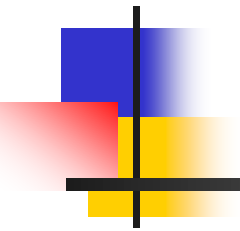
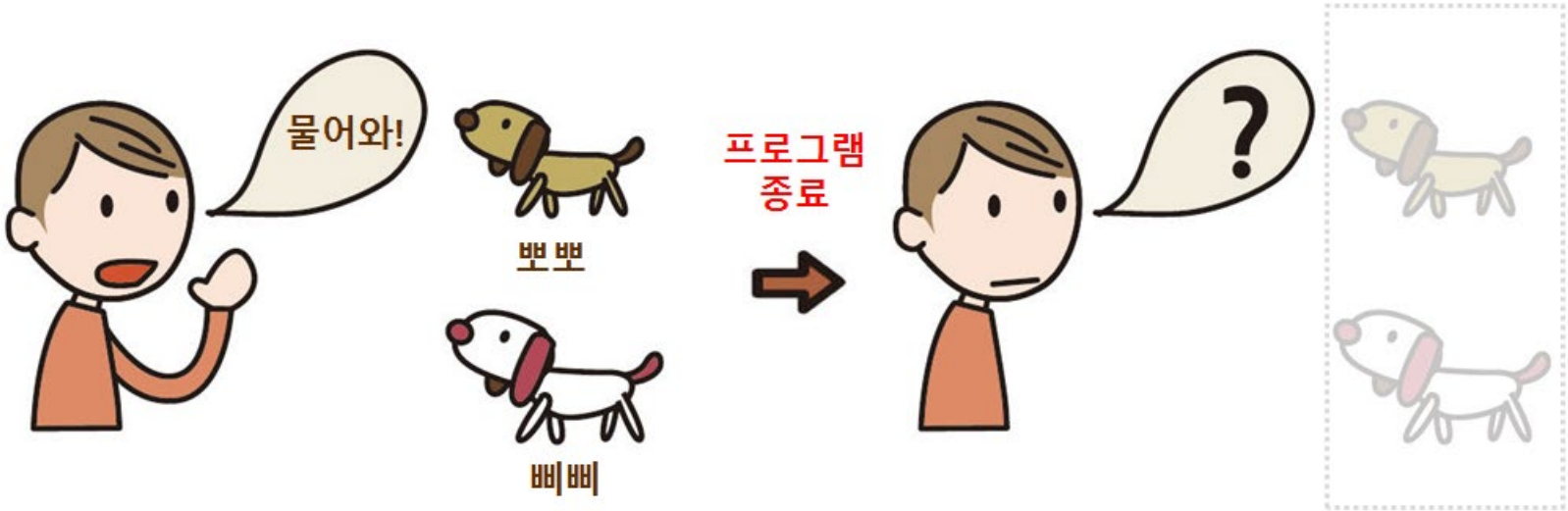


I/O Stream

경복대학교
소프트웨어융합과
배희호 교수

File의 필요성

- Program이 종료되면 만들어 둔 강아지는 모두 사라질 수 있어요

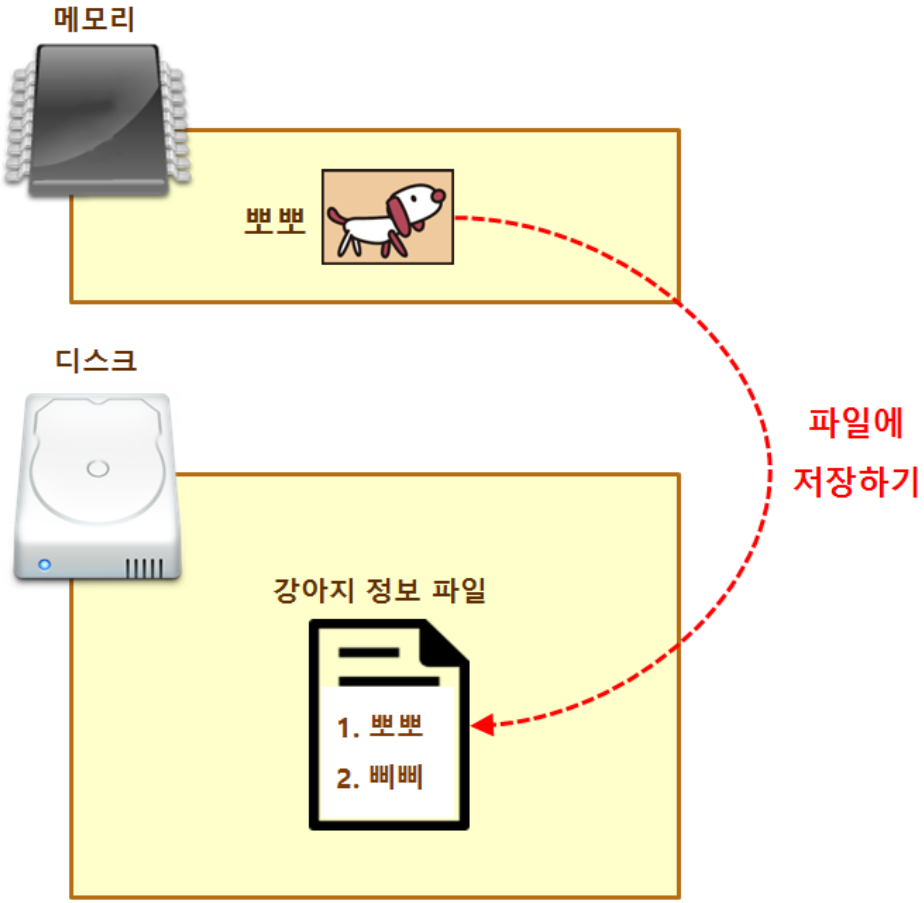


File의 필요성

- 만들어둔 강아지의 이름을 적어 두어야 기억할 수 있어요

File의 필요성

- File이라는 저장 공간



File의 필요성

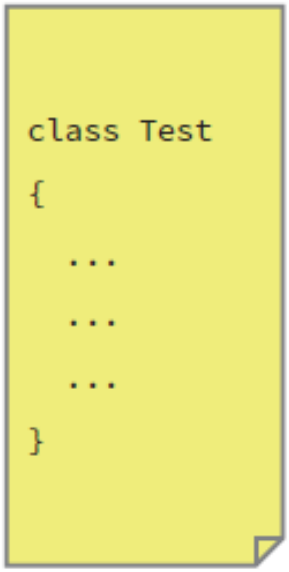
- 책이 어디에 있건 읽거나 쓰는 방법은 똑같이 정해둘 수 있어요

이렇게 읽어요.

