14-1. 입출력 스트림

혼자 공부하는 자바 (신용권 저)

목차

시작하기 전에

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

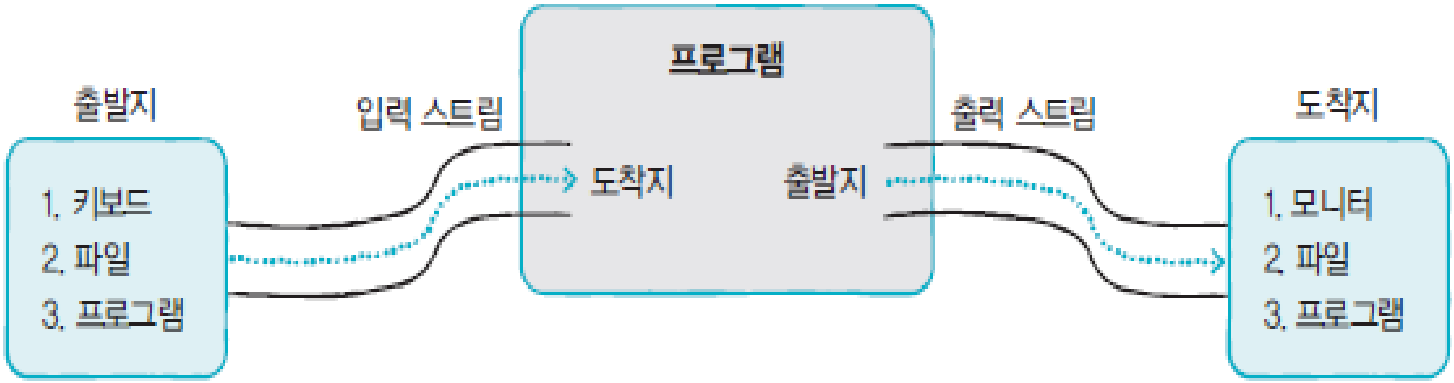
[핵심 포인트]

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

시작하기 전에

❖ 스트림 (Stream)

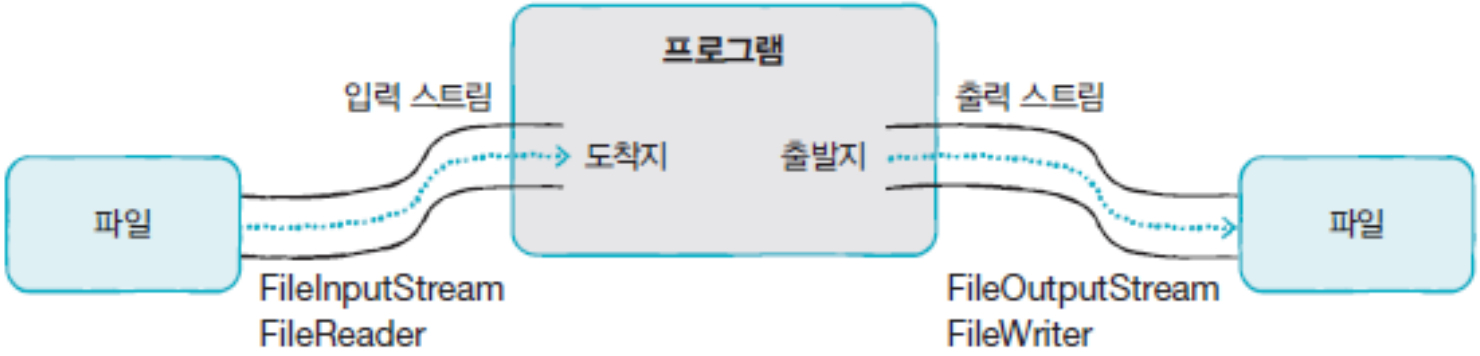
- 자바에서 데이터는 스트림을 통해 입출력됨
- 프로그램이 데이터의 출발지인지 도착지인지의 여부에 따라 사용하는 스트림의 종류가 결정



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

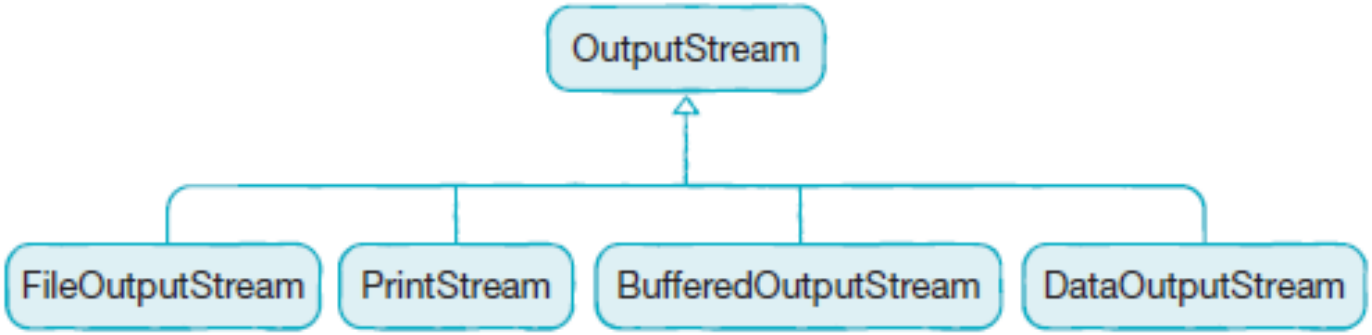
구분	바이트 기반 스트림		문자 기반 스트림	
	입력 스트림	출력 스트림	입력 스트림	출력 스트림
최상위 클래스	InputStream	OutputStream	Reader	Writer
하위 클래스 (예)	XXXInputStream (FileInputStream)	XXXOutputStream (FileOutputStream)	XXXReader (FileReader)	XXXWriter (FileWriter)



바이트 출력 스트림 : OutputStream

❖ OutputStream

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

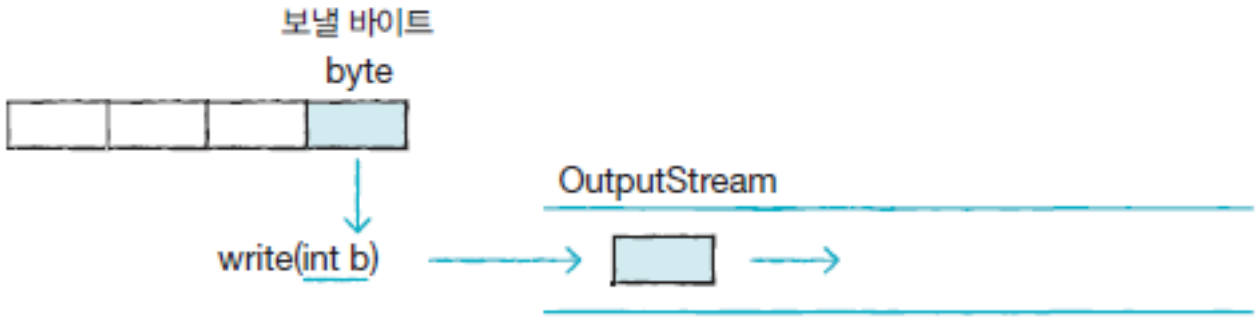


리턴 타입	메소드	설명
void	write(int b)	1byte를 출력합니다.
void	write(byte[] b)	매개값으로 주어진 배열 b의 모든 바이트를 출력합니다.
void	write(byte[] b, int off, int len)	매개값으로 주어진 배열 b[off]부터 len개까지의 바이트를 출력합니다.
void	flush()	출력 버퍼에 잔류하는 모든 바이트를 출력합니다.
void	close()	출력 스트림을 닫습니다.

