**Documento de Programa**

**General**

A

**Tools**

* Unity (ver. 2021.3.30f)
* Visual Studio 2022 (C#)

# **Objetos**

## 1. Obj\_Player

Es el objeto principal del objeto, contiene todos los hijos y componentes esenciales para el personaje.

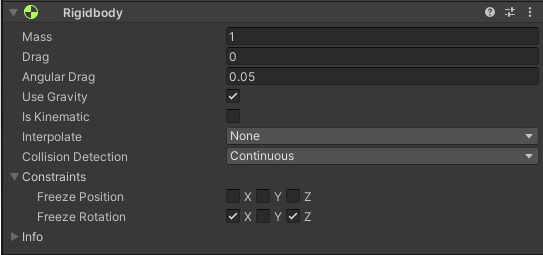
### 1.1 - Transform

Determina la posición, rotación y escala del objeto.



### 1.2 - RigidBody

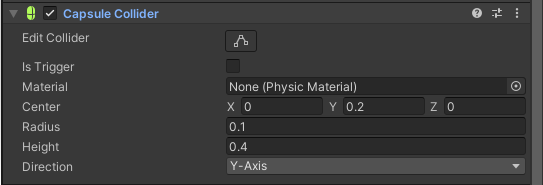
Proporciona la física a un objeto.

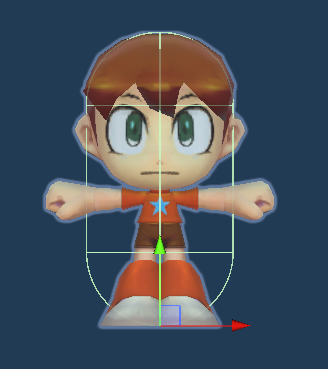


### 1.3 – Capsule Collider

Determina el espacio que ocupa el personaje.

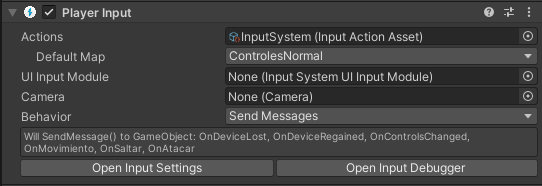
Usamos Radio y Altura para mover los límites.



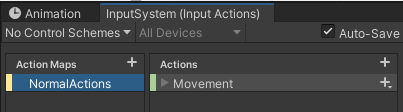


### 1.4 – Player Input

Determina el control y tipo de controles que tiene el personaje.



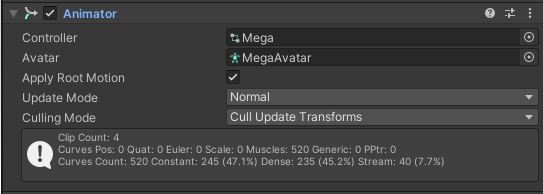
Controles Normal



* Movimiento – Stick Izquierdo

### 1.5 - Animator

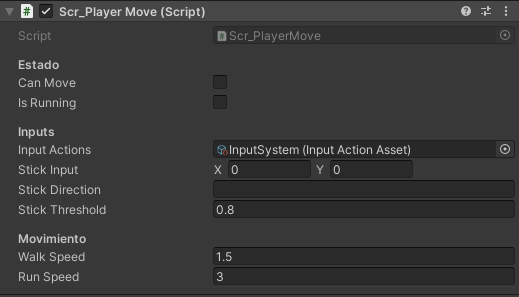
Este componente gestiona las animaciones del objeto.



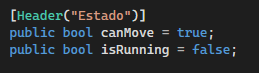
Como crear animaciones:

### 1.6 – Scr\_PlayerMove

Controla el movimiento del jugador.

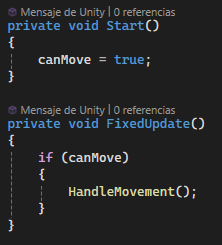


#### Codigo: Estado

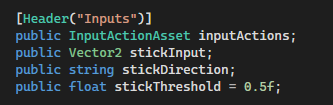


canMove: Determina si el personaje puede moverse.

isRunning: Indica si el personaje está corriendo.