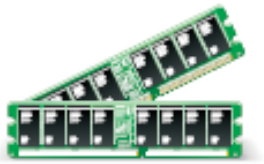
이렇게 써요.

File의 필요성

- 실행 중이던 Program이 종료되면 ?



프로그램



메모리

객체는 모두 메모리에 만들어지고 이것들은 모두 전원이 꺼지면 사라진다.

하드 디스크

하드 디스크에 파일 형태로 저장하면 전원이 꺼지더라도 데이터가 보존된다.

데이터를 영구히 보관하려면 파일에 저장해야 합니다.
스트림을 사용하면 됩니다.

File의 필요성

- Program에서 변수 및 배열에 Data를 저장하는 것은 일시적임 (Memory 사용)
- 많은 양의 Data를 장기간 보존하기 위해서는 File을 사용
- Data를 사용하는 Program이 종료 된 후에도 보존됨
- 영구 Data – Program 실행 기간 외에 존재 함
- 보조 저장 장치에 저장된 File

Data Hierarchy

- Computer는 모든 Data 항목을 0과 1의 조합으로 처리
- Data Hierarchy(데이터 계층)
  - Computer에서 처리하는 Data 항목은 Data 계층을 형성하여 Bit에서 File까지 더 커지고 복잡해 짐
  - Bit(비트)
    - Computer에서 가장 작은 Data 항목으로 0 또는 1 값을 가질 수 있음
  - Byte(바이트)
    - 8 Bit

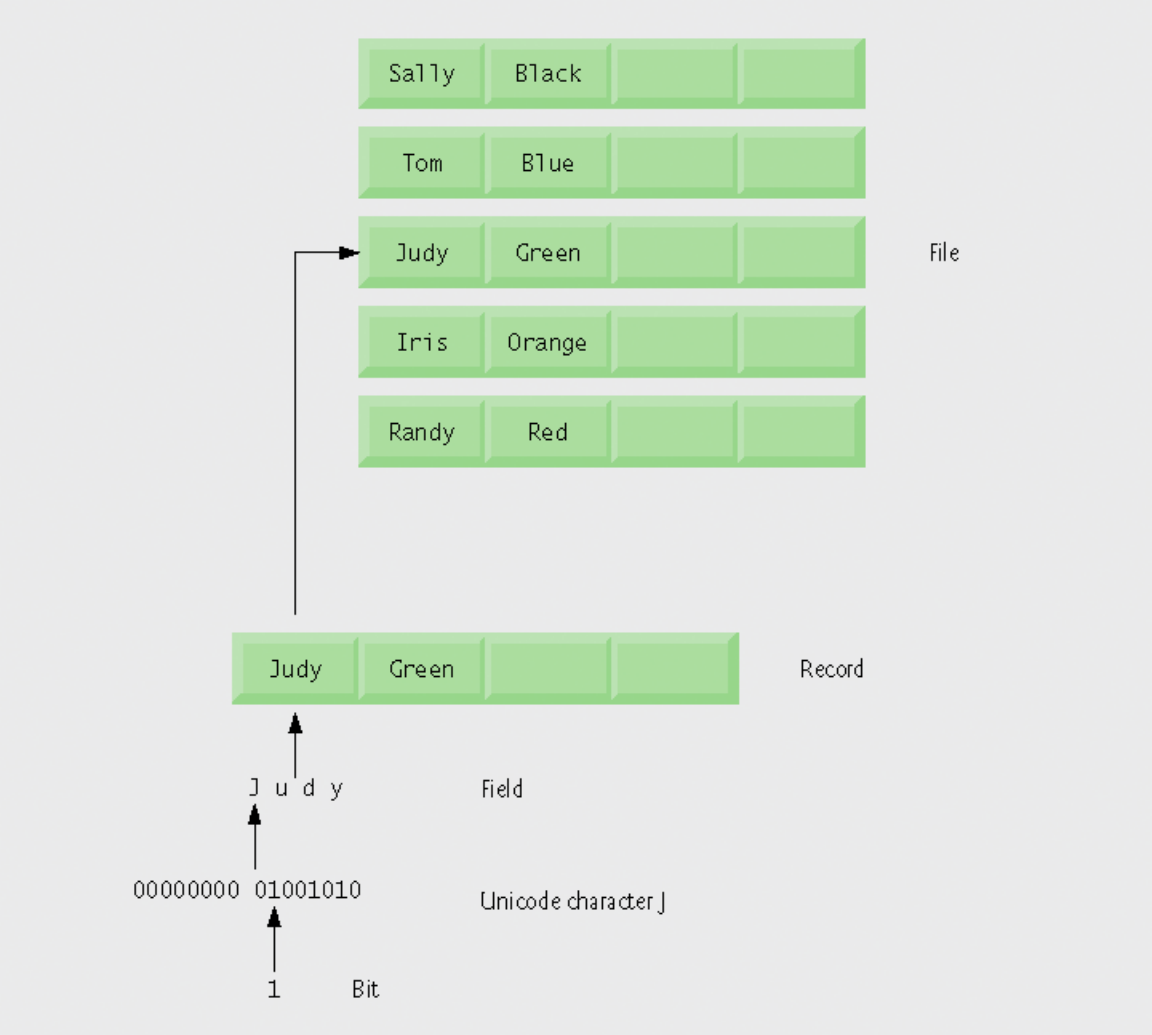
Data Hierarchy

- Characters(문자)
  - 큰 Data 항목
  - 10 진수, 문자 및 특수 기호로 구성
  - 문자 집합 – Program을 작성하고 Data 항목을 나타내는 데 사용되는 모든 문자 집합
    - UNICODE – 2 Byte로 구성된 문자
    - ASCII
- Field(필드)
  - 의미를 전달하는 문자 또는 Byte Group
- Record(레코드)
  - 관련 Field Group
- File(파일)
  - 관련 Record Group

