[기말프로젝트 2차 보고서]

2019111677

김지연

I. 1차 보고서

[분석]

1. 출처 – 블로그

컴퓨터와 가위바위보 게임을 하는 프로그램

https://m.blog.naver.com/PostView.nhn?blogId=korn123&logNo=30097964803&proxyReferer=https:%2F%2Fwww.google.com%2F

2. 기존 프로그램의 클래스 구조 및 관계

클래스가 JFrame을 상속받으면서 동시에 Actionistener인테페이스도 구현하는 방법으로 되어있다. 클래스 안에 actionPerformed()가 정의되어 있어야 하며, 객체를 이벤트 리스너로 버튼에 등록해야 한다.

Main() – startMix()드를 호출하고(게임을 시작할 때 컴퓨터의 값을 결정해주기 위함), 객체 생성하여 호출

생성자 – 윈도우리스너 구현, start_display메소드 호출

Start_display() – 전체적인 UI 생성, 버튼에 이벤트 리스너 객체로 등록

actionPerformed() – 각각의 버튼을 눌렀을 때 호출할 메소드들 구현. 가위바위보 버튼은 getFight 메소드 호출, reset버튼은 리스트 초기화와 startMix() 호출(게임을 시작할 때 컴퓨터의 값을 결정해주기 위함), end버튼은 시스템 종료를 한다.

startMix() – 컴퓨터의 가위바위보를 결정. 랜덤함수를 사용한다.

getFight() – 컴퓨터와 사용자의 값을 비교하여 누가 지고 이겼는지 처리하여 결과 출력. 마지막에 startMix()를 호출하여 컴퓨터의 값을 다시 변경해준다.

3. 사용된 라이브러리의 종류 및 의미

프로그램을 구현하기 위해 swing과 awt 컴포넌트를 추가해 준다. Awt는 GUI컴포넌트를 위한 부모 클래스 들을 제공하고, awt.event는 GUI컴포넌트로부터 발생되는 이벤트를 처리하기 위한 클래스와 인터페이스를 가지고 있다. Swing은 버튼이나 텍스트 필드, 프레임 패널과 같은 GUI컴포넌트들을 가지고 있다.

RockPaperSissorsSwing클래스 안에 필요한 생성자, 메소드, 컴포넌트들을 구현해준다.

우선 필요한 변수들을 선언해준다. JButton(가위바위보, reset, end 버튼), user(사용자의 가위바위보를 담을 변수), com(컴퓨터의 값을 담을 변수), mixSu(컴퓨터가 내는 가위바위보를 초기화 하는 과정에서 값을 담을 변수), list를 선언한다.

생성자를 구현한다. 윈도우 창을 만들기 위하여 windowListener를 무명클래스로 정의하였다. Window어탭터 클래스를 이용하여 windowClosing만 재정의 할 수 있도록 한다. Try catch문은 setLookAndFeel을 구현하기 위해 사용하였다. 프로그램 전체의 UI의 모습을 변경하기 위해 사용하였다. 마지막으로 start_display메소드를 호출한다.

Start_display()를 구현한다. 컨테이너를 생성하고, layout은 null로 배경색은 하얀색으로 설정한다. 버튼 5개와 list를 생성한다. Layout이 null이기 때문에 버튼의 배치는 절대 위치로 배치한다. 컨테이너에 버튼과 리스트를 추가한다. 각각의 버튼에 이벤트 리스너를 객체로 등록한다.

리스너를 정의한다. 가위바위보 버튼을 눌렀다면 사용자가 선택한 값을 리스트에 출력하고 getFight()를 호출한다. reset버튼을 눌렀다면 list의 모든 값을 초기화하고 startMix()를 호출한다. 게임을 시작할 때 컴퓨터가 가위바위보를 랜덤으로 가지고 있어야 하기 때문이다.

startMix()를 구현한다. 컴퓨터가 내는 값을 랜덤으로 초기화하는 메소드이다. 랜덤함수를 사용하여 mixSu변수에는 0~2중에서 랜덤한 값이 들어가게 된다. mixSu의 값에 따라 컴퓨터가 가지는 가위바위보가 달라지게 된다.

getFight()를 구현한다. 이 메소드는 변수 2개를 비교하여 가위바위보를 연산한다. 컴퓨터가 가지는 값을 기준으로 사용자가 입력한 값을 비교하여 이겼는지, 졌는지, 비겼는 지를 list에 출력하여 보여준다. 연산을 끝낸 후에는 startMix()를 호출하여 컴퓨터가 가지고 있는 값을 변경한다.

4. 프로그램의 동작 및 실행과정

프로그램을 실행하면 가위바위보 버튼 중 한 개를 눌러 컴퓨터와 가위바위보를 한다. 가위바위보 결과는 list에 저장된다. reset버튼을 누르면 list가 초기화 되며 end버튼을 누르면 프로그램을 종료한다.

