

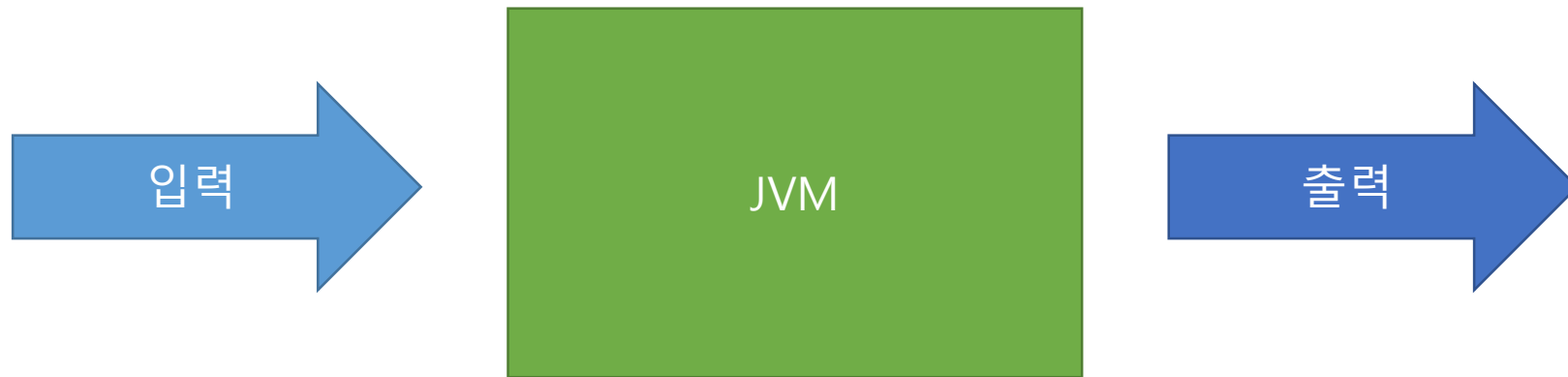
18강 IO기반 입출력 스트림

IO기반 입출력 스트림

- 입출력의 개념
- 입출력 스트림 클래스
- 콘솔, 파일 입출력
- 보조 스트림

입출력의 개념

- IO에서 I는 input, O는 output을 가르킨다.
- 여기서 이야기 하는 입력과 출력은 무엇을 기준으로 입출력을 이야기 하는가?



- 입출력은 자바프로그램을 기준으로 결정한다.

입출력의 개념

- 자바에서 입출력에 관련된 클래스는 java.io패키지에 모여있다
- 데이터 입출력을 위해서 필요한 것이 스트림이라는 것이 필요하다.
- 스트림은 흐름을 뜻하는데 데이터의 흐름, 바뀌서 이야기 하면 데이터가 한방향으로 흘러간다는 의미를 가진다.
- 흐름이 단방향으로 흐르는 빨대를 생각해도 좋다.!

입출력의 개념

- 스트림은 어떤 데이터를 전송하는가를 기준으로 두가지로 구분된다.
 - 바이트 전송용 스트림: 음악,그림등 모든 파일을 전송할 수 있다
 - 문자 전송용 스트림 : 텍스트 문자 전송에 특화되어있다

입출력의 개념

- 각각의 스트림은 또다시 입력과 출력을 담당하는 두개의 추상 클래스로 구분한다
 - 바이트 전송 스트림
 - 입력 : inputStream
 - 출력 : outputStream
 - 문자 전송 스트림
 - 입력 : reader
 - 출력 : writer

입출력 스트림 클래스

- 바이트 입력 스트림 : `InputStream`

- `read()` : 입력 스트림으로 부터 1바이트를 읽고 `int`타입(4바이트)으로 변환한다. 더 이상 읽을 데이터가 없으면 -1을 반환한다.
- `read(byte[] b)` : 주어진 배열 길이만큼 바이트를 읽고 배열에 저장한다. 더 이상 읽을 데이터가 없으면 -1을 반환 : 보다 많은 데이터를 처리할 때 효율적이다.
- `read(byte[] b, int off, int len)` : 주어진 배열의 시작과 길이를 지정할 수 있다. 시작을 0, 길이는 배열의 길이로 주면 `byte[] b`와 같다.
- `close()` 자원 반납(빨대를 뱉는다)

