

Chapter 05. 문자단위 IO 클래스

**Originally made by Prof. Hanku Lee
Modified by Mingyu Lim**

**Collaborative Computing Systems Lab.
School of Internet & Multimedia Engineering
Konkuk University, Seoul, Korea**

1. 문자 단위 IO클래스

□ 문자스트림의 특징

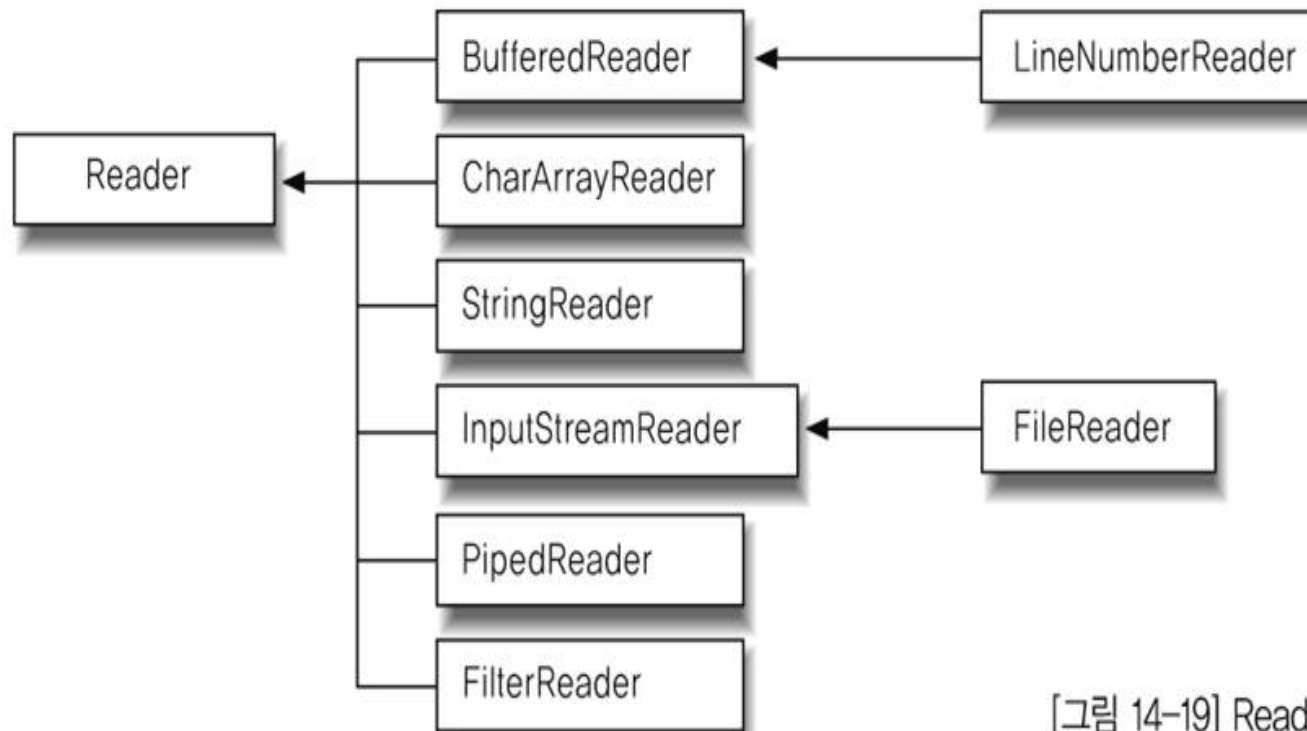
- 바이트 스트림에 추가하여 Reader와 Writer 클래스를 제공하는데, 이것은 2 바이트를 입출력 할 수 있는 문자 기반 스트림이다.
- 바이트 스트림은 1바이트를 입출력하기 때문에 일반적으로 영문자로 구성된 파일, 동영상 파일, 음악 파일의 입출력 등에 적합한 스트림이다.
- 문자 스트림은 2바이트를 입출력하기 때문에 세계 모든 언어로 구성된 파일을 입출력 하기에 적합하다.

□ 문자 스트림의 구조

- 문자 스트림은 Reader와 Writer로 나눈다.
- 문자 입력 스트림 - Reader
- 문자 출력 스트림 - Writer

1. 문자 단위 IO클래스(계속)

