중앙대학교 CHUNG-ANG UNIVERSITY

Project 6

과목명	소프트웨어 프로젝트
담당교수	박창윤 교수님
학부	소프트웨어학부
학번	20186274
이름	김명승
제출일	2019.06.10 (월)

목차

Ⅰ. 서론

밑의 코드는 Project 5에서 제작했던 GUI class의 일부분이다.

```
public class GUI extends JFrame{

    private JFrame friendListFrame;

    public void createFriendListGUI() {
        friendListFrame = new JFrame("친구 목록");
        friendListFrame.setSize(1000, 500);
...
```

상속의 개념을 정확히 이해하지 못했기 때문에 JFrame을 상속하고 또 다시 JFrame을 인스턴스화 하여 사용하였다. 또 하나의 class안에 친구 목록 Frame과 친구 추가 Frame을 동시에 작성하였기 때문에 매우 구조화되지 못한 c스타일의 코드가 되었다. 때문에 이번 Project에서 저번 Project의 코드들을 전부 갈아 엎고 상속과 구조화 개념을 포함하여 처음부터 작성하기로 하였다.

Ⅱ. 본론

1. 설계

Project5에서는 하나의 GUI class에서 두개의 JFrame을 열도록 코드를 작성했지만, 이제 상속의 개념을 활용할 것임으로 하나의 상속된 class는 하나의 JFrame만 열도록 작성할 것이다.

우선 친구 목록을 보여주는 Frame에 관한 정보들을 기술한 class인 FriendListGUI를 작성해보았다. JFrame을 extends하여 작성하였다.

```
public class FriendListGUI extends JFrame {

    private JPanel jpnButtons;
    private JButton[] jbtArray;
    private FriendList friendList;

    public FriendListGUI(FriendList friendList) {
        super();    //JFrame을 상속받는다
        this.friendList = friendList;
        this.setTitle("친구 목록");
        this.setSize(WIDTH, HEIGHT);
        this.setLayout(new BorderLayout());

        jbtArray = new JButton[4]; //Frame 우측에 붙을 Button들의 Array
        jpnButtons = new JPanel(); //Button들을 추가시킬 JPanel
        setButtons(jbtArray);
        this.add(jpnButtons, BorderLayout.EAST);
    }

    public void setButtons(JButton[] buttonArray) {
        jpnButtons.setLayout(new GridLayout(buttonNameArray.length, 1));
        ...
```

super(), this 키워드를 사용하여 상속한 JFrame을 활용하는 것까지는 성공적이었다. 하지만 JPanel인 jpnButtons에 JButton들을 추가시키는 메서드인 setButtons()를 구현하면서 무언가 이상한 점을 느꼈다. 위 setButtons()메서드는 오로지 JPanel jpnButtons만을 위한 메서드가 된 것이므로, 재활용을 할 수 없기에 굉장히 비효율적인 코드가 된 것이다. 그래서 다시 JPanel을 argument로 받고, 이를 return하는 코드로 작성해보았다.

```
...
        jpnButtons = new JPanel( );
        jpnButtons = setButtons(jpnButtons, jbtArray);
        this.add(jpnButtons, BorderLayout.EAST);
    }

    public JPanel setButtons(JPanel jpn, JButton[] buttonArray) {
        jpn.setLayout(new GridLayout(buttonNameArray.length, 1));
        ...
        return jpn
     }
```

이 코드 또한 작성하며 의문점을 갖게 되었다. setButtons은 JPanel에 관한 행동을 취하는 것인데, 그 행동이 JFrame의 메서드로 작성된 것이다.

이 문제점들을 해결할 방법을 생각하다, JPanel에 관한 모든 행동(메서드)들을 새로운 class에 기술하는 방법을 생각해냈다. FriendListGUI가 JFrame을 extends한 덕분에 JFrame에 관련된 행동들을 메서드로 호출했듯이, JPanel을 상속하는 class를 만들어 JPanel에 관련된 행동들을 기술하고, 그 class를 FriendListGUI에서 JPanel대신 사용할 것이다. 또 JFrame에 추가할 모든 요소들은 JPanel 형태를 사용할 것임으로, 새로 상속하여 만든 class에서 모든 요소들을 관리 할 수 있다.

