



# 데이터 입출력(I/O)





- **Stream** 개념을 안다.
- 입력(**InputStream**)와 출력(**OutputStream**)의 기본 메서드와 사용방법을 안다.
- **File** 관련된 내용의 객체와 메서드의 기능과 내용을 정확하게 파악하여 파일을 전송하거나 복사할 수 있다.
- **File**의 내용을 **Stream** 객체를 이용해서 읽어오거나 입력할 수 있다.



## 생각해 봅시다 :

- **Stream**으로 어떤 데이터들이 전송할 수 있을까? **binary data**
  - 다양한 전송할 때, 필요한 기능들은 어떤 것이 있을까? **Buffer**
    - data 단위별 숫자, 문자, image, 동영상
    - 한 자, 한 라인, 한번에 담을 수 있는 buffer 용량지정
    - 로컬(내부 pc), 네트워크
  - **Exception**는 과연 필요로 하는지?
    - 자바에서는 필수 처리.
-



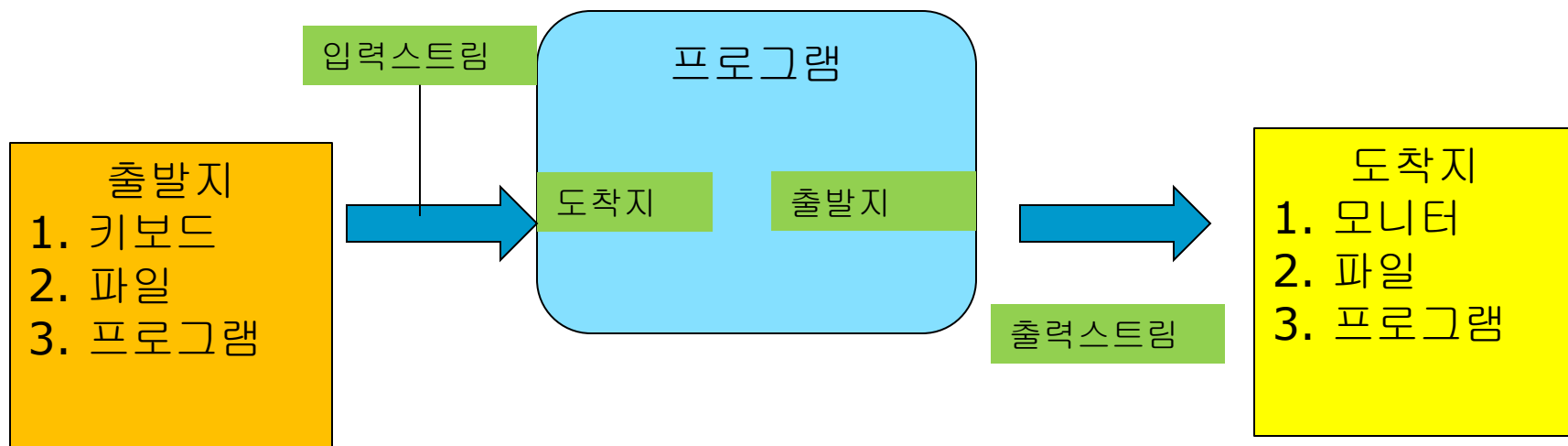
## IO 패키지 :

- 프로그램에서 데이터를 외부에 읽고 다시 외부로 출력하는 작업이 일어난다.
- 데이터 입력
  - 사용자 키보드 입력
  - 파일 또는 네트워크를 통해서 입력
- 데이터 출력
  - 모니터/프린터를 통해 출력
  - 파일 출력, 네트워크 출력 및 전송
- **Stream** : 자바에서 데이터를 입출력해 주는 객체
  - 단일 방향으로 연속적으로 흘러가는 것을 말함.