Data Hierarchy

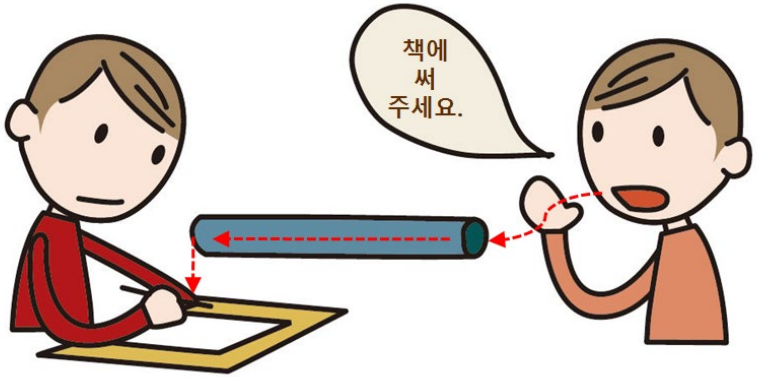
- Record Key
  - Record를 특정 개인 또는 Entity를 식별할 수 있는 것
  - 특정 Record를 쉽게 검색하는 데 사용
- Sequential File(순차 파일)
  - Record가 Record Key Field에 의해 순서대로 저장되는 File
- Database(데이터베이스)
  - 관련 File Group
- Database Management System
  - Database를 만들고 관리하도록 설계된 Program 모음

Data Hierarchy



Stream

- Stream은 “Byte들의 연속적인 흐름”
- Stream이란 입출력 Data의 형식이나 길이가 정해지지 않고, 문자형 또는 Byte열의 Data가 일렬로 이동하는 것을 나타냄
- 입력 Stream은 Keyboard, File, Memory, Buffer등으로부터 입력되는 Data의 일관된 표현이고, 출력 Stream은 Monitor, File, Memory, Buffer등에 출력하는 Data의 일관된 표현 임
- Stream 상태의 Data 송수신은 Internet 환경에서의 Data 송수신 방식과 일치함



Stream

- JAVA는 모든 File을 Sequential Byte Stream으로 봄
  - JAVA에서 Data는 Data를 생산해 내는 Source로부터 Data를 소비하는 Destination까지 Stream 상태로 이동하게 됨
- 운영 체제(OS)는 File의 끝을 결정하는 Mechanism을 제공
  - File의 끝 Marker(EOF : -1)
  - File의 총 Byte 수
  - Byte Stream을 처리하는 JAVA Program은 Program이 Stream의 끝 부분에 도달하면 운영 체제(OS)로부터 지시를 받음

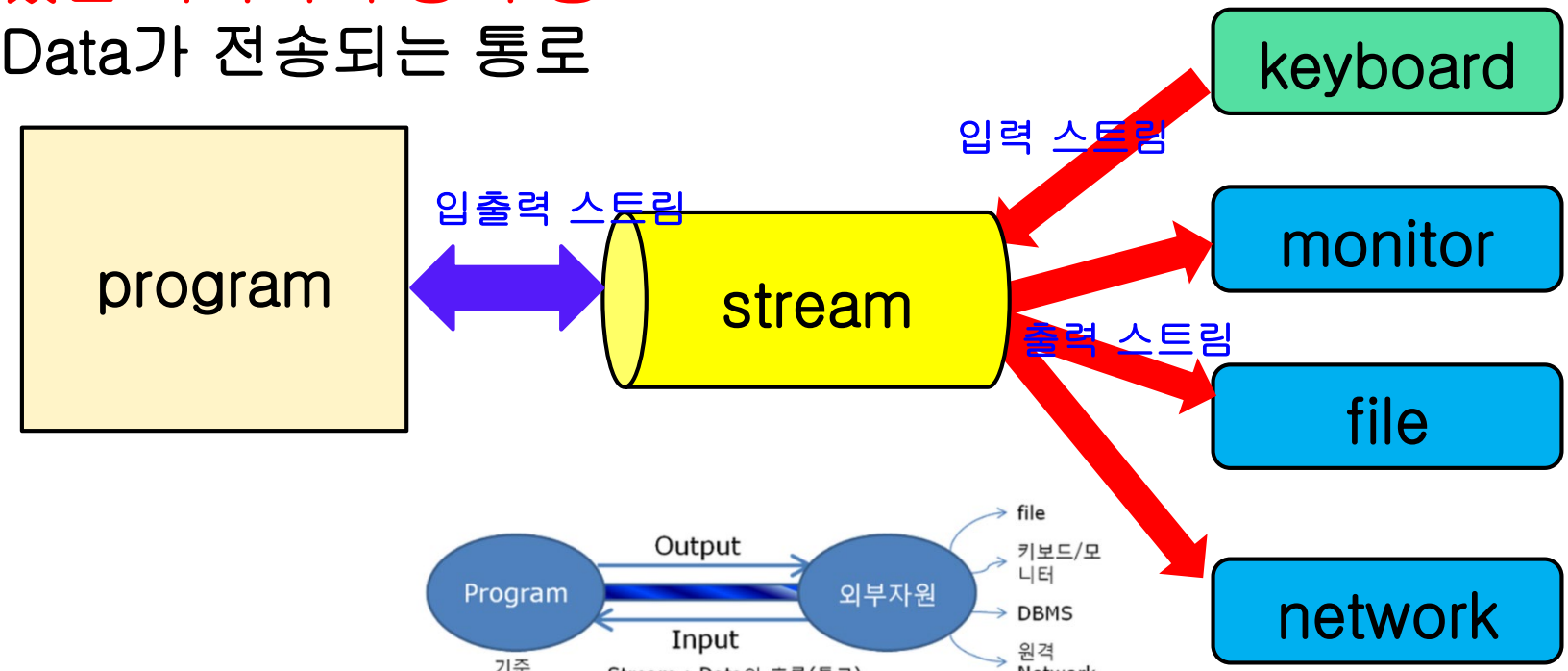
n Byte의 File

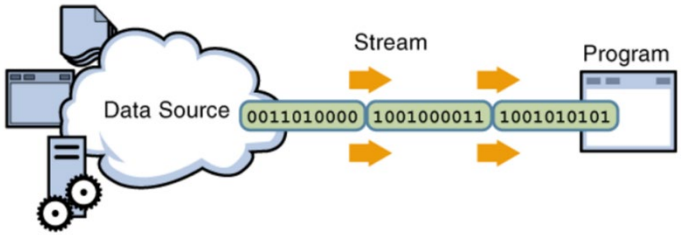
Stream

- JAVA의 Stream
  - JAVA Stream은 입출력 장치와 JAVA 응용 Program 연결
  - 입출력 장치와 Program사이의 Data 흐름을 처리하는 Software Module
- JAVA Stream의 종류
  - 입력 Stream
    - 입력 장치로부터 JAVA Program으로 Data를 전달하는 Software Module
  - 출력 Stream
    - JAVA Program에서 출력 장치로 Data를 보내는 Software Module
  - 입출력 Stream 기본 단위 : Byte

