

{ JAVA }

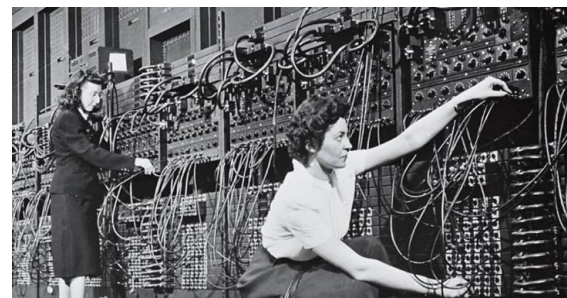
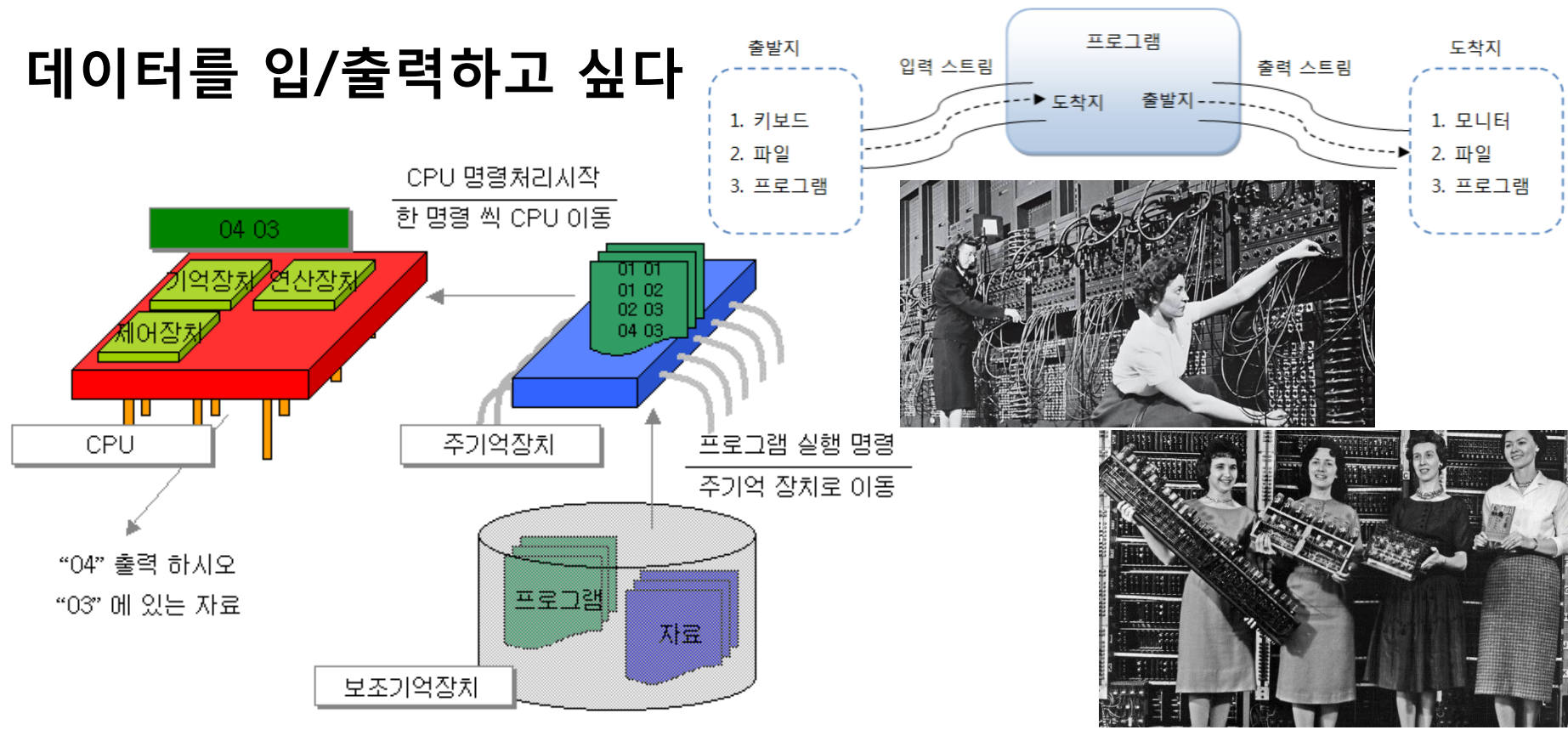
선린인터넷
웹 운영 과