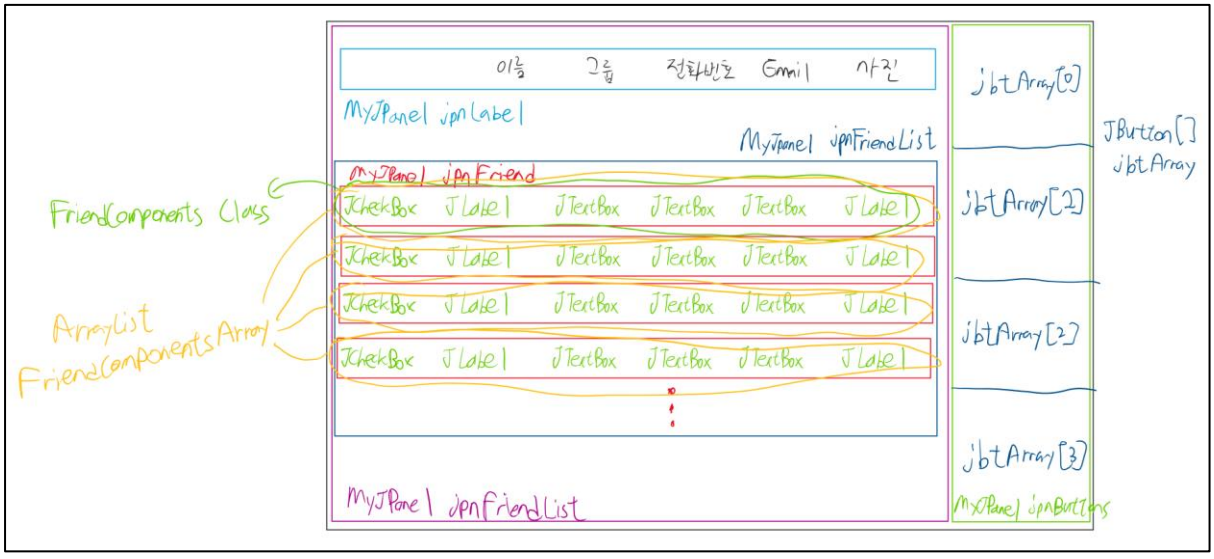


그림 1. JFrame FriendListGUI 구성요소

2. FriendList class

Friend의 배열이 아닌 ArrayList를 사용하여 FriendList class를 다시 작성했다. 또 PrintWriter을 사용하여 FriendList를 data파일로 내보내는 makeFile()메서드도 작성하였다.

```
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;

public class FriendList {

    private ArrayList<Friend> listOfFriend;

    public FriendList() {
        listOfFriend = new ArrayList<Friend>();
    }

    public void addFriend(Friend friend) {
        listOfFriend.add(friend);
    }

    public void deleteFriend(int i) {
        listOfFriend.remove(i);
    }

    public int numFriends() {
        return listOfFriend.size();
    }

    public Friend getFriend(int i) {
        return listOfFriend.get(i);
    }

    public void makeFile() throws IOException {
        PrintWriter output = new PrintWriter("friendlist-norm.data");
        String friendLine;

        output.println(new String(
                "// Friend List\r\n" + "// Format::  Name : Group Number : Phone :
email : profile picture\r\n"
                        + "// Group Number Family:1 , School:2 , Job 3 "));
        for (int i = 0; i < numFriends(); i++) {
            friendLine = new String(getFriend(i).getName() + ":" + getFriend(i).
             getGroup() + ":" + getFriend(i).getPhoneNumber() + ":"+ getFriend(i).
             getEmailAddress() + ":" + getFriend(i).getPhoto());
            output.println("//");
            output.println(friendLine);
        }
        output.close();
    }

    public boolean isNameOverlap(String name) {
        for (int i = 0; i < numFriends(); i++) {
            if (name.equals(getFriend(i).getName())) {
                return true;
            }
        }
        return false;
    }
}
```

