 Reader 인터페이스를 제공해주는 보조스트림

5. BufferReader

* Buffer를 통해 char 단위로 읽고 쓸수 있게 해주는 보조 스트림

6. OutputStreamWriter

* byte 단위로 쓰는 OutputStream에 대해 특정 인코딩의 char 단위로 읽고 쓰게 해주는 보조 스트림

7. PrintWriter

* text로 대표되는 객체들을 출력할 수 있게 해주는 보조 스트림