

## 8차시 정리

```
// 스트림 입출력
// 버퍼를 가지고 순차적으로 이루어지는 입출력

// 자바의 입출력 스트림
// 입력 스트림 : 입력 장치로부터 자바 프로그램으로 데이터를 전달, 입력장치와 응용 프로그램을 연결하는 객체
// 출력 스트림 : 출력 장치로 데이터 출력, 출력 장치와 응용 프로그램을 연결하는 객체
// 입력장치 ---> 입력 스트림 ---> 자바 응용 프로그램 ---> 출력스트림 ---> 출력 장치

// 스트림
// 스트림의 양 끝에 입출력 장치와 자바 응용프로그램 연결
// 스트림은 단방향으로 선입선출의 구조를 가지고 있다. (큐)
// 바이트 스트림의 경우 : 바이트, 입출력되는 데이터를 단순 바이트로 처리
// 문자 스트림의 경우 : 문자(자바에서는 문자1개가 2바이트), 만자만 입출력하는 스트림

// JDK는 입출력 스트림을 구현한 다양한 클래스 제공
// 바이트 단위 : InputStream, OutputStream(공통적으로 Stream으로 끝남)
// 문자 단위 : Reader, Writer(공통적으로 Reader, Writer로 끝남)

// 표준 입력 스트림 System.in에 InputStreamReader 스트림을 연결한 사례
// 입력 ---> System.in ---> rd(InputStreamReader) ---> 자바응용프로그램

// 문자 스트림
// 유니 코드(2바이트) 문자를 입출력 하는 스트림

// 문자 스트림을 다루는 클래스
// Reader/Writer
// InputStreamReader/OutputStreamWriter
// FileReader/FileWriter

// filereader로 텍스트 파일 읽기
import java.io.*;

public class FileReaderEx {
    public static void main(String[] args) {
        // 파일에 대한 존재가 확실치 않기 때문에 try-catch 구문을 사용해서 안전하게 사용해야 한다.
        // 없으면 구동이 되지 않는다.
        try {
            FileReader fin = new FileReader("c:\\windows\\system.ini");
            int c;
            while ((c = fin.read()) != -1) { // 한 문자씩 파일 끝까지 읽기
                System.out.print((char)c);
            }
            fin.close();
        }
        catch (IOException e) {
            System.out.println("입출력 오류");
        }
    }
}
```

```
// 출력
// ; for 16-bit app support
// [386Enh]
// woafont=dosapp.fon
// EGA80WOA.FON=EGA80WOA.FON
// EGA40WOA.FON=EGA40WOA.FON
// CGA80WOA.FON=CGA80WOA.FON
// CGA40WOA.FON=CGA40WOA.FON
//
// [drivers]
// wave=mmdrv.dll
// timer=timer.drv
//
// [mci]
```