3. MyJPanel class

```java
import java.awt.Dimension;
import java.awt.GridLayout;
import java.util.ArrayList;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.SwingConstants;

/**
 * javax.swing.JPanel를 확장하여 작성한 class이다
 * JFrame을 상속하는 FriendListGUI, AddFriendGUI에 add할 JPanel들에게 필요한 method들을 확장하여 작성하였다
 */
public class MyJPanel extends JPanel {
    private static final int NAME = 0, GROUP = 1, PHONE = 2, EMAIL = 3, PHOTO = 4;
    private static final boolean NOBLANK = false, BLANK = true;

    public MyJPanel() {
        super(); // 부모Class의 생성자
    }

    /**
     * @brief JPanel에 Button들을 위에서 아래로 추가한다
     * @param JButton[] buttonArray 추가할 Button들의 배열
     * @param String[] buttonNameArray 추가할 Button들의 이름들의 배열
     */
    public void setButtons(JButton[] buttonArray, String[] buttonNameArray) {
        this.setLayout(new GridLayout(buttonNameArray.length, 1));

        for (int i = 0; i < buttonNameArray.length; i++) {
            buttonArray[i] = new JButton(buttonNameArray[i]);
            this.add(buttonArray[i]);
        }
    }

    /**
     * @brief JPanel에 Friend 정보 Label들을 추가한다
     * @param boolean blankFlag 앞에 공백 포함여부
     */
    public void setLabels(boolean blankFlag) {
        Friend friend = new Friend();

        if (blankFlag == NOBLANK) {
            this.setLayout(new GridLayout(1, friend.getNumOfInfo()));
        }
        else {
            this.setLayout(new GridLayout(1, friend.getNumOfInfo() + 1));
            this.add(new JLabel(""));
        }
        this.add(new JLabel("이름", SwingConstants.CENTER));
        this.add(new JLabel("그룹", SwingConstants.CENTER));
        this.add(new JLabel("전화번호", SwingConstants.CENTER));
        this.add(new JLabel("Email", SwingConstants.CENTER));
        this.add(new JLabel("사진", SwingConstants.CENTER));
    }
```

```
    /**
     * @brief JPanel에 Friend의 정보 한 줄을 추가한다
     * @param FriendComponents friendComponents 하나의 friend 정보를 보여줄 component
     */
    public void setFriend(FriendComponents friendComponents) {
        Friend friend = new Friend();
        this.setLayout(new GridLayout(1, friend.getNumOfInfo() + 1));

        this.add(friendComponents.jcb);
        this.add(friendComponents.jlbName);
        this.add(friendComponents.jtfGroup);
        this.add(friendComponents.jtfPhoneNumber);
        this.add(friendComponents.jtfEmailAddress);
        this.add(friendComponents.jlbPhoto);
    }

    /**
     * @brief JPanel에 FriendList의 정보들을 추가한다
     * @param ArrayList<FriendComponents> friendComponentsArray 하나의 friend 정보를 보여줄
     *       components들의 모임인 FriendComponents class의 ArrayList
     */
    public void setFriendList(ArrayList<FriendComponents> friendComponentsArray) {
        MyJPanel jpnFriend, jpnLabels = new MyJPanel(), jpnEmpty = new MyJPanel();

        this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));
        jpnLabels.setMaximumSize(new Dimension(Integer.MAX_VALUE, 1000));
        jpnLabels.setLabels(BLANK);
        this.add(jpnLabels);
        // friend의 component들을 JPanel에 추가하는 것을 반복하여 모든 친구들을 JPanel에 추가한다
        for (int i = 0; i < friendComponentsArray.size(); i++) {
            jpnFriend = new MyJPanel();
            jpnFriend.setMaximumSize(new Dimension(Integer.MAX_VALUE, 1000));
            jpnFriend.setFriend(friendComponentsArray.get(i));
            this.add(jpnFriend);
        }
        this.add(jpnEmpty);
    }

    /**
     * @brief AddFriendGUI class에서 사용할 JTextBox들을 JPanel에 추가한다
     * @param JTextField[] jtfArray JPanel에 추가할 JTextBox들의 배열
     */
    public void setAddFriendTextField(JTextField[] jtfArray) {
        MyJPanel jpnLabels = new MyJPanel(), jpnJtfArray = new MyJPanel();
        jpnLabels.setLabels(NOBLANK);
        this.add(jpnLabels);

        jpnJtfArray.setLayout(new GridLayout(1, jtfArray.length));
        // 사용자가 입력해야 할 format을 default값으로 설정
        for (int i = 0; i < jtfArray.length; i++) {
            if (i == GROUP)
                jtfArray[i] = new JTextField("number");
            else if (i == PHONE)
                jtfArray[i] = new JTextField("with hyphen(x-x-x)");
            else if (i == EMAIL)
                jtfArray[i] = new JTextField("(x@x.x)");
            else
                jtfArray[i] = new JTextField();

            jpnJtfArray.add(jtfArray[i]);
        }
        this.add(jpnJtfArray);
    }
}
```

