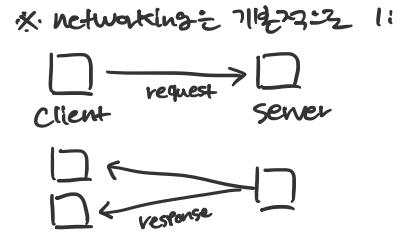
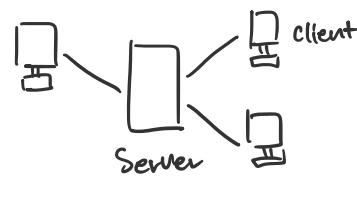


Networking and Threads

2018년 5월 25일 금요일

오후 12:30

• Client-Server



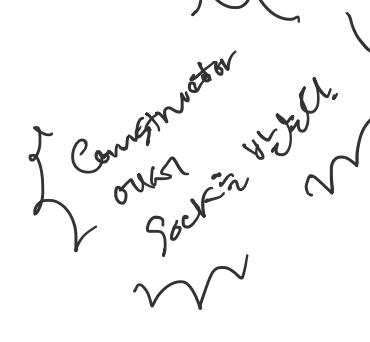
• Connecting, Sending, and Receiving

- Connect : 클라이언트가 서버에 연결을 요청, 승인되면 Socket object가 만들어짐.
Socket은 I/O Stream은 예상.

- Send
- Receive

• Socket Connection

```
Socket soc = new Socket("196.164.1.103", 8080);
```



• Port number

- TCP Port는 16bit number (0~1023)
- Well-known TCP Port : 20(FTP), 23(Telnet), 443(HTTPS), 80(HTTP) ...

• Read

```
InputStreamReader stream = new InputStreamReader(soc.getInputStream());
BufferedReader reader = new BufferedReader(stream);
String msg = reader.readLine();
```

• Write

```
PrintWriter writer = new PrintWriter(soc.getOutputStream());
writer.println("message");
```

• Writing a Server

- ServerSocket이 필요.

```
ServerSocket soc = new ServerSocket(5000);
Socket sock = soc.accept(); ← 여기서 Client의 request를 대기. (blocking mode)
- 연결은 5000 포트로 되지만
  이를 I/O는 다른 포트로 이용함.
  (Server가 여러 Client를 5000
  포트로 받았어야 하기 때문.)
  Socket soc = new Socket("196.164.1.103", 5000);
  이 Client에서 서버로 되면 block이 풀림.
  block이 풀리면 SOCK object가 만들어짐.
  PrintWriter writer = new PrintWriter(sock.getOutputStream());
  writer.println("message");
  writer.close();
  위처럼 soc를 이용해 I/O 수행.
```

• Multithread

```
Runnable r = new ThreadJob();
Thread t = new Thread(r);
t.start();
```

Runnable 클래스에는 run() 메소드가 있음.

• States of a thread

- time sharing 방식으로 thread의 state가 New, Runnable,



Blocked : CPU가 입력을 기다릴 때

file의 데이터를 load하는 때

• Thread Scheduler

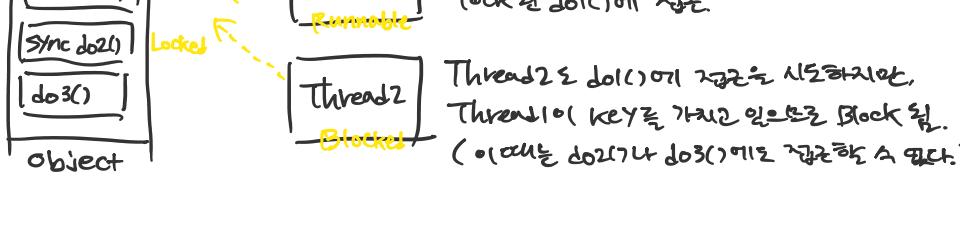
- 스레드 running이 되지 못하는 경우.
- 결정할 예상할 수 있음. (스레드 사이 어떤 코드가 먼저 실행될지 예상)

• Concurrency issue

- 스레드 할당 synchronization을 잘 처리해야 함.
- Thread.sleep(...)을 쓸 수도. ← 바로 안 할 방법.
- Peterson's Solution처럼 flag를 Critical section에 진입하는 걸 제어하는 수도.
↳ C, atomic still running해야 함. C에서는 System call에 접근했지만, Java에서는 더 간단한 방법이 있음.
일단 코드에 대해서 **private synchronized void do()**를 넣음.

• Using Object's Lock

- 모든 자바 object는 lock을 가진다.
- 하나의 lock은 하나의 key만 가질 수 있다.



• Synchronized Block

```
void do() {
    synchronized (obj) {
```

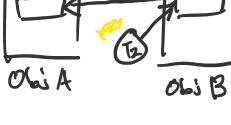
...
 }
}

← obj block의 lock이 걸린다.

...

}

• Deadlock



스레드가 동시에 서로에게
key를 요구해 스레드가
모두 block되는 현상.

- Wait(): key는 object에게 반환하고 block 해.

- Notify(): block 되어 있는 thread에게 key를 가져온다.