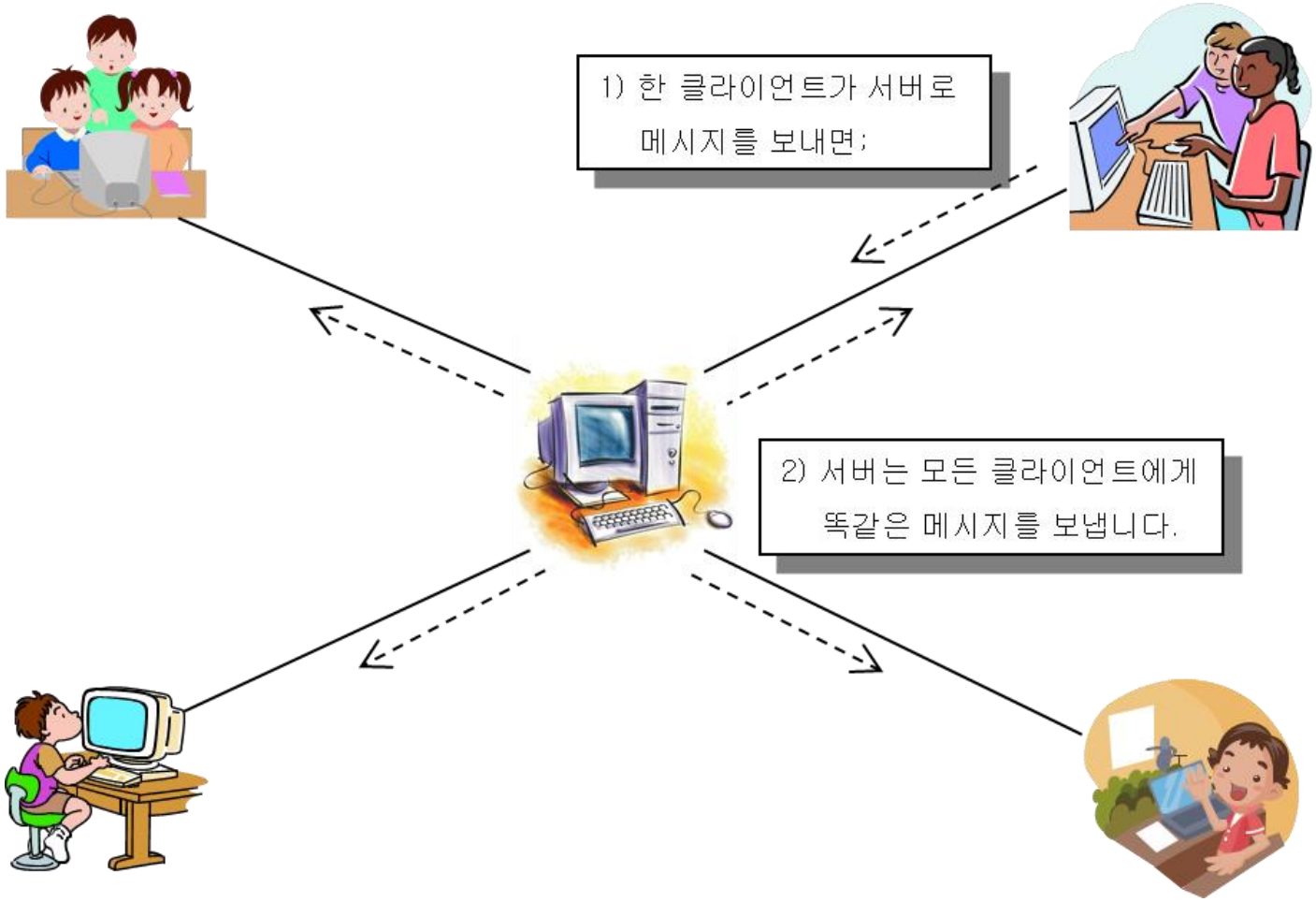
다중채팅

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램

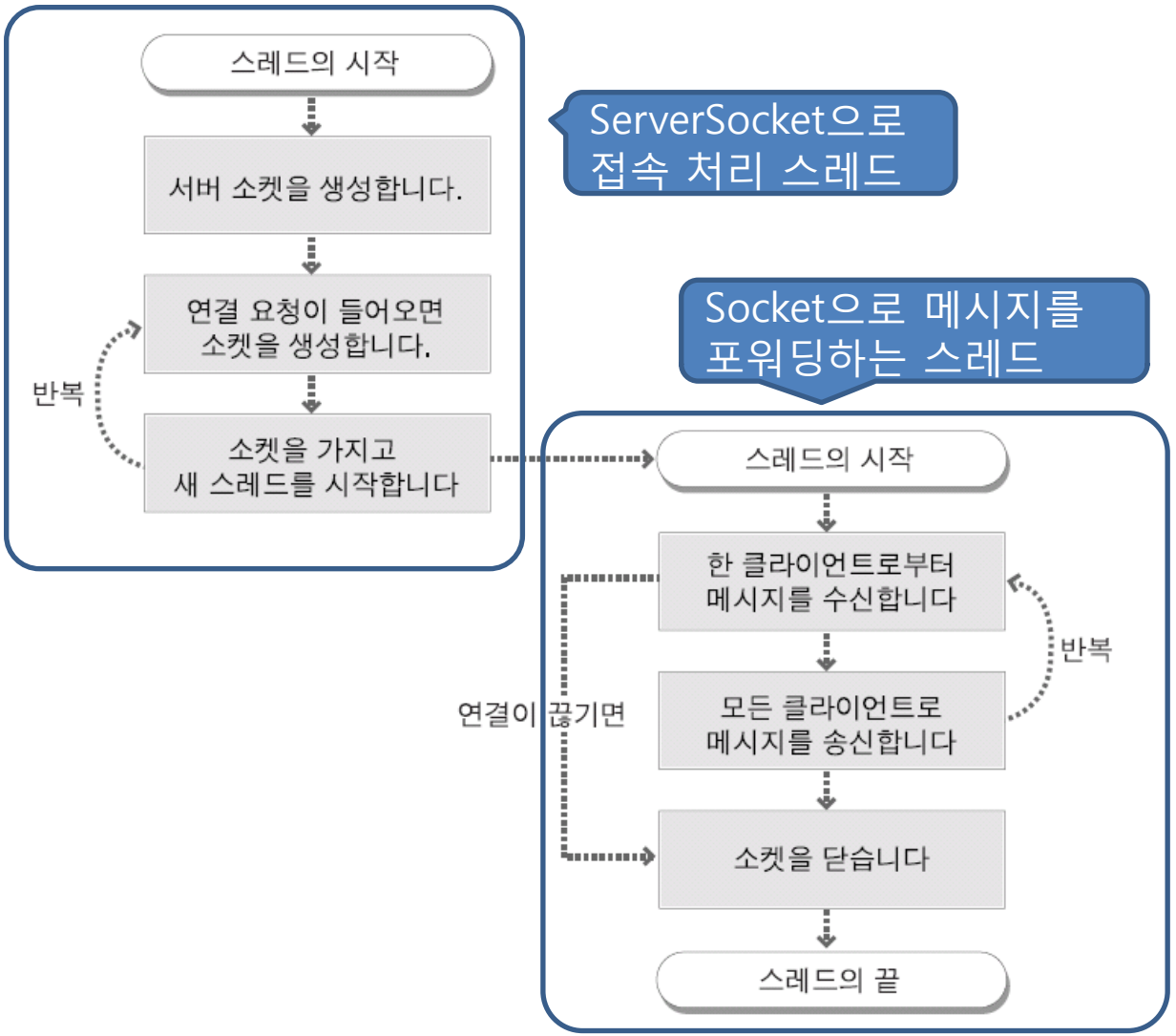
- ㅇ 패키지명 : multichatting
- ㅇ 일반적인 채팅 프로그램의 작동 방식



다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램

- 여러 명이 참여하는 채팅 프로그램의 실행 흐름



다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램

- 여러 사용자가 함께 채팅하는 프로그램 – 서버 프로그램 (미완성)

```
import java.net.*;
class MultiChattingServer {
    public static void main(String[] args) {
        ServerSocket serverSocket = null;
        try {
            serverSocket = new ServerSocket(9002);
            while (true) {
                Socket socket = serverSocket.accept();
                Thread thread = new PerClientThread(socket);
                thread.start();
            }
        }
        catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램

- 여러 사용자가 함께 채팅하는 프로그램 – 서버 프로그램 (미완성)

