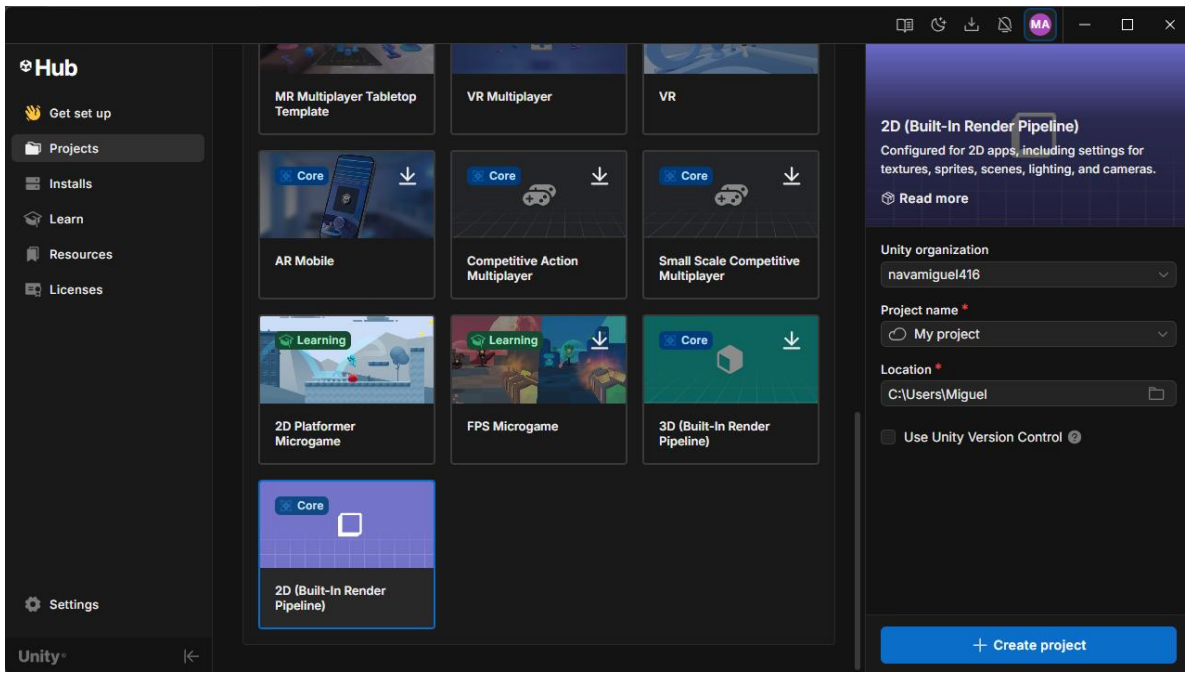
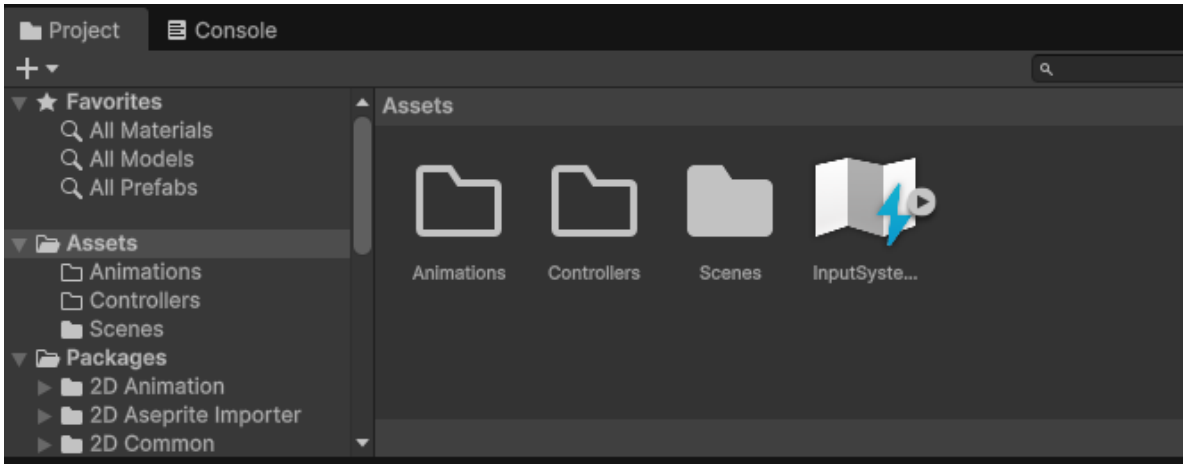
Crear un nuevo proyecto en Unity