[소스코드]

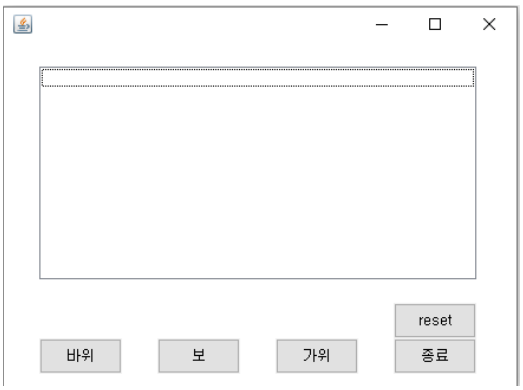
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class RockPaperSissorsSwing extends JFrame implements ActionListener {
	static String user = "";// 유저입력
	static String com = "";
	static int mixSu = 0; // 0~2 숫자를 랜덤으로 출력해서 저장하는 변수
	JButton rock;
	JButton paper;
	JButton sissors;
	JButton end;
	JButton reset;
	static List list;

	public static void main(String args[]) {
		startMix();
		RockPaperSissorsSwing rps = new RockPaperSissorsSwing();
		rps.setSize(450, 330);
		rps.setVisible(true);
	}

	// 초기화 메서드
	public RockPaperSissorsSwing() {

		this.addWindowListener(new WindowAdapter() {
			public void windowClosing(WindowEvent e) {
				System.exit(0);
			}
		});
		try {

	UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFe
el");

	SwingUtilities.updateComponentTreeUI(RockPaperSissorsSwing.this);
		} catch (Exception e) {
		}

		start_display();
	}

	// 사용자 UI 생성

	public void start_display() {
		Container cpane = getContentPane();
		cpane.setLayout(null);
		Color bg = new Color(255, 255, 255);
		cpane.setBackground(bg);

		rock = new JButton("바위");
		paper = new JButton("보");
```

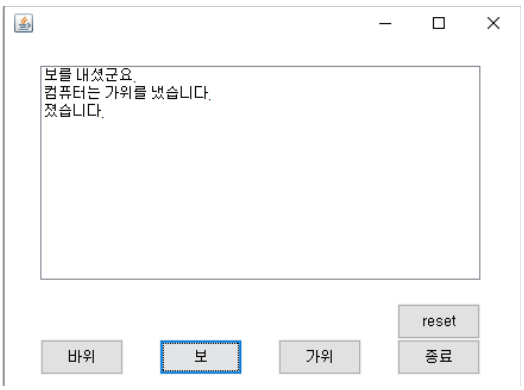
```
		sissors = new JButton("가위");
		end = new JButton("종료");
		reset = new JButton("reset");
		list = new List();

		rock.setBounds(30, 250, 70, 30);
		paper.setBounds(130, 250, 70, 30);
		sissors.setBounds(230, 250, 70, 30);
		end.setBounds(330, 250, 70, 30);
		list.setBounds(30, 20, 370, 180);
		reset.setBounds(330, 220, 70, 30);

		cpane.add(rock);
		cpane.add(paper);
		cpane.add(sissors);
		cpane.add(end);
		cpane.add(list);
		cpane.add(reset);

		rock.addActionListener(this);
		paper.addActionListener(this);
		sissors.addActionListener(this);
		end.addActionListener(this);
		reset.addActionListener(this);
	}

// 리스너 정의

	public void actionPerformed(ActionEvent e) {

		if (e.getSource() == rock) {
			list.add("바위를 내셨군요.");
			getFight(com, "바위");
		} else if (e.getSource() == paper) {
			list.add("보를 내셨군요.");
			getFight(com, "보");
		} else if (e.getSource() == sissors) {
			list.add("가위를 내셨군요.");
			getFight(com, "가위");
		} else if (e.getSource() == reset) {
			list.removeAll();
			startMix();
		} else if (e.getSource() == end) {
			System.exit(0);
		}
	}

	// 컴퓨터가 내는 가위,바위,보 초기화
	static void startMix() {
		mixSu = (int) (Math.random() * 3);
		switch (mixSu) {
		case 0:
			com = "가위";
```

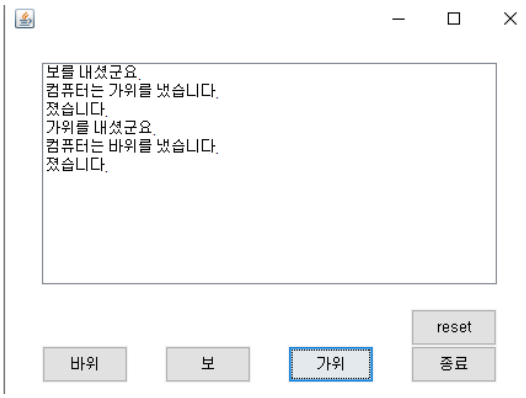
```
			break;
		case 1:
			com = "바위";
			break;
		default:
			com = "보";
			break;
		}
}