바이트 출력 스트림 : OutputStream

❖ write(int b) 메소드

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



바이트 출력 스트림 : OutputStream

- 예시 – 1byte씩 출력하기

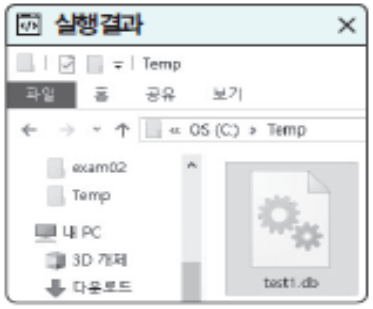
```
01  package sec01.exam01;
02
03  import java.io.FileOutputStream;
04  import java.io.OutputStream;
05
06  public class WriteExample {
07    public static void main(String[] args) throws Exception {
08      OutputStream os = new FileOutputStream("C:/Temp/test1.db");
09
10      byte a = 10;
11      byte b = 20;
12      byte c = 30;
13
14      os.write(a);
15      os.write(b);
16      os.write(c);
17
18      os.flush();
19      os.close();
20    }
21  }
```

데이터 도착지를 test1.db로 하는 바이트 기반 파일 출력 스트림을 생성

1byte씩 출력

출력 버퍼에 잔류하는 모든 바이트를 출력

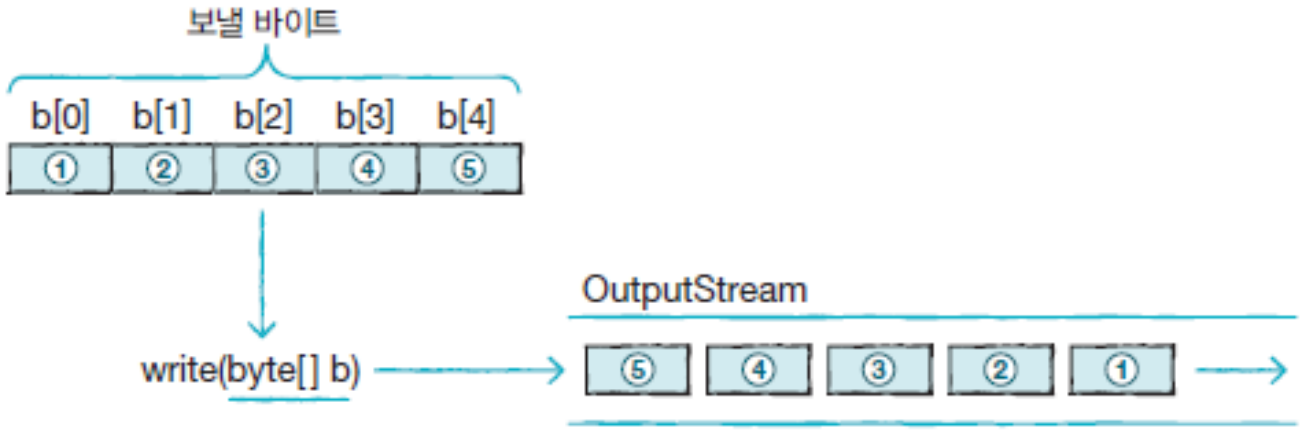
출력 스트림을 닫음



바이트 출력 스트림 : OutputStream

❖ write(byte[] b) 메소드

- 매개값으로 주어진 배열의 모든 바이트를 출력 스트림으로 보냄



바이트 출력 스트림 : OutputStream

- 예시 – 배열 전체를 출력하기

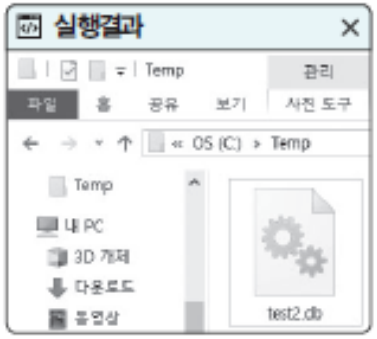
```
01  package sec01.exam02;
02
03  import java.io.FileOutputStream;
04  import java.io.OutputStream;
05
06  public class WriteExample {
07    public static void main(String[] args) throws Exception {
08      OutputStream os = new FileOutputStream("C:/Temp/test2.db");
09
10      byte[] array =  { 10, 20, 30 };
11
12      os.write(array);
13
14      os.flush();
15      os.close();
16    }
17  }
```

데이터 도착지를 test2.db로 하는 바이트 기반 파일 출력 스트림을 생성

배열의 모든 바이트를 출력

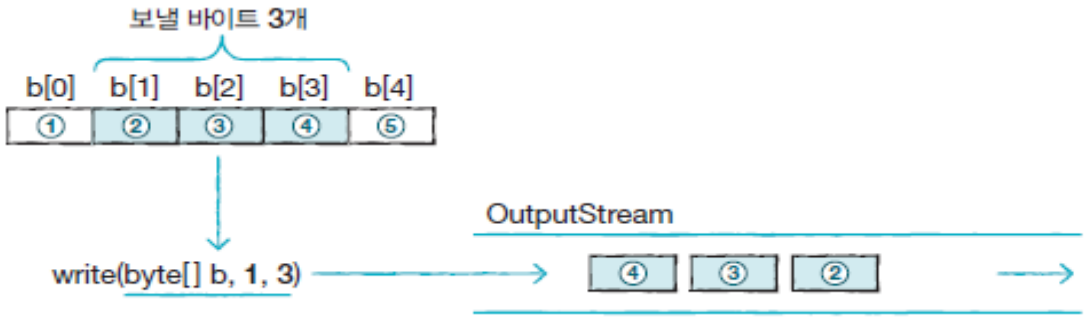
출력 버퍼에 잔류하는 모든 바이트를 출력

출력 스트림을 닫음



바이트 출력 스트림 : OutputStream

- 예시 – 배열 일부를 출력하기



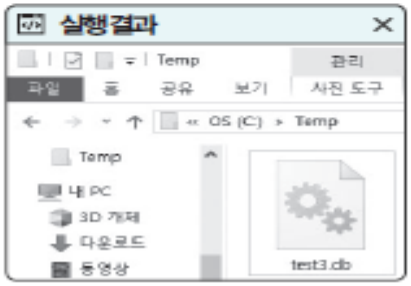

```
01  package sec01.exam03;
02
03  import java.io.FileOutputStream;
04  import java.io.OutputStream;
05
06  public class WriteExample {
07    public static void main(String[] args) throws Exception {
08      OutputStream os = new FileOutputStream("C:/Temp/test3.db");
09
10      byte[] array =  { 10, 20, 30, 40, 50 };
11
12      os.write(array, 1, 3);
13
14      os.flush();
15      os.close();
16    }
17  }
```

