

Documentación Extendida del Juego de la Serpiente en SpriteBach

El clásico juego de la Serpiente (Snake), cuando se implementa en un entorno de desarrollo de juegos 2D basado en *tiles* como **SpriteBach**, se rige por un diseño de programación fundamental que se centra en la abstracción de coordenadas, el control de tiempo preciso y la gestión de listas para el cuerpo del personaje.

1. La Cuadrícula Lógica y el Sistema de Coordenadas

El éxito de este juego radica en operar enteramente en un sistema de **Coordenadas Lógicas**, separando la *lógica* del juego de su *representación visual* en la pantalla.

A. Coordenadas Lógicas (Grid/Tile System)

- **Definición:** El mundo del juego se divide en una cuadrícula (grid) regular de celdas o *tiles*. Cada posición se define por un par de números **enteros** (X, Y), donde X es la columna y la y es la fila. Por ejemplo, en un tablero de 20×20 , las coordenadas van desde hasta
- **Componentes Afectados:**
 - **Serpiente:** Cada segmento del cuerpo de la serpiente (incluyendo la cabeza) se almacena como una coordenada lógica.
 - **Comida:** La posición del ítem de comida también es una única coordenada lógica.
 - **Muros/Bordes:** Los límites del tablero se definen por los valores máximos y mínimos de X e Y en esta cuadrícula.

B. Abstracción de Píxeles (Renderización)

- **El Método Draw:** El sistema de SpriteBach solo utiliza las coordenadas de píxeles para el **dibujado (Draw)**. Existe un factor de escala, el **Tamaño de Celda** (ej. **Tile Size** = 32 píxeles), que actúa como un conversor.
- **Fórmula de Conversión:** Si la posición lógica es y el tamaño de celda es SS , la posición de píxel se calcula como:

Esta separación garantiza que la lógica del juego sea independiente de la resolución de la pantalla.

2. El Movimiento Temporizado y la Lógica del Update

El movimiento no es fluido a nivel de píxeles, sino un **avance por pasos discretos** en la cuadrícula, controlado por un temporizador interno.

A. Control de la Velocidad del Juego

- **Uso de DeltaTime:** En el ciclo de juego (Update), la variable **DeltaTime** (el tiempo transcurrido desde el *último* fotograma) se acumula en un contador interno, por ejemplo, tiempo_acumulado.

- **Umbral de Movimiento:** La serpiente solo se mueve cuando tiempo_acumulado excede un valor predefinido llamado **Intervalo de Movimiento** (ej. \$0.15\$ segundos). Esto asegura que la velocidad del juego sea constante independientemente de la tasa de fotogramas (FPS) del ordenador.
 - *Mecánica:* Cuando se supera el intervalo, el movimiento se ejecuta, y el tiempo_acumulado se reinicia (o se le resta el intervalo).

B. Almacenamiento y Manejo del Cuerpo (Estructura de Datos)

- El cuerpo de la serpiente se representa típicamente como una **Lista** (o *Array*) de Coordenadas Lógicas. El elemento **al inicio** de la lista es siempre la **cabeza** de la serpiente.
- *Ejemplo:* cuerpo = [(5, 3), (5, 2), (5, 1)]

C. El Mecanismo de Arrastre (Desplazamiento)

Este es el paso lógico central que ocurre **solo** cuando el temporizador lo permite:

1. **Cálculo de la Nueva Cabeza:** Se determina la posición nueva la que debe moverse la cabeza actual, basándose en la **Dirección Actual** (Arriba, Abajo, Izquierda, Derecha).
2. **Inserción:** Se añade esta nueva coordenada al **inicio** de la lista del cuerpo. Ahora la lista tiene un segmento extra.
3. **Desplazamiento (No Comida):** Si la serpiente **no ha comido** en este paso: se elimina el elemento que está al **final** de la lista (la cola).
 - *Efecto:* La serpiente mantiene su longitud. Visualmente, el segmento de la cabeza avanza y el segmento de la cola desaparece, creando el efecto de *arrastre* de una serpiente que se mueve.
4. **Crecimiento (Comida):** Si la serpiente **sí ha comido** en este paso: **NO** se elimina el elemento de la cola.
 - *Efecto:* Al haber insertado la nueva cabeza, la lista ahora tiene un elemento más que en el paso anterior, resultando en un aumento permanente de la longitud de la serpiente.

3. Interacciones, Colisiones y Fin del Juego

La **Gestión de Colisiones** es sencilla y se realiza comparando si la coordenada de la **nueva cabeza** coincide con la coordenada de la comida, de un borde o de su propio cuerpo.

A. Colisión con la Comida (Crecimiento)

- **Condición:** Coordenada de Nueva Cabeza Coordenada de Comida}\$.
- **Acción 1: Crecimiento:** Se ejecuta el paso 4 del Mecanismo de Arrastre (no eliminar la cola).

- **Acción 2: Reinicio de Comida:** La comida actual se marca como 'comida', se incrementa la puntuación y se invoca una función para generar una **nueva posición de comida**.
 - **Regla de Generación:** La nueva posición debe ser **aleatoria** dentro de los límites del tablero Y debe garantizarse que **no coincida** con ninguna de las coordenadas ocupadas por el cuerpo de la serpiente.

B. Colisiones de Fin de Juego (Game Over)

El juego entra en el estado *Game Over* si la **Coordenada de la Nueva Cabeza** viola cualquiera de las siguientes reglas de la cuadrícula:

1. **Colisión con el Cuerpo (Self-Collision):**
 - **Condición:** La Coordenada de Nueva Cabeza es igual a la coordenada de **cualquier** otro segmento que ya existe en la lista del cuerpo. Esto se comprueba iterando sobre la lista del cuerpo, excluyendo la nueva cabeza.
2. **Colisión con los Bordes (Wall Collision):**
 - **Condición:** La Coordenada de Nueva Cabeza excede los límites predefinidos de la cuadrícula, por ejemplo:
 - (Sale por el borde izquierdo)
 - (Sale por el borde derecho)
 - (Sale por el borde superior)
 - (Sale por el borde inferior)