4. FriendListGUI class

```java
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.IOException;
import java.util.ArrayList;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.SwingConstants;

/**
 * 친구목록을 출력하는 JFrame에 관한 정보들을 기술한 class
 * javax.swing.JFrame을 상속한다
 */
public class FriendListGUI extends JFrame {

    private final static int WIDTH = 1000, HEIGHT = 500,
                             ADD = 0, DELETE = 1, MODIFY = 2, SAVE_FILE = 3;
    private final String[] buttonNameArray = { "Add", "Delete", "Modify", "Save File" };

    private MyJPanel jpnButtons, jpnFriendList;
    private JButton[] jbtArray;
    //FriendComponents: Panel의 한 줄에 Friend정보를 출력할 component들을 변수로 가지는 class
    private ArrayList<FriendComponents> friendComponentsArray;
    private FriendList friendList;

    /**
     * @brief JFrame을 초기화하고 창을 연다.
     * @param FriendList friendList JFrame에서 보여줄 FriendList
     */
    public FriendListGUI(FriendList friendList) {
        super();    //JFrame을 상속받는다
        this.friendList = friendList;
        this.setTitle("친구 목록");
        this.setSize(WIDTH, HEIGHT);
        this.setLayout(new BorderLayout());

        friendComponentsArray = new ArrayList<FriendComponents>();
        for (int i = 0; i < friendList.numFriends(); i++) {
            //Friend정보를 FriendComponets에 set하고 array에 추가시킨다
            addfriendComponentsArray(friendList.getFriend(i));
        }

        jbtArray = new JButton[buttonNameArray.length];    //Frame 우측에 붙을 Button들의 Array
        jpnButtons = new MyJPanel();    //Button들을 추가시킬 JPanel
        jpnButtons.setButtons(jbtArray, buttonNameArray);
        this.add(jpnButtons, BorderLayout.EAST);
        for (int i = 0; i < jbtArray.length; i++) {
            jbtArray[i].addActionListener(new Listener());
        }

        jpnFriendList = new MyJPanel();//Frame 좌측에 붙을 FriendList의 정보들
        jpnFriendList.setFriendList(friendComponentsArray);
        this.add(jpnFriendList, BorderLayout.CENTER);

        this.setVisible(true);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
```

```java
/**
 * @brief friendComponentsArray에 Friend의 정보가 담긴 FriendComponents를 추가시킨다
 * @param Friend friend Array에 추가시킬 FriendComponents의 정보가 담긴 Friend
 */
public void addfriendComponentsArray(Friend friend) {
    FriendComponents temp = new FriendComponents();

    temp.jlbName.setText(friend.getName());
    temp.jlbName.setHorizontalAlignment(SwingConstants.CENTER);
    temp.jtfGroup.setText(friend.getGroup());
    temp.jtfPhoneNumber.setText(friend.getPhoneNumber());
    temp.jtfEmailAddress.setText(friend.getEmailAddress());
    temp.jlbPhoto.setText(friend.getPhoto());
    temp.jlbPhoto.setHorizontalAlignment(SwingConstants.CENTER);

    //각 component들에 actionListener를 추가한다
    temp.jcb.addActionListener(new Listener());
    temp.jtfGroup.addActionListener(new Listener());
    temp.jtfEmailAddress.addActionListener(new Listener());
    temp.jtfPhoneNumber.addActionListener(new Listener());

    friendComponentsArray.add(temp);
}

    /**
 * @brief FriendListGUI의 FriendList를 반환한다
 */
public FriendList getFriendList() {
    return friendList;
}

/**
 * @brief AddFriendGUI에서 입력된 Friend를 FriendListGUI의 FriendList에 추가시킨다
 * @param Friend friend FriendList에 추가시킬 Friend
 */
public void addToFriendList(Friend friend) {
    friendList.addFriend(friend);
}

/**
 * @brief Frame 좌측에 위치한 jpnFriendList Panel을 업데이트하기 위해 Panel을 지우고 다시 생성한다.
 */
public void updateJpnFriendList() {
    this.remove(jpnFriendList);
    jpnFriendList = new MyJPanel();
    jpnFriendList.setFriendList(friendComponentsArray);
    this.add(jpnFriendList, BorderLayout.CENTER);
    this.setVisible(true);
}

/**
 * interface인 ActionListener를 implements하여 components들의 event들을 처리하는 class
 */
class Listener implements ActionListener {

    public void actionPerformed(ActionEvent e) {

        //add버튼이 눌렸을 시
        if (e.getSource() == jbtArray[ADD]) {
            AddFriendGUI addFriendGUI = new AddFriendGUI(FriendListGUI.this);
        }

        //delete버튼이 눌렸을 시
```

```
            else if (e.getSource() == jbtArray[DELETE]) {
                for (int i = 0; i < friendComponentsArray.size(); i++) {
                    //CheckBox가 선택됐을시에만 delete 진행
                    if (friendComponentsArray.get(i).jcb.isSelected()) {
                        friendList.deleteFriend(i);
                        friendComponentsArray.remove(i);
                        JOptionPane.showMessageDialog(null, "Delete Finish");
                        updateJpnFriendList();
                        break;
                    }
                }
            }

            //modify버튼이 눌렸을 시
            else if (e.getSource() == jbtArray[MODIFY]) {
                boolean normalDataFlag = false;
                int temp = 0;


                for (int i = 0; i < friendComponentsArray.size(); i++) {
                    //CheckBox가 선택됐을시에만 modify 진행
                    if (friendComponentsArray.get(i).jcb.isSelected()) {
                        normalDataFlag = true;

