

3강

TCP 소켓 프로그래밍 II

1.멀티스레드 프로그래밍

- 1.멀티스레드 프로그래밍
- 2.다중처리 echo 서버 만들기
- 3.바이트스트림 vs 문자스트림

1. Thread(스레드) 란?

❑ 스레드(thread) : 프로그램의 실행 흐름

```
class Total {  
    public static void main(String args[]) {  
        int total = 0;  
        for (int cnt = 0; cnt < 3; cnt++)  
            total += cnt;  
        System.out.println(total);  
    }  
}
```

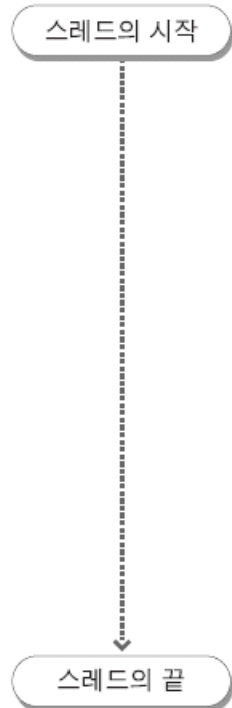
프로그램의
실행 흐름(스레드)



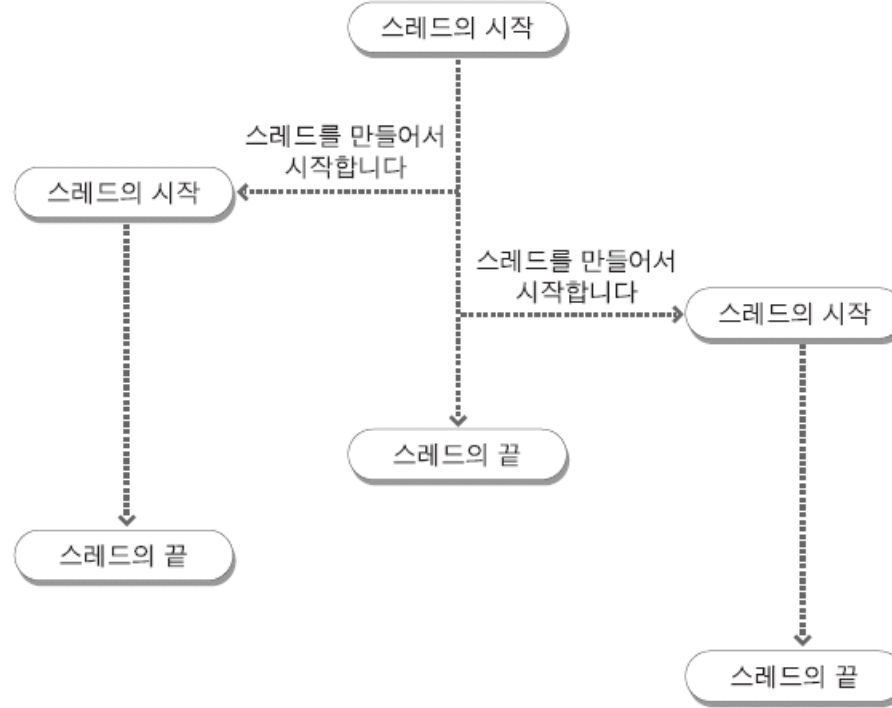
- 싱글스레드(single thread) 프로그램: 스레드가 하나뿐인 프로그램
- 멀티스레드(multi-thread) 프로그램: 스레드가 둘 이상인 프로그램

