

# MINIGAME DELUXE

## 결과 보고서

자바 팀 프로젝트  
4조

16 김성수 15 정재훈  
14 정택규 16 한서현



# 목차

- 🌱 제목 및 개요
- 🌱 팀원 소개 및 역할 분담
- 🌱 전체 클래스 구조도
  - 🌱 클래스 설명
- 🌱 UI 기능 정의 내용
- 🌱 발표 후 추가 사항
- 🌱 프로젝트에 대한 개인 의견
- 🌱 GitHub

# 제목 및 개요

## MINIGAME DELUXE



내려오는 치킨들을 후라이드는  
왼쪽 방향키, 양념은 오른쪽  
방향키로 옮겨주세요



타이밍에 맞춰 스페이스 바를  
눌러서 점프를 이용해 장애물을  
넘어주세요

🌱 제목: MINIGAME DELUXE


🌱 간단한 키보드 입력만으로도 즐길 수  
있는 미니게임 두 개를 모아 놓은 게임


# 제목 및 개요

SCORE  
24

TIME  
22

## 치킨게임


 내려오는 치킨들을 후라이드는 왼쪽 방향키, 양념은 오른쪽 방향키로 옮겨주세요


 목표: 제한 시간 안에 최대한 많은 치킨들을 올바르게 옮기기

후라이드 < > 양념

