Shri Govindram Seksaria Institute of technology Indore (M.P.)

Shivkumar Rawat

November 7, 2023

Programming Practices : Mini Project(Snake Game)

https://github.com/Shivkumar Rawat/pp/

Objective

The objective of creating a mini project like a Snake game in Java with the aim of providing an entertaining and interactive gaming experience. Develop the core game logic and mechanics of the classic Snake game, allowing the player to control a snake that grows by eating food items while avoiding collisions with the boundaries and itself.

Language Used

Java language(OOPS)

Purpose

1.Developing a Snake game in Java can help individuals, especially beginners, learn and practice Java programming concepts. It provides a hands-on way to understand Java's syntax, data structures, and object-oriented principles 2.Creating a simple game like Snake allows you to gain

experience in game development. You'll learn about game loops, rendering graphics, handling user input, and implementing game mechanics. 3.

Game development often involves solving various challenges, such as collision detection, game logic, and user interaction. This helps improve your problem-solving and critical thinking skills

---

Scope

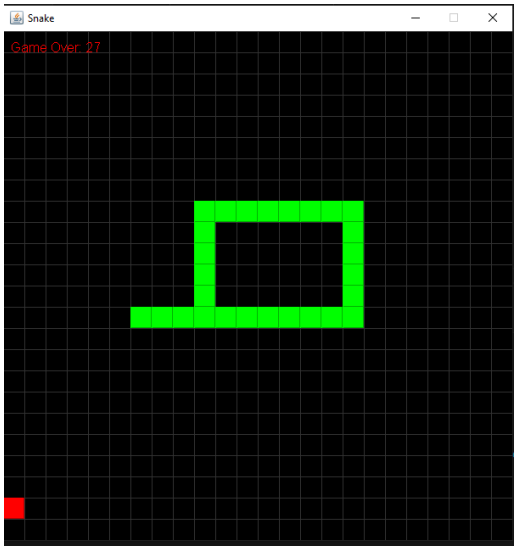
1. The scope of a mini project like a Snake game in Java can vary depending on your objectives and the complexity of the game you want to develop. Here are some aspects of the project's scope that you might consider.
2. Define the gameplay features you want to include in your Snake game. This might involve deciding on the rules of the game, such as how the snake moves, how it grows, how the game ends, and any power-ups or obstacles.
3. Determine the level of graphics and design you want to implement. This could range from simple geometric shapes to more elaborate graphics and animations. Consider how the game elements will be displayed on the screen.
4. Decide on the user interface components, such as a menu system, score display, and game over screen. Implementing user-friendly UI elements can enhance the player's experience.
5. If you want to add a competitive element, consider implementing a scoring system and possibly a leaderboard to encourage players to achieve higher scores.
6. Decide whether you want to include sound effects and background music in the game. This can add to the immersion and enjoyment of the game.

   can introduce different levels with varying degrees of difficulty to keep the game engaging and challenging for players as they progress.

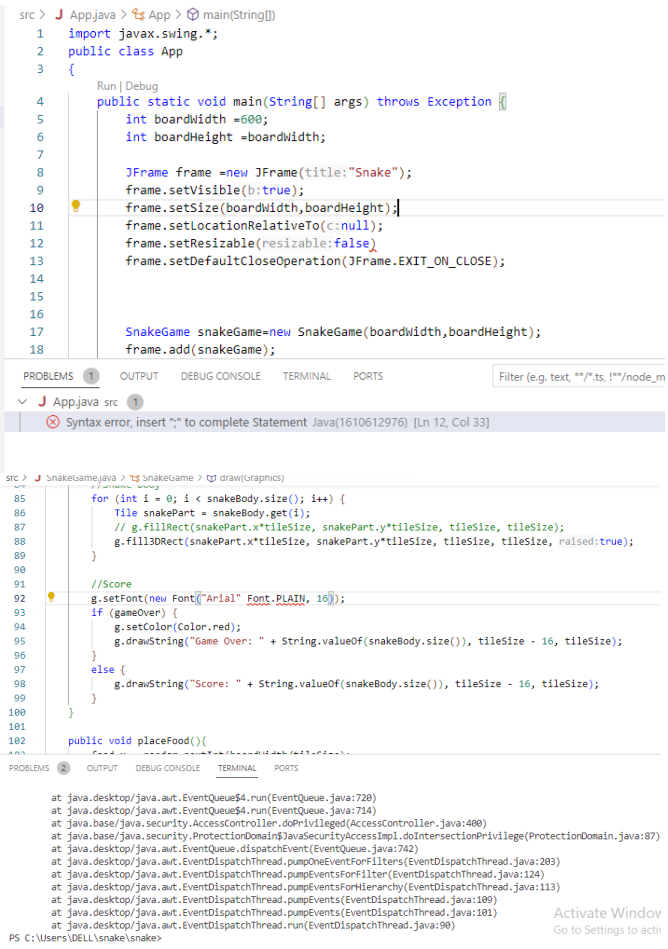
7. Define what happens when the game ends. Players may want the option to restart the game, view their final score, or return to the main menu.
8. Consider adding settings that allow players to customize aspects of the game, such as the snake's appearance or game speed.
9. If you want to expand the scope, you could add multiplayer functionality, enabling players to compete with or cooperate with others online.
10. Allocate time for thorough testing and debugging to ensure the game functions as expected and is free from critical bugs.
11. Create documentation, including a user manual or README file, to explain how to play the game, the rules, and any customization options.