Stream

- 한 곳에서 다른 곳으로 순서가 있는 일련의 Data를 보낼 수 있는 하나의 추상적 통로
- Data가 전송되는 통로



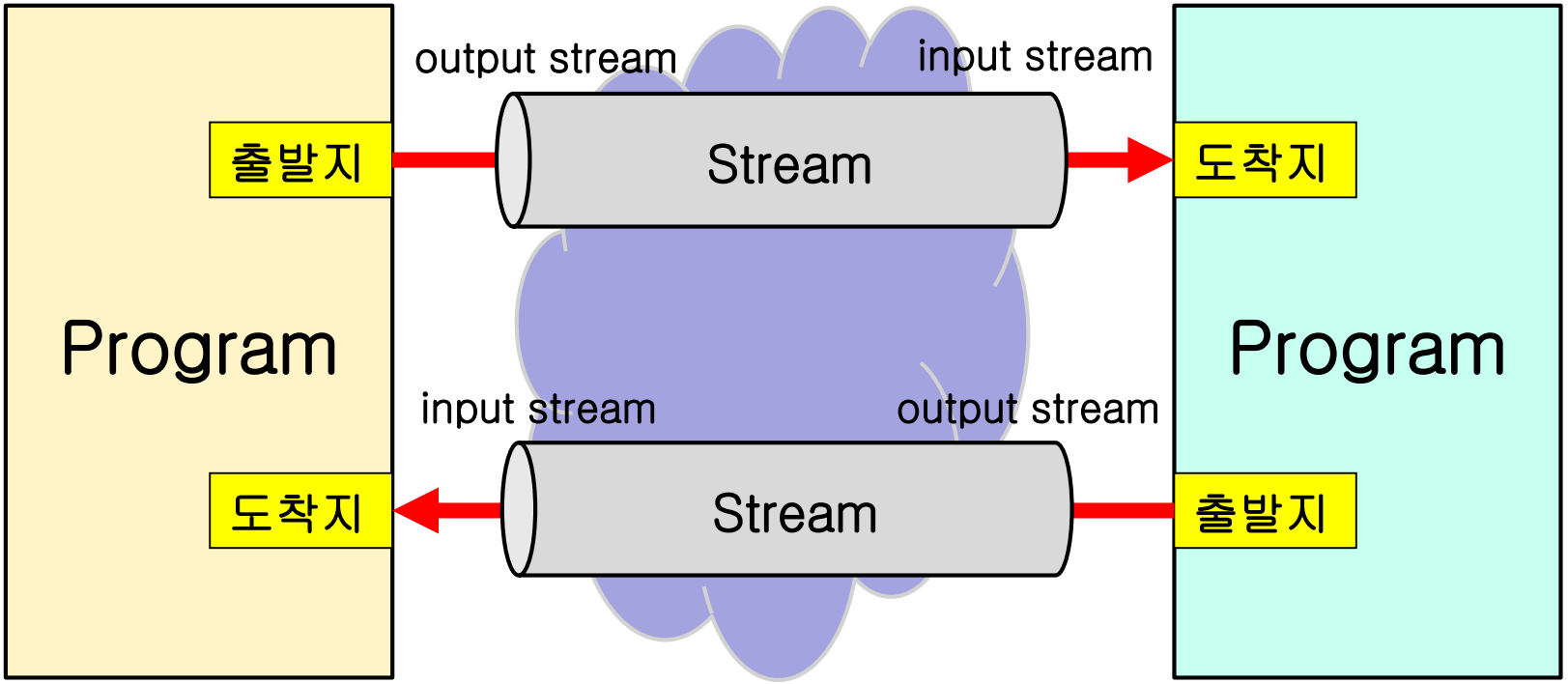


Stream

- 전송되는 Data의 관점에서 보면 연속된 일련의 Byte
- C 언어를 만든 사람 중 한명인 Denis Ritchie가 UNIX 운영 체제를 만들면서 처음으로 제안
- Stream의 종류
  - File
  - Memory
  - 다른 Program의 내부
  - Network
- JAVA의 Stream은 Stream의 종류에 상관없이 Data 이동 방식은 동일
  - File 입출력이든 Monitor, Printer, Internet 출력, Keyboard 입력 등 모든 입출력이 거의 유사한 방식으로 이뤄짐

Stream

- Data가 전송되는 통로
- 단 방향 IO
  - 읽기 Stream은 읽기만, 쓰기 Stream은 쓰기만 가능



Stream

- JAVA 입출력 Stream
  - Stream은 입출력을 물의 흐름처럼 간주하는 것

- 흐름의 방향에 따른 분류

Stream

- JAVA 입출력 Stream 특징
  - 흐름 – 2진 Data의 흐름
    - 입력 Stream : Keyboard와 같은 입력 장치
    - 출력 Stream : Printer나 Monitor 등과 같은 출력 장치
  - FIFO(First Input First Output) 구조
  - 단 방향
  - Stream은 “순서가 있는 Data의 연속적인 흐름”
- 표준 입출력
  - 표준 입력(stdin), 표준 출력(stdout), 표준 에러(stderr)를 기본으로 하여 물리적인 File을 Stream으로 읽고 쓸 수 있음
  - java.lang Package의 System Class에 정의된 변수 in, out, err를 이용하여 표준 입출력 및 Error를 출력

