14-1. 입출력 스트림

혼자 공부하는 자바 (신용권 저)

목차

시작하기 전에

[핵심 키워드] : 입출력 스트림, InputStream, OutputStream, Reader, Writer

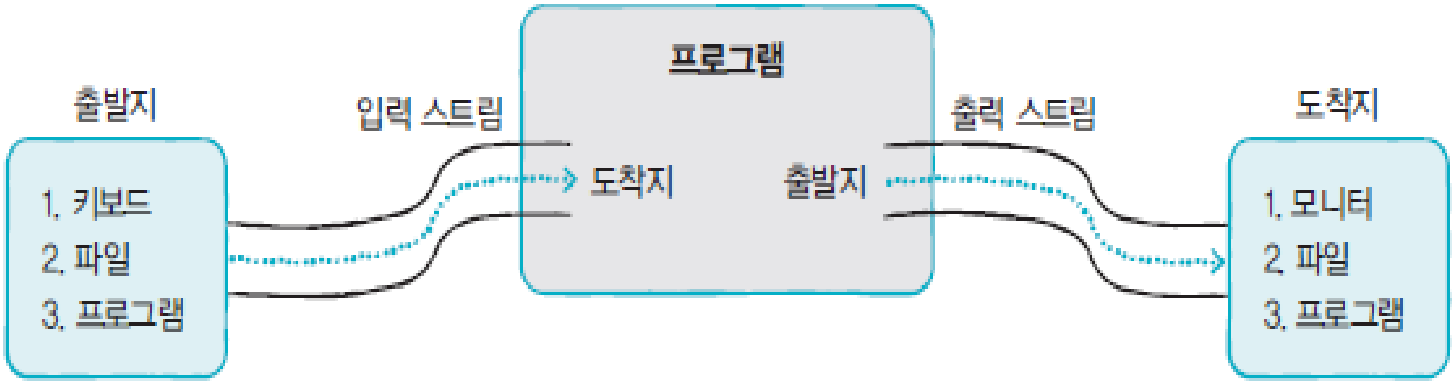
[핵심 포인트]

프로그램은 데이터를 읽고 출력하는 작업을 빈번히 수행한다. 데이터를 읽고 출력하기 위해 사용되는 입출력 API에 대해 알아본다.

시작하기 전에

❖ 스트림 (Stream)

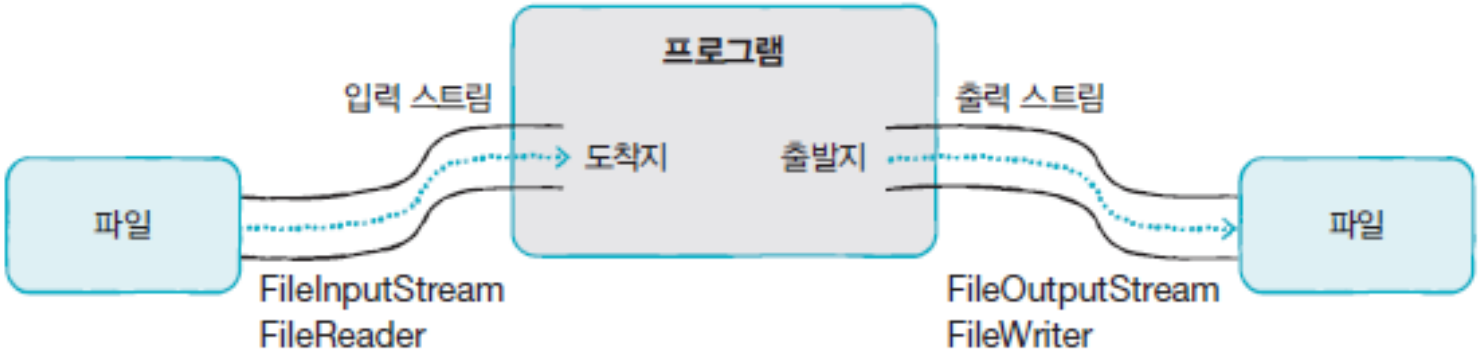
- 자바에서 데이터는 스트림을 통해 입출력됨
- 프로그램이 데이터의 출발지인지 도착지인지의 여부에 따라 사용하는 스트림의 종류가 결정



입출력 스트림의 종류

- 바이트 기반 스트림
  - 그림, 멀티미디어 등의 바이너리 데이터를 읽고 출력
- 문자 기반 스트림
  - 문자 데이터를 읽고 출력할 때 사용
- 최상위 클래스로 스트림 클래스의 바이트 / 문자 기반 판단

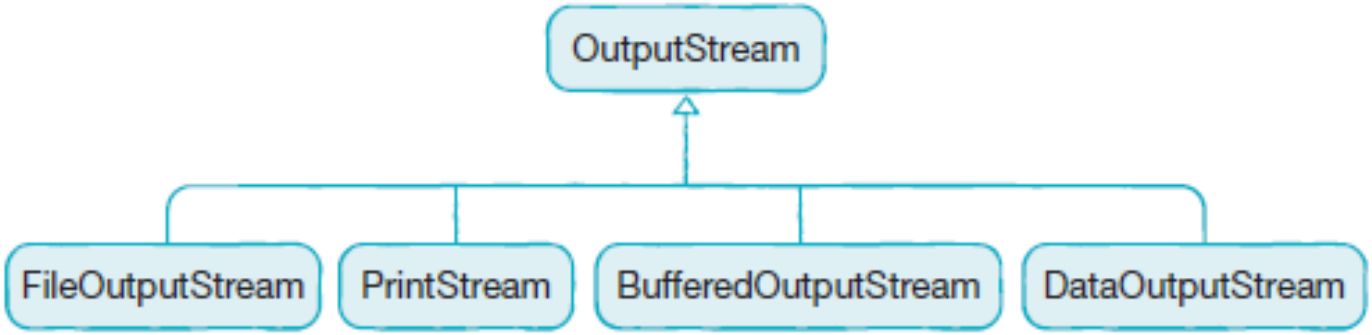
구분	바이트 기반 스트림		문자 기반 스트림	
	입력 스트림	출력 스트림	입력 스트림	출력 스트림
최상위 클래스	InputStream	OutputStream	Reader	Writer
하위 클래스 (예)	XXXInputStream (FileInputStream)	XXXOutputStream (FileOutputStream)	XXXReader (FileReader)	XXXWriter (FileWriter)



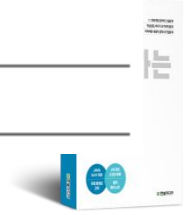
바이트 출력 스트림 : OutputStream

❖ OutputStream

- 바이트 기반 출력 스트림의 최상위 클래스
- 모든 바이트 기반 출력 스트림 클래스는 OutputStream 클래스 상속받음



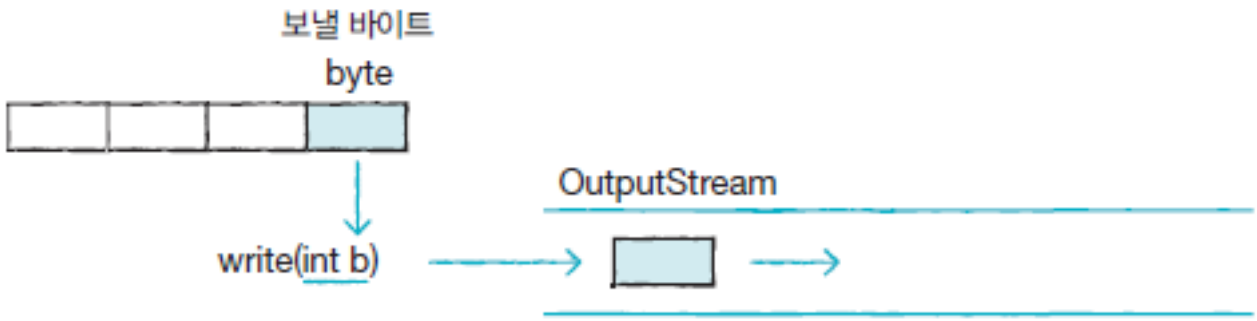
리턴 타입	메소드	설명
void	write(int b)	1byte를 출력합니다.
void	write(byte[ ] b)	매개값으로 주어진 배열 b의 모든 바이트를 출력합니다.
void	write(byte[ ] b, int off, int len)	매개값으로 주어진 배열 b[off]부터 len개까지의 바이트를 출력합니다.
void	flush( )	출력 버퍼에 잔류하는 모든 바이트를 출력합니다.
void	close( )	출력 스트림을 닫습니다.