#### Codigo: Inputs

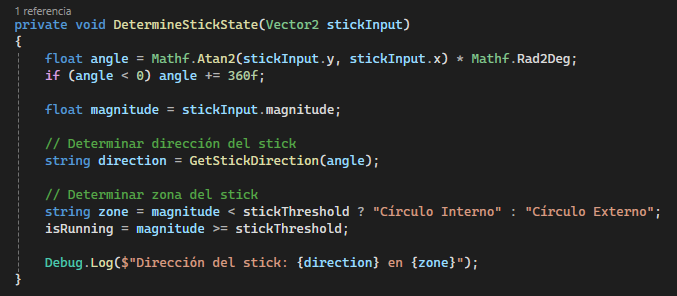


InputActions se refiere al archvio del new Input Manager que contiene los Action Maps.

stikcInput es un Vector2 que muestra lo posicion actual del Stick.

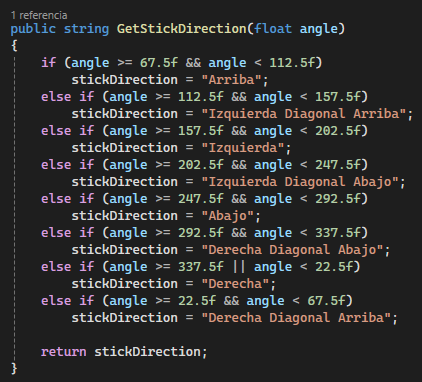
stickDirection es un string que indica la direccion actual.

stickThreshold es lo que determina la circunferencia que separa el circulo Interior y Exterior del Stick.



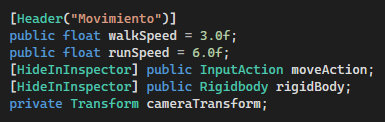
La funcion actionMap determina la dirección del Stick.

Luego, determina si el stick está en el "Círculo Interno" o "Círculo Externo" dependiendo de la magnitud de la entrada del stick, comparándola con el stickThreshold. Si la magnitud supera el umbral, el personaje estará corriendo.



Este método determina la dirección del stick en función del ángulo calculado y lo guarda en stickDirection. Cada rango de ángulos corresponde a una dirección cardinal o diagonal.

#### Codigo: Movimiento con Stick

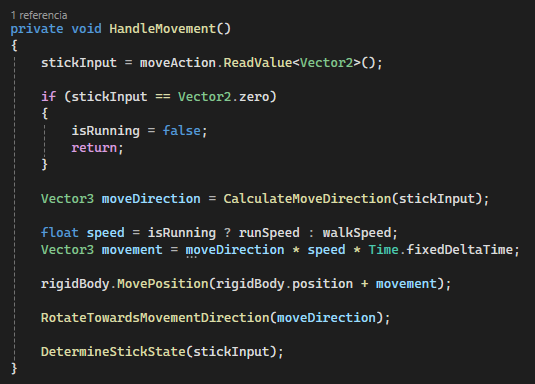


walkSpeed y runSpeed definen las velocidades de caminar y correr.

moveAction hace referencia a la acción de movimiento definida en el sistema de Input.

rigidBody es el Rigidbody del personaje, que permite aplicar física.

cameraTransform es el Transform de la cámara, usada para calcular la dirección de movimiento basada en la orientación de la cámara.



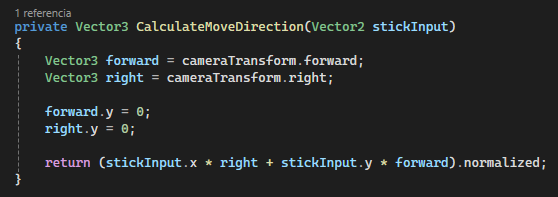
La funcion HandleMovement es la base del movimiento. Lee el valor de stickInput, y si detecta una entrada, el personaje empieza moverse.