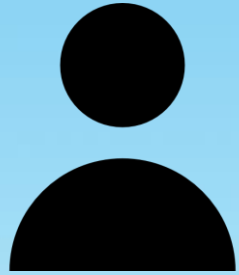
SCORE : 000318 M

## 런닝게임





 타이밍에 맞춰 스페이스 바를 눌러서 점프를 이용해 장애물을 넘어주세요

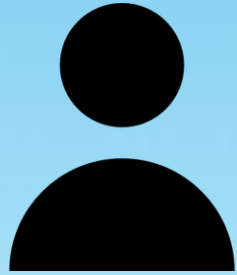
 목표: 장애물에 걸리지 않고 최대한 많은 거리를 가기

# 팀원 소개 및 역할 분담





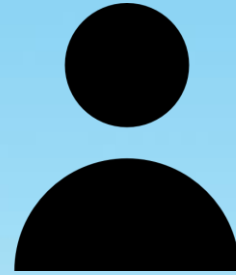
 김성수

-  치킨게임 프로그래밍
-  제안서, UI 정의서 초안 작성
-  사운드 프로그래밍
-  게임 리소스 조사






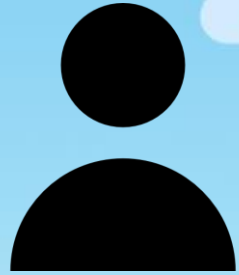
 정재훈


-  런닝게임, 메인 화면 씬 연결 프로그래밍
-  UML 작성







 정택규

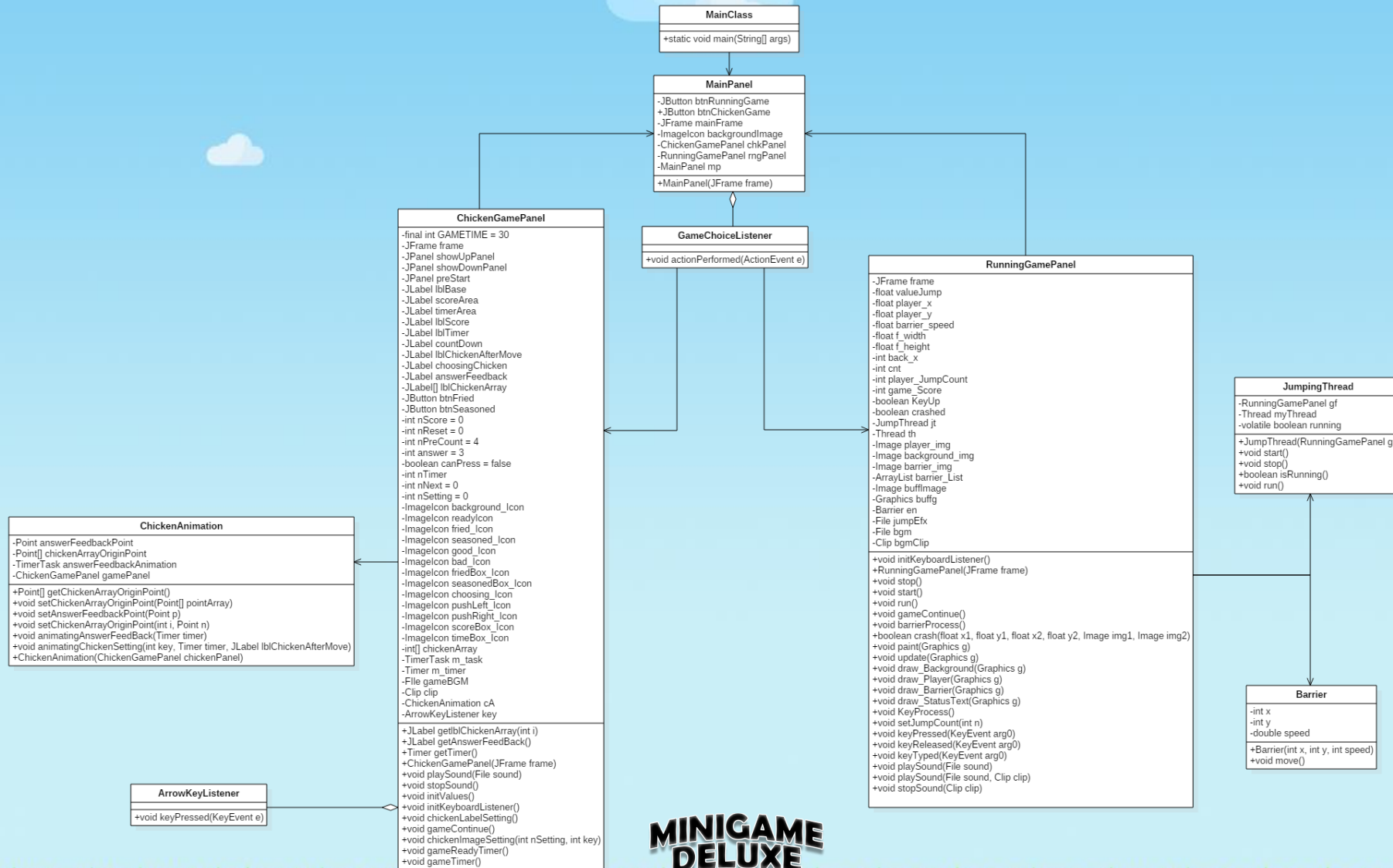
-  런닝게임 프로그래밍
-  제안서 초안 작성
-  게임 리소스 조사



 한서현

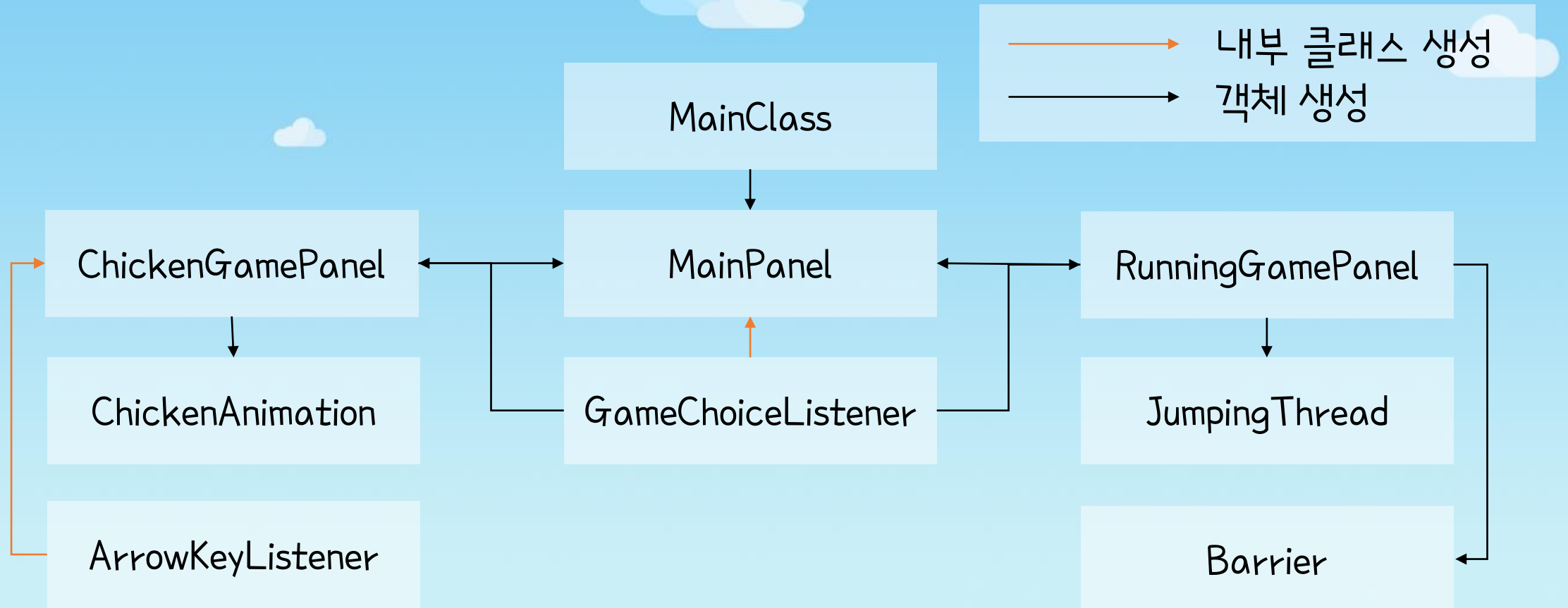
-  조장: 발표, PPT