1. Thread(스레드) 란?

❑ 싱글 스레드(Single Thread) vs 멀티스레드(Multi-Thread)



a) 싱글스레드 프로그램의 실행 흐름



b) 멀티스레드 프로그램의 실행 흐름

2. 멀티스레드 프로그래밍

❑ 멀티스레드 프로그램의 작성 방법

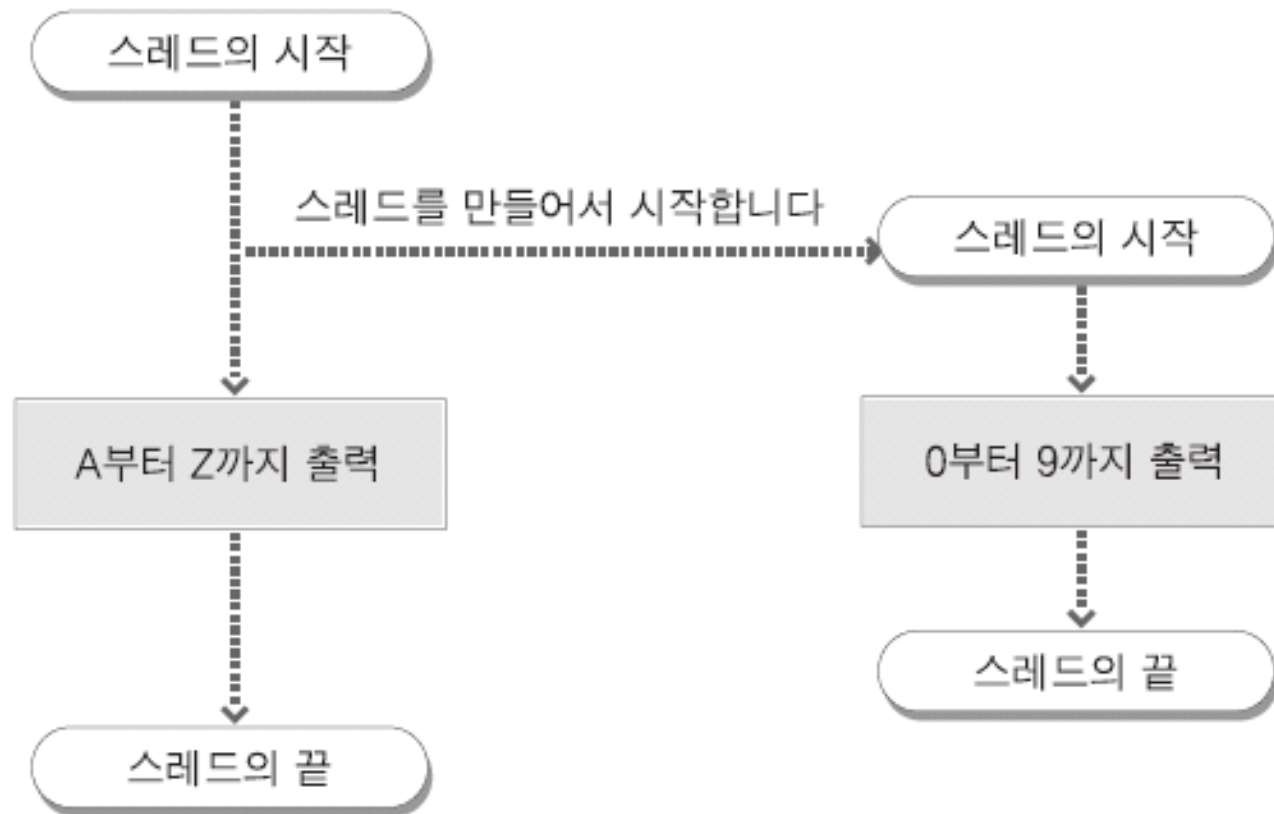
(1) `java.lang.Thread` 클래스를 사용하는 방법

(2) `java.lang.Runnable` 인터페이스를 이용하는 방법

2. 멀티스레드 프로그래밍

(1) java.lang.Thread 클래스를 사용하는 방법

[실습] MultithreadEx01.java



2. 멀티스레드 프로그래밍

(1) java.lang.Thread 클래스를 사용하는 방법

[실습] MultithreadEx01.java

main 메소드를 포함하는 클래스

```
1 class MultithreadEx01 {  
2     public static void main(String args[]) {  
3         Thread thread = new DigitThread();    // 스레드를 생성  
4         thread.start();                        // 스레드를 시작  
5         for (char ch = 'A'; ch <= 'Z'; ch++)  
6             System.out.print(ch);  
7     }  
8 }
```

숫자를 출력하는 스레드 클래스

```
1 class DigitThread extends Thread {  
2     public void run() {  
3         for (int cnt = 0; cnt < 10; cnt++)  
4             System.out.print(cnt);  
5     }  
6 }
```

실행 결과를 예측하고 확인한다.