                        //jtf
                        if (friendList.getFriend(i).isStringInt
                                (friendComponentsArray.get(i).jtfGroup.getText())) {
                            friendList.getFriend(i).setGroup
                                    (friendComponentsArray.get(i).jtfGroup.getText());
                        }
                        else {
                            JOptionPane.showMessageDialog(null, "Group is not a correct
format(Only number)","Format Error", JOptionPane.WARNING_MESSAGE);
                            //잘못된 데이터 타입이 입력됐음으로 TextBox를 기존 data로 변경
                            friendComponentsArray.get(i).jtfGroup.setText
                                    (friendList.getFriend(i).getGroup());
                            normalDataFlag = false;
                        }

                        if (friendList.getFriend(i).isStringPhoneNumberFormat
                                (friendComponentsArray.get(i).jtfPhoneNumber.getText())) {
                            friendList.getFriend(i).setPhoneNumber
                                    (friendComponentsArray.get(i).jtfPhoneNumber.getText());
                        }
                        else {
                            JOptionPane.showMessageDialog(null, "Phone number is not a correct
format", "Format Error", JOptionPane.WARNING_MESSAGE);
                            friendComponentsArray.get(i).jtfPhoneNumber.setText
                                    (friendList.getFriend(i).getPhoneNumber());
                            normalDataFlag = false;
                        }

                        if (friendList.getFriend(i).isStringEmailFormat
                                (friendComponentsArray.get(i).jtfEmailAddress.getText())) {
                            friendList.getFriend(i).setEmailAddress
                                    (friendComponentsArray.get(i).jtfEmailAddress.getText());
                        }
                        else {
                            JOptionPane.showMessageDialog(null, "Email address is not a correct
format", "Format Error", JOptionPane.WARNING_MESSAGE);
                            friendComponentsArray.get(i).jtfEmailAddress.setText
                                    (friendList.getFriend(i).getEmailAddress());
                            normalDataFlag = false;
                        }
                        temp = i;
                        break;
                    }
```

```
                }
                if (normalDataFlag == true) {  //입력된 data가 올바른 형식이라면 modify 진행
                    friendComponentsArray.get(temp).jcb.setSelected(false);
                    friendComponentsArray.get(temp).setJtfEditible(false);
                    updateJpnFriendList();
                    JOptionPane.showMessageDialog(null, "Modify Finish");
                }
            }
            //save file버튼이 눌렸을 시
            else if (e.getSource() == jbtArray[SAVE_FILE]) {
                try {
                    friendList.makeFile();
                } catch (IOException e1) {
                    System.out.println("file save Error!");
                }
                JOptionPane.showMessageDialog(null, "Save Finish");
            }

            //어떠한 CheckBox가 클릭되었다면
            else {
                updateJpnFriendList();

                for (int i = 0; i < friendComponentsArray.size(); i++) {
                    if (e.getSource() == friendComponentsArray.get(i).jcb) {

                        //CheckBox가 선택되었다면 그 줄의 TextBox들을 수정 가능하게 변경
                        if(friendComponentsArray.get(i).jcb.isSelected()==true) {
                            friendComponentsArray.get(i).setJtfEditible(true);
                        }
                        //CheckBox가 해제되었다면 그 줄의 TextBox들을 수정 불가능하게 변경
                        else if(friendComponentsArray.get(i).jcb.isSelected()==false) {
                            friendComponentsArray.get(i).setJtfEditible(false);
                        }
                        //클릭된 CheckBox열을 제외한 모든 열들의 TextBox들을 수정불가능하게 변경
                        for (int j = 0; j < friendComponentsArray.size(); j++) {
                            if (i != j) {
                                friendComponentsArray.get(j).jcb.setSelected(false);
                                friendComponentsArray.get(j).setJtfEditible(false);

                                friendComponentsArray.get(j).jtfGroup.setText
                                    (friendList.getFriend(j).getGroup());
                                friendComponentsArray.get(j).jtfPhoneNumber.setText
                                    (friendList.getFriend(j).getPhoneNumber());
                                friendComponentsArray.get(j).jtfEmailAddress.setText
                                    (friendList.getFriend(j).getEmailAddress());
                            }
                        }
                        break;
                    }
                }
            }
        }
    }
}
```

