Data Communication (CE38442)

Computer Communication Laboratory
Dept. of Computer Science and Engineering
Chungnam National University

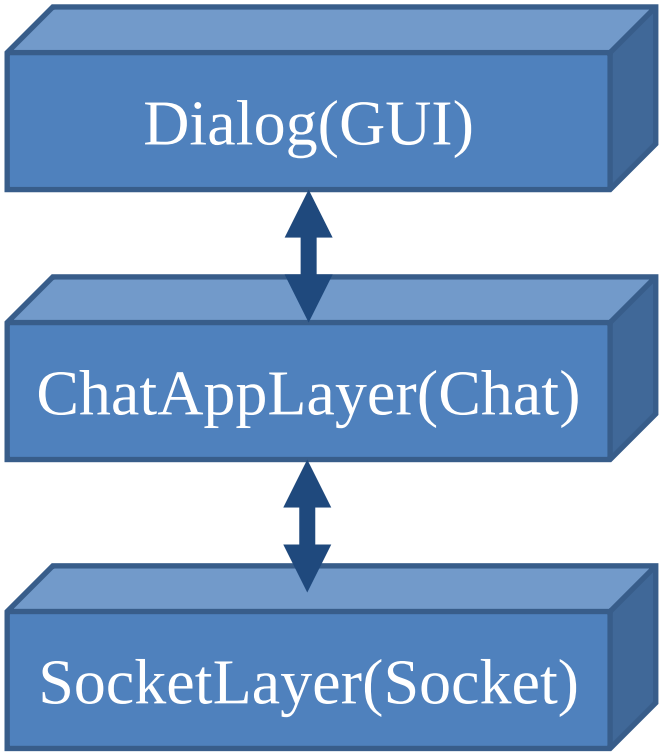
DongYeong Seo - 00, 01반

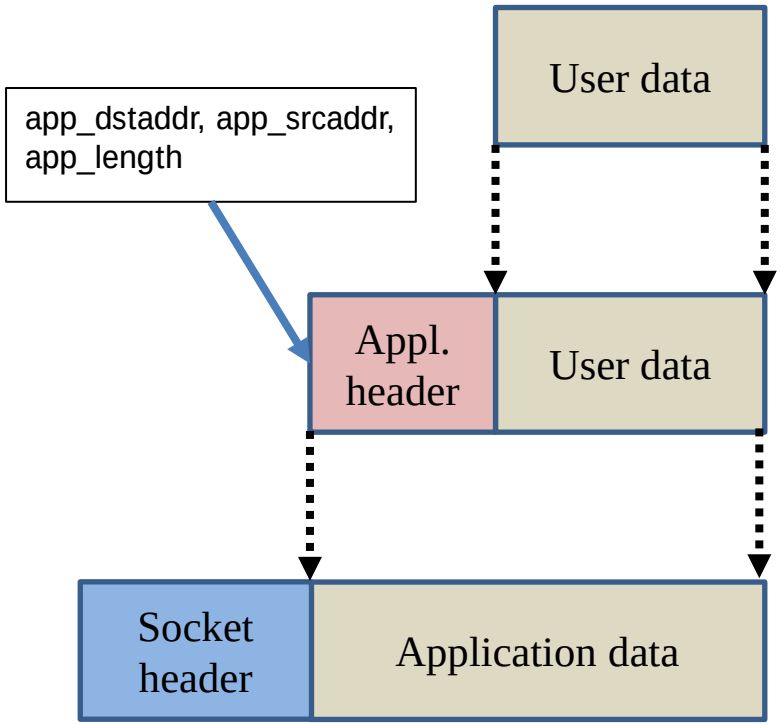
Contents

- Inter-Process Communication Implement
  - IPC (Inter-Process Communication)
- HomeWork#02 (revision)

