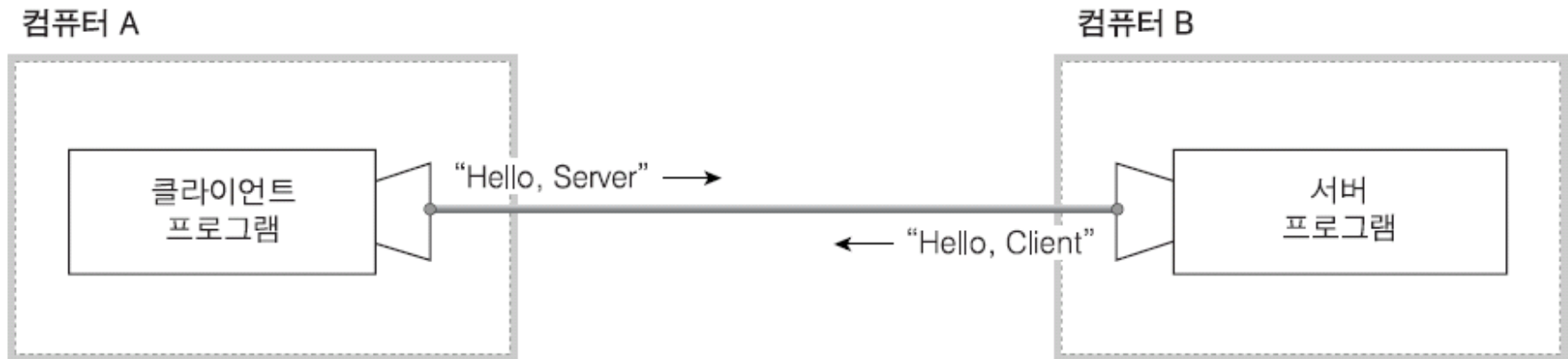


에코 (Echo)

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- 지금부터 작성할 예제가 하는 일
 - Echo Client/Echo Server



- 패키지명 : echo
- 프로그램 : ClientExample1, ServerExample1

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 클라이언트 : 바이트 단위 전송

```
import java.io.*;
import java.net.*;
class ClientExample1 {
    public static void main(String[] args) {
        try(Socket socket = new Socket("127.0.0.1", 10000)) {
            InputStream in = socket.getInputStream();    // 수신(입력) 스트림
            OutputStream out = socket.getOutputStream(); // 전송(출력) 스트림

            String str = "Hello, Server";
            out.write(str.getBytes());    // 메시지 전송
            out.flush();

            byte arr[] = new byte[100]; // 수신 버퍼
            in.read(arr);    // 메시지 수신
            System.out.println(new String(arr));
        }
        catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 서버 : 바이트 단위 전송

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

class ServerExample1 {
    public static void main(String[] args) {
        Socket socket=null;
        // 서버 소켓 생성
        try(ServerSocket serverSocket = new ServerSocket(10000);) {
            socket = serverSocket.accept(); // 접속 대기

            // 수신(입력) 스트림
            InputStream in = socket.getInputStream();

            // 전송(출력) 스트림
            OutputStream out = socket.getOutputStream();
```