-  치킨게임 프로그래밍
-  제안서, UI 정의서 작성
-  게임 리소스 제작

# 전체 클래스 구조도






# 전체 클래스 구조도



# 클래스 설명

## MainClass

MainClass
+static void main(String[] args)

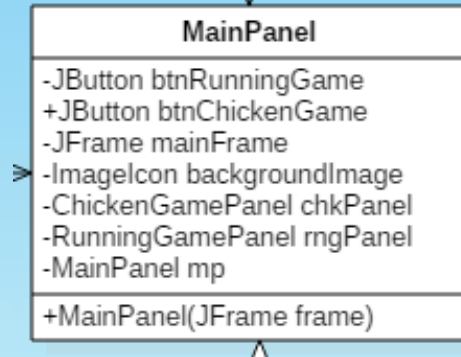
 main(String[] args) 프레임 생성과 메인 패널(MainPanel) 생성 및 프레임에 추가를 하는 메인 메소드



# 클래스 설명



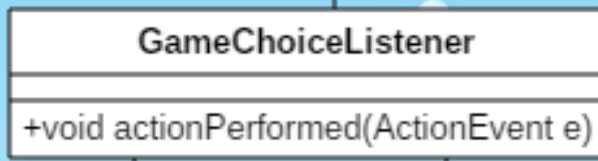
## MainPanel





MainPanel(JFrame frame) 메인 패널의 인스턴스 변수들을 초기화하는 생성자

# 클래스 설명

## GameChoiceListener



 메인 패널에서 버튼의 입력을 받기 위해 메인 패널 내부에서 선언된 클래스

 `actionPerformed(ActionEvent e)` 각 버튼에 따라 알맞은 동작을 하도록 `ActionListener` 인터페이스의 메소드를 재정의한 메소드