바이트 출력 스트림 : OutputStream

❖ write(int b) 메소드

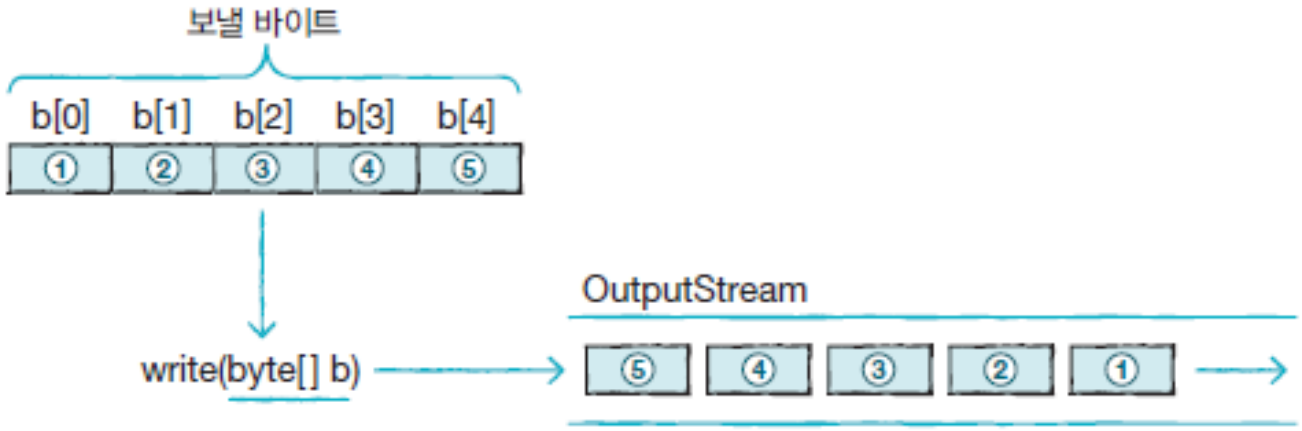
- 매개 변수로 주어지는 int(4byte)에서 끝 1byte만 출력 스트림으로 보냄



바이트 출력 스트림 : OutputStream

❖ write(byte[] b) 메소드

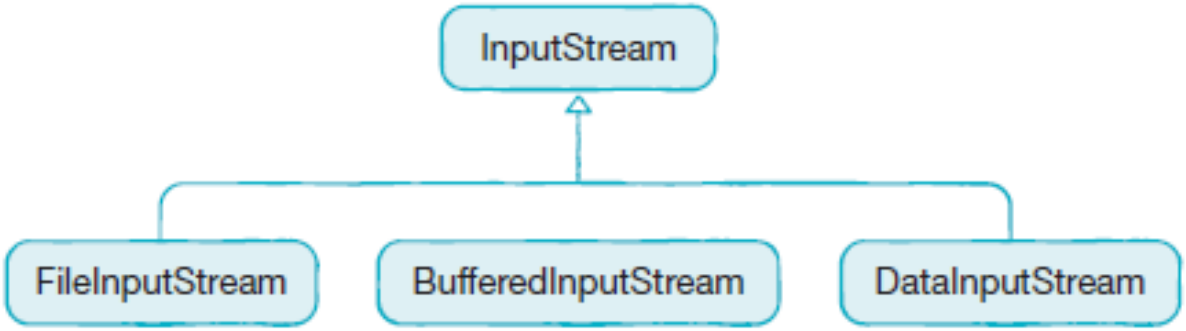
- 매개값으로 주어진 배열의 모든 바이트를 출력 스트림으로 보냄



바이트 입력 스트림 : InputStream

❖ InputStream

- 바이트 기반 입력 스트림의 최상위 클래스
- 모든 바이트 기반 입력 스트림은 InputStream 클래스 상속받아 만들어짐

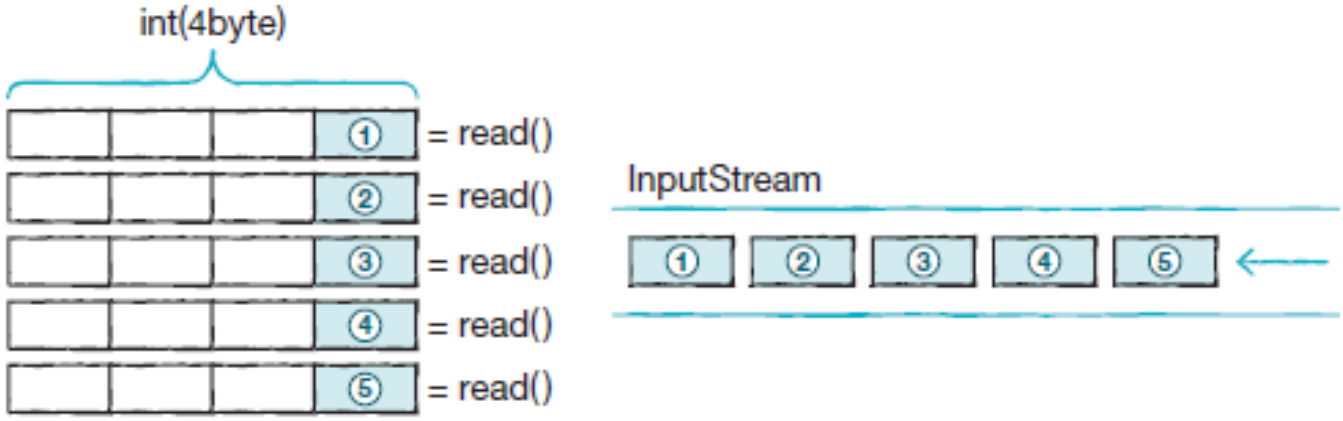


리턴 타입	메소드	설명
int	read()	1byte를 읽고 읽은 바이트를 리턴합니다.
int	read(byte[ ] b)	읽은 바이트를 매개값으로 주어진 배열에 저장하고 읽은 바이트 수를 리턴합니다.
int	read(byte[ ] b, int off, int len)	len개의 바이트를 읽고 매개값으로 주어진 배열에서 b[off]부터 len개까지 저장합니다. 그리고 읽은 바이트 수를 리턴합니다.
void	close()	입력 스트림을 닫습니다.

바이트 입력 스트림 : InputStream

❖ read() 메소드

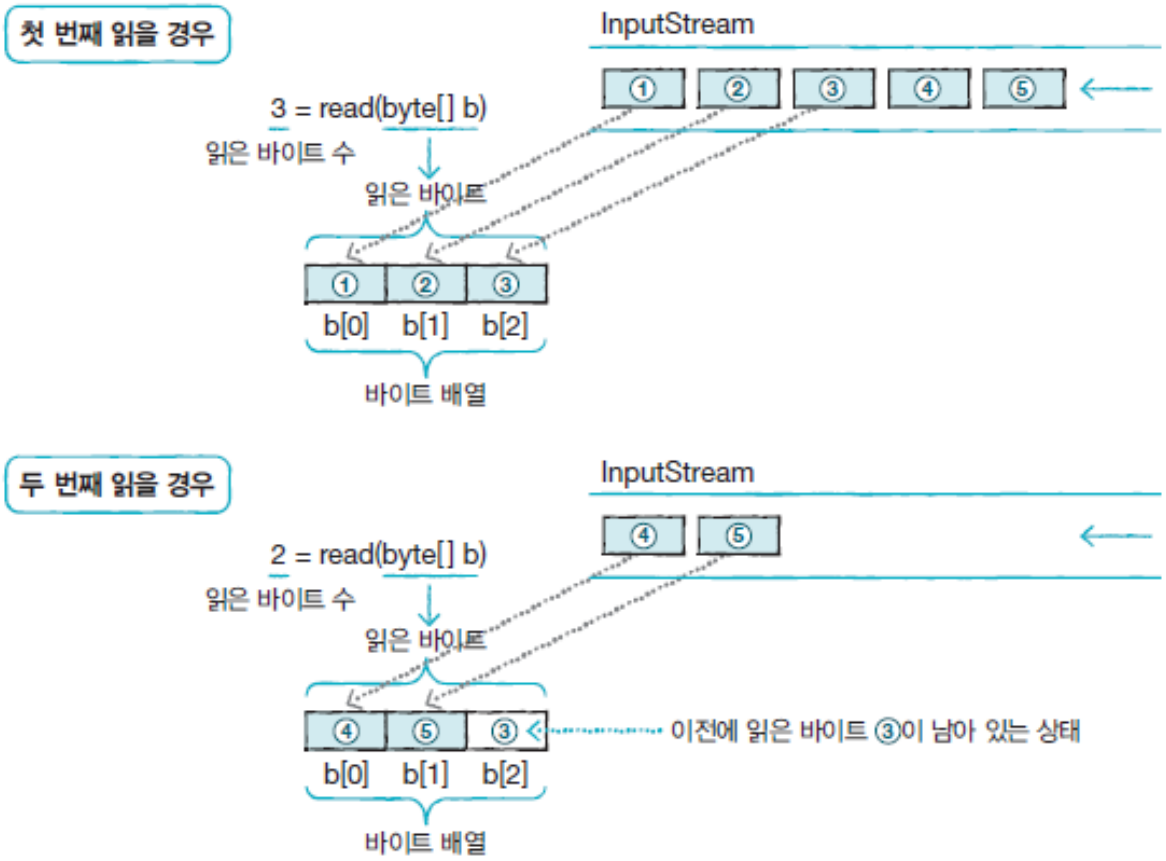
- 입력 스트림으로부터 1byte 읽고 int(4byte) 타입으로 리턴
- 리턴된 4byte 중 끝 1byte에만 데이터 들어 있음
- 더 이상 입력 스트림으로부터 바이트 읽을 수 없게 되면 -1 리턴
  - 읽을 수 있는 마지막 바이트까지 반복하여 1byte씩 읽을 수 있음