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 서버 : 바이트 단위 전송

```
byte arr[] = new byte[100];
in.read(arr);    // 메시지 수신
System.out.println(new String(arr));

String str = "Hello, Client";
out.write(str.getBytes()); // 메시지 전송
out.flush();

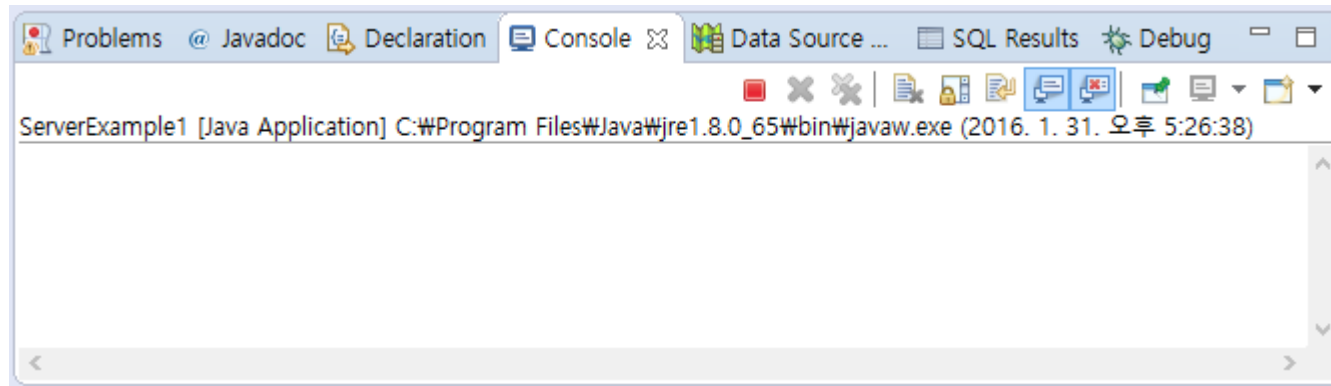
} catch (Exception e) {
    System.out.println(e.getMessage());
} finally {
    try { socket.close(); } catch (Exception e) { }
}

}
}
```