데이터 도착지를 test3.db로 하는 바이트 기반 파일 출력 스트림을 생성

배열의 1번 인덱스부터 3개를 출력

출력 버퍼에 잔류하는 모든 바이트를 출력

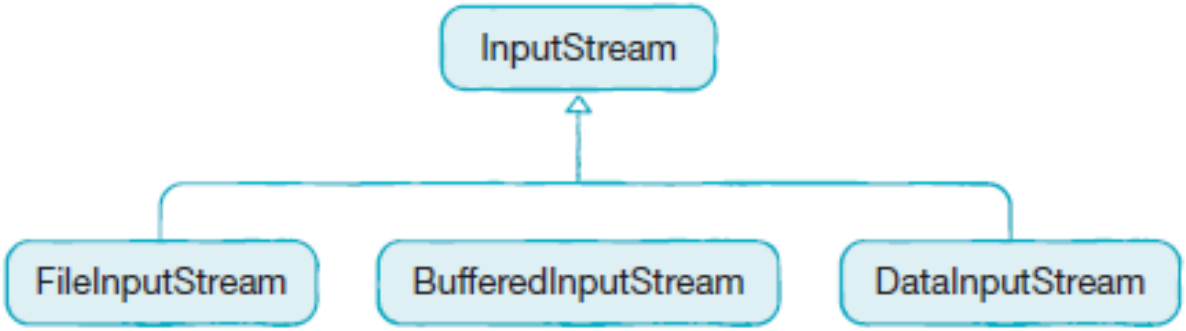
출력 스트림을 닫음



바이트 입력 스트림 : InputStream

❖ InputStream

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

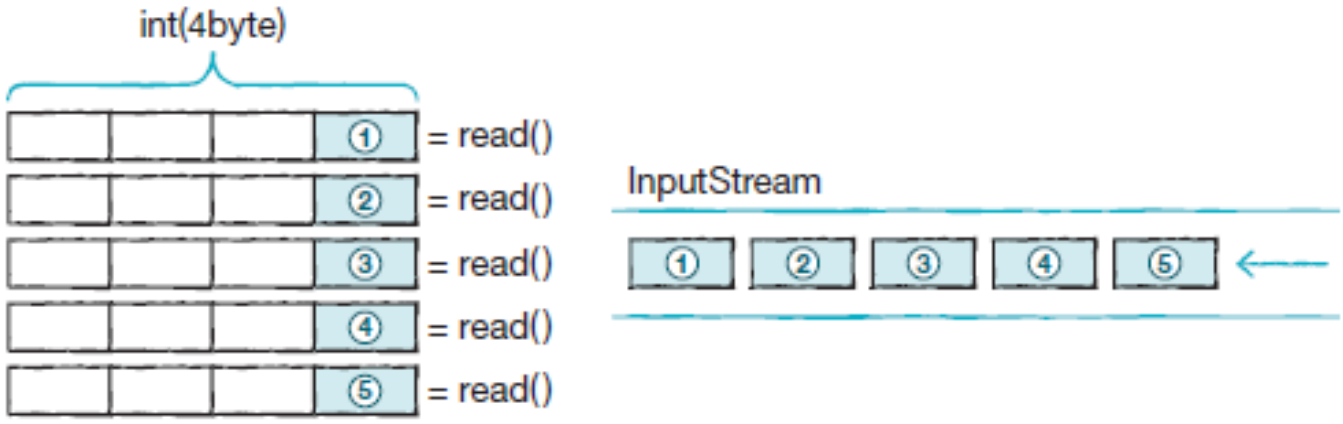


리턴 타입	메소드	설명
int	read()	1byte를 읽고 읽은 바이트를 리턴합니다.
int	read(byte[ ] b)	읽은 바이트를 매개값으로 주어진 배열에 저장하고 읽은 바이트 수를 리턴합니다.
int	read(byte[ ] b, int off, int len)	len개의 바이트를 읽고 매개값으로 주어진 배열에서 b[off]부터 len개까지 저장합니다. 그리고 읽은 바이트 수를 리턴합니다.
void	close()	입력 스트림을 닫습니다.

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

- 예시 – 1byte씩 읽기

```
01  package sec01.exam04;
02
03  import java.io.FileInputStream;
04  import java.io.InputStream;
05
06  public class ReadExample {
07    public static void main(String[] args) throws Exception {
08      InputStream is = new FileInputStream("C:/Temp/test1.db");
09      while(true) {
10        int data = is.read();
11        if(data == -1) break;
12        System.out.println(data);
13      }
14      is.close();
15    }
16  }
```