Llama a CalculateMoveDirection(stickInput) para obtener la dirección en la que el personaje debe moverse en función de la orientación de la cámara.

Si isRunning es true, usará la velocidad de carrera, de lo contrario, usará la velocidad de caminar.

Llama a RotateTowardsMovementDirection(moveDirection) para rotar al personaje hacia la dirección en la que se está moviendo.

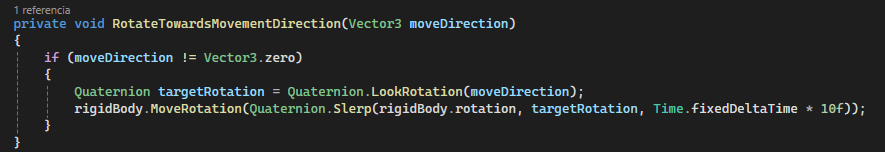
Llama a DetermineStickState(stickInput) para obtener la dirección del stick y si está en el círculo interno o externo.



El método CalculateMoveDirection calcula la dirección de movimiento en función de la cámara.

El vector forward es la dirección hacia adelante de la cámara, y right es la dirección hacia la derecha.

Se eliminan las componentes en el eje Y (para evitar movimientos verticales) y se normaliza el vector de movimiento.



Este método rota al personaje hacia la dirección en la que se está moviendo.

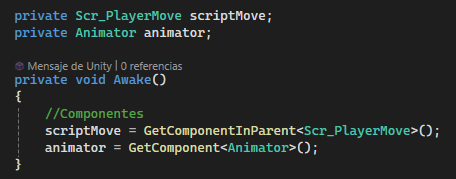
Si moveDirection no es cero, se calcula la rotación objetivo con Quaternion.LookRotation() y luego se interpola suavemente hacia esa rotación usando Slerp().

### 1.7 - Scr\_PlayerAnimations

Se usa para manejar las animaciones, la bools y los eventos.

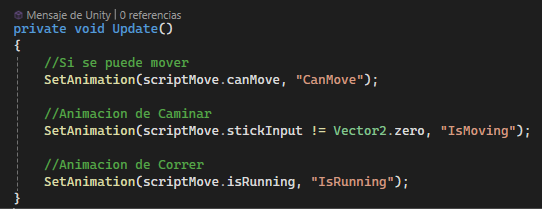
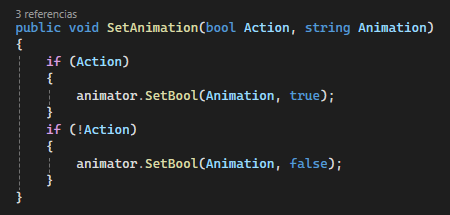


#### Code: Componentes



Tomamos como componentes el Animator y el Scr\_PlayerMove.

#### Code: Bool de Animaciones



La funcion SetAnimation tiene dos variables, Action que es la condicion y Animation que es el nombre de la bool de animacion que se prende si la condicion se cumple.

## 1.2. TargetCamara

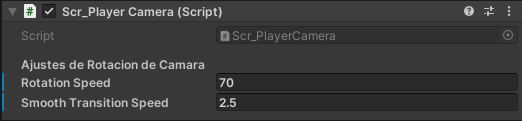
Es el objeto al que enfoca la camara y contiene parte de los componentes para controlarla.

### 1.1 - Transform

Determina la posición, rotación y escala del objeto.



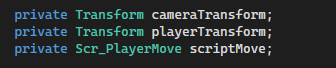
### 1.6 – Scr\_PlayerCamara



rotationSpeed: Velocidad de la rotación de la cámara.

smoothTransitionSpeed: Velocidad de rotación.

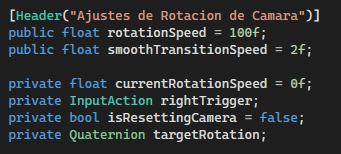
#### Codigo: Componentes



cameraTransform: Referencia al transform de la cámara virtual.

playerTransform: Referencia al transform del jugador.

scriptMove: Referencia al script que controla el movimiento del jugador (Scr\_PlayerMove).



