

The Very Hungry Caterpillar Game Requirements

Authors: Marlon Hernandez, Adam Shehata, Kevyn Girao, Alexis Salgado

Functional Specification

The Very Hungry Caterpillar Game is a simple arcade style game. The player controls a caterpillar that moves continuously in one direction until another direction is chosen. Movement is controlled with the keyboard arrow keys: UP, DOWN, LEFT, and RIGHT.

Apples appear randomly on the game board at the start of the game and each time one is eaten. When the caterpillar eats an apple, its length increases and the score is incremented. The objective of the game is to eat as many apples as possible without colliding into the walls of the screen or into the caterpillar's own body.

If a collision occurs, the game ends. A "Game Over" message is displayed along with the final score showing how many apples were eaten.

Use Cases

Actors

- Player: Controls the caterpillar using the keyboard.
- System (Game Engine): Handles movement, collision detection, scoring, and rendering.

Use Case 1: Start Game

Precondition: The game program is launched.

Main Flow:

1. Player runs the `CatepillarGame` main program.
 2. System opens the game window (`GameFrame`) and initializes the game panel.
 3. System starts the game loop, displays the caterpillar, and generates the first apple.
- Postcondition: The game is running with the caterpillar and an apple visible on the board.

Use Case 2: Control Caterpillar

Precondition: The game is running.

Main Flow:

1. Player presses arrow keys.

2. System updates the caterpillar's direction using `MyKeyAdapter`.
 3. On each timer tick, the caterpillar moves in the chosen direction.
- Postcondition: The caterpillar moves continuously in the selected direction.

Use Case 3: Eat Apple

Precondition: The game is running and an apple exists on screen.

Main Flow:

1. Caterpillar's head reaches the apple's coordinates.
 2. System increments the caterpillar's body length and score.
 3. System generates a new apple at a random location.
- Postcondition: The caterpillar grows longer, the score increases, and a new apple is displayed.

Use Case 4: Collision Detection

Precondition: The game is running.

Main Flow:

1. System checks if the caterpillar collides with itself.
 2. System checks if the caterpillar collides with the walls.
- Postcondition: If a collision is detected, the game ends and `running = false`.

Use Case 5: Game Over

Precondition: A collision has occurred.

Main Flow:

1. System stops the timer.
 2. System displays "Game Over" and the final score.
- Postcondition: The player can no longer control the caterpillar.

Glossary

Caterpillar: The player controlled character that grows in length by eating apples.

Apple: An item that appears randomly on the board. Eating it increases the caterpillar's length and score.

Game Board: The rectangular playing area where the caterpillar moves.

Collision: When the caterpillar hits a wall or its own body, causing the game to end.

Score: The number of apples the player has eaten.

Game Loop: The continuous cycle of updating the game state and rendering graphics.

SCREEN_WIDTH / SCREEN_HEIGHT: Dimensions of the game window.

UNIT_SIZE: The size of one segment of the caterpillar.

DELAY: The speed of the caterpillar's movement.

bodyParts: Initial size of the caterpillar. **applesEaten:**

Counter for apples collected. **appleX / appleY:** Coordinates

of the apple on the board. **direction:** The current movement

direction of the caterpillar. **running:** Boolean value

indicating if the game is active.

GamePanel(): Sets up the background of the game. **startGame():**

Initializes the game and begins play. **draw(Graphics):** Renders the

caterpillar, apples, and text on the screen. **newApple():** Generates a

new apple at a random location. **move():** Updates the caterpillar's

position.

checkApple(): Detects apple collisions, grows the caterpillar, updates score, and spawns a new

apple. **checkCollisions():** Ends the game if the caterpillar collides with itself or a wall.

gameOver(): Displays "Game Over" and the final score.

MyKeyAdapter: Connects the keyboard arrow keys to caterpillar movement.

resetButton(): When game ends, a reset button allows for the game to be replayed.