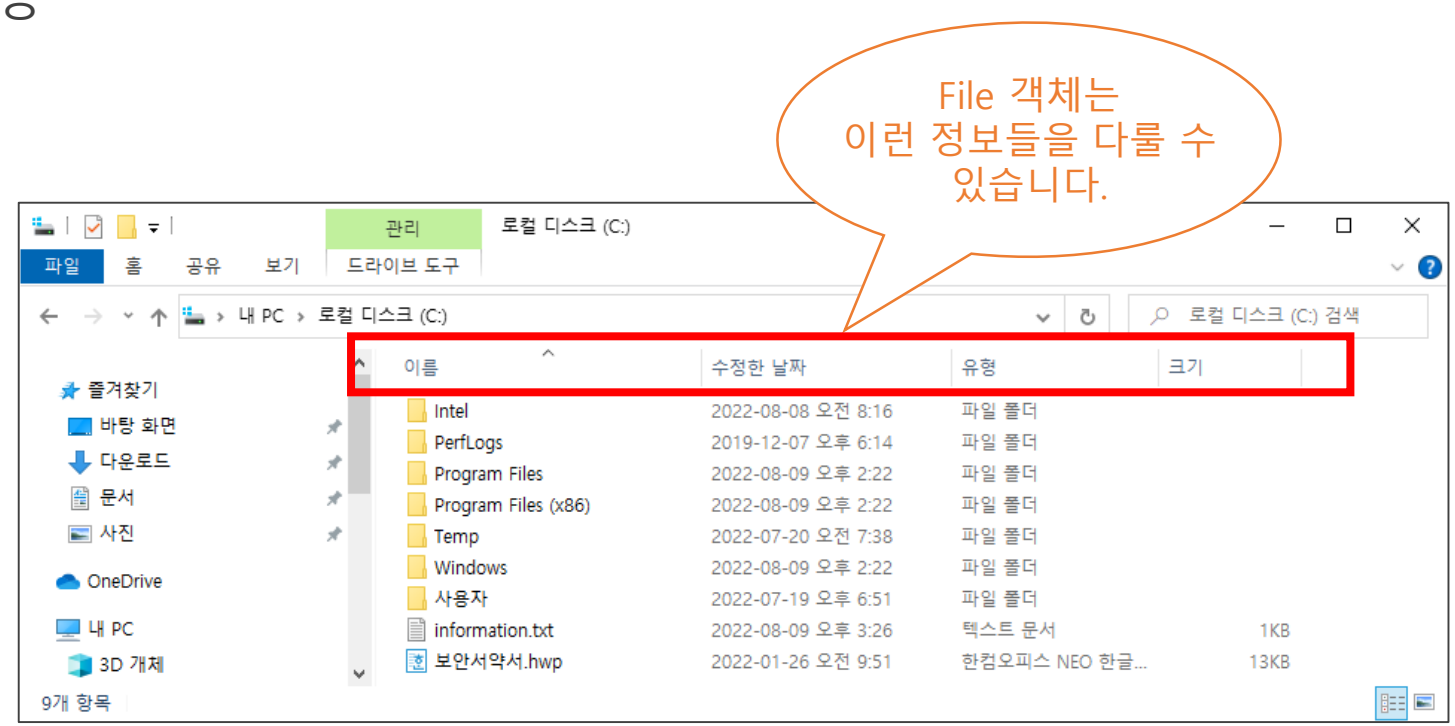
입출력 및 파일

File 클래스

- 패키지 : java.io
- 파일의 경로와 파일명을 이용해 해당 파일을 객체로 생성
- File 객체는 파일의 내용을 읽거나 쓰기를 할 수 없음
- 각종 파일 및 디렉터리 기능 제공
  - ✓ 파일명 확인
  - ✓ 경로 확인
  - ✓ 수정한 날짜 확인
  - ✓ 파일 크기 확인
  - ✓ 파일 생성 및 삭제
  - ✓ 디렉터리 생성 및 삭제



