Universidad Autónoma de Baja California

Facultad de ingeniería, Arquitectura y diseño

Ingeniería en Software y Tecnologías Emergentes

Paradigmas de la Programación

Docente: Gallegos Mariscal José Carlos

Alumno: Diarte Salas Gilberto

Matricula: 360954

Grupo: 941

Práctica: Snake con Memoria dinámica

Fecha de Entrega: 31/mayo/2024

Introducción

¡Bienvenido al juego de Snake con Raylib!, en este juego, te sumergirás en el clásico desafío de controlar una serpiente y hacerla crecer mientras evitas chocar con las paredes y tu propia cola.

Objetivo del Juego

El objetivo del juego es simple: controlar la serpiente para que coma la mayor cantidad de comida posible sin chocar con las paredes o su propia cola. Cada vez que la serpiente come comida, su longitud aumenta y tu puntuación también.

Características del Juego

Gráficos y Sonido

El juego está diseñado con gráficos simples pero efectivos, utilizando la librería Raylib para proporcionar una experiencia visual atractiva. Además, cuenta con efectos de sonido para mejorar la inmersión del jugador.

Control de la Serpiente

Controlarás la serpiente utilizando las teclas de dirección (arriba, abajo, izquierda, derecha) para moverla por el tablero. La serpiente se desplaza continuamente en la dirección en la que se mueve, y tu objetivo es evitar chocar con las paredes y tu propia cola. El juego se reinicia al presionar la tecla “r” al morir.

Comida y Puntuación

En el tablero aparecerá comida en ubicaciones aleatorias. Tu objetivo es dirigir a la serpiente hacia la comida para que pueda comerla y crecer.

Dificultad Progresiva

Conforme avances en el juego, la velocidad de la serpiente aumentará gradualmente, para ello tenemos una pantalla suficientemente grande para liberta de movimiento, lo que aumentará la dificultad y el desafío. ¡Prepárate para mantener tus reflejos agudos y tu concentración en su punto máximo!

Tecnologías Utilizadas

Lenguaje de Programación: C

El juego está programado en lenguaje C, aprovechando su eficiencia y potencia para crear una experiencia de juego fluida y receptiva.

Librería Gráfica: Raylib

Raylib es una librería gráfica sencilla y fácil de usar que proporciona todas las herramientas necesarias para crear gráficos en 2D de alta calidad y juegos interactivos en C.

Memoria Dinámica y Listas Enlazadas

Para almacenar la cola de la serpiente y manejar su crecimiento dinámico, utilizamos memoria dinámica y listas enlazadas. Esto nos permite mantener un control preciso sobre la longitud de la serpiente y su posición en el tablero.

Sin Variables Globales

Se evitarán las variables globales para promover una mejor organización del código y una estructura más modular y mantenible.

Desarrollo del Juego

El juego consta de varios elementos clave, incluyendo la serpiente (Snake), la fruta (Fruit), y la interacción del usuario para controlar el movimiento de la serpiente. A continuación, se presentan los fragmentos de código más importantes que definen estas funcionalidades:

Definición de la Serpiente (Snake)

La estructura Snake representa la serpiente en el juego. Está compuesta por una lista enlazada de nodos (Node), donde cada nodo representa un segmento del cuerpo de la serpiente. La dirección de movimiento de la serpiente se define mediante el enum Direction.

```
typedef enum
{
    UP,
    DOWN,
    LEFT,
    RIGHT
} Direction;

typedef struct Node
{
    int x;
    int y;
    struct Node *next;
} Node;

typedef struct
{
    Node *head;
    int length;
    Direction direction;
} Snake;
```

Inicialización del Juego

La función init_game se encarga de inicializar el estado del juego, incluyendo la posición inicial de la serpiente y la fruta, así como la velocidad inicial de la serpiente.

```
78  void init_game(Snake *snake, Fruit *fruit, int *snake_speed)
79  {
80      snake->head = create_node(SCREEN_WIDTH / 2, SCREEN_HEIGHT / 2);
81      snake->length = 1;
82      snake->direction = RIGHT;
83
84      fruit->x = (rand() % (SCREEN_WIDTH / GRID_SIZE)) * GRID_SIZE;
85      fruit->y = (rand() % (SCREEN_HEIGHT / GRID_SIZE)) * GRID_SIZE;
86      *snake_speed = 5;
87  }
```

Actualización del Juego

La función update_game se encarga de actualizar el estado del juego en cada fotograma. Aquí se lleva a cabo la lógica de movimiento de la serpiente, la detección de colisiones con la fruta y los límites del tablero, así como la comprobación de colisiones consigo misma.

```
void update_game(Snake *snake, Fruit *fruit, int *game_over, Sound eat_sound, Sound die_sound, int *snake_speed)
{
    static int frame_counter = 0;

    if (frame_counter >= 60 / *snake_speed)
    {
        Node *new_head = create_node(snake->head->x, snake->head->y);
        if (snake->direction == UP)
            new_head->y -= GRID_SIZE;
        if (snake->direction == DOWN)
            new_head->y += GRID_SIZE;
        if (snake->direction == LEFT)
            new_head->x -= GRID_SIZE;
        if (snake->direction == RIGHT)
            new_head->x += GRID_SIZE;
```

(parte del código, la función tiene más líneas)

Dibujo del Juego

La función draw_game se encarga de dibujar los elementos del juego en la pantalla, incluyendo la serpiente, la fruta y el fondo del tablero.

```
void draw_game(Snake *snake, Fruit *fruit, Texture2D apple, Texture2D texture)
{
    DrawTexture(texture, 0, 0, RAYWHITE);
    bool is_red = true;

    Node *current = snake->head;
    while (current != NULL)
    {
        Color color = (is_red) ? RED : BLACK;
        DrawCircle(current->x + GRID_SIZE / 2, current->y + GRID_SIZE / 2, GRID_SIZE / 2, color);
        is_red = !is_red;
        current = current->next;
    }

    DrawTexturePro(apple, (Rectangle){0, 0, apple.width, apple.height}, (Rectangle){fruit->x, fruit->y, 20, 20}, (Vector2){0, 0}, 0.0f, RAYWHITE);
}
```

Conclusiones

En resumen, el desarrollo de un juego de Snake en C utilizando la biblioteca Raylib es una tarea emocionante que combina conceptos de programación, diseño de videojuegos y creatividad. A través de los fragmentos de código presentados, hemos explorado las funcionalidades principales del juego y hemos destacado la importancia de cada una en la experiencia de juego global. Con el código proporcionado y un poco de creatividad, los desarrolladores pueden personalizar y mejorar aún más este juego clásico para crear una experiencia única y divertida para los jugadores.