

Java UDP Socket Programming

□ UDP chatting Program

- UDP Echo Program
- **UDP Chatting Program**
 - Main
 - no-GUI vs. GUI
 - Socket Interface
 - Chatting Protocol

UDP Echo programming

Goal: UDP를 사용하여 메시지를 주고 받음

UDPMyEchoServer에서는 포트번호를 매개변수로 받아 DatagramSocket 타입의 **socket** 객체를 만듦. DatagramPacket의 객체를 이용하여 패킷을 수신하고 동일 패킷을 에코한다. UDPMyEcho에서는 DatagramSocket을 만들고 키보드에서 입력을 받아 서버로 보내고 그 결과를 받아 화면에 프린트한다.

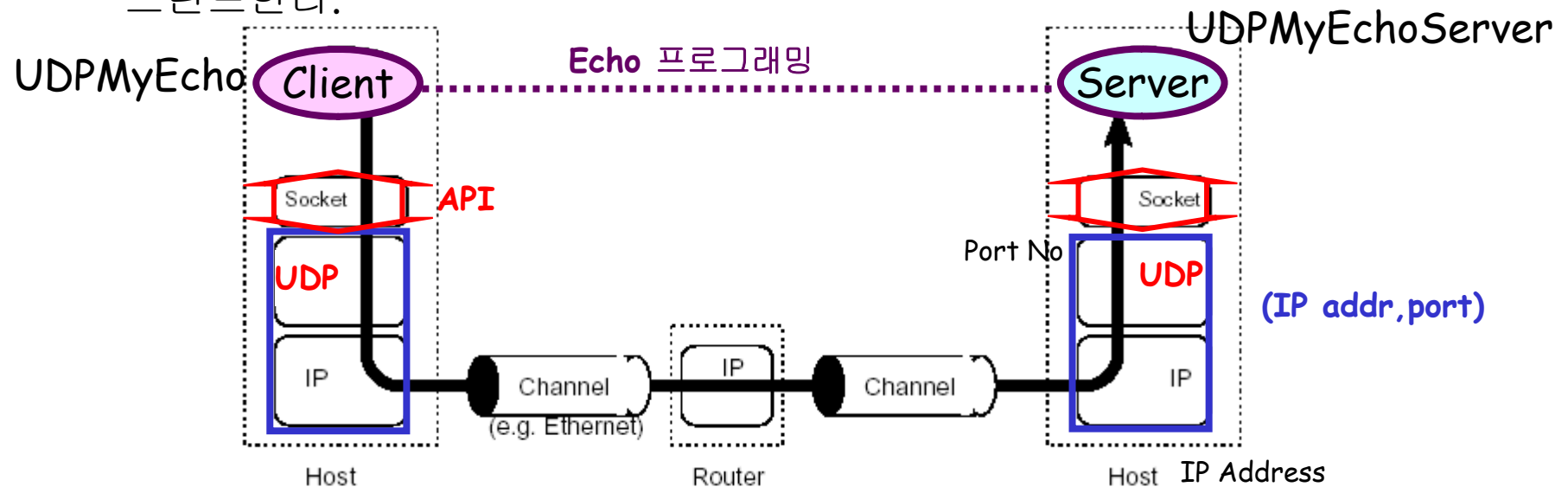


Figure 1.1: A TCP/IP Network

UDP Echo Working Chat

□ Client: Initiates the connection

Client: UDPMYEcho

Server: UDPMYEchoServer

1. Create a DatagramSocket(SrvIP,port);
2. Datagram.send(DatagramSocket.send(Datagram_packet))
3. Receive & Display (DatagramSocket.receive(Datagram_packet))
4. Repeatedly, 2-3:

1. Create a DatagramSocket (DatagramSocket(args[0]))
2. Set DatagramSocket to receive (Datagram packet)
3. Echo: (Datagram_packet)

Repeatedly, 2-3:

Mission 1: local IP addr & port

❑ Server: initially waits to respond

// 파일명 : UDPMYEchoServer.java

import java.net.*;

import java.io.*;

public class UDPMYEchoServer {

final int MAXBUFFER = 512;

public static void main (String[] args) {

int arg_port = Integer.parseInt(args[0]); // 포트 번호

new UDPMYEchoServer().work(arg_port);

}

void work(int arg_port) {

int port = arg_port;

try {

/*

* UDP Socket 생성 (UDP server Socket)

*/

DatagramSocket Dsocket /* Fill in the blank */;

System.out.println ("Running the UDP Echo Server...");

while (true) {

/* UDP Packet 생성

* (UDP server Socket으로부터 데이터 수신을 위한 UDP packet 생성)