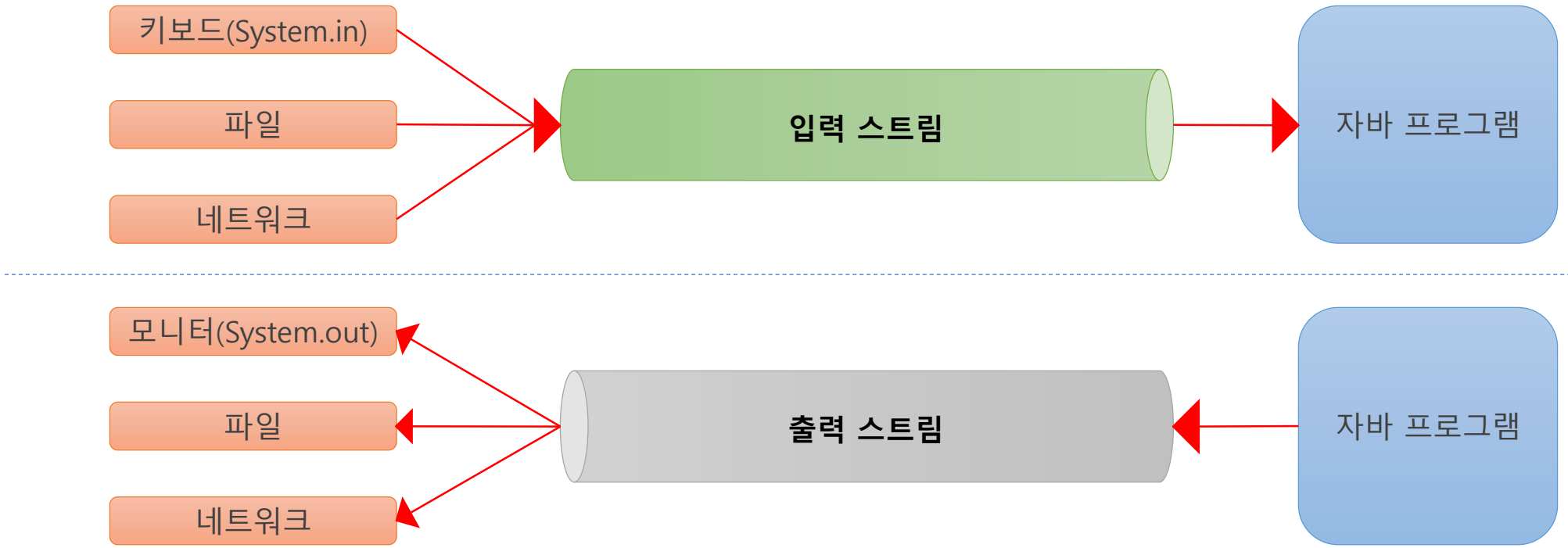
Java

File 클래스 주요 메소드

메소드	역할
boolean isDirectory()	디렉터리(폴더) 여부를 true/false로 반환
boolean isFile()	파일 여부를 true/false로 반환
String getName()	파일 이름 반환
String getParent()	부모 디렉터리(파일이 저장된 디렉터리) 이름 반환
String getPath()	전체 경로(디렉터리 이름 + 파일 이름) 반환
String[] list()	경로 내의 디렉터리와 파일 이름을 문자열 배열로 반환
File[] listFiles()	경로 내의 디렉터리와 파일 이름을 File 객체 배열로 반환
boolean mkdirs()	존재하지 않는 모든 하위 디렉터리 생성하고 성공유무를 true/false로 반환
boolean createNewFile()	새로운 파일을 생성하고 성공유무를 true/false로 반환
boolean delete()	파일이나 디렉터리를 삭제
boolean deleteOnExit()	JVM이 종료되면 파일이나 디렉터리를 삭제
boolean exists()	파일이나 디렉터리의 존재여부를 true/false로 반환

스트림(stream)

- 입력과 출력이 이루어지는 가상의 연결 통로
- 키보드, 모니터, 파일, 네트워크 등은 자바 프로그램과 스트림을 통해서 데이터를 주고 받음
- 입력이나 출력 중 하나만 처리할 수 있어서 입력 스트림과 출력 스트림으로 구분함
  (입력과 출력을 동시에 할 수 없음)



스트림 구분

- 바이트 스트림
  - ✓ 입출력 데이터를 바이트(byte) 단위로 처리
  - ✓ 이름이 InputStream/OutputStream으로 끝나는 클래스
  - ✓ 모든 파일 형식 처리 가능
    - 이미지 파일 입출력
    - 동영상 파일 입출력
    - 바이너리 파일 입출력

- 문자 스트림
  - ✓ 입출력 데이터를 문자 단위로 처리
  - ✓ 이름이 Reader/Writer로 끝나는 클래스
  - ✓ 문자로 된 파일 형식만 처리 가능
    - 텍스트 파일 입출력

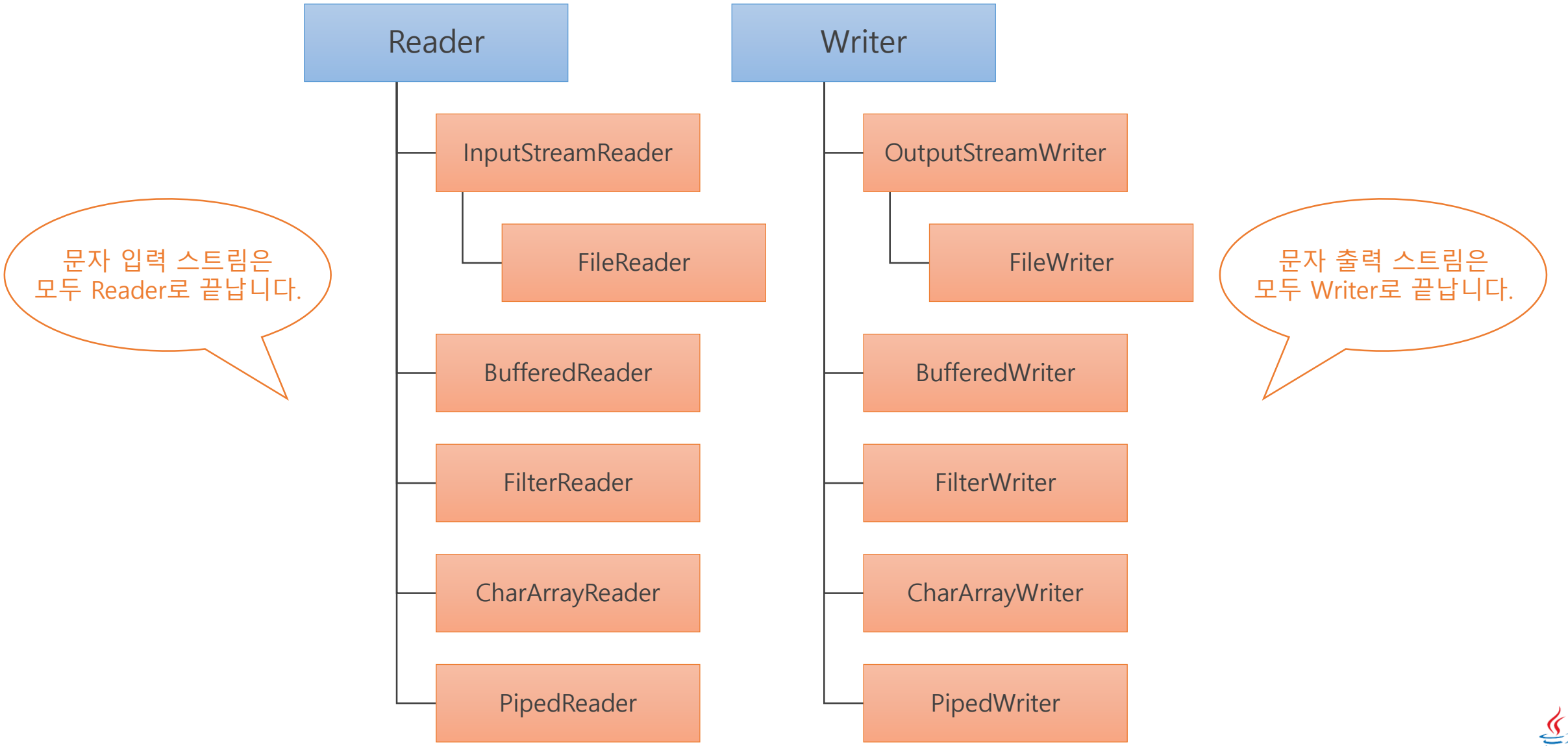
- 오직 영문으로 구성된 텍스트 파일은 바이트 스트림으로 처리할 수 있는가?
  - ✓ Yes. 영문 1글자는 1바이트 크기를 가지므로 바이트 스트림을 사용해도 영문 1글자씩 처리 가능
  - ✓ But. 한글은 1글자가 2~3바이트 크기를 가지므로 반드시 문자 스트림을 사용해야 함