입출력 스트림 클래스

- 바이트 출력 스트림 : `OutputStream`
 - `write()` : 값을 끝의 1바이트만 출력 스트림으로 보낸다.
 - `write(byte[] b)` : 주어진 배열 길이만큼 출력 스트림으로 보낸다.
 - `write(byte[] b, int off, int len)` : 주어진 배열의 시작과 길이를 지정할 수 있다.
 - `flush()` 버퍼에 잔류하는 모든 바이트를 출력한다.
 - `close()` 자원 반납(빨대를 뱉는다)

입출력 스트림 클래스

- 문자 입력 스트림 : `reader :inputStream`과 같지만 단위가 `char`
 - `read()` : 입력스트림으로 부터 하나의 문자를 전송받는다
 - `read(char[] b)` : 주어진 배열 길이만큼 문자를 읽고 배열에 저장한다.
더 이상 읽을 데이터가 없으면 `-1`을 반환 : 보다 많은 데이터를 처리할 때 효율적이다.
 - `read(char[] b, int off, int len)` : 주어진 배열의 시작과 길이를 지정할 수 있다. 시작을 `0`, 길이는 배열의 길이로 주면 `byte[] b`와 같다.
 - `close()` 자원 반납(빨대를 뱉는다)

입출력 스트림 클래스

- 문자 출력 스트림 : `writer` : `output` 스트림과 같다
 - 단 `char` 배열과 `String` 도 존재
 - `write(int c)` : `c` 값을 `char`로 변환 하여 전송
 - `write(char[] cbuf)` : 주어진 배열 길이만큼 출력 스트림으로 보낸다.
 - `write(char[] b, int off, int len)` : 주어진 배열의 시작과 길이를 지정할 수 있다.
 - `write(String str)`: 문자열을 출력스트림으로 보낸다
 - `flush()` 버퍼에 잔류하는 모든 바이트를 출력한다.
 - `close()` 자원 반납(빨대를 뱉는다)

입출력 매체

- 입출력을 위한 대상은 파일을 통한 입출력, 콘솔 입출력 등이 있다

입출력 매체

- 콘솔 입출력
- 콘솔이라함은 시스템 조작 체계를 의미한다. 즉 시스템을 제어하기 위한 명령어를 입력하는 도구로 이해하는 것이 좋다.
- 콘솔의 종류로는 리눅스나 유닉스의 터미널, 윈도우의 명령프롬프트나 파워셸 등이 있다.(이클립스도 지원)

콘솔

- 콘솔 입력 방법

```
public static void main(String[] args) throws Exception {  
    InputStream is = System.in;  
  
    int asciiCode = is.read();  
  
    System.out.println(asciiCode);  
}
```

<termi

a

97

콘솔

- 콘솔 입력 방법 - 문자로 변환

```
public static void main(String[] args) throws Exception {  
    InputStream is = System.in;  
  
    char asciiLetter = (char)is.read();  
  
    System.out.println(asciiLetter);  
}
```

a
a

콘솔

- 콘솔 입력 방법 - 한글등의 문자는 글자당 2바이트 이상의 공간을 차지하므로 String으로 받을 필요가 있다.

```
public static void main(String[] args) throws Exception {  
    InputStream is = System.in;  
    byte[] byteDate = new byte[15];  
  
    int readByteNo = System.in.read(byteDate);  
  
    String str = new String(byteDate);  
  
    System.out.println(str);  
  
}
```

<termi

강
강

콘솔

- 콘솔 출력 방법

```
public static void main(String[] args) throws Exception {  
    OutputStream os = System.out;  
    int b = 97;  
    os.write(b);  
    os.close();  
  
}
```

a

콘솔

- 콘솔 출력 방법 - 한글등의 문자는 2바이트 공간을 차지 하므로 바이트 배열을 사용한다.

```
public static void main(String[] args) throws Exception {  
    OutputStream os = System.out;  
    String name = "고길동";  
    byte[] nameBytes = name.getBytes(); //문자열을 바이트 배열로 전환하는 메소드  
    os.write(nameBytes);  
    os.close();  
}
```