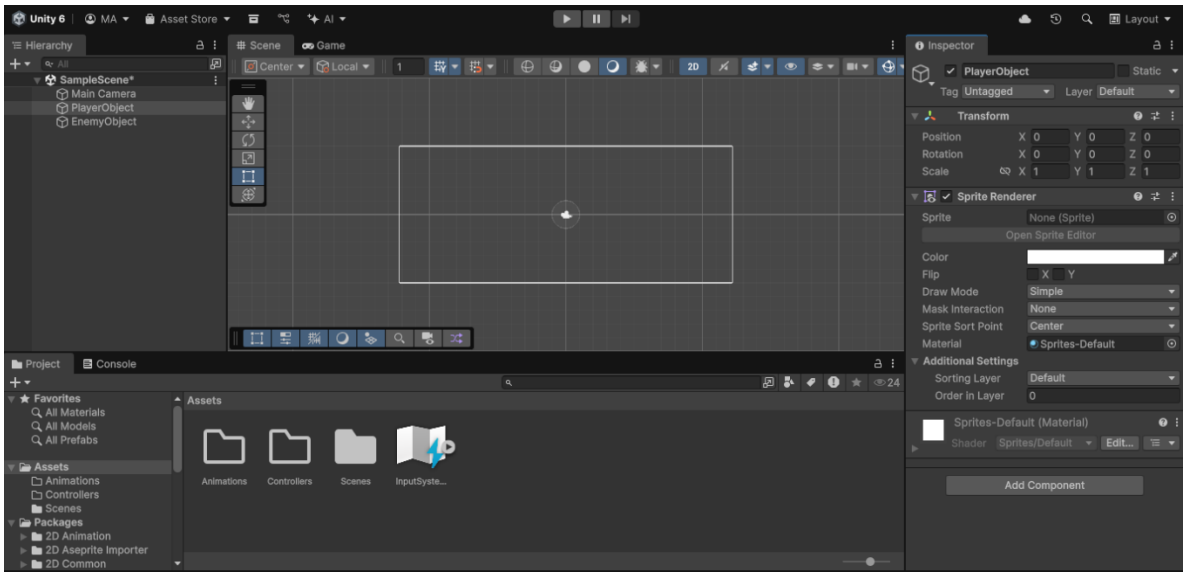
Instrucciones:

1. Explora el entorno Unity: Abre un proyecto nuevo en Unity y observa las diferentes ventanas que aparecen por defecto.

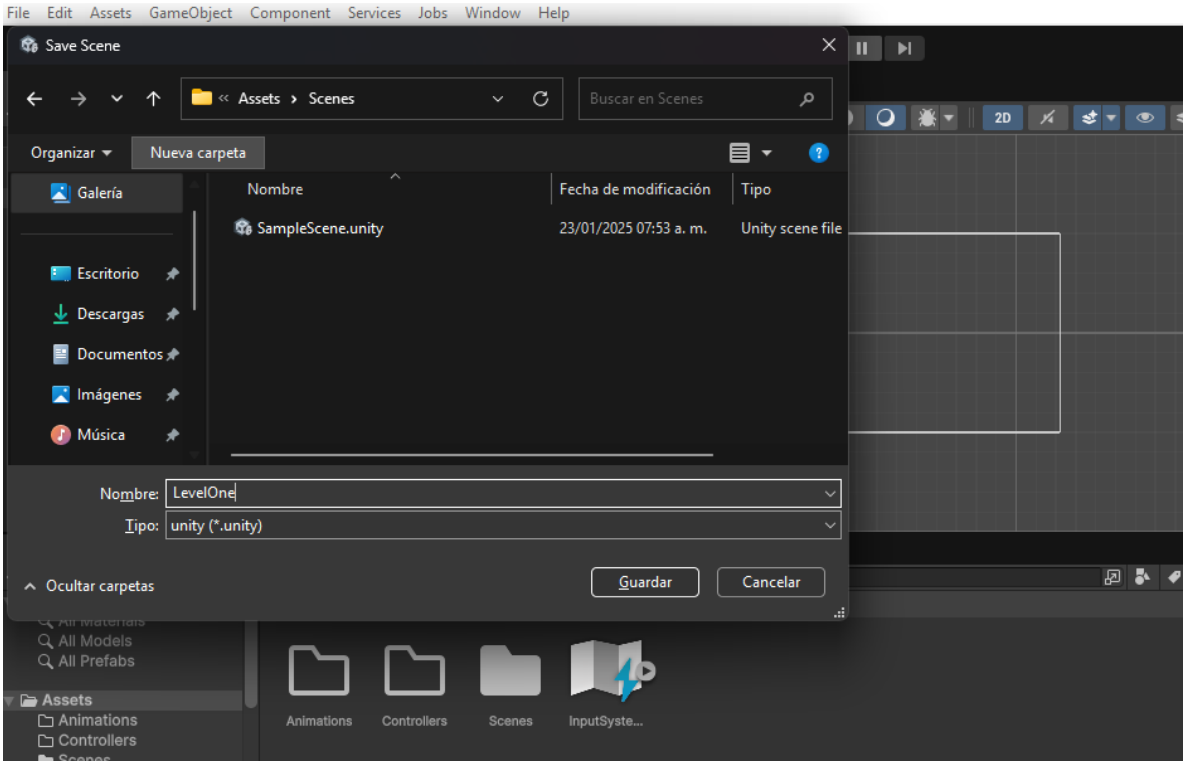
2. Relaciona funciones: Completa el siguiente cuadro relacionando cada vista con su función principal:

