



Físicas de Videojuegos

Plataformas



Introducción

¿De Qué vamos a hablar?

- Físicas de videojuegos: Plataformas
- AABB
- Colisiones
- Mapas
- Ejemplo con Raylib

Plataformas

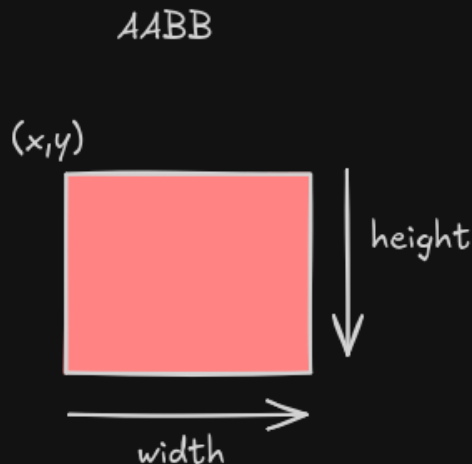
Es muy común en los videojuegos de plataformas que el personaje principal pueda saltar y aterrizar sobre plataformas. Para implementar esta mecánica, es necesario detectar colisiones entre el personaje y las plataformas.

En esta presentación, vamos a explorar cómo implementar la detección de colisiones entre un personaje y plataformas utilizando el concepto de AABB (Axis-Aligned Bounding Box) y cómo implementar esta mecánica en un juego de plataformas.



AABB (Axis-Aligned Bounding Box)

El AABB es una técnica de colisión que utiliza cajas alineadas con los ejes para representar la forma de un objeto. Estas cajas se definen por sus límites mínimos y máximos en los ejes X e Y.

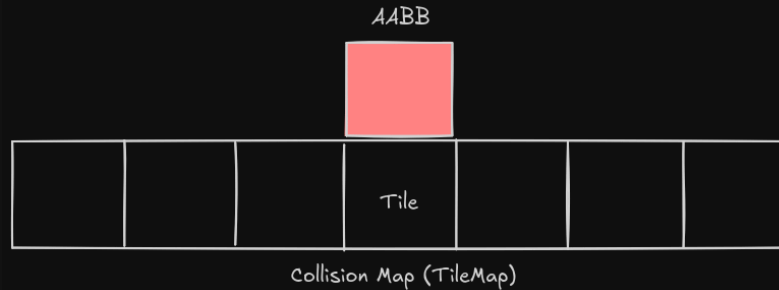


Normalmente, se utiliza un rectángulo para representar la caja de colisión de un objeto. Para detectar colisiones entre dos objetos, se comparan sus cajas AABB para ver si se superponen.

Mapa de Colisiones

Sin embargo, cuando utilizamos un objeto por si solo para representar un AABB, es difícil detectar colisiones con plataformas que tienen formas irregulares o múltiples plataformas en el mismo nivel. Para solucionar este problema, se puede utilizar un mapa de colisiones.

Un mapa de colisiones es una representación en forma de matriz que indica qué áreas del nivel son sólidas (donde el personaje puede colisionar) y cuáles son vacías (donde el personaje puede pasar). Cada celda del mapa puede tener un valor que indique si es sólida o no. Cada celda se suele llamar Tile, y el mapa se llama Tilemap.



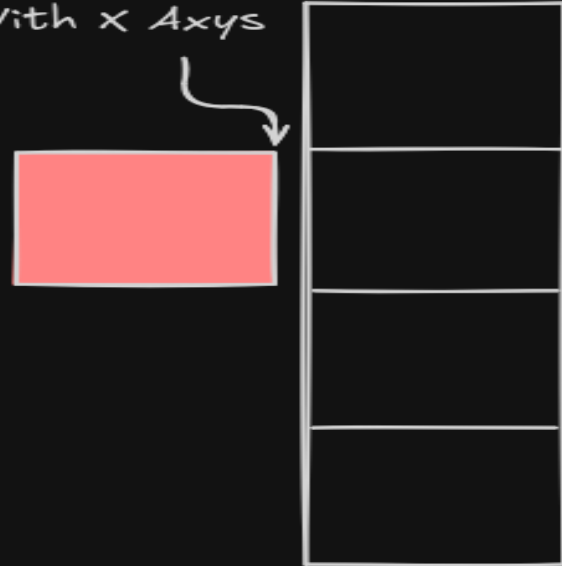
Si establecemos que cada celda del mapa de colisiones representa un área sólida, podemos detectar colisiones entre el personaje y las plataformas simplemente verificando si la posición del personaje coincide con una celda sólida en el mapa.