Add 버튼이 눌렸을 시에는 AddFriendGUI JFrame을 새로 열어서 그 class에서 데이터를 처리하도록 하였다.

Delete 버튼이 눌렸을 시에는 우선 각 줄마다 JCheckBox를 확인하여 선택된 것이 있는지 확인하고, 만약 있다면 그 체크된 줄의 FriendList와 FriendComponentsArray를 지워 내부 데이터들을 먼저 업데이트 한다. 그리고 그 데이터들을 GUI로 보여주는 jpnFriendList를 업데이트하는 updateJpnFriendList() 메서드를 호출해주어 내부 데이터의 삭제, 화면의 재정비도 완료하였다.

```
if (friendComponentsArray.get(i).jcb.isSelected()) {
    friendList.deleteFriend(i);
    friendComponentsArray.remove(i);
    JOptionPane.showMessageDialog(null, "Delete Finish");
    updateJpnFriendList();
}
```

Modify 버튼도 Delete 버튼 클릭 시 행동과 비슷하다. 다만 데이터를 수정하기 전 데이터가 올바른 형식인지 확인해야 하기 때문에 확인하는 절차를 추가하였다. 먼저 형식을 체크하고 맞다면 수정한 내용이 담긴 JTextBox의 내용으로 내부 데이터인 FriendList을 수정한다. 형식이 맞지 않는다면 어떤 형식이 맞지 않는지 에러 창을 띄우고, 바꾸기 전 기존의 데이터를 JTextBox에 넣는다. 그리고 normalDataFlag를 false로 설정하여 updateJpnFriendList() 메서드를 호출하지 않도록 한다.

```
if (friendList.getFriend(i).isStringInt(friendComponentsArray.get(i).jtfGroup.getText())) {
    friendList.getFriend(i).setGroup(friendComponentsArray.get(i).jtfGroup.getText());
}
else {
    JOptionPane.showMessageDialog(null, "Group is not a correct format(Only number)","Format
Error", JOptionPane.WARNING_MESSAGE);
    friendComponentsArray.get(i).jtfGroup.setText(friendList.getFriend(i).getGroup());
    normalDataFlag = false;
}
```

위 상황들과 Save File 버튼 이외의 event 가 발생했다는 것은 JCheckBox 에 대한 event 가 발생했다는 것이다. JCheckBox 가 복수 개 선택 되는 것을 방지하기 위해, 하나의 JCheckBox 에 event 가 발생하면 모든 다른 JCheckBox 들은 선택을 해제하는 방식을 선택하였다.

5. FriendComponents class

```
import javax.swing.JCheckBox;
import javax.swing.JLabel;
import javax.swing.JTextField;

public class FriendComponents {
    JCheckBox jcb = new JCheckBox();
    JLabel jlbName = new JLabel();
    JTextField jtfGroup = new JTextField();
    JTextField jtfPhoneNumber = new JTextField();
    JTextField jtfEmailAddress = new JTextField();
    JLabel jlbPhoto = new JLabel();

    FriendComponents(){
        jtfGroup.setEditable(false);
        jtfPhoneNumber.setEditable(false);
        jtfEmailAddress.setEditable(false);
    }

    public void setJtfEditible(boolean flag) {
        jtfGroup.setEditable(flag);
        jtfPhoneNumber.setEditable(flag);
        jtfEmailAddress.setEditable(flag);
    }
}
```

JFrame FriendListGUI의 왼쪽 JPanel jpnFriendList의 한 줄인 JPanel jpnFriend에 add할 component들이다. 초기에는 모든 JTextField를 수정할 수 없도록 하였고, setJtfEditible() 메서드를 사용하여 JCheckBox의 상태에 따라 JTextField의 수정여부를 결정하도록 하였다.