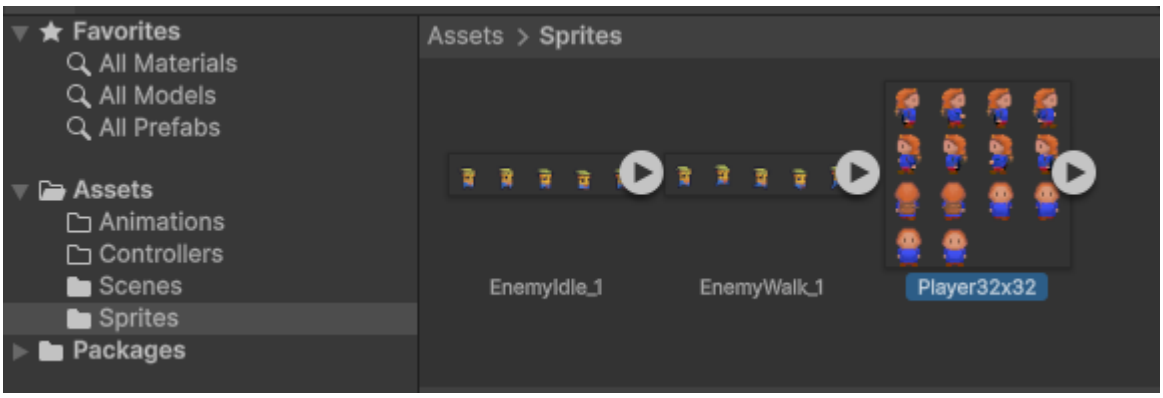
Vista	Función Principal	¿Qué puedes hacer ahí?
Scene	Visualizar y editar la escena en 3D	Mover, rotar, escalar objetos; navegar por el mundo 3D; colocar GameObjects
Game	Mostrar cómo se ve el juego jugable	Ver el resultado final como lo verá el jugador; probar la cámara y UI
Hierarchy	Organizar objetos en estructura jerárquica	Crear, eliminar, renombrar objetos; establecer relaciones padre-hijo
Inspector	Mostrar propiedades del objeto seleccionado	Modificar componentes; ajustar valores; añadir/quitar componentes
Project	Administrar archivos y recursos	Importar assets; crear scripts; organizar carpetas; gestionar materiales
Console	Mostrar mensajes del sistema	Ver errores de código; debug; warnings; mensajes personalizados

