Programming Language/Java

[JAVA] 소켓(socket) 프로그래밍

kyo kyo 2021. 1. 20. 20:54 수정 삭제

소켓 프로그래밍

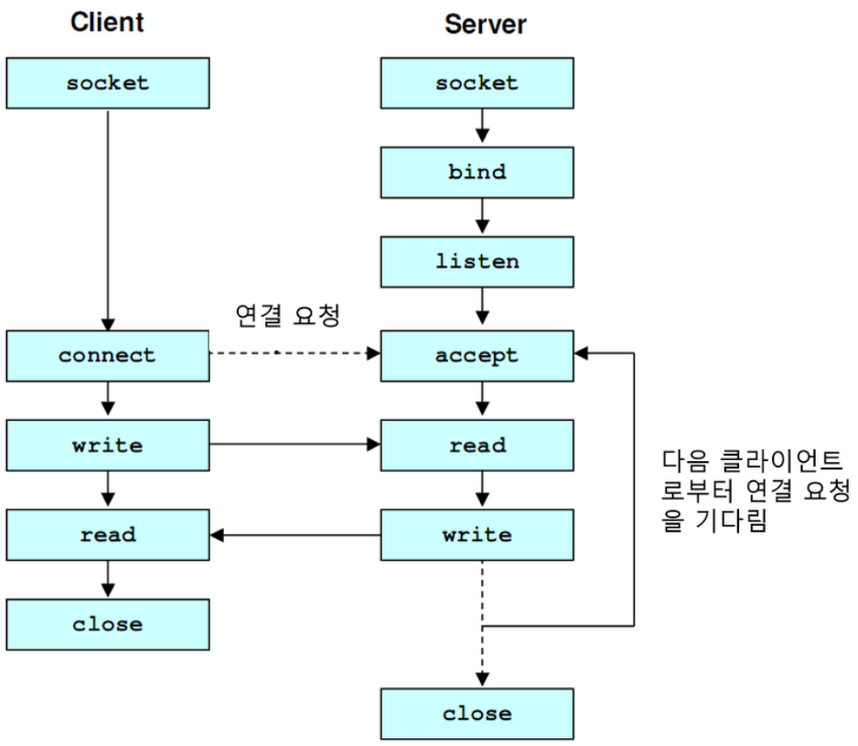
소켓(socket) 이란 프로세스간의 통신에 사용되는 양쪽 끝단을 의미한다. 자바에서는 java.net패키지를 통해 소캣 프로그래밍을 지원하는데, 소켓 통신에 사용되는 프로토콜에 따라 다른 종류의 소켓을 구현하여 제공한다.

1. TCP(Transmission Control Protocol)

인터넷상에서 데이터를 메세지의 형태로 보내기 위해 IP와 함께 사용하는 프로토콜. 일반적으로 TCP와 IP를 함께 사용하는데, IP가 데이터의 배달을 처리한다면 TCP는 패킷을 추적 및 관리한다.
TCP는 연속성보다 신뢰성있는 전송이 중요할 때에 사용하는 프로토콜로 예를 들면 파일 전송과 같은 경우에 사용한다.

패킷(Packet)이란?