Inter-Process Communication Implement

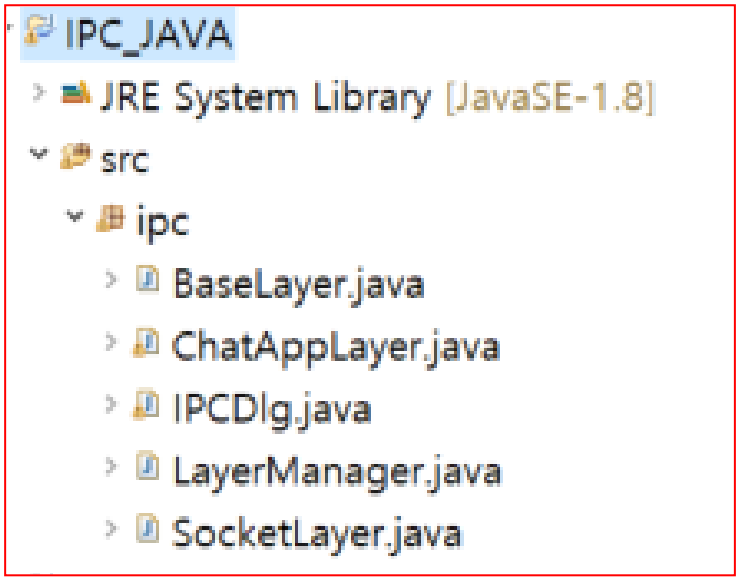
Hierarchical Architecture





IPC Implement

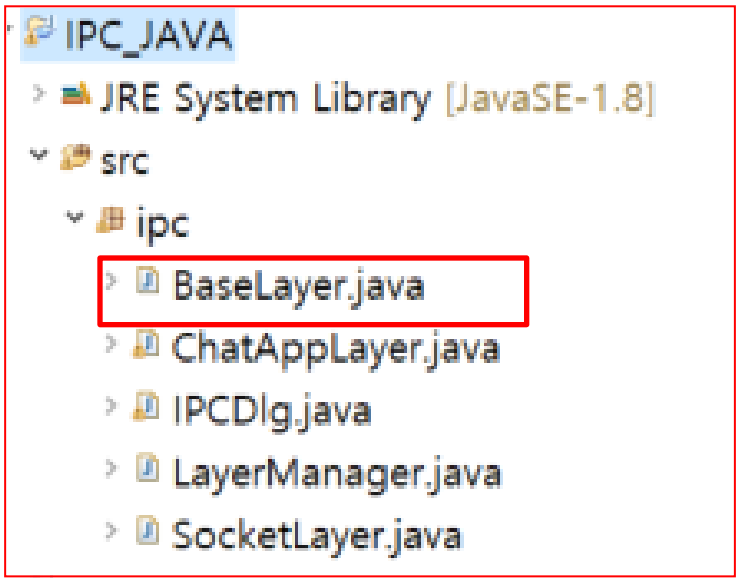
◈ IPC 구현



1. BaseLayer
2. ChatAppLyer
3. IPCDlg
4. LayerManager
5. SocketLayer

IPC Implement

◈ IPC 구현



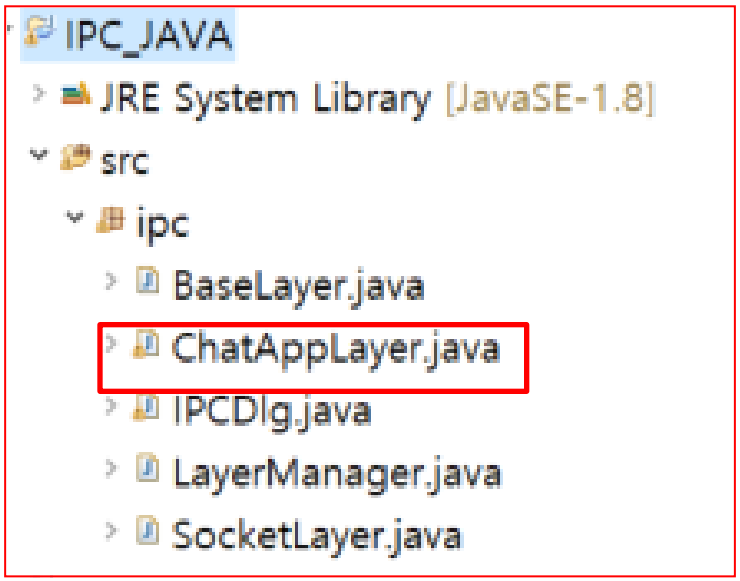
1. BaseLayer.java

- 각 Layer의 기본적인 틀을 잡아주고, 상- 하위 계층 사이에 통신을 가능하게 해주는 계층적 구조를 구현한 인터페이스

- BaseLayer를 상속받는 모든 클래스에서는 실제적으로 Send()와 Receive() 함수를 자신의 클래스에 맞게 정의 후 호출

IPC Implement

◈ IPC 구현



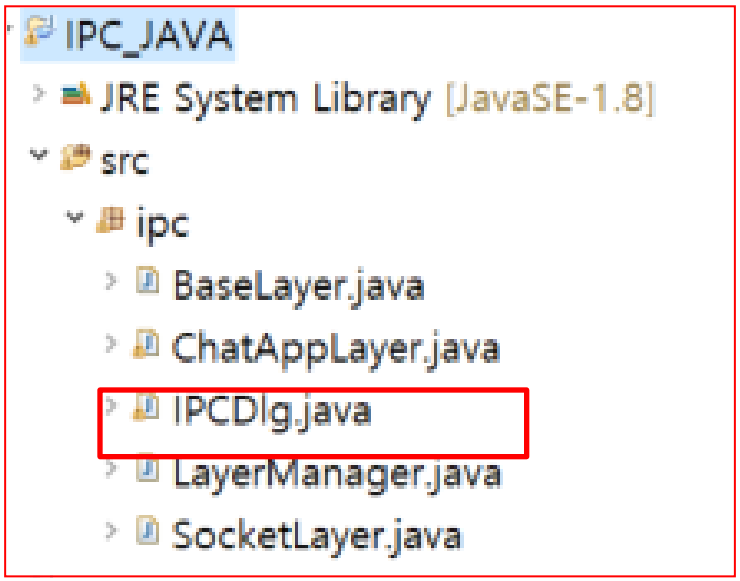
2. ChatAppLayer

- _CAPP_HEADER() :
  src, dst 포트번호와 그것들을
  보관할 Byte 배열의 크기를 저장

- Dialog에서 입력한 String값의
  Byte화 된 data에 헤더를 붙여주고
  소켓레이어로 보내는 과정에
  필요한 ResetHeader(),
  ObjToByte() 메소드 구현

IPC Implement

◈ IPC 구현

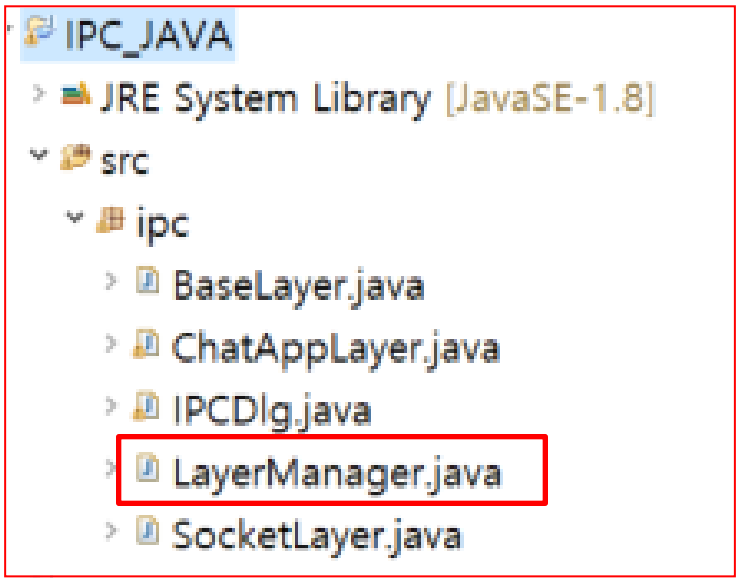


3. IPCDlg

- Main(String[] args){ }
  모든 레이어를 추가하고 연결하는 메소드를 불러 상하위 순서 지정
- setAddressListener:
  버튼 클릭 시 행동을 지정해주는 메소드
- Receive(byte[] input){}
  올려보낸 input을 수신측에서 잘 받았다고 표시해주는 함수