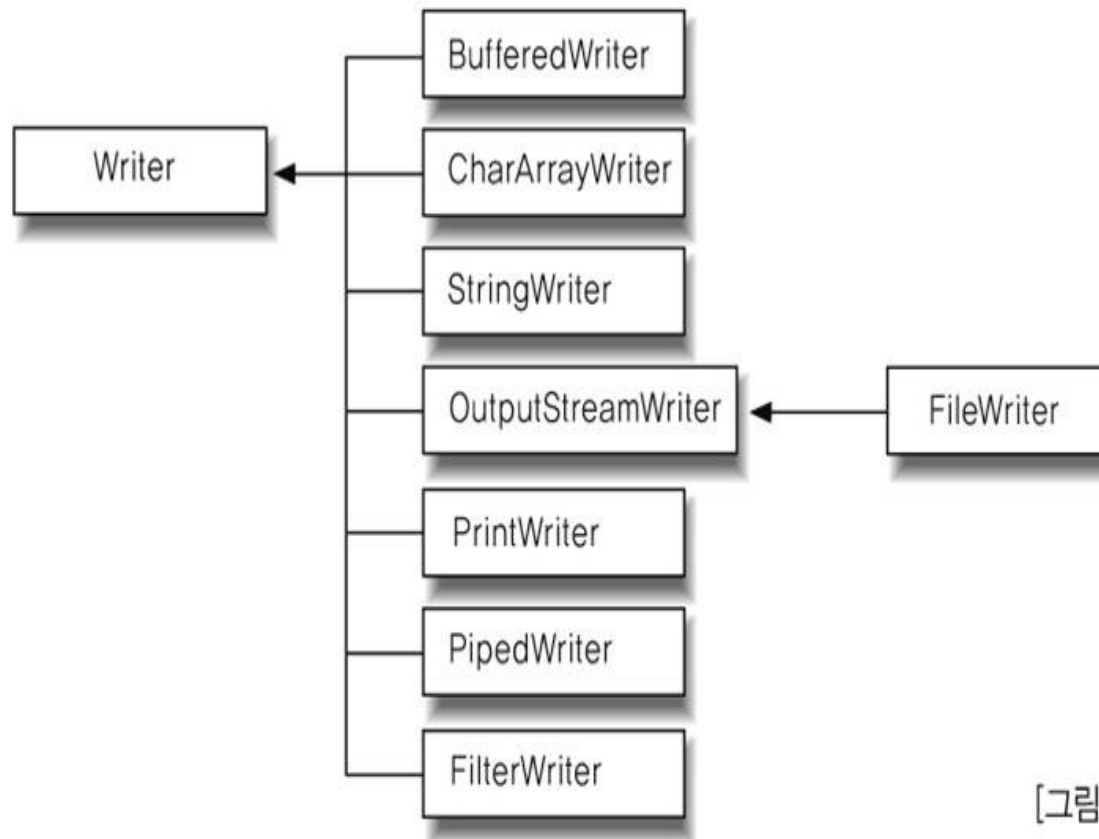
□ 문자입력 스트림의 구조(1)



[그림 14-19] Reader 클래스의 상속도

1. 문자 단위 IO클래스(계속)

□ 문자출력 스트림의 구조(2)



[그림 14-18] Writer 클래스의 상속도

2. Reader와 Writer

□ Reader 와 Writer 클래스

- 문자 단위 입출력 스트림중에서 가장 기본이 되는 클래스
- Reader와 Writer 는 추상클래스 이다.
 - 즉 Reader와 Writer 는 객체화 될수 없다.
- 바이트 단위 입출력 스트림인 InputStream, OutputStream과 사용법이 비슷
- 틀린점은 바이트 단위 입출력 스트림은 바이트 배열을 읽고 쓰는것에 비해, 문자 단위 입출력 스트림은 문자나 문자 배열을 읽고 쓴다.

2. Reader와 Writer(계속)

□ Reader 클래스 중요 메소드

[표 5-1] Reader 클래스의 중요 메소드

메소드	설명
abstract void close() throws IOException	문자 입력 스트림을 닫는다.
void mark(int limit) throws IOException	문자 입력 스트림의 현재 위치를 표시한다.
int read() throws IOException	문자 입력 스트림에서 단일 문자를 읽는다.
int read(char buf[]) throws IOException	문자 입력 스트림에서 buf[] 크기만큼을 읽어 buf에 저장하고 읽은 문자 수를 반환한다.
abstract int read(char buf[], int off, int len) throws IOException	문자 입력 스트림에서 len만큼을 읽어 buf[]의 off 위치에 저장하고 읽은 문자 수를 반환한다.
boolean ready() throws IOException	문자 입력 스트림이 준비되었는지 확인하기 위해 리턴한다.
void reset() throws IOException	문자 입력 스트림을 표시(mark)된 위치로 되돌린다.
long skip(long l) throws IOException	주어진 개수 l만큼의 문자를 건너뛴다.

2. Reader와 Writer(계속)

□ Writer 클래스 중요 메소드

[표 5-2] Writer 클래스의 중요 메소드

메소드	설명
abstract void close() throws IOException	문자 출력 스트림을 닫는다.
abstract void flush() throws IOException	버퍼에 남은 출력 스트림을 출력한다.
void write(String s) throws IOException	주어진 문자열 s를 출력한다.
void write(char buf[]) throws IOException	buf의 내용을 출력한다.
void write(char buf[], int off, int len) throws IOException	buf의 off 위치부터 len만큼의 문자를 출력한다.
void write(String s, int off, int len) throws IOException	주어진 문자열 s에 있는 문자들을 off 위치부터 len만큼 출력한다.