# 클래스 설명



## ChickenGamePanel

```
ChickenGamePanel
- final int GAMETIME = 30
- JFrame frame
- JPanel showUpPanel
- JPanel showDownPanel
- JPanel preStart
- JLabel lblBase
- JLabel scoreArea
- JLabel timerArea
- JLabel lblScore
- JLabel lblTimer
- JLabel countDown
- JLabel lblChickenAfterMove
- JLabel choosingChicken
- JLabel answerFeedback
- JLabel[] lblChickenArray
- JButton btnFried
- JButton btnSeasoned
- int nScore = 0
- int nReset = 0
- int nPreCount = 4
- int answer = 3
- boolean canPress = false
- int nTimer
- int nNext = 0
- int nSetting = 0
- ImageIcon background_Icon
- ImageIcon readyIcon
- ImageIcon fried_Icon
- ImageIcon seasoned_Icon
- ImageIcon good_Icon
- ImageIcon bad_Icon
- ImageIcon friedBox_Icon
- ImageIcon seasonedBox_Icon
- ImageIcon choosing_Icon
- ImageIcon pushLeft_Icon
- ImageIcon pushRight_Icon
- ImageIcon scoreBox_Icon
- ImageIcon timeBox_Icon
- int[] chickenArray
- TimerTask m_task
- Timer m_timer
- File gameBGM
- Clip clip
- ChickenAnimation cA
- ArrowKeyListener key
+ JLabel getLblChickenArray(int i)
+ JLabel getAnswerFeedBack()
+ Timer getTimer()
+ ChickenGamePanel(JFrame frame)
+ void playSound(File sound)
+ void stopSound()
+ void initValues()
+ void initKeyListener()
+ void chickenLabelSetting()
+ void gameContinue()
+ void chickenImageSetting(int nSetting, int key)
+ void gameReadyTimer()
+ void gameTimer()
```



get/set 메소드들



ChickenGamePanel() 인스턴스 변수를 초기화하는 생성자



playSound(File) BGM 재생 시켜주는 메소드



stopSound() BGM 재생을 멈추는 메소드



initValues() 게임 초기화 메소드



initKeyListener() 키보드 입력이 가능하도록 포커스를 패널에 맞추는 메소드



chickenLabelSetting() 치킨을 랜덤으로 설정해주는 메소드



gameContinue() 게임 종료 후 점수를 알려주고 다시 실행할지 물어보는 팝업창을 띄워주는 메소드



chickenImageSetting(int, int) 치킨 배열 값(int)를 받아서 게임 내에 이미지를 맞춰 표시해주는 메소드



gameReadyTimer() 게임 시작 전 3초 준비 타이머 메소드



gameTimer() 게임 30초 제한시간 타이머 메소드

# 클래스 설명



## ChickenAnimation

ChickenAnimation
-Point answerFeedbackPoint -Point[] chickenArrayOriginPoint -TimerTask answerFeedbackAnimation -ChickenGamePanel gamePanel
+Point[] getChickenArrayOriginPoint() +void setChickenArrayOriginPoint(Point[] pointArray) +void setAnswerFeedbackPoint(Point p) +void setChickenArrayOriginPoint(int i, Point n) +void animatingAnswerFeedBack(Timer timer) +void animatingChickenSetting(int key, Timer timer, JLabel lblChickenAfterMove) +ChickenAnimation(ChickenGamePanel chickenPanel)



ChickenAnimation(ChickenGamePanel chickenPanel) 인스턴스 변수를 초기화하는 생성자



get/set 메소드들



animatingAnswerFeedBack(Timer) GOOD, BAD 애니메이션 메소드




animatingChickenSetting(int, Timer, JLabel) 치킨들을 한 칸씩 당기고, 치킨을 옮기는 애니메이션 메소드

# 클래스 설명

 ArrowKeyListener

ArrowKeyListener
+void keyPressed(KeyEvent e)