IPC Implement

◈ IPC 구현

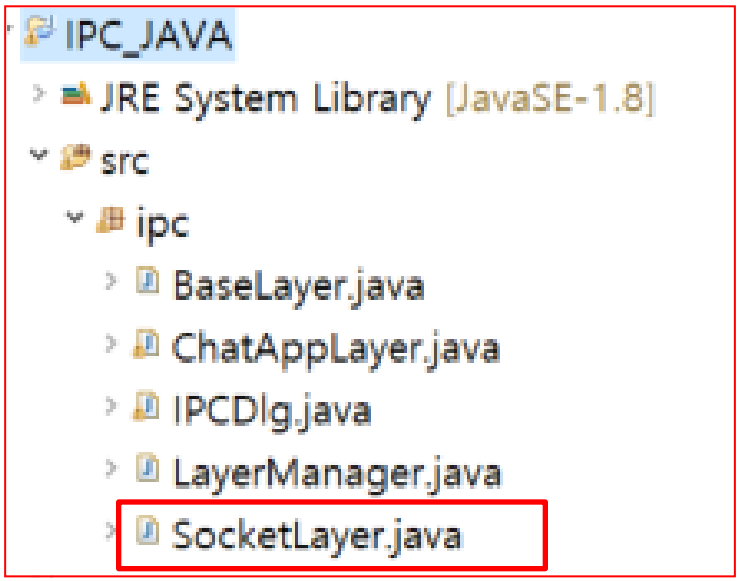


4. LayerManager

- 어떤 계층적 구조를 만들 수 있도록 프로토콜 관리를 담당하는 클래스
- IPC 과제를 위한 3가지 Layer(IPCDlg, ChatAppLayer, SocketLayer)를 연결해주는 객체

IPC Implement

◈ IPC 구현

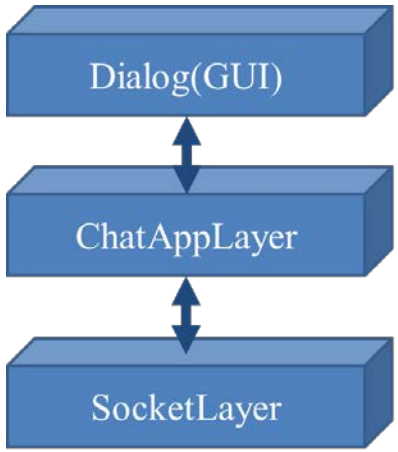


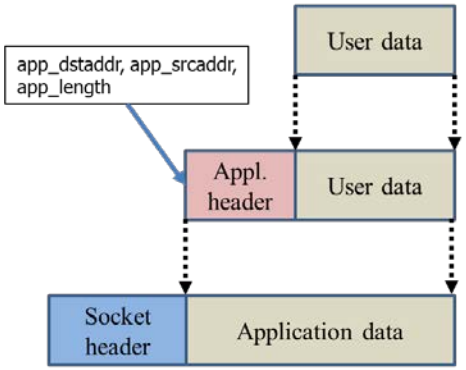
5. SocketLayer

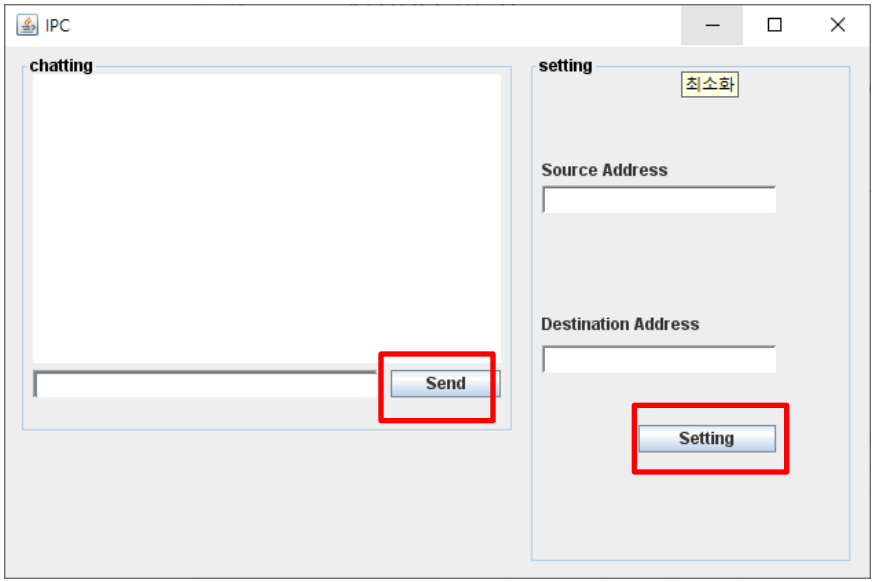
- setClinetPort(int dstAddress)
  setServerPort(int srcAddress)
  받은 포트번호드를 저장하는
  메소드
- Send() 함수를 통해 클라이언트
  초기화 및 input data를
  상대방에게 보내고
  상대방 프로세스에서는 Receive()
  함수를 통해 상위 게층으로
  올려보낼 준비를 함

IPC Implement

- 제공된 기본 소스에서 IPC 구현을 위해 해야 할 일
  - 1) Layer 연결 (IPCDlg.java)(LayerManager.java)
  - 2) Setting 버튼이 눌렸을 때의 동작 처리 (IPCDlg.java)(SocketLayer의 setClientPort, setServerPort)
  - 3) Send 버튼이 눌렸을 때의 동작 처리 (IPCDlg.java)
  - 4) 각 Layer의 send, receive 함수 구현(ChatAppLayer.java)
  - 5) ChatAPPLayer의 RemoveCappHeader 함수 구현(ChatAppLayer.java)







IPC Implement

◈ Layer 연결 (IPCDlg.java)

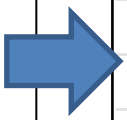
```
public static void main(String[] args) {
    // TODO Auto-generated method stub
    m_LayerMgr.AddLayer(new SocketLayer("Socket"));
    /*
    과제 빈칸 채우기
     ChatApp LayerManager에 추가

     */
    m_LayerMgr.AddLayer(new IPCDlg("GUI"));

    /*
    과제 ChatApp 연결하기 아래부분 수정

     */
    m_LayerMgr.ConnectLayers(" Socket ( *GUI ) ");
}
```

```
public static void main(String[] args) {

    // TODO Auto-generated method stub

    /* **********************fill in the blank************************************ */
    m_LayerMgr.AddLayer(new SocketLayer("Socket"));
    m_LayerMgr.AddLayer(new ChatAppLayer("Chat"));
    m_LayerMgr.AddLayer(new IPCDlg("GUI"));

    m_LayerMgr.ConnectLayers(" Socket ( *Chat ( *GUI ) ) ");

    /* **********************fill in the blank************************************ */

}
```

IPC Implement

◈ Layer 연결 (LayerManager.java)

```
public void ConnectLayers(String pcList){
    MakeList(pcList);
    LinkLayer(mp_sListHead);
}
```



```
LayerManager  (id=31)
")" (id=63)
LayerManager  (id=31)
")" (id=61)
LayerManager  (id=31)
")" (id=58)
LayerManager  (id=31)
"*GUI" (id=55)
LayerManager  (id=31)
"(" (id=52)
LayerManager  (id=31)
"*Chat" (id=48)
LayerManager  (id=31)
"(" (id=46)
LayerManager  (id=31)
"Socket" (id=37)
```