바이트 입력 스트림 : InputStream

❖ read(byte[] b) 메소드

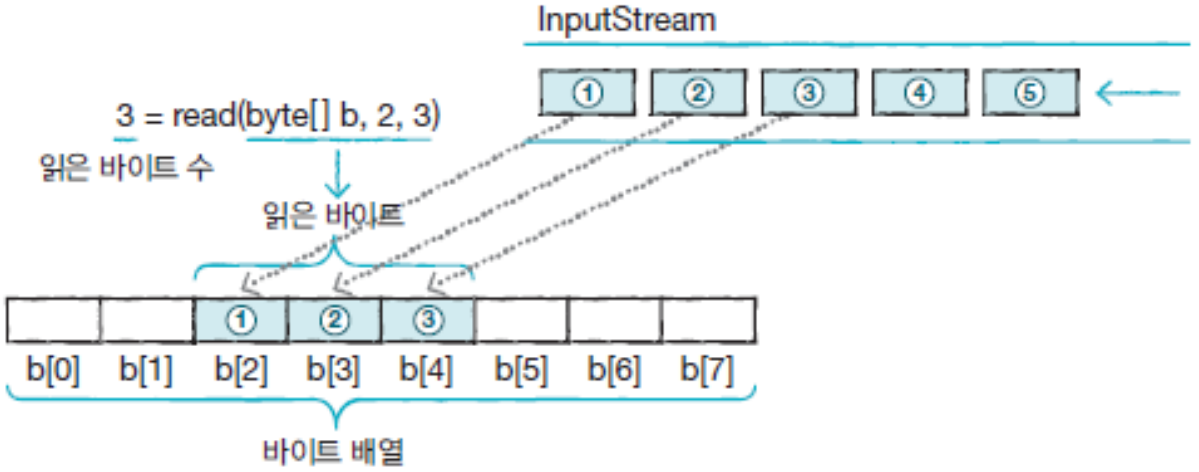
- 입력 스트림으로부터 매개값으로 주어진 배열의 길이만큼 바이트 읽고 해당 배열에 저장, 그리고 읽은 바이트 수를 리턴



바이트 입력 스트림 : InputStream

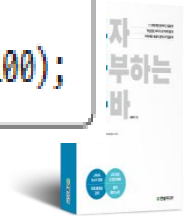
❖ read(byte[] b, int off, int len) 메소드

- 입력 스트림으로부터 len 개의 바이트만큼 읽고 매개값으로 주어진 바이트 배열 b[off]부터 len 개 까지 저장, 그리고 읽은 바이트 수인 len개 리턴




```
InputStream is = …;
byte[] readBytes = new byte[100];
int readByteNo=is.read(readBytes);
```

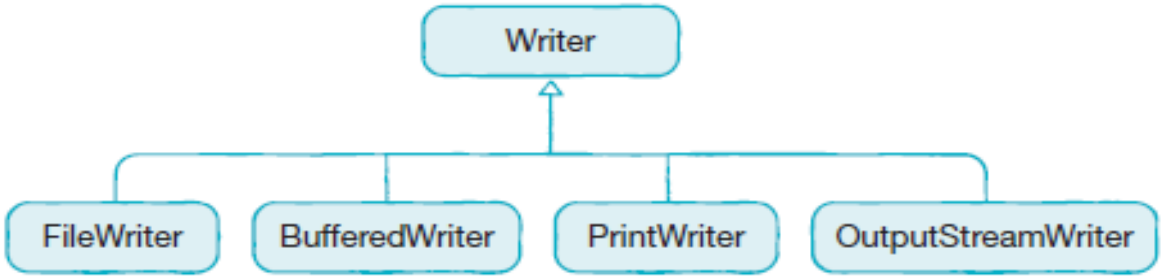
```
InputStream is = …;
byte[] readBytes = new byte[100];
int readByteNo=is.read(readBytes, 0, 100);
```



문자 출력 스트림 : Writer

❖ Writer

- 문자 기반 출력 스트림의 최상위 클래스
- 모든 문자 기반 출력 스트림 클래스는 Writer 클래스를 상속받아 만들어짐

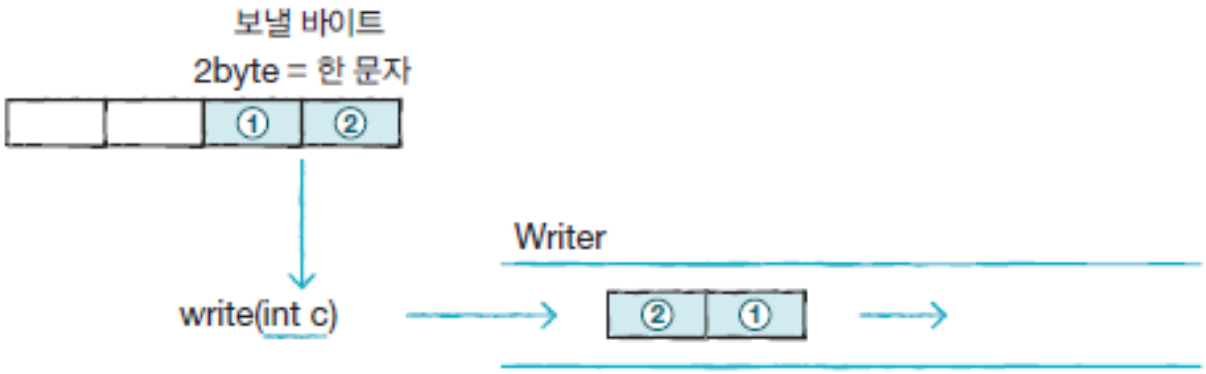


리턴 타입	메소드	설명
void	write(int c)	매개값으로 주어진 한 문자를 보냅니다.
void	write(char[ ] cbuf)	매개값으로 주어진 배열의 모든 문자를 보냅니다.
void	write(char[ ] cbuf, int off, int len)	매개값으로 주어진 배열에서 cbuf[off]부터 len개까지의 문자를 보냅니다.
void	write(String str)	매개값으로 주어진 문자열을 보냅니다.
void	write(String str, int off, int len)	매개값으로 주어진 문자열에서 off 순번부터 len개까지의 문자를 보냅니다.
void	flush( )	버퍼에 잔류하는 모든 문자를 출력합니다.
void	close( )	출력 스트림을 닫습니다.

문자 출력 스트림 : Writer

❖ writer(int c) 메소드

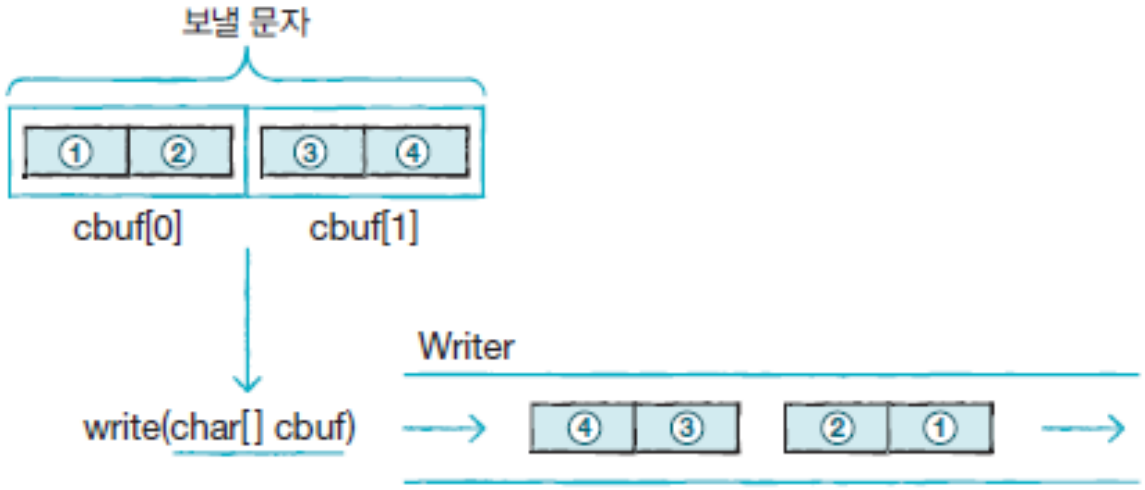
- 매개 변수로 주어지는 int(4byte)에서 끝 2byte(1개 문자)만 출력 스트림을 보냄



문자 출력 스트림 : Writer

❖ write(char[] cbuf) 메소드

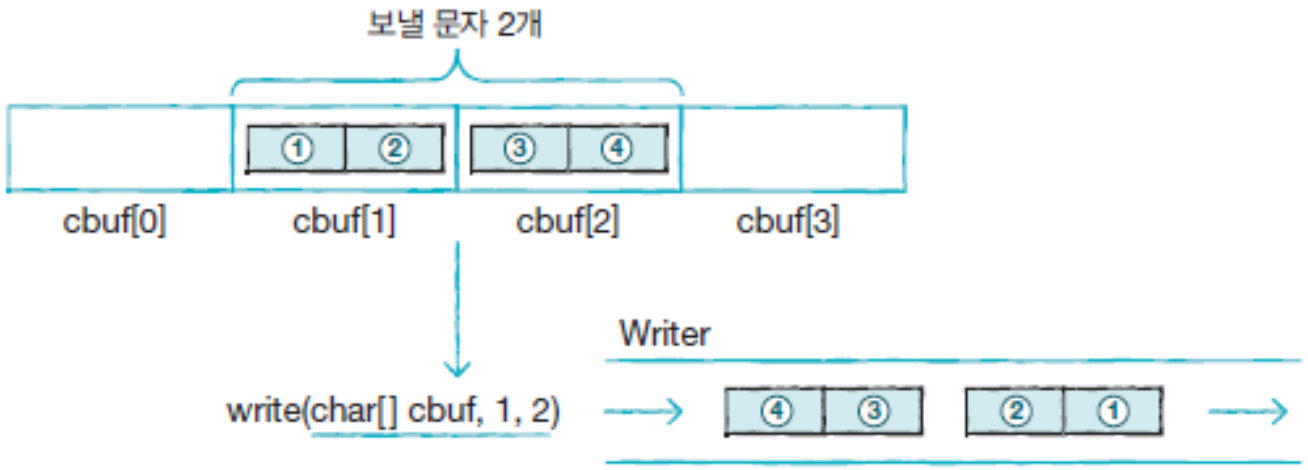
- 매개값으로 주어진 char[] 배열의 모든 문자를 출력 스트림으로 보냄