// 가위바위보 처리 메서드

static void getFight(String com, String user) {
		if (com.equals("바위")) {
			if (user.equals("바위")) {
				list.add("컴퓨터는 바위를 냈습니다.");
				list.add("비겼습니다.");
			} else if (user.equals("보")) {
				list.add("컴퓨터는 바위를 냈습니다.");
				list.add("이겼습니다.");
			} else {
				list.add("컴퓨터는 바위를 냈습니다.");
				list.add("졌습니다.");
			}
		}

		if (com.equals("가위")) {
			if (user.equals("바위")) {
				list.add("컴퓨터는 가위를 냈습니다.");
				list.add("이겼습니다.");
			} else if (user.equals("보")) {
				list.add("컴퓨터는 가위를 냈습니다.");
				list.add("졌습니다.");
			} else {
				list.add("컴퓨터는 가위를 냈습니다.");
				list.add("비겼습니다.");
			}
		}
		if (com.equals("보")) {
			if (user.equals("바위")) {
				list.add("컴퓨터는 보를 냈습니다.");
				list.add("졌습니다.");
			} else if (user.equals("보")) {
				list.add("컴퓨터는 보를 냈습니다.");
				list.add("비겼습니다.");
```

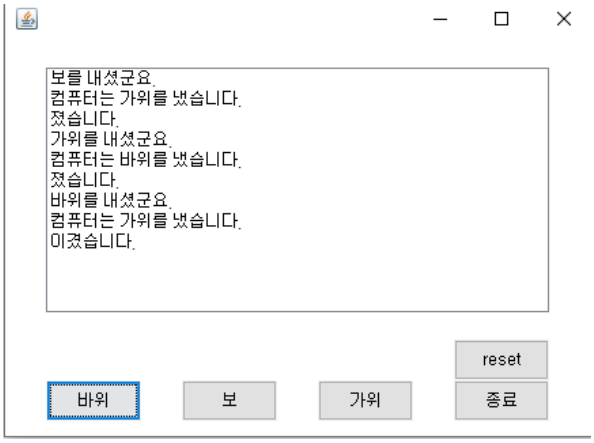
```
                } else {
                        list.add("컴퓨터는 보를 냈습니다.");
                        list.add("이겼습니다.");
                }
        }
        startMix(); // 가위바위보가 1회 끝나면 컴퓨터가 내는 가위바위보를
변경한다.
    }
}
```

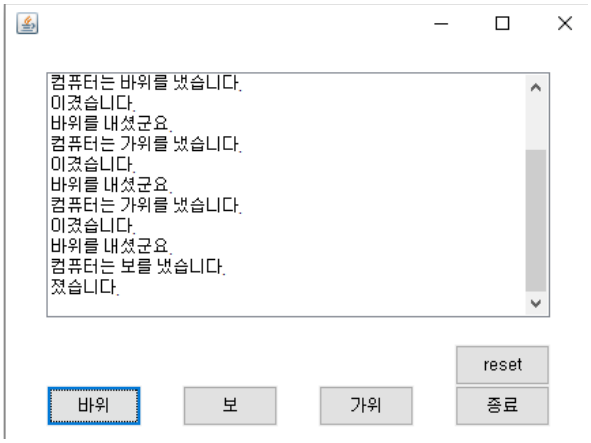
[실행결과]

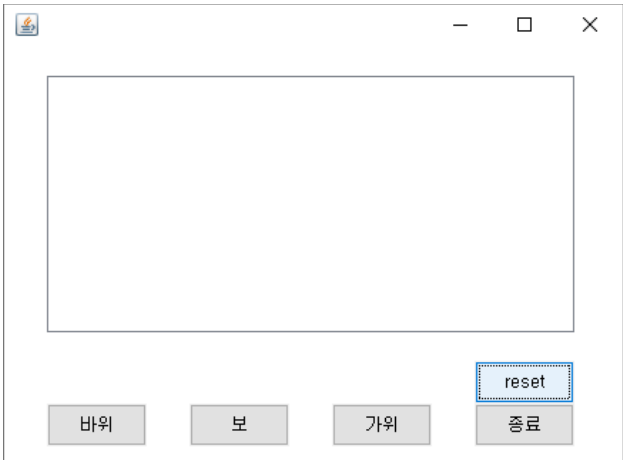







보를 내셨군요.
컴퓨터는 가위를 냈습니다.
졌습니다.
가위를 내셨군요.
컴퓨터는 바위를 냈습니다.
졌습니다.
바위를 내셨군요.
컴퓨터는 가위를 냈습니다.
이겼습니다.
reset
바위
보
가위
종료


컴퓨터는 바위를 냈습니다.
이겼습니다.
바위를 내셨군요.
컴퓨터는 가위를 냈습니다.
이겼습니다.
바위를 내셨군요.
컴퓨터는 가위를 냈습니다.
이겼습니다.
바위를 내셨군요.
컴퓨터는 보를 냈습니다.
졌습니다.
reset
바위
보
가위
종료


reset
바위
보
가위
종료

[기획의도]

1. 레이아웃을 절대위치가 아닌 배치관리자를 설정하여 배치하기

-> 절대위치는 수정이 쉽지 않고, 배치관리자를 사용하면 폰트의 크기, 컨테이너의 크기 변경 등 장점이 더 많기 때문이다.

2. 가위바위보를 시각적으로도 결과를 알 수 있도록 이미지를 추가한다.

-> 기존 프로그램에서는 사용자와 컴퓨터가 설정한 가위바위보 값과 승패를 리스트에서 확인할 수 있다. 이를 시각적으로도 승패를 알 수 있도록 컴퓨터와 사용자의 가위바위보 값을 이미지로 호출한다.

3. 컴퓨터를 한 개 더 추가하여 3인용 가위바위보 게임을 구현한다.

-> 기존 프로그램에서 연산을 더 추가하여 컴퓨터 2개와 사용자 1명의 가위바위보 게임을 구현한다.

II. 현재 완성된 부분

[소스코드]