 keyPressed(KeyEvent) 게임 진행을 위해 키 입력을 받는 메소드

# 클래스 설명



## RunningGamePanel

RunningGamePanel
<div><div><div>-JFrame frame</div><div>-float valueJump</div><div>-float player_x</div><div>-float player_y</div><div>-float barrier_speed</div><div>-float f_width</div><div>-float f_height</div><div>-int back_x</div><div>-int cnt</div><div>-int player_JumpCount</div><div>-int game_Score</div><div>-boolean KeyUp</div><div>-boolean crashed</div><div>-JumpThread jt</div><div>-Thread th</div><div>-Image player_img</div><div>-Image background_img</div><div>-Image barrier_img</div><div>-ArrayList barrier_List</div><div>-Image buffImage</div><div>-Graphics buffg</div><div>-Barrier en</div><div>-File jumpEfx</div><div>-File bgm</div><div>-Clip bgmClip</div></div><div><div>+void initKeyListener()</div><div>+RunningGamePanel(JFrame frame)</div><div>+void stop()</div><div>+void start()</div><div>+void run()</div><div>+void gameContinue()</div><div>+void barrierProcess()</div><div>+boolean crash(float x1, float y1, float x2, float y2, Image img1, Image img2)</div><div>+void paint(Graphics g)</div><div>+void update(Graphics g)</div><div>+void draw_Background(Graphics g)</div><div>+void draw_Player(Graphics g)</div><div>+void draw_Barrier(Graphics g)</div><div>+void draw_StatusText(Graphics g)</div><div>+void KeyProcess()</div><div>+void setJumpCount(int n)</div><div>+void keyPressed(KeyEvent arg0)</div><div>+void keyReleased(KeyEvent arg0)</div><div>+void keyTyped(KeyEvent arg0)</div><div>+void playSound(File sound)</div><div>+void playSound(File sound, Clip clip)</div><div>+void stopSound(Clip clip)</div></div></div>



RunningGamePanel(JFrame frame) 인스턴스 변수 초기화와 Thread 실행을 하는 생성자



initKeyListener() 키보드 입력이 가능하도록 포커스를 맞춰주는 메소드



stop() 게임 Thread를 중지하는 메소드



init() 변수 초기화 메소드



start() 게임 Thread를 실행하는 메소드



run() 게임 Thread가 작업할 것들을 구현한 메소드



gameContinue() 게임오버가 되면 게임을 다시 할 것인지, 메뉴로 되돌아갈 것인지를 묻는 창을 띄우는 메소드



barrierProcess() 게임에 등장하는 장애물들의 생성, 소멸, 위치 갱신 등을 담당하는 메소드



KeyProcess() 매 프레임이 갱신 될 때마다 불리며 인풋을 받는 메소드. 스페이스 바가 눌렸을 경우 플레이어 오브젝트의 점프 애니메이션을 실행하며, 2번 이상의 입력을 제한하는 역할도 한다.



crash() 플레이어와 장애물의 충돌여부를 반환하는 메소드



paint(Graphics g) 각 오브젝트들의 갱신된 새로운 위치에 맞게 그림을 그리도록 재정의하는 메소드



update(Graphics g) 더블 버퍼링을 위해 재정의하는 메소드



# 클래스 설명



## RunningGamePanel

```
RunningGamePanel
-JFrame frame
-float valueJump
-float player_x
-float player_y
-float barrier_speed
-float f_width
-float f_height
-int back_x
-int cnt
-int player_JumpCount
-int game_Score
-boolean KeyUp
-boolean crashed
-JumpThread jt
-Thread th
-Toolkit tk
-Image player_img
-Image background_img
-Image barrier_img
-ArrayList barrier_List
-Image buffImage
-Graphics buffg
-Barrier en
-File jumpEfx
-File bgm
-Clip bgmClip

+void initKeyboardListener()
+RunningGamePanel(JFrame frame)
+void stop()
+void start()
+void run()
+void gameContinue()
+void barrierProcess()
+boolean crash(float x1, float y1, float x2, float y2, Image img1, Image img2)
+void paint(Graphics g)
+void update(Graphics g)
+void draw_Background(Graphics g)
+void draw_Player(Graphics g)
+void draw_Barrier(Graphics g)
+void draw_StatusText(Graphics g)
+void KeyProcess()
+void setJumpCount(int n)
+void keyPressed(KeyEvent arg0)
+void keyReleased(KeyEvent arg0)
+void keyTyped(KeyEvent arg0)
+void playSound(File sound)
+void playSound(File sound, Clip clip)
+void stopSound(Clip clip)
```