문자 스트림

- 문자 스트림
  - ✓ 유니코드(2바이트) 문자를 대상으로 입출력을 수행함
  - ✓ 한글의 경우 2바이트 이상의 크기를 가지므로 반드시 문자 스트림을 사용해야 함
  - ✓ 텍스트 이외의 데이터(이미지, 동영상 등)는 처리할 수 없음
  - ✓ 모든 문자 입력 스트림은 Reader 클래스의 자식 클래스이고, 문자 출력 스트림은 Writer 클래스의 자식 클래스임

- 주요 문자 스트림 클래스
  - ✓ InputStreamReader / OutputStreamWriter
    - 바이트 스트림을 문자 스트림으로 변환
  - ✓ FileReader / FileWriter
    - 텍스트 파일 입출력
  - ✓ BufferedReader / BufferedWriter
    - 버퍼(Buffer)를 이용해 입출력 속도 향상
    - 여러 번 입출력 해야 할 데이터를 버퍼에 모아 두었다가 한 번에 처리

문자 스트림의 구조


Reader
Writer
InputStreamReader
FileReader
BufferedReader
FilterReader
CharArrayReader
PipedReader
OutputStreamWriter
FileWriter
BufferedWriter
FilterWriter
CharArrayWriter
PipedWriter
문자 입력 스트림은
모두 Reader로 끝납니다.
문자 출력 스트림은
모두 Writer로 끝납니다.

FileWriter

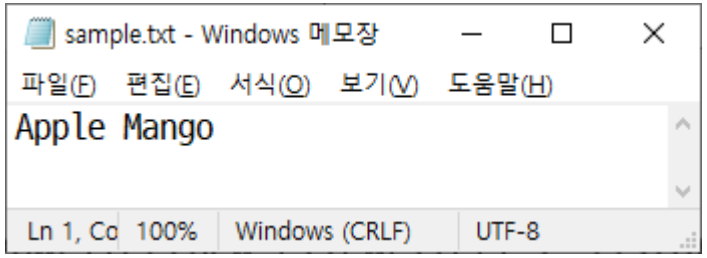
- FileWriter 클래스
  - ✓ 텍스트 파일 출력 스트림
  - ✓ 텍스트 파일을 만들 때 사용

✓주요 메소드

메소드	역할
void flush()	버퍼에 남은 데이터를 출력 스트림으로 강제로 보냄
void write(int c)	int c에 저장된 단일 문자를 파일에 저장
void write(char[] cbuf)	char 배열 cbuf에 저장된 모든 문자를 파일에 저장
void write(char[] cbuf, int off, int len)	char 배열 cbuf에 저장된 문자 중 인덱스 off부터 len개 문자만 파일에 저장
void write(String str)	String str에 저장된 전체 문자열을 파일에 저장
void write(String str, int off, int len)	String str에 저장된 문자열 중 인덱스 off부터 len개 문자만 파일에 저장
void close()	스트림 닫기

FileWriter 파일 생성 코드

```
/* sample.txt 생성하기 */

try {
    FileWriter writer = new FileWriter("C:\\storage\\sample.txt");
    int c = 'A';
    char[] cbuf = {'p', 'p', 'l', 'e'};
    String str = " Mango";
    writer.write(c);     // int 단위 출력 가능(1글자만 출력)
    writer.write(cbuf);  // char 배열 단위 출력 가능
    writer.write(str);   // String 단위 출력 가능
    writer.close();
} catch(IOException e) {
    e.printStackTrace();
}
```



sample.txt 생성 결과

FileReader

- FileReader 클래스
  - ✓ 텍스트 파일 입력 스트림
  - ✓ 텍스트 파일의 데이터를 읽어 들이는 경우 사용

✓주요 메소드

메소드	역할
int read()	단일 문자를 읽어서 반환 읽은 문자가 없으면 -1 반환
int read(char[] cbuf)	최대 char 배열 cbuf의 크기만큼의 문자를 읽어서 배열 cbuf에 저장 실제로 읽은 글자의 개수 반환 읽은 문자가 없으면 -1 반환
int read(char[] cbuf, int off, int len)	읽어 들인 문자를 char 배열 cbuf의 인덱스 off부터 len개만큼 저장 실제로 읽은 글자의 개수 반환 읽은 문자가 없으면 -1 반환
void close()	스트림 닫기