6

2. 멀티스레드 프로그래밍

(1) java.lang.Thread 클래스를 사용하는 방법

[실습] MultithreadEx01.java

Thread.sleep(1000) 메소드를 사용해서 각각의 Thread를 루프 안에서 1초 동안 쉬면서 출력하게 했을 때, 실행 결과를 예측하고 확인한다.

2. 멀티스레드 프로그래밍

(1) java.lang.Thread 클래스를 사용하는 방법

[실습] MultithreadEx02.java

다음 코드를 참고해서 4개의 스레드가 병렬로 실행하는 멀티스레드 프로그램을 작성하고 테스트 한다.

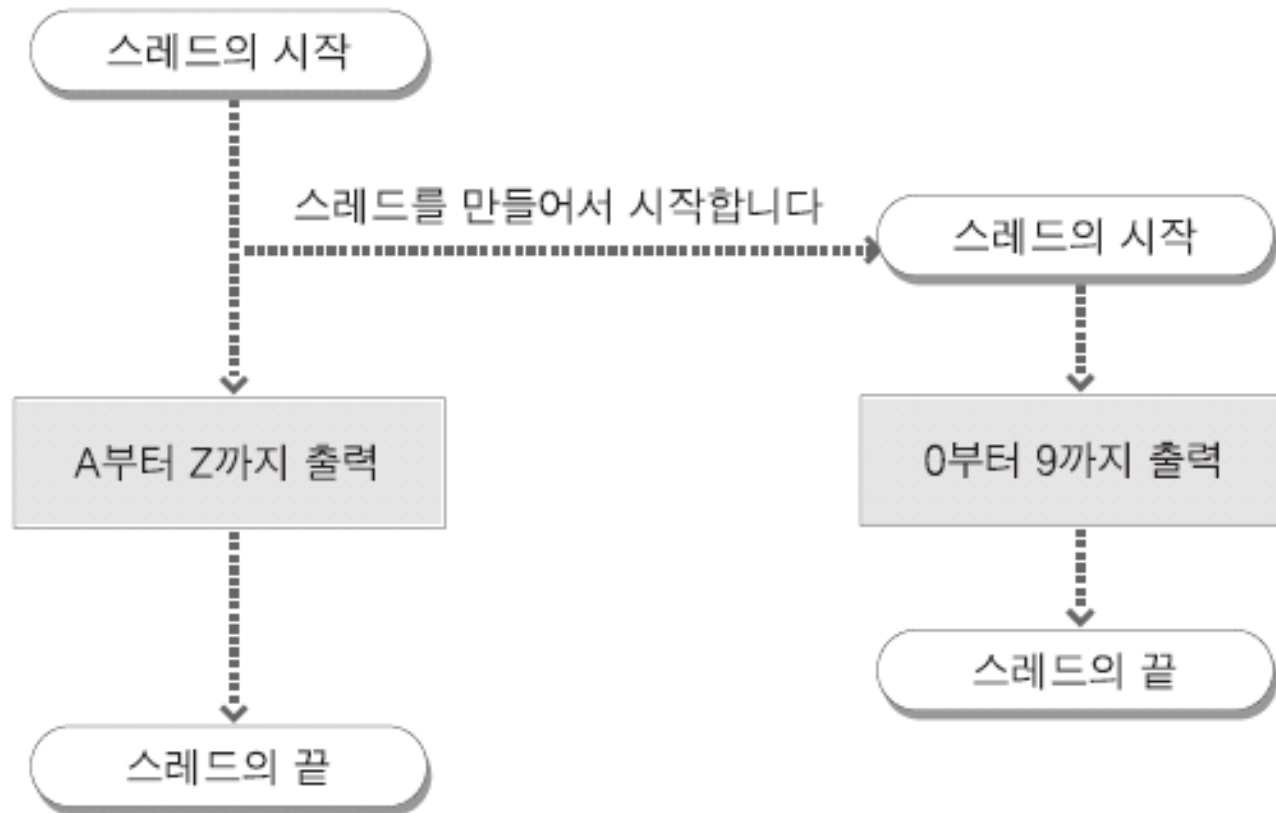
```
1 class MultithreadEx02 {  
2     public static void main(String args[]) {  
3         Thread thread1 = new DigitThread();  
4         Thread thread2 = new DigitThread();  
5         Thread thread3 = new AlphabetThread();  
6         thread1.start();  
7         thread2.start();  
8         thread3.start();  
9     }  
10 }
```

