Chapter 15 파일 입출력

15.1 입출력 스트림

입출력 스트림이란?

- 자바에서 스트림 : 바이트들의 연속적이 흐름
- 자바에서 어떤 장치에 데이터를 쓰려면 장치와 연결된 스트림을 생성한 후에 스트림에 데이터를 쓰면 되고, 어떤 장치에서 데이터를 읽으려면 장치와 연결된 스트림을 생성한 후에 스트림에서 데이터를 읽으면 된다.
- 만약 스트림이라는 공통적인 입출력 모델이 없었다면 프로그래머는 입출력 장치마다 데이터를 읽고 쓰는 메소드들을 따로 작성해야 할 것이다.
- System.in, System.out에서 System은 우리가 사용하고 있는 컴퓨터 시스템을 나타낸다.

입출력 스트림의 종류

- 바이트 스트림
- 문자 스트림

어떤 스트림을 사용해야 할까?

- 이미지나 압축 파일처럼 이진 파일에서 데이터를 입출력할 때 -> 바이트 스트림
- 텍스트 파일에서 읽을 때 -> 문자 스트림

15.2 문자 스트림

- 문자 스트림은 입출력 단위가 바이트가 아니라 문자다.
- 자바 플랫폼은 유니코드를 사용하여 문자를 저장한다.
- 문자 스트림은 자동적으로 유니코드 문자를 지역 문자 집합으로 변환한다.
- 모든 문자 스트림 클래스는 Reader와 Writer Class로부터 상속되며 모든 문자 스트림은 Reader와 Writer로부터 파생된다.

스트림 생성하기

- FileReader 생성자는 디스크에서 "test.txt" 파일을 찾아서 열고 파일에 스트림을 붙인다.
- 파일을 열 때는 예외가 발생할 수 있으므로 try-catch 문을 사용하여 FileNotFoundException 예외 처리하여야 한다.

```
FileReader fr = new FileReader("test.txt");
```

기본 입출력 메소드

- 문자 스트림에서 문자를 읽고 쓰는 기본 메소드는 read()와 write()다.
- 파일에서 문자를 읽는 경우 반복 루프를 사용한다.
- read()는 int 타입을 반환형으로 해야만 입력 스트림의 끝을 표시하는 -1을 인식할 수 있기 때문에 char 타입이 아닌 int 타입을 반환한다.

```
int ch;
while((ch = fr.read()) != -1)
    System.out.print((char) ch + " ");
```

스트림 닫기

- 스트림은 상당히 소중한 자원이므로 사용이 끝나면 바로 받아야 한다.

```
fr.close();
```

예제 15-1 텍스트 파일 읽기

```
import java.io.*;

public class FileReaderExample {
    public static void main(String[] args) {
        FileReader fr;
        try {
            fr = new FileReader("hello.pdf");
            int ch;
            while ((ch = fr.read()) != -1)
                System.out.print((char) ch + " ");
            fr.close();
        } catch (IOException e) { e.printStackTrace();}
    }
}
```

try-with-resource 사용

```
import java.io.FileReader;
import java.io.IOException;

public class FileReaderExample2 {
    public static void main(String args) throws Exception {
        try (FileReader fr = new FileReader("hello.rtf")) {
            int ch;
            while ((ch = fr.read()) != -1)
                System.out.print((char) ch);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

예제 15-2

```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public class CopyFile1 {
    public static void main(String [] args) throws IOException {
        try(FileReader in = new FileReader("hello.rtf");
            FileWriter out = new FileWriter("hello2.pdf")) {
                int c;
                while ((c = in.read()) != -1) {
                    out.write(c);
                }
            }
    }
}
```

15.3 바이트 스트림

- 바이트 스트림(Byte Stream) : 8비트 단위로 입출력을 수행하는 스트림
- 모든 Byte Stream은 InputStream과 OutStream에서 파생된다.
- FileInputStream은 파일에서 바이트를 읽고 FileOutputStream은 파일에 바이트를 쓴다.
- 기본 메소드는 read() and write()

예제 15-3 이진 파일 쓰기

```
import java.io.FileWriter;
import java.io.FileOutputStream;

import java.io.IOException;

public class FileStreamTest {
    public static void main(String[] args) {
        byte [] list = {10, 20, 30, 40, 50, 60, 70};

        try (FileOutputStream out = new FileOutputStream("test.rtf")) {
            for (byte b : list)
                out.write(b);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

예제 15-4 이진 파일 읽기

```
import java.io.*;
```

```
public class FileStreamTest2 {
    public static void main(String [] args) {
        byte [] list = new byte[6];

        try (FileInputStream out = new FileInputStream("test.rtf")) {
            out.read(list);
        } catch (IOException e) {
            e.printStackTrace();
        }

        for (byte b : list) {
            System.out.print(b + " ");
        }
        System.out.println();
    }
}
// 10 20 30 40 50 60
```

예제 15-5 이미지 파일 복사하기

```

```

15.4 중간 처리 스트림

15,5 객체 저장하기

15.6 파일 객체