IPC Implement

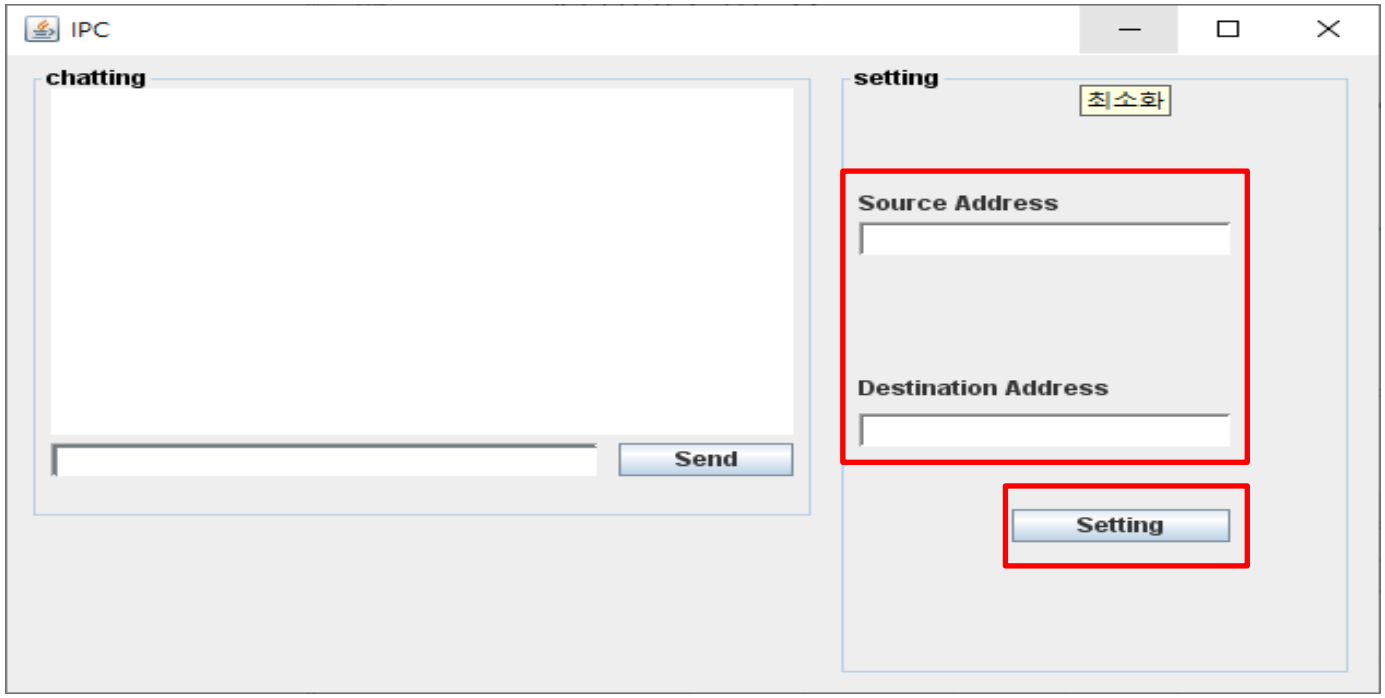
◈ Layer 연결 (LayerManager.java)

```
private void LinkLayer(_NODE pNode){
    BaseLayer pLayer = null;

    while(pNode != null){
        if( pLayer == null)
            pLayer = GetLayer (pNode.token);
        else{
            if(pNode.token.equals("("))
                Push (pLayer);
            else if(pNode.token.equals(")"))
                Pop();
            else{
                char cMode = pNode.token.charAt(0);//*GUI -> *
                String pcName = pNode.token.substring(1, pNode.token.length());//*GUI - > GUI

                pLayer = GetLayer (pcName);//Name으로 Layer 가져오기

                switch(cMode){
                case '*':
                    Top().SetUpperUnderLayer( pLayer );
                    break;
                case '+':
                    Top().SetUpperLayer( pLayer );
                    break;
                case '-':
                    Top().SetUnderLayer( pLayer );
                    break;
                }
            }
        }

        pNode = pNode.next;

    }
}
```

IPC Implement

◈ Setting 버튼이 눌렸을 때의 동작 처리(IPCDlg.java)

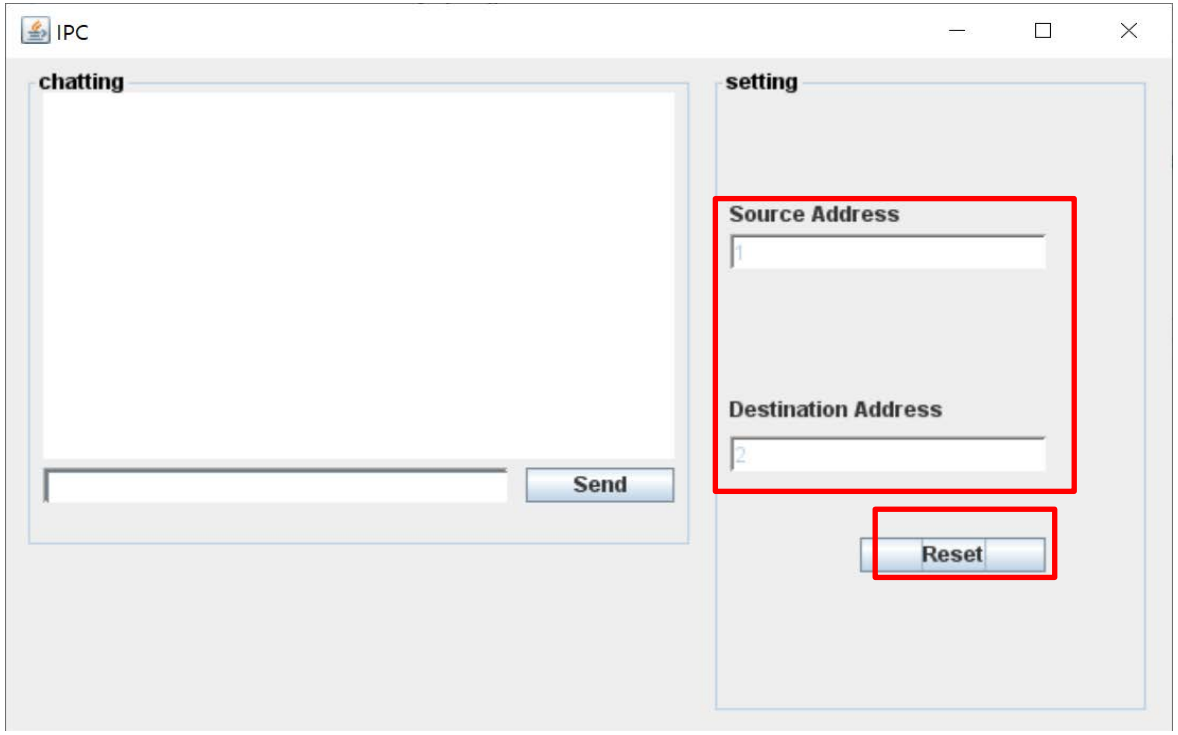
1. SrcAddress의 text를 ChatAppLayer header에 저장
2. DstAddress의 text를 ChatAppLayer header에 저장
3. SrcAddress의 text를 SocketLayer Server port에 저장
4. DstAddress의 text를 SocketLayer Client port에 저장
5. SocketLayer의 서버를 실행시킴(SocketLayer Thread 동작)