# IO 패키지 :

- 단일 방향으로 연속적으로 흘러가는 것을 말함.
  - ex) 물이 높은 곳에서 낮은 곳으로 흘러가듯 데이터 출발지에서 나와 도착지로 들어간다는 개념





# 자바의 기본적인 데이터 입출력(**IO**):

- 자바에서는 입출력 API는 **java.io** 패키지를 통해서 제공하고 있다.
    - **File**클래스 : 파일시스템의 정보
    - 데이터를 입출력하기 위한 다양한 입출력 스트림 클래스
-



# java.io 패키지 주요 클래스 :

패키지명	내용
File	파일 시스템(파일,폴드)의 정보를 얻기 위한 클래스
Console	콘솔로부터 문자를 입출력하기 위한 클래스
InputStream/OutputStream	바이트 단위 입출력 하기 위한 최상위 입출력 클래스
FileInputStream/FileOutputStream DataInputStream/DataOutputStream ObjectInputStream/ObjectOutputStream PrintStream BufferedInputStream/BufferedOutputStream	바이트 단위 입출력을 위한 하위 스트림 클래스
Reader/Writer	문자 단위 입출력을 위한 최상위 스트림
FileReader/FileWriter InputStreamReader/OutputStreamWriter PrintWriter BufferedReader/BufferedWriter	문자 단위 입출력 위한 하위 스트림클래스



## 스트림 클래스 2가지 종류 :

- byte 기반 스트림
  - 그림, 멀티미디어, 문자 등 모든 종류의 데이터를 입력, 출력
- 문자(character) 기반 스트림
  - 문자만 받고 보낼 수 있음.



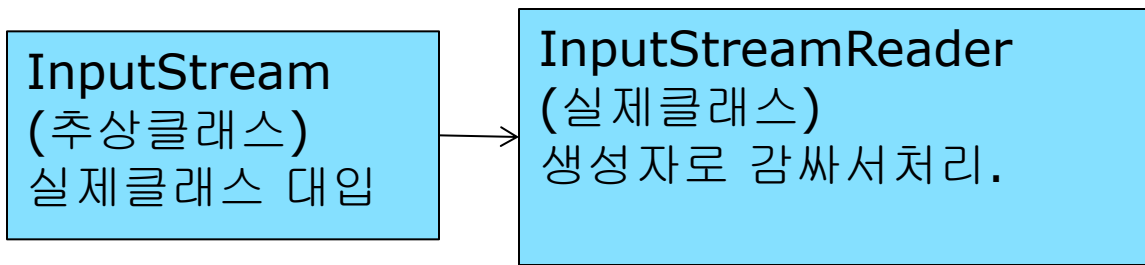


# InputStream :

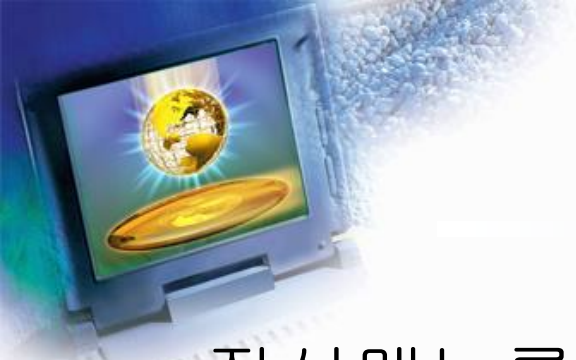
- 바이트 기반 입력 스트림의 최상위 클래스로 추상 클래스
- 메서드
  - read() : 1byte를 읽어 들임.
  - read(byte[] b) : byte[]b에 문자를 저장
  - read(byte[]b, int off, int len) : byte[]b에 범위(시작:off, 길이:len)를 정하여 입력받게 처리
  - close() : 스트림 자원해제
- System.in 통해서 read()처리시.
  - 문자 한자 char 입력 3개 으로 인식
  - enter키 : 13(캐리지리턴)+10(라인피드) ==> char값 2개인식

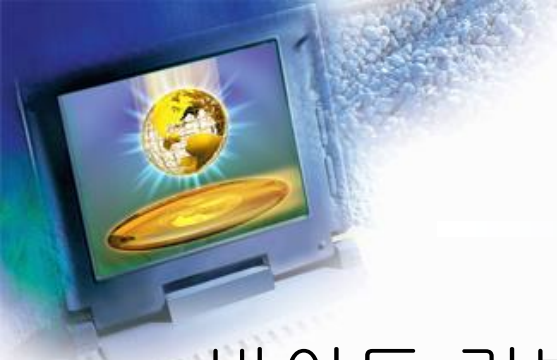


## 스트림 구조 객체 활용 :



- InputStream **in** = System.in;
- InputStreamReader reader = new InputStreamReader(**in**);