고길동

콘솔

- 입력을 도와주는 도구 클래스 Scanner
- 콘솔 출력을 도와주는 보조스트림 : PrintStream

파일

- 파일 입출력을 진행하기 위해서는 파일 객체를 생성할 필요가 있다.(실제 파일이 존재하지 않더라도 객체 생성은 가능)
 - Exists()메소드가 있다.
 - createNewFile() 새로운 파일을 생성
 - Mkdir() 디렉토리 생성
 - Mkdirs() 경로에 모든 디렉토리 생성
 - Delete() 파일 또는 디렉토리 삭제
 - canExecute() 실행 파일인지 여부
 - canRead() 읽을 수 있는 파일인지 여부
 - getName()파일 이름 반환
 - getParent() 부모 디렉토리 반환
 - getPath() 경로 반환
 - Length() 파일 크기 반환
 - List() 디렉토리의 포함된 파일 및 서브 디렉토리 목록 전부를 String 배열로 반환

파일 입출력 - 문자

- 텍스트 파일을 읽기 위한 문자 단위 스트림
 - `fileReader fr = new FileReader(file);`
 - `read()`는 1바이트씩 읽는다.
 - `char[]`배열로 읽는 것이 효율적

```
public static void main(String[] args) throws Exception {
    File file = new File("E:\\Study\\LECTURE\\Workspace\\Ja
    FileReader fr = new FileReader(file);

    int readCharNo;
    char[] cbuf = new char[100];
    while ((readCharNo=fr.read(cbuf)) != -1) {
        String data = new String(cbuf);
        System.out.print(data);
    }
    fr.close();
}
```

```
package chapter18;

import java.io.IOException;
import java.io.OutputStream;

public class Main05 {

    public static void main(Str
        OutputStream os = S
        String name = "고길
        byte[] nameBytes =
        os.write(nameBytes)
        os.close();
}
```

파일 입출력 - 문자

- 텍스트 파일을 출력을 위한 문자 단위 스트림
 - `fileWriter fw = new fileWriter(file);`

```
public static void main(String[] args) throws Exception {  
    File file = new File("E:/Temp/file.txt");  
    FileWriter fw = new FileWriter(file, true);  
    fw.write("FileWriter는 한글로된 " + "\r\n");  
    fw.write("문자열을 바로 출력할 수 있다." + "\r\n");  
    fw.flush();  
    fw.close();  
    System.out.println("파일에 저장되었습니다.");  
}
```

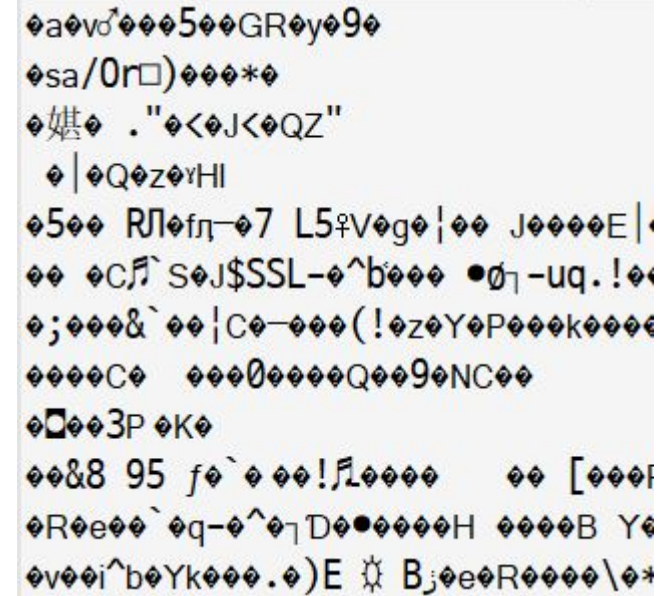
<terminated> main07 (1) [Java Application] C
파일에 저장되었습니다.

file.txt - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
FileWriter는 한글로된
문자열을 바로 출력할 수 있다.

파일 입출력 - 이미지

- 이미지 파일을 읽기 위한 바이트 단위 스트림
 - `FileInputStream fis = new FileInputStream(file);`
 - `read()`는 1바이트씩 읽는다.

```
public static void main(String[] args) throws Exception {  
    File file = new File("E:\\image\\woman-1149911_1920.jpg");  
    FileInputStream fis = new FileInputStream(file);  
    int data;  
    while ( (data = fis.read() ) != -1 ) {  
        System.out.write(data);  
    }  
    fis.close();  
}
```