* UDP Server Socket에서 UDP packet을 받기 위한 대기

*/

DatagramPacket recy_packet /* Fill in the blank */;

Dsocket.receive (/* Fill in the blank */);

// 으로부터 Echo을 위한 송신 UDP packet 생성

// 에코 데이터 생성을 위해 수신된 UDP packet

DatagramPacket send_packet /* Fill in the blank */;

Dsocket.send (send_packet);

}

} catch(IOException e) {

System.out.println(e);

}

}

}

Additional Mission 1: 다음을 찾아 Display 할 것
자신 IP 주소와 자신의 UDP Port 번호

Mission 2: remote IP addr & port

□ Client: Initiates the connection

```
// 파일명: UDPMYEcho.java
import java.net.*;
import java.io.*;
public class UDPMYEcho {
    final static int MAXBUFFER = 512;
    public static void main(String[] args) {
        if (args.length != 2) {System.out.println("사용법: java UDPMYEcho localhost port"); System.exit(0);}
        int port = Integer.parseInt(args[1]);
        try {
            InetAddress inetaddr = Fill in the blank args[0];
            DatagramSocket socket = Fill in the blank;
            DatagramPacket send_packet; // 송신용 데이터그램 패킷
            DatagramPacket recv_packet; // 수신용 데이터그램 패킷
            BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
            while (true) { // 키보드 입력 읽기
                byte buffer[] = new byte[MAXBUFFER];
                System.out.print("Input Data : ");
                String data = br.readLine();
                if (data.length() == 0) break;
                buffer = data.getBytes(); // 스트링을 바이트 배열로 바꿈
                // 데이터 송신
                send_packet = Fill in the blank;
                socket.send(send_packet);
                // 에코 데이터 수신
                recv_packet = Fill in the blank;
                socket.receive(recv_packet);
                // 화면 출력
                String result = new String(buffer);
                System.out.println("Echo Data : " + result);
            }
        } catch (UnknownHostException ex) {
            System.out.println("Error in the host address ");
        } catch (IOException e) {
            System.out.println(e);
        }
    }
}
```