Agregar el componente (Sprite Renderer) para PlayerObject y EnemyObject

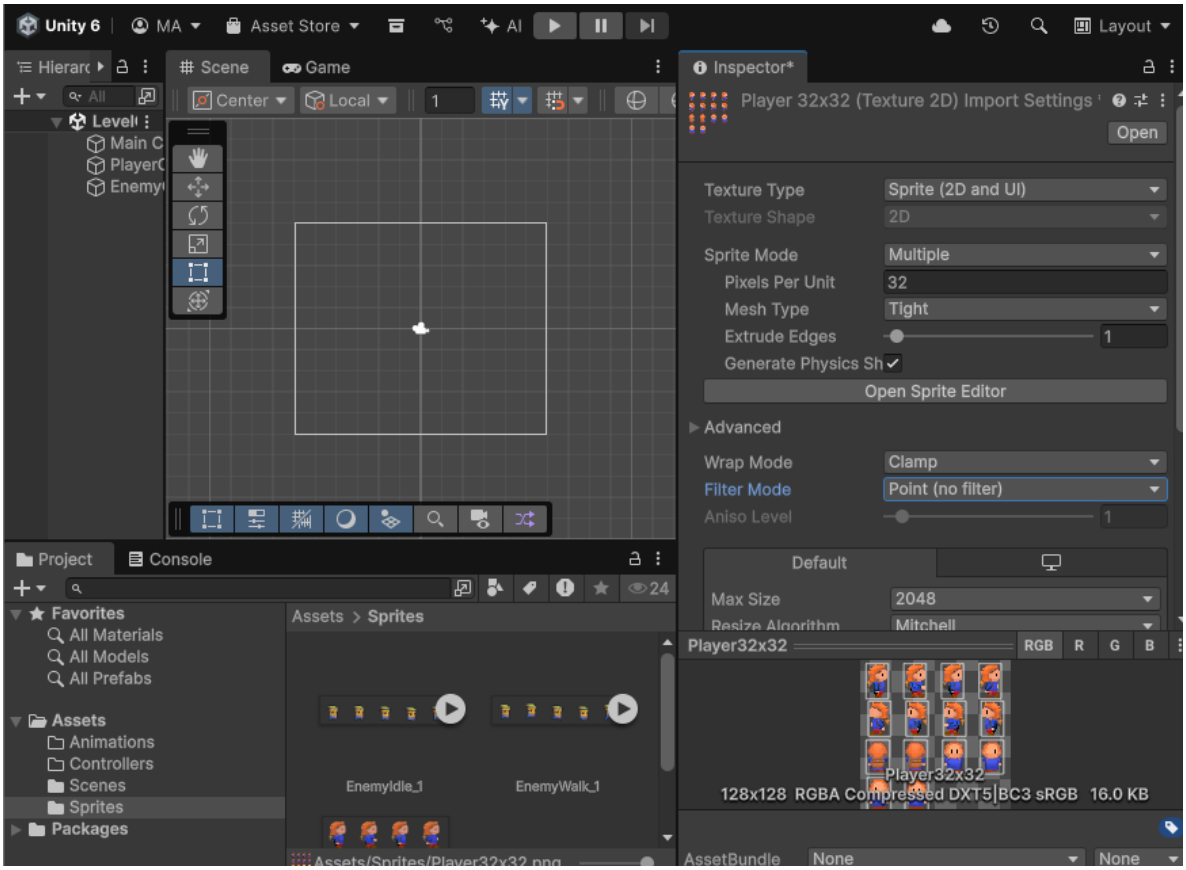


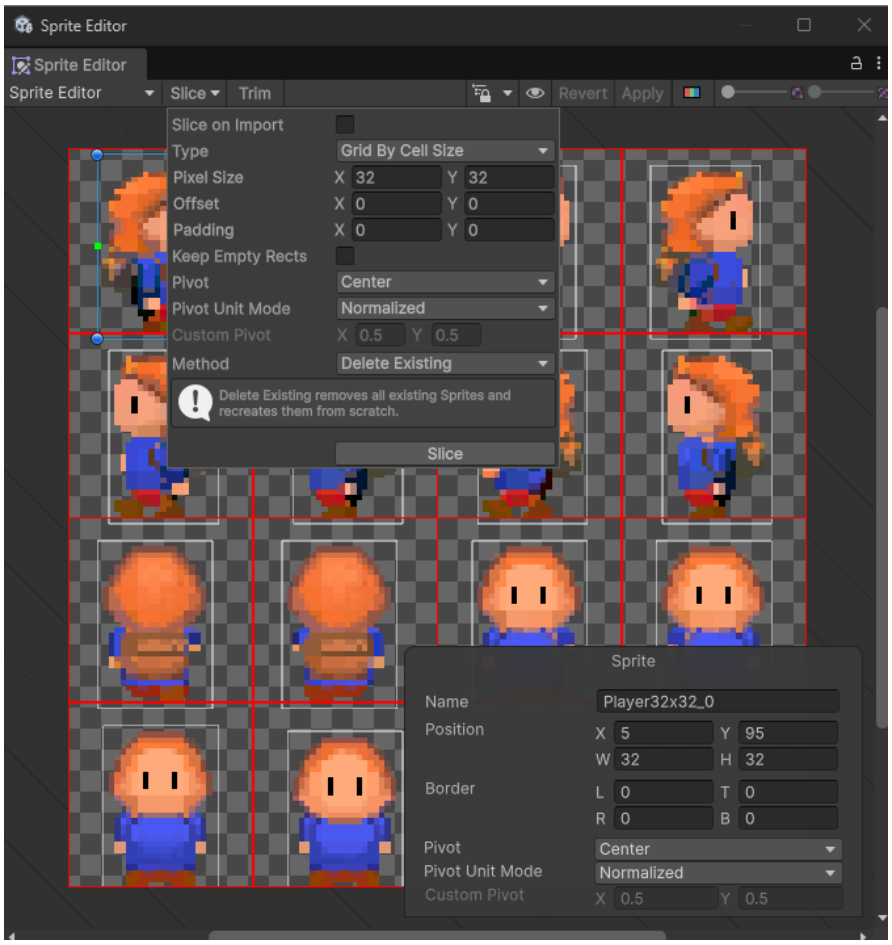
Guardarlo en Scenes y darle el nombre de LevelOne

