■ 채팅 프로그램 만들기

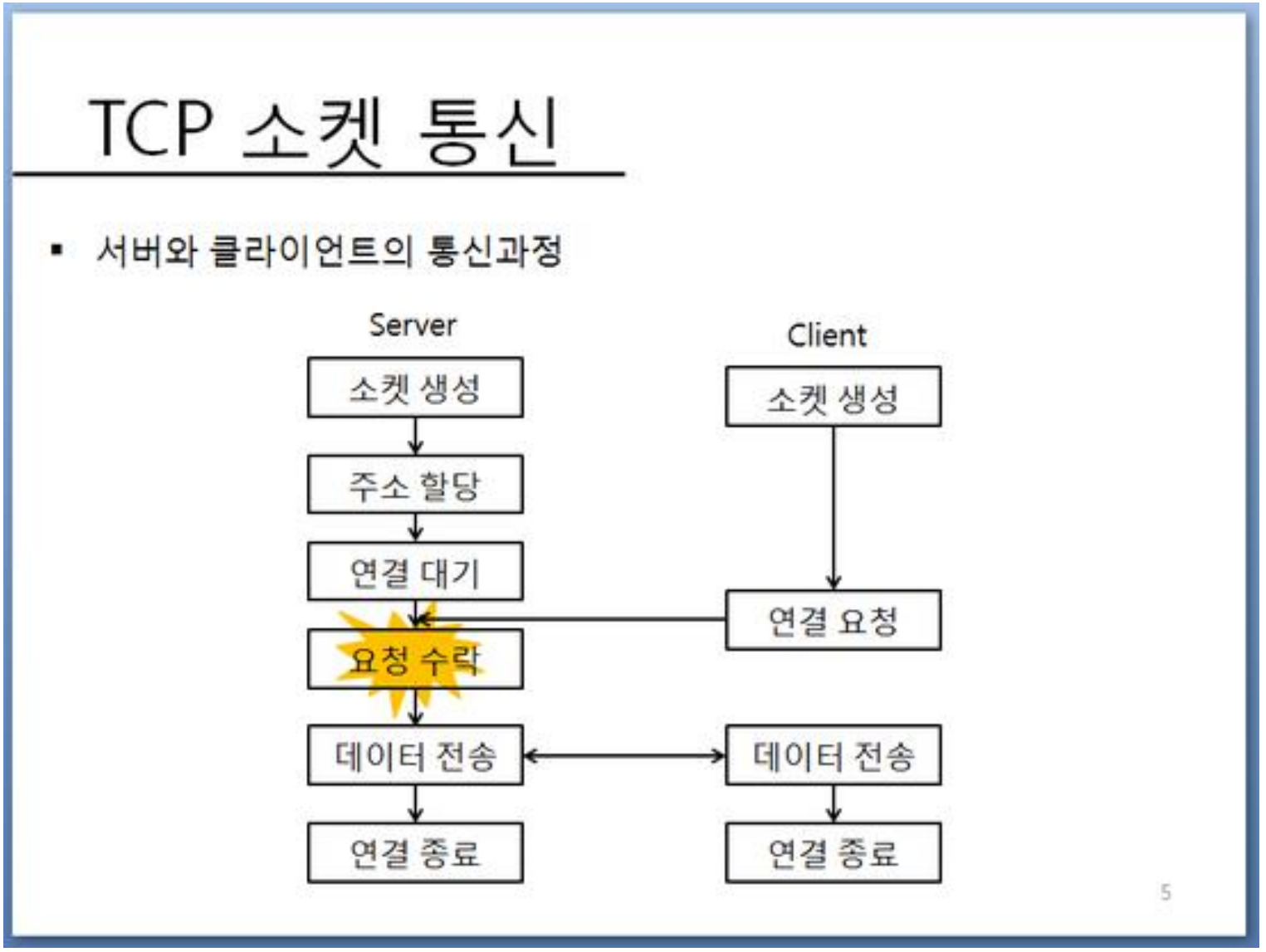
스레드 미사용

1. 단방향 (클라이언트 → 서버) / 한번
2. 단방향 (클라이언트 → 서버) / 계속
3. 반이중 (클라이언트 → 서버, 클라이언트 ← 서버) / 동시 전송 불가