draw\_Background(Graphics g) 매 프레임마다 갱신된 위치에 배경 이미지를 그리는 메소드

draw\_Player(Graphics g) 매 프레임마다 갱신된 위치에 플레이어 오브젝트를 그리는 메소드

draw\_Barrier(Graphics g) 매 프레임마다 갱신된 위치에 장애물들을 그리는 메소드

draw\_StatusText(Graphics g) 매 프레임마다 갱신된 점수 레이블을 그리는 메소드

setJumpCount(int n) 플레이어의 점프횟수를 파라미터로 받은 수 만큼 초기화하는 메소드

keyPressed(KeyEvent arg0) 스페이스바를 눌렀을 때 점프 횟수와 key가 눌렸는 지의 boolean변수를 갱신하는 keyListener 인터페이스 메소드

keyReleased(KeyEvent arg0), keyTyped(KeyEvent arg0) 재정의 안함

playSound(File sound) 파라미터로 받은 음악 파일을 재생하는 메소드

playSound(File sound, Clip clip) 파라미터로 받은 음악 파일을 재생하며, 배경 음악과 같이 중간에 멈추는 등의 동작이 필요한 파일의 경우 clip을 파라미터로 받아 이를 가 능하도록 오버로딩한 메소드

stopSound(Clip clip) 재생 중인 음악 파일을 정지하는 메소드




# 클래스 설명


## JummpingThread


JumpingThread
-RunningGamePanel gf -Thread myThread -volatile boolean running
+JumpThread(RunningGamePanel gf) +void start() +void stop() +boolean isRunning() +void run()

 JumpThread(RunningGamePanel gf) 게임 패널 객체를 인스턴스 변수에 저장하는 생성자

 start() JumpThread 실행 메소드

 stop() JumpThread의 작동을 멈추는 메소드

 isRunning() JumpThread가 동작 중인지의 여부를 반환하는 메소드

 run() JumpThread의 Thread 동작을 구현하여 플레이어의 프레임 별 위치를 갱신하는 메소드


# 클래스 설명


## Barrier

▼

### Barrier

-int x -int y -double speed
+Barrier(int x, int y, int speed) +void move()

