

# 자바 프로그래밍

2021663037 백지호

# 프레젠테이션 개요

## 목차

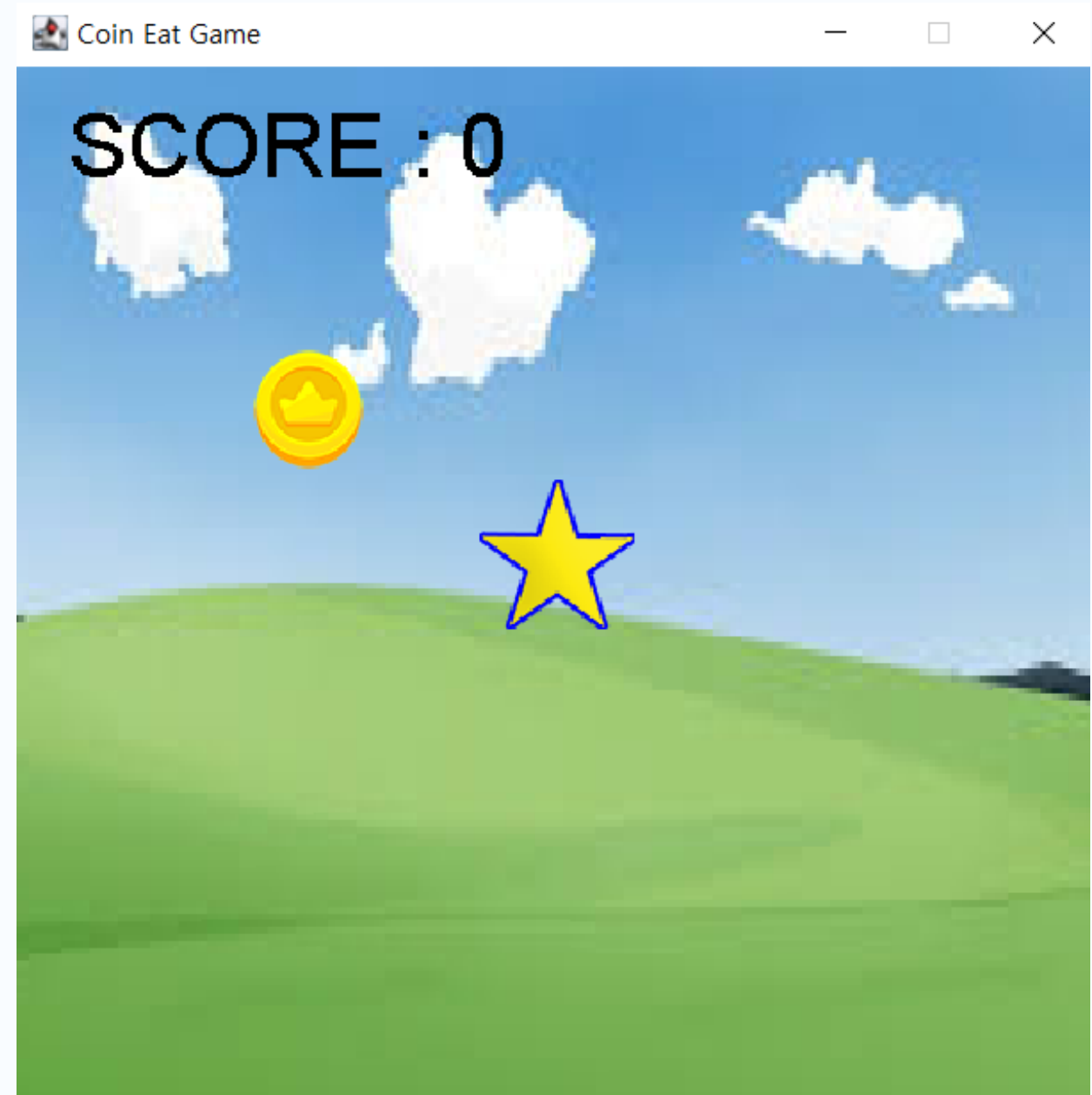
1. 게임 프로젝트 소개
2. 프로그램 실행 코드
3. 중요 소스 코드 분석

# 게임 프로젝트 소개

## COIN EAT GAME (동전 먹기 게임)

별이 자유롭게 움직이면서 코인에 가까워져 닿으면 점수가 100점씩 올라가는 게임입니다.

코인은 한번 닿으면 점수가 올라가며 다른 위치로 이동하게 됩니다.



```

1 package coinEat;
2
3 import java.awt.Color;
4 import java.awt.Font;
5 import java.awt.Graphics;
6 import java.awt.Image;
7 import java.awt.event.KeyAdapter;
8 import java.awt.event.KeyEvent;
9
10 import javax.swing.ImageIcon;
11 import javax.swing.JFrame;
12
13 public class CoinEat extends JFrame {
14     Image backgroundImage;
15     Graphics screenGraphic;
16
17     private Image background = new ImageIcon("src/mainScreen.jf
18     private Image player = new ImageIcon("src/player.png").getI
19     private Image coin = new ImageIcon("src/coin.png").getImage
20
21     int playerX, playerY;
22     int playerWidth = player.getWidth(null);
23     int playerHeight = player.getHeight(null);
24     int coinX, coinY;
25     int coinWidth = coin.getWidth(null);
26     int coinHeight = coin.getHeight(null);
27
28     int score;