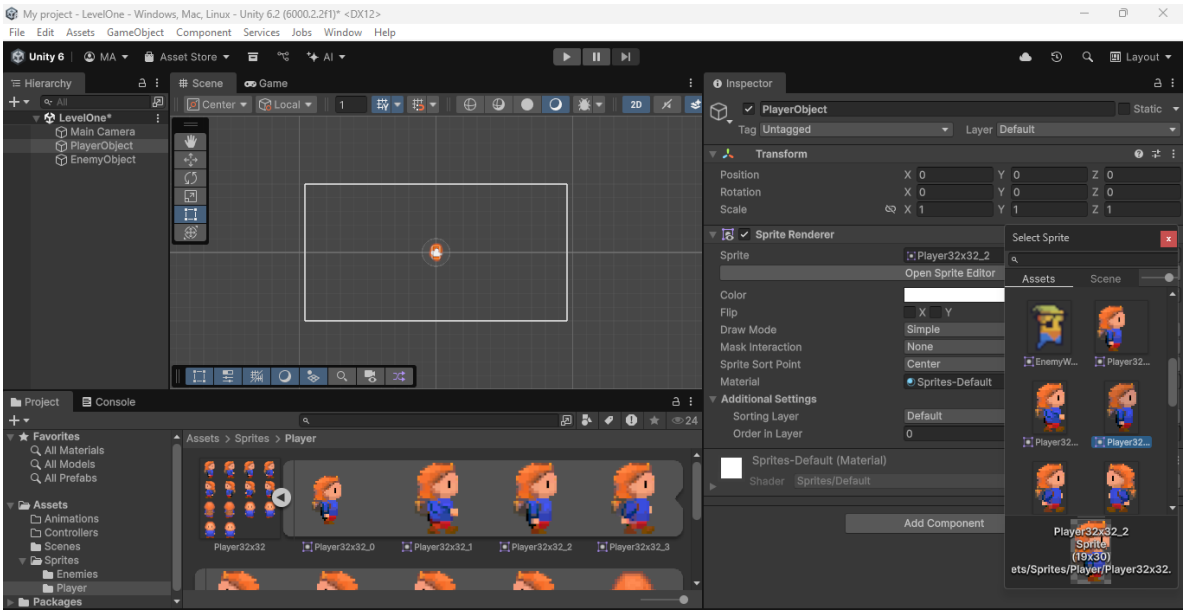




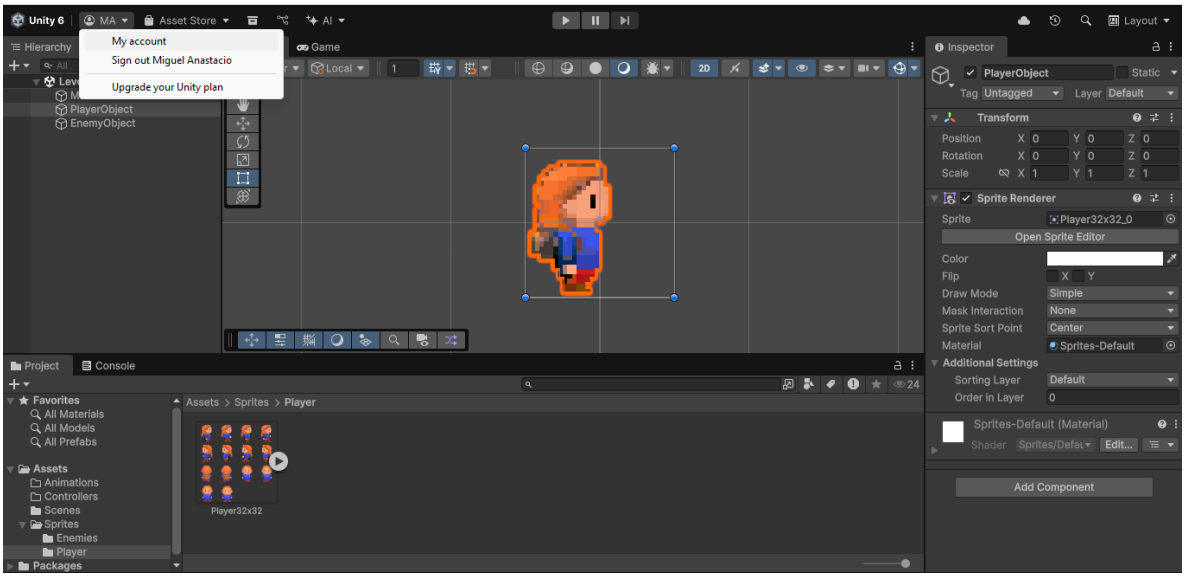
Se hicieron modificaciones para el jugador