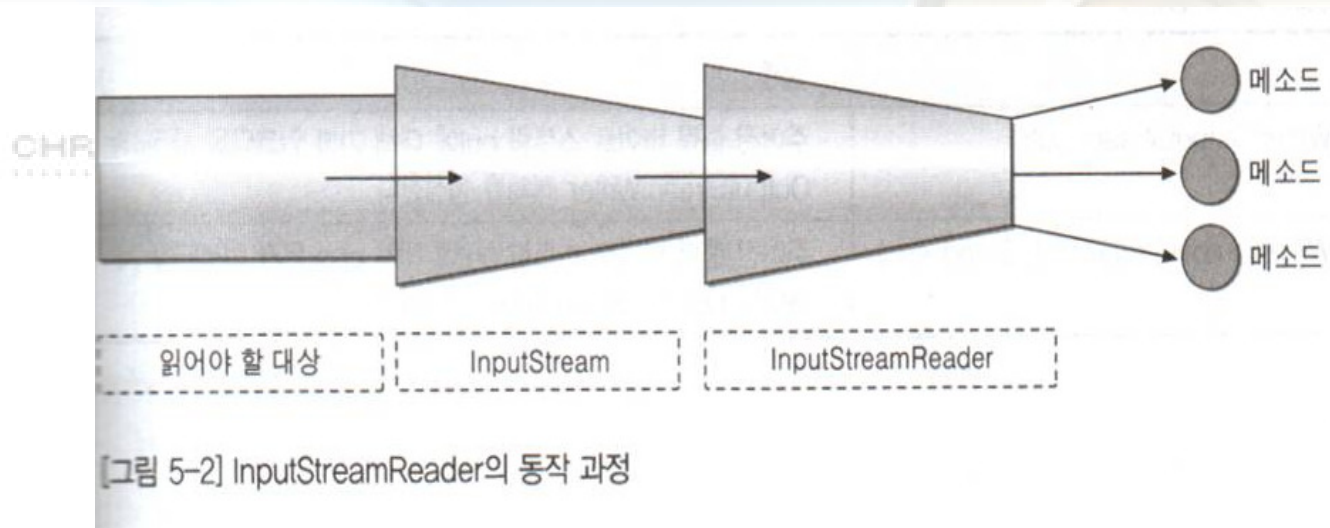
- Reader와 Writer는 객체화될 수 없지만 다음과 같이 사용가능

```
• Reader r = new FileReader("a.txt");
```

3. InputStreamReader와 OutputStreamWriter

□ InputStreamReader의 동작과정

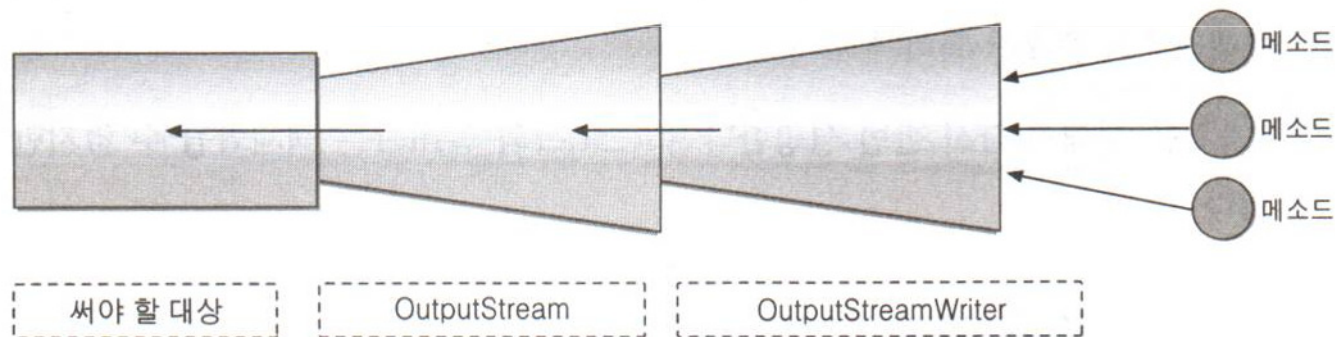
- InputStreamReader는 Reader를 상속 받는다.
- InputStream을 생성자에서 받아들인다.
- InputStreamReader는 바이트 단위로 읽어 들이는 InputStream을 통해 데이터를 읽어 들여 문자 단위로 읽어 들이는 방식으로 변형



3. InputStreamReader와 OutputStreamWriter

□ OutputStreamWriter의 동작과정

- OutputStreamWriter에 있는 메소드를 이용해서 문자를 출력
- OutputStream을 내부적으로 이용해서 써야할 대상에 바이트 단위로 출력



[그림 5-3] OutputStreamWriter의 동작 과정

3. InputStreamReader와 OutputStreamWriter

□ InputStreamReader 생성자

- 인자로 InputStream을 받아들이는것은 InputStream의 자식과 자손 객체를 받아들일 수 있다는것을 의미

[표 5-3] InputStreamReader 생성자

생성자	설명
InputStreamReader(InputStream in)	주어진 입력 바이트 스트림 in에 대해 기본 인코딩을 사용하는 InputStreamReader 객체를 생성한다.
InputStreamReader(InputStream in, String enc)	주어진 입력 바이트 스트림 in에 대해 enc 문자 인코딩을 사용하는 InputStreamReader 객체를 생성한다.