```
class PerClientThread extends Thread {
    static List<PrintWriter> list = new ArrayList<PrintWriter>();
    Socket socket;
    PrintWriter writer;

    PerClinetThread(Socket socket) {
        this.socket= socket;
        try {
            writer = new PrintWriter(socket.getOutputStream());
            list.add(writer);
        }
        catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
```

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램

- 여러 사용자가 함께 채팅하는 프로그램 – 서버 프로그램 (미완성)

```
public void run() {
    String name = null;
    try {
        BufferedReader reader = new BufferedReader(
            new InputStreamReader(socket.getInputStream()));

        name = reader.readLine();
        sendAll("#" + name + "님이 들어오셨습니다");

        while (true) {
            String str = reader.readLine();  // 메시지 수신
            if (str == null)
                break;
            sendAll(name + ">" + str);
         }
    }
```

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램

- ㅇ 여러 사용자가 함께 채팅하는 프로그램 - 서버 프로그램 (미완성)

```
        catch (Exception e) {
            System.out.println(e.getMessage());
        }
        finally {
            list.remove(writer);   // 리스트에서 출력 스트림 제거
            sendAll("#" + name + "님이 나가셨습니다");
            try { socket.close(); }
            catch (Exception e) {   }
        }
    }

    // 접속한 모든 사용자에게 메시지 포워딩
    private void sendAll(String str) {
        for (PrintWriter writer : list) {
            writer.println(str);
            writer.flush();
        }
    }
}
```

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램

- ArrayList 객체의 멀티 스레드 접근을 안전하게 만드는 방법

```
List list2 = Collections.synchronizedList(list1);
```

동기화된 ArrayList 객체

ArrayList 객체

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램

- 여러 사용자가 함께 채팅하는 프로그램 – 서버 프로그램 (미완성)

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

class PerClientThread extends Thread {
    static List<PrintWriter> list = Collections.synchronizedList(
                                        new ArrayList<PrintWriter>());
    :
}
```

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램 - 클라이언트

- 여러 사용자가 함께 채팅하는 프로그램

```
class MultiChattingClient {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        System.out.print("user name > ");
        String userName = s.nextLine();

        try {
            Socket socket = new Socket("###.###.###.###", 9002);
            Thread thread1 = new SenderThread(socket, username);
            Thread thread2 = new ReceiverThread(socket);
            thread1.start();
            thread2.start();
        }
        catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램 - 클라이언트

- 여러 사용자가 함께 채팅하는 프로그램 – 메시지 전송 스레드

```
class SenderThread extends Thread {
    Socket socket;
    String name;

    SenderThread(Socket socket, String name) {
        this.socket = socket;
        this.name = name;
    }
```

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램 - 클라이언트

- 여러 사용자가 함께 채팅하는 프로그램 – 메시지 전송 스레드

```
public void run() {
    try {
        Scanner s = new Scanner(System.in));
        PrintWriter writer = new PrintWriter(
                                        socket.getOutputStream());
        writer.println(name);
        writer.flush();

        while (true) {
            String str = s.nextLine();
            if (str.equals("bye"))     // 채팅 종료
                break;
            writer.println(str);
            writer.flush();
        }
    }
```

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램 - 클라이언트

- 여러 사용자가 함께 채팅하는 프로그램 – 메시지 전송 스레드

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

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램 - 클라이언트

- 여러 사용자가 함께 채팅하는 프로그램 – 메시지 수신 스레드

```
class ReceiverThread extends Thread {
    Socket socket;

    ReceiverThread(Socket socket) {
        this.socket = socket;
    }
```

다중 채팅 프로그램

❖ 여러 명이 참여하는 채팅 프로그램 - 클라이언트

- 여러 사용자가 함께 채팅하는 프로그램 – 메시지 수신 스레드

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

            while (true) {
                String str = reader.readLine();
                if (str == null)
                    break;
                System.out.println(str);
            }
        }
        catch (IOException e) {
            System.out.println(e.getMessage());
        }
    }
}
```