스레드 사용

4. 전이중 (클라이언트 → 서버, 클라이언트 ← 서버) / 동시 전송 가능
5. N개의 클라이언트와 서버 통신

■ 채팅 프로그램 만들기



이미지 출처 : http://blog.naver.com/PostView.nhn?blogId=awesomedev&logNo=220690707290

■ 채팅 프로그램 만들기

1. 단방향 (클라이언트 → 서버) / 한번

ex.Server1Exam

```
ServerSocket sSocket = new ServerSocket(20000);
Socket socket = sSocket.accept();

InputStream in = socket.getInputStream();
InputStreamReader isr = new InputStreamReader(in);
BufferedReader reader = new BufferedReader(isr);

String data = reader.readLine();
System.out.println(data);

sSocket.close();
```

ex.Client1Exam

```
Socket socket = new Socket("127.0.0.1", 20000);

OutputStream out = socket.getOutputStream();
PrintWriter writer = new PrintWriter(out, true);

String data = "Hello~";
writer.println(data);

socket.close();
```

■ 채팅 프로그램 만들기

1. 단방향 (클라이언트 → 서버) / 한번

① Server 구동 → 대기

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server1Exam
```

② Client 접속 → 데이터 전송 → Client 종료

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client1Exam

D:₩dev₩workspace-eclipse₩JavaStudy₩bin>
```

③ 데이터 수신 → 출력 → Server 종료

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server1Exam
Hello~

D:₩dev₩workspace-eclipse₩JavaStudy₩bin>
```

■ 채팅 프로그램 만들기

2. 단방향 (클라이언트 → 서버) / 계속

ex.Server2Exam

```
ServerSocket sSocket = new ServerSocket(20000);
Socket socket = sSocket.accept();

InputStream in = socket.getInputStream();
InputStreamReader isr = new InputStreamReader(in);
BufferedReader reader = new BufferedReader(isr);

while (true) {
  String data = reader.readLine();
  if(data == null) break;
  System.out.println(data);
}

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

■ 채팅 프로그램 만들기

2. 단방향 (클라이언트 → 서버) / 계속

ex.Client2Exam

```
Socket socket = new Socket("127.0.0.1", 20000);

OutputStream out = socket.getOutputStream();
PrintWriter writer = new PrintWriter(out, true);

Scanner scan = new Scanner(System.in);
while (true) {
  String data = scan.nextLine();
  if(data.equals("q!")) break;
  writer.println(data);
}

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

■ 채팅 프로그램 만들기

2. 단방향 (클라이언트 → 서버) / 계속

① Server 구동 → 대기

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server2Exam
```

② Client 접속 → 데이터 직접 입력 후 전송

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client2Exam
Hello~
```

③ 데이터 수신 → 출력 → 대기

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server2Exam
Hello~
```

④ 데이터 직접 입력 후 전송

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client2Exam
Hello~
```

⑤ 데이터 수신 → 출력 → 대기

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server2Exam
Hello~
Bye~
```

⑥ 데이터 직접 입력 후 전송 → 종료

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client2Exam
Hello~
Bye~
q!

D:₩dev₩workspace-eclipse₩JavaStudy₩bin>
```

■ 채팅 프로그램 만들기

3. 반이중 (클라이언트 → 서버, 클라이언트 ← 서버) / 동시 전송 불가

ex.Server3Exam

```
ServerSocket sSocket = new ServerSocket(20000);
Socket socket = sSocket.accept();

InputStream in = socket.getInputStream();
InputStreamReader isr = new InputStreamReader(in);
BufferedReader reader = new BufferedReader(isr);

OutputStream out = socket.getOutputStream();
PrintWriter writer = new PrintWriter(out, true);

while (true) {
  String data = reader.readLine();
  if(data == null) break;
  System.out.println(data);
  writer.println(data);
}