```
import java.awt.*;
import java.awt.event.*;
import java.awt.image.BufferedImage;
import java.io.File;
import javax.imageio.ImageIO;
import javax.swing.*;

public class RockPaperSissorsSwing extends JFrame implements ActionListener {
	static String user = "";// 유저입력
	static String com = "";
	static int mixSu = 0; // 0~2 숫자를 랜덤으로 출력해서 저장하는 변수

	// 버튼 5개
	JButton rock; // 바위
	JButton paper; // 보
	JButton sissors; // 가위
	JButton end; // 종료
	JButton reset; // 리셋

	// 결과 보여줄 리스트
	static List list;

	// 패널 4개
	JPanel panel; // 버튼 놓을 패널
	JPanel panelA; // 컴퓨터 결과를 이미지로 나타낼 패널A
	JPanel panelB; // 사용자 결과를 이미지로 나타낼 패널B
	JPanel panelC; // vs이미지 나타낼 패널C

	// 이미지만 나타낼 레이블 변수
	JLabel comImg; // '컴퓨터' 이미지 나타낼 변수
	JLabel userImg; // '나'이미지 나타낼 변수
	JLabel vsImg; // 'vs' 이미지 나타낼 변수
	static JLabel comA; // 컴퓨터가 낼 가위바위보 이미지 나타낼 변수
	static JLabel userA; // 사용자가 낼 가위바위보 이미지 나타낼 변수

	public static void main(String args[]) {
		startMix();
		RockPaperSissorsSwing rps = new RockPaperSissorsSwing();
	}

	// 패널 배치
	public RockPaperSissorsSwing() {
		this.setTitle("가위바위보 게임");
```

```
this.setSize(1000, 1500);
this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
this.setVisible(true);

panel = new JPanel();
rock = new JButton("바위");
paper = new JButton("보");
sissors = new JButton("가위");
end = new JButton("종료");
reset = new JButton("reset");
list = new List();

// panel에 버튼 5개 놓기
panel.add(rock);
panel.add(paper);
panel.add(sissors);
panel.add(end);
panel.add(reset);

// 이미지 레이블 객체 생성
comImg = new JLabel(new ImageIcon("com.jpg"));
userImg = new JLabel(new ImageIcon("user.jpg"));
vsImg = new JLabel(new ImageIcon("vs.jpg"));
comA = new JLabel(new ImageIcon("rock.jpg")); //처음에는 바위로 시작
userA = new JLabel(new ImageIcon("rock.jpg"));  //처음에는 바위로
시작

// 각각의 패널에 레이블 배치
panelA = new JPanel();
panelA.setBackground(Color.WHITE);
panelA.setLayout(new GridLayout(2, 1, 0, 0));
panelA.add(comImg);
panelA.add(comA);
panelB = new JPanel();
panelB.setBackground(Color.WHITE);
panelB.setLayout(new GridLayout(2, 1, 0, 0));
panelB.add(userImg);
panelB.add(userA);
panelC = new JPanel();
panelC.setBackground(Color.WHITE);
panelC.add(vsImg);
panelC.setLayout(new GridLayout(1, 1));

// 패널과 리스트 borderlayout으로 추가
this.add(panelB, BorderLayout.EAST);
this.add(panelA, BorderLayout.WEST);
this.add(panelC, BorderLayout.CENTER);
this.add(panel, BorderLayout.SOUTH);
this.add(list, BorderLayout.NORTH);
this.setVisible(true);
pack();

rock.addActionListener(this);
paper.addActionListener(this);
```

```
        sissors.addActionListener(this);
        end.addActionListener(this);
        reset.addActionListener(this);
}

// 리스너 정의
public void actionPerformed(ActionEvent e) {
        if (e.getSource() == rock) {
                list.add("바위를 내셨군요.");
                getFight(com, "바위");
        } else if (e.getSource() == paper) {
                list.add("보를 내셨군요.");
                getFight(com, "보");
        } else if (e.getSource() == sissors) {
                list.add("가위를 내셨군요.");
                getFight(com, "가위");
        } else if (e.getSource() == reset) {
                list.removeAll();
                startMix();
        } else if (e.getSource() == end) {
                System.exit(0);
        }
}