데이터 출발지를 test1.db로 하는 바이트 기반 파일 입력 스트림을 생성

1byte씩 읽기

파일 끝에 도달했을 경우

입력 스트림을 닫음

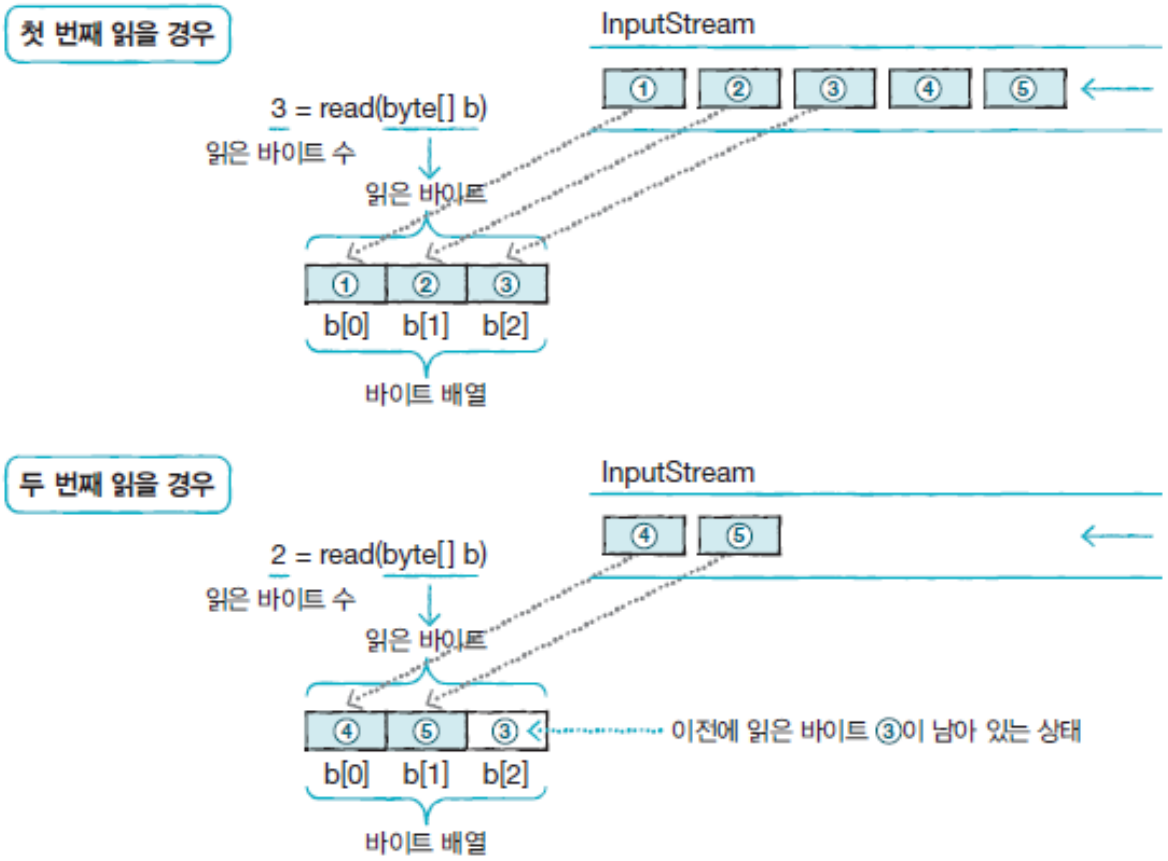
실행결과

```
10
20
30
```

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

- 예시 – 배열 길이만큼 읽기

```
01  package sec01.exam05;
02
03  import java.io.FileInputStream;
04  import java.io.InputStream;
05
06  public class ReadExample {
07    public static void main(String[] args) throws Exception {
08      InputStream is = new FileInputStream("C:/Temp/test2.db");
09
10      byte[] buffer = new byte[100];
11
12      while(true) {
13        int readByteNum = is.read(buffer);
14        if(readByteNum == -1) break;
15        for(int i=0; i<readByteNum; i++) {
16          System.out.println(buffer[i]);
17        }
18      }
19
20      is.close();
21    }
22  }
```

데이터 출발지를 test2.db로 하는 바이트 기반 파일 입력 스트림을 생성

길이 100인 배열 생성

배열 길이만큼 읽기

파일 끝에 도달했을 경우

읽은 바이트 수만큼 반복하면서 배열에 저장된 바이트를 출력

입력 스트림을 닫음

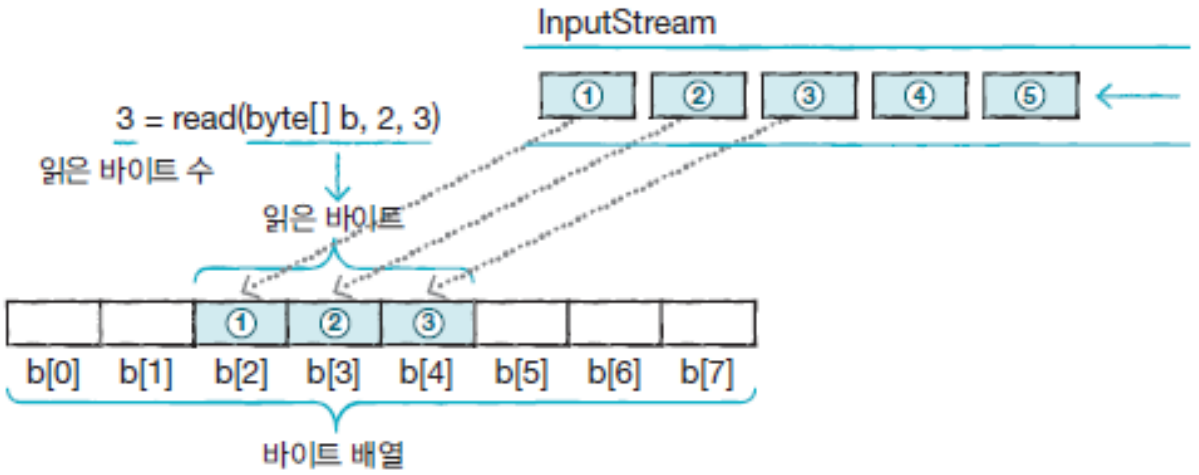
실행결과

```
10
20
30
```

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 len개 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);
```

바이트 입력 스트림 : InputStream

- 예시 – 지정한 길이만큼 읽기

```
01  package sec01.exam06;
02
03  import java.io.FileInputStream;
04  import java.io.InputStream;
05
06  public class ReadExample {
07    public static void main(String[] args) throws Exception {
08      InputStream is = new FileInputStream("C:/Temp/test3.db");
09
10      byte[] buffer = new byte[5];
11
12      int readByteNum = is.read(buffer, 2, 3);
13      if(readByteNum != -1) {
14        for(int i=0; i<buffer.length; i++) {
15          System.out.println(buffer[i]);
16        }
17      }
18
19      is.close();
20    }
21  }
```

데이터 출발지를 test3.db로 하는 바이트 기반 파일 입력 스트림을 생성

입력 스트림으로부터 3byte를 읽고 buffer[2] buffer[3], buffer[4]에 각각 저장

읽은 바이트가 있다면

배열 전체를 읽고 출력

입력 스트림을 닫음

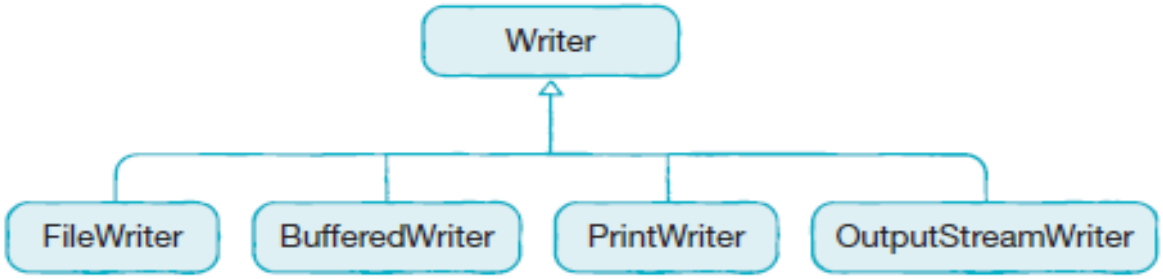
실행결과