Detección de Colisiones

Si bien el mapa de colisiones es una forma eficiente de detectar colisiones con plataformas, también es importante implementar la lógica para manejar las colisiones una vez que se detectan. Esto incluye determinar la dirección de la colisión (por ejemplo, si el personaje está colisionando desde arriba, abajo, izquierda o derecha) y ajustar la posición del personaje en consecuencia para evitar que se quede atascado en la plataforma.

También es importante considerar la física del juego, como la gravedad y el salto, para asegurarse de que el personaje se comporte de manera realista al interactuar con las plataformas.

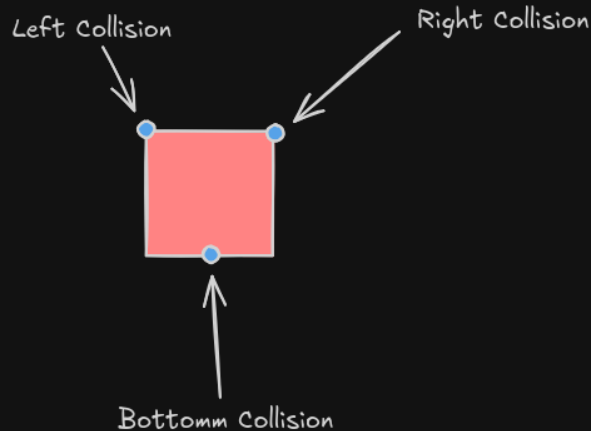
Collision
Detected
With X Axys



Cálculo de Colisiones

Es importante determinar la dirección de la colisión para ajustar la posición del personaje de manera adecuada. Por ejemplo, si el personaje está colisionando desde arriba, se debe ajustar su posición para que quede justo encima de la plataforma. Si está colisionando desde abajo, se debe ajustar su posición para que quede justo debajo de la plataforma.

Si el personaje está colisionando desde la izquierda o la derecha, se debe ajustar su posición para que quede justo al lado de la plataforma. Esto puede implicar mover al personaje fuera de la plataforma o ajustar su velocidad para simular la física del juego.



Detección de Colisiones con AABB

A la hora de implementar la detección de colisiones con AABB, se pueden seguir los siguientes pasos:

1. Definir las cajas AABB para el personaje y las plataformas. Esto implica determinar los límites mínimos y máximos en los ejes X e Y para cada objeto.
2. Comparar las cajas AABB para detectar colisiones. Esto se puede hacer verificando si los límites de las cajas se superponen en ambos ejes. Si las cajas se superponen, significa que hay una colisión entre el personaje y la plataforma.
3. Determinar la dirección de la colisión. Esto se puede hacer comparando las posiciones relativas de las cajas AABB para determinar si el personaje está colisionando desde arriba, abajo, izquierda o derecha.
4. Ajustar la posición del personaje en consecuencia para evitar que se quede atascado en la plataforma. Esto puede implicar mover al personaje fuera de la plataforma o ajustar su velocidad para simular la física del juego.

Ejemplo de Plataformas con Raylib

Vamos a ver un ejemplo simple de cómo implementar un ejemplo de plataformas utilizando Raylib, una biblioteca de desarrollo de juegos en C.

<https://github.com/makeclassicgames/Platforms>

Conclusiones

1. Las físicas de videojuegos son fundamentales para crear experiencias de juego realistas y atractivas.
2. Las plataformas son una mecánica común en los videojuegos, y la detección de colisiones es esencial para su implementación.
3. El uso de AABB y mapas de colisiones puede facilitar la detección de colisiones entre el personaje y las plataformas.
4. Es importante considerar la física del juego, como la gravedad y el salto, para asegurarse de que el personaje se comporte de manera realista al interactuar con las plataformas.
5. La implementación de estas mecánicas puede ser compleja, pero con la práctica y el uso de bibliotecas como Raylib, es posible crear juegos de plataformas divertidos y funcionales.

Referencias

- The Guide for implementing Platforms: <http://higherorderfun.com/blog/2012/05/20/the-guide-to-implementing-2d-platformers/>
- Raylib: <https://www.raylib.com/>
- "Game Programming Patterns" de Robert Nystrom
- Tiled: <https://www.mapeditor.org/>