0a0v0000500GR0y090
0sa/0r0)000*0
0娼0 ."0<0J<0QZ"
0|0Q0z0yHI
0500 R0f0r0-07 L50V0g0|00 J0000E|0
00 0C0f0`S0J\$SSL-0^b000 007-uq.00
0;000&`00|C0-000(!0z0Y0P000k0000
0000C0 00000000Q0090NC00
00003P0K0
00&8 95 f0`000!00000 00 [0000
0R0e00`0q-0^07D00000H 0000B Y0
0v00i^b0Yk000.0)E 0 B0j0e0R0000\00

파일 입출력 -0|미|지|

- 이미지 파일을 읽기 위한 바이트 단위 스트림
 - `FileInputStream fis = new FileInputStream(file);`
 - `read(byte[] b)` 배열로 읽는 것이 효율적

```
public static void main(String[] args) throws Exception {
    File file = new File("E:\\image\\woman-1149911_1920.jpg");
    FileInputStream fis = new FileInputStream(file);
    int readByteNo;
    byte[] readBytes = new byte[100];
    while((readByteNo = fis.read(readBytes)) != -1){
        System.out.println(new String(readBytes));
    }
    fis.close();
}
```

♀♂L_000500 ♀-00 fut v
♂k@f@07|R0ed?0X]0j0K0
Ã0000S0~ZP&0P2060 ♂ 0q0
03 e0l020600
0Wp0^U_007\ s 0oга@0©0♂-4-200c
d_гm00m0000# 000
k0m000 ^V000\0□00
00_00 00|B*zт'010 :0000
0●q00 0C0-nR.0000sN□ 8 V00E0.
00γ00m0026br+040"0Dl0'0
0 0i00:B0S

파일 입출력 - 이미지

- 이미지 파일을 출력을 위한 문자 단위 스트림
 - `FileOutputStream fos = new FileOutputStream(file);`

```
public static void main(String[] args) throws Exception {  
    File file = new File("E:\\Temp\\woman.jpg");  
    FileOutputStream fos = new FileOutputStream(file);  
  
    int readByteNo;  
    byte[] readBytes = new byte[100];  
    fos.write(readBytes);  
  
    fos.flush();  
    fos.close();  
    System.out.println("이미지 출력 완료");  
}
```

<terminated> main0 (1) [Java]
이미지 출력 완료

E:\TEMP\woman.jpg

파일 입출력 - 이미지

• 이미지 복제 예제

```
public static void main(String[] args) throws Exception {
    String originalFileName = "E:\\image\\spring-276014_1920.jpg";
    String targetFileName = "E:\\Temp\\spring.jpg";

    File originalFile = new File(originalFileName);
    File targetFile = new File(targetFileName);

    FileInputStream fis = new FileInputStream(originalFile);
    FileOutputStream fos = new FileOutputStream(targetFile);

    int readByteNo;
    byte[] readBytes = new byte[100];
    while( (readByteNo = fis.read(readBytes)) != -1 ) {
        fos.write(readBytes, 0, readByteNo);
    }

    fos.flush();
    fos.close();
    fis.close();

    System.out.println("복사가 잘 되었습니다.");
}
```

복사가 잘 되었습니다.

보조 스트림

- 보조 스트림은 단독으로 사용할 수는 없고

기본 입출력 스트림에 붙여서 부가적인 기능을 제공하는 목적으로 사용되는 스트림이다.

보조 스트림-Buffered

- 성능향상을 위한 보조스트림 -BufferedInput/OutoutStream

- 프로그램 실행 성능은 컴퓨터 부품중에 가장 느린 장치의 속도를 따라간다.
- 일반적으로 플로피 디스크나 Cd-Rom등이 가장 느린 장치이나 요새는 사용하지 않는 고로 하드디스크가 대체로 가장 느린 장치에 해당한다. (SSD가 나오면서 그 래도 빨라지기는 했다)

- 입출력 스트림을 사용할때 중간 메모리 버퍼를 사용함으로써 입출력시 성능향상을 기대 할수 있다.