6. AddFriendGUI class

친구 목록에 친구를 추가하기 위한 Frame에 관한 정보들을 기술한 class이다. FriendListGUI와 마찬가지로 JFrame을 extends하여 작성하였다.

```
import java.awt.BorderLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.FocusEvent;
import java.awt.event.FocusListener;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JTextField;

/**
 * 친구목록에 친구를 추가하는 JFrame에 관한 정보들을 기술한 class
 * javax.swing.JFrame을 상속한다
 */
public class AddFriendGUI extends JFrame {

    private static final int WIDTH = 800, HEIGHT = 200,
                             NAME = 0, GROUP = 1, PHONE = 2, EMAIL = 3, PHOTO = 4;
    private JButton jbtDone;
    private JTextField[] jtfFriendInfos;
    private MyJPanel jpnAddFriend;
    private Friend friend = new Friend();
    private FriendListGUI friendListGUI;
    private boolean normalDataFlag = true;

    /**
     * @brief JFrame을 초기화하고 창을 연다.
     * @param FriendListGUI friendListGUI 현재 Frame을 호출시킨 FriendListGUI의 정보
     */
    public AddFriendGUI(FriendListGUI friendListGUI) {
        super();
        this.friendListGUI = friendListGUI;
        this.setTitle("추가할 친구 정보");
        this.setSize(WIDTH, HEIGHT);
        this.setLayout(new BorderLayout());

        jbtDone = new JButton("Done");
        this.add(jbtDone, BorderLayout.EAST);
        jbtDone.addActionListener(new Listener());

        //정보를 입력할 TextBox의 ArrayList
        jtfFriendInfos = new JTextField[friend.getNumOfInfo()];
        jpnAddFriend = new MyJPanel();
        jpnAddFriend.setLayout(new GridLayout(2, 1));
        jpnAddFriend.setAddFriendTextField(jtfFriendInfos);
        for (int i = 0; i < jtfFriendInfos.length; i++) {
            jtfFriendInfos[i].addFocusListener(new FListener());
        }

        this.add(jpnAddFriend, BorderLayout.CENTER);
        this.setVisible(true);
    }


    /**
     * interface인 ActionListener를 implements하여 components들의 event들을 처리하는 class
     */
```

```
    class Listener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            //사용자가 입력을 마치고 done버튼이 눌렸을 시
            if (e.getSource() == jbtDone) {
                if (e.getSource() == jbtDone) {
                    if(!friendListGUI.getFriendList().isNameOverlap
                         (jtfFriendInfos[NAME].getText())) {
                    friend.setName(jtfFriendInfos[NAME].getText());
                } else {
                    JOptionPane.showMessageDialog(null, "Name is Overlapped", "경고
메시지", JOptionPane.WARNING_MESSAGE);
                    normalDataFlag = false;
                }

                if (friend.isStringInt(jtfFriendInfos[GROUP].getText())) {
                    friend.setGroup(jtfFriendInfos[GROUP].getText());
                } else {
                    JOptionPane.showMessageDialog(null, "Group is not a correct format(Only
number)", "경고 메시지", JOptionPane.WARNING_MESSAGE);
                    normalDataFlag = false;
                }

                if (friend.isStringPhoneNumberFormat(jtfFriendInfos[PHONE].getText())) {
                    friend.setPhoneNumber(jtfFriendInfos[PHONE].getText());
                } else {
                    JOptionPane.showMessageDialog(null, "Phone number is not a correct
format", "경고 메시지", JOptionPane.WARNING_MESSAGE);
                    normalDataFlag = false;
                }

                if (friend.isStringEmailFormat(jtfFriendInfos[EMAIL].getText())) {
                    friend.setEmailAddress(jtfFriendInfos[EMAIL].getText());
                } else {
                    JOptionPane.showMessageDialog(null, "Email address is not a correct
format", "경고 메시지", JOptionPane.WARNING_MESSAGE);
                    normalDataFlag = false;
                }

                friend.setPhoto(jtfFriendInfos[PHOTO].getText());

                if (normalDataFlag == true) {
                 //입력된 data가 올바른 형식이라면 FriendList에 add 진행
                    friendListGUI.addToFriendList(friend);
                    friendListGUI.addfriendComponentsArray(friend);
                    friendListGUI.updateJpnFriendList();
                    AddFriendGUI.this.dispose();    //창을 닫는다

                } else {
                    normalDataFlag = true;
                }

            }
        }
    }

    /**
     * interface인 FocusListener를 implements하여 JTextBox들의 focus event들을 처리하는 class
     */
    class FListener implements FocusListener {
        public void focusPerformed(FocusEvent e) {
        }

        public void focusLost(FocusEvent e) {
        }
```

```
        public void focusGained(FocusEvent e) {
        //TextBox가 눌렸을 시 입력 format을 작성해놓은 default값들을 지움
            if (e.getSource() == jtfFriendInfos[NAME]) {
                jtfFriendInfos[NAME].setText("");
            } else if (e.getSource() == jtfFriendInfos[GROUP]) {
                jtfFriendInfos[GROUP].setText("");
            } else if (e.getSource() == jtfFriendInfos[PHONE]) {
                jtfFriendInfos[PHONE].setText("");
            } else if (e.getSource() == jtfFriendInfos[EMAIL]) {
                jtfFriendInfos[EMAIL].setText("");
            } else if (e.getSource() == jtfFriendInfos[PHOTO]) {
                jtfFriendInfos[PHOTO].setText("");
            }
        }
    }
}
```

JFrame AddFriendGUI 는 JFrame FriendListGUI 의 Add 버튼을 눌렀을 시에만 인스턴스를 생성하도록 구현하였으며, 두 class 사이에서 FriendList 를 공유하기 위해 생성자로 이를 받는다.

Done 버튼이 눌렸을 시 FriendListGUI 의 Modify 와 비슷하게 format 을 확인하여 데이터를 추가하고 업데이트하는 모습을 볼 수 있었다. 다만 차이점을 찾는다면 사용자가 잘못 입력한 데이터를 볼 수 있도록 기존의 JTextBox 값을 바꾸지 않고 그대로 두었다.