3. InputStreamReader와 OutputStreamWriter

□ OutputStreamWriter 생성자

- 인자로 OutputStream을 받아들이는것은 OutputStream의 자식과 자손 객체를 받아들일 수 있다는것을 의미

[표 5-4] OutputStreamWriter 생성자

생성자	설명
OutputStreamWriter(OutputStream out)	주어진 출력 바이트 스트림 out에 대해 기본 인코딩을 사용하는 OutputStreamWriter 객체를 생성한다.
OutputStreamWriter(OutputStream out, String enc)	주어진 출력 바이트 스트림 out에 대해 enc 문자 인코딩을 사용하는 OutputStreamWriter 객체를 생성한다.

3. InputStreamReader와 OutputStreamWriter

□ 문자 단위로 파일 내용을 읽어 들여 화면에 출력하기(예제)

```
StreamReaderTest.java X

import java.io.*;

public class StreamReaderTest {

    public static void main(String[] args) {
        if(args.length != 1){
            System.out.println("사용법 : java StreamReaderTest 파일명");
            System.exit(0);
        }

        FileInputStream fis = null;
        InputStreamReader isr = null;
        OutputStreamWriter osw = null;
        try{
            fis = new FileInputStream(args[0]);
            isr = new InputStreamReader(fis);
            osw = new OutputStreamWriter(System.out);
            char[] buffer = new char[512];
            int readcount = 0;
            while((readcount = isr.read(buffer)) != -1){
                osw.write(buffer, 0, readcount);
            }
        } catch (Exception ex) {
            System.out.println(ex);
        } finally{
            try{
                fis.close();
                isr.close();
                osw.close();
            } catch (Exception e) {}
        }
    }
}
```

3. InputStreamReader와 OutputStreamWriter

- 문자 단위로 파일 내용을 읽어 들여 화면에 출력하기(결과)

```
C:\Wjava\workspace>cd bin
C:\Wjava\workspace>java StreamReaderTest
Exception in thread "main" java.lang.NoClassDefFoundError: StreamReaderTest

C:\Wjava\workspace>cd bin
C:\Wjava\workspace>java StreamReaderTest ..\src\StreamReaderTest.java

import java.io.*;

public class StreamReaderTest {

    public static void main(String[] args) {
        if(args.length != 1){
            System.out.println("사용법 : java StreamReaderTest 파일명");
            System.exit(0);
        }

        FileInputStream fis = null;
        InputStreamReader isr = null;
        OutputStreamWriter osw = null;
        try{
            fis = new FileInputStream(args[0]);
            isr = new InputStreamReader(fis);
            osw = new OutputStreamWriter(System.out);
            char[] buffer = new char[512];
            int readcount = 0;
            while((readcount = isr.read(buffer)) != -1){
                osw.write(buffer, 0, readcount);
            }
        }catch(Exception ex){
            System.out.println(ex);
        }finally{
            try{
                fis.close();
                isr.close();
                osw.close();
            }catch(Exception e){}
        }
    }
}
```

4. FileReader와 FileWriter

□ FileReader 클래스 생성자

생성자	설 명
FileReader(String filepath) throws FileNotFoundException	Filepath로 지정한 파일에 대한 FileReader 객체를 생성한다.
FileReader(File fileObj) throws FileNotFoundException	FileObj로 지정한 파일에 대한 FileReader 객체를 생성한다.

□ FileWriter 클래스 생성자

생성자	설 명
FileWriter(String filepath) throws IOException	Filepath로 지정한 파일에 대한 출력 스트림을 생성한다.
FileWriter(String filepath, Boolean append) throws IOException	지정한 파일로 출력 스트림을 생성한다. append 인자로 출력할때 파일에 대한 append 모드를 설정한다.
FileWriter(File fileObj) throws IOException	FileObj로 지정된 파일에 대한 출력 스트림을 생성한다.

4. FileReader와 FileWriter(계속)

□ FileReader와 FileWriter를 이용한 텍스트 파일 복사(예제)

- 실제 개발시 파일을 복사하는 프로그램의 작성은 바이트 단위 IO 클래스인 `FileInputStream`과 `FileOutputStream`을 이용하는 것이 좋다
 - 모든 파일이 바이트 단위로 구성되어 있기 때문

```
StreamReaderTest.java  *FileCopy.java x

import java.io.*;

public class FileCopy {

    public static void main(String[] args) {
        if(args.length != 2){
            System.out.println("사용법 : java FileCopy 파일명1 파일명2");
            System.exit(0);
        }

        FileReader fr = null;
        FileWriter fw = null;
        try{
            fr = new FileReader(args[0]);
            fw = new FileWriter(args[1]);
            char[] buffer = new char[512];
            int readcount = 0;
            while((readcount = fr.read(buffer)) != -1){
                fw.write(buffer, 0, readcount);
            }
            System.out.println("파일을 복사하였습니다.");
        } catch (Exception ex) {
            System.out.println(ex);
        } finally{
            try{
                fr.close();
                fw.close();
            } catch (Exception e) {}
        }
    }
}
```

4. FileReader와 FileWriter(계속)

□ FileReader와 FileWriter를 이용한 텍스트 파일 복사(결과)

```
C:\> 명령 프롬프트
C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin>java FileCopy
y ..\src\FileCopy.java a.java
파일을 복사하였습니다.

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin>cd..

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장>cd bin

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin>dir/w
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 747A-FF42

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin 디렉터리
[.]                [...]
FileCopy.class      StreamReaderTest.class
3개 파일           3,725 바이트
2개 디렉터리       84,231,651,328 바이트 남음

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin> _
```


5. BufferedReader와 BufferedWriter

□ BufferedReader

- 바이트 입력 스트림의 `BufferedInputStream`과 동일한 기능을 제공하는 `BufferedReader` 클래스는 문자 출력 스트림의 효율적인 버퍼링을 가능하게 한다.
- `BufferedReader` 클래스에는 `readLine()` 메소드가 추가 되었는데, 이 메서드는 한 줄 단위로 읽는 메소드 이다.