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

```
public class FileReadHangulSuccess {
    public static void main(String[] args) {

        // 파일에 대한 존재가 확실치 않기 때문에 try-catch 구문을 사용해서 안전하게 사용해야 한다.
        // 없으면 구동이 되지 않는다.
        try {

            // 파일을 열겠다는 작업을 하기 위해 선언
            FileInputStream fin = new FileInputStream("C:\\Users\\User\\Desktop\\CH8_1113\\hangul.txt");
            // 일어난 파일에서 객체의 글자를 읽어오는 작업을 하기 위해 선언
            // 텍스트 파일을 UTF-8로 만들어서 오류가 없다.
            InputStreamReader in = new InputStreamReader(fin, "UTF-8");
            int c;

            System.out.println("인코딩 문자 집합은 " + in.getEncoding()); // 인코딩 문자가 무엇인지 알려주는 함수
            while ((c = in.read()) != -1) {
                // 실제로 2가지 단계를 거쳐야 한다. 따라서 각 단계에 필요한 함수를 알고 있어야 함
                System.out.print((char)c);
            }

            in.close(); // 생성 종료
            fin.close(); // 생성 종료
        } catch (IOException e) {
            System.out.println("입출력 오류");
        }
    }
}
```

```
// 출력
// 인코딩 문자 집합은 UTF8
// 가나다라마바사아자차카타파하
```

```

import java.io.*;

public class FileReadHangulFail {
    public static void main(String[] args) {

        // 파일에 대한 존재가 확실치 않기 때문에 try-catch 구문을 사용해서 안전하게 사용해야 한다.
        // 없으면 구동이 되지 않는다.
        try {

            // 파일을 열겠다는 작업을 하기 위해 선언
            FileInputStream fin = new FileInputStream("C:\\Users\\User\\Desktop\\CH8_1113\\hangul.txt");
            // 일어난 파일에서 객체의 글자를 읽어오는 작업을 하기 위해 선언
            // 텍스트 파일을 UTF-8로 만들어서 오류.
            InputStreamReader in = new InputStreamReader(fin, "US-ASCII");
            int c;

            System.out.println("인코딩 문자 집합은 " + in.getEncoding()); // 인코딩 문자가 무엇인지 알려주는 함수
            while ((c = in.read()) != -1) {
                // 실제로 2가지 단계를 거쳐야 한다. 따라서 각 단계에 필요한 함수를 알고 있어야 함
                System.out.print((char)c);
            }

            in.close(); // 생성 종료
            fin.close(); // 생성 종료
        } catch (IOException e) {
            System.out.println("입출력 오류");
        }
    }
}

// 인코딩 문자 집합은 ASCII
// ��

```

```
// FileWriter 사용 예
// 문자 단위로 파일쓰기
// FileWriter fout = new FileWriter("링크");
// fout.write('A');
// fout.close();

// 블록 단위 쓰기
// char [] buf = new char [1024];
// fout.write(buf, 0, buf.length); // 값을 입력을 하든 안하든 buf.length는 1023으로 출력