문자 출력 스트림 : Writer

❖ write(char[] cbuf, int off, int len) 메소드

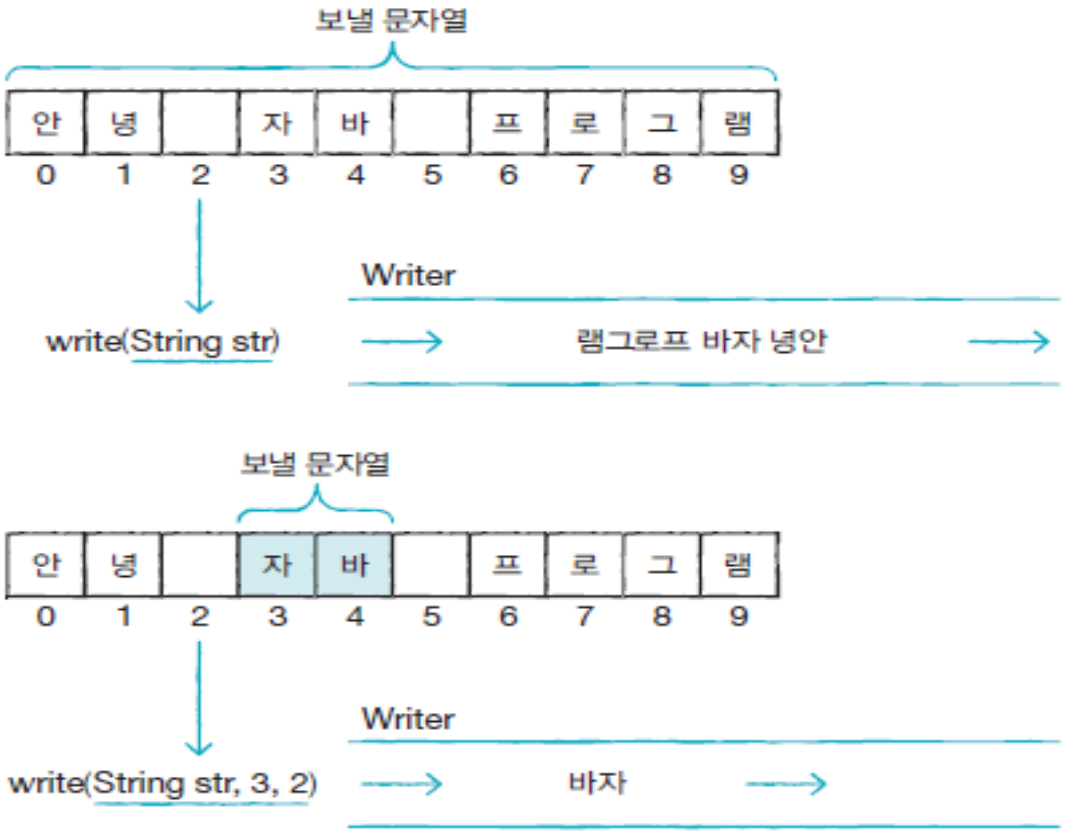
- c[off]부터 len개의 문자를 출력 스트림으로 보냄



문자 출력 스트림 : Writer

❖ write(String str)와 write(String str, int off, int len) 메소드

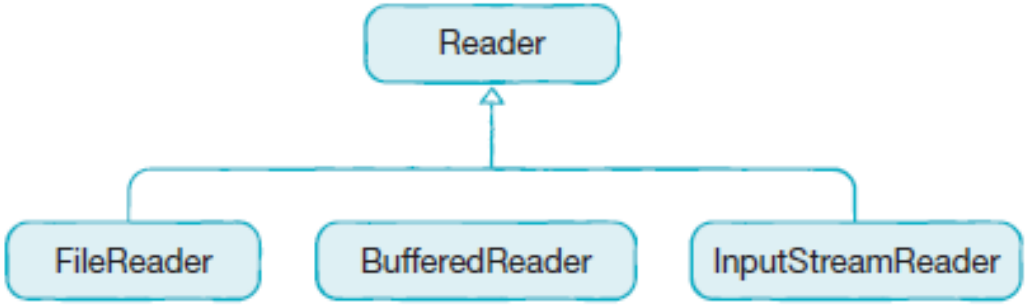
- write(String str)은 문자열 전체를 출력 스트림으로 보냄
- write(String str, int off, int len)은 주어진 문자열 off순번부터 len개까지의 문자 보냄



문자열 입력 스트림 : Reader

❖ Reader

- 문자 기반 입력 스트림의 최상위 클래스
- 모든 문자 기반 입력 스트림은 Reader 클래스 상속받아 만들어짐

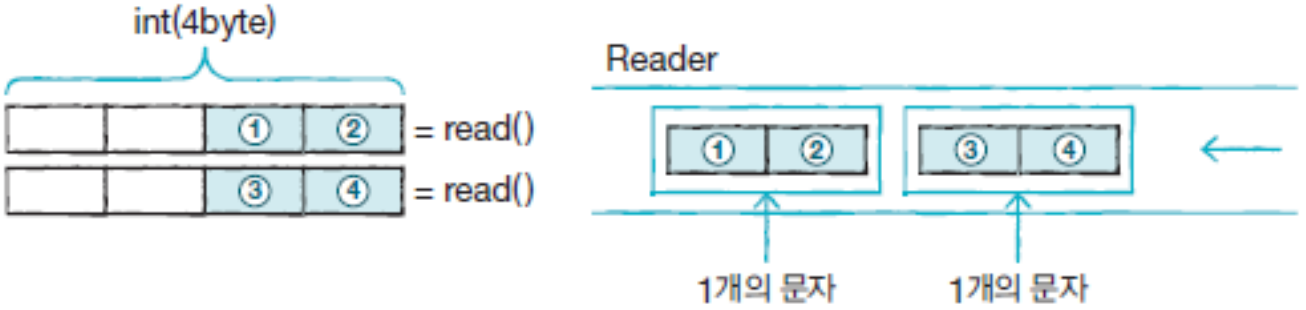


리턴 타입	메소드	설명
int	read()	1개의 문자를 읽고 리턴합니다.
int	read(char[] cbuf)	읽은 문자들을 매개값으로 주어진 문자 배열에 저장하고 읽은 문자 수를 리턴합니다.
int	read(char[] cbuf, int off, int len)	len개의 문자를 읽고 매개값으로 주어진 문자 배열에서 cbuf[off] 부터 len개까지 저장합니다. 그리고 읽은 문자 수를 리턴합니다.
void	close()	입력 스트림을 닫습니다.

문자열 입력 스트림 : Reader

❖ read() 메소드

- 입력 스트림으로부터 1개의 문자(2byte) 읽고 int(4byte) 타입으로 리턴
- 리턴된 4byte 중 끝 2byte에 문자 데이터 들어 있음



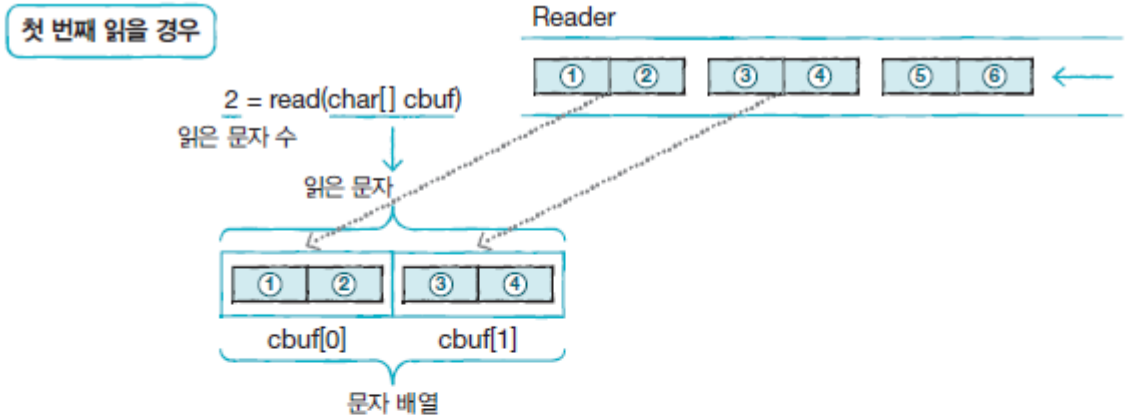
- read() 메소드가 리턴한 int 값 char 타입으로 변환하면 읽은 문자 얻을 수 있음