IPC Implement

◈ Setting 버튼이 눌렸을 때의 동작 처리 (IPCDlg.java)(cont.)

6. “Setting” Button을 “Reset” Button으로 변경
7. SrcAddress, DstAddress의 값 변경 못하게 설정
8. “Reset” Button을 누를 시 SrcAddress, DstAddress 의 text를 공백으로 변경
9. “Reset” Button - > “Setting” Button



IPC Implement

◈ Setting 버튼이 눌렸을 때의 동작 처리(IPCDlg.java)

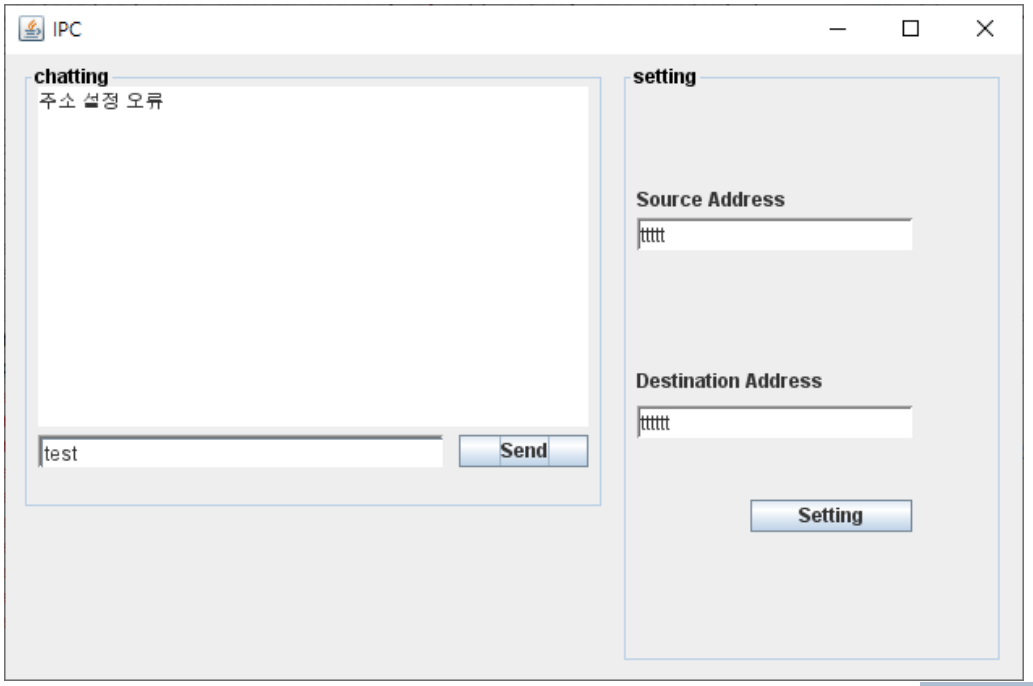
```
class setAddressListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        /*
         *
         * 과제 Setting 버튼과 Send 버튼을 누를 시 행동
         *
         * Setting 버튼 누를 시 SocketLayer에서 포트 설정
         *
         */
    }
}
```

```
class setAddressListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {

        if(e.getSource() == Setting_Button){
            if(Setting_Button.getText() == "Reset"){

                srcAddress.setText("");
                dstAddress.setText("");
                dstAddress.setEnabled(true);
                srcAddress.setEnabled(true);
                Setting_Button.setText("Setting");
            }
            else {
                String Ssrc = srcAddress.getText();
                String Sdst = dstAddress.getText();
                int src = Integer.parseInt(Ssrc);
                int dst = Integer.parseInt(Sdst);
                ((SocketLayer) m_LayerMgr.GetLayer("Socket")).setClientPort(dst);
                ((SocketLayer) m_LayerMgr.GetLayer("Socket")).setServerPort(src);
                ((ChatAppLayer) m_LayerMgr.GetLayer("Chat")).SetEnetSrcAddress(src);
                ((ChatAppLayer) m_LayerMgr.GetLayer("Chat")).SetEnetDstAddress(dst);
                ((SocketLayer) m_LayerMgr.GetLayer("Socket")).Receive();
                Setting_Button.setText("Reset");
                dstAddress.setEnabled(false);
                srcAddress.setEnabled(false);
            }
        }
        if(e.getSource() == Chat_send_Button){
            if(Setting_Button.getText() == "Reset"){
                String writtenChat = ChattingWrite.getText();
                ChattingArea.append("[Send] : " + writtenChat  + "\n");
                byte[] sendingChat = writtenChat.getBytes();
                ((ChatAppLayer) m_LayerMgr.GetLayer("Chat")).Send(sendingChat, sendingChat.length);
            }
            else {
                ChattingArea.append("주소 설정 오류");
            }
        }
    }
}
```

IPC Implement

◈ Send 버튼이 눌렸을 때의 동작 처리(IPCDlg.java)

1. Setting Button이 “Reset”인지 확인
2. ChattingWrite에 적은 Text를 ChattingArea에 보여준다.
3. ChatAppLayer에 Send()호출해서 String을 Byte형식으로 변경해서 보낸다.
4. 주소 값이 없으면 “주소 설정 오류” MessageDialog를 띄운다.