import java.io.*;
import java.util.*;

public class FileWriterEx {
    public static void main(String[] args) {

        try {

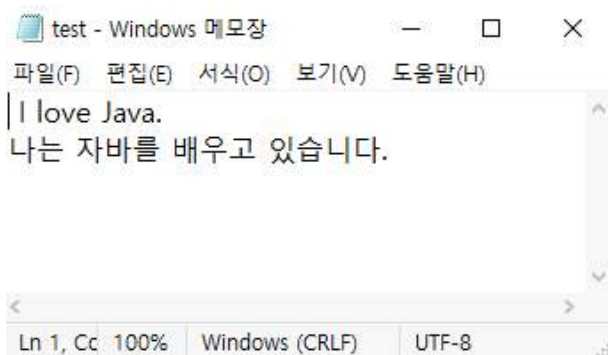
            System.out.println("저장할 값을 입력해 주세요.");
            Scanner scanner = new Scanner(System.in);
            FileWriter fout = new FileWriter("C:\\Users\\User\\Desktop\\CH8_1113\\test.txt");

            while(true) {
                String line = scanner.nextLine(); // 라인의 문자를 읽어서 저장한다.
                if(line.length() == 0) // 입력된 값이 없다면, 즉 엔터를 친다면
                    break; // 반복 중단

                fout.write(line, 0, line.length()); // 읽은 문장을 test파일에 입력
                fout.write("\r\n", 0, 2); // 입력이 끝나면 줄바꿈을 실행한다.
            }
            fout.close(); // 파일 입력 종료
            scanner.close(); // 사용자 입력 종료

        } catch (IOException e) {
            System.out.println("입출력 오류");
        }
    }
}

// 출력
// 저장할 값을 입력해 주세요.
// I love Java.
// 나는 자바를 배우고 있습니다.
//
```



```

// 바이크 스트림
// 바이트 단위의 바이너리 값을 읽고 쓰는 스트림

// 바이트 스트림 클래스
// InputSteam/OutputSteam
// 추상 클래스, 바이트 스트림을 다루는 모든 클래스의 클래스

// FileInputStream/FileOutputStream
// 파일로부터 바이트 단위로 읽거나 저장하는 클래스
// 바이너리 파일의 입출력 용도
// 그냥 문자로 저장하는 것보다 바이트로 저장하는 것이 공간을 효율적으로 쓸수 있기 때문에
// 속달되었다면 되도록 바이트 단위로 저장하는 것이 효율적이다.
// 각 숫자는 1바이트에 저장되므로 8비트, 즉 4비트 2개(헥사 2개)로 저장되기에 16진수 2자리로 표현된다.
// 단, 바이크로 저장했다면 따로 코딩을 통해 보는 방법이 아니면 내용을 알기 힘들다.
// 파일열기, 미리보기 등등 불가

// DataInputStream/DataOutputStream
// 기본 데이터 타입의 값을 바이너리 값 그대로 입출력
// 문자열도 바이너리 형태로 입출력

// FileOutputStream으로 바이너리 파일 쓰기
import java.io.*;

public class FileOutputStreamEx {
    public static void main(String[] args) {
        byte b[] = {7,51,3,4,-1,24};
        try {
            FileOutputStream fout = new FileOutputStream("C:\\Users\\User\\Desktop\\CH8_1113\\test.out");
            for(int i=0; i<b.length; i++)
                fout.write(b[i]); // 배열 b의 바이너리를 기록, 기록된 것을 읽기 위해서는 따로 코드를 짜야한다.
            fout.close();
        }
        catch(IOException e) {
            System.out.println( "C:\\Users\\User\\Desktop\\CH8_1113\\test.out에 저장할 수 없었습니다. 경로명을 확인해 주세요.");
            return;
        }
        System.out.println("C:\\Users\\User\\Desktop\\CH8_1113\\test.out을 저장하였습니다.");
    }
}

// 출력
// C:\Users\User\Desktop\CH8_1113\test.out을 저장하였습니다.