[표 14-15] `BufferedReader`의 생성자

생성자	설명
<code>BufferedReader(Reader in)</code>	매개변수인 <code>Reader</code> 객체로 <code>BufferedReader</code> 객체를 생성한다.
<code>BufferedReader(Reader in, int size)</code>	매개변수인 <code>Reader</code> 객체로 <code>BufferedReader</code> 객체를 생성하고 <code>size</code> 는 버퍼의 크기를 정하는 부분인데, 만약 지정하지 않으면 8192문자를 정할 수 있는 버퍼가 생성된다.

5. BufferedReader와 BufferedWriter(계속)

□ BufferedWriter

- 바이트 출력 스트림의 BufferedOutputStream과 동일한 기능을 제공하며, 문자 출력 스트림의 효율적인 버퍼링을 가능하게 한다.
- 객체를 생성할 때는 예외처리를 하지 않아도 된다.

[표 14-18] BufferedWriter의 주요 생성자

생성자	설명
BufferedWriter(Writer out)	매개변수인 Reader 객체로 BufferedWriter 객체를 생성한다.
BufferedWriter (Writer out, int size)	매개변수인 Reader 객체로 BufferedWriter 객체를 생성하고 size는 버퍼의 용량을 정하는 부분인데, 만약 지정하지 않으면 8192문자를 정할 수 있는 버퍼가 생성된다.