Mission 2: 다음을 찾아 Display 할 것
상대 IP 주소와 상대의 UDP Port 번호

InetAddress

Goal: InetAddress 형태의 주소로부터 정보를 받음

string getHostName(): domain name 획득

String getHostAddress(): dotted decimal 주소 획득

Byte[] getAddress(): 4 byte IP address

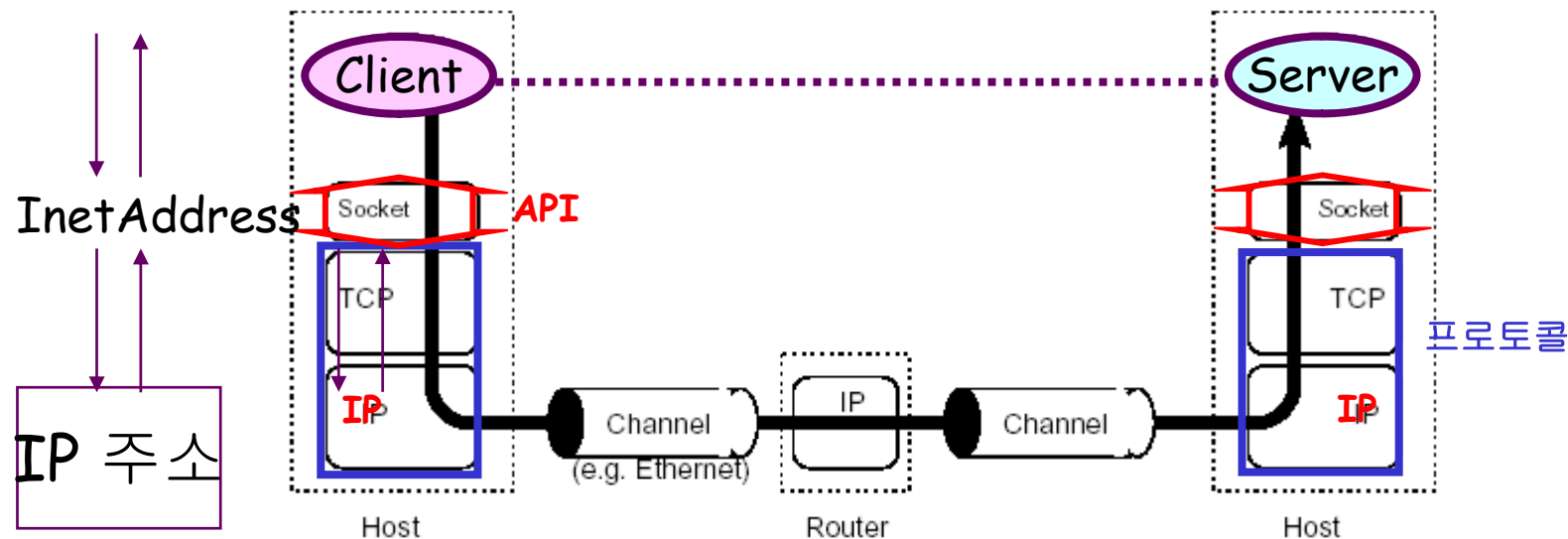


Figure 1.1: A TCP/IP Network

InetAddress programming

```
public class IPAddress {
    TextField inputText; // 호스트 이름 입력 창
    TextArea output; // 결과 출력 창
    public static void main (String args[]) {
        new IPAddress().work();
    }
    public void work() {
        makeFrame();
        // 자신의 IP 주소 찾기
        try {
            InetAddress inetaddr = InetAddress.getLocalHost();
            output.append("\nYour Host name is : " + inetaddr.getHostName());
            output.append("\nYour IP Address is : " + inetaddr.getHostAddress());
        } catch (UnknownHostException ex) {
            output.append("\nError in getLocalHost()\n");
            // 임의의 호스트 IP 주소 찾기
        }
    }
    class AddressListener implements ActionListener {
        public void actionPerformed(ActionEvent ae) {
            String h_name = inputText.getText(); //domain name or IP 주소
            try {
                InetAddress inetaddr = InetAddress.getByName(h_name);
                output.append("\n\nFor the Host : " + inetaddr.getHostName());
                output.append("\n IP Address is : " + inetaddr.getHostAddress());
            } catch (UnknownHostException ex) {
                output.append("\nFailed to find : " + h_name);
            }
        }
    }
}
```

Mission 3: UDP Echo programming with multiple Threads

Goal: UDP를 사용한 Echo program,
단, client 당 하나의 Thread 할당.

Multi-Thread Program

UDPMyEcho

UDPMyEchoServer

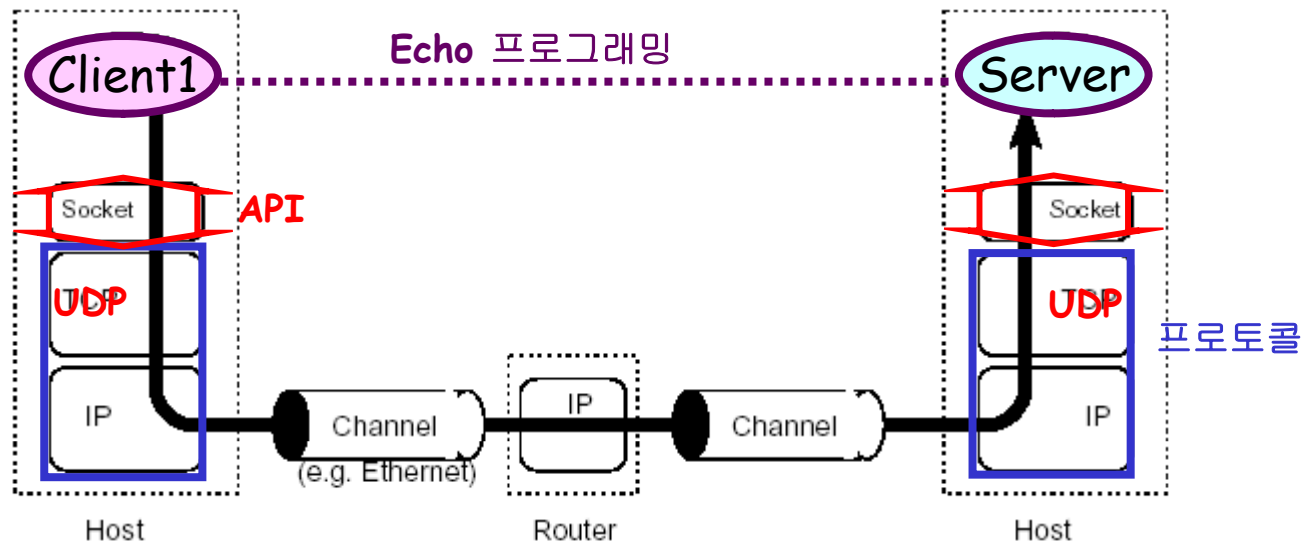


Figure 1.1: A TCP/IP Network

Clients and Servers (UDP) with multiple Threads

The Server-Thread Class

```
public class UDPMYEchoServerMultiThread {
    final int MAXBUFFER = 512;
    private static int a;
    static DatagramSocket Dsocket;
    public static void main (String[] args) {
        int arg_port = Integer.parseInt(args[0]); // 포트 번호
        work(arg_port);
    }
    static void work(int port) {
        try {
            DatagramSocket Dsocket /* Fill in the blank */ ;
            Thread r1 /* Fill in the blank */ ;
            r1.start();
        } catch (SocketException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