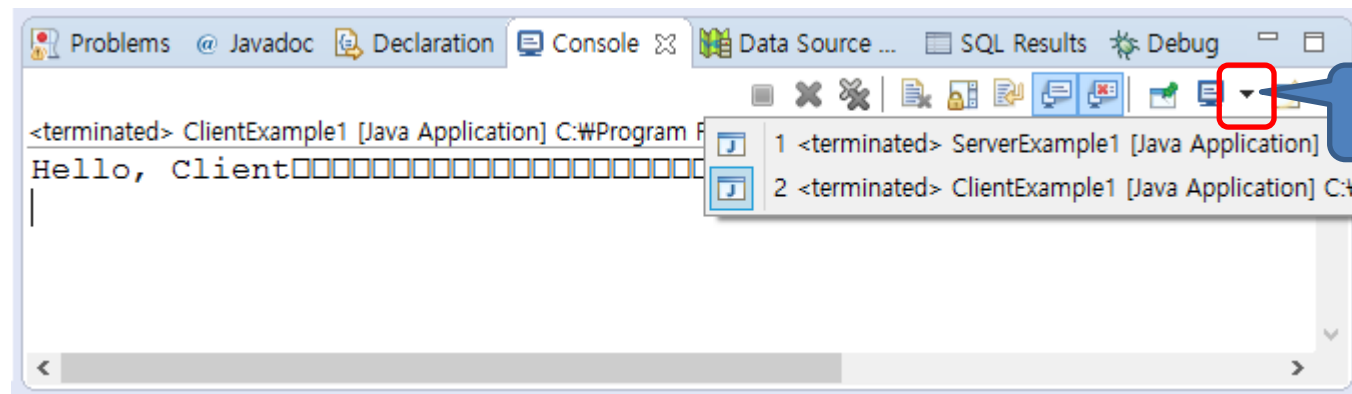
TCP/IP 통신 프로그램 작성

❖ 실행

- 이클립스에서 2개의 어플리케이션 동시에 실행하기
- 먼저 서버 ServerExample1을 실행

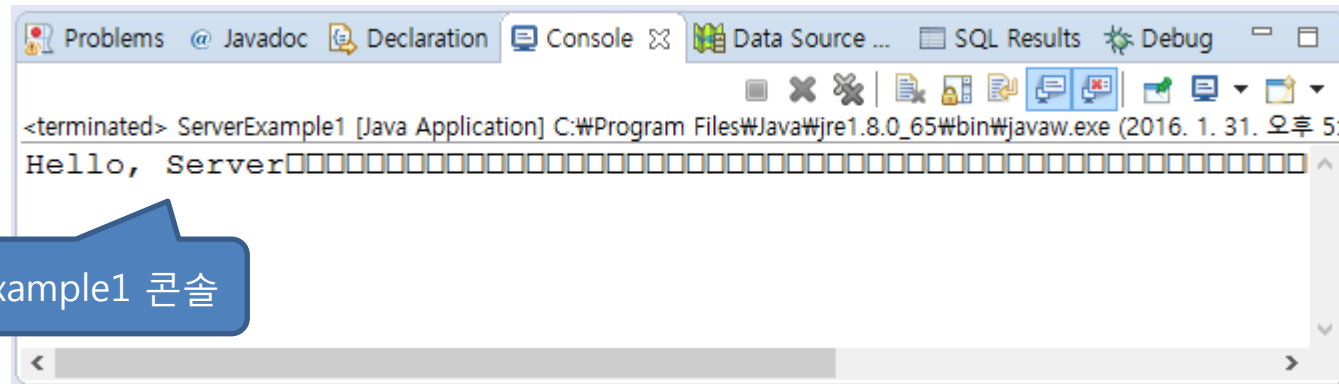
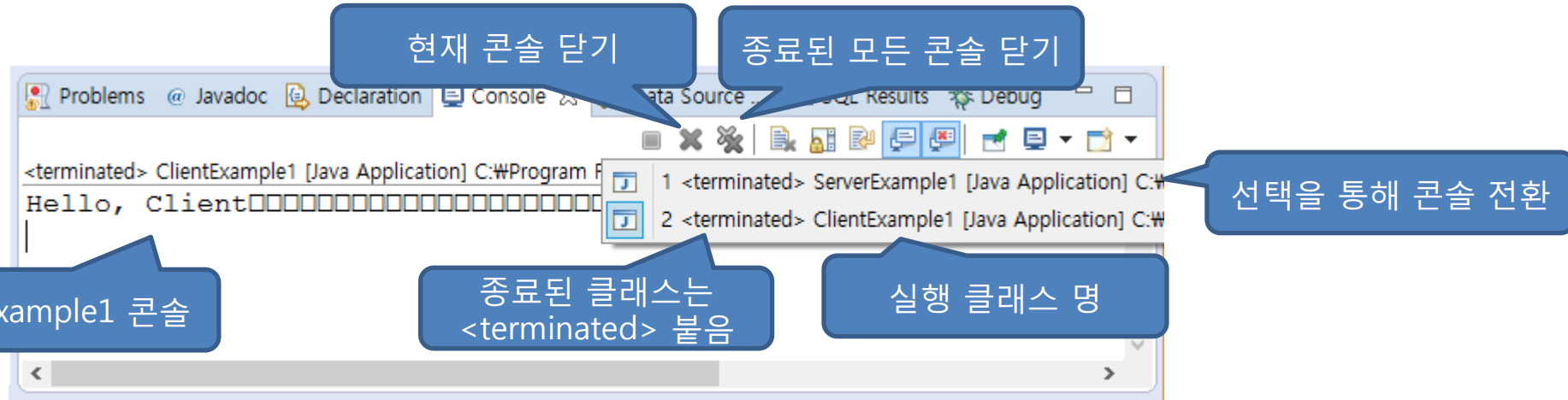