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

■ 채팅 프로그램 만들기

3. 반이중 (클라이언트 → 서버, 클라이언트 ← 서버) / 동시 전송 불가

ex.Client3Exam

```
Socket socket = new Socket("127.0.0.1", 20000);

OutputStream out = socket.getOutputStream();
PrintWriter writer = new PrintWriter(out, true);

InputStream in = socket.getInputStream();
InputStreamReader isr = new InputStreamReader(in);
BufferedReader reader = new BufferedReader(isr);

Scanner scan = new Scanner(System.in);
while (true) {
  String data = scan.nextLine();
  if(data.equals("q!")) break;
  writer.println(data);
  System.out.println(reader.readLine());
}

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

■ 채팅 프로그램 만들기

3. 반이중 (클라이언트 → 서버, 클라이언트 ← 서버) / 동시 전송 불가

① Server 구동 → 대기

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server3Exam
```

② 접속 → 데이터 직접 입력 후 전송 → 대기

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client3Exam
hello~
hello~
```

④ 데이터 수신 → 출력

③ 데이터 수신 → 출력 → 수신 데이터 다시 전송

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server2Exam
Hello~
```

⑤ 종료

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client3Exam
hello~
hello~
q!

D:₩dev₩workspace-eclipse₩JavaStudy₩bin>
```

■ 채팅 프로그램 만들기

4. 전이중 (클라이언트 → 서버, 클라이언트 ← 서버) / 동시 전송 가능

ex.Server4Exam

```
ServerSocket sSocket = new ServerSocket(20000);
Socket socket = sSocket.accept();

InputStream in = socket.getInputStream();
Server4Listener listener = new Server4Listener(in);
listener.start();

OutputStream out = socket.getOutputStream();
PrintWriter writer = new PrintWriter(out, true);

Scanner scan = new Scanner(System.in);
while (true) {
  String data = scan.nextLine();
  if(data == null) break;
  writer.println(data);
}

scan.close();
socket.close();
sSocket.close();
```

데이터 수신용 작업 스레드 작성

■ 채팅 프로그램 만들기

4. 전이중 (클라이언트 → 서버, 클라이언트 ← 서버) / 동시 전송 가능

ex.Server4Listener

```
private InputStream in;

public Server4Listener(InputStream in) {
  this.in = in;
}

@Override
public void run() {
  InputStreamReader isr = null;
  BufferedReader reader = null;
  try {
    isr = new InputStreamReader(in);
    reader = new BufferedReader(isr);
    while(true) {
      String data = reader.readLine();
      if(data == null) break;
      System.out.println(data);
    }
  } catch (IOException e) { e.printStackTrace(); }
}
```

■ 채팅 프로그램 만들기

4. 전이중 (클라이언트 → 서버, 클라이언트 ← 서버) / 동시 전송 가능

ex.Client4Exam

```
Socket socket = new Socket("127.0.0.1", 20000);

OutputStream out = socket.getOutputStream();
PrintWriter writer = new PrintWriter(out, true);

InputStream in = socket.getInputStream();
Client4Listener listener = new Client4Listener(in);
listener.start();

Scanner scan = new Scanner(System.in);
while (true) {
  String data = scan.nextLine();
  if(data == null) break;
  writer.println(data);
}

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

데이터 수신용 작업 스레드 작성

■ 채팅 프로그램 만들기

4. 전이중 (클라이언트 → 서버, 클라이언트 ← 서버) / 동시 전송 가능

ex.Client4Listener

```
private InputStream in;

public Client4Listener(InputStream in) {
  this.in = in;
}

@Override
public void run() {
  InputStreamReader isr = null;
  BufferedReader reader = null;
  try {
    isr = new InputStreamReader(in);
    reader = new BufferedReader(isr);
    while(true) {
      String data = reader.readLine();
      if(data == null) break;
      System.out.println(data);
    }
  } catch (IOException e) { e.printStackTrace(); }
}
```

■ 채팅 프로그램 만들기

4. 전이중 (클라이언트 → 서버, 클라이언트 ← 서버) / 동시 전송 가능

① Server 구동 → 대기

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server4Exam
```

② 접속 → 데이터 입력 후 전송

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client4Exam
hello~
```

③ 데이터 수신 → 출력 → 데이터 입력 후 전송

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server4Exam
hello~
서버에서 데이터 입력
```

④ 데이터 수신 → 출력 → 데이터 입력 후 전송

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client4Exam
hello~
서버에서 데이터 입력
클라이언트에서 데이터 입력~
```