```
0
0
20
30
40
```

문자 출력 스트림 : Writer

❖ Writer

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

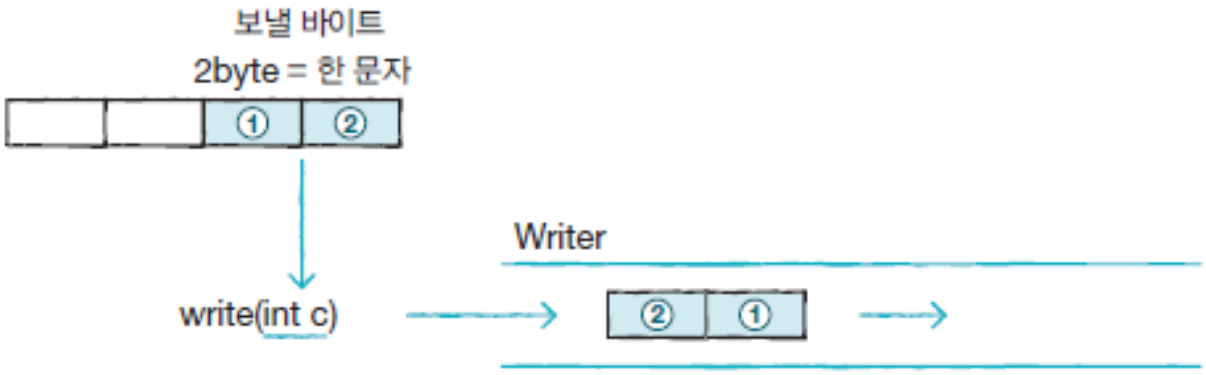


리턴 타입	메소드	설명
void	write(int c)	매개값으로 주어진 한 문자를 보냅니다.
void	write(char[ ] cbuf)	매개값으로 주어진 배열의 모든 문자를 보냅니다.
void	write(char[ ] cbuf, int off, int len)	매개값으로 주어진 배열에서 cbuf[off]부터 len개까지의 문자를 보냅니다.
void	write(String str)	매개값으로 주어진 문자열을 보냅니다.
void	write(String str, int off, int len)	매개값으로 주어진 문자열에서 off 순번부터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

- 예시 – 한 문자씩 출력하기

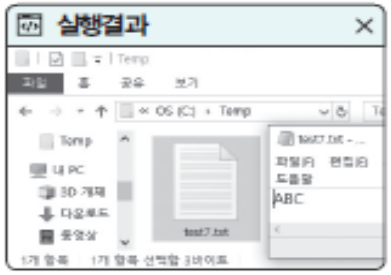
```
01  package sec01.exam07;
02
03  import java.io.FileWriter;
04  import java.io.Writer;
05
06  public class WriteExample {
07    public static void main(String[] args) throws Exception {
08      Writer writer = new FileWriter("C:/Temp/test7.txt");
09
10      char a = 'A';
11      char b = 'B';
12      char c = 'C';
13
14      writer.write(a);
15      writer.write(b);
16      writer.write(c);
17
18      writer.flush();
19      writer.close();
20    }
21  }
```

데이터 도착지를 test7.txt로 하는 문자 기반 파일 출력 스트림을 생성

한 문자씩 출력

출력 버퍼에 잔류하는 모든 문자를 출력

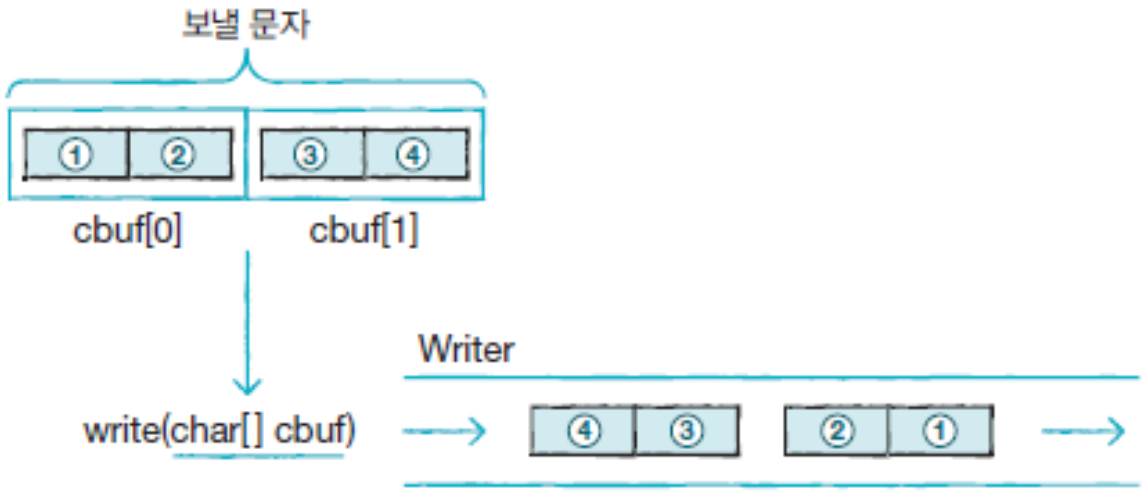
출력 스트림을 닫음



문자 출력 스트림 : Writer

❖ write(char[] cbuf) 메소드

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



문자 출력 스트림 : Writer

- 예시 – 배열 전체를 출력하기

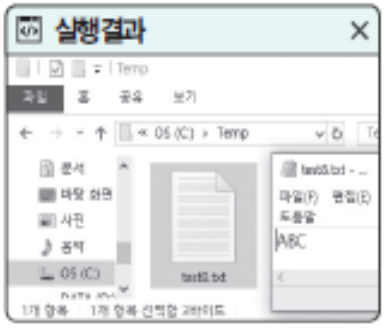
```
01  package sec01.exam08;
02
03  import java.io.FileWriter;
04  import java.io.Writer;
05
06  public class WriteExample {
07    public static void main(String[] args) throws Exception {
08      Writer writer = new FileWriter("C:/Temp/test8.txt");
09
10      char[] array = { 'A', 'B', 'C' };
11
12      writer.write(array);
13
14      writer.flush();
15      writer.close();
16    }
17  }
```

데이터 도착지를 test8.txt로 하는 문자 기반 파일 출력 스트림을 생성

배열의 모든 문자를 출력

출력 버퍼에 잔류하는 모든 문자를 출력

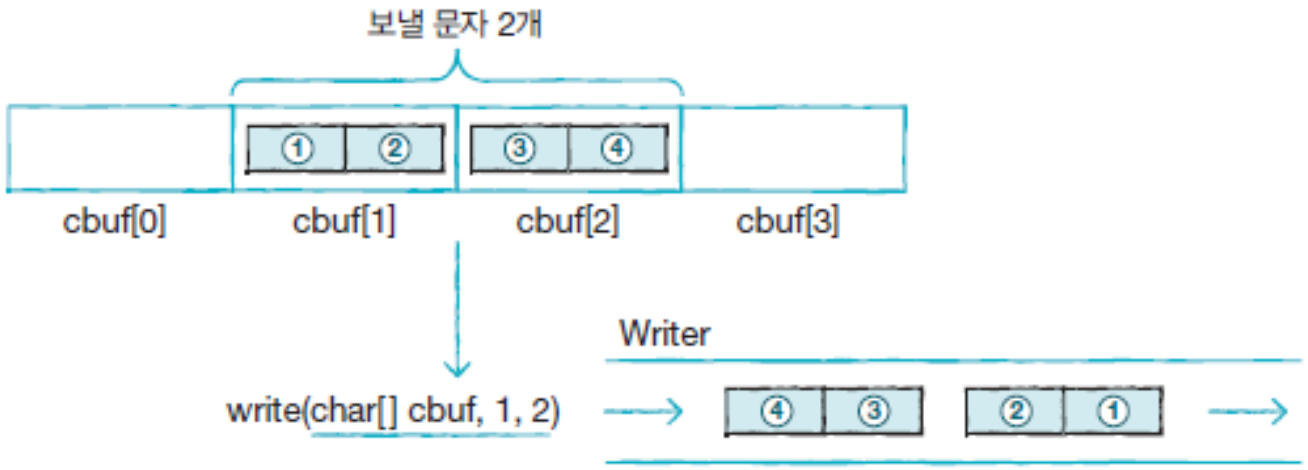