    your project's source code, including comments and explanations, to make it more understandable and maintainable.
12. Consider whether you want to publish your game for others to play. This might involve creating executable files, distributing the game online, or sharing it with friends and peers.

Output



Debugging


src > App.java > App > main(String[])
1 import javax.swing.*;
2 public class App
3 {
Run | Debug
4 public static void main(String[] args) throws Exception {
5 int boardWidth =600;
6 int boardHeight =boardWidth;
7
8 JFrame frame =new JFrame(title:"Snake");
9 frame.setVisible(b:true);
10 frame.setSize(boardWidth,boardHeight);
11 frame.setLocationRelativeTo(c:null);
12 frame.setResizable(resizable:false);
13 frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
14
15
16
17 SnakeGame snakeGame=new SnakeGame(boardWidth,boardHeight);
18 frame.add(snakeGame);
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Filter (e.g. text, **/*.ts, !**/node_m
App.java src 1
Syntax error, insert ";" to complete Statement Java(1610612976) [Ln 12, Col 33]
src > SnakeGame.java > SnakeGame > draw(Graphics)
85 for (int i = 0; i < snakeBody.size(); i++) {
86 Tile snakePart = snakeBody.get(i);
87 // g.fillRect(snakePart.x*tileSize, snakePart.y*tileSize, tileSize, tileSize);
88 g.fill3DRect(snakePart.x*tileSize, snakePart.y*tileSize, tileSize, tileSize, raised:true);
89 }
90
91 //Score
92 g.setFont(new Font("Arial" Font.PLAIN, 16));
93 if (gameOver) {
94 g.setColor(Color.red);
95 g.drawString("Game Over: " + String.valueOf(snakeBody.size()), tileSize - 16, tileSize);
96 }
97 else {
98 g.drawString("Score: " + String.valueOf(snakeBody.size()), tileSize - 16, tileSize);
99 }
100 }
101
102 public void placeFood(){
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS
at java.desktop/java.awt.EventQueue$4.run(EventQueue.java:720)
at java.desktop/java.awt.EventQueue$4.run(EventQueue.java:714)
at java.base/java.security.AccessController.doPrivileged(AccessController.java:400)
at java.base/java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(ProtectionDomain.java:87)
at java.desktop/java.awt.EventQueue.dispatchEvent(EventQueue.java:742)
at java.desktop/java.awt.EventDispatchThread.pumpOneEventForFilters(EventDispatchThread.java:203)
at java.desktop/java.awt.EventDispatchThread.pumpEventsForFilter(EventDispatchThread.java:124)
at java.desktop/java.awt.EventDispatchThread.pumpEventsForHierarchy(EventDispatchThread.java:113)
at java.desktop/java.awt.EventDispatchThread.pumpEvents(EventDispatchThread.java:109)
at java.desktop/java.awt.EventDispatchThread.pumpEvents(EventDispatchThread.java:101)
at java.desktop/java.awt.EventDispatchThread.run(EventDispatchThread.java:90)
Activate Windov
Go to Settings to acti
PS C:\Users\DELL\snake\snake>