// 컴퓨터가 내는 가위,바위,보 초기화
static void startMix() {
        mixSu = (int) (Math.random() * 3);
        switch (mixSu) {
        case 0:
                com = "가위";
                break;
        case 1:
                com = "바위";
                break;
        default:
                com = "보";
                break;
        }
}
// 가위바위보 처리 메서드
// 컴퓨터와 사용자의 가위바위보에 따라 이미지 아이콘 바꾸기
static void getFight(String com, String user) {
        if (com.equals("바위")) {
                comA.setIcon(new ImageIcon("rock.jpg"));
                if (user.equals("바위")) {
                        userA.setIcon(new ImageIcon("rock.jpg"));
                        list.add("컴퓨터는 바위를 냈습니다.");
                        list.add("비겼습니다.");
                } else if (user.equals("보")) {
```

```
                        userA.setIcon(new ImageIcon("paper.jpg"));
                        list.add("컴퓨터는 바위를 냈습니다.");
                        list.add("이겼습니다.");
                    } else {
                        userA.setIcon(new ImageIcon("sissors.jpg"));
                        list.add("컴퓨터는 바위를 냈습니다.");
                        list.add("졌습니다.");
                    }
                }

                if (com.equals("가위")) {
                    comA.setIcon(new ImageIcon("sissors.jpg"));
                    if (user.equals("바위")) {
                        userA.setIcon(new ImageIcon("rock.jpg"));
                        list.add("컴퓨터는 가위를 냈습니다.");
                        list.add("이겼습니다.");
                    } else if (user.equals("보")) {
                        userA.setIcon(new ImageIcon("paper.jpg"));
                        list.add("컴퓨터는 가위를 냈습니다.");
                        list.add("졌습니다.");
                    } else {
                        userA.setIcon(new ImageIcon("sissors.jpg"));
                        list.add("컴퓨터는 가위를 냈습니다.");
                        list.add("비겼습니다.");
                    }
                }
                if (com.equals("보")) {
                    comA.setIcon(new ImageIcon("paper.jpg"));
                    if (user.equals("바위")) {
                        userA.setIcon(new ImageIcon("rock.jpg"));
                        list.add("컴퓨터는 보를 냈습니다.");
                        list.add("졌습니다.");