출력 스트림을 닫음



문자 출력 스트림 : Writer

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

- c[off]부터 len개의 문자를 출력 스트림으로 보냄



문자 출력 스트림 : Writer

- 예시 – 배열 일부를 출력하기

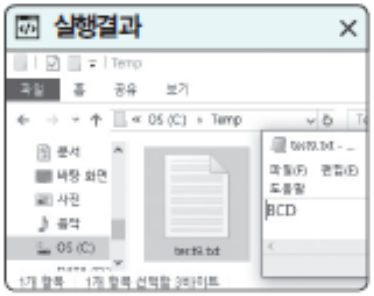
```
01  package sec01.exam09;
02
03  import java.io.FileWriter;
04  import java.io.Writer;
05
06  public class WriteExample {
07    public static void main(String[] args) throws Exception {
08      Writer writer = new FileWriter("C:/Temp/test9.txt");
09
10      char[] array = { 'A', 'B', 'C', 'D', 'E' };
11
12      writer.write(array, 1, 3);
13
14      writer.flush();
15      writer.close();
16    }
17  }
```

데이터 도착지를 test9.txt로 하는 문자 기반 파일 출력 스트림을 생성

배열의 1번 인덱스부터 3개를 출력

출력 버퍼에 잔류하는 모든 문자를 출력

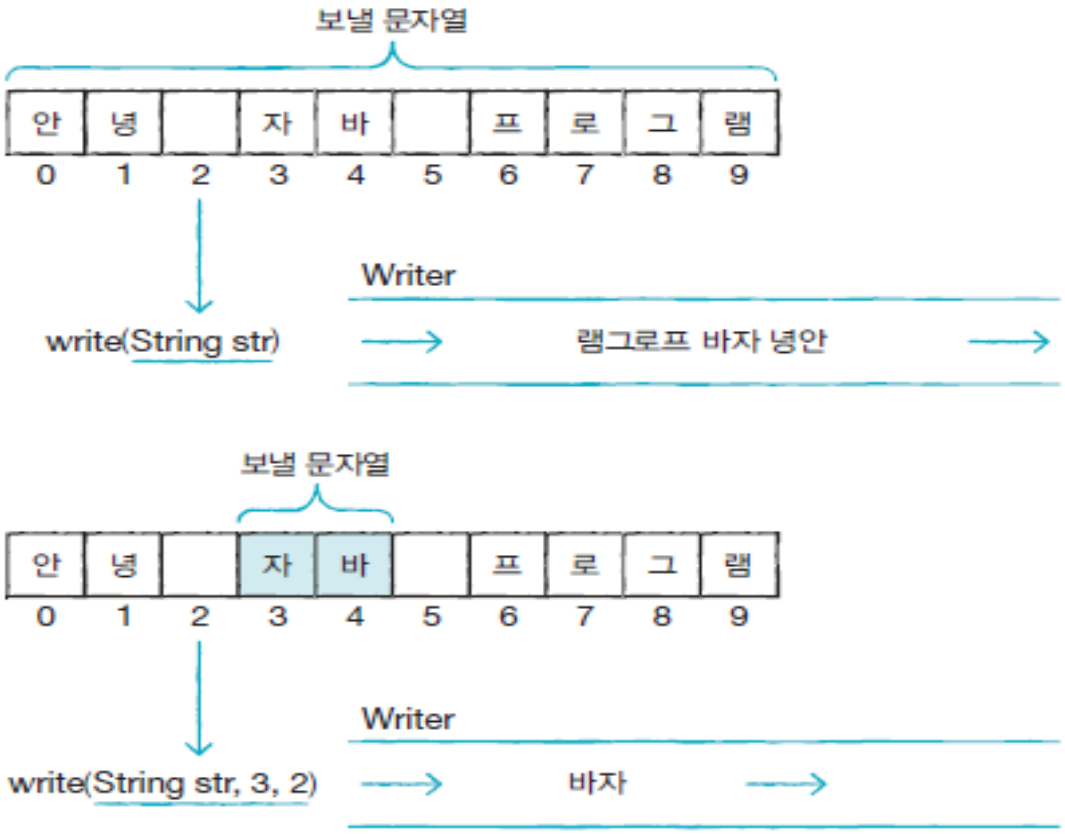
출력 스트림을 닫음



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



문자 출력 스트림 : Writer

- 예시 – 문자열 출력하기

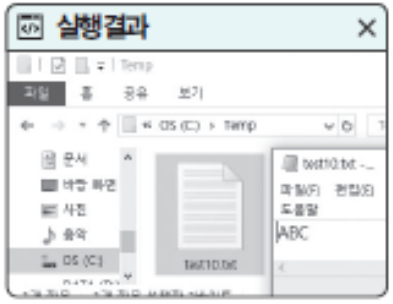
```
01  package sec01.exam10;
02
03  import java.io.FileWriter;
04  import java.io.Writer;
05
06  public class WriteExample {
07    public static void main(String[] args) throws Exception {
08      Writer writer = new FileWriter("C:/Temp/test10.txt");
09
10      String str = "ABC";
11
12      writer.write(str);
13
14      writer.flush();
15      writer.close();
16    }
17  }
```

데이터 도착지를 test10.txt로 하는 문자 기반 파일 출력 스트림을 생성

문자열 전체를 출력

출력 버퍼에 잔류하는 모든 문자열을 출력

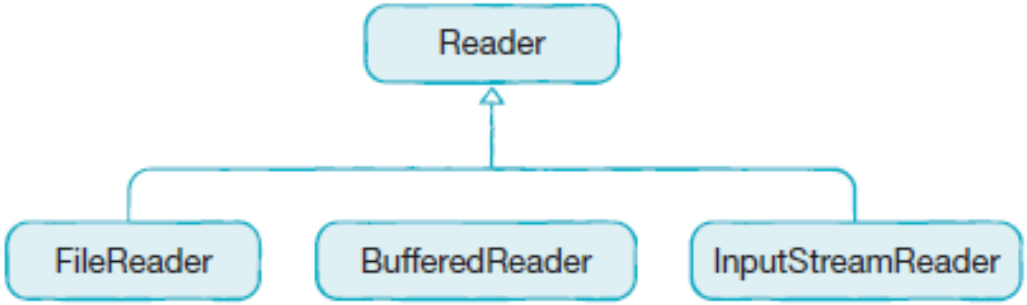
출력 스트림을 닫음



문자열 입력 스트림 : Reader

❖ Reader

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

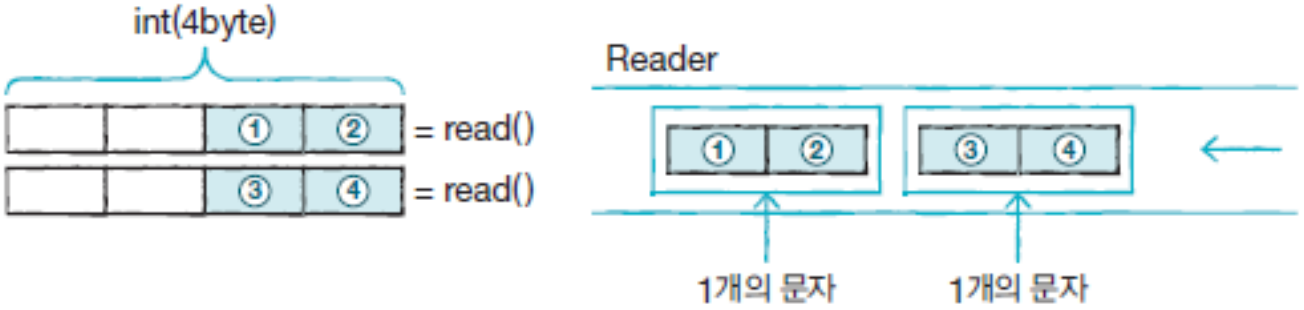


리턴 타입	메소드	설명
int	read()	1개의 문자를 읽고 리턴합니다.
int	read(char[] cbuf)	읽은 문자들을 매개값으로 주어진 문자 배열에 저장하고 읽은 문자 수를 리턴합니다.
int	read(char[] cbuf, int off, int len)	len개의 문자를 읽고 매개값으로 주어진 문자 배열에서 cbuf[off] 부터 len개까지 저장합니다. 그리고 읽은 문자 수를 리턴합니다.
void	close()	입력 스트림을 닫습니다.

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

- 예시 – 한 문자씩 읽기