 Barrier(int x, int y, int speed) 장애물의 위치, 속도를 파라미터로 받아 인스턴스 변수에 저장하는 생성자

 move() Barrier의 x좌표를 객체의 속도만큼 감소시켜 왼쪽으로 움직이는 메소드

# UI 기능 정의 내용

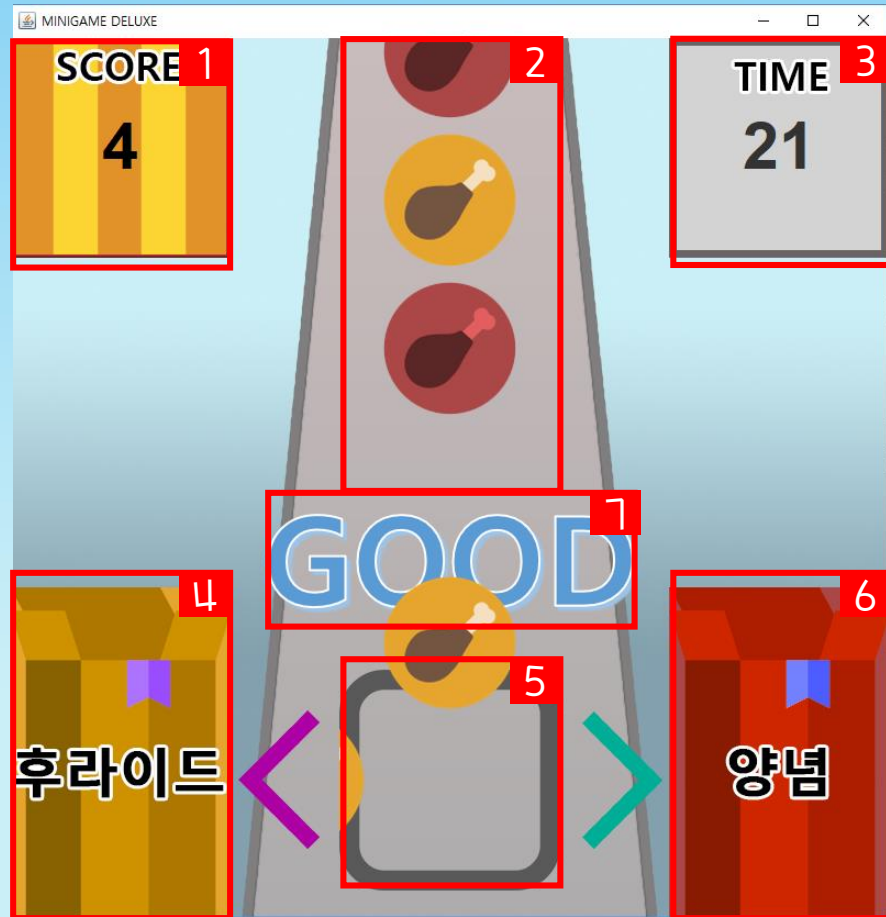
## 메인 화면



순번	이름	화면/ 기능 설명
1	게임 이름	게임 이름을 이미지로 표시합니다.
2	미니게임1	미니게임1이 어떤 게임인지 설명이 있는 버튼입니다. 해당 버튼을 마우스로 클릭하면 미니게임1 씬으로 이동합니다.
3	미니게임2	미니게임2이 어떤 게임인지 설명이 있는 버튼입니다. 해당 버튼을 마우스로 클릭하면 미니게임2 씬으로 이동합니다.

# UI 기능 정의 내용

## 치킨 게임



순번	이름	화면/기능 설명
1	점수(Score)	현재의 점수(int)를 실시간으로 표시합니다. 게임 시작 시 0점으로 표시됩니다. 현재의 치킨을 맞는 박스에 보내면 1점을 획득하고, 틀린 박스에 보내면 2점을 감점합니다. 틀린 박스에 보냈을 때는 1초 동안 점수의 글씨가 빨갱게 변합니다. 점수는 0점 미만으로 떨어지지 않습니다.
2	다음 순서의 치킨들	다음 순서의 치킨이 무엇인지 3개까지 표시합니다. 게임이 시작될 때 함께 표시되며, 랜덤으로 생성된 1000개의 변수를 가진 단일 배열로 이루어져 있습니다. 예시처럼 세로로 표시되어 있으며 아래쪽에 있는 치킨 일수록 '현재의 치킨'의 다음 순서입니다.
3	남은 시간 초	남은 시간(int)를 실시간으로 표시합니다. 30초부터 시작하여 1초씩 줄어 들고, 0초가 되면 게임이 종료됩니다.
4	후라이드 치킨 박스	후라이드 치킨들을 보내야 하는 곳입니다.
5	현재의 치킨	현재의 치킨이 무엇인지 나타냅니다. 이곳에 있는 치킨들을 좌우 방향키를 이용해서 양쪽의 박스로 보내야 합니다. 가장 최근에 누른 방향키에 따라 화살표의 색상이 바뀝니다.
6	양념 치킨 박스	양념 치킨들을 보내야 하는 곳입니다.
7	정답 피드백	올바르게 치킨을 옮겼으면 GOOD, 틀리게 치킨을 옮겼으면 BAD 이미지가 등장합니다.

