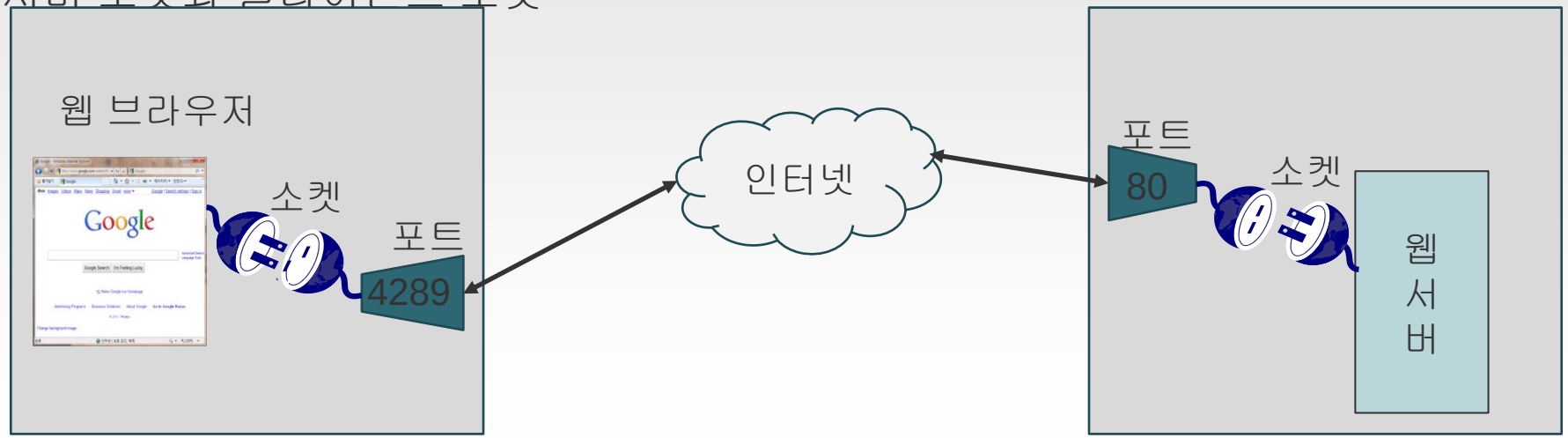
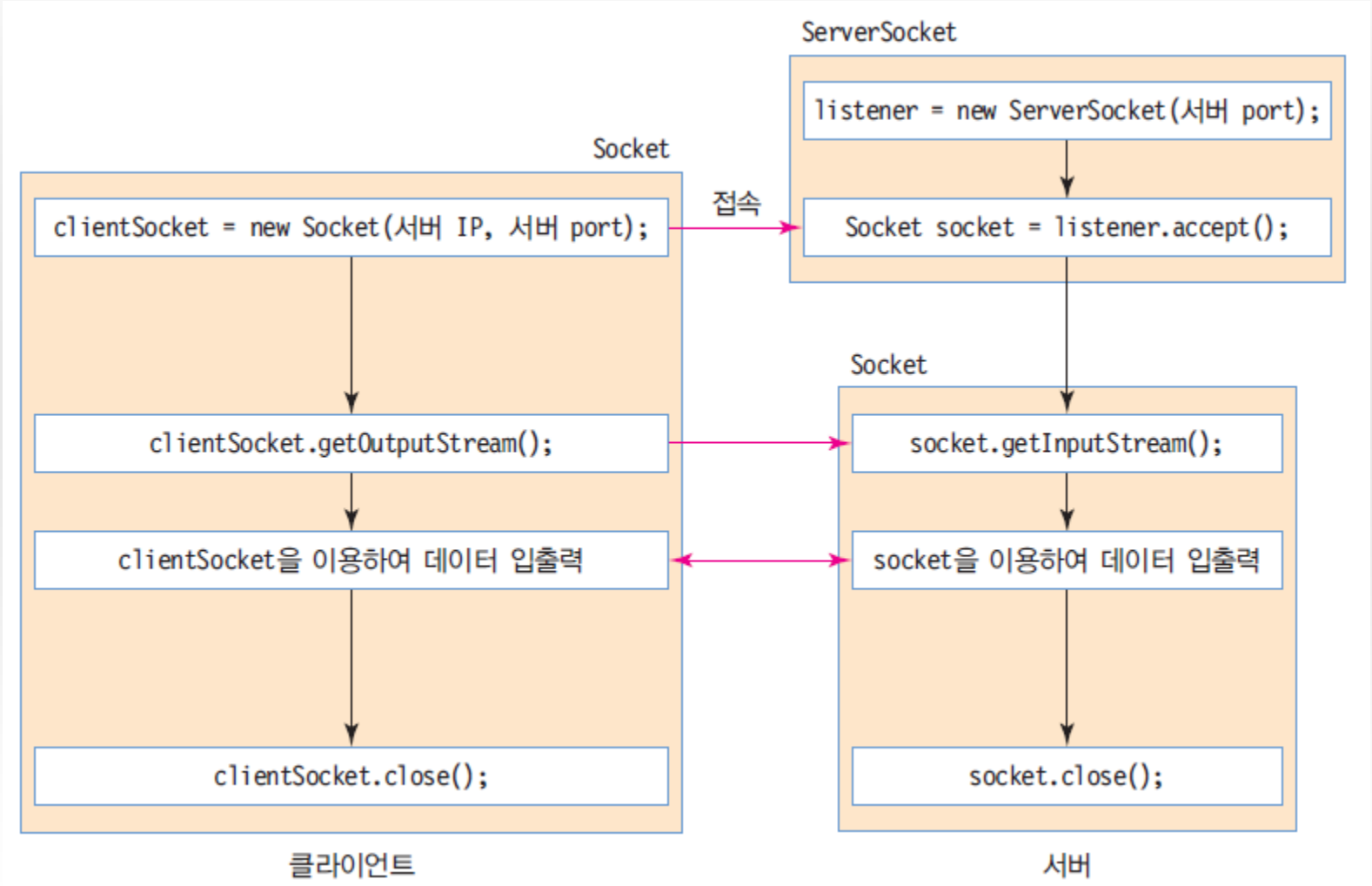
5주차
Java 소켓 프로그래밍

