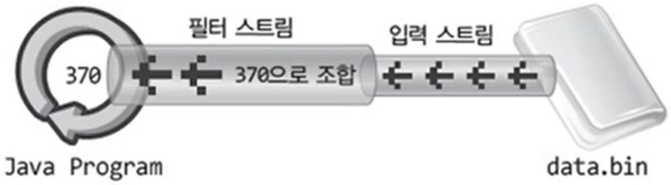
Java I/O Stream Ⅱ - FilterStream

Byte만 입출력 할 수 있나?

- 앞서 살펴본 InputStream, OutputStram은 Byte 또는 Byte 배열 단위로 데이터를 입출력을 진행했다.
- 그런데 길이가 4인 byte[] 형 배열이 있다고 해서 조합된 int형 데이터가 아니다!
- 그래서 int형 데이터로 입출력을 진행하기 위해서 byte[] 배열을 하나의 int형 데이터로 조합해야 한다.

FilterStream

- **Java Program과 I/O Stream 사이에서 I/O Stream에서 전달하는 byte 데이터를 원하는 자료형으로 재조합하여 전달해주는 클래스**



필터 스트림은 물이 뿜어져 나오는 호스에 연결된 샤워기 꼭지에 비유할 수 있다.

- 필터 스트림도 입력용, 출력용으로 구분된다.

필터 입력 스트림	입력 스트림을 연결하는 필터 스트림
필터 출력 스트림	출력 스트림을 연결하는 필터 스트림

- 필터 스트림 생성하기

i. 기본 입출력 스트림 인스턴스 생성

ii. 필터스트림 인스턴스를 생성하면서 생성자에 기본 스트림의 참조값을 전달하여 연결

FilterStream의 생성과 Stream 연결하기

DataInputStream, DataOutputStream 클래스는 Byte 데이터를 기본 자료형 단위로 재조합하여 데이터 입출력을 가능하게 하는 클래스이다.

```
public class FilterStreamDemo {
     private static final String FILE_PATH = "D:/Documents/data.bin";

     public static void main(String[] args) throws IOException {
          OutputStream out = new FileOutputStream(FILE_PATH);
          DataOutputStream filterOut = new DataOutputStream(out);
```

`filterOut.writeInt(275);`	int형 275 → byte 단위로 변환해서 쓰기
`filterOut.writeDouble(45.79);`	double형 45.79 → byte 단위로 변환해서 쓰기

```
        filterOut.close();

        InputStream in = new FileInputStream(FILE_PATH);
        DataInputStream filterIn = new DataInputStream(in);
```

`int num1 = filterIn.readInt();`	byte 단위 → int 단위로 조합해서 읽기
`double num2 = filterIn.readDouble();`	byte 단위 → double 단위로 조합해서 읽기

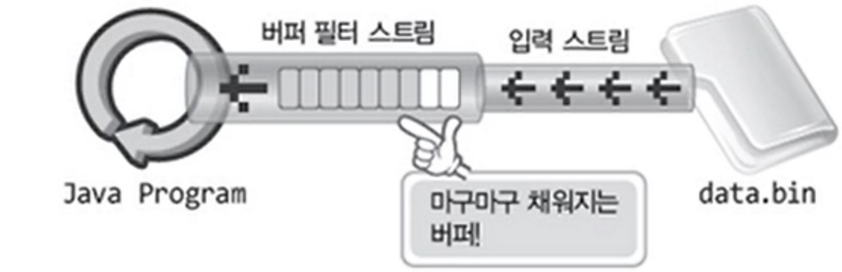
```
        filterIn.close();

        System.out.println(num1);
        System.out.println(num2);
    }
}
```

버퍼링 기능을 제공하는 FilterStream - BufferedStream

- **Buffer (버퍼, 완충기)**
  데이터를 한 곳에서 다른 한 곳으로 전송하는 동안 일시적으로 그 데이터를 보관하는 메모리영역이고 버퍼를 이용하여 일정량의 Byte 데이터를 일시보관하고 버퍼의 데이터를 받는 것을 버퍼링이라고 한다.

- 데이터를 1Byte 씩 전송하기에는 너무 속도가 느리다.
- 그래서 속도의 향상을 위해 버퍼를 수행하는 BufferedStream을 연결하여 데이터를 미리 버퍼링한다.




```
public class BufferedStreamDemo {
    private static final String ORG_FILE_PATH = "D:/Documents/org.txt";
    private static final String COPY_FILE_PATH = "D:/Documents/copy.txt";
```

```
    public static void main(String[] args) throws IOException {
        InputStream in = new FileInputStream(ORG_FILE_PATH);
        OutputStream out = new FileOutputStream(COPY_FILE_PATH);

        BufferedInputStream bis = new BufferedInputStream(in);
        BufferedOutputStream bos = new BufferedOutputStream(out);

        int bData;
        int byteCount = 0;

        while(true){
            bData = bis.read();
            if(bData == -1)
                break;

            bos.write(bData);
            byteCount++;
        }

        bos.close();
        bis.close();
        out.close();
        in.close();

        System.out.println("복사된 바이트 크기 : "+byteCount);
    }
}
```

둘 이상의 FilterStream 연결하기

- **기본 자료형 데이터 입출력 스트림**

```
OutputStream out = new FileOutputStream("data.bin");
DataOutputStream dos = new DataOutputStream(out);
```

- **버퍼 스트림**

```
OutputStream out = new FileOutputStream("data.bin");
BufferedOutputStream bos = new BufferedOutputStream(out);
```

- **어떤 필터 스트림에 먼저 연결할까?**
  - Java Program → BufferedOutputStream → DataOutputStream→ FileOutputStream X
    (위의 연결로는 데이터를 기본자료형 단위로 재조합을 할 수가 없다.)
  - **Java Program → DataOutputStream → BufferedOutputStream → FileOutputStream O**
    (데이터를 재조합 후, 버퍼 스트림에 데이터를 쓰고, 버퍼 스트림이 출력스트림에 쓰는 과정이 되어야한다.)

- **생성자와 상속관계를 통해 확인하기**
  DataOutputStream은 OuputStream을 상속하는 모든 인스턴스를 생성자의 인자로 받을 수 있다.
  BufferedStream 또한 OutputStream을 상속하는 클래스 이므로 인자로 전달 가능하다!

DataOutputStream

public DataOutputStream(OutputStream out)

Constructor Detail

DataOutputStream

public **DataOutputStream**(OutputStream out)

java.io

Class BufferedOutputStream

java.lang.Object
  └ java.io.OutputStream
    └ java.io.FilterOutputStream
      └ **java.io.BufferedOutputStream**