```

// 바이너리 코드로 저장한 파일의 내용을 읽어오는 코드이다.

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

```
public class FileInputStreamEx {
    public static void main(String[] args) {
        byte b[] = new byte [6]; // 비어 있는 byte 배열
        try {
            FileInputStream fin = new FileInputStream("C:\\Users\\User\\Desktop\\CH8_1113\\test.out");

            // 파일읽기
            int n=0, c;
            while((c = fin.read())!= -1) {
                b[n] = (byte)c;
                n++;
            }

            // 파일출력
            System.out.println("C:\\Users\\User\\Desktop\\CH8_1113\\test.out에서 읽은 배열을 출력합니다.");
            for(int i=0; i<b.length; i++)
                System.out.print(b[i] + " ");
            System.out.println();

            fin.close();
        } catch(IOException e) {
            System.out.println( "C:\\Users\\User\\Desktop\\CH8_1113\\test.out에서 읽지 못했습니다. 경로명을 체크해보세요");
        }
    }
}
```

// 출력

// C:\Users\user\Desktop\1113자바수업\test.out에서 읽은 배열을 출력합니다.

// 7 51 3 4 -1 24

```

// 버퍼 스트림
// 버퍼를 가진 스트림
// 입출력 데이터를 일시적으로 저장하는 버퍼를 이용하여 입출력 효율 개선

// 버퍼 입출력의 목적
// 입출력 시 운영체제의 API 호출 횟수를 줄여 입출력 성능 개선
// 출력시 여러 번 출력 되는 데이터를 버퍼에 모아두고 한번에 장치로 출력
// 입력 장치 ---> 입력버퍼(버퍼 입력 스트림) ---> 프로그램 ---> 출력버퍼(버퍼출력스트림) ---> 출력장치

// 바이트 버퍼 스트림
// 바이트 단위의 바이너리 데이터를 처리하는 버퍼 스트림
// BufferedInputStream와 BufferedOutputStream

// 문자 버퍼 스트림
// 유니코드의 문자 데이터만 처리하는 버퍼 스트림
// BufferedReader와 BufferedWriter

import java.io.*;
import java.util.Scanner;

public class BufferedIOEx {
    public static void main(String[] args) {

        try {
            FileReader fin = new FileReader("C:\\Users\\User\\Desktop\\CH8_1113\\test2.txt");
            // 5바이트 크기의 버퍼 설정. System.out 표준 스트림에 추력
            BufferedOutputStream out = new BufferedOutputStream(System.out, 5);
            int c;


            // 출력스트림의 버퍼 크기가 5이므로 파일을 읽어 8개를 출력하려 했지만 5개만 출력이되고
            // 3개는 버퍼에 남아있어 보이지 않음
            while ((c = fin.read()) != -1) {
                out.write(c); // 버퍼가 꽉 찰 때 문자가 화면에 출력
            }

            // Enter키를 받으면 flush()를 실행하여 버퍼에 남아있던 3개 문자를 강제로 출력
            new Scanner(System.in).nextLine();
            out.flush(); // 버퍼에 남아 있던 문자 모두 출력

            fin.close();
            out.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

// 출력
// 12345
// (enter입력시)
// 678

```

 test2 - Wind

파일(F) 편집(E)

|12345678

```

// file 클래스
// 파일의 경로명을 다루는 클래스
// java.io.File
// 파일과 디렉터리 경로명의 추상적 표현

// 파일 관리 기능
// 파일 이름 삭제, 변경, 디렉터리 생성, 크기 등 파일 관리
// File 객체는 파일 읽기 쓰기 기능 없음 -> 따로 구현 필요

import java.io.File;

public class FileEx {
    public static void listDirectory(File dir) {
        System.out.println("-----" + dir.getPath() + "의 서브 리스트 입니다.-----");
        File[] subFiles = dir.listFiles(); // 파일 및 서브디렉터리 리스트 얻기

        for(int i=0; i<subFiles.length; i++) {
            File f = subFiles[i];
            long t = f.lastModified();
            System.out.print(f.getName()); // 파일 이름
            System.out.print("\t파일 크기: " + f.length()); // 경로 전체
            System.out.printf("\t수정된 시간: %tb %td %ta %tT\n",t, t, t, t); // 폴더 경로만
        }
    }

    public static void main(String[] args) {
        File f1 = new File("c:\\windows\\system.ini");
        System.out.println(f1.getPath() + ", " + f1.getParent() + ", " + f1.getName());
        String res="";

        if(f1.isFile()) // 파일인 경우
            res = "파일";
        else if(f1.isDirectory()) // 디렉터리인 경우
            res = "디렉토리";

        System.out.println(f1.getPath() + "은 " + res + "입니다.");

        File f2 = new File("C:\\Users\\User\\Desktop\\CH8_1113\\java_sample");

        if(!f2.exists()) {
            f2.mkdir(); // 존재하지 않으면 디렉토리 생성
        }

        listDirectory(new File("C:\\Users\\User\\Desktop\\CH8_1113\\"));
        f2.renameTo(new File("C:\\Users\\User\\Desktop\\CH8_1113\\javasample"));
        listDirectory(new File("C:\\Users\\User\\Desktop\\CH8_1113\\"));
    }
}