3개의 스레드를
생성해서 시작

2. 멀티스레드 프로그래밍

(2) java.lang.Runnable 인터페이스를 이용하는 방법

[실습] MultithreadEx03.java



2. 멀티스레드 프로그래밍

(2) java.lang.Runnable 인터페이스를 이용하는 방법

[실습] MultithreadEx03.java

main 메소드를 포함하는 클래스

```
1 class MultithreadEx04 {
2     public static void main(String args[]) {
3         Thread thread = new Thread( new DigitRunnableImpl() ); // 스레드를 생성
4         thread.start(); // 스레드를 시작
5         for (char ch = 'A'; ch <= 'Z'; ch++)
6             System.out.print(ch);
7     }
8 }
```

숫자를 출력하는 클래스

```
1 public class DigitRunnableImpl implements Runnable {
2     @Override
3     public void run() {
4         for( int i = 0; i < 9; i++ ) {
5             System.out.print( i );
6         }
7     }
8 }
```

2. 멀티스레드 프로그래밍

(2) java.lang.Runnable 인터페이스를 이용하는 방법

[실습] MultithreadEx03.java

이미 구현해 놓은 일반 클래스

```
1 public class Alphabet {  
2     public void print() {  
3         for( char c = 'A'; c <= 'Z'; c++ ) {  
4             System.out.print( c );  
5         }  
6     }  
7 }
```

Alphabet 객체의 print() 메소드를 스레드에서 실행해야 할 때
(단 Alphabet 클래스는 변경하지 말아야 한다.)