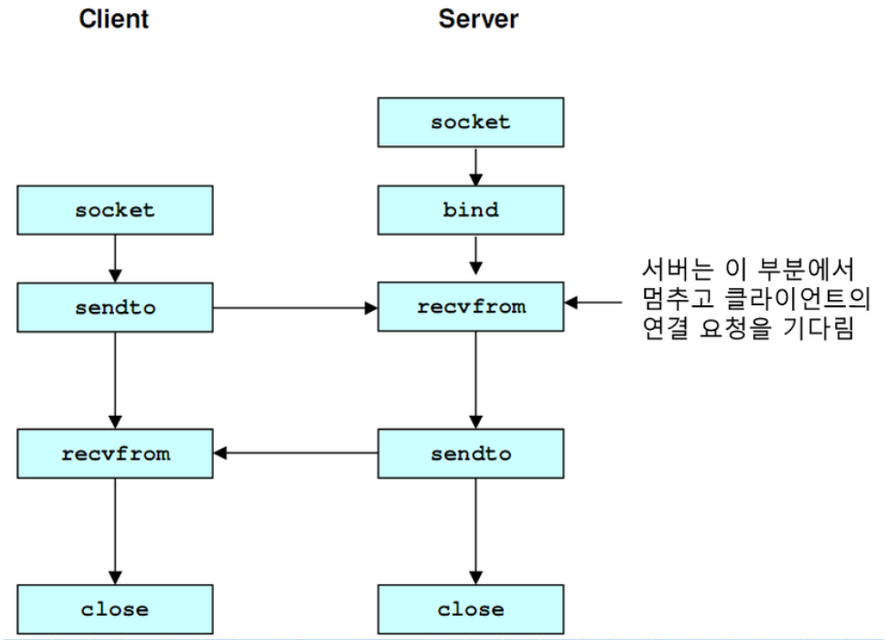
인터넷 내에서 데이터를 보내기 위한 경로배정(라우팅)을 효율적으로 하기 위해서 데이터를 여러 개의 조각들로 나누어 전송을 하는데 이때, 이 조각을 패킷이라고 한다.



TCP Flow

2. UDP(User Datagram Protocol)

데이터를 데이터그램 단위로 처리하는 프로토콜.
데이터그램이란 독립적인 관계를 지니는 패킷이라는 뜻으로 TCP와 달리 UDP는 비연결형 프로토콜. 즉, 연결을 위해 할당되는 논리적인 경로가 없는데, 그렇기 때문에 각각의 패킷은 다른 경로로 전송되고, 각각의 패킷은 독립적인 관계를 지니게 되는데 이렇게 데이터를 서로 다른 경로로 독립적으로 처리하게 된다.
서로 다른 경로로 독립적으로 처리함에도 패킷에 순서를 부여하여 재조립을 하거나 흐름 제어 또는 혼잡 제어와 같은 기능도 처리하지 않기에 TCP보다 속도가 빠르며 네트워크 부하가 적다는 장점이 있지만 신뢰성있는 데이터의 전송을 보장하지는 못한다.
그렇기 때문에 신뢰성보다는 연속성이 중요한 서비스 예를 들면 실시간 서비스(streaming)에 자주 사용된다.



UDP Flow

TCP와 UDP의 특징과 차이

	TCP	UDP
연결방식	연결기반 (connection-oriented) - 연결 후 통신 - 1:1 통신방식	비연결기반(connectionless-oriented) - 연결없이 통신(소포) - 1:1, 1:n, n:n 통신방식
특징	데이터의 경계를 구분안함 (byte-stream) 신뢰성 있는 데이터 전송 - 데이터의 전송순서가 보장됨 - 데이터의 수신여부를 확인 함 (데이터가 손실되면 재전송됨) - 패킷을 관리할 필요가 없음 UDP 보다 전송속도가 느림	데이터의 경계를 구분함 (datagram) 신뢰성 없는 데이터 전송 - 데이터의 전송순서가 바뀔 수 있음 - 데이터의 수신여부를 확인 안 함 (데이터가 손실되어도 알 수 없음) - 패킷을 관리해주어야 함 TCP보다 전송속도가 빠름
관련 클래스	Socket ServerSocket	DatagramSocket DatagramPacket MulticastSocket