I/O 입출력

java.io 패키지 2

Byte Stream

데이터를 입/출력하고 싶다



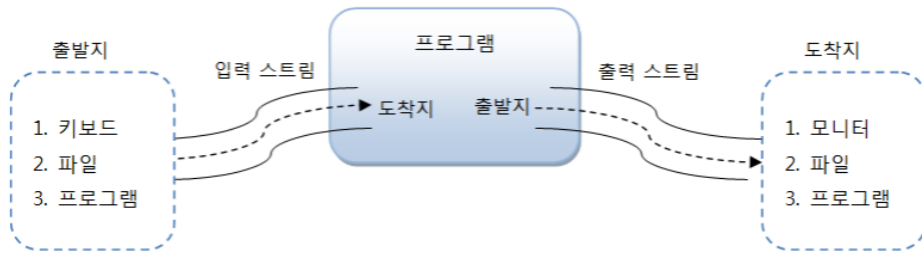
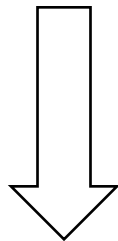
java 

스트림은 데이터들이 들어오고 나가는 통로이며, 단방향이다.

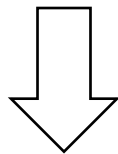
Stream : 자바 프로그램에서 데이터를 입출력하기 위해서는 스트림을 이용한다.

- 스트림 = 자바 프로그램의 입출력을 도와주는 매개체
- JAVA → 각 장치마다 데이터의 입출력을 담당하는 스트림 class 제공
- 개발자 → 각각 입출력 장치와 관련하여 존재하는 스트림에 대해서만 작업
(장치와 스트림 간의 세부적인 작동원리는 몰라도 됨)

데이터를 입/출력하고 싶다 : 키보드, 모니터, 파일, 네트워크 등



스트림 생성 = 해당 클래스 타입의 객체를 생성 (단방향: FIFO)



스트림으로부터 데이터 입출력 = 객체가 제공하는 메소드 이용

❖ 스트림 종류

■ 입력 스트림

- 키보드, 마우스, 네트워크 등과 같은 입력장치로부터 입력된 데이터가 순서대로 프로그램으로 흘러가는 데이터의 흐름 또는 소프트웨어 장치

■ 출력 스트림

- 프로그램에서 출력된 데이터가 프린터, 콘솔, 네트워크 등과 같은 출력장치로 순서대로 전송되도록 보장하는 데이터의 흐름 또는 소프트웨어 장치

❖ 스트림 특징

- 스트림을 통해 흘러가는 데이터의 기본 단위 : **Byte**
 - 바이트 스트림과 문자 스트림으로 나눌 수 있다.
- **단방향**이다.
- **FIFO구조**이다 : 먼저 들어온 데이터가 먼저 처리된다.
- 같은 용도의 스트림 끼리는 **서로 연결될 수 있다**.

Node계열 스트림의 특징	Filter계열 스트림의 특징
* 단순한 입출력 작업만 담당한다	* 입력 데이터를 가공 하거나 출력한 데이터를 가공 한다
* 장치에 직접적으로 연결된다	* 효율적인 입출력 이 가능하다

❖ 자바 I/O 준비학습

- Java 입출력 방식은 크게 ① **바이트(Byte)** 단위 ② **문자(Char)** 단위로 구분된다.
※ 단방향
- Java I/O는 **데코레이터 패턴**으로 구성된다. (주인공**Node** 과 장식**Filter**의 구분)
- **표준 입/출력 : 미리 생성**된 스트림 = 사용자가 직접 생성할 필요 없음
 - 교과서125 쪽 참고 -
 - 표준입력 : **System.in** (InputStream 타입) - 키보드로 데이터를 입력하는 것
 - 표준출력 : **System.out** (PrintStream 타입) - 화면(모니터)으로 메시지를 출력하는 것
 - 표준에러 : **System.err** (PrintStream 타입) - 화면(모니터)으로 오류를 출력하는 것