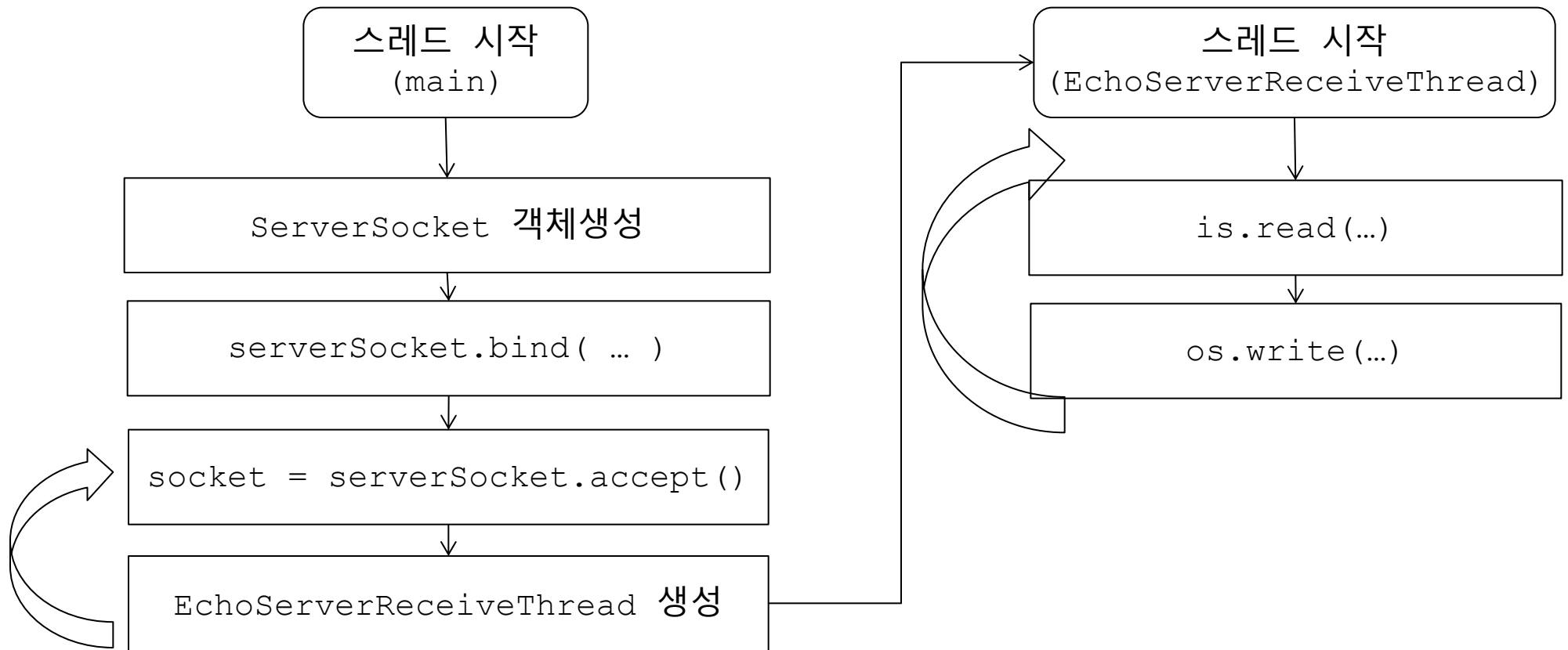
3강

TCP 소켓 프로그래밍 II

2.다중처리 echo 서버 만들기

- 1.멀티스레드 프로그래밍
- 2.다중처리 echo 서버 만들기
- 3.바이트스트림 vs 문자스트림

1. Thread 설계(실행흐름)



2.과제

다음코드를 참고해서 다중처리 **EchoServer**를 완성한다.

```
// 소켓 생성
serverSocket = new ServerSocket();

// binding
serverSocket.bind( new InetSocketAddress( InetAddress.getLocalHost().getHostAddress(), PORT ) );

while( true ) {
    // 연결 요청 기다림
    System.out.println( "[서버] 연결 기다림");
    Socket socket = serverSocket.accept();

    Thread thread = new EchoServerReceiveThread( socket );
    thread.start();
}
```

3강

TCP 소켓 프로그래밍 II

3.바이트스트림 vs 문자스트림

- 1.멀티스레드 프로그래밍
- 2.다중처리 echo 서버 만들기
- 3.바이트스트림 vs 문자스트림

1. 스트림(Stream) 이란?

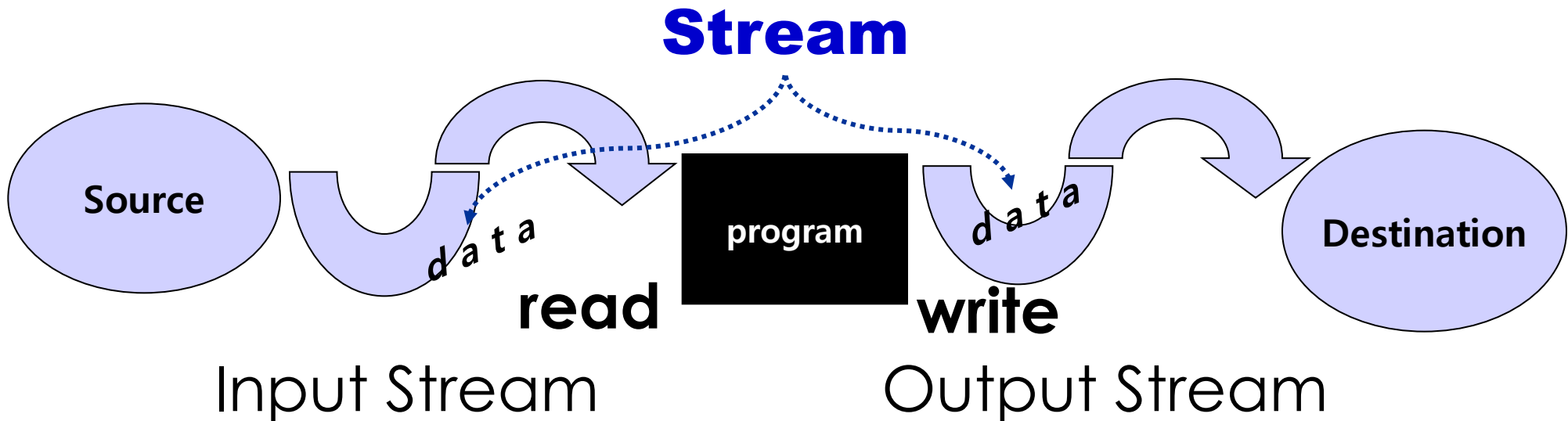
- ❑ 프로그램과 I/O 객체를 연결하여 데이터를 소스에서 전달하거나 목적지로 수신하는 길(Path) 소스: 키보드 및 스크린을 포함한 파일, Network Connections, 다른 프로그램 등
- ❑ 한 방향, 즉 일방통행 (Unidirectional)으로 프로그램과 I/O 객체를 연결