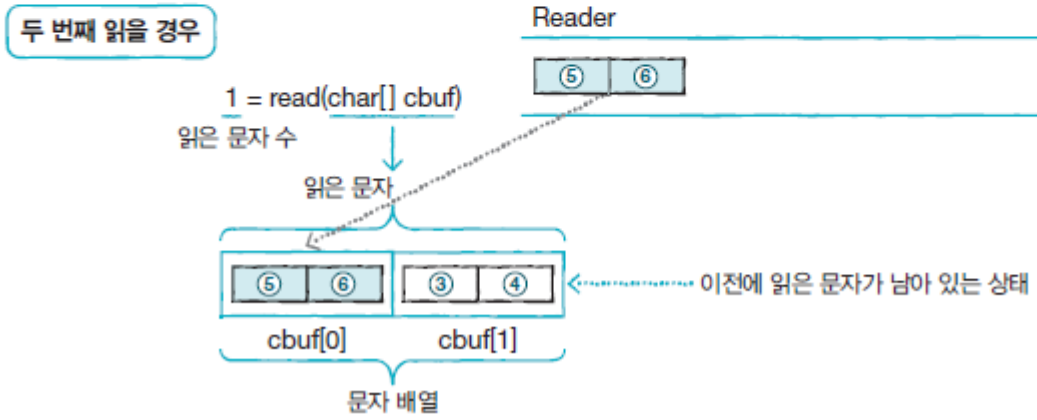
```
char charData = (char) read();
```

문자열 입력 스트림 : Reader

❖ read(char[] cbuf) 메소드

- 입력 스트림으로부터 매개값으로 주어진 문자 배열의 길이만큼 문자 읽고 배열에 저장, 그리고 읽은 문자 수를 리턴

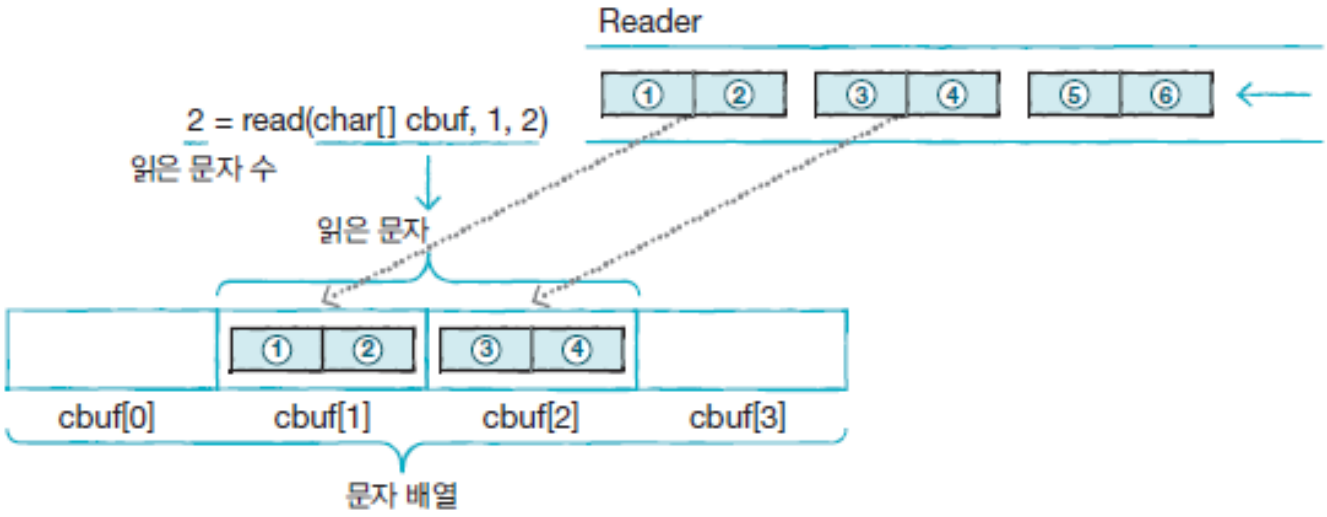




문자열 입력 스트림 : Reader

❖ read(char[] cbuf, intoff, intlen) 메소드

- 입력 스트림으로부터 len개의 문자만큼 읽고 매개값으로 주어진 문자 배열에서 cbuf[off]부터 len개까지 저장, 그리고 읽은 문자 수인 len개 리턴




```
Reader reader = …;
char[] cbuf = new char[100];
int readCharNo=is.read(cbuf);
```

```
Reader reader = …;
char[] cbuf = new char[100];
int readCharNo=is.read(cbuf, 0, 100);
```

키워드로 끝내는 핵심 포인트

- 입출력 스트림 : 자바에서 데이터는 스트림을 통해 입출력됨. 프로그램이 출발지인지 도착지인지 여부에 따라 사용하는 스트림의 종류가 결정됨.

- InputStream : 바이트 기반 입력 스트림의 최상위 클래스, 추상 클래스. 모든 바이트 기반 입력 스트림은 InputStream 클래스를 상속받아 만들어짐.

- OutputStream : 바이트 기반 출력 스트림의 최상위 클래스, 추상 클래스. 모든 바이트 기반 출력 스트림 클래스는 OutputStream 클래스 상속받아 만들어짐

- Reader : 문자 기반 입력 스트림의 최상위 클래스, 추상 클래스. 모든 문자 기반 입력 스트림은 Reader 클래스 상속받아 만들어짐

- Writer : 문자 기반 출력 스트림의 최상위 클래스, 추상 클래스. 모든 문자 기반 출력 스트림 클래스는 Writer 클래스 상속받아 만들어짐.

14-2. 보조 스트림

혼자 공부하는 자바 (신용권 저)

목차

시작하기 전에

[핵심 키워드] : 보조 스트립, 문자 변환, 성능 향상, 기본 타입 입출력, 개행 출력

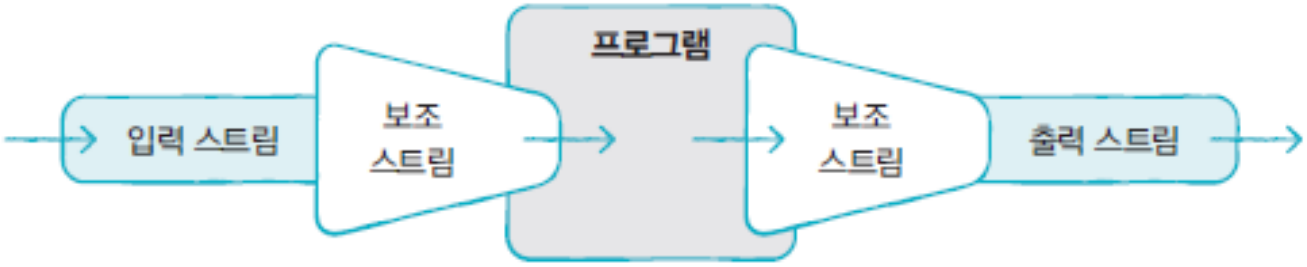
[핵심 포인트]

기본 스트림을 직접 사용해서 데이터를 입출력할 수도 있지만, 데이터를 변환해서 입출력하거나 데이터의 출력 형식을 지정하려는 경우 등에는 보조 스트림을 연결하여 사용하면 편리하다.

시작하기 전에

❖ 보조 스트림

- 다른 스트림과 연결되어 여러가지 편리한 기능을 제공하는 스트림
- 자체적으로 입출력 수행할 수 없기 때문에 입출력 소스와 바로 연결되는 InputStream, OutputStream, Reader, Writer 등에 연결하여 입출력 수행



- 프로그램은 입력 스트림으로부터 직접 데이터 읽지 않고, 보조 스트림에서 제공하는 기능 이용하여 데이터 읽음. 또한 출력 스트림으로 직접 데이터 보내지 않고 보조 스트림에서 제공하는 기능 이용하여 데이터 보냄

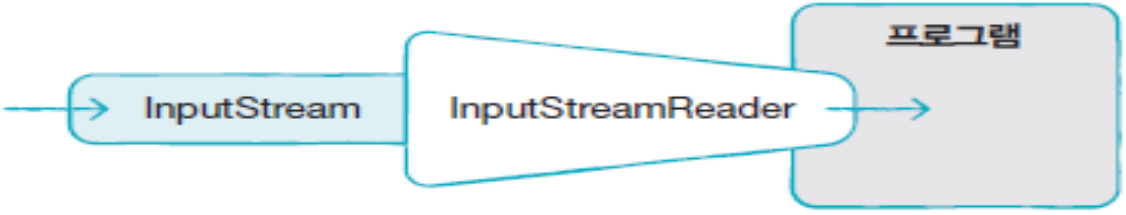
보조 스트림 연결하기

❖ 보조 스트림 연결하기

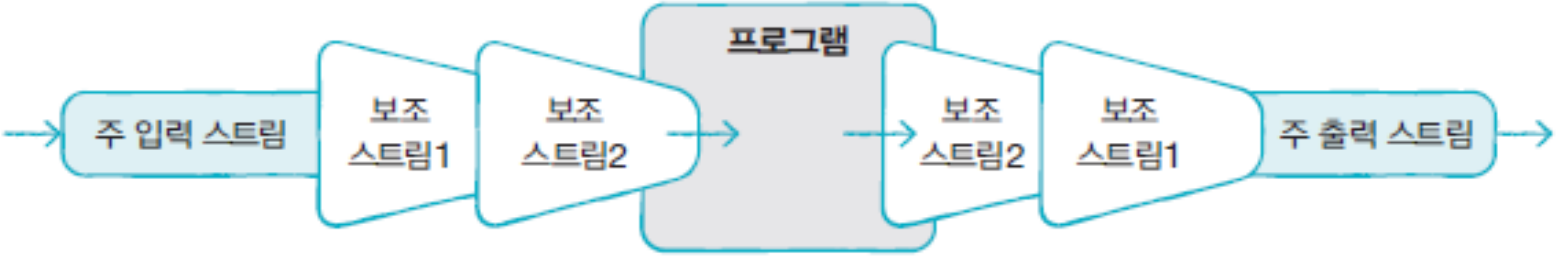
- 보조 스트림 생성 시 자신이 연결될 스트림을 생성자의 매개값으로 제공

```
보조스트림 변수 = new 보조스트림(연결스트림)
```

```
InputStream is = …;
InputStreamReader reader = new InputStreamReader(is);
```