- 이 상태에서 ClientExample1을 실행



TCP/IP 통신 프로그램 작성

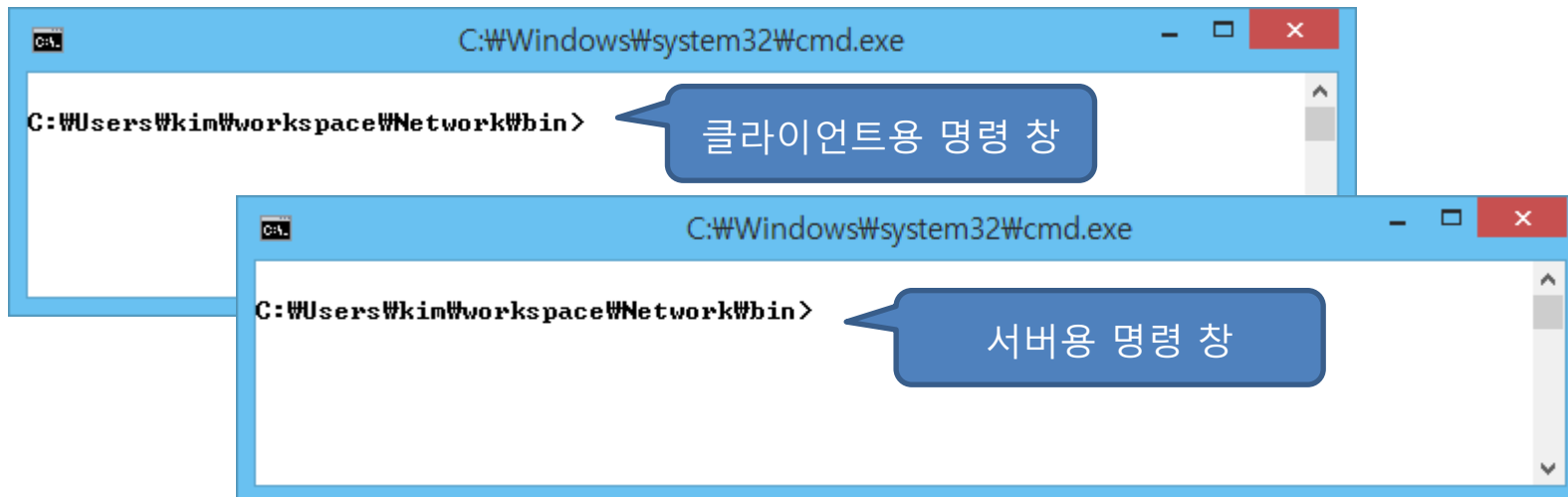
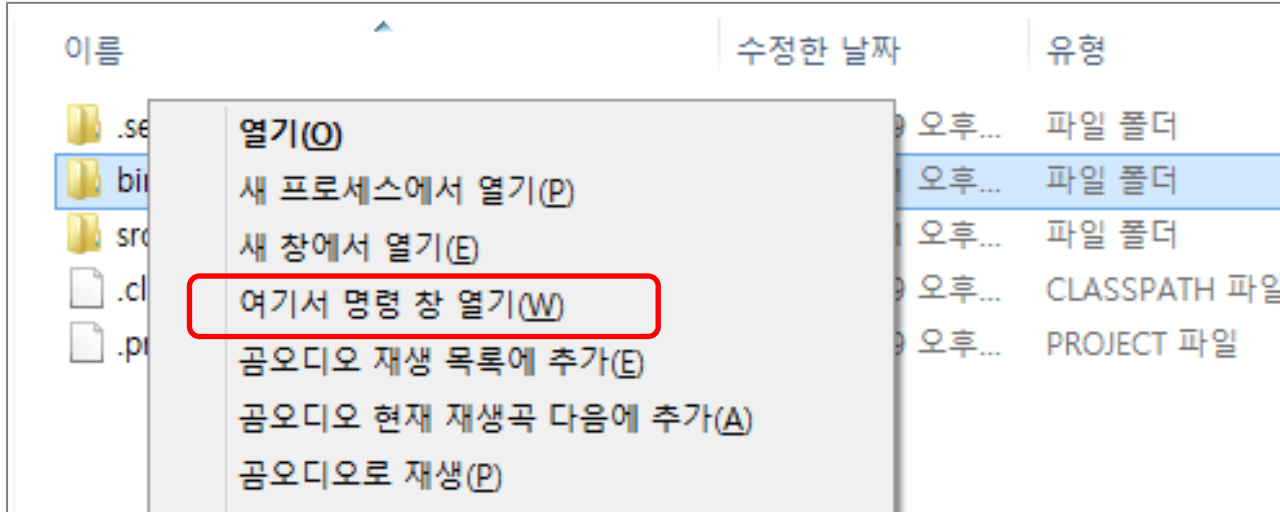
❖ 실행