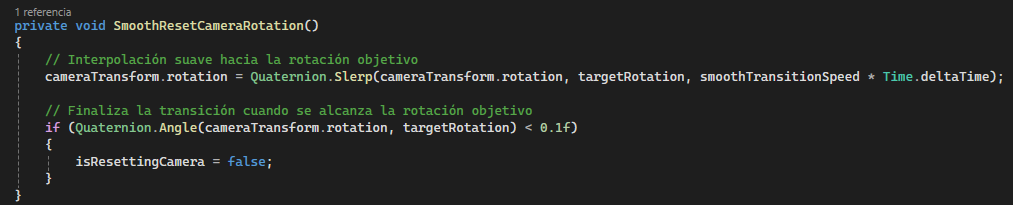
currentRotationSpeed: Velocidad actual de rotación que puede variar suavemente.

rightTrigger: Acción asociada al gatillo derecho para resetear la cámara.

isResettingCamera: Bool que indica si la cámara está en proceso de reseteo.

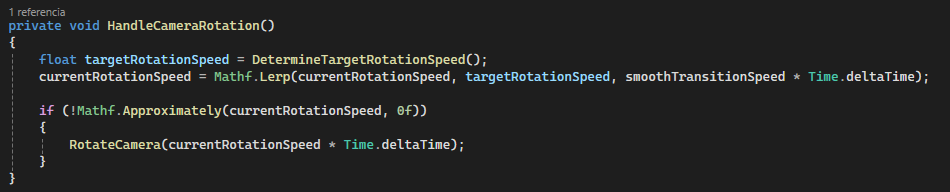
targetRotation: Rotación objetivo de la cámara durante el reseteo.

#### Codigo: Metodos



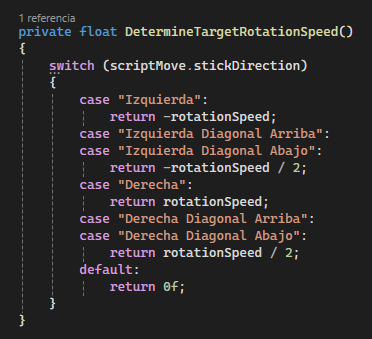
Método SmoothResetCameraRotation().

Realiza una interpolación suave de la rotación actual de la cámara hacia la targetRotation. Cuando la cámara casi alcanza la rotación objetivo se detiene el proceso de reseteo.



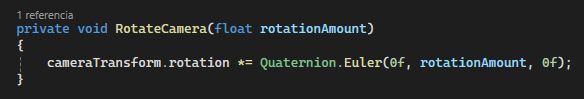
Método HandleCameraRotation().

Maneja la rotación de la cámara basándose en la dirección del stick. Calcula la velocidad de rotación objetivo y realiza una interpolación suave hacia esta velocidad. Si la velocidad de rotación actual no es cero, rota la cámara en consecuencia.



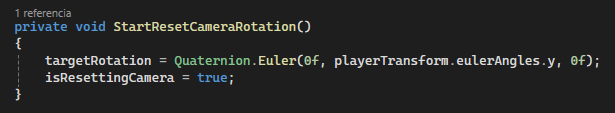
Método DetermineTargetRotationSpeed().

Este método determina la velocidad de rotación de la cámara basándose en la dirección del stick (por ejemplo, rotación completa a la izquierda, rotación diagonal más lenta, etc.).



Método RotateCamera().

Rota la cámara aplicando una rotación en el eje Y, multiplicando el quaternion actual de la cámara por uno nuevo calculado en función de rotationAmount.



Método StartResetCameraRotation().

Se activa cuando el jugador presiona el gatillo derecho. Establece la rotación objetivo de la cámara alineada con la rotación del jugador en el eje Y y activa el proceso de reseteo (isResettingCamera).