```



```
// 출력
// c:\windows\system.ini, c:\windows, system.ini
// c:\windows\system.ini은 파일입니다.
// -----C:\Users\User\Desktop\CH8_1113의 서브 리스트 입니다.-----
// .metadata      파일 크기: 4096      수정한 시간: 11월 14 화 18:00:58
// ~~생략~~
// hangul.txt      파일 크기: 42      수정한 시간: 11월 13 월 18:55:16
// java_sample     파일 크기: 0      수정한 시간: 11월 15 수 00:08:00
// test.out        파일 크기: 6      수정한 시간: 11월 14 화 23:23:06
// test.txt파일 크기: 57      수정한 시간: 11월 14 화 23:05:12
// test2.txt       파일 크기: 8      수정한 시간: 11월 13 월 19:46:42
// TextCopyEx      파일 크기: 4096   수정한 시간: 11월 14 화 23:36:02
// -----C:\Users\User\Desktop\CH8_1113의 서브 리스트 입니다.-----
// .metadata      파일 크기: 4096   수정한 시간: 11월 14 화 18:00:58
// ~~생략~~
// FileWriterEx   파일 크기: 4096   수정한 시간: 11월 14 화 22:49:43
// hangul.txt      파일 크기: 42     수정한 시간: 11월 13 월 18:55:16
// javasample     파일 크기: 0      수정한 시간: 11월 15 수 00:08:00
// test.out       파일 크기: 6      수정한 시간: 11월 14 화 23:23:06
// test.txt파일 크기: 57      수정한 시간: 11월 14 화 23:05:12
// test2.txt      파일 크기: 8      수정한 시간: 11월 13 월 19:46:42
// TextCopyEx     파일 크기: 4096   수정한 시간: 11월 14 화 23:36:02
```

// 백업 파일 프로그래밍

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

```
public class TextCopyEx {
    public static void main(String[] args){

        File src = new File("c:\\windows\\system.ini"); // 원본 파일 경로명
        File dest = new File("C:\\Users\\User\\Desktop\\CH8_1113\\system.txt"); // 복사 파일 경로명
        int c;

        try {

            FileReader fr = new FileReader(src);
            FileWriter fw = new FileWriter(dest);

            while((c = fr.read()) != -1) { // 문자 하나 읽고
                fw.write((char)c); // 문자 하나 쓰고
            }
            System.out.println(src.getPath()+ "를 " + dest.getPath()+ "로 복사하였습니다.");

            fr.close();
            fw.close();
        } catch (IOException e) {
            System.out.println("파일 복사 오류");
        }
    }
}
```

```
// 출력
// c:\windows\system.ini를 C:\Users\User\Desktop\CH8_1113\system.txt로 복사하였습니다.
```

system - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

for 16-bit app support

[386Enh]

woafont=dosapp.fon

EGA80WOA.FON=EGA80WOA.FON

EGA40WOA.FON=EGA40WOA.FON

CGA80WOA.FON=CGA80WOA.FON

CGA40WOA.FON=CGA40WOA.FON

[drivers]

wave=mmdrv.dll

timer=timer.drv

[mci]