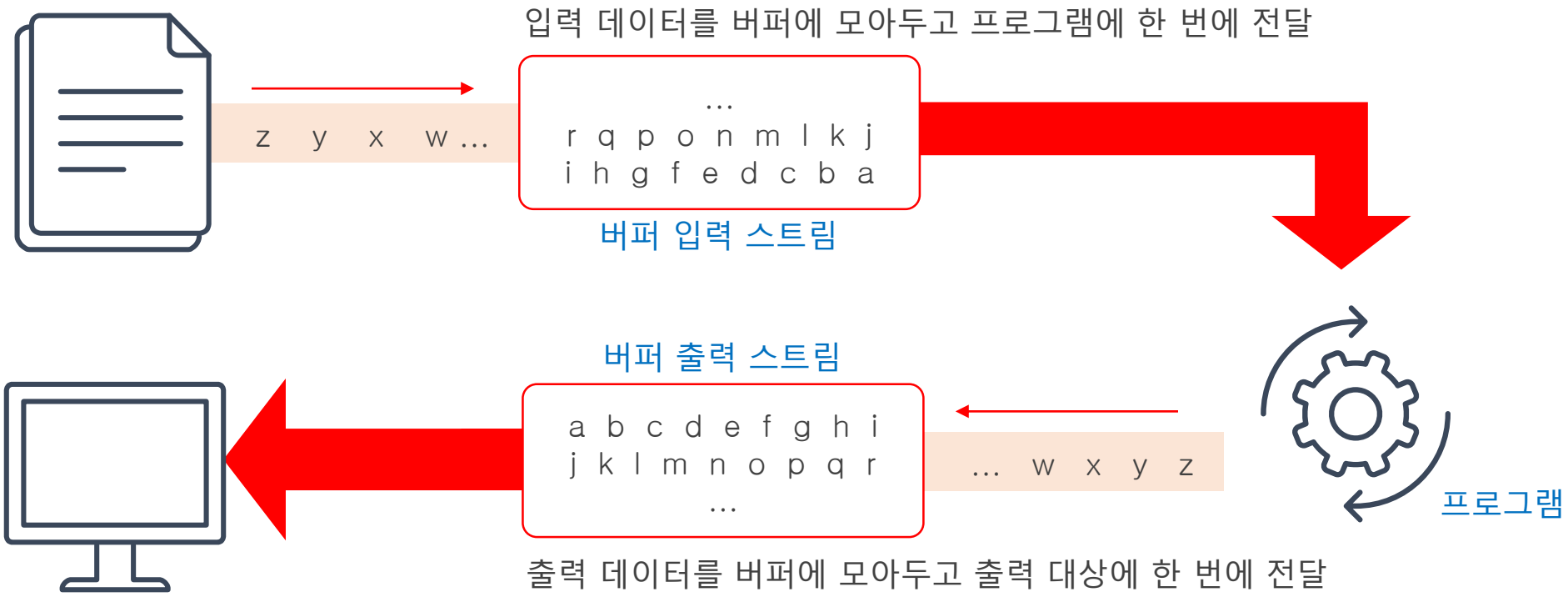
FileReader 파일 읽기 코드

```
/* sample.txt 전체 내용을 StringBuiler에 저장하기 */

try {
    FileReader reader = new FileReader("C:\\storage\\sample.txt");
    StringBuilder sb = new StringBuilder();
    int c;
    while((c = reader.read()) != -1) {  // read() 메소드는 파일이 끝나면 -1 반환
        sb.append((char)c);             // 1글자씩 읽어서 StringBuilder에 순서대로 저장
    }
    System.out.println(sb.toString());  // sample.txt 파일 내용을 화면에 출력
    reader.close();
} catch(IOException e) {
    e.printStackTrace();
}
```

1글자씩 읽기

버퍼 입출력

- 버퍼 입출력 동작 원리
  - ✓ 입출력(I/O) 횟수를 줄여서 입출력 성능을 개선하기 위해 버퍼를 사용



BufferedWriter

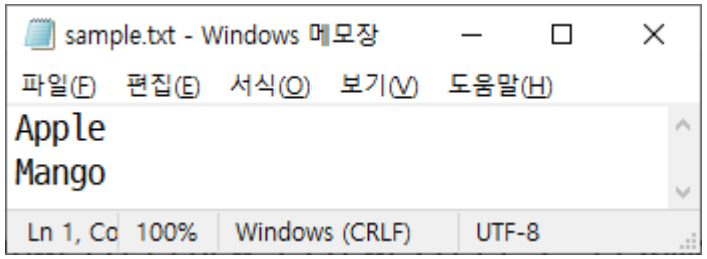
- BufferedWriter 클래스
  - ✓ 버퍼(Buffer, 임시기억장소)를 가지고 있는 문자 출력 스트림
  - ✓ 단독으로 사용하지 못하고 문자 출력 스트림과 함께 사용할 수 있는 보조 스트림
  - ✓ 출력 속도가 향상됨

✓주요 메소드

메소드	역할
void flush()	버퍼에 남은 데이터를 출력 스트림으로 강제로 보냄
void newLine()	줄 바꿈
void write(int c)	int c에 저장된 단일 문자를 파일에 저장
void write(char[] cbuf)	char 배열 cbuf에 저장된 모든 문자를 파일에 저장
void write(char[] cbuf, int off, int len)	char 배열 cbuf에 저장된 문자 중 인덱스 off부터 len개 문자만 파일에 저장
void write(String str)	String str에 저장된 전체 문자열을 파일에 저장
void write(String str, int off, int len)	String str에 저장된 문자열 중 인덱스 off부터 len개 문자만 파일에 저장
void close()	스트림 닫기