5. BufferedReader와 BufferedWriter(계속)

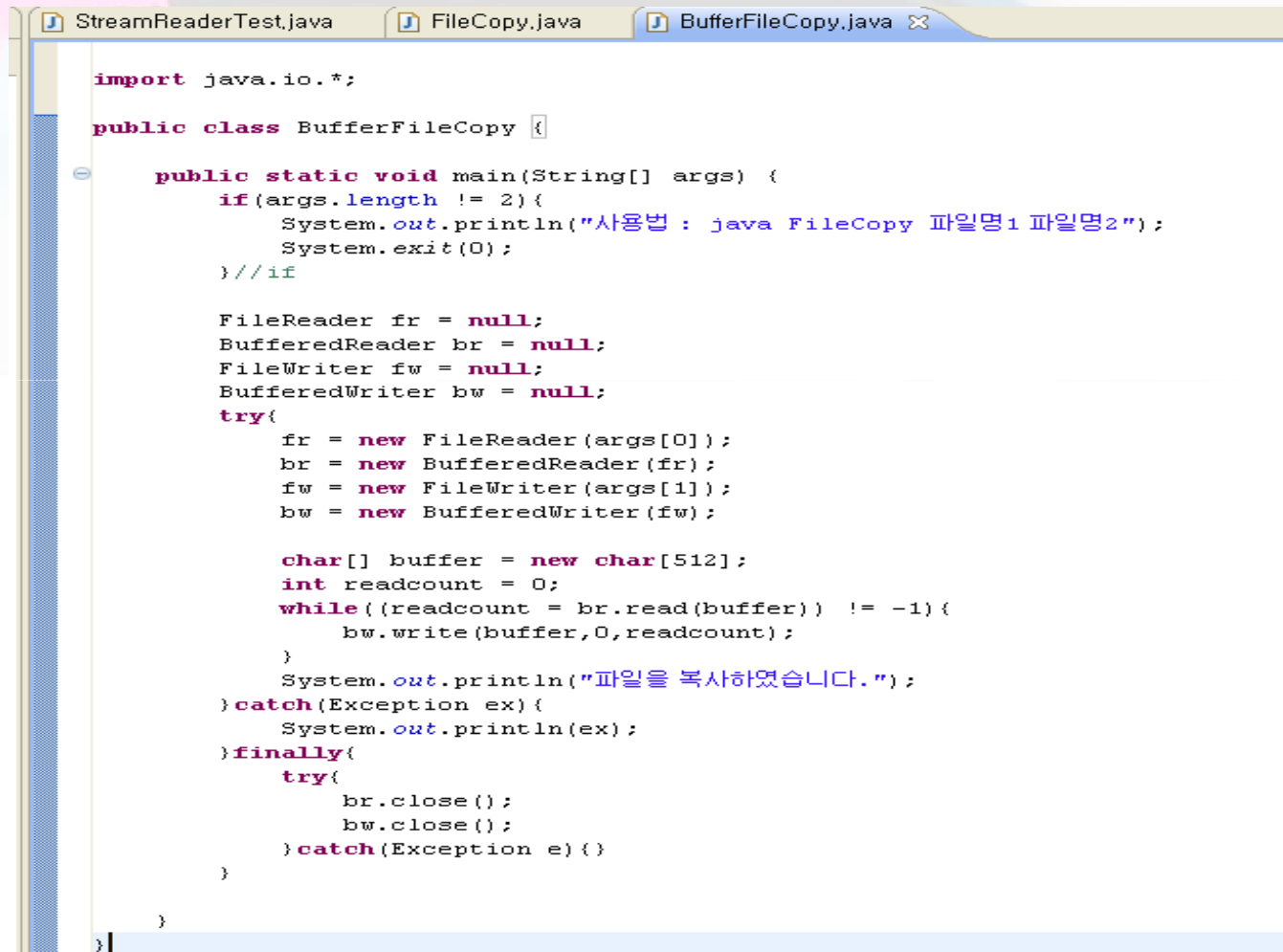
□ FileCopy.java 예제 프로그램의 개선

- FileCopy.java 예제에서 사용된 FileReader와 FileWriter 클래스는 버퍼가 없기 때문에 병목현상 발생
- 짧은 파일은 상관없지만 큰 파일을 복사할 때에는 속도가 느릴수 있다.
- 이를 해결하기 위해서는 BufferedReader와 BufferedWriter를 이용해서 IO 클래스에게 버퍼를 추가하면 된다

ACHROMATIC COLOR

5. BufferedReader와 BufferedWriter(계속)

□ FileCopy.java 예제 프로그램의 개선(예제)



```
import java.io.*;

public class BufferFileCopy {

    public static void main(String[] args) {
        if (args.length != 2) {
            System.out.println("사용법 : java FileCopy 파일명1 파일명2");
            System.exit(0);
        } //if

        FileReader fr = null;
        BufferedReader br = null;
        FileWriter fw = null;
        BufferedWriter bw = null;
        try {
            fr = new FileReader(args[0]);
            br = new BufferedReader(fr);
            fw = new FileWriter(args[1]);
            bw = new BufferedWriter(fw);

            char[] buffer = new char[512];
            int readcount = 0;
            while ((readcount = br.read(buffer)) != -1) {
                bw.write(buffer, 0, readcount);
            }
            System.out.println("파일을 복사하였습니다.");
        } catch (Exception ex) {
            System.out.println(ex);
        } finally {
            try {
                br.close();
                bw.close();
            } catch (Exception e) {}
        }
    }
}
```

5. BufferedReader와 BufferedWriter(계속)

□ FileCopy.java 예제 프로그램의 개선(결과)

```
C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io.nio5.장\bin>java BufferFileCopy ..\src\BufferFileCopy.java b.java 명령창
파일을 복사하였습니다.

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io.nio5.장\bin>dir/w
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 747A-FF42

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io.nio5.장\bin 디렉터리
[.] 생성된 파일
b.java
[.]
BufferFileCopy.class
a.java
FileCopy.class
StreamReaderTest.class
5개 파일 6,257 바이트
2개 디렉터리 84,231,364,608 바이트 남음

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io.nio5.장\bin>_
```

6. PrintWriter

□ PrintWriter 클래스

- PrintWriter 클래스에는 print(), println() 등의 다양한 출력 메소드 존재
- print(), println() 메소드는 여러 종류의 데이터를 출력가능 토록 오버로딩 되어있다.
- print(), println() 메소드 다른점은 println(s) 의경우 출력시 개행문자를 덧붙여서 출력한다.
- PrintWriter 클래스 생성자

생성자	설 명
PrintWriter(OutputStream out)	OutputStream out 을 인자로 전달받아 PrintWriter 객체를 생성
PrintWriter(OutputStream out, boolean autoFlush)	OutputStream out 을 인자로 전달받아 PrintWriter 객체를 생성 이때 두번째 인자인 autoFlush를 true로 지정하면 출력 시에 자동으로 flush() 메소드를 호출한 효과가 발생한다.
PrintWriter(Write out)	Write out 을 인자로 전달받아 PrintWriter 객체를 생성한다.
PrintWriter(Write out, boolean autoFlush)	Write out 을 인자로 전달받아 PrintWriter 객체를 생성한다. 이때 두번째 인자인 autoFlush를 true로 지정하면 출력 시에 자동으로 flush()메소드를 호출한 효과가 발생한다.

6. PrintWriter(계속)

□ PrintWriter 클래스

• PrintWriter 중요 메소드

[표 5-11] PrintWriter의 중요 메소드

메소드	설명
void print(boolean b)	boolean b 값을 출력한다.
void print(char c)	char c를 출력한다.
void print(int i)	int i를 출력한다.
void print(long l)	long l을 출력한다.
void print(float f)	float f를 출력한다.
void print(double d)	double d를 출력한다.
void print(char[] s)	char 배열 s를 출력한다.
void print(String s)	문자열 s를 출력한다.
void print(Object obj)	Object obj의 toString() 메소드가 반환하는 문자열을 출력한다.
void println()	개행문자를 출력한다.
void println(boolean b)	boolean b 값을 출력한다. 그 후 개행문자를 출력한다.
void println(char c)	char c를 출력한다. 그 후 개행문자를 출력한다.
void println(int i)	int i를 출력한다. 그 후 개행문자를 출력한다.
void println(long l)	long l을 출력한다. 그 후 개행문자를 출력한다.
void println(float f)	float f를 출력한다. 그 후 개행문자를 출력한다.
void println(double d)	double d를 출력한다. 그 후 개행문자를 출력한다.
void println(char[] s)	char 배열 s를 출력한다. 그 후 개행문자를 출력한다.
void println(String s)	문자열 s를 출력한다. 그 후 개행문자를 출력한다.
void println(Object obj)	Object obj의 toString() 메소드가 반환하는 문자열을 출력한다. 그 후 개행문자를 출력한다.