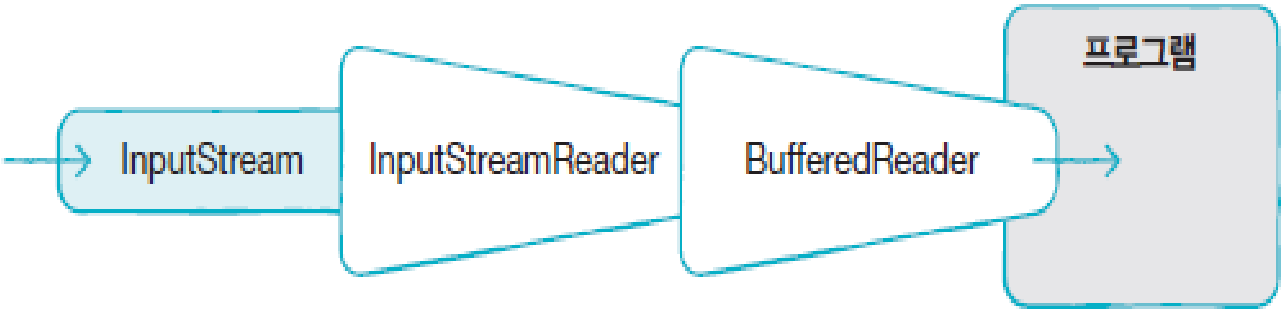
- 보조 스트림을 연속적으로 연결할 수도 있음



보조 스트림 연결하기

- 예시

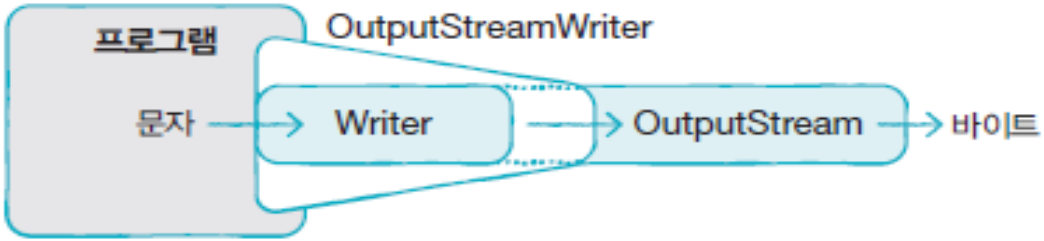
```
InputStream is = System.in;
InputStreamReader reader = new InputStreamReader(is);
BufferedReader br = new BufferedReader(reader);
```



문자 변환 보조 스트림

- 소스 스트림이 바이트 기반 스트림이면서 입출력 데이터가 문자일 경우 Reader와 Writer로 변환해서 사용하는 것이 편리함

- OutputStreamWriter
  - 바이트 기반 출력 스트림에 연결되어 문자 출력 스트림인 Writer로 변환하는 보조 스트림



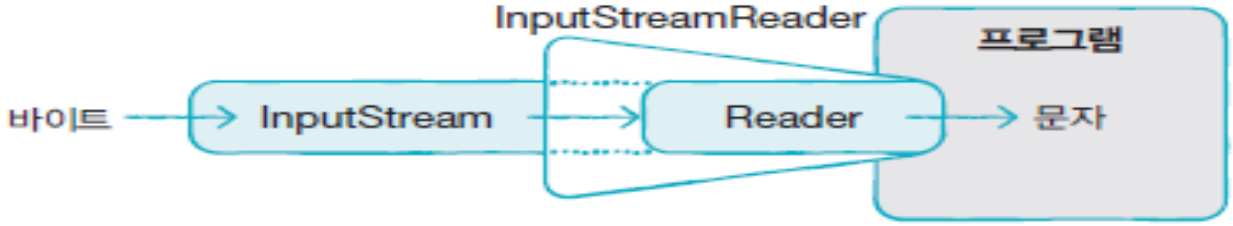

```
Writer writer = new OutputStreamWriter(바이트 기반 출력 스트림);
```

```
FileOutputStream fos = new FileOutputStream("C:/Temp/test1.txt");
Writer writer = new OutputStreamWriter(fos);
```

문자 변환 보조 스트림

❖ InputStreamReader

- 바이트 기반 입력 스트림에 연결되어 문자 입력 스트림인 Reader로 변환하는 보조 스트림




```
Reader reader = new InputStreamReader(바이트 기반 입력 스트림);
```

```
FileInputStream fis = new FileInputStream("C:/Temp/test1.txt);
Reader reader = new InputStreamReader(fis);
```

성능 향상 보조 스트림

❖ 성능 향상 보조 스트림

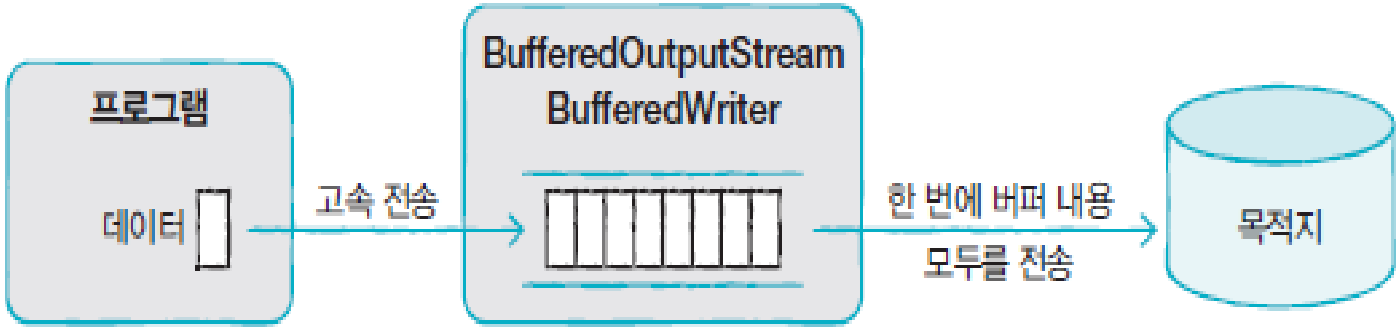
- 메모리 버퍼를 추가로 제공하여 프로그램의 실행 성능을 향상
- 바이트 기반 스트림에서는
  - BufferInputStream, ButteredOutputStream
- 문자 기반 스트림에서는
  - BufferReader, BufferWriter

❖ BufferedOutputStream과 BufferedWriter

- 바이트 기반 출력 스트림에 연결되어 버퍼 제공하는 보조 스트림
- 문자 기반 출력 스트림에 연결되어 버퍼 제공하는 보조 스트림

- 프로그램에서 전송한 데이터를 내부 버퍼에 쌓아두었다가 버퍼가 꽉 차면 한꺼번에 보냄
- 생성자의 매개값으로 준출력 스트림과 연결되어 추가 내부 버퍼 제공

성능 향상 보조 스트림



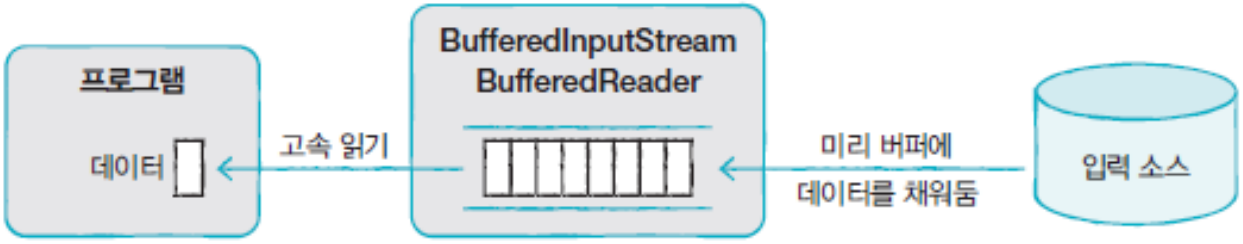

```
BufferedOutputStream bos = new BufferedOutputStream(바이트 기반 출력 스트림);
BufferedWriter bw = new BufferedWriter(문자 기반 출력 스트림);
```

성능 향상 보조 스트림

❖ BufferedInputStream과 BufferedReader

- 바이트 기반 입력 스트림에 연결되어 버퍼 제공하는 보조 스트림
- 문자 기반 입력 스트림에 연결되어 버퍼를 제공하는 보조 스트림

- 입력 소스로부터 내부 버퍼 크기만큼 데이터를 미리 읽고 버퍼에 저장해둠
- 프로그램이 외부 소스로부터 직접 읽는 것이 아닌 버퍼로부터 읽음으로써 읽기 성능 향상



- 생성자의 매개값으로 준입력 스트림과 연결되어 추가 내부 버퍼 제공

```
BufferedInputStream bis = new BufferedInputStream(바이트 기반 입력 스트림);
BufferedReader br = new BufferedReader(문자 기반 입력 스트림);
```

성능 향상 보조 스트림

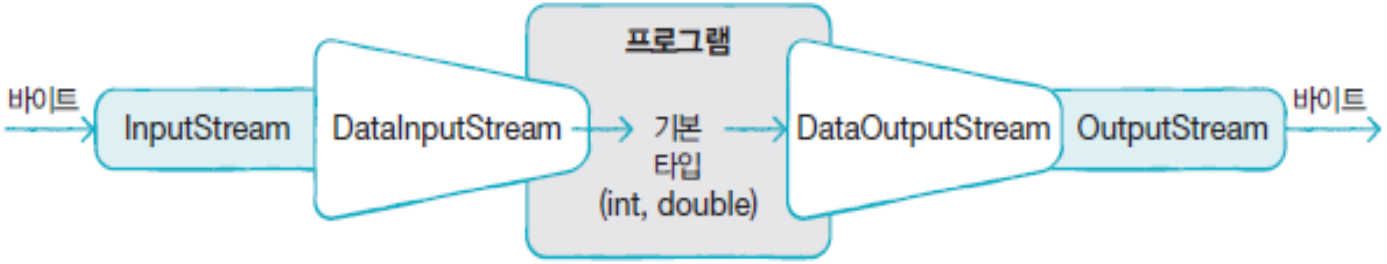
- BufferReader는 라인 단위로 문자열 읽는 readLine() 메소드 제공
  - [Enter]키 이전의 모든 문자열 읽고 리턴
  - 키보드에서 입력한 내용 및 파일 내용을 라인 단위로 읽을 수 있음
  - 예시 – 라인 단위로 구분된 문자열

```
01    C 언어
02    Java 언어
03    Python 언어
```