BufferedWriter 파일 생성 코드

```
/* sample.txt 생성하기 */

try {
    FileWriter writer = new FileWriter("C:\\storage\\sample.txt");
    BufferedWriter bw = new BufferedWriter(writer);
    String str = "Apple\nMango";
    bw.write(str);  // String 단위 출력 가능
    bw.close();
} catch(IOException e) {
    e.printStackTrace();
}
```



sample.txt 생성 결과

BufferedReader

- BufferedReader 클래스
  - ✓ 버퍼(Buffer, 임시기억장소)를 가지고 있는 문자 입력 스트림
  - ✓ 단독으로 사용하지 못하고 문자 입력 스트림과 함께 사용할 수 있는 보조 스트림
  - ✓ 입력 속도가 향상됨

✓주요 메소드

메소드	역할
int read()	단일 문자를 읽어서 반환 읽은 문자가 없으면 -1 반환
int read(char[] cbuf)	최대 char 배열 cbuf의 크기만큼의 문자를 읽어서 배열 cbuf에 저장 실제로 읽은 글자의 개수 반환 읽은 문자가 없으면 -1 반환
int read(char[] cbuf, int off, int len)	읽어 들인 문자를 char 배열 cbuf의 인덱스 off부터 len개만큼 저장 실제로 읽은 글자의 개수 반환 읽은 문자가 없으면 -1 반환
String readLine()	문자열을 라인(Line) 단위로 읽어서 반환 읽은 문자열이 없으면 null 반환
void close()	스트림 닫기

BufferedReader 파일 읽기 코드

```
/* sample.txt 전체 내용을 StringBuiler에 저장하기 */

try {
    FileReader reader = new FileReader("C:\\storage\\sample.txt");
    BufferedReader br = new BufferedReader(reader);
    StringBuilder sb = new StringBuilder();
    String line = null;
    while((line = br.readLine()) != null) {  // readLine() 메소드는 파일이 끝나면 null 반환
        sb.append(line);                     // 라인(Line) 단위로 읽어서 StringBuilder에 순서대로 저장
    }
    System.out.println(sb.toString());       // sample.txt 파일 내용을 화면에 출력
    br.close();
} catch(IOException e) {
    e.printStackTrace();
}
```