```
01  package sec01.exam11;
02
03  import java.io.FileReader;
04  import java.io.Reader;
05
06  public class ReadExample {
07    public static void main(String[] args) throws Exception {
08      Reader reader = new FileReader("C:/Temp/test7.txt");
09      while(true) {
10        int data = reader.read();
11        if(data == -1) break;
12        System.out.println((char)data);
13      }
14      reader.close();
15    }
16  }
```

데이터 출발지를 test7.txt로 하는 문자 기반 파일 입력 스트림을 생성

한 문자씩 읽기

파일 끝에 도달했을 경우

입력 스트림을 닫음

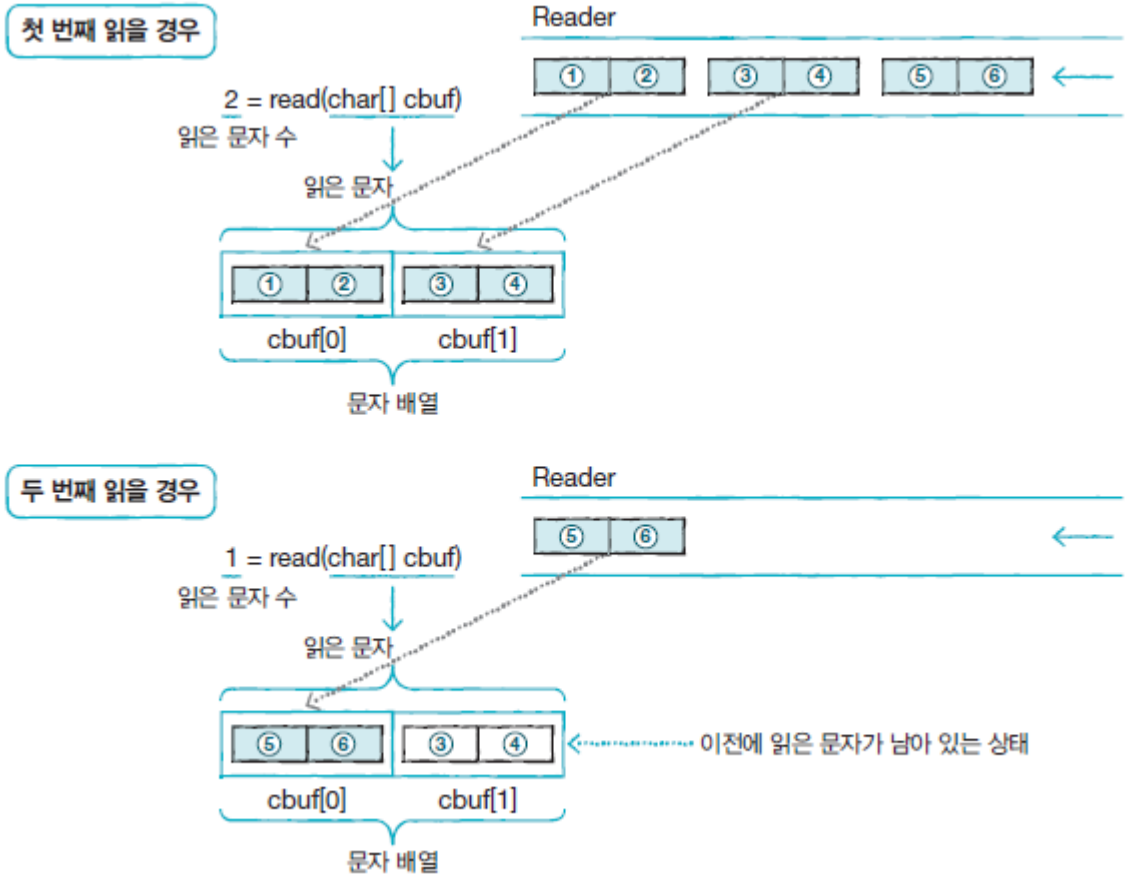
실행결과