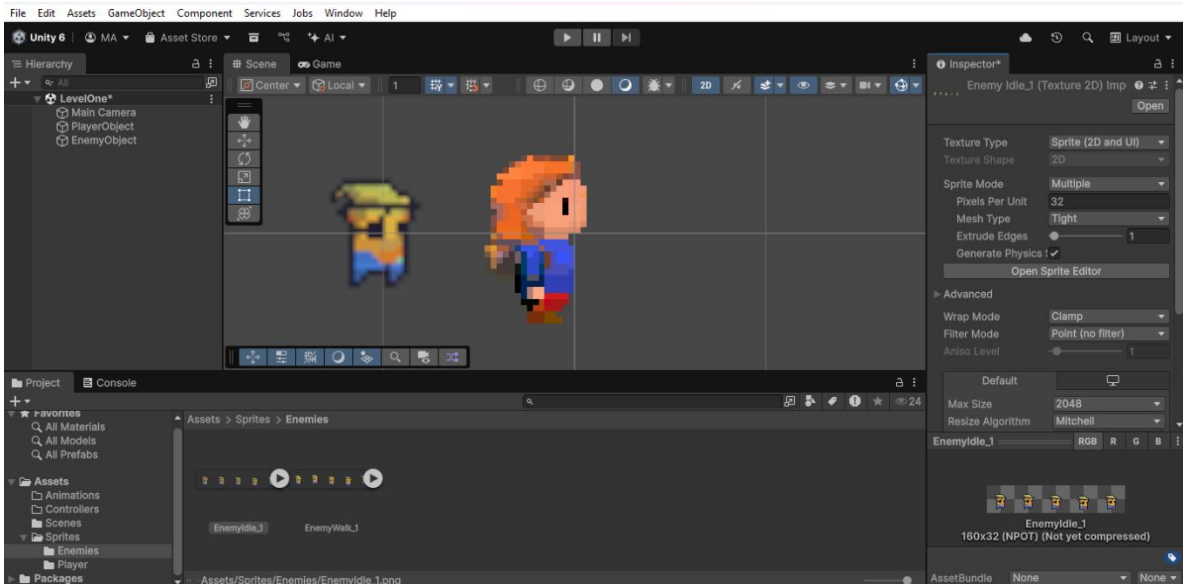



Sprite Editor
Sprite Editor
Sprite Editor
Slice
Trim
Revert
Apply
Slice on Import
Type
Grid By Cell Size
Pixel Size
X 32
Y 32
Offset
X 0
Y 0
Padding
X 0
Y 0
Keep Empty Rects
Pivot
Center
Pivot Unit Mode
Normalized
Custom Pivot
X 0.5
Y 0.5
Method
Delete Existing
Delete Existing removes all existing Sprites and recreates them from scratch.
Slice
Sprite
Name
Player32x32_0
Position
X 5
Y 95
W 32
H 32
Border
L 0
T 0
R 0
B 0
Pivot
Center
Pivot Unit Mode
Normalized
Custom Pivot
X 0.5
Y 0.5

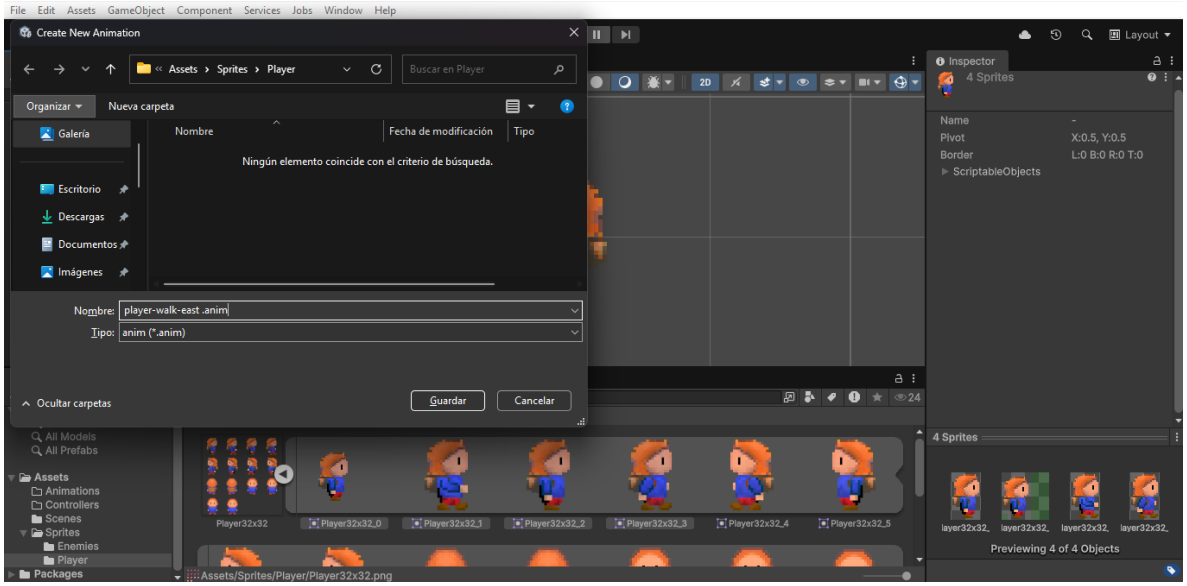

My project - LevelOne - Windows, Mac, Linux - Unity 6.2 (6000.2.2f1)* <DX12>
File Edit Assets GameObject Component Services Jobs Window Help
Unity 6
Asset Store
Layout
Hierarchy
Scene
Game
Inspector
LevelOne*
Main Camera
PlayerObject
EnemyObject
PlayerObject
Static
Tag Untagged
Layer Default
Transform
Position
Rotation
Scale
Sprite Renderer
Sprite
Player32x32_2
Open Sprite Editor
Color
Flip
Draw Mode
Simple
Mask Interaction
None
Sprite Sort Point
Center
Material
Sprites-Default
Additional Settings
Sorting Layer
Default
Order in Layer
0
Sprites-Default (Material)
Shader
Sprites/Default
Add Component
Select Sprite
Assets
Scene
Project
Console
Favorites
All Materials
All Models
All Prefabs
Assets
Animations
Controllers
Scenes
Sprites
Enemies
Player
Packages
Assets > Sprites > Player
Player32x32
Player32x32_0
Player32x32_1
Player32x32_2
Player32x32_3
Player32x32_2
Sprite
(19x30)
ets/Sprites/Player/Player32x32.