Java 소켓 프로그래밍

- 소켓 (socket)
  - 소켓은 네트워크 상에서 수행되는 두 프로그램 간의 양방향 통신 링크의 한쪽 끝 단을 의미
  - 소켓은 특정 포트 번호와 연결되어 있음
    - TCP에서 데이터를 보낼 응용프로그램을 식별할 수 있음.
  - 자바에서의 데이타 통신 시 소켓 사용
  - 소켓 종류
    - 서버 소켓과 클라이언트 소켓



Java 소켓 프로그래밍



Java 소켓 프로그래밍

- Socket 클래스
  - 클라이언트 소켓에 사용되는 클래스
  - java.net 패키지에 포함
  - 주요 생성자

생성자	설명
Socket(InetAddress address, int port)	소켓을 생성하여 지정된 IP 주소와 포트 번호에 연결한다.
Socket(String host, int port)	소켓을 생성하여 지정된 호스트와 포트 번호에 연결한다. 호스트 이름이 null인 경우는 루프백(loopback) 주소로 가정한다.

Java 소켓 프로그래밍

- 주요 메소드

메소드	설명
void close()	소켓을 닫는다.
void connect(SocketAddress endpoint)	소켓을 서버에 연결
InetAddress getInetAddress()	소켓이 연결한 서버의 주소 반환
InputStream getInputStream()	소켓에 대한 입력 스트림 반환
InetAddress getLocalAddress()	소켓이 연결된 로컬 주소 반환
int getLocalPort()	소켓이 연결된 로컬 포트 번호 반환
int getPort()	소켓이 연결한 서버의 포트 번호 반환
OutputStream getOutputStream()	소켓에 대한 출력 스트림 반환
boolean isBound()	소켓이 로컬 주소에 연결되어있으면 true 반환
boolean isConnected()	소켓이 서버에 연결되어 있으면 true 반환
boolean isClosed()	소켓이 닫혀있으면 true 반환
void setSoTimeout(int timeout)	데이터 읽기 타임아웃 시간 지정. 0이면 타임아웃 해제.

Java 소켓 프로그래밍

- 클라이언트 소켓 생성 및 서버에 접속

```
Socket clientSocket = new Socket("128.12.1.1", 5550);
```

  - Socket 객체의 생성되면 곧 바로 128.12.1.1의 주소로 자동 접속
- 네트워크 입출력 스트림 생성

```
BufferedReader in = new BufferedReader(
                        new InputStreamReader(clientSocket.getInputStream()));
BufferedWriter out = new BufferedWriter(
                        new OutputStreamWriter(clientSocket.getOutputStream()));
```

  - 일반 스트림을 입출력 하는 방식과 동일
- 서버로 데이터 전송
  - flush()를 호출하면 스트림 속에 데이터를 남기지 않고 모두 전송

```
out.write("hello"+"\n");
out.flush();
```

- 서버로부터 데이터 수신

```
int x = in.read();                  // 서버로부터 한 개의 문자 수신
String line = in.readline(); //서버로부터 한 행의 문자열 수신
```

- 네트워크 접속 종료

```
clientSocket.close();
```

Java 소켓 프로그래밍

- ServerSocket 클래스
  - 서버 소켓에 사용되는 클래스
  - java.net 패키지에 포함
  - 주요 생성자

생성자	설명
ServerSocket(int port)	소켓을 생성하여 지정된 포트 번호에 연결한다.