- 
- 점심메뉴를 입력 받아 출력하세요
    - 오늘의 점심은?:
    - 선택한 점심은 @@@



# OutputStream :

- 바이트 기반 출력 스트림의 최상위 클래스
    - 하위: **FileOutputStream**, PrintStream, BufferedOutputStream, DataOutputStream
  - 메서드
    - write(int b): 1바이트 단위로 출력처리..
    - write(byte[] b) : 바이트 배열 만큼 전송하여 출력처리
    - write(byte[] b, int off, int len) : 출력 스트림으로 주어진 바이트 배열의 크기 만큼 보낸다.
    - flush() : 버퍼에 잔류하는 모든 바이트 출력.
    - close() : 자원 반납 출력 스트림을 닫는다.
-



## 파일에 자바의 문자열 출력 :

- `FileOutputStream("txt경로")`
- 파일에 보낼 문자
  - `byte[] data = "ABC".getBytes();`
    - 문자열을 `byte[]` 변경처리..
- `FileOutputStream`
  - `write(data[index])` : 파일에 등록처리..

- **System.in** 활용하여
  - 데이터를 `byte[]` 입력 받아서..
- **OutputStream, FileOutputStream** 을 활용하여  
특정 파일 `test02.txt`에 문자열을 저장하세요.



- 문자 기반 출력 스트림의 최상위 클래스
- 하위 클래스 : **Writer** 클래스를 상속
  - `FileWriter`, `BufferedWriter`, `PrintWriter`, `OutputStreamWriter`
- 주요 메서드
  - `write(int c)` : 스트림으로 한 문자를 보내는 처리
  - `write(char[] cb)` : 스트림으로 문자 배열을 보내는 처리
  - `write(char[] cb, int off, int len)` : 스트림으로 문자 배열을 부분을 추출해서 보내는 처리
  - `write(String str)` : 문자열 보내는 처리
  - `write(String str, int off, int len)` : 스트림으로 문자열 추출
  - `flush()` : 버퍼에 잔류하는 모든 문자열 출력
  - `close()` : 사용한 시스템 자원을 반납, 출력 스트림 닫기



# 콘솔 입출력 :

## ■ 콘솔

- 시스템을 사용하기 위해 키보드로 입력/화면으로 출력하는 소프트웨어
- 유닉스/리눅스 ==> 터미널, windows ==> 명령프롬프트
- eclipse에서 키보드 입력 받는 내용/출력을 지원,

## ■ System.in 필드

- 자바 프로그램이 콘솔로부터 데이터를 입력
  - ex) `InputStream is = System.in;`
  - `int ascii = is.read(); char inC=(char)ascii;`

## ■ System.out 필드

- 자바 프로그램이 콘솔로부터 데이터를 출력
  - ex) `OutputStream os=System.out;`





## 콘솔 입출력 :

- Console클래스

- 자바6부터 문자열 쉽게 읽을 수 있게 지원
- `java.io.Console`클래스
- ex) `Console con = System.console();`

- 메서드

- `readLine()` : Enter키 입력 전에 모든 문자열 읽음
  - `readPassword()` : 키보드 입력내용을 콘솔에 보여 주지 않고 읽음
-



## 콘솔 입출력 :

- Scanner 클래스

- 콘솔로부터 문자열 읽을 수 있음.

- `java.util.Scanner`

- 입력

- `Scanner scan = new Scanner(System.in);`

- 메서드

- `nextXXXX()` 메서드를 통해서 해당 데이터 type 맞게 처리

- `nextInt()` : int값으로 읽는다.

- `nextDouble()` : double값 읽는다

- `nextLine()` : 문자열 값을 읽는다.

---



## 파일 입출력 ●

### ■ java.io.File 클래스

- 파일 : 크기, 속성, 파일 이름 정보, 생성, 삭제
- 디렉토리 : 생성, 디렉토리에 포함된 파일 리스트

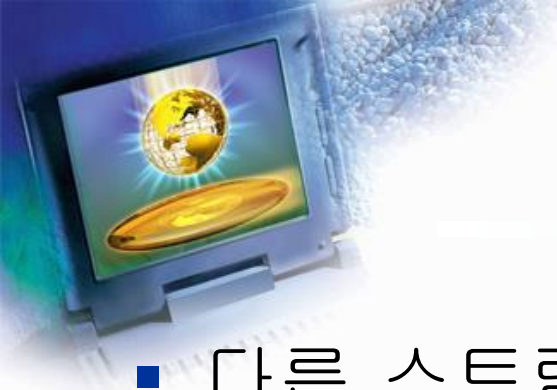
### ■ 생성

- `File f01 = new File("경로명/파일명");`
  - 물리적인 파일이나 디렉토리 생성X, 만일, 해당 파일이 있으면 인식
  - `.exists()` : 현재 파일이나 디렉토리가 있는지 여부를 `boolean(true/false)`
  - `.createNewFile()` : 물리적인 파일이나 디렉토리 생성