Poner los sprint a trabajar como jugador y enemigo

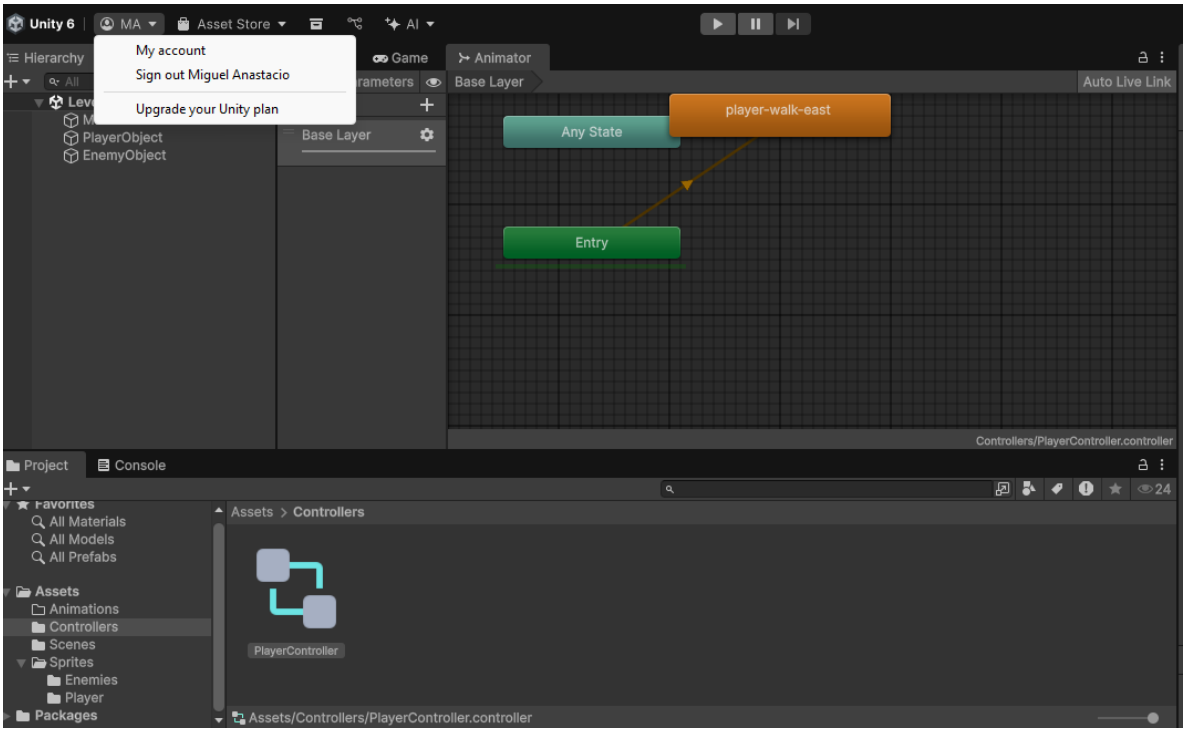




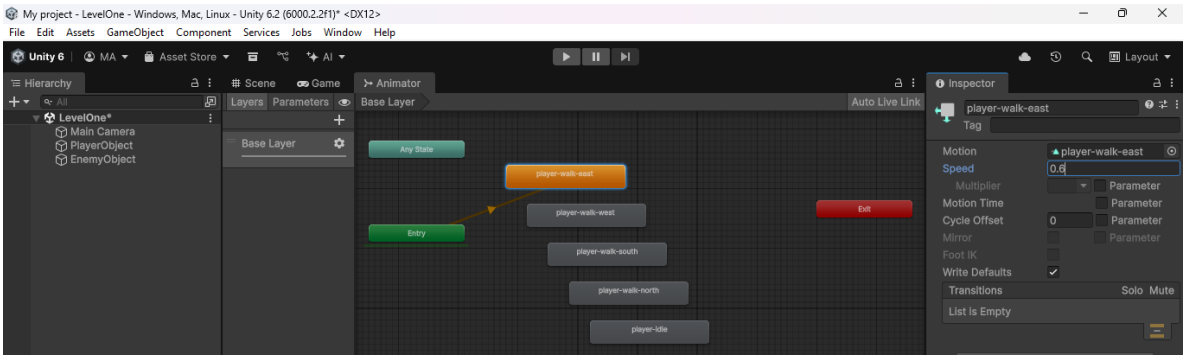
Mover las primeras 4 objetos del jugador a PlayerObject y nombraslos como player-walk-east .anim

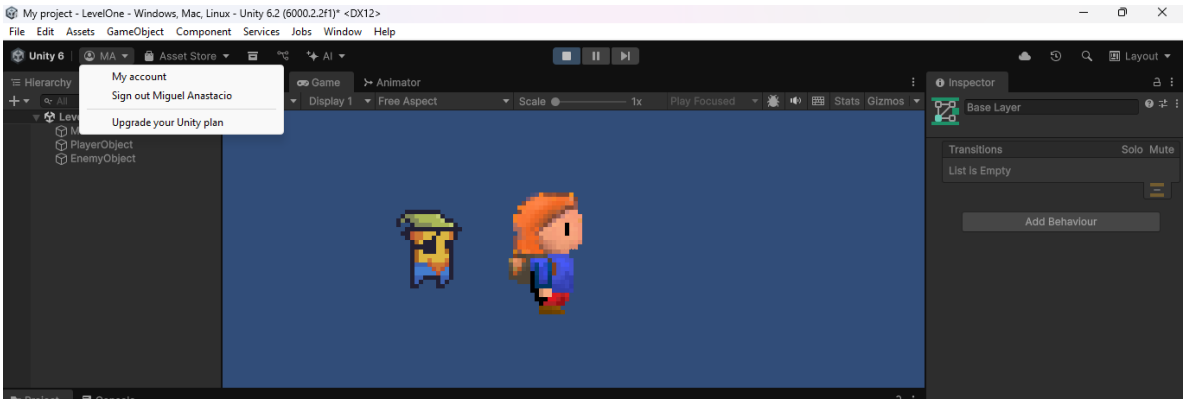


Renombra el controller como PlayerController y arrastralo a la carpeta Animations|Controllers