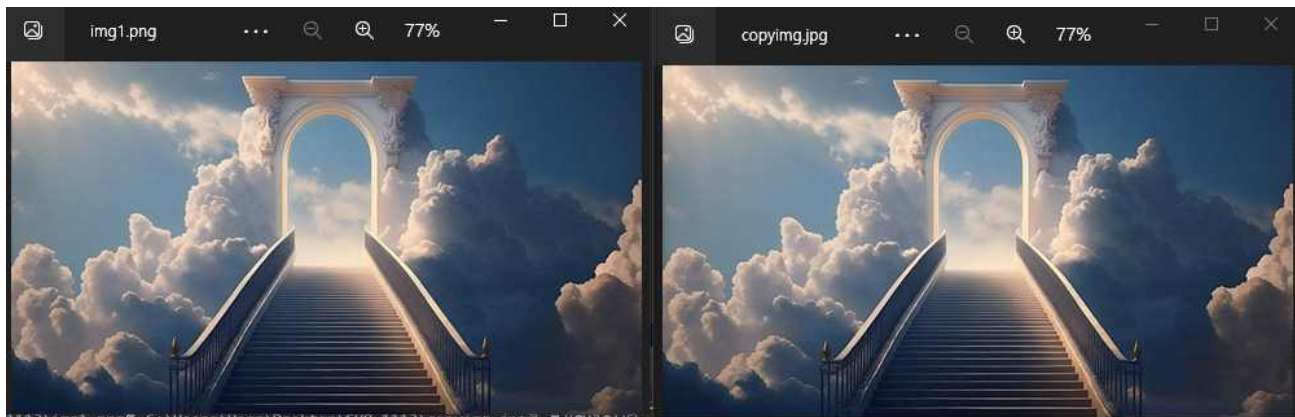
// 바이너리 파일 복사 방법을 이용하여 이미지를 복사하는 코드 작성

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

```
public class BinaryCopyEx {  
    public static void main(String[] args) {  
        File src = new File("C:\\Users\\User\\Desktop\\CH8_1113\\img1.png");  
        File dest = new File("C:\\Users\\User\\Desktop\\CH8_1113\\copyimg.jpg");  
        int c;  
  
        try {  
            FileInputStream fi = new FileInputStream(src);  
            FileOutputStream fo = new FileOutputStream(dest);  
  
            while((c = fi.read()) != -1) {  
                fo.write((byte)c);  
            }  
            System.out.println(src.getPath()+ "를 " + dest.getPath()+ "로 복사하였습니다.");  
  
            fi.close();  
            fo.close();  
        }  
        catch (IOException e) {  
            System.out.println("파일 복사 오류");  
        }  
    }  
}
```

// 출력

// C:\Users\User\Desktop\CH8\_1113\img1.png를 C:\Users\User\Desktop\CH8\_1113\copyimg.jpg로 복사하였습니다.



// 위에 파일을 10KB 단위로 읽고 쓰도록 수정하여 고속으로 파일을 복사하라

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

```
public class BlockBinaryCopyEx {
    public static void main(String[] args) {
        File src = new File("c:\\Windows\\Web\\Wallpaper\\Theme1\\img1.jpg");
        File dest = new File("C:\\Users\\User\\Desktop\\CH8_1113\\desert.jpg");
        try {
            FileInputStream fi = new FileInputStream(src);
            FileOutputStream fo = new FileOutputStream(dest);

            byte [] buf = new byte [1024*10]; // 10KB 버퍼

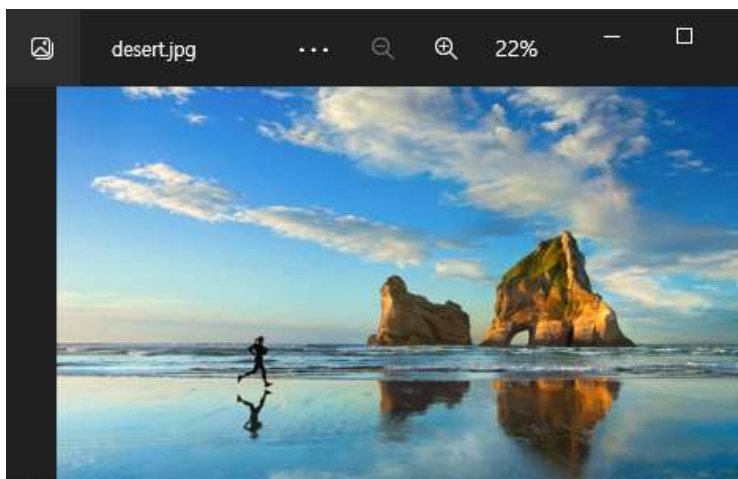
            while(true) {
                int n = fi.read(buf); // 버퍼 크기만큼 읽기. n은 실제 읽은 바이트
                fo.write(buf, 0, n); // buf[0]부터 n 바이트 쓰기

                if(n < buf.length)
                    break;
            }
            fi.close();
            fo.close();

            System.out.println( src.getPath() + "를 " + dest.getPath() + "로 복사하였습니다.");
        }
        catch (IOException e) {
            System.out.println("파일 복사 오류");
        }
    }
}
```