6. PrintWriter(계속)

□ 키보드로부터 한 줄씩 입력 받아 파일에 저장하기(예제)

```
StreamReaderTest.java  FileCopy.java  BufferFileCopy.java  LineWriter.java x
import java.io.*;

public class LineWriter {

    public static void main(String[] args) {
        if (args.length != 1) {
            System.out.println("사용법 : java LineWriter 파일명");
            System.exit(0);
        } // if

        BufferedReader br = null;
        PrintWriter pw = null;
        try {
            br = new BufferedReader(new InputStreamReader(System.in));
            pw = new PrintWriter(new BufferedWriter(new FileWriter(args[0])));
            String line = null;
            while ((line = br.readLine()) != null) {
                System.out.println("읽어들인 문자열 : " + line);
                pw.println(line);
            }
        } catch (Exception ex) {
            System.out.println(ex);
        } finally {
            try {
                pw.close();
            } catch (Exception e) {}
        }
    } // main
}
```

- 키보드로부터 한줄씩 입력받으려면 `System.in`과 `BufferedReader`의 `readLine()` 이용
- 파일에 읽어들이 내용을 저장하려면 `FileWriter`, `BufferedWriter`, `PrintWriter` 이용

6. PrintWriter(계속)

□ 키보드로부터 한 줄씩 입력 받아 파일에 저장하기(결과)

```
C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin>dir/w
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 747A-FF42

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin 디렉터리

[.]
a.java
BufferFileCopy.class
FileCopy.class
StreamReaderTest.class
2개 파일
2개 디렉터리
[.]
b.java
CharArrayInputOutputTest.class
LineWriter.class
8,293 바이트
84,117,671,936 바이트 남음

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin>java LineWriter line.txt
열심히 공부합니다.
읽어들인 문자열 : 열심히 공부합니다.
자바 io nio 프로그램 공부
읽어들인 문자열 : 자바 io nio 프로그램 공부
^Z

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin>type lint.txt
t
지정된 파일을 찾을 수 없습니다.

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin>type line.txt
t
열심히 공부합니다.
자바 io nio 프로그램 공부

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin>
```

7. CharArrayReader와 CharArrayWriter

□ CharArrayReader 생성자

생성자	설 명
CharArrayReader(char[] buf)	char 배열 buf로부터 읽어 들이는 CharArrayReader객체를 생성
CharArrayReader(char[] buf, int offset, int length)	Char 배열 buf의 offset 위치부터 length까지 읽어 들이는 CharArrayReader 객체를 생성한다.

□ CharArrayWriter 생성자

생성자	설 명
CharArrayWriter()	char 배열을 내부적으로 저장할 수 있는 CharArrayWriter 객체를 생성한다.