IPC Implement

- Header class(ChatAppLayer.java)

```
private class _CAPP_HEADER {
    int capp_src;
    int capp_dst;
    byte[] capp_totlen;


    public _CAPP_HEADER() {
        this.capp_src = 0x00000000;
        this.capp_dst = 0x00000000;
        this.capp_totlen = new byte[2];

    }
}

_CAPP_HEADER m_sHeader = new _CAPP_HEADER();
```

IPC Implement

◈ ObjToByte 함수 구현 (ChatAppLayer.java)

```
public byte[] ObjToByte(_CAPP_HEADER Header, byte[] input, int length) {
    byte[] buf = new byte[length + 10];
    byte[] srctemp = intToByte4(Header.capp_src);
    byte[] dsttemp = intToByte4(Header.capp_dst);

    buf[0] = dsttemp[0];
    buf[1] = dsttemp[1];
    buf[2] = srctemp[0];
    buf[3] = srctemp[1];
    buf[4] = dsttemp[0];
    buf[5] = dsttemp[1];
    buf[6] = srctemp[0];
    buf[7] = srctemp[1];
    buf[8] = (byte) (length % 256);
    buf[9] = (byte) (length / 256);

    for (int i = 0; i < length; i++)
        buf[10 + i] = input[i];

    return buf;
}
```

IPC Implement

◈ intToByte4 함수 구현 (ChatAppLayer.java)

```
byte[] intToByte4(int value) {
        byte[] temp = new byte[4];

        temp[0] |= (byte) ((value & 0xFF000000) >> 24);
        temp[1] |= (byte) ((value & 0xFF0000) >> 16);
        temp[2] |= (byte) ((value & 0xFF00) >> 8);
        temp[3] |= (byte) (value & 0xFF);
        return temp;
    }
```

IPC Implement

◈ Layer의 Send,Receive함수 구현 (ChatAppLayer.ja

```
public boolean Send(byte[] input, int length) {
    /*
     * 과제
     *
     */
    return true;
}

public byte[] RemoveCappHeader(byte[] input, int length) {

/*
 * 과제
 *
 *
 * */
    return input;// 변경하세요 필요하시면
}
```

```
public boolean Send(byte[] input, int length) {

    byte[] bytes = ObjToByte(m_sHeader, input, length);
    this.GetUnderLayer().Send(bytes, length +10);

    return true;
}

public byte[] RemoveCappHeader(byte[] input, int length) {

    for (int i = 0; i < (input.length - 10); i++) {
        input[i] = input[i+10];
    }

    return input;
}
```

HOMEWORK #2 (REV)

Homework #2

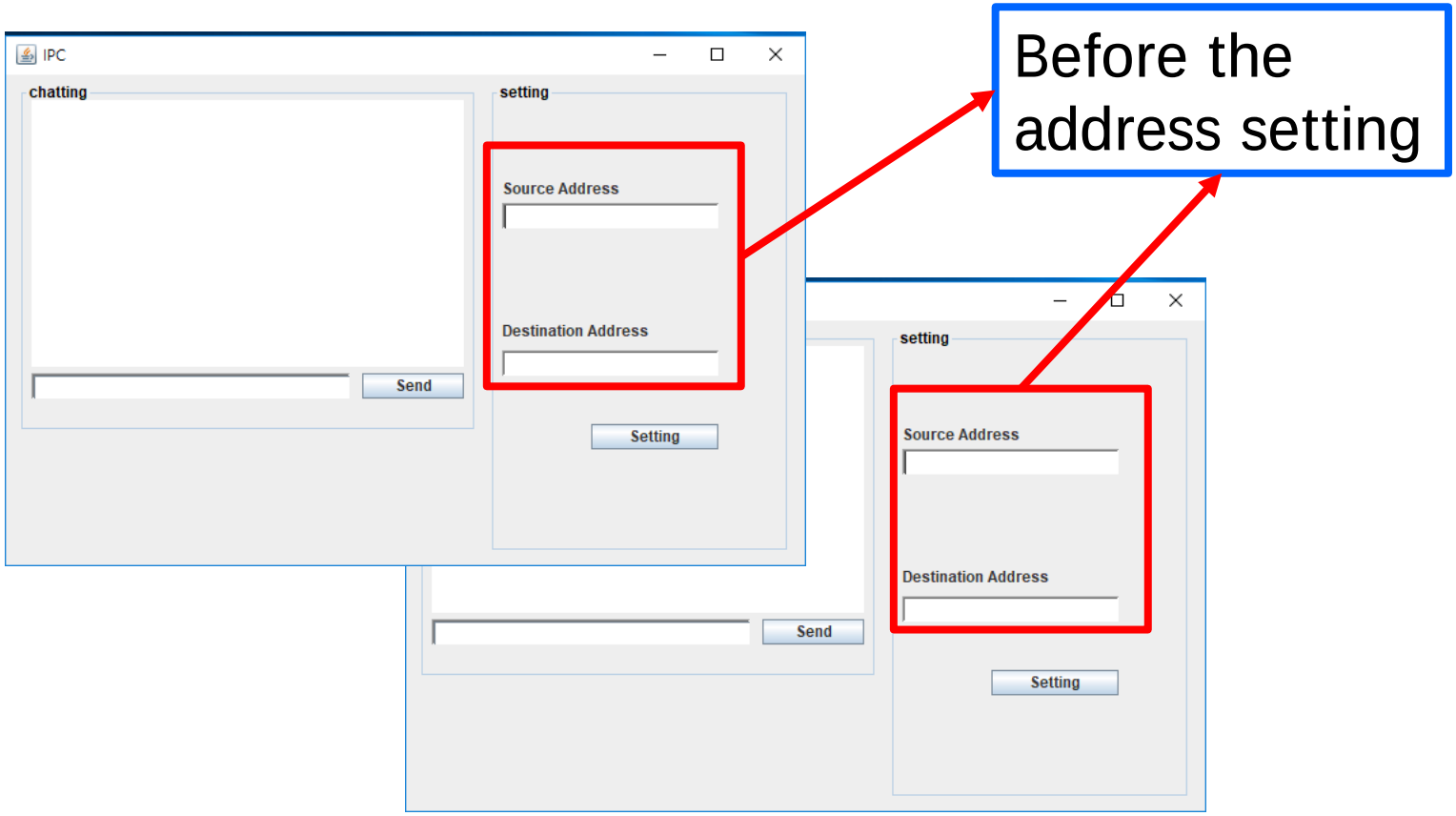
- Protocol Stack에 따른 메시지 송신
  - ChatAPPLayer ➔ SocketLayer
    - ChatAppLayer:
      Data에 ChatApp header를 붙여서 SocketLayer에게 전체를 전달
    - SocketLayer:
      ChatAppLayer로 전송 받은 데이터를 Server로 전송
- Protocol Stack에 따른 메시지 수신
  - ChatAPPLayer ➔ SocketLayer
    - SocketLayer:
      Client로부터 전달된 메시지를 받은 후 데이터를 ChatAppLayer로 전달
    - ChatAppLayer:
      SocketLayer로부터 받은 데이터 중에서 ChatApp header를 분리한 후 destination address가 자신이 맞는지 확인 후 Dialog로 전달