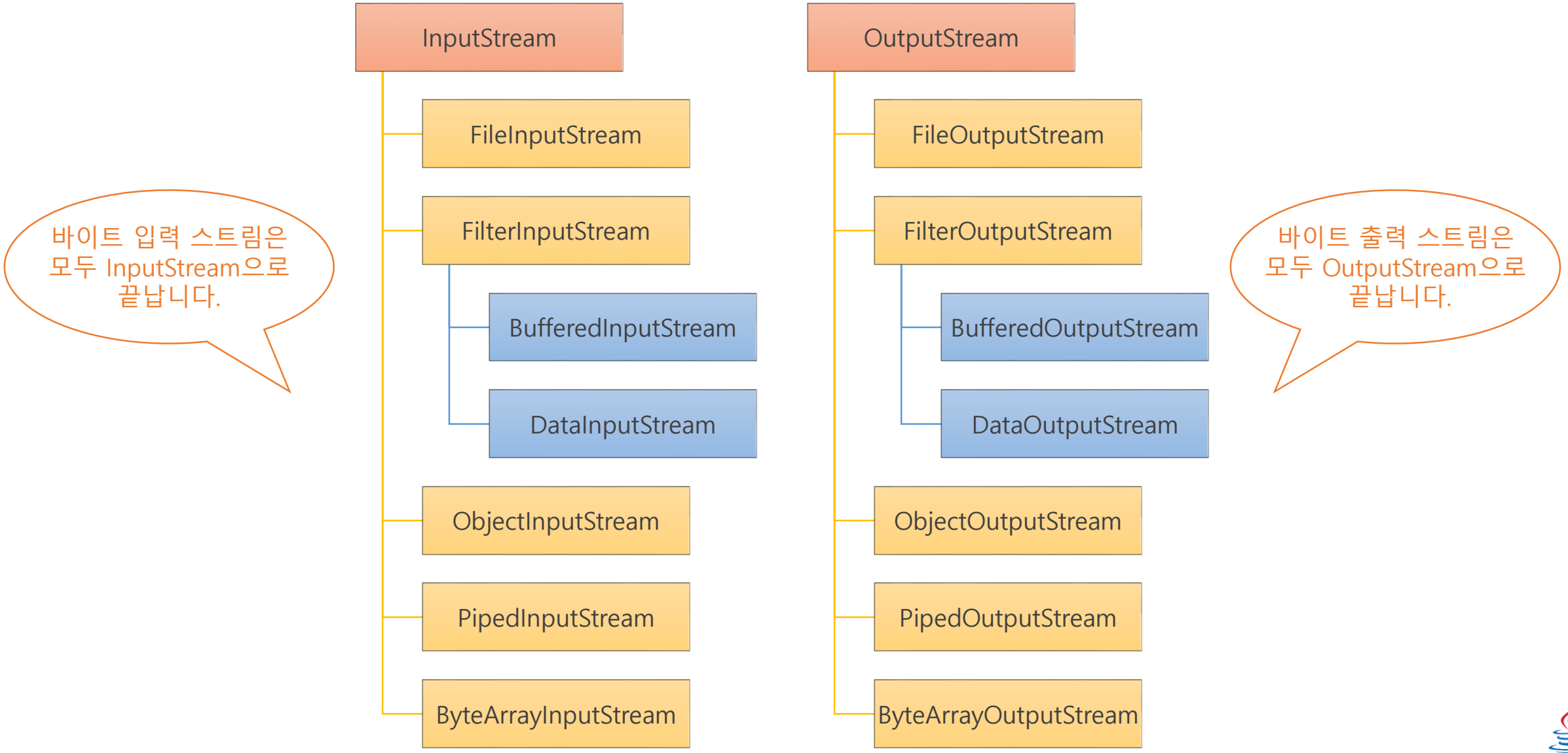
readLine() 메소드는
BufferedReader만 가능

바이트 스트림

- 바이트 스트림
  - ✓ 바이트 단위의 바이너리(binary) 데이터를 대상으로 입출력을 수행함
  - ✓ 한글의 경우 2바이트 이상의 크기를 가지므로 바이트 스트림을 사용하면 문제가 발생할 수 있음
  - ✓ 이미지, 동영상 등 바이너리 데이터와 자바의 변수/객체도 처리할 수 있음
  - ✓ 모든 바이트 입력 스트림은 InputStream 클래스의 자식 클래스이고, 바이트 출력 스트림은 OutputStream 클래스의 자식 클래스임

- 주요 바이트 스트림 클래스
  - ✓ FileInputStream / FileOutputStream
    - 바이너리 파일 입출력
  - ✓ DataInputStream / DataOutputStream
    - 자바 변수를 바이너리 값 그대로 입출력
  - ✓ ObjectInputStream / ObjectOutputStream
    - 자바 객체(인스턴스)를 바이너리 값 그대로 입출력
  - ✓ BufferedInputStream / BufferedOutputStream
    - 버퍼(Buffer)를 이용해 입출력 속도 향상
    - 여러 번 입출력 해야 할 데이터를 버퍼에 모아 두었다가 한 번에 처리

바이트 스트림의 구조


바이트 입력 스트림은 모두 InputStream으로 끝납니다.
InputStream
FileInputStream
FilterInputStream
BufferedInputStream
DataInputStream
ObjectInputStream
PipedInputStream
ByteArrayInputStream
OutputStream
FileOutputStream
FilterOutputStream
BufferedOutputStream
DataOutputStream
ObjectOutputStream
PipedOutputStream
ByteArrayOutputStream
바이트 출력 스트림은 모두 OutputStream으로 끝납니다.

FileOutputStream

- FileOutputStream 클래스
  - ✓ 바이너리 파일 출력 스트림
  - ✓ 바이너리 파일을 만들 때 사용

✓주요 메소드

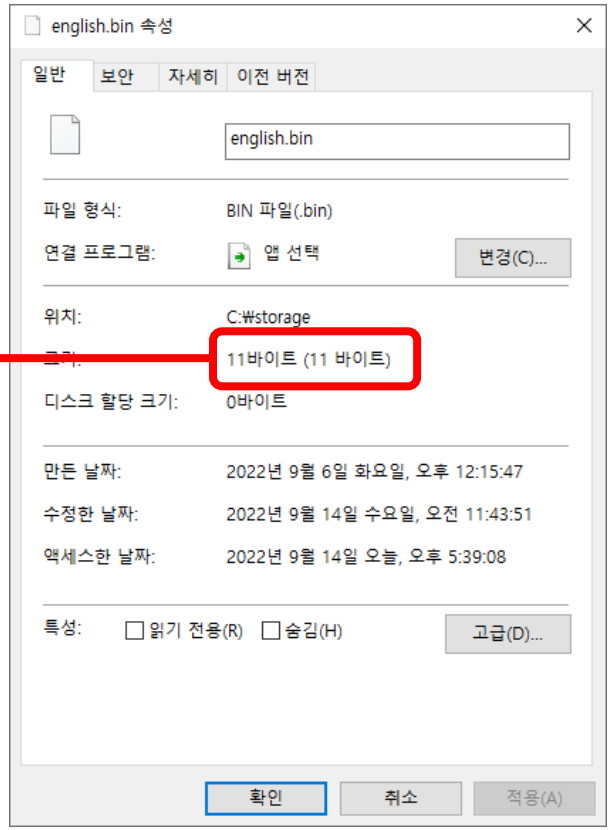
메소드	역할
void flush()	버퍼에 남은 데이터를 출력 스트림으로 강제로 보냄
void write(int b)	int b에 저장된 이진 데이터 1바이트를 파일에 저장
void write(byte[] b)	바이트 배열 b에 저장된 이진 데이터 전체를 파일에 저장
void write(byte[] b, int off, int len)	바이트 배열 b에 저장된 이진 데이터를 인덱스 off부터 len바이트만큼 파일에 저장
void close()	스트림 닫기

FileOutputStream 파일 생성 코드

```
/* english.bin 생성하기 */

try {
    FileOutputStream fout = new FileOutputStream("C:\\storage\\english.bin");
    int a = 'A';
    String str = "pple Mango";
    byte[] b = str.getBytes();    // getBytes() : String을 byte 배열로 변환
    fout.write(a);                // int 단위 출력
    fout.write(b);                // byte 배열 단위 출력
    fout.close();
} catch(IOException e) {
    e.printStackTrace();
}
```