Stream

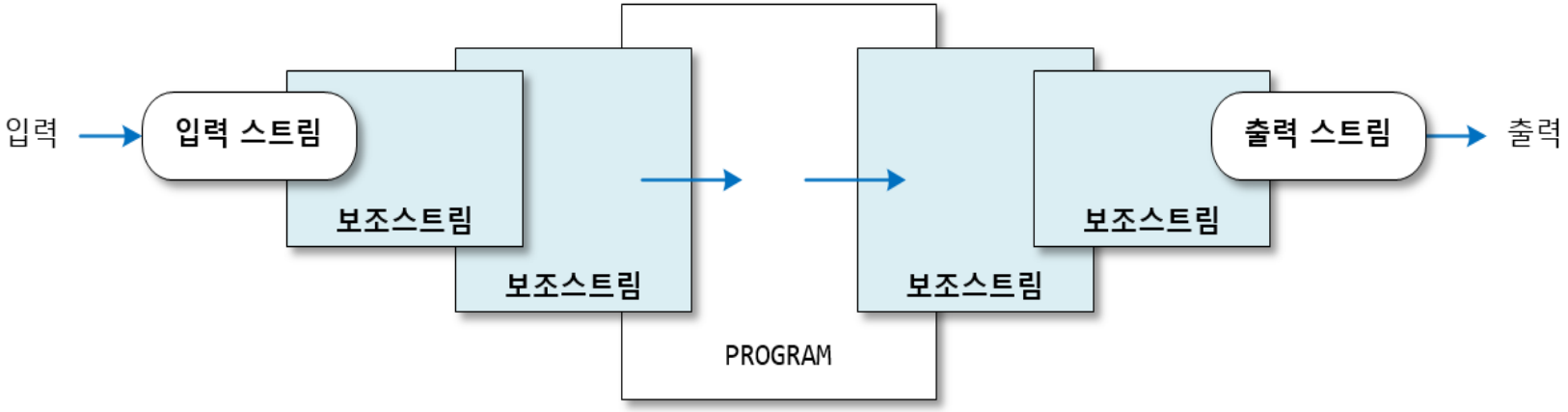
- 표준 Stream
  - 각 Stream을 redirection 할 수 있음
  - System.in
    - 표준 입력 Stream Object이며 setIn() Method로 redirection 될 수 있음
  - System.out
    - 표준 출력 Stream Object로 setOut() Method를 사용하여 redirection 할 수 있음
  - System.err
    - 표준 Error Stream Object로 setErr() Method를 사용하여 redirection 될 수 있음

Stream

- 8 Bit Stream, 16 Bit Stream 2가지 제공
  - 8 Bit Stream (Byte 기반 Stream)
    - 모든 Data를 읽고 쓸 수 있음
  - 16 Bit Stream (문자열 기반 Stream)
    - 문자열 Data만 읽고 쓸 수 있음

Stream

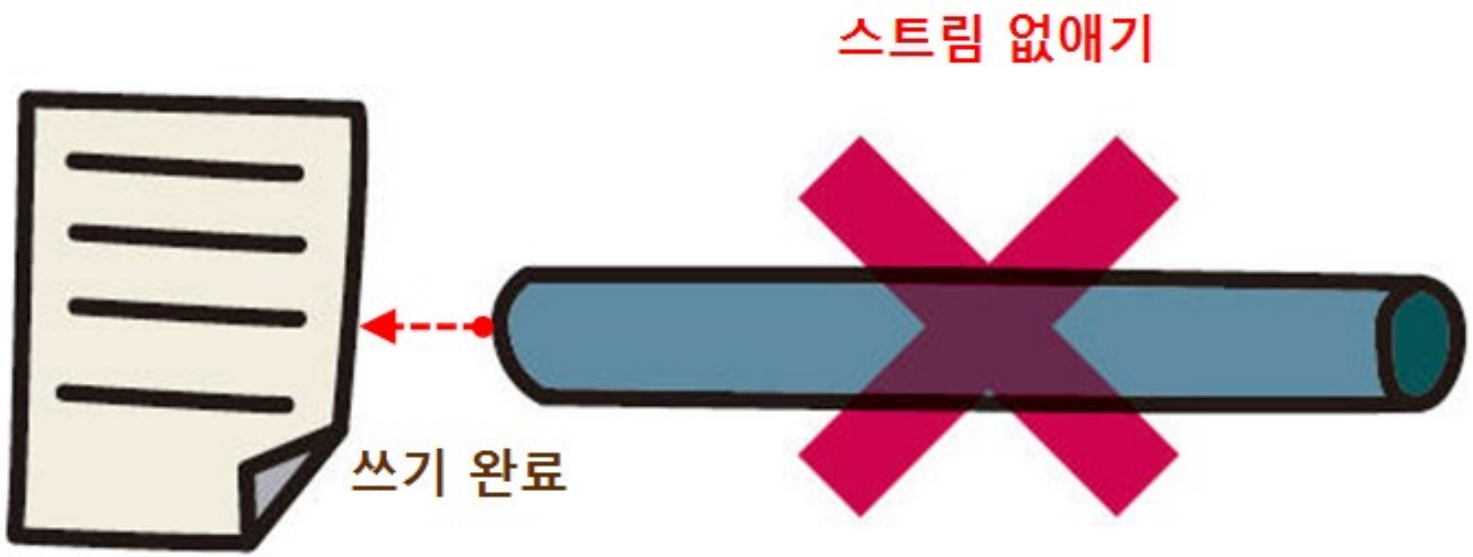
- 순서가 있는 Data의 흐름을 Stream이라고 함
  - File로 Data를 출력하려면 이 Data들이 순서를 가지고 내보내 져야 함
  - File로부터 Data를 읽어 오려면 역시 File의 내용이 순서대로 읽어와 져야 함



- 보조 Stream이란 “Program에서” File을 읽기/쓰기 할 수 있도록 해주며, 주 Stream은 “외부에서” File 읽기/쓰기를 수행