TCP/IP 통신 프로그램 작성

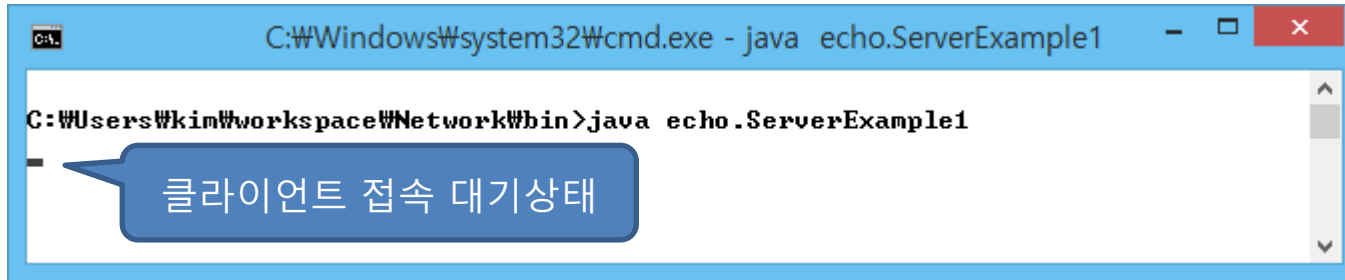
❖ 명령창에서 실행하기

- 프로젝트의 bin 디렉토리 선택 > Shift + 마우스 우측버튼 > 여기서 명령 창 열기



TCP/IP 통신 프로그램 작성

❖ 서버 실행 : java echo.ServerExample1

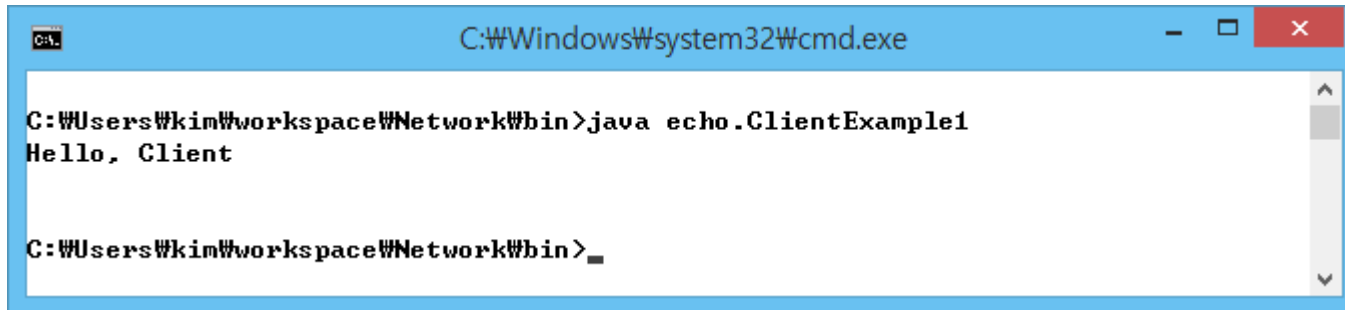


```
C:\Windows\system32\cmd.exe - java echo.ServerExample1

C:\Users\Wkin\workspace\Network\bin>java echo.ServerExample1

클라이언트 접속 대기상태
```

❖ 클라이언트 실행 : java echo.ClientExample1

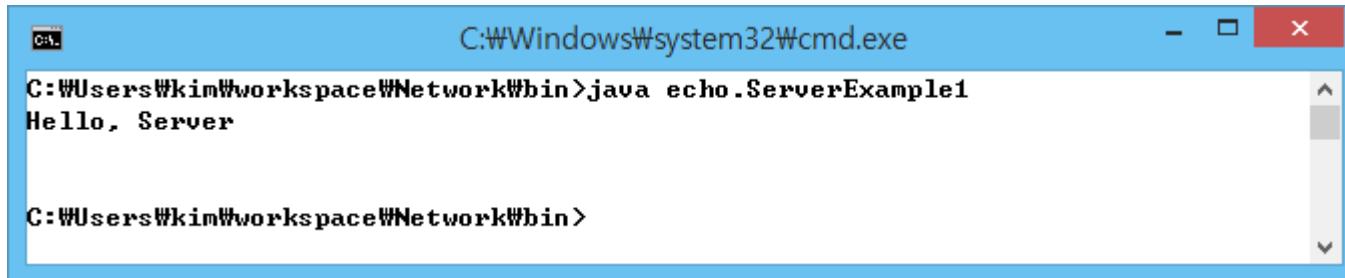


```
C:\Windows\system32\cmd.exe

C:\Users\Wkin\workspace\Network\bin>java echo.ClientExample1
Hello, Client

C:\Users\Wkin\workspace\Network\bin>
```