Homework #2

- 실행 예제
  - 프로그램 화면에 나타나야 할 정보를 정의한다.
    - Source / Destination Address, Message etc.
  - 메시지 출력 방식
    - 1번 Process에서 2번 Process로 정상적인 메시지 전송이 이루어진 경우
      - Ex.) 안녕하세요 -> “[SEND] : 안녕하세요.” (1번 화면)
        “[RECV] : 안녕하세요.” (2번 화면)
  - Setting 버튼을 누를 시 SocketLayer 서버 포트가 정해집니다.

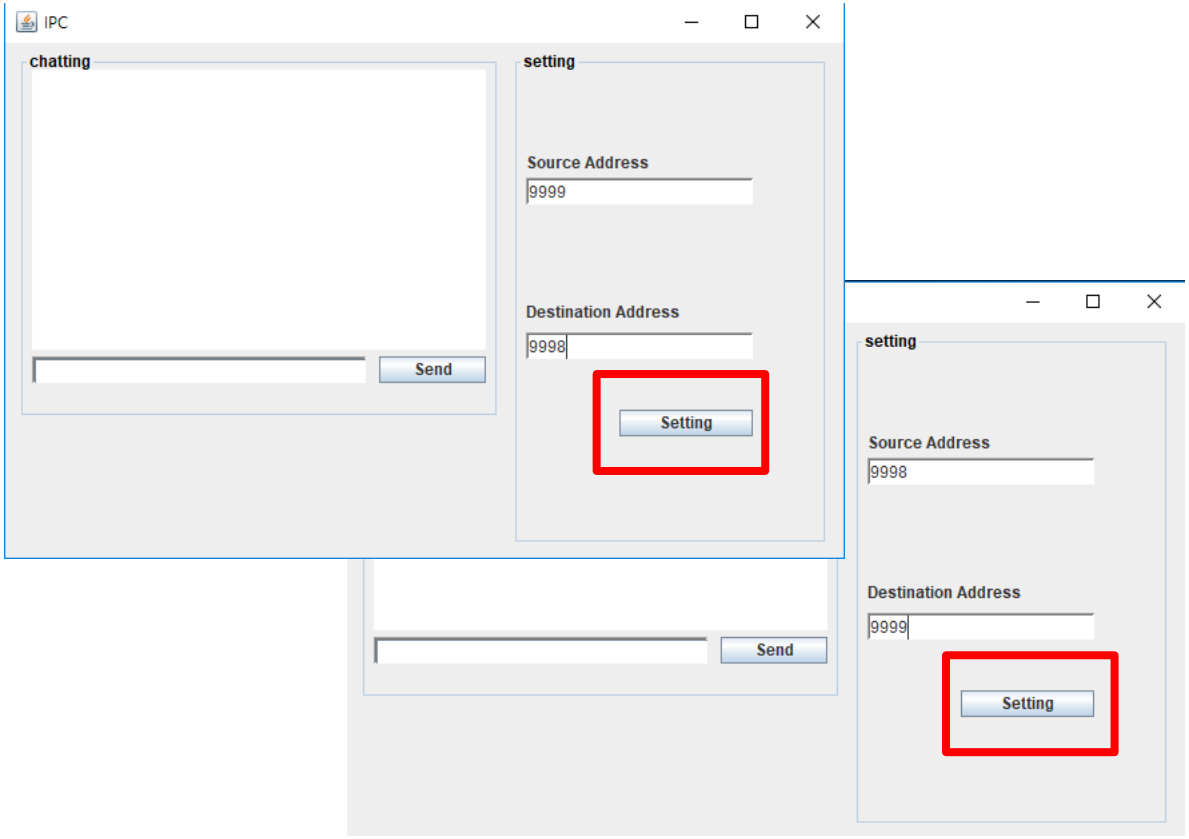
Homework #2

- Source & Destination Address 입력



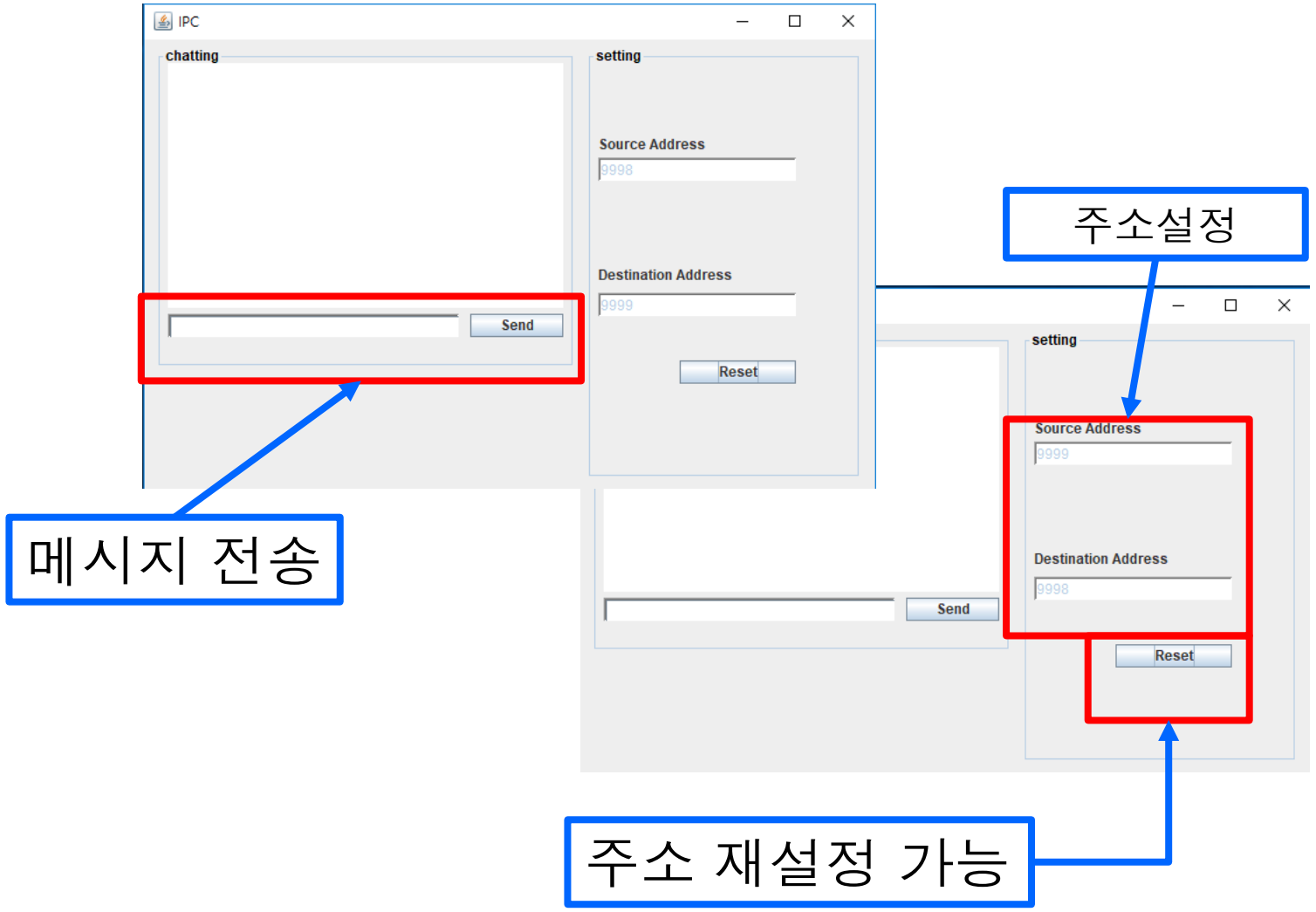
Homework #2

◈ 입력한 주소는 서버의 포트 번호이기에 서로 반대로 적용



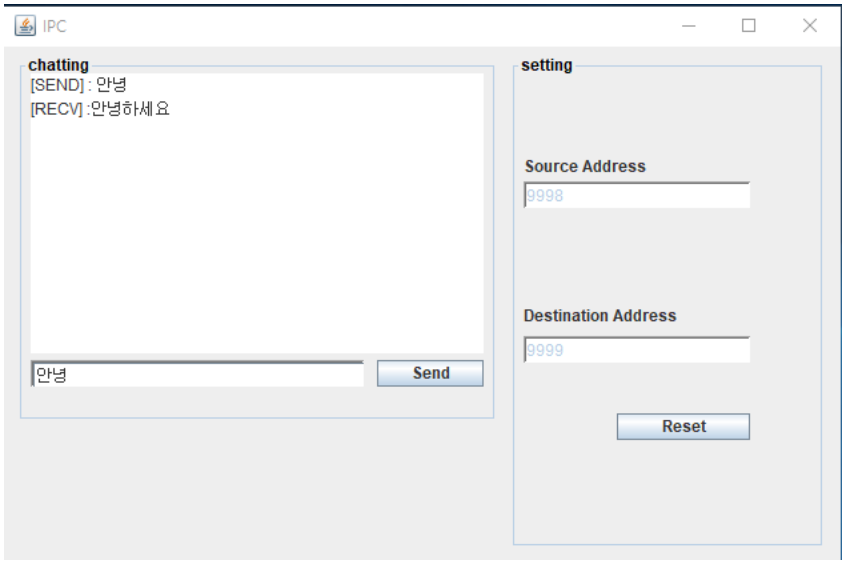
Homework #2

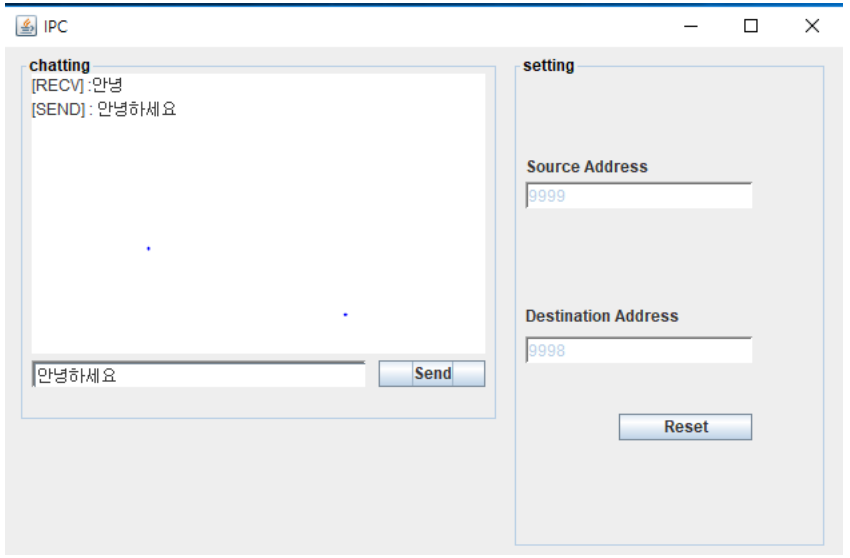
◈ 텍스트 입력 모드



Homework #2

◈ 텍스트 입력 시 결과 화면





Homework #2

- 제출 파일
  - 프로젝트 파일 (Eclipse project) + ~~.exe파일~~