                    } else if (user.equals("보")) {
                        userA.setIcon(new ImageIcon("paper.jpg"));
                        list.add("컴퓨터는 보를 냈습니다.");
                        list.add("비겼습니다.");
                    } else {
                        userA.setIcon(new ImageIcon("sissors.jpg"));
                        list.add("컴퓨터는 보를 냈습니다.");
                        list.add("이겼습니다.");
                    }
                }
                startMix(); // 가위바위보가 1회 끝나면 컴퓨터가 내는 가위바위보를
변경한다.
            }
}
```

[구현한 부분 분석]

기존 프로그램에서는 start_display()에서 패널을 사용하지 않고 컨테이너로 화면을 만들어 버튼과 리스트들을 절대 관리자로 구현하였다. 또한 RockPaperSissorsSwing()메서드에서 윈도우클로징을 사용하여 화면을 초기화 하고 start_display()를 호출하여 화면을 생성하였는데, 이렇게 구현하게 되면 이미지를 불러오기 어렵고 수정이 힘들기 때문에 다음과 같이 수정하였다.

- 레이아웃을 BorderLayout으로 바꾸기

Start_display() 메소드를 따로 구현하지 않고 그 내용들을 RockPaperSissorsSwing()에 넣어 주었다. 또한 RockPaperSissorsSwing()에 있던 window구문은 지우고 패널을 생성하여 버튼과 이미지들을 추가하고, 화면에 BorderLayout으로 배치하였다.

배치해야 하는 요소 – list, 버튼 5개 (가위, 바위, 보, end, reset), 이미지들 (comA, userA, comImg, userImg, vsImg)