// 출력

// c:\Windows\Web\Wallpaper\Theme1\img1.jpg를 C:\Users\User\Desktop\CH8\_1113\desert.jpg로 복사하였습니다.



```

import java.io.IOException;
import java.io.*;

public class Project1113 {
    public static void listDirectory(File dir, FileWriter Project) {

        try {
            System.out.println("-----" + dir.getPath() + "의 서브 리스트 입니다.-----");
            File[] subFiles = dir.listFiles();

            int r;
            for(int i=0; i<subFiles.length; i++) {
                File f = subFiles[i];
                long t = f.lastModified();
                String text3 = f.getName() + "\t파일 크기: " + f.length()
                    + "\t수정한 시간: %tb %td %ta %tT\n".formatted(t, t, t, t);

                Project.write(text3, 0, text3.length());
                System.out.print(f.getName());
                System.out.print("\t파일 크기: " + f.length());
                System.out.printf("\t수정한 시간: %tb %td %ta %tT\n",t, t, t, t);

            }
        } catch (IOException e) {
            System.out.println("복사 오류");
        }
    }

    public static void main(String[] args) {
        File f1 = new File("c:\\windows\\system.ini");
        try {
            // 복사 파일 경로명
            FileWriter Project = new FileWriter("C:\\Users\\User\\Desktop\\CH8_1113\\Project.txt");
            String text1 = f1.getPath() + ", " + f1.getParent() + ", " + f1.getName();
            System.out.println(f1.getPath() + ", " + f1.getParent() + ", " + f1.getName());
            Project.write(text1, 0, text1.length());

            String res="";
            if(f1.isFile())
                res = "파일";
            else if(f1.isDirectory())
                res = "디렉토리";

            String text2 = f1.getPath() + "은 " + res + "입니다.";
            System.out.println(f1.getPath() + "은 " + res + "입니다.");
            Project.write(text2, 0, text2.length());

            File f2 = new File("C:\\Users\\user\\Desktop\\1113자바수업\\Project_java_sample");
            if(!f2.exists()) {
                f2.mkdir(); // 존재하지 않으면 디렉토리 생성
            }
            listDirectory(new File("C:\\Users\\User\\Desktop\\CH8_1113\\"), Project);
            f2.renameTo(new File("C:\\Users\\User\\Desktop\\CH8_1113\\Projectjavasample"));
            listDirectory(new File("C:\\Users\\User\\Desktop\\CH8_1113\\"), Project);

            Project.close();
        } catch (IOException e) {
            System.out.println("복사 오류");
        }
    }
}

```

```
// 출력
// c:\windows\system.ini, c:\windows, system.ini
// c:\windows\system.ini은 파일입니다.
// -----C:\Users\User\Desktop\CH8_1113의 서브 리스트 입니다.-----
// .metadata      파일 크기: 4096      수정한 시간: 11월 14 화 18:00:58
// Project_java_sample      파일 크기: 0      수정한 시간: 11월 15 수 00:45:04
// system.txt      파일 크기: 219      수정한 시간: 11월 15 수 00:15:12
// TextCopyEx      파일 크기: 4096      수정한 시간: 11월 14 화 23:36:02
// 코드순서.txt      파일 크기: 227      수정한 시간: 11월 15 수 00:20:41
// -----C:\Users\User\Desktop\CH8_1113의 서브 리스트 입니다.-----
// .metadata      파일 크기: 4096      수정한 시간: 11월 14 화 18:00:58
// Projectjavasample      파일 크기: 0      수정한 시간: 11월 15 수 00:45:04
// system.txt      파일 크기: 219      수정한 시간: 11월 15 수 00:15:12
// TextCopyEx      파일 크기: 4096      수정한 시간: 11월 14 화 23:36:02
// 코드순서.txt      파일 크기: 227      수정한 시간: 11월 15 수 00:20:41
```