7. CharArrayReader와 CharArrayWriter(계속)

□ CharArrayWriter 의 중요 메소드

생성자	설 명
public char[] toCharArray()	CharArrayWriter의 내부 저장 공간에 저장된 char 배열을 반환

7. CharArrayReader와 CharArrayWriter(계속)

□ Char 배열 형태로 만든후 이 내용을 화면에 출력하기(예제)

```
import java.io.*;

public class CharArrayInputOutputTest {

    public static void main(String[] args) {
        if(args.length != 1){
            System.out.println("사용법 : java CharArrayInputOutputTest 파일이름");
            System.exit(0);
        } // if end

        FileReader fis = null;
        CharArrayReader bais = null;
        CharArrayWriter baos = null;
        try{
            fis = new FileReader(args[0]);
            baos = new CharArrayWriter();
            char[] buffer = new char[512];
            int readcount = 0;

            while ((readcount = fis.read(buffer)) != -1){
                baos.write(buffer, 0, readcount);
            }

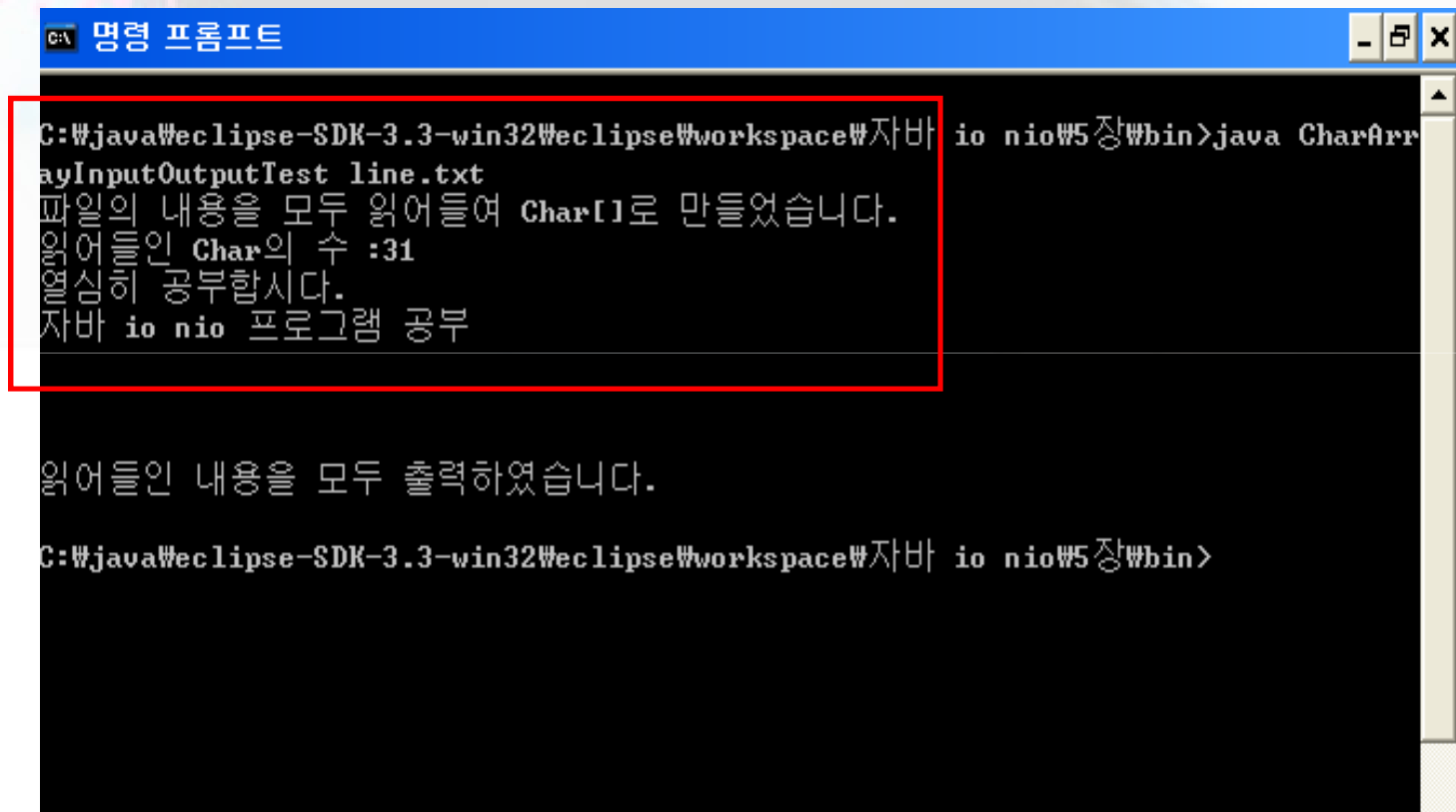
            char[] fileArray = baos.toCharArray();
            System.out.println("파일의 내용을 모두 읽어들이 Char[] 로 만들었습니다.");
            System.out.println("읽어들인 Char의 수 : " + fileArray.length);

            bais = new CharArrayReader(fileArray);
            BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(System.out));

            while ((readcount = bais.read(buffer)) != -1){
                bw.write(buffer, 0, readcount);
                bw.flush();
            }
            System.out.println("\n\n");
            System.out.println("읽어들인 내용을 모두 출력하였습니다.");
        } catch (Exception ex) {
            System.out.println(ex);
        } finally{
            try{
                fis.close();
                bais.close();
                baos.close();
            } catch (IOException ioe) {
                System.out.println(ioe);
            }
        }
    }
}
```

7. CharArrayReader와 CharArrayWriter(계속)

- Char 배열 형태로 만든후 이 내용을 화면에 출력하기(결과)



```
C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin>java CharArrayInputOutputTest line.txt
파일의 내용을 모두 읽어들이 Char[]로 만들었습니다.
읽어들인 Char의 수 :31
열심히 공부합시다.
자바 io nio 프로그램 공부

읽어들인 내용을 모두 출력하였습니다.

C:\Wjava\weclipse-SDK-3.3-win32\weclipse\workspace\자바 io nio\5장\bin>
```

7. StringReader와 StringWriter

□ StringReader / StringWriter 클래스

- CharArrayReader, CharArrayWriter 와 흡사한 클래스
- StringReader는 문자열로부터 읽어 들이기 위한 클래스
- StringWriter는 자신의 내부 저장공간에 출력된 내용을 문자열로 반환하는 기능을 가진 클래스
- StringReader 클래스 생성자

생성자	설 명
StringReader(String s)	String s로부터 읽어 들이는 StringReader 객체를 생성한다.

7. StringReader와 StringWriter(계속)

□ StringReader / StringWriter 클래스

- StringWriter 클래스 생성자

생성자	설 명
StringWriter()	내부 저장 공간이 있는 StringWriter 객체를 생성한다.

- StringWriter 클래스의 중요 메소드

생성자	설 명
public StringBuffer getBuffer()	StringWriter의 내부 저장 공간에 저장된 문자열을 StringBuffer 형태로 반환한다.
public String toString()	StringWriter의 내부 저장 공간에 저장된 문자열을 String 형태로 반환한다.