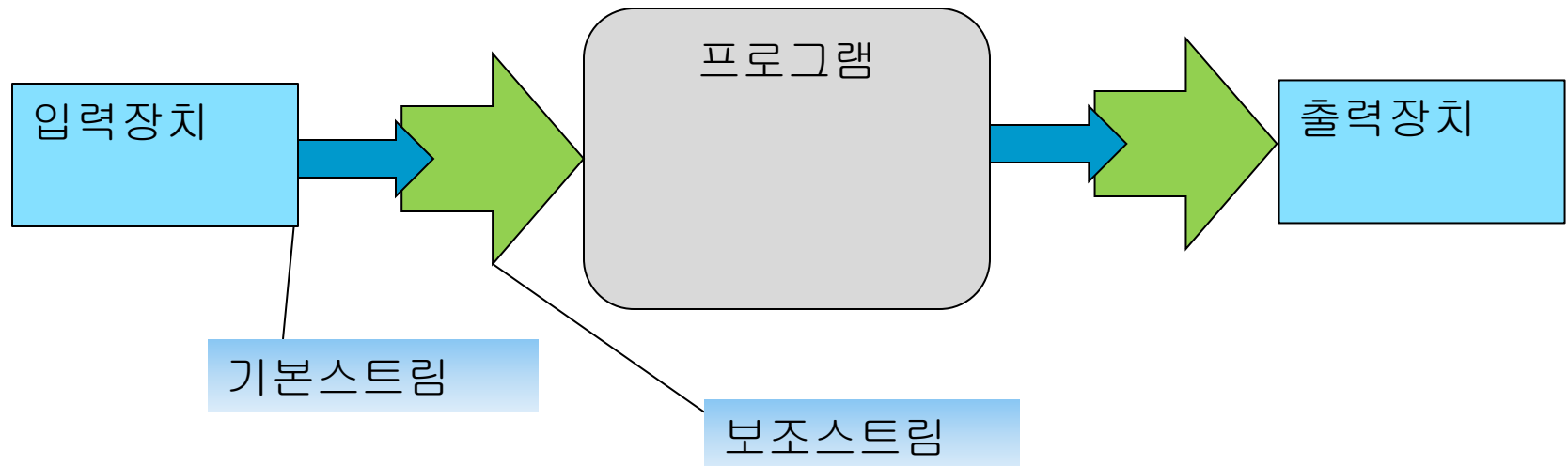
## 파일 입출력 :

- 메서드(생성/삭제)
    - `createNewFile()` : 물리적 파일 생성
    - `mkdir()` : 물리적 디렉토리 생성
    - `delete()` : 파일 또는 디렉토리 삭제
  - 메서드(정보)
    - `canExecute()` : 실행할 수 있는 파일여부(boolean)
    - `getName()` : 파일의 이름을 리턴
    - `getPath()` : 전체 경로를 리턴
    - `isFile()`, `isDirectory()` : 파일/디렉토리 인지 여부
    - `length()` : 파일의 크기
    - `list()` : 디렉토리인 경우에 포함된 파일의 문자열배열
-