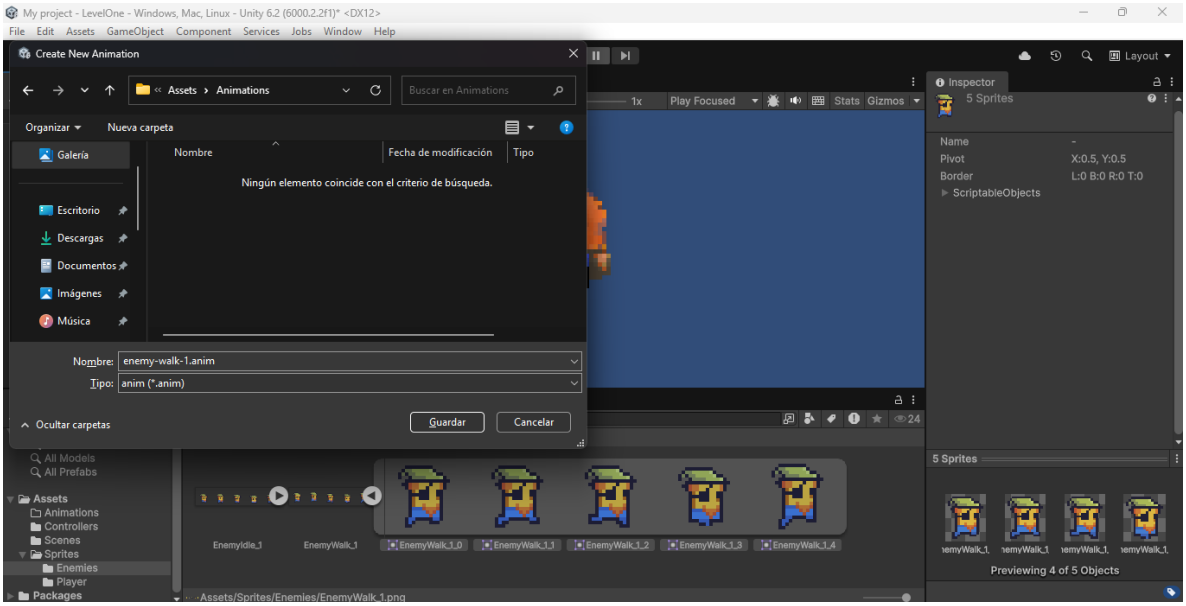


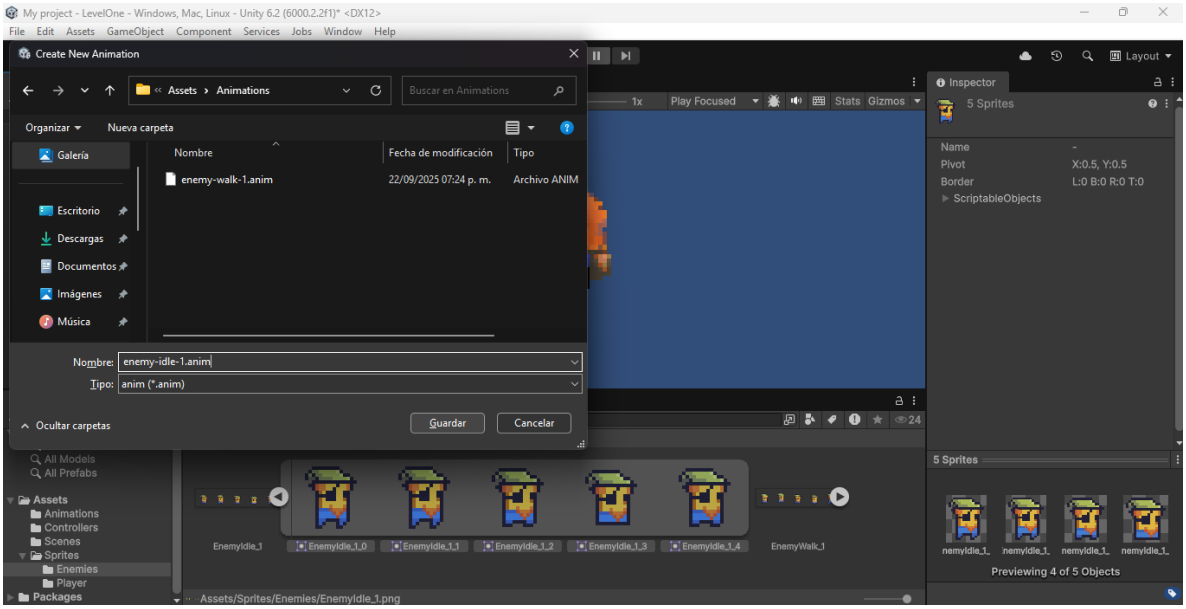
Cambiar la velocidad del jugador

