java.io 패키지

#01. io 패키지 소개

io란?

프로그램 안으로 데이터가 들어오는 처리(input)와 프로그램으로부터 데이터가 외부로 나가는 처리(output)를 합쳐서 부르는 용어

- 파일에 대한 input : 파일열기(파일읽기)
- 파일에 대한 output : 파일내보내기(파일저장) 파일쓰기
- 네트워크에 대한 input : 다운로드
- 네트워크에 대한 output : 업로드

#02. java.io.File 클래스

파일이나 폴더에 대한 정보를 제공하는 클래스

1) 객체 생성하기

대상의 절대경로를 통해 객체를 생성하는 경우

```
File file = new File("C:/photo/food.jpg");
```

상대경로를 통해 객체를 생성하는 경우

```
File file = new File ("../food.jpg");
```

폴더와 파일 이름을 나누어서 생성자에 전달하는 경우

```
File file = new File ("C:/photo", "food.jpg");
```

> 주어진 경로의 파일이나 폴더가 실제로 존재하지 않아도 객체 생성 가능

2) 절대경로와 상대경로

경로

파일이나 폴더의 위치를 의미하는 문자열

절대경로

해당 파일의 실제 경로를 하드디스크의 최상위 위치부터 순차적으로 명시

대상	Windows 표시값	실제 절대경로 문자열
사용자 홈 디렉토리	C:\사용자\사용자이름\	C:\Users\사용자이름
바탕 화면	C:\사용자\사용자이름\바탕 화면	C:\Users\사용자이름\Desktop
다운로드	C:\사용자\사용자이름\다운로드	C:\Users\사용자이름\Downloads
문서	C:\사용자\사용자이름\문서	C:\Users\사용자이름\Documents

> 사용자 홈 디렉토리는 운영체제에 로그인한 개개인에게 부여되는 개인 폴더. 바탕화 면, 다운로드, 문서 폴더 등이 사용자 홈 디렉토리 안에 위치하고 있다.

상대경로

현재 위치하고 있는 폴더를 기준으로 하는 문자열

표현 방법	의미
.	현재 폴더를 의미 (생략 가능)
..	상위 폴더를 의미

```
/
├── photo1.jpg --> ../photo1.jpg
│
├─folder1       --> ../folder1
│   └── photo2.jpg     --> ../folder1/photo2.jpg
│
└─folder2       --> 현재 위치하고 있는 폴더(기준)
    ├── page.html      --> page.html 혹은 ./page.html
    ├── photo3.jpg     --> photo3.html 혹은 ./photo.html
    │
    └─sub       --> sub 혹은 ./sub
└── photo4.jpg  --> sub/photo4.jpg 혹은 ./sub/photo4.jpg
```

3) File 객체의 메서드

메서드	설명
boolean exists()	File 객체가 담고 있는 경로의 파일이 실제로 존재하지 않을 경우 false 리턴
boolean isFile()	File 객체가 담고 있는 경로의 파일이 존재하지 않거나 폴더인 경우 false 리턴
boolean isDirectory()	File 객체가 담고 있는 경로의 파일이 존재하지 않거나 파일인 경우 false 리턴
boolean isHidden()	File 객체가 담고 있는 경로의 파일이 숨김파일/폴더인지 검사
String getAbsolutePath()	File 객체가 담고 있는 경로의 파일의 절대경로 값을 추출
boolean mkdirs()	폴더 생성하기

메서드	설명
boolean delete()	삭제하기

#03. 파일 입출력

1) Stream이란?

디바이스의 종류에 구애받지 않고 일관된 방식으로 입출력이 수행되도록 추상화 된 형 태.

데이터를 byet[] 형식으로 변환한 상태를 의미한다.

2) java.io 패키지의 OutputStream 인터페이스

스트림 데이터를 프로그램 외부로 내보내는 기능을 제공하는 인터페이스.

OutputStream의 구현체

클래스	설명
FileOutputStream	파일 쓰기 기능을 처리
ByteArrayOutputStream	파일 외의 형태로 데이터를 내보내는 경우 사용 (ex. 네트워크를 통한 송출 기능)

FileOutputStream과 ByteArrayOutputStream은 모두 동일한 인터페이스를 구현하고 있 기 때문에 포함하고 있는 메서드가 같다

즉, 자바에서는 파일 저장이나 네트워크 전송이 구현 방법에 차이가 없다는 의미

FileOutputStream을 통한 파일 쓰기 패턴

```
/** 1) 기본 준비 */
// 저장할 파일의 경로
String filePath = "./test.txt";
// 파일에 저장할 내용
String content = "안녕하세요. 자바";

/** 2) 저장할 내용을 스트림(byte배열->이진수모음)으로 변환 */
byte[] buffer = null;
try {
    buffer = content.getBytes("utf-8");
} catch (UnsupportedEncodingException e) {
    e.printStackTrace();
}

/** 3) 파일 쓰기 */
OutputStream os = null;
try {
    os = new FileOutputStream(filePath);
    os.write(buffer);
```

```
} catch (FileNotFoundException e) {
    System.err.println("[ERROR] " + e.getMessage());
    System.err.println("-------------");
    e.printStackTrace();
} catch (IOException e) {
    System.err.println("[ERROR] " + e.getMessage());
    System.err.println("-------------");
    e.printStackTrace();
} catch (Exception e) {
    System.err.println("[ERROR] " + e.getMessage());
    System.err.println("-------------");
    e.printStackTrace();
} finally {
    // 사용한 스트림은 에러 발생여부에 상관 없이 반드시 닫아야 한다.
    if (os != null) {
        try {
            os.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

3) java.io 패키지의 InputStream 인터페이스

스트림 데이터를 프로그램 내부로 읽어내는 기능을 제공하는 인터페이스.

InputStream의 구현체 클래스

클래스	설명
FileInputStream	파일 읽기 기능을 처리
ByteArrayInputStream	파일 이외의 형태로 존재하는 데이터를 가져오는 경우 사용 (ex. 네트워크를 통한 수신 기능)

FileInputStream을 통한 파일 읽기 패턴

```
/** 1) 기본 준비 */
// 읽어올 파일의 경로
String filePath = "./test.txt";
// 읽어온 내용이 저장될 스트림
byte[] buffer = null;
// 읽어온 내용이 저장되어 있는 스트림이 변환될 문자열
String content = null;

/** 3) 파일 읽기 */
InputStream is = null;

try {
    is = new FileInputStream(filePath);
```

```
        // 파일의 크기(용량)만큼 byte[]의 사이즈를 생성
        buffer = new byte[is.available()];
        // 파일의 내용을 buffer에 담는다.
        is.read(buffer);
    } catch (FileNotFoundException e) {
        System.err.println("[ERROR] " + e.getMessage());
        System.err.println("-------------");
        e.printStackTrace();
    } catch (IOException e) {
        System.err.println("[ERROR] " + e.getMessage());
        System.err.println("-------------");
        e.printStackTrace();
    } catch (Exception e) {
        System.err.println("[ERROR] " + e.getMessage());
        System.err.println("-------------");
        e.printStackTrace();
    } finally {
        if (is != null) {
            try {
                is.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    /** 3) 읽은 내용(byte[])을 문자열로 변환 */
    // buffer 배열에 내용이 있다면 ?
    if (buffer != null) {
        try {
            content = new String(buffer, "utf-8");
            System.out.println(content);
        } catch (UnsupportedEncodingException e) {
            System.out.println("[ERROR] 인코딩 지정 에러");
            e.printStackTrace();
        }
    }
```