  - 보고서
    - 실습 개요
      - 실습 목적
    - 프로토콜 스택
      - 구조 설명 (프로토콜의 역할 등)
        - 소켓 통신은 TCP/IP 계층에서 어디에 있는지 구조 설명
      - 실제 TCP/IP 계층에서의 소켓이 어디에 있는지 조사 및 이해한 내용을 설명 (절대적인 정답 채점하는 것이 아닌 조사해 보는 것에 의의를 둠)
    - 구현 설명
    - 실행 결과
      - 프로그램 결과 화면
  - 압축해서 제출(프로젝트 파일 + 보고서(Word, 한글, PDF 중 택 1))
- 제출일
  - **4/10 (**금) 23:59
  - 사이버캠퍼스에 제출

Introduction (cont.)

- 과제 제출 안내
  - 제출 방법
    - 이러닝 시스템
      - http://e-learn.cnu.ac.kr
    - 강의자료 다운로드
      - 마이페이지>>나의강의실>>[데이터통신 00분반]>>강의자료
    - 압축 파일 제출
      - 과제 제출 안내 준수
      - 개별
        - 마이페이지>>나의강의실>>[데이터통신 00분반]>>과제제출
        - hw[과제번호]_[학번]_[이름].zip (e.g. hw01_201701234_이름.zip)

감사합니다.