  - 라인 단위로 문자열 얻으려면 BufferedReader 보조 스트림 생성하고 readline()으로 반복해서 읽음

기본 타입 입출력 보조 스트림

❖ DataInputStream과 DataOutputStream 보조 스트림 연결하면 기본 타입인 boolean, char, short, int, long, float, double 입출력할 수 있음



- 객체 생성 코드

```
DataInputStream dis = new DataInputStream(바이트 기반 입력 스트림);
DataOutputStream dos = new DataOutputStream(바이트 기반 출력 스트림);
```

기본 타입 입출력 보조 스트림

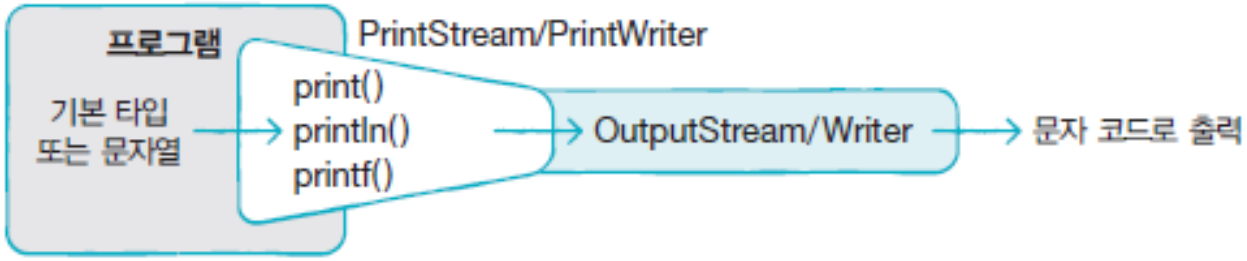
DataInputStream		DataOutputStream	
boolean	readBoolean()	void	writeBoolean(boolean v)
byte	readByte()	void	writeByte(int v)
char	readChar()	void	writeChar(int v)
double	readDouble()	void	writeDouble(double v)
float	readFloat()	void	writeFloat(float v)
int	readInt()	void	writeInt(int v)
long	readLong()	void	writeLong(long v)
short	readShort()	void	writeShort(int v)
String	readUTF()	void	writeUTF(String str)

- 데이터 타입의 크기가 모두 다르므로 DataOutputStream으로 출력한 데이터를 다시 DataInputStream으로 읽어올 때는 출력한 순서와 동일한 순서로 읽어야 함

프린터 보조 스트림

❖ PrintStream과 PrintWriter는 프린터와 유사하게 출력하는 print(), println() 메소드 가진 보조 스트림

- 각기 바이트 기반 출력 스트림 및 문자 기반 출력 스트림과 연결




```
PrintStream ps = new PrintStream(바이트 기반 출력 스트림);
PrintWriter pw = new PrintWriter(문자 기반 출력 스트림);
```

- println() 메소드는 print()와 달리 출력할 데이터 끝에 개행 문자 \n 추가하여 줄바꿈 일어남

프린터 보조 스트림

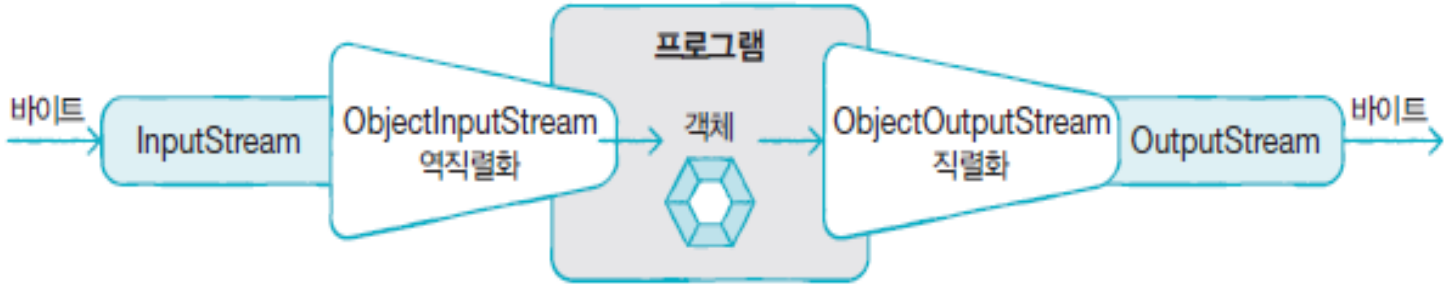
- 출력할 데이터 타입에 따른 오버로딩

PrintStream / PrintWriter			
void	print(boolean b)	void	println(boolean b)
void	print(char c)	void	println(char c)
void	print(double d)	void	println(double d)
void	print(float f)	void	println(float f)
void	print(int i)	void	println(int i)
void	print(long l)	void	println(long l)
void	print(Object obj)	void	println(Object obj)
void	print(String s)	void	println(String s)
		void	println( )

객체 입출력 보조 스트림

❖ ObjectOutputStream과 ObjectInputStream 보조 스트림 연결하면 메모리에 생성된 객체를 파일 또는 네트워크로 출력

- 각기 객체 직렬화 및 역직렬화 역할



- 연결할 바이트 기반 입출력 스트림을 생성자 매개값으로 받음

```
ObjectInputStream ois = new ObjectInputStream(바이트 기반 입력 스트림);
ObjectOutputStream oos = new ObjectOutputStream(바이트 기반 출력 스트림);
```

객체 입출력 보조 스트림

- ObjectOutputStream의 writeObject() 메소드는 객체 직렬화하여 출력 스트림으로 보냄

```
oos.writeObject(객체);
```

- ObjectInputStream의 readObject() 메소드는 입력 스트림에서 읽은 바이트를 역직렬화하여 객체로 다시 복원해 리턴함
  - 원래 타입으로 강제변환 필요

```
객체타입 변수 = (객체타입) ois.readObject();
```

- 자바는 java.io.Serializable 인터페이스 구현한 객체만 직렬화함
  - Serializable 인터페이스는 메소드 선언 없으므로, 네트워크로 전송하려는 경우 클래스 선언 시 implements Serializable 추가해야 함

```
public class XXX implements Serializable { … }
```

키워드로 끝내는 핵심 포인트

- 보조 스트림 : 다른 스트림과 연결되어 여러 편리한 기능 제공해주는 스트림. 자체적으로 입출력 수행할 수 없기 때문에 입출력 소스와 바로 연결되는 InputStream, Outputstream, Reader, Writer 등에 연결해서 입출력 수행. 문자 변환, 입출력 성능 향상, 기본 타입 입출력 등 기능 제공

- 문자 변환 : 소스 스트림이 바이트 기반 스트림이면서 입출력 데이터가 문자라면 Reader와 Writer로 변환해서 사용하는 것이 편리함. OutputStreamWriter는 Writer로 변환하는 보조 스트림이며, InputStreamReader는 Reader로 변환하는 보조 스트림

- 성능 향상 : 메모리 버퍼 추가로 제공하여 프로그램의 실행 성능을 향상시키는 것들이 있음. 바이트 기반 스트림에서는 BufferedInputStream, BufferedOutputStream, 문자 기반 스트림에서는 BufferedReader, BufferedWriter가 있음

- 기본 타입 입출력 : DataInputStream과 DataOutputStream 보조 스트림 연결하면 기본 타입인 boolean, char, short, int, long, float, double 입출력 가능

- 개행 출력 : PrintStream/PrintWriter의 println() 메소드는 출력할 데이터 끝에 개행 문자인 \n 추가하여 출력 시 콘솔이나 파일에서 줄 바꿈 일어남

14-3. 입출력 관련 API

혼자 공부하는 자바 (신용권 저)

목차

시작하기 전에

[핵심 키워드] : System.in, System.out, Scanner, File

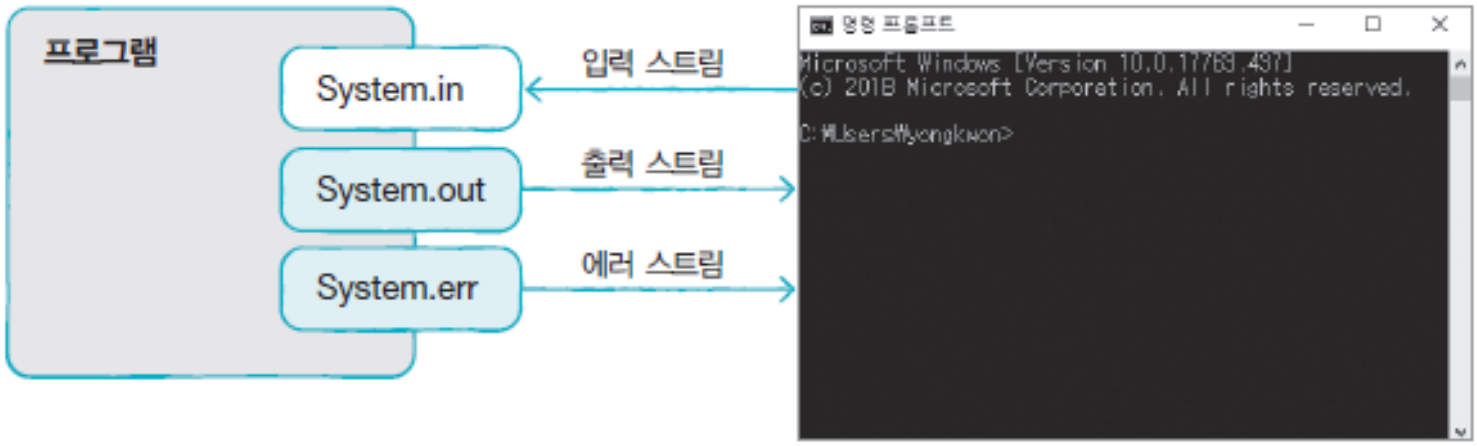
[핵심 포인트]

자바 표준 API에서는 스트림을 이용해 다양한 기능을 제공한다. 대표적으로 콘솔에서 입출력에 사용되는 System.in은 InputStream 타입이고, System.out은 PrintStream 타입이다. 콘솔 입출력 시 보조 스트림을 사용하는 방법과 파일 입출력 시 부가적인 정보를 제공하는 File 클래스에 대해 알아본다.