보조 스트림 - buffered

```
public static void main(String[] args) throws Exception {  
    long start = 0;  
    long end = 0;  
    String fileName = "E:\\image\\spring-276014_1920.jpg";  
    File file = new File(fileName);  
  
    FileInputStream fis1 = new FileInputStream(file);  
    start = System.currentTimeMillis();  
    while(fis1.read() != -1) {}  
    end = System.currentTimeMillis();  
    System.out.println("사용하지 않았을 때: " + (end-start) + "ms");  
    fis1.close();  
  
    FileInputStream fis2 = new FileInputStream(file);  
    BufferedInputStream bis = new BufferedInputStream(fis2);  
    start = System.currentTimeMillis();  
    while(bis.read() != -1) {}  
    end = System.currentTimeMillis();  
    System.out.println("사용했을 때: " + (end-start) + "ms");  
    bis.close();  
    fis2.close();  
}
```

사용하지 않았을 때: 924ms
사용했을 때: 9ms

보조 스트림 - buffered

```
public static void main(String[] args) throws Exception {
    String originalFileName = "E:\\image\\spring-276014_1920.jpg";
    String targetFileName = "E:\\Temp\\spring.jpg";

    File originalFile = new File(originalFileName);
    File targetFile = new File(targetFileName);

    int data = -1;
    long start = 0;
    long end = 0;

    FileInputStream fis1 = new FileInputStream(originalFile);
    BufferedInputStream bis1 = new BufferedInputStream(fis1);
    FileOutputStream fos1 = new FileOutputStream(targetFile);
    start = System.currentTimeMillis();
    while((data = bis1.read()) != -1) {
        fos1.write(data);
    }
    fos1.flush();
    end = System.currentTimeMillis();
    fos1.close(); bis1.close(); fis1.close();
    System.out.println("사용하지 않았을 때: " + (end-start) + "ms");

    FileInputStream fis2 = new FileInputStream(originalFile);
    BufferedInputStream bis2 = new BufferedInputStream(fis2);
    FileOutputStream fos2 = new FileOutputStream(targetFile);
    BufferedOutputStream bos2 = new BufferedOutputStream(fos2);
    start = System.currentTimeMillis();
    while((data = bis2.read()) != -1) {
        bos2.write(data);
    }
    bos2.flush();
    end = System.currentTimeMillis();
    bos2.close(); fos2.close(); bis2.close(); fis2.close();
    System.out.println("사용했을 때: " + (end-start) + "ms");
}
```

사용하지 않았을 때: 1212ms
사용했을 때: 19ms

보조 스트림 – PrintStream/PrintWriter

- 출력을 도와주는 보조 스트림
 - println()
 - print()
 - printf()
 - 등의 메소드를 사용가능하다

```
public static void main(String[] args) throws Exception {  
    File file = new File("E:/Temp/file.txt");  
    FileWriter fw = new FileWriter(file, true);  
  
    PrintWriter ps = new PrintWriter(fw);  
    ps.println("안녕하세요");  
    ps.print("프린트 출력");  
  
    ps.flush();  
    ps.close();  
    System.out.println("기록 완료");  
}
```

기록 완료

보조 스트림 – PrintStream/PrintWriter

- printf() 형식화된 문자열 출력에 최적화
 - printf(형식화된 문자열, 문자열에 제공할 매개값....)

형식	의미
%d	정수
%숫자d	왼쪽 공백
%-숫자d	오른쪽 공백
%0숫자d	왼쪽 0채움
%f	실수
%10.2f	전체 자리수10, 소수점 아래 2자리
%s	문자열
%%	%표시