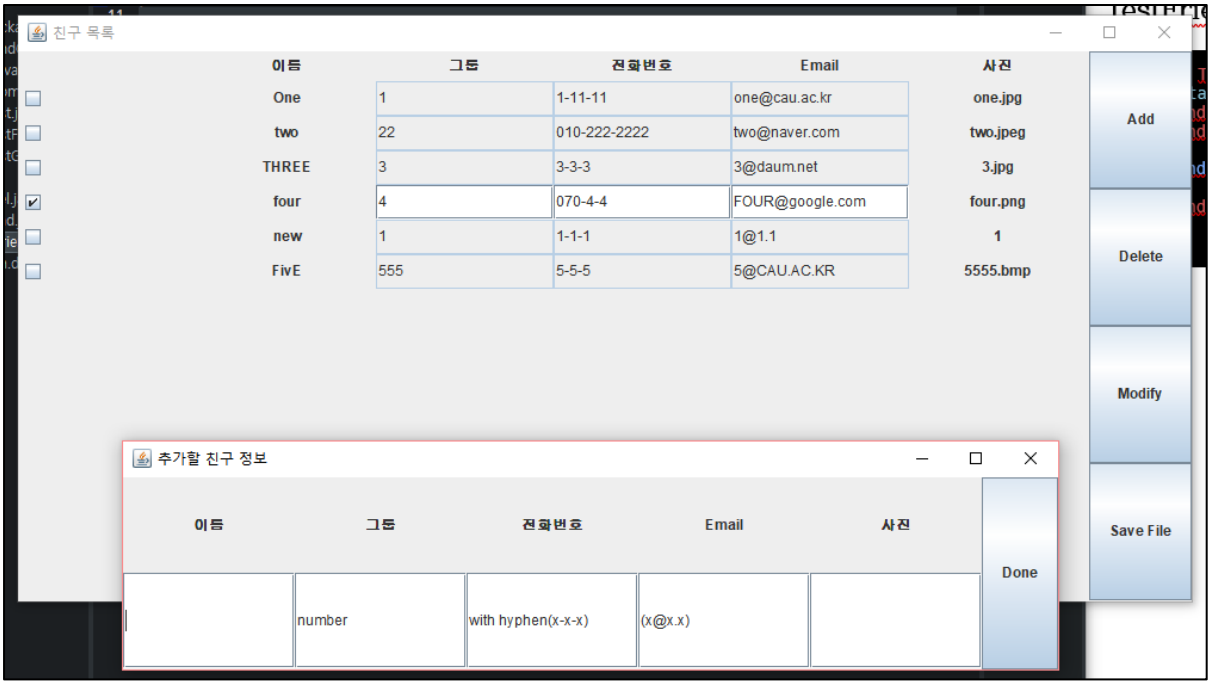
또 FocusListenenr 를 implements 한 새로운 내부 클래스를 만들었다. 그 이유는 초기에 사용자에게 데이터 format 을 안내하기 위해 JTextBox 에 default 로 설정해 놓은 문장들을 JTextBox 클릭 시 지워지게 하고자 였다.

7. TestFriend class

```
public class TestFriend {
    public static void main(String[] args) {
        FriendListFile friendListFile = new FriendListFile();
        FriendList friendList = new FriendList();

        friendList = friendListFile.readFileToList("friendlist-norm.data");

        FriendListGUI friendListGUI = new FriendListGUI(friendList);
    }
}
```



자세한 실행 케이스들은 TestFriend.mp4 영상파일로 제출하였습니다.

Ⅲ. 결론

평가 항목	학생 자체 평가 (리포트 해당 부분 표시 및 간단한 의견)	평가 (빈칸)	점수 (빈칸)
완성도 (동작 여부) - “초기” 동작 - “Add” 동작 - “Modify” 동작 - “Delete” 동작 - “Save” 동작 - 기타 비정상 동작 실험 (구현 한계 점검)	모든 행동들이 의도한대로 작동하는 것을 프로그램 실행 영상 파일을 통해 확인할 수 있다. format에 맞게 입력해야만 Add 및 Modify 가능, JCheckBox가 눌린 열만 Modify 및 Delete 가능하게 코드를 작성하였다.		
설계 노트 - 주요 결정사항 및 근거 - 한계/문제점 - 해결 방안 - 필수내용: * 프로그램 구성(Class 구조) * member visibility	친구 목록, 친구 추가하는 두 개의 창을 JFrame을 상속하는 두 개의 class로 설계하였다. 모든 요소들은 JPanel의 형태로 JFrame에 붙일 것이라 JPanel을 상속한 MyJPanel이라는 class를 만들었다. 이 class에서 모든 Panel의 행동을 조정한다. 모든 멤버변수는 private로 설정하였고, 어떤 변수를 다른 클래스로 넘겨야 할 필요가 있을 시 set메서드 작성하였다.		
리포트 - 평가자 시각으로 리포트 검토 - 위의 평가 요소들이 명확하게 기술되었는가?	평가 요소들을 모두 작성하였고, 확실한 실행 모습을 보여주기 위해 영상파일을 첨부하였다.		
총평/계	프로그램 구조화, 전체적인 java의 의미와 사용방법을 배울 수 있었던 Project였다.		