TCP/IP 서버 구현

<TcpIpServer.java> 서버 코드

```
package SocketProgramming;
import java.util.Date;
import java.io.*;
import java.net.*;
import java.text.SimpleDateFormat;

public class TcpIpServer {
	public static void main(String[] args) {
		ServerSocket serverSocket = null;

		try {
			serverSocket = new ServerSocket(8888);
			System.out.println(getTime() + "서버가 준비되었습니다.");
		} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		while (true) {
			try {
			System.out.println(getTime() + "연결요청을 기다리는중 . . .");

			//서버 소켓은 클라이언트의 연결요청이 올때까지 실행을 멈추고 계속 기다린다.
			//클라이언트의 연결요청이 들어오면 클라이언트 소켓과 통신할 새로운 소켓을 생성한다.
			Socket socket = serverSocket.accept();
			System.out.println(getTime() + socket.getInetAddress() + "로  부터 연결 요
청이 들어왔습니다.");

			//소켓의 출력스트림을 얻는다.
			OutputStream out = socket.getOutputStream();
			DataOutputStream dos = new DataOutputStream(out);

			//원격 소켓에 데이터를 보낸다.
			dos.writeUTF("test 메세지 입니다");
			System.out.println(getTime() + "데이터를 전송했습니다.");

			//스트림과 소켓을 닫아준다.
			dos.close();
			socket.close();
			} catch (IOException e) {
				e.printStackTrace();
			}
		}
	}
	//현재 시간을 문자열로 반환하는 함수
	private static String getTime() {
		SimpleDateFormat f = new SimpleDateFormat("[hh.mm.ss]");
		return f.format(new Date());
	}
}
```

<TcpIpClient.java> 클라이언트 코드

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

public class TcpIpClient {
	public static void main(String[] args) {
		try {
			String serverIp = "211.195.37.73";

			System.out.println("서버에 연결 중 입니다. 서버 IP : " + serverIp);

			//소켓을 생성하여 연결을 요청.
			Socket socket = new Socket(serverIp, 8888);

			//소켓의 입력 스트림을 얻는다.
			InputStream in = socket.getInputStream();
			DataInputStream dis = new DataInputStream(in);

			//소켓으로 받은 데이터 출력.
```

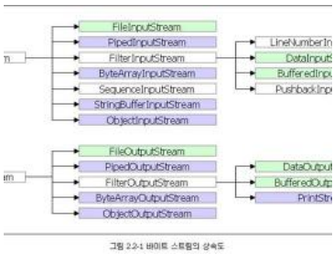
```
                        System.out.println("받은 메세지 : " + dis.readUTF());
                        System.out.println("연결을 종료합니다 .");

                        //스트림과 소켓을 닫는다.
                        dis.close();
                        socket.close();
                        System.out.println("연결이 종료되었습니다.");
                } catch (ConnectException ce) {
                        ce.printStackTrace();
                } catch (IOException ie) {
                        ie.printStackTrace();
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
}
```

공감

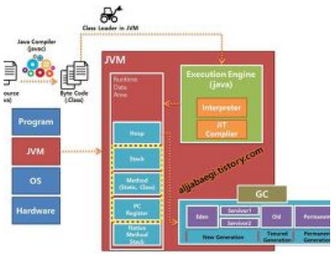
'**Programming Language** > **Java**' 카테고리의 다른 글

[JAVA] 소켓(socket) 프로그래밍 (0)	2021.01.20
[JAVA]입출력 IO & NIO (0)	2021.01.20
[JAVA]자바 컴파일은 어떻게 동작할까? (0)	2021.01.18
[JAVA] Garbage Collection (가비지컬렉션)이란? (0)	2021.01.18
[JAVA] JDK, JRE, JVM이란? (0)	2021.01.18

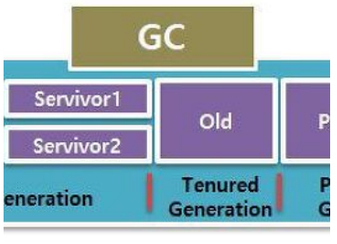
'Programming Language/Java' Related Articles



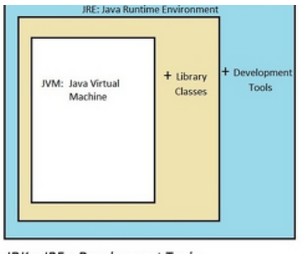
[JAVA]입출력 IO & NIO



[JAVA]자바 컴파일은 어떻게 동작할까?



[JAVA] Garbage Collection (가비지컬…



[JAVA] JDK, JRE, JVM이란?

여러분의 소중한 댓글을 입력해주세요.

댓글달기