형식	의미
%tF	날짜
%tY	년도
%ty	2자리 년도
%tm	월
%td	일
%tH	24시간 시
%th	12시간 시
%tM	분
%tS	초

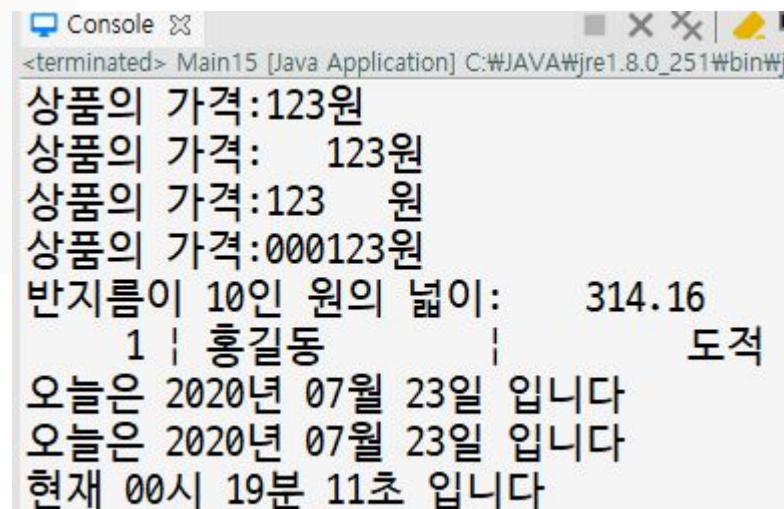
보조 스트림 – PrintStream/PrintWriter

```
public static void main(String[] args) {
    System.out.printf("상품의 가격:%d원\n", 123);
    System.out.printf("상품의 가격:%6d원\n", 123);
    System.out.printf("상품의 가격:%-6d원\n", 123);
    System.out.printf("상품의 가격:%06d원\n", 123);

    System.out.printf("반지름이 %d인 원의 넓이:%10.2f\n", 10, Math.PI*10*10);

    System.out.printf("%6d | %-10s | %10s\n", 1, "홍길동", "도적");

    Date now = new Date();
    System.out.printf("오늘은 %tY년 %tm월 %td일 입니다\n", now, now, now);
    System.out.printf("오늘은 %1$tY년 %1$tm월 %1$td일 입니다\n", now);
    System.out.printf("현재 %1$tH시 %1$tM분 %1$tS초 입니다\n", now);
}
```



```
<terminated> Main15 [Java Application] C:\JAVAW\jre1.8.0_251\bin\W
상품의 가격:123원
상품의 가격:   123원
상품의 가격:123   원
상품의 가격:000123원
반지름이 10인 원의 넓이:   314.16
   1 | 홍길동           |   도적
오늘은 2020년 07월 23일 입니다
오늘은 2020년 07월 23일 입니다
현재 00시 19분 11초 입니다
```


보조 스트림 – dataInput/OutputStream

- 기본 타입 보조스트림

- 바이트 스트림은 바이트 단위로 입출력을 하므로 자바의 기본 데이터 타입 단위로 입출력이 불가능하다.
- 메소드를 통해서 바이트를 기본 타입으로 전환 해준다.

- 다만 기록 순서와 같은 순서로 읽어야 한다.

기록순서가 String double int라면

읽기도 String double int순으로 읽어야 한다. 반대로 입력한 경우 에러발생

- 기록시 writeXXX()메소드
- 읽어올때 readXXX()메소드

보조 스트림 – dataInput/OutputStream

```
public static void main(String[] args) throws Exception {
    String targetFileName = "E:/Temp/aa.xx";
    File targetFile = new File(targetFileName);
    //기록
    FileOutputStream fos = new FileOutputStream(targetFile);
    DataOutputStream dos = new DataOutputStream(fos);

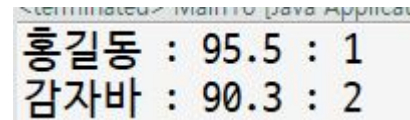
    dos.writeUTF("홍길동");
    dos.writeDouble(95.5);
    dos.writeInt(1);

    dos.writeUTF("감자바");
    dos.writeDouble(90.3);
    dos.writeInt(2);

    dos.flush(); dos.close(); fos.close();
    //읽어오기
    FileInputStream fis = new FileInputStream(targetFile);
    DataInputStream dis = new DataInputStream(fis);

    for(int i=0; i<2; i++) {
        String name = dis.readUTF();
        double score = dis.readDouble();
        int order = dis.readInt();
        System.out.println(name + " : " + score + " : " + order);
    }

    dis.close(); fis.close();
}
```



terminated - My Java Application

홍길동	:	95.5	:	1
감자바	:	90.3	:	2

보조 스트림 – objectInput/OutputStream

- 객체 입출력 보조스트림

- 메모리상에 생성되어 있는 인스턴스 그 자체를 바이트 형식으로 출력할 수 있다.

- writeObject(객체)객체를 직렬화하는 메소드다
- (객체타입)readObject()는 객체를 역 직렬화하는 메소드이다.