Stream

- Stream을 쓰고 나면 꼭 닫아 주어야 함



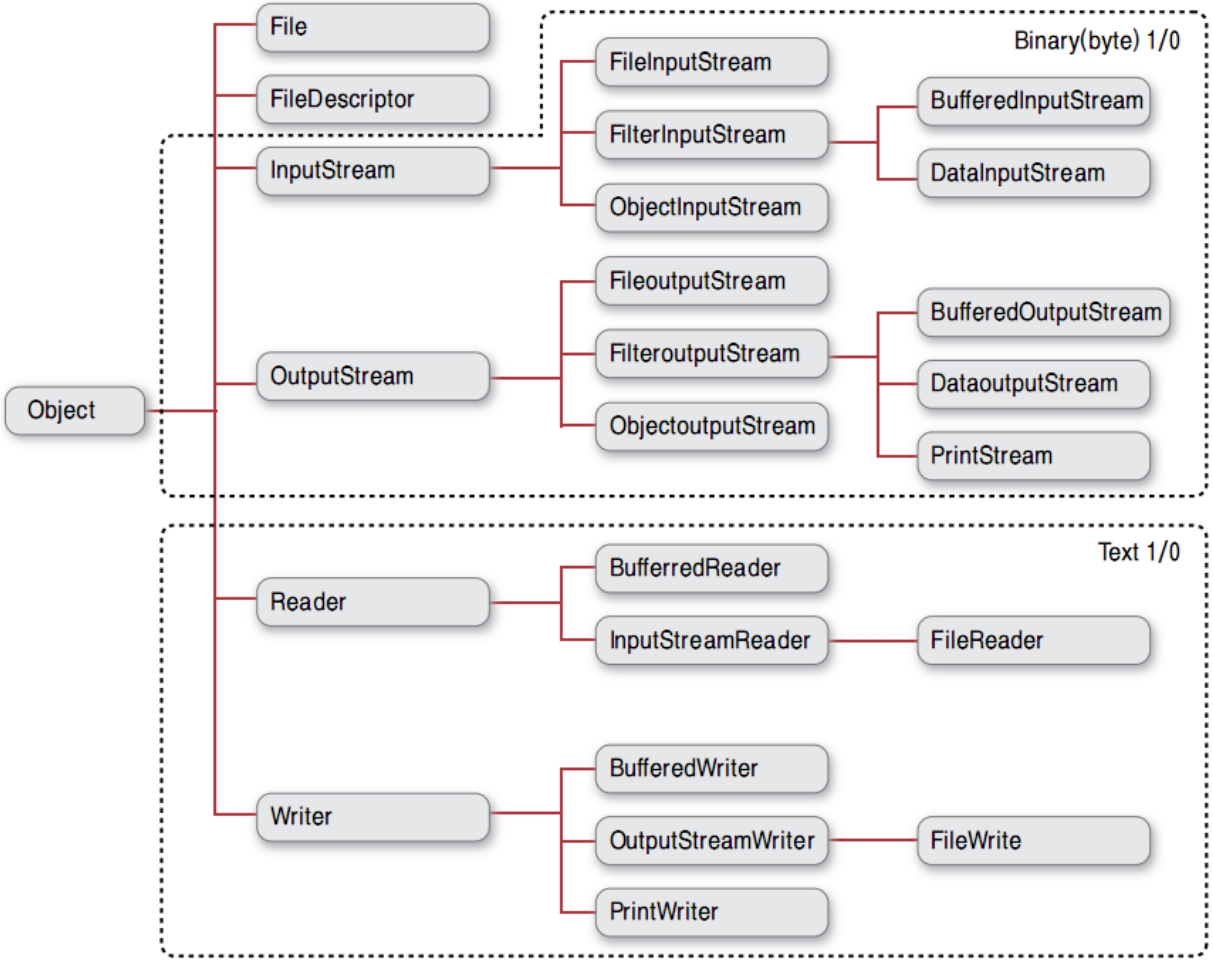
Stream

- JAVA에서 입출력을 위한 Stream을 생성하고 다루는 Class는 java.io Package에 포함되어 있으므로, JAVA Program의 첫 부분에 이 Package를 import하여야 함
- Stream의 종류
  - 입출력의 단위에 따라서 분류

  - Byte Data를 담당하는 Class의 이름은 모두 Stream으로 끝남
  - 초기 버전의 JAVA에서도 지원됨

Stream

- java.io Package의 Class 계층 구조



Stream

Byte Stream 클래스	특징	Character Stream 클래스
InputStream	기본 입력 Stream 클래스	Reader
FileInputStream	File 입력 Stream 클래스	FileReader
FilterInputStream	다른 Filter 클래스의 최상위 클래스	FilteredReader
BufferedInputStream	Buffer 기능이 있어 편리	BufferedReader
DataInputStream	JAVA 기본형 Data를 읽는데 편리	없음
OutputStream	기본 출력 Stream 클래스	Writer
FileOutputStream	File 출력 Stream 클래스	FileWriter
BufferedOutputStream	Buffer 기능이 있어서 편리	BufferedWriter
DataOutputStream	JAVA 기본형 Data를 출력할 때 유용	없음
PrintStream	표준 출력 System으로 나감	PrintWriter

Stream

- 그외의 java.io Class

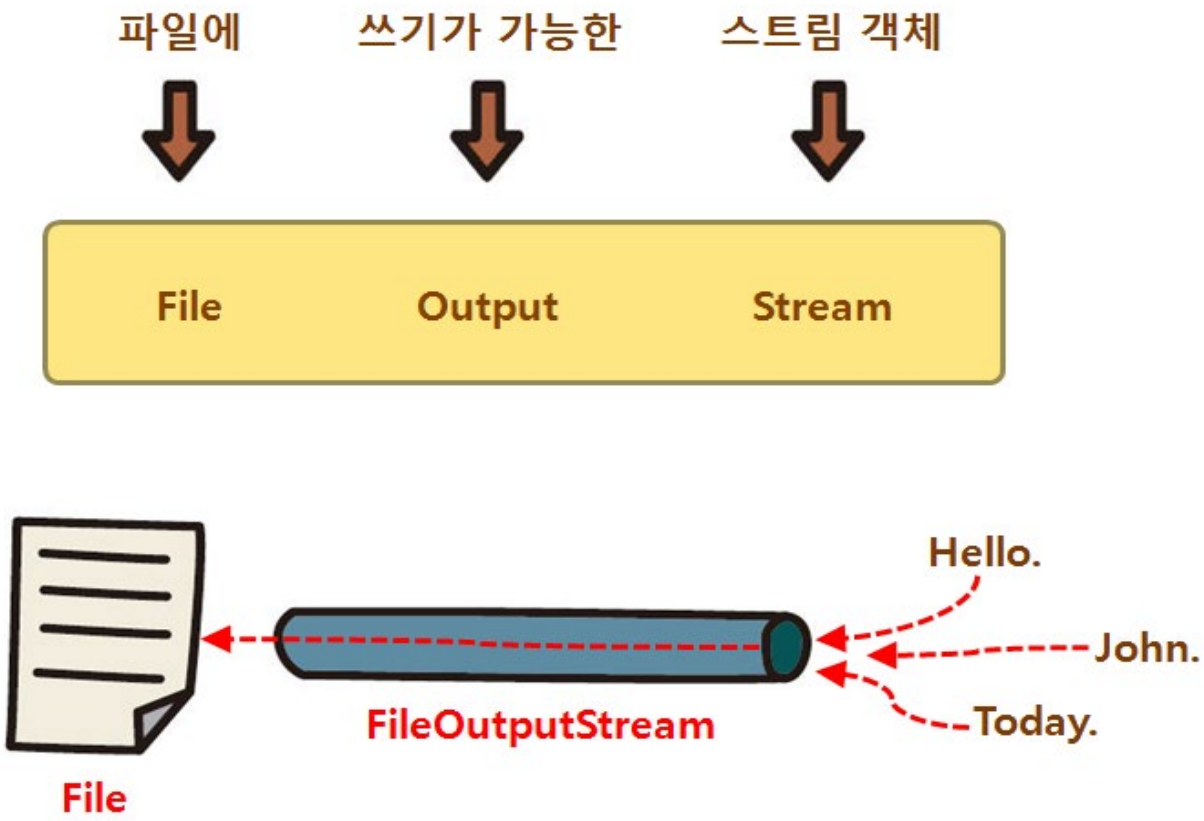
클래스	설 명
console	명령행에서 쉽게 입력을 받고, 정형화된 출력을 명령행에 쉽게 출력할 수 있음
File	File Object를 생성
FileDescriptor	물리적 File에 대한 현재의 연결을 나타내기 위한 Class
FilePermission	File 및 Directory에 Access 접근 권한을 관리하는 Class
RandomAccessFile	Random Access File로부터 읽기와 쓰기가 동시에 이루어질 수 있음
SerializablePermission	직렬화 가능 Access 권한을 위한 Class
StreamTokenizer	입력 Stream을 인수에 취해, 그것을 [토큰]에 구문 분석 해, 한 번에 1개 토큰을 읽음