```
public class receiveFrame extends Thread {
    DatagramSocket Dsocket;
    receiveFrame (DatagramSocket s) {
        Dsocket = s;
    }
    public void run() {
        while (true) {
            // 데이터 수신
            // 에코 데이터 생성 및 송신
            byte buffer[] = new byte[512];

            try {
                DatagramPacket rcv_packet /* Fill in the blank */ ;
                Dsocket.receive (/* Fill in the blank */ );
                DatagramPacket send_packet /* Fill in the blank */ ;
                Dsocket.send (send_packet);
            } catch (IOException e) {
                System.out.println(e);
            }
        }
    }
} // end ReceiverThread class
```

Mission 4: UDP Echo programming with Timeout

Goal: UDP를 사용한 Echo program,

단, 송신 결과에 대한 Timeout 설정.

서버가 Timeout 이내에 client에게 받은 메시지를 되돌려 보냄.

`Timeout tclick = new Timeout(); tout.Timeoutset(sn,1000,pp)`

UDPMYEchoTimeout

UDPMYEchoTimeoutServer

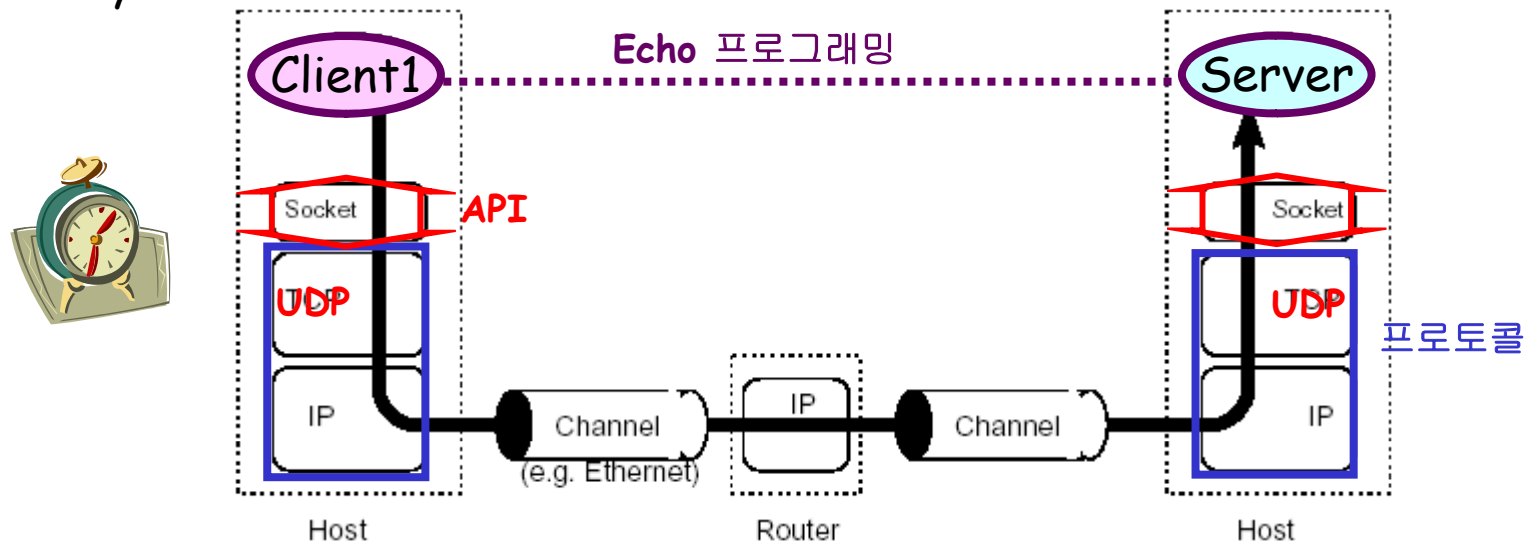


Figure 1.1: A TCP/IP Network