int a에 저장된 이진 데이터('A') 1바이트
&
바이트 배열 b에 저장된 전체 데이터("pple Mango") 10바이트

int a에 저장된 'A'는
1바이트이므로
정상적으로 출력됨



english.bin 생성 결과

FileInputStream

- FileInputStream 클래스
  - ✓ 바이너리 파일 입력 스트림
  - ✓ 바이너리 파일을 읽을 때 사용

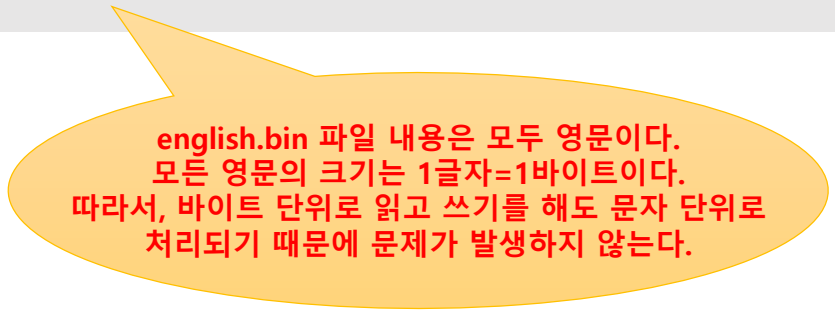
✓주요 메소드

메소드	역할
int read()	입력 스트림에서 바이트(byte) 데이터를 읽어 반환 읽은 데이터가 없으면 -1 반환
int read(byte[] b)	입력 스트림에서 최대 바이트 배열 b의 길이만큼 바이트(byte) 데이터를 읽어 바이트 배열 b에 저장 실제로 읽은 바이트 수를 반환 읽은 데이터가 없으면 -1 반환
int read(byte[] b, int off, int len)	읽어 들인 데이터를 byte 배열 b의 인덱스 off부터 len바이트만큼 저장 실제로 읽은 바이트 수를 반환 읽은 데이터가 없으면 -1 반환
void close()	스트림 닫기

Java

FileInputStream 파일 읽기 코드

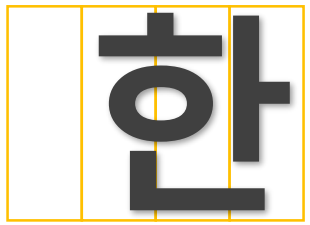
```
/* english.bin 전체 내용을 화면에 출력하기 */

try {
    FileInputStream fin = new FileInputStream("C:\\storage\\english.bin");
    int c;
    while((c = fin.read()) != -1) {
        System.out.print((char)c);
    }
    fin.close();
} catch(IOException e) {
    e.printStackTrace();
}
```



Java

바이트 스트림의 한글 처리

- 바이트 출력 스트림이 한글을 처리하지 못하는 이유



저장된 문자
(4바이트 중 3바이트 차지)

바이트 출력 스트림이
처리하는 1바이트

바이트 출력 스트림은
int c에 저장된 문자의 크기를
고려하지 않고 1바이트만 출력 처리함

- 바이트 입력 스트림이 한글을 처리하지 못하는 이유

6바이트에
저장된 문자

바이트 입력 스트림이
2바이트씩 읽은 경우

바이트 입력 스트림은
저장된 문자의 크기를 고려하지 않고
배열의 크기만큼 입력 처리함

BufferedOutputStream

- BufferedOutputStream 클래스
  - ✓ 버퍼(Buffer, 임시기억장소)를 가지고 있는 바이트 출력 스트림
  - ✓ 단독으로 사용하지 못하고 바이트 출력 스트림과 함께 사용할 수 있는 보조 스트림
  - ✓ 출력 속도가 향상됨

✓주요 메소드

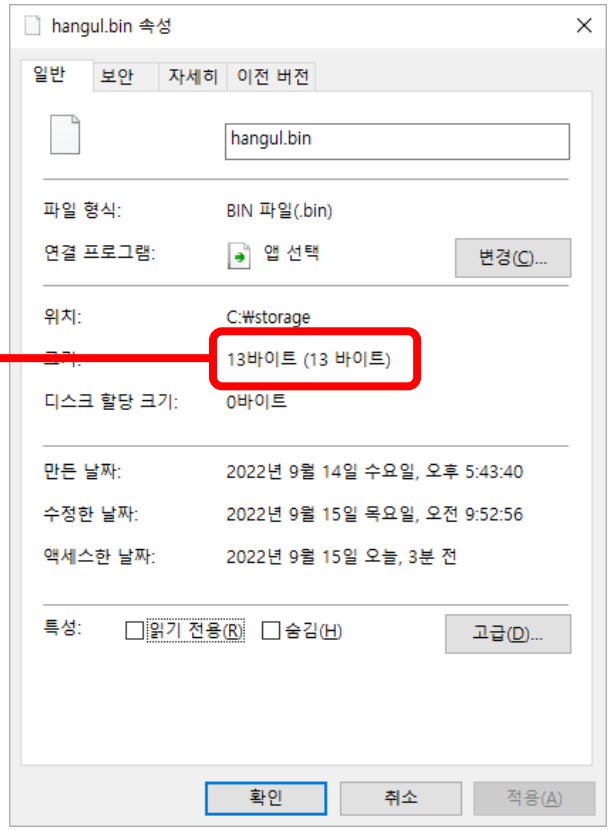
메소드	역할
void flush()	버퍼에 남은 데이터를 출력 스트림으로 강제로 보냄
void write(int b)	int b에 저장된 이진 데이터 1바이트를 파일에 저장
void write(byte[] b)	바이트 배열 b에 저장된 이진 데이터 전체를 파일에 저장
void write(byte[] b, int off, int len)	바이트 배열 b에 저장된 이진 데이터를 인덱스 off부터 len바이트만큼 파일에 저장
void close()	스트림 닫기

BufferedOutputStream 파일 생성 코드

```
/* hangul.bin 생성하기 */

try {
    FileOutputStream fout = new FileOutputStream("C:\\storage\\hangul.bin");
    BufferedOutputStream bout = new BufferedOutputStream(fout);
    int a = '안';
    String str = "녕하세요";
    byte[] b = str.getBytes();  // getBytes() : String을 byte 배열로 변환
    bout.write(a);              // int 단위 출력
    bout.write(b);              // byte 배열 단위 출력
    bout.close();
} catch(IOException e) {
    e.printStackTrace();
}
```

int a에 저장된 이진 데이터('안') 중 1바이트
&
바이트 배열 b에 저장된 전체 데이터("녕하세요") 12바이트

int a에 저장된 '안'은
2~3바이트인데 바이트 스트림은
그 중 1바이트만 사용하므로
정상적으로 출력이 안 됨!