```

COINEAT

```

1 package coinEat;
2
3 public class CoinEatDriver {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         CoinEat coinEat = new CoinEat();
8
9
10    }
11
12 }
13

```

COINEATDRIVER

# 중요 소스 코드 분석

## KEYPROCESS, CRASHCHECK

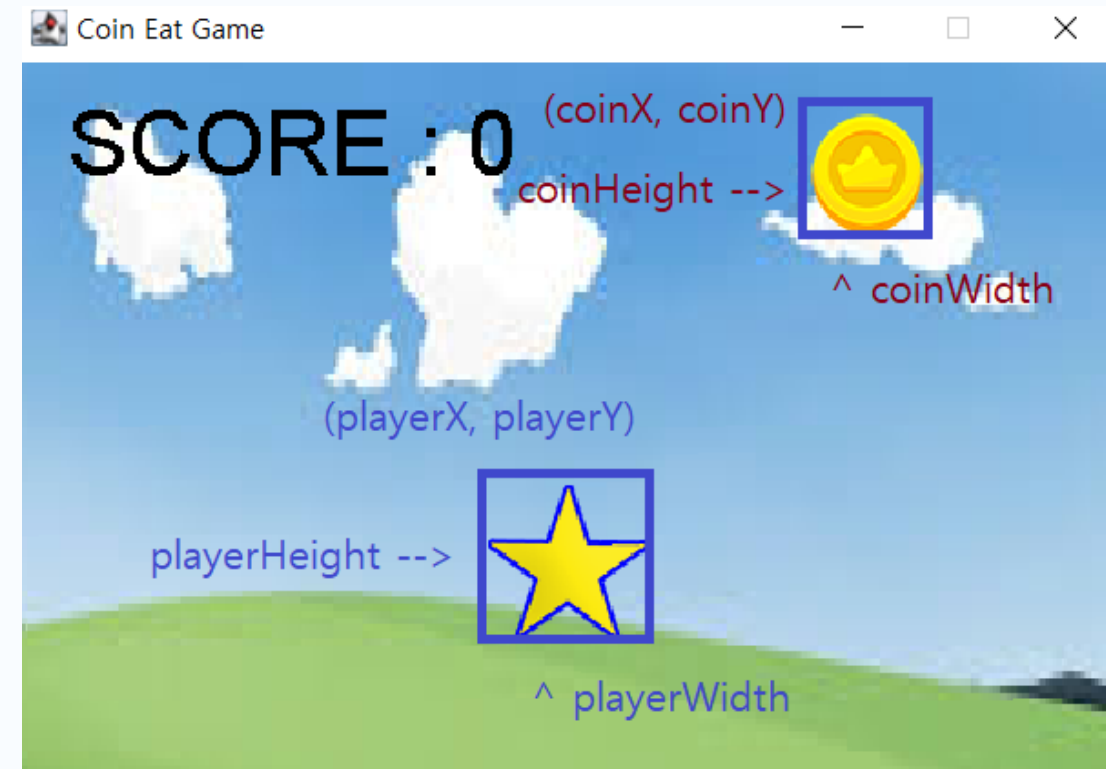
키를 통해 불리언 값으로 어떻게 이동시킬지 결정하는 코드이다.  
이때 플레이어의 가로, 세로의 길이, 이동거리 등을 고려한다.  
플레이어와 코인 충돌 체크하는 코드 이다.

코인과 플레이어가 닿았을때 점수를 올려주고 코인 위치를 다시 이동시킨다

playerX, playerY = 플레이어가 그려지기 시작하는 x,y좌표

coinX, coinY = 코인이 그려지기 시작하는 x,y 좌표입니다.

```
public void keyProcess() {  
    if (up && playerY - 3 > 30) playerY-=3;  
    if (down && playerY + playerHeight + 3 < 500) playerY+=3;  
    if (left && playerX - 3 > 0) playerX-=3;  
    if (right && playerX + playerWidth + 3 < 500) playerX+=3;  
}
```



```
public void crashCheck() {  
    if (playerX + playerWidth > coinX && coinX + coinWidth > playerX && playerY + playerHeight > coinY && coinY + coinHeight > playerY)  
        score+=100;  
    coinX = (int)(Math.random()*(501-playerWidth));  
    coinY = (int)(Math.random()*(501-playerHeight-30))+30;  
}
```


# 더블 버퍼링

더블 버퍼링(Double Buffering)은 이중 버퍼링이라 불리기도 하며, 그래픽 객체에 이미지를 그릴때 사용되는 기법이다.

이미지를 구현할때 움직임과 동시에 깜빡거리는 현상이 나타났을때 더블 버퍼링을 이용하여 이미지 전환이 매끄럽게 이어질 수 있도록 하는 것이다.

```
public void screenDraw(Graphics g) {  
    g.drawImage(background, 0, 0, null);  
    g.drawImage(coin, coinX, coinY, null);  
    g.drawImage(player, playerX, playerY, null);  
    g.setColor(Color.BLACK);  
    g.setFont(new Font("Arial", Font.PLAIN, 40));  
    g.drawString("SCORE : " + score, 30, 80);  
    this.repaint();  
}
```

```
public void paint(Graphics g) {  
  
    bufferImage = createImage(500, 500);  
    screenGraphic = bufferImage.getGraphics();  
    screenDraw(screenGraphic);  
    g.drawImage(bufferImage, 0, 0, null);  
}
```



**감사합니다**