Ex) InputStream: 데이터를 읽어 들이는 객체 ,

OutputStream: 데이터를 써서 내보내는 객체

System.in: 프로그램과 키보드 연결

System.out: 프로그램과 스크린을 연결



2. 스트림 종류 - 기본 스트림

□ 바이트 스트림(Byte stream)

- 1과 0으로 구성된 **Binary** 데이터의 입출력 처리를 위한 스트림 **Ex)** 이미지, 사운드 등

□ 문자 스트림(Character stream)

- 문자, 텍스트 형태의 데이터 입출력 처리를 위한 스트림 **Ex)** 단순 텍스트, 웹 페이지, 키보드 입력 등
- 문자 각각 1 **Byte**의 아스키코드 (**ASCII Code**) 할당

□ JAVA I/O 클래스

- **java.io** 패키지에 I/O를 위한 4개의 추상 클래스 **Reader, Writer, InputStream, OutputStream** 정의
- 해당 클래스에서 상속받아 파일에 데이터를 읽고 씀

InputStream	Reader
abstract int read() int read(byte [] b) int read(byte [] b, int off, int len)	int read() int read(char [] cbuf) abstract int read(char [] cbuf, int off, int len)
OutputStream	Writer
abstract void write(int b) void write(byte [] b) void write(byte [] b, int off, int len)	void write(int c) void write(char [] cbuf) abstract void write(char [] cbuf, int off, int len) void write(String str) void write(String str, int off, int len)

3. 보조 스트림

❑ InputStreamReader와 OutputStreamWriter

- 바이트기반스트림을 문자기반스트림처럼 쓸 수 있게 해준다.
- 인코딩(encoding)을 변환하여 입출력할 수 있게 해준다.
- 콘솔(console, 화면)로부터 라인단위로 입력받기

```
InputStreamReader isr = new InputStreamReader(System.in);  
BufferedReader br = new BufferedReader(isr);  
String line = br.readLine();
```

- 인코딩 변환하기

```
FileInputStream fis = new FileInputStream("korean.txt");  
InputStreamReader isr = new InputStreamReader(fis, "KSC5601");
```

4. 보조스트림

❑ BufferedReader와 BufferedWriter

- 입출력 효율을 높이기 위해 버퍼(char[])를 사용하는 보조스트림
- 라인(line)단위의 입출력이 편리하다.

`String readLine()` - 한 라인을 읽어온다. (BufferedReader의 메서드)

`void newLine()` - '라인 구분자(개행문자)'를 출력한다. (BufferedWriter의 메서드)

5. 과제

다음코드를 참고해서 **EchoServer** 바이트 스트림을 문자 스트림으로 바꿉니다.

```
bufferedReader = new BufferedReader( new InputStreamReader( socket.getInputStream(), "UTF-8" ) );  
printWriter = new PrintWriter( socket.getOutputStream() );
```