list는 패널을 따로 사용하지 않고 NORTH에 배치한다. 버튼 5개는 panel에 추가하여 SOUTH에 배치한다. comImg('컴퓨터')와 comA(컴퓨터가 내는 가위바위보)는 panelA에 그리드 레이아웃으로 정렬하여 WEST에 배치한다. userImg('나')와 userA(사용자가 내는 가위바위보)는 panelB에 그리드 레이아웃으로 정렬하여 EAST에 배치한다. vsImg는 panelC에 추가하여 CENTER에 배치한다.

- 이미지 띄우기

이미지 - '컴퓨터', '나', 'VS', '가위', '바위', '보'

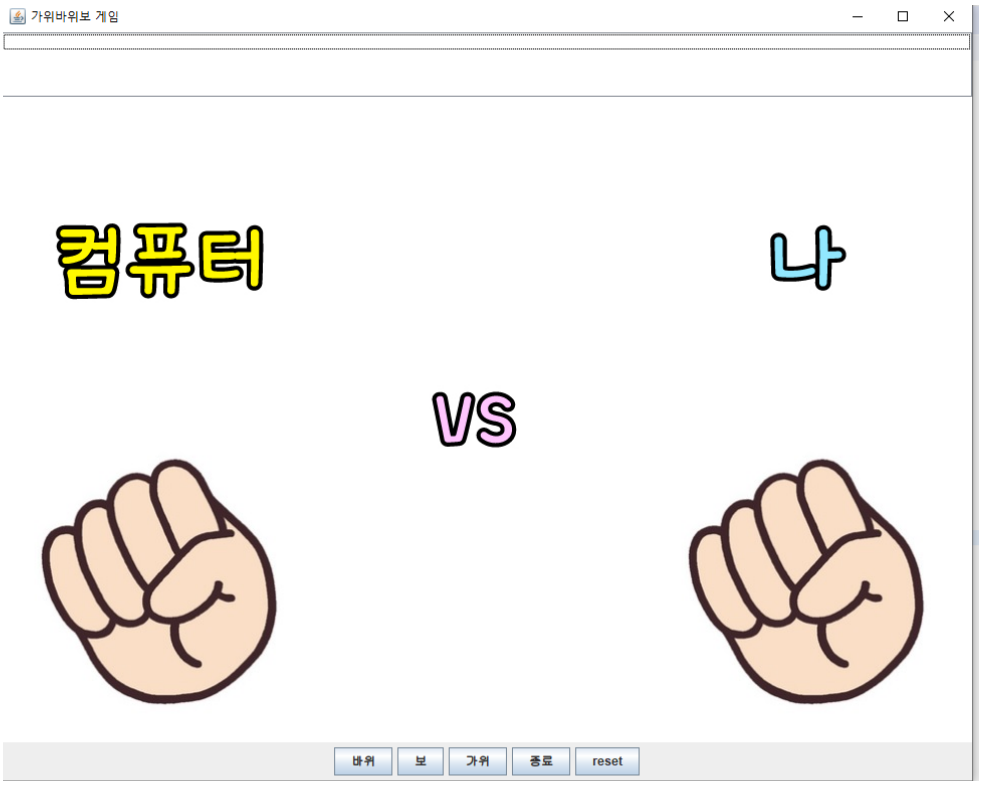
이미지를 띄우는 방식은 변형이 필요하지 않기 때문에 제일 간단하게 사용할 수 있는 ImageIcon 인스턴스를 생성하여 파일을 읽어오는 방식으로 구현하였다.

이미지 아이콘 객체를 생성한다. comImg는 '컴퓨터' 이미지, userImg는 '나' 이미지, vsImg는 'VS' 이미지, comA는 컴퓨터가 내는 가위바위보 이미지를 나타낼 변수, userA는 사용자가 낼 가위바위보 이미지를 나타낼 변수이다. comImg, userImg, vsImg, comA, userA 이미지들을 패널에 추가하여 화면에서 보이도록 한다. comA와 userA는 바위로 시작한다.

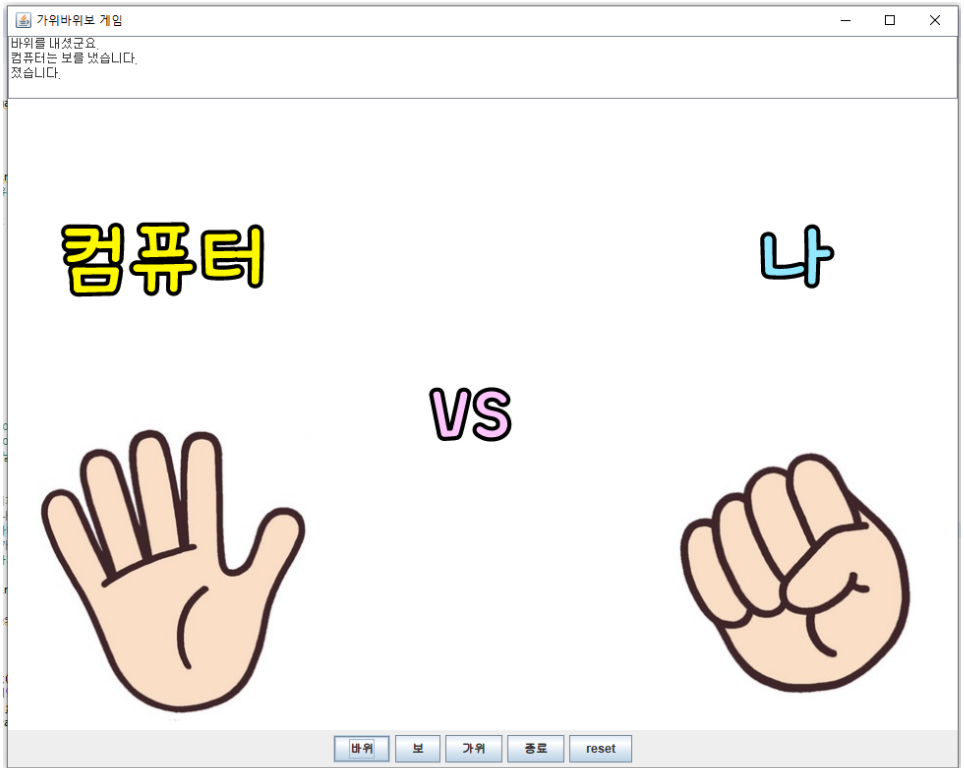
가위바위보 이미지는 버튼을 누를 때 마다 이미지가 바뀌어야 함으로 getFight()메소드에서 comA와 userA에 각각 메세징 하여 호출해 주어야 한다. if문 안에서 com과 user의 가위바위보가 바뀔 때 마다 아이콘 객체를 호출하여 버튼을 누를 때 그에 맞는 가위바위보 이미지가 화면에 보이도록 한다.

[실행화면]

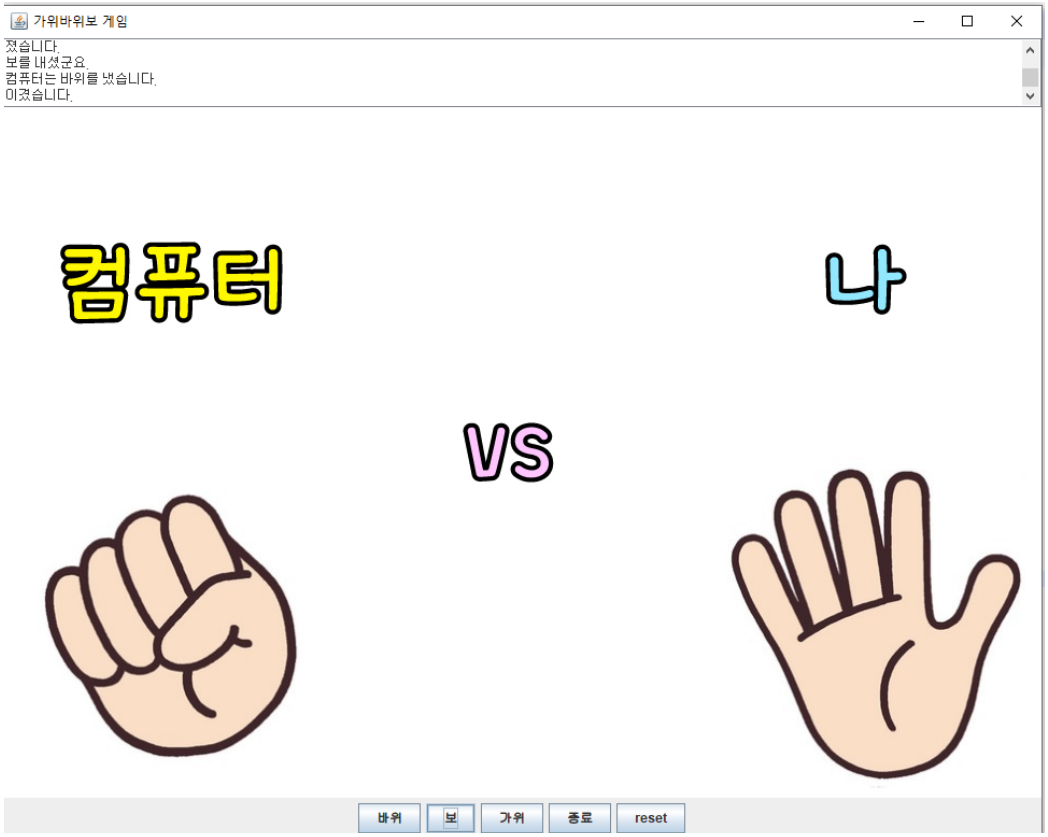
맨처음 실행한 화면



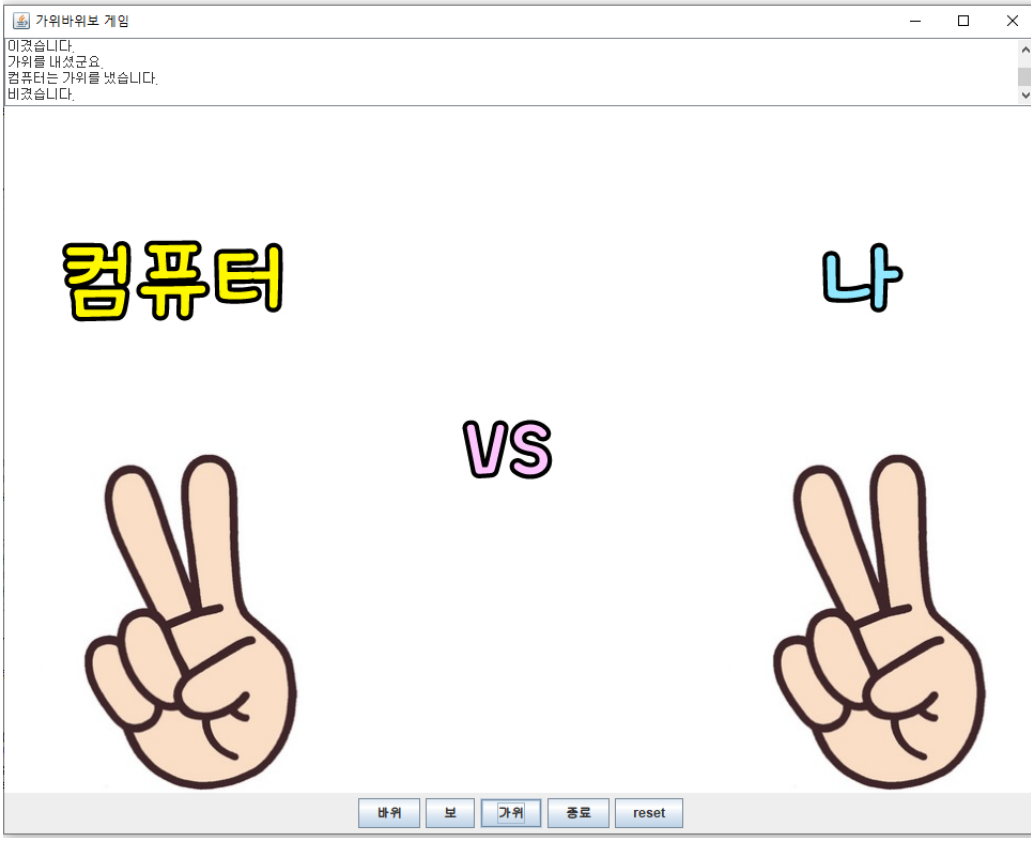
바위를 눌렀을 때



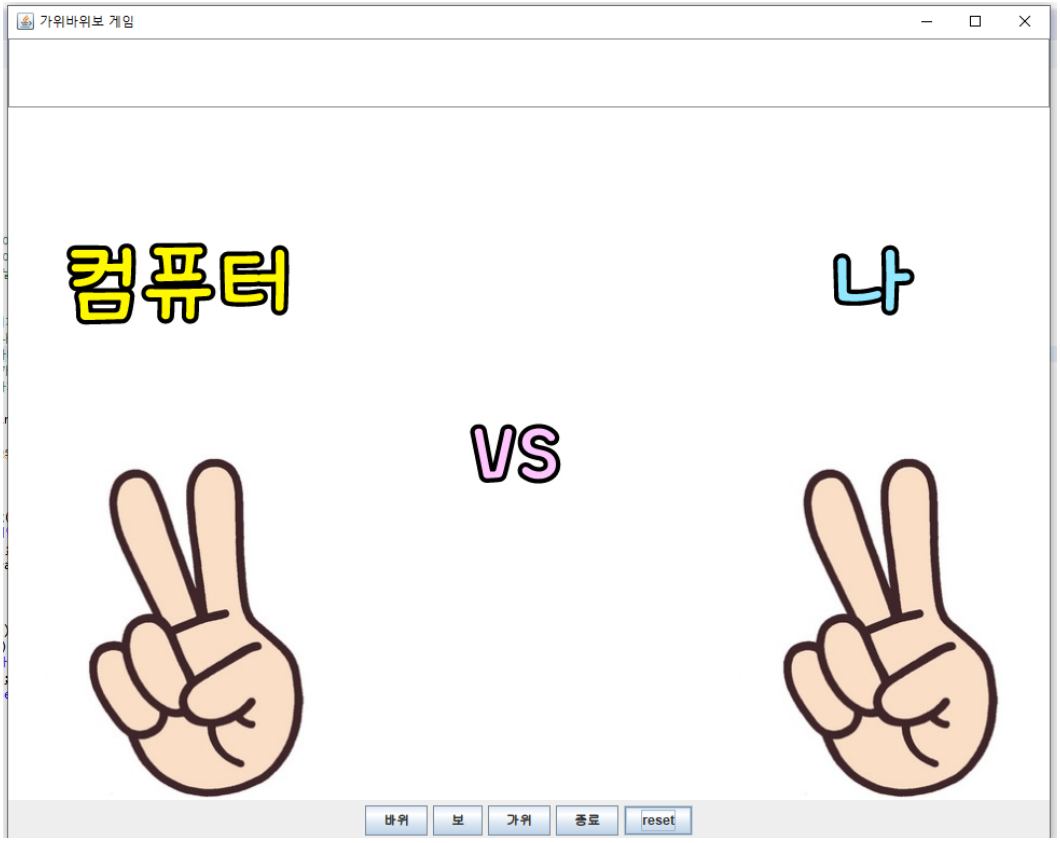
보를 눌렀을 때



가위를 눌렀을 때



Reset 눌렀을 때



III. 미완성 목록 리스트

- 컴퓨터를 한 개 더 추가하여 컴퓨터 2개와 사용자 1명의 3인용 가위바위보 게임 구현 부분

IV. 기획서 대비 진행률

40%