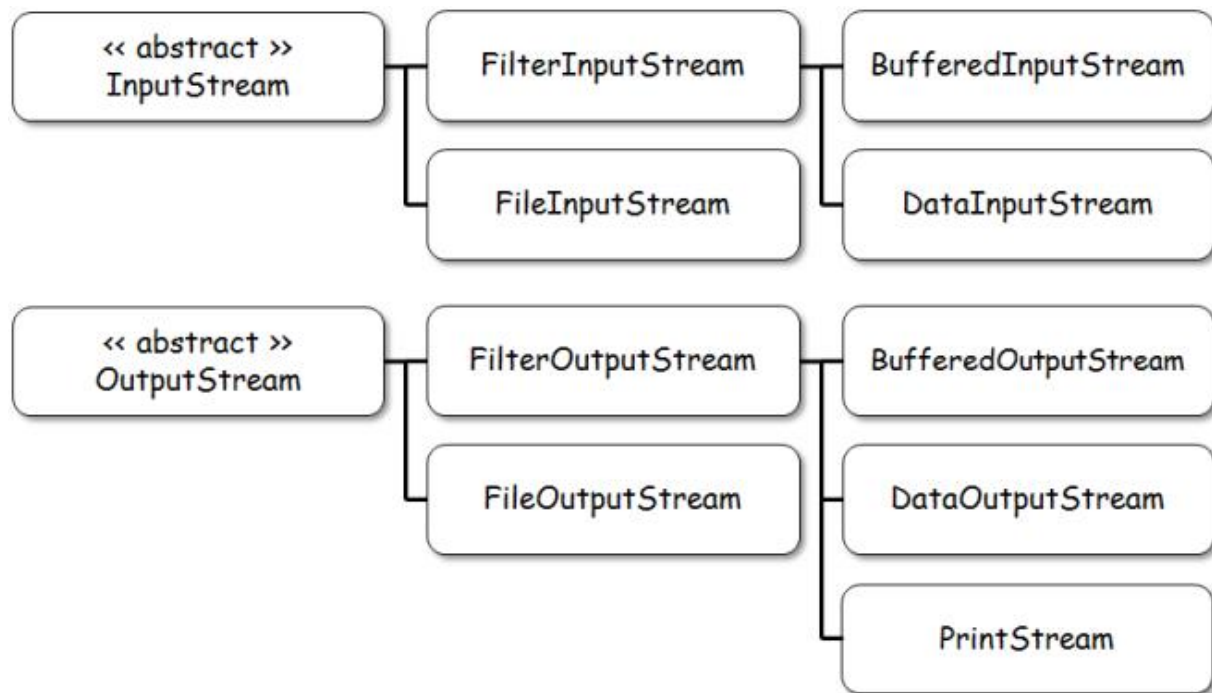
교과서 129쪽

바이트 스트림

Byte Stream

❖ 바이트(Byte) 스트림 : jdk1.0

- 8bit의 1바이트 단위로 읽고 쓰기 위한 스트림 : **read(), write() 메소드**



❖ 바이트(Byte) 스트림 : 주요 클래스

입력 클래스	특징	출력 클래스	특징
InputStream	기본 입력 (추상클래스)	OutputStream	기본 출력 (추상클래스)
FileInputStream	파일 입력	FileOutputStream	파일 출력
BufferedInputStream	버퍼 이용한 입력	BufferedOutputStream	버퍼 이용한 출력
DataInputStream	기본형 데이터 입력	DataOutputStream	기본형 데이터 출력
		PrintStream	표준출력 장치로 출력

❖ InputStream 주요 메소드

메 소 드	설 명
abstract int read ()	* 스트림으로부터 1Byte를 읽는 메소드 * 스트림의 끝은 -1 을 반환한다
int read (byte[] b)	* 바이트 배열 에 데이터를 읽어 들이는 메소드 * 한 번에 여러 데이터(byte 배열)를 읽어 들인다
void close ()	* 해당 스트림과 관련된 모든 자원을 해제하는 메소드

❖ OutputStream 주요 메소드

메 소 드	설 명
abstract void write (int b)	* 인자로 전달된 데이터를 출력 스트림으로 출력한다
void write (byte[] b)	* 인자로 전달된 바이트 배열 을 출력한다
void write (byte[] b, int off , int len)	* b배열에서 off의 위치로부터 len크기의 byte데이터를 출력스트림으로 출력한다
void flush ()	* 출력되어야 할 데이터를 버퍼에서 비운 후 출력한다
void close ()	* 출력 스트림을 닫고, 관련된 자원을 해제한다