Stream

- I/O Class의 이름과 의미
  - Stream으로 끝나는 Class
    - Byte 단위로 입출력을 수행하는 Class
  - Reader / Writer로 끝나는 Class
    - 문자(Character) 단위로 입출력을 수행하는 Class
  - File로 시작하는 Class
    - HDD의 File을 사용하는 Class
  - Data로 시작하는 Class
    - JAVA의 기본 Data Type을 출력하기 위한 Class
  - Buffered로 시작하는 Class
    - System Buffer를 사용하는 Class

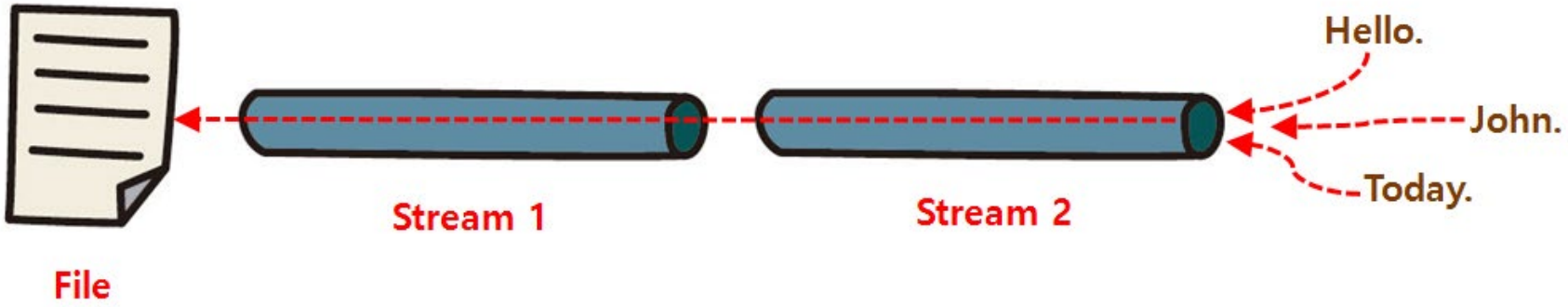
Stream

- FileOutputStream Object



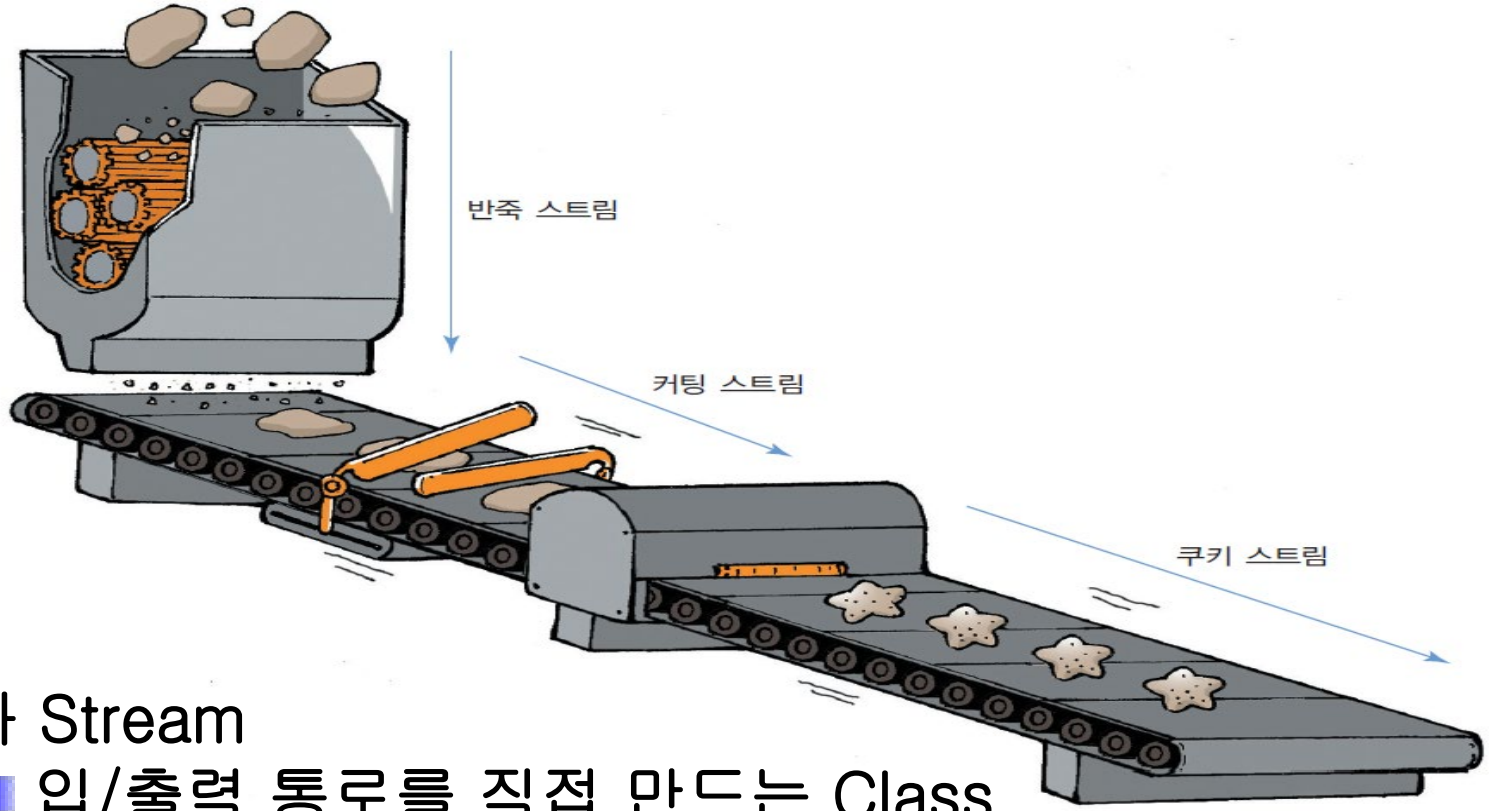
Stream

- 2개의 Stream을 연속해서 사용하기



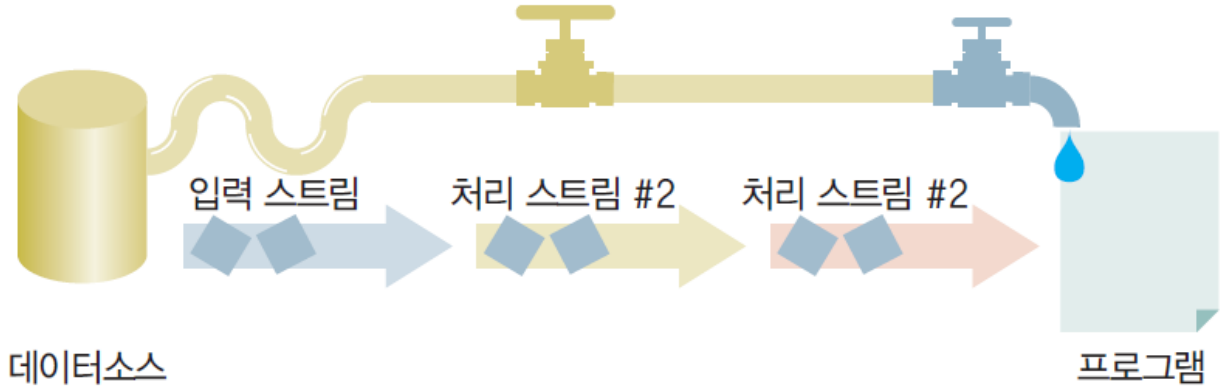
Stream

- Stream들은 연결될 수 있음 (별 모양의 쿠키를 굽는 Stream)

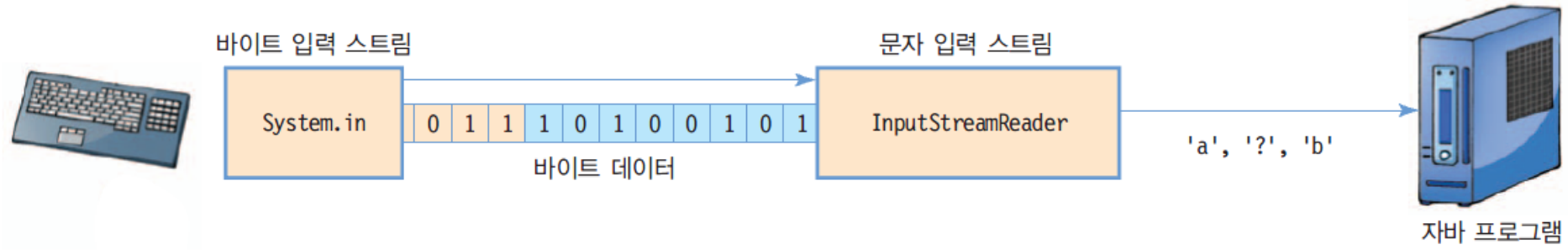


- 1차 Stream
  - 입/출력 통로를 직접 만드는 Class
- 2차 Stream
  - 기존의 통로를 이용하여 새로운 기능을 더하는 Class

Stream



- 표준 입력 Sream System.in에 InputStreamReader Stream을 연결하는 사례




```
InputStreamReader rd = new InputStreamReader(System.in);
int ch = rd.read();      // 키보드에서 문자 읽음
```

학습 정리

- java.io Package 개요
  - JAVA의 입출력은 Hardware와 독립적으로 설계되어 어떠한 Computer에서나 일관된 입출력을 수행
  - JAVA의 입출력은 Stream을 사용
  - Stream은 순서가 있는 일련의 Data 흐름을 의미
- File과 Directory