보조 스트림 – objectInput/OutputStream

```
public static void main(String[] args) throws Exception {
    String targetFileName = "E:/Temp/Object.xx";
    File targetFile = new File(targetFileName);

    FileOutputStream fos = new FileOutputStream(targetFile);
    ObjectOutputStream oos = new ObjectOutputStream(fos);

    oos.writeObject(new Integer(10));
    oos.writeObject(new Double(3.14));
    oos.writeObject(new int[] { 1, 2, 3 });
    oos.writeObject(new String("홍길동"));

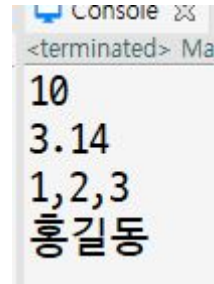
    oos.flush();    oos.close(); fos.close();

    FileInputStream fis = new FileInputStream(targetFile);
    ObjectInputStream ois = new ObjectInputStream(fis);

    Integer obj1 = (Integer) ois.readObject();
    Double obj2 = (Double) ois.readObject();
    int[] obj3 = (int[]) ois.readObject();
    String obj4 = (String) ois.readObject();

    ois.close(); fis.close();

    System.out.println(obj1);
    System.out.println(obj2);
    System.out.println(obj3[0] + "," + obj3[1] + "," + obj3[2]);
    System.out.println(obj4);
}
```



Console

<terminated> Ma

10
3.14
1,2,3
홍길동

보조 스트림 – objectInput/OutputStream

- 객체 입출력 보조스트림
 - 다만 아무 인스턴스나 직렬화가 가능한 것은 아니다.
 - Serializable 인터페이스를 구현해야 직렬화가 가능하다
- 해당 인터페이스를 구현해도 직렬화가 안되는 필드들이 있다
 - static 필드
 - transient 제어자가 있는 필드
 - Serializable 인터페이스를 구현하지 않는 클래스를 타입으로 가지는 필드

보조 스트림 – objectInput/OutputStream

```
public class ClassA implements Serializable{
    int field1;
    ClassB field2 = new ClassB();
    static int field3;    //static은 직렬화 불가
    transient int field4; //직렬화 제외 키워드
}
```

```
public static void main(String[] args) throws Exception {
    String targetFileName = "E:/Temp/Object.xxx";
    File targetFile = new File(targetFileName);
    FileOutputStream fos = new FileOutputStream(targetFile);
    ObjectOutputStream oos = new ObjectOutputStream(fos);
    ClassA classA = new ClassA();
    classA.field1 = 1;
    classA.field2.field1 = 2;
    classA.field3 = 3;
    classA.field4 = 4;
    oos.writeObject(classA);
    oos.flush();    oos.close(); fos.close();
}
```

```
FileInputStream fis = new FileInputStream(targetFile);
ObjectInputStream ois = new ObjectInputStream(fis);
ClassA v = (ClassA) ois.readObject();
System.out.println("field1: " + v.field1);
System.out.println("field2.field1: " + v.field2.field1);
System.out.println("field3: " + v.field3);
System.out.println("field4: " + v.field4);
}
```

```
public class ClassB {
    int field1;
}
```

```
Exception in thread "main" java.io.NotSerializableException:
    at java.io.ObjectOutputStream.writeObject0(Unknown So
    at java.io.ObjectOutputStream.defaultWriteFields(Unkn
    at java.io.ObjectOutputStream.writeSerialData(Unknown
    at java.io.ObjectOutputStream.writeOrdinaryObject(Unk
    at java.io.ObjectOutputStream.writeObject0(Unknown So
    at java.io.ObjectOutputStream.writeObject(Unknown Sou
    at chapter18.Main18.main(Main18.java:22)
```

보조 스트림 – objectInput/OutputStream

```
public class ClassA implements Serializable{
    int field1;
    ClassB field2 = new ClassB();
    static int field3;    //static은 직렬화 불가
    transient int field4;    //직렬화 제외 키워드
}
```

```
public static void main(String[] args) throws Exception {
    String targetFileName = "E:/Temp/Object.xxx";
    File targetFile = new File(targetFileName);
    FileOutputStream fos = new FileOutputStream(targetFile);
    ObjectOutputStream oos = new ObjectOutputStream(fos);
    ClassA classA = new ClassA();
    classA.field1 = 1;
    classA.field2.field1 = 2;
    classA.field3 = 3;
    classA.field4 = 4;
    oos.writeObject(classA);
    oos.flush();    oos.close(); fos.close();

    FileInputStream fis = new FileInputStream(targetFile);
    ObjectInputStream ois = new ObjectInputStream(fis);
    ClassA v = (ClassA) ois.readObject();
    System.out.println("field1: " + v.field1);
    System.out.println("field2.field1: " + v.field2.field1);
    System.out.println("field3: " + v.field3);
    System.out.println("field4: " + v.field4);
}
```

```
public class ClassB implements Serializable {
    int field1;
}
```

```
terminated: main@0: java App
field1: 1
field2.field1: 2
field3: 3
field4: 0
```