예제1 : 키보드로부터 문자를 입력 받아 출력하는 프로그램 (입력끝: **Ctrl+z**)

```
1 import java.io.*;
2 public class ByteExam1 {
3     public static void test(InputStream is) throws IOException {
4         int i;
5         while (true) {
6             i = is.read();
7             if (i == -1)
8                 break;
9             System.out.print((char)i);
10        }
11    }
12    public static void main(String[] args) throws IOException {
13        test(System.in);
14    }
15 }
```

한글처리가 불가능하다

Console

<terminated> ByteEx:

Test

Test

홍길동

??±æ?¿

<

예제2 : 키보드로부터 문자를 입력 받아 출력하는 프로그램

```
1 import java.io.IOException;
2 public class ByteExam2 {
3     public static void main(String[] args) throws IOException{
4         byte[] b= new byte[1024];
5         int len = 0;
6         len = System.in.read(b);
7         System.out.write(b, 0 ,len);
8     }
9 }
```

 Console

<terminated> B:

홍길동

홍길동

실습문제 : 예제2를 참고해서 한글처리가 가능하도록 수정하자.

- 아래 코드는 예제1 프로그램 입니다. 적절하게 수정해봅시다. - (입력끝: **Ctrl+z**)

```
1 import java.io.*;
2 public class ByteExam1 {
3     public static void test(InputStream is) throws IOException {
4         int i;
5         while (true) {
6             i = is.read();
7             if (i == -1)
8                 break;
9             System.out.print((char)i);
10        }
11    }
12    public static void main(String[] args) throws IOException {
13        test(System.in);
14    }
15 }
```

참고하기 : 다음 프로그램을 실행하면 출력결과가 어떻게 될까요?

```
1 import java.io.*;
2 public class ByteExam3 {
3     public static void main(String[] args) throws IOException{
4         OutputStream os = System.out;
5         os.write(72); os.write(101); os.write(108);
6         os.write(108); os.write(111); os.write(0);
7
8         os.write(87); os.write(111); os.write(114);
9         os.write(108); os.write(100); os.write(33);
10
11         // os.flush();
12         os.close();
13     }
14 }
```



I/O Stream

참고하기 : 다음 프로그램을 실행하면 출력결과가 어떻게 될까요? - **확인해봅시다**

Dec	Hex	Sym	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	NUL	32	20		64	40	@	96	60	`
1	1	SOH	33	21	!	65	41	A	97	61	a
2	2	STX	34	22	"	66	42	B	98	62	b
3	3	ETX	35	23	#	67	43	C	99	63	c
4	4	EOT	36	24	\$	68	44	D	100	64	d
5	5	ENQ	37	25	%	69	45	E	101	65	e
6	6	ACK	38	26	&	70	46	F	102	66	f
7	7	BEL	39	27	'	71	47	G	103	67	g
8	8	BS	40	28	(72	48	H	104	68	h
9	9	TAB	41	29)	73	49	I	105	69	i
10	A	LF	42	2A	*	74	4A	J	106	6A	j
11	B	VT	43	2B	+	75	4B	K	107	6B	k
12	C	FF	44	2C	,	76	4C	L	108	6C	l
13	D	CR	45	2D	-	77	4D	M	109	6D	m
14	E	SO	46	2E	.	78	4E	N	110	6E	n
15	F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	
29	1D	GS	61	3D	=	93	5D]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	

```
{
main(String[] args) throws IOException{
    = System.out;
    os.write(101); os.write(108);
    os.write(111); os.write(0);

    os.write(111); os.write(114);
    os.write(100); os.write(33);
}
```

Console  <terminated> ByteExam3 [Hello World!

<https://www.ieee.li/computer/ascii.htm>

	000	001	002	003	004	005	006	007
0	0	@	P	`
1	!	1	A	Q
2	"	2	B	R
3	#	3	C	S
4	\$	4	D	T
5	%	5	E	U
6	&	6	F	V
7	'	7	G	W
8	(8	H	X
9)	9	I	Y
A	*	:	J	Z
B	+	;	K	[
C	<	L	\]
D	=	M]	m
E	>	N	^	n
F	/	O	o	

정리하기

바이트(Byte) 스트림 : 1byte 단위로 읽고 쓰기 위한 스트림 : **read(), write()**