❖ 클라이언트 접속 후 서버 창



```
C:\Windows\system32\cmd.exe

C:\Users\Wkin\workspace\Network\bin>java echo.ServerExample1
Hello, Server

C:\Users\Wkin\workspace\Network\bin>
```

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- 바이트 스트림을 문자 스트림으로 바꾸는 방법

- InputStream 객체를 가지고 InputStreamReader 객체를 만들 수 있습니다.


```
InputStreamReader reader = new InputStreamReader(in);
```



InputStream 객체

- OutputStream 객체를 가지고 PrintWriter 객체를 만들 수 있습니다.

```
PrintWriter writer = new PrintWriter(out);
```



OutputStream 객체

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 클라이언트 : Reader/Writer를 통한 문자 단위 전송 및 수신

```
public class EchoClient {  
    public static void main(String[] args) {  
  
        try(Socket socket = new Socket("127.0.0.1", 10000)) {  
            // 수신용 Reader 만들기  
            BufferedReader r = new BufferedReader(  
                new InputStreamReader(socket.getInputStream()));  
            // 전송용 Writer 만들기  
            PrintWriter w = new PrintWriter(  
                socket.getOutputStream());  
  
            Scanner s = new Scanner(System.in);  
            System.out.print("문자열 입력> ");  
            String line = s.nextLine(); // 전송할 문자열 입력  
  
            // 메시지 전송  
            w.println(line);  
            w.flush();  
        }  
    }  
}
```

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 클라이언트 : Reader/Writer를 통한 문자 단위 전송 및 수신

```
// 메시지 수신
String receiveLine = r.readLine();
System.out.println("수신 메시지 : " + receiveLine);

} catch (Exception e) {
    e.printStackTrace();
}
}
```

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 서버 : Reader/Writer를 통한 문자 단위 전송 및 수신

```
:
import java.net.ServerSocket;
import java.net.Socket;

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

        Socket socket = null;
        try(ServerSocket server = new ServerSocket(10000)) {
            socket = server.accept();

            BufferedReader r = new BufferedReader(
                new InputStreamReader(socket.getInputStream()));

            PrintWriter w = new PrintWriter(
                socket.getOutputStream());
        }
    }
}
```

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 서버 : Reader/Writer를 통한 문자 단위 전송 및 수신

```
// 메시지 수신
String line = r.readLine();
System.out.println("서버 수신 메시지 : " + line);

// 메시지 전송
w.println(line);
w.flush();

} catch (Exception e) {
    e.printStackTrace();
} finally {
    try {socket.close();} catch (Exception e) {}
}
}
```

TCP/IP 통신 프로그램 작성

❖ Echo 서버 프로그램의 개선

- 앞의 echo 서버는 1번만 서비스 함
- 반복문을 이용해서 계속 서비스 하기(정상 종료 불가)

```
Socket socket = null;
try(ServerSocket server = new ServerSocket(10000)) {
    while(true) {
        socket = server.accept();
        :
        socket.close();
    }
} catch (Exception e) {
    e.printStackTrace();
}
```

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 서버2 : 반복문을 통한 접속 처리

```
public class EchoServer2 {
    public static void main(String[] args) {
        Socket socket = null;
        try(ServerSocket server = new ServerSocket(10000)) {
            while(true) {
                socket = server.accept();
                BufferedReader r = new BufferedReader(
                    new InputStreamReader(socket.getInputStream()));

                PrintWriter w = new PrintWriter(
                    socket.getOutputStream());

                String line = r.readLine();
                System.out.println("서버 수신 메시지 : " + line);

                w.println(line);
                w.flush();
                socket.close();
            } // end of while
        }
    }
}
```


TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

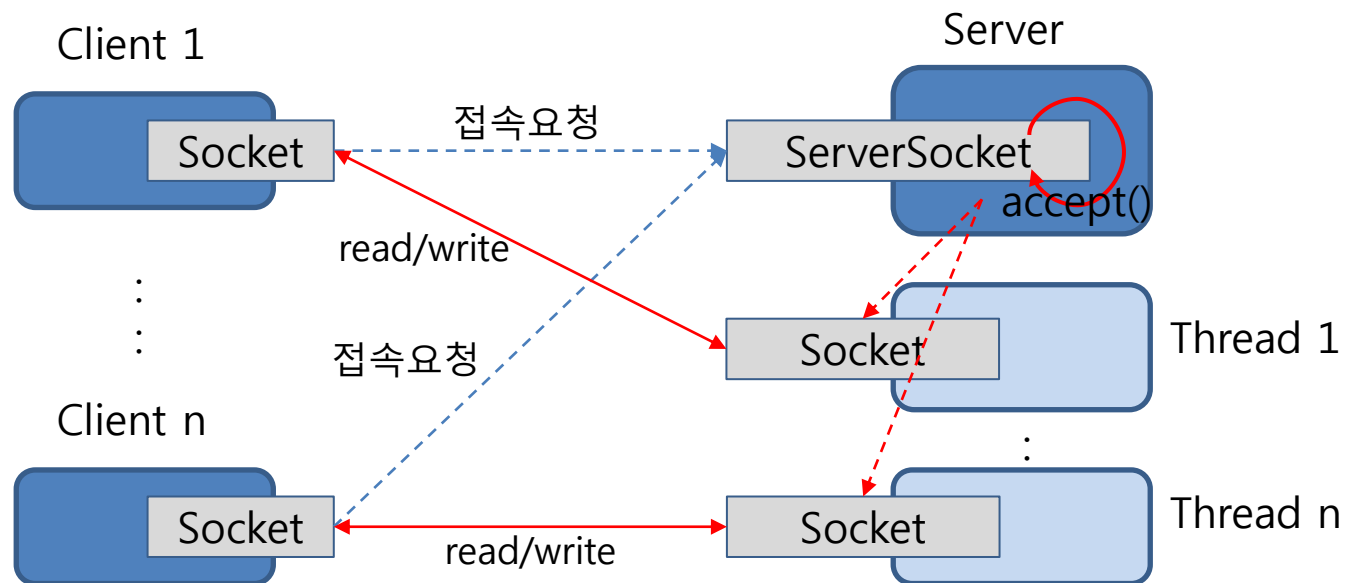
- Echo 서버2 : 반복문을 통한 접속 처리

```
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
}
```

TCP/IP 통신 프로그램 작성

❖ Echo 서버 프로그램의 개선

- 앞의 echo 서버는 동시성 지원하지 않음
 - 동시 접속자가 있을 경우 1개만 서비스되고 나머지는 대기
 - 스레드를 통해 동시성 지원
 - 스레드가 클라이언트와 통신



TCP/IP 통신 프로그램 작성

❖ Echo 서버 프로그램의 개선

- 스레드를 이용한 통신 골격

```
Socket socket = null;
try(ServerSocket server = new ServerSocket(10000)) {
    while(true) {
        socket = server.accept();
        WorkThread work = new WorkThread(socket);
        work.start();
    }
} catch (Exception e) {
    e.printStackTrace();
}
```

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 서버3 : 스레드를 통한 **extends Thread**작업

```
public class WorkThread extends Thread{
    private Socket socket;

    public WorkThread(Socket socket) { // 생성자를 통해 클라이언트 socket 받음
        this.socket = socket;
    }

    public void run() {
        try {
            BufferedReader r = new BufferedReader(
                new InputStreamReader(socket.getInputStream()));

            PrintWriter w = new PrintWriter(
                socket.getOutputStream());
```

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 서버3 : 스레드를 통한 작업

```
// 메시지 수신
String line = r.readLine();
System.out.println("서버 수신 메시지 : " + line);

// 메시지 전송
w.println(line);
w.flush();

} catch (IOException e) {
    System.out.println(e);
} finally {
    try {socket.close();} catch(IOException e) {System.out.println(e);}
}
}
```

TCP/IP 통신 프로그램 작성

❖ Echo 클라이언트/서버 프로그램 작성

- Echo 서버3 : 스레드를 통한 작업

```
public class EchoServer3 {  
    public static void main(String[] args) {  
        Socket socket = null;  
        try(ServerSocket server = new ServerSocket(10000)) {  
            while(true) {  
                socket = server.accept();  
                WorkThread work = new WorkThread(socket);  
                work.start();  
            } // end of while  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
}
```