시작하기 전에

❖ 콘솔 (Console)

- 시스템 사용하기 위해 키보드로 입력받고 모니터로 출력하는 소프트웨어
- 자바는 콘솔로부터 데이터 입력받을 때 System.in 사용하고, 콘솔에 데이터 출력할 때 System.out 사용
- 에러 출력 시에는 System.err 사용



System.in 필드

- 자바는 콘솔에서 키보드 데이터 입력받을 수 있도록 System 클래스의 in 정적 필드 제공
  - System.in은 InputStream 타입의 필드이므로 다음과 같이 InputStream 변수로 참조 가능

```
InputStream is = System.in;
```

  - 키보드로부터 어떤 키가 입력되었는지 확인하려면 InputStream의 read() 메소드로 1byte 읽음

```
int keyCode = is.read();
```

  - [Enter]키 입력 후 라인 단위로 전체 문자열 읽는 방식으로 변경할 경우

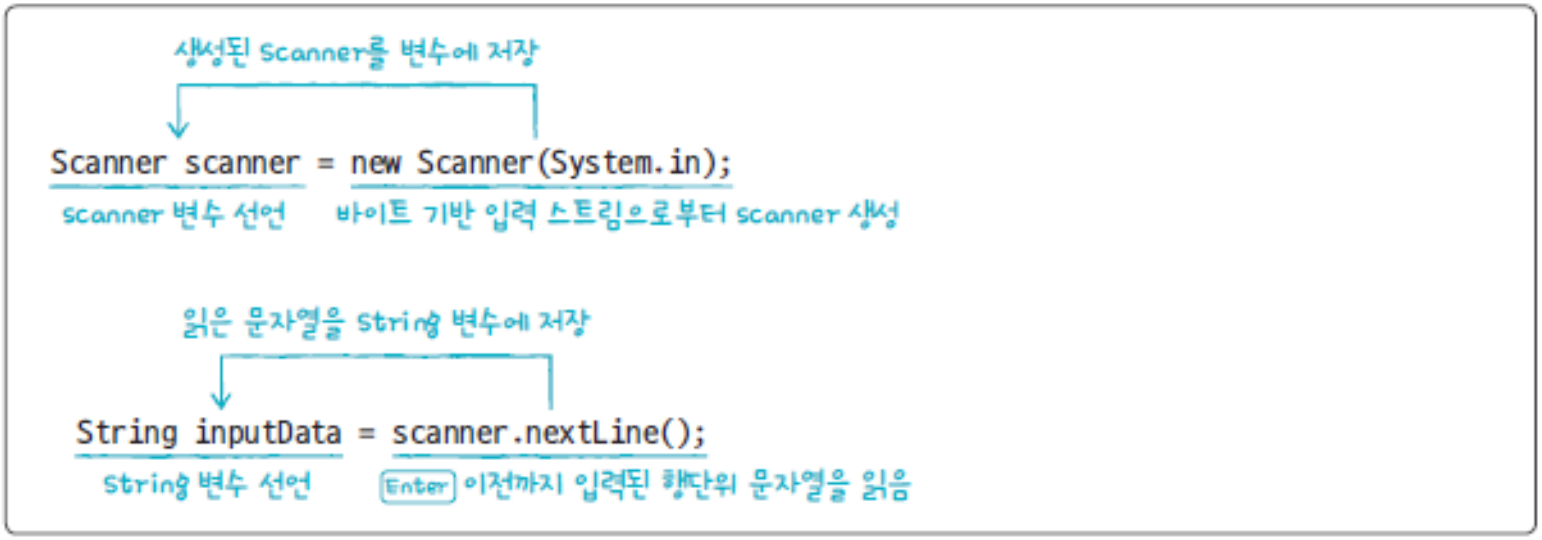
```
InputStream is = System.in;
Reader reader = new InputStreamReader(is);
BufferedReader br = new BufferedReader(reader);
```

    - InputStreamReader 보조스트림 연결하여 Reader로 변환 후 라인 단위로 읽기 위해 BufferedReader 보조 스트림 추가 연결

```
String lineStr = br.readLine();
```

System.out 필드

- 콘솔에서 모니터로 데이터 출력하기 위해 System 클래스의 out 정적 필드 사용
  - PrintStream이 제공하는 print(), println(), printif() 등 메소드 사용



File 클래스

❖ java.io 패키지에서 제공하는 File 클래스는 파일 및 폴더 정보 제공 역할 함

```
File file = new File("C:/Temp/file.txt");
File file = new File("C:\\Temp\\file.txt");
```

- 경로 구분자는 운영체제마다 조금씩 다름
  - 윈도우에서는 \ 및 / 모두 사용 가능
- 해당 경로에 실제로 파일이나 폴더 있는지 확인하려면 File 객체 생성후 exists() 메소드 호출

```
boolean isExist = file.exists();
```

- exists() 메소드의 리턴값이 false일 경우 다음 메소드로 파일 혹은 폴더 생성

리턴 타입	메소드	설명
boolean	createNewFile()	새로운 파일을 생성합니다.
booelan	mkdir()	새로운 폴더를 생성합니다.
boolean	mkdirs()	경로상에 없는 모든 폴더를 생성합니다.

File 클래스

- exists() 메소드의 리턴값이 true라면 다음 메소드 사용 가능

리턴 타입	메소드	설명
boolean	delete()	파일 또는 폴더를 삭제합니다.
boolean	canExecute()	실행할 수 있는 파일인지 여부를 확인합니다.
boolean	canRead()	읽을 수 있는 파일인지 여부를 확인합니다.
boolean	canWrite()	수정 및 저장할 수 있는 파일인지 여부를 확인합니다.
String	getName()	파일의 이름을 리턴합니다.
String	getParent()	부모 폴더를 리턴합니다.
File	getParentFile()	부모 폴더를 File 객체로 생성 후 리턴합니다.
String	getPath()	전체 경로를 리턴합니다.
boolean	isDirectory()	폴더인지 여부를 확인합니다.
boolean	isFile()	파일인지 여부를 확인합니다.
boolean	isHidden()	숨김 파일인지 여부를 확인합니다.
long	lastModified()	마지막 수정 날짜 및 시간을 리턴합니다.
long	length()	파일의 크기를 리턴합니다.
String[]	list()	폴더에 포함된 파일 및 서브 폴더 목록 전부를 String 배열로 리턴합니다.
String[]	list(FilenameFilter filter)	폴더에 포함된 파일 및 서브 폴더 목록 중에 FilenameFilter에 맞는 것만 String 배열로 리턴합니다.
File[]	listFiles()	폴더에 포함된 파일 및 서브 폴더 목록 전부를 File 배열로 리턴합니다.
File[]	listFiles(FilenameFilter filter)	폴더에 포함된 파일 및 서브 폴더 목록 중에 FilenameFilter에 맞는 것만 File 배열로 리턴합니다.

File 클래스

- File 객체는 파일 입출력 스트림 객체 생성 시 경로 정보 제공 목적으로도 사용됨
  - 스트림 생성자에 문자열 경로 대신 File 객체를 대입

```
//첫 번째 방법
FileInputStream fis = new FileInputStream("C:/Temp/image.gif");

//두 번째 방법
File file = new File("C:/Temp/image.gif");
FileInputStream fis = new FileInputStream(file);
```

키워드로 끝내는 핵심 포인트

- System.in : 자바는 콘솔에서 키보드의 데이터 입력받을 수 있도록 System 클래스의 in 정적 필드 제공함. 주로 InputStreamReader 보조 스트림과 BufferedReader 보조 스트림을 연결해서 사용하거나 Scanner를 이용해서 입력된 문자열 읽음

- System.out : 콘솔에서 키보드로 입력된 데이터를 System.in로 읽었다면 반대로 콘솔에서 모니터로 데이터 출력하기 위해서는 System 클래스의 out 정적 필드 사용. PrintStream이 제공하는 print(), println(), printf()와 같은 메소드 이용하여 모니터로 출력 가능.

- Scanner : Scanner 클래스는 입출력 스트림도, 보조 스트림도 아님. Scanner는 문자 파일이나 바이트 기반 입출력 스트림에서 라인 단위 문자열을 쉽게 읽도록 하기 위해 java.util 패키지에서 제공하는 클래스.

- File : java.io 패키지에서 제공하는 File 클래스는 파일 및 폴더 정보 제공하는 역할 함.

Thank You!

혼자 공부하는 자바 (신용권 저)

한빛미디어 Hanbit Media, Inc.