# UI 기능 정의 내용

## 런닝 게임

SCORE : 001177 **1**

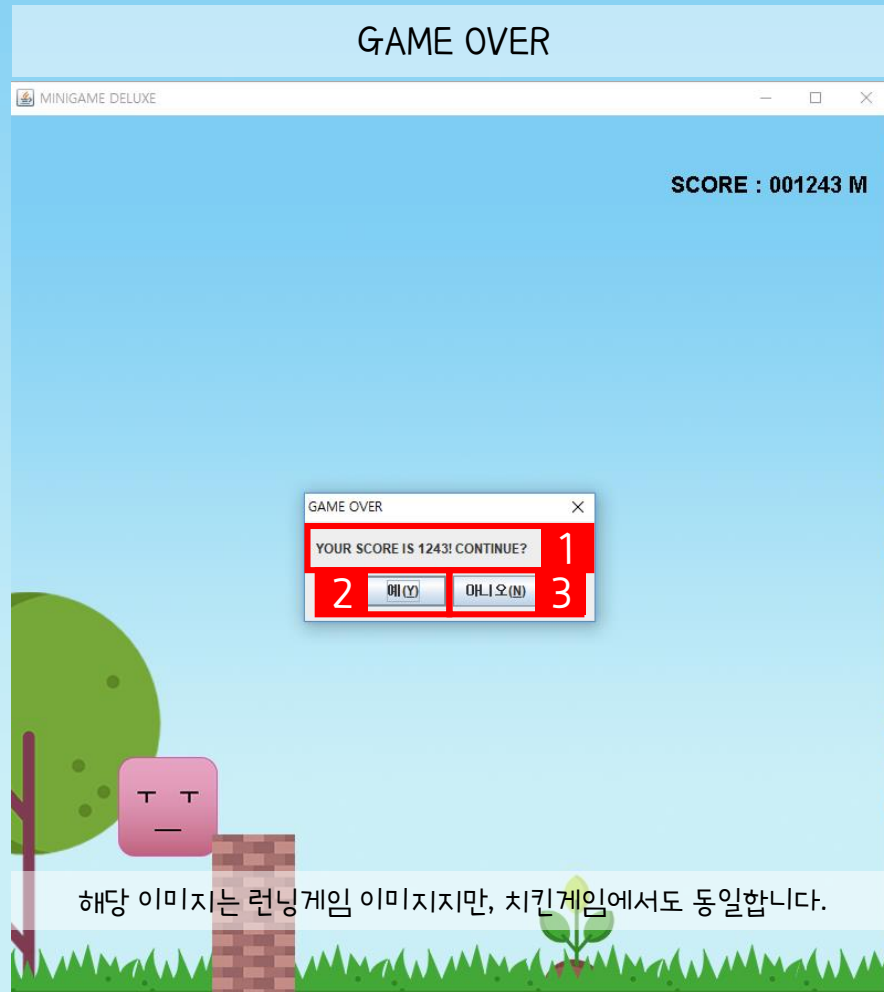
**3**

**2**

순번	이름	화면/기능 설명
1	이동한 거리	이동한 거리를 실시간으로 숫자(int)로 표시합니다. 이동한 거리는 0에서 시작하여 배경 이미지가 x축 방향으로 이동한 픽셀을 뜻합니다.
2	장애물	장애물은 1단 장애물, 2단 장애물, 공중 장애물이 있고 랜덤하게 등장합니다. 장애물이 오른쪽에서 왼쪽으로 일정한 속도로 이동합니다.
3	캐릭터	플레이어가 점프를 통해 조작할 수 있는 캐릭터입니다. 캐릭터의 x좌표는 고정시키고, 점프를 하면 y좌표가 변합니다.



# UI 기능 정의 내용



순번	이름	기능
1	플레이어의 점수	게임이 종료되었을 때 게임 화면을 멈추고 GAME OVER JOptionPane에서 플레이어의 점수를 표시합니다. "YOUR SCORE IS 0000!! CONTINUE?" 라고 텍스트로 표시합니다. 0000에는 플레이어가 획득한 점수를 표시합니다.
2	해당 미니게임 다시 플레이하기	예 버튼을 누르면 해당 미니게임을 다시 플레이할 수 있습니다.
3	메인 메뉴로 돌아가기	아니오 버튼을 누르면 메인 게임 씬으로 이동하여 미니게임의 종류를 선택할 수 있습니다.

# 발표 후 추가 사항

🌱 추가한 기능은 없습니다

🌱 블랙보드에 음악파일도 포함한 파일을 올렸더니 용량이 너무 커서 제대로 업로드가 되지 않았습니다. 그래서 블랙보드에 업로드한 파일은 배경음악 파일을 뺀 파일입니다. 배경음악이 포함된 파일은 메일로 보냈습니다. 죄송합니다.

🌱 GitHub에 업로드 했습니다



# GitHub

<https://github.com/HanSeoHyun/MINIGAME-DELUXE>



감사합니다