  - JAVA는 File과 Directory를 다루기 위해 File Class 제공
  - 다양한 Method를 이용하여 Directory와 File에 관한 정보를 얻을 수 있음
- Character Stream과 Byte Stream
  - Stream에는 Character Stream과 Byte Stream 2가지 형태가 있음
  - Character Stream은 16 Bit 문자나 문자열들을 읽고 쓰기 위한 Stream이고, Byte Stream(또는 Binary Stream)은 8 Bit의 Byte를 읽고 쓰기 위한 Stream

학습 정리

- Character Stream
  - Writer와 Reader Class는 Character Stream 입출력을 대표하는 Abstract Class로서 다양한 Method를 가지고 있음
  - 이러한 Abstract Class는 Sub Class에서 Overriding되어 사용
  - FileWriter Class와 FileReader Class는 File에 Character Stream을 입 출력하기 위해 사용하는 Class

학습 정리

- Byte Stream
  - OutputStream Class와 InputStream Class는 Byte Stream 입출력을 대표하는 Abstract Class로서 다양한 Method를 가지고 있음
  - 이러한 Abstract Class는 Sub Class에서 Overriding되어 사용
  - FileOutputStream Class와 FileInputStream Class는 File에 Byte Stream을 입 출력하기 위해 사용하는 Class
  - DataOutputStream Class와 DataInputStream Class는 JAVA의 기본 Data Type Data를 Byte로 입 출력하기 위해 사용하는 Class
  - OjbectOutputStream Class와 OjbectInputStream Class는 Object를 입 출력하기 위한 Class
  - Object를 입 출력하기 위해 JAVA는 직렬화된 Data를 사용