hangul.bin 생성 결과

BufferedInputStream

- BufferedInputStream 클래스
  - ✓ 버퍼(Buffer, 임시기억장소)를 가지고 있는 바이트 입력 스트림
  - ✓ 단독으로 사용하지 못하고 바이트 입력 스트림과 함께 사용할 수 있는 보조 스트림
  - ✓ 입력 속도가 향상됨

✓주요 메소드

메소드	역할
int read()	입력 스트림에서 바이트(byte) 데이터를 읽어 반환 읽은 데이터가 없으면 -1 반환
int read(byte[] b)	입력 스트림에서 최대 바이트 배열 b의 길이만큼 바이트(byte) 데이터를 읽어 바이트 배열 b에 저장 실제로 읽은 바이트 수를 반환 읽은 데이터가 없으면 -1 반환
int read(byte[] b, int off, int len)	읽어 들인 데이터를 byte 배열 b의 인덱스 off부터 len바이트만큼 저장 실제로 읽은 바이트 수를 반환 읽은 데이터가 없으면 -1 반환
void close()	스트림 닫기

BufferedInputStream 파일 읽기 코드

```
/* hangul.bin 전체 내용을 화면에 출력하기 */

try {
    FileInputStream fin = new FileInputStream("C:\\storage\\hangul.bin");
    BufferedInputStream bin = new BufferedInputStream(fin);
    byte[] bytes = new byte[4];
    int readByte;
    while((readByte = bin.read(bytes)) != -1) {
        System.out.print(new String(bytes, 0, readByte));
    }
    bin.close();
} catch(IOException e) {
    e.printStackTrace();
}
```

일단!
이미 hangul.bin 파일은
잘못 만들어져 있기 때문에
정상적으로 읽지 못한다.

hangul.bin 파일이 정상이라고 가정해도,
hangul.bin 파일의 내용은 모두 한글이기 때문에
바이트 단위로 읽고 쓰기를 하면
정상적으로 처리되지 않고 깨질 수 있다.

Java

DataOutputStream

- DataOutputStream 클래스
  - ✓ 자바 기본 타입(Primitive Type) 및 String 값을 이진 데이터 그대로 출력할 수 있는 바이트 출력 스트림
  - ✓ 단독으로 사용하지 못하고 바이트 출력 스트림과 함께 사용할 수 있는 보조 스트림
  - ✓ DataInputStream 클래스를 이용해서 이진 데이터로 저장된 변수의 값을 그대로 꺼내서 사용할 수 있음

✓주요 메소드

메소드	역할
void writeBoolean(boolean v)	boolean v값을 1바이트로 스트림 출력
void writeByte(int v)	int v에 저장된 byte 값을 1바이트로 스트림 출력
void writeBytes(String s)	String s에 저장된 문자열을 바이트 배열로 스트림 출력
void writeChar(int v)	int v에 저장된 char 값을 2바이트로 스트림 출력
void writeDouble(double v)	double v에 저장된 값을 8바이트로 스트림 출력
void writeInt(int v)	int v에 저장된 값을 4바이트로 스트림 출력
void writeLong(long v)	long v에 저장된 값을 8바이트로 스트림 출력
void writeUTF(String str)	String str에 저장된 문자열을 UTF-8로 인코딩하여 스트림 출력

DataOutputStream 파일 생성 코드

```
/* sample.dat 생성하기 */

try {
    FileOutputStream fout = new FileOutputStream("C:\\storage\\sample.dat");
    DataOutputStream dout = new DataOutputStream(fout);
    boolean a = true;
    int b = 10;
    double c = 1.5;
    String d = "안녕하세요";
    dout.writeBoolean(a);  // boolean 그대로 출력 가능
    dout.writeInt(b);      // int 그대로 출력 가능
    dout.writeDouble(c);   // double 그대로 출력 가능
    dout.writeUTF(d);      // String 그대로 출력 가능
    dout.close();
} catch(IOException e) {
    e.printStackTrace();
}
```

DataInputStream

- DataInputStream 클래스
  - ✓ 이진 데이터를 자바 기본 타입(Primitive Type)이나 String으로 입력 받을 수 있는 바이트 입력 스트림
  - ✓ 단독으로 사용하지 못하고 바이트 입력 스트림과 함께 사용할 수 있는 보조 스트림
  - ✓ DataOutputStream 클래스를 이용해서 전달되거나 저장된 데이터를 읽을 수 있음

✓주요 메소드

메소드	역할
boolean readBoolean()	입력 스트림에 저장된 boolean 값을 읽음
byte readByte()	입력 스트림에 저장된 byte 값을 읽음
char readChar()	입력 스트림에 저장된 char 값을 읽음
double readDouble()	입력 스트림에 저장된 double 값을 읽음
int readInt()	입력 스트림에 저장된 int 값을 읽음
long readLong()	입력 스트림에 저장된 long 값을 읽음
String readUTF()	입력 스트림에 저장된 UTF-8로 인코딩 된 String 값을 읽음

DataInputStream 파일 생성 코드

```
/* sample.dat 읽기 */

try {
    FileInputStream fin = new FileInputStream("C:\\storage\\sample.dat");
    DataInputStream din = new DataInputStream(fin);
    boolean a = din.readBoolean();
    int b = din.readInt();
    double c = din.readDouble();
    String d = din.readUTF();
    System.out.println(a);   // true
    System.out.println(b);   // 10
    System.out.println(c);   // 1.5
    System.out.println(d);   // 안녕하세요
    din.close();
} catch(IOException e) {
    e.printStackTrace();
}
```