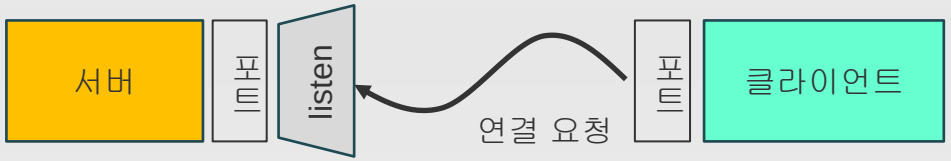
  - 주요 메소드

메소드	설명
Socket accept()	연결 요청을 기다리다 연결 요청이 들어오면 수락하고 새 Socket 객체를 반환
void close()	서버 소켓을 닫는다.
InetAddress getInetAddress()	서버 소켓에 연결된 로컬 주소 반환
int getLocalPort()	서버 소켓이 연결 요청을 모니터링하는 포트 번호 반환
boolean isBound()	서버 소켓이 로컬 주소에 연결되어있으면 true 반환
boolean isClosed()	서버 소켓이 닫혀있으면 true 반환
void setSoTimeout(int timeout)	accept()에 대한 타임 아웃 시간 지정. 0이면 타임아웃이 해제.

Java 소켓 프로그래밍

- 클라이언트와 서버 연결
  - 서버는 서버 소켓으로 들어오는 연결 요청을 기다림

  - 클라이언트가 서버에게 연결 요청



  - 서버가 연결 요청 수락하고 새로운 소켓을 만들어 클라이언트와 연결 생성

Java 소켓 프로그래밍

- 서버 소켓 생성

```
ServerSocket serverSocket = new ServerSocket(5550);
```

  - 이미 사용 중인 포트 번호를 지정하면 오류가 발생
- 클라이언트로부터 접속 기다림

```
Socket socket = serverSocket.accept();
```

  - accept() 메소드는 연결 요청이 오면 새로운 Socket 객체 반환
  - 서버에서 클라이언트와의 데이터 통신은 새로 만들어진 Socket 객체를 통해서 이루어짐
  - ServerSocket 클래스는 Socket 클래스와 달리 주어진 연결에 대해 입출력 스트림을 만들어주는 메소드가 없음
- 네트워크 입출력 스트림 생성

```
BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
BufferedWriter out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
```

  - accept() 메소드에서 얻은 Socket 객체의 getInputStream()과 getOutputStream() 메소드를 이용하여 데이터 스트림 생성
  - 일반 스트림을 입출력하는 방식과 동일하게 네트워크 데이터 입출력

Java 소켓 프로그래밍

- 클라이언트로부터 데이터 수신

```
int x = in.read();              // 클라이언트로부터 한 개의 문자 수신
String line = in.readline(); //클라이언트로부터 한 행의 문자열 수신
```

- 클라이언트로 데이터 전송

```
out.write("Hi!, Client"+"\n");
out.flush();
```

  - flush()를 호출하면 스트림 속에 데이터를 남기지 말고 모두 전송
- 네트워크 접속 종료

```
socket.close();
```

- 서버 응용프로그램 종료

```
serverSocket.close();
```

  - 더 이상 클라이언트의 접속을 받지 않고 서버 응용 프로그램을 종료하고자 하는 경우 ServerSocket 종료

Java 소켓 프로그래밍

- 간단한 채팅 프로그램 예제
  - 서버와 클라이언트가 1:1로 채팅 하는 간단한 예제
  - 서버와 클라이언트 간의 메시지 구분을 위해 서버는 메시지 앞에 “서버>”을 접두어로 붙여 메시지를 전송하며 클라이언트는 “클라이언트>”를 접두어로 붙여 메시지 전송
  - 서버와 클라이언트가 번갈아 가면서 메시지 전송 및 수신
  - 클라이언트가 bye를 보내면 프로그램 종료