```
A
B
C
```

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

- 예시 – 배열 길이만큼 읽기

```
01  package sec01.exam12;
02
03  import java.io.FileReader;
04  import java.io.Reader;
05
06  public class ReadExample {
07    public static void main(String[] args) throws Exception {
08      Reader reader = new FileReader("C:/Temp/test8.txt");
09
10      char[] buffer = new char[100];
11
12      while(true) {
13        int readCharNum = reader.read(buffer);
14        if(readCharNum == -1) break;
15        for(int i=0; i<readCharNum; i++) {
16          System.out.println(buffer[i]);
17        }
18      }
19
20      reader.close();
21    }
22  }
```

데이터 출발지를 test8.txt로 하는 문자 기반 파일 입력 스트림을 생성

길이 100인 배열 생성

배열 길이만큼 읽기

파일 끝에 도달했을 경우

읽은 문자 수만큼 반복하면서 배열에 저장된 문자를 출력

입력 스트림을 닫음

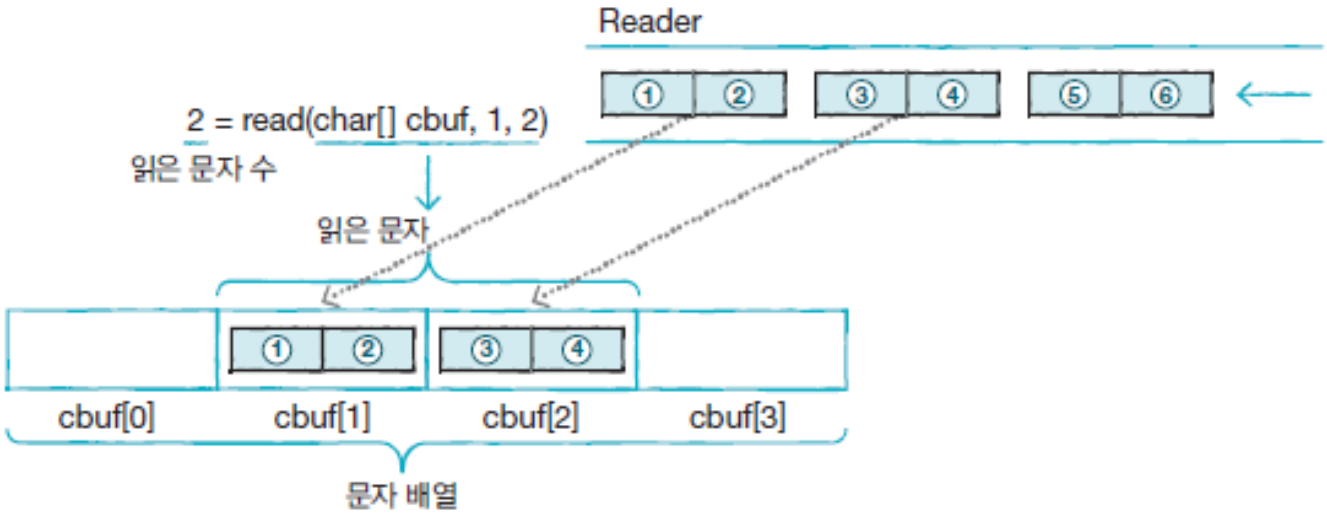
실행결과

```
A
B
C
```

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);
```

문자열 입력 스트림 : Reader

- 예시 – 지정한 길이만큼 읽기

```
01  package sec01.exam13;
02
03  import java.io.FileReader;
04  import java.io.Reader;
05
06  public class ReadExample {
07    public static void main(String[] args) throws Exception {
08      Reader reader = new FileReader("C:/Temp/test9.txt");
09
10      char[] buffer = new char[5];
11
12      int readCharNum = reader.read(buffer, 2, 3);
13      if(readCharNum != -1) {
14        for(int i=0; i<buffer.length; i++) {
15          System.out.println(buffer[i]);
16        }
17      }
18
19      reader.close();
20    }
21  }
```

데이터 출발지를 test9.txt로 하는 문자 기반 파일 입력 스트림을 생성

입력 스트림으로부터 3개의 문자를 읽고 buffer[2], buffer[3], buffer[4]에 각각 저장

읽은 문자가 있다면

배열 전체를 읽고 출력

입력 스트림을 닫음

실행결과

```
B
C
D
```

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.

확인문제

- 입출력 스트림에 대한 설명으로 맞는 것에 O, 틀린 것에 X 하세요
  - 하나의 스트림으로 입력과 출력이 동시에 가능하다 ( )
  - 프로그램을 기준으로 데이터가 들어오면 입력 스트림이다 ( )
  - 프로그램을 기준으로 데이터가 나가면 출력 스트림이다 ( )
  - 파일에 데이터를 저장하려면 출력 스트림을 사용해야 한다 ( )

- InputStream과 Reader에 대한 설명으로 맞는 것에 O, 틀린 것에 X 하세요
  - 이미지 데이터는 InputStream 또는 Reader로 모두 읽을 수 있다 ( )
  - Reader의 read() 메소드는 1문자를 읽고 리턴한다 ( )
  - InputStream의 read() 메소드는 1byte를 읽고 리턴한다 ( )
  - InputStream의 read(byte[] b)메소드는 읽은 바이트 수를 리턴한다 ( )

확인문제

- 출력 스트림에서 데이터를 출력한 후 flush() 메소드를 호출하는 이유가 무엇입니까?
  - 출력 스트림의 버퍼에 있는 데이터를 모두 출력하고 버퍼를 비운다
  - 출력 스트림을 메모리에서 제거한다
  - 출력 스트림의 버퍼에 있는 데이터를 모두 삭제한다
  - 출력 스트림을 닫는 역할을 한다

Thank You !

혼자 공부하는 자바 (신용권 저)