## 보조 스트림 :

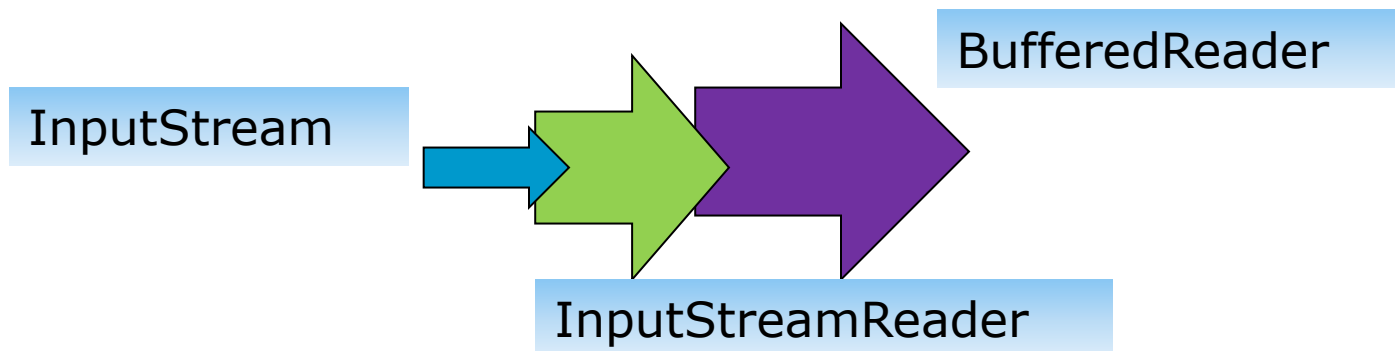
- 다른 스트림과 연결되어 여러가지 편리한 기능을 제공하는 스트림을 말한다. 필터 스트림이라고 하기도 한다. 기본스트림에 상속받아 **문자변환, 입출력 기능 향상, 기본 데이터타입 입출력, 객체 입출력** 등의 기능





## 보조 스트림과 함께 :

- 스트림(기본 클래스) `InputStream`
  - 하위 클래스(보조스트림1) : `InputStreamReader`
  - 하위 클래스(보조스트림2) : `BufferedReader`
- `InputStream is = System.in;`
- `InputStreamReader reader = new InputStreamReader( is );`
- `BufferedReader br = new BufferedReader( reader );`





## 확인예제(A14\_DataSteamExp) :

- 1단계 아래 데이터를 prodData.dat에 저장하고 호출

물건명	가격	갯수
사과	3000	2
바나나	4000	3
딸기	12000	3

- 2단계 **Scanner** 클래스를 활용하여 위와 같은 데이터를 입력하여 데이터 파일에 저장, 저장된 파일의 데이터를 불러서 리스트하는 처리..



1. 입출력 스트림에 대한 설명 중 틀린 것은 무엇입니까?
    - 1) 하나의 스트림으로 입력과 출력이 동시에 가능하다.
    - 2) 프로그램을 기준으로 데이터가 들어오면 입력 스트림이다.
    - 3) 프로그램을 기준으로 데이터가 나가면 출력 스트림이다.
    - 4) 콘솔에 출력하거나, 파일에 저장하려면 출력 스트림을 사용해야 한다.
  2. **InputStream**과 **Reader**에 대한 설명으로 틀린 것은 무엇입니까?
    - 1) 이미지데이터는 **InputStream** 또는 **Reader**로 모두 읽을 수 있다.
    - 2) **Reader**의 **read()**메소드는 **1**문자를 읽는다.
    - 3) **InputStream**의 **read()** 메소드는 **1**바이트를 읽는다.
    - 4) **InputStreamReader**를 이용하면 **InputStream**을 **Reader**로 변환시킬 수 있다.
-





## 1. **InputStream**의 **read(byte[] b, int off, int len)** 메소드에 대한 설명으로 틀린 것은 무엇입니까?

- 1) 메소드의 리턴값을 읽는 바이트 수이다.
- 2) 첫번째 매개값 **b**에는 읽은 데이터가 저장된다.
- 3) 두번째 매개값 **off**에는 첫번째 매개값 **b**에서 데이터가 저장될 시작 인덱스이다.
- 4) 세번째 매개값 **len**은 첫번째 매개값 **b**에서 데이터가 저장된 마지막 인덱스이다.

## 2. 출력 스트림에서 데이터 출력 후, **flush()** 메소드를 호출하는 이유가 무엇입니까?

- 1) 출력 스트림의 버퍼에 있는 데이터를 모두 출력시키고 버퍼를 비운다
- 2) 출력 스트림을 메모리에서 제거한다.
- 3) 출력 스트림의 버퍼에 있는 데이터를 모두 삭제한다.
- 4) 출력 스트림을 닫는 역할을 한다.



## 참고자료:

---



다음장은 기본문법 활용입니다!