⑤ 데이터 수신 → 출력

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server4Exam
hello~
서버에서 데이터 입력
클라이언트에서 데이터 입력~
```

⑥ 반복 또는 종료

■ 채팅 프로그램 만들기

5. N개의 클라이언트와 서버 통신

ex.Server5Exam

```
ServerSocket sSocket = new ServerSocket(20000);

boolean isStart = true;

while(isStart) {
  Socket socket = sSocket.accept();
                                          각 클라이언트 담당 작업 스레드 작성
  Server5Thread client = new Server5Thread(socket);
  client.start();

  Server5Controller.getInstance().addClient(client);
}                                         관리 객체에 클라이언트 추가

sSocket.close();
```

■ 채팅 프로그램 만들기

5. N개의 클라이언트와 서버 통신

ex.Server5Controller

```
private static Server5Controller controller = new Server5Controller();

private Vector<Server5Thread> clients = new Vector<Server5Thread>();

private Server5Controller() {}
```

Singleton Pattern - 객체를 1개만 생성

```
public static Server5Controller getInstance() {
  if(controller == null) controller = new Server5Controller();
  return controller;
}

public void addClient(Server5Thread client) {
  clients.add(client);
}

public void removeClient(Server5Thread client) {
  clients.remove(client);
}

public void sendAll(String message) {
  for(Server5Thread client : clients) {
    client.send(message);
  }
}
```

■ 채팅 프로그램 만들기

5. N개의 클라이언트와 서버 통신

ex.Server5Thread

```
@Override
public void run() {
  System.out.printf("[%s] 접속\n", this.nickname);
  Server5Controller.getInstance().sendAll("[" + this.nickname + "] 접속!");

  try {
    while(true) {
      String message = this.reader.readLine();
      Server5Controller.getInstance().sendAll(
          "[" + this.nickname + "] " + message);
    }
  } catch (IOException e) {
    e.printStackTrace();

    try { this.socket.close(); }
    catch (IOException e1) { e1.printStackTrace(); }
  }
}

public void send(String message) {
  this.writer.println(message);
}
```

■ 채팅 프로그램 만들기

5. N개의 클라이언트와 서버 통신

ex.Client5Exam

```
Scanner scan = new Scanner(System.in);
System.out.print("nickname 입력 => ");
String nickname = scan.nextLine();

Socket socket = new Socket("127.0.0.1", 20000);

OutputStream out = socket.getOutputStream();
PrintWriter writer = new PrintWriter(out, true);

InputStream in = socket.getInputStream();
Client5Listener listener = new Client5Listener(in);
listener.start();

writer.println(nickname);

while (true) {
  String data = scan.nextLine();
  if(data == null) break;
  writer.println(data);
}

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

■ 채팅 프로그램 만들기

5. N개의 클라이언트와 서버 통신

ex.Client5Listener

```
private InputStream in;

public Client5Listener(InputStream in) {
  this.in = in;
}

@Override
public void run() {
  InputStreamReader isr = null;
  BufferedReader reader = null;
  try {
    isr = new InputStreamReader(in);
    reader = new BufferedReader(isr);
    while(true) {
      String data = reader.readLine();
      if(data == null) break;
      System.out.println(data);
    }
  } catch (IOException e) { e.printStackTrace(); }
}
```

■ 채팅 프로그램 만들기

5. N개의 클라이언트와 서버 통신

Server

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server5Exam
```

① Server 구동 → 대기

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Server5Exam
[A] 접속
[B] 접속
```

④ A, B 접속 알림

Client 1번

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client5Exam
nickname 입력 => A
[A] 접속!
```

② A nickname으로 접속

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client5Exam
nickname 입력 => A
[A] 접속!
[B] 접속!
안녕하세요
[A] 안녕하세요
[B] 네 안녕하세요
```

⑤ 메시지 전송

Client 2번

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client5Exam
nickname 입력 => A
[A] 접속!
```

③ B nickname으로 접속

```
D:₩dev₩workspace-eclipse₩JavaStudy₩bin>java ex.Client5Exam
nickname 입력 => B
[B] 접속!
[A] 안녕하세요
네 안녕하세요
[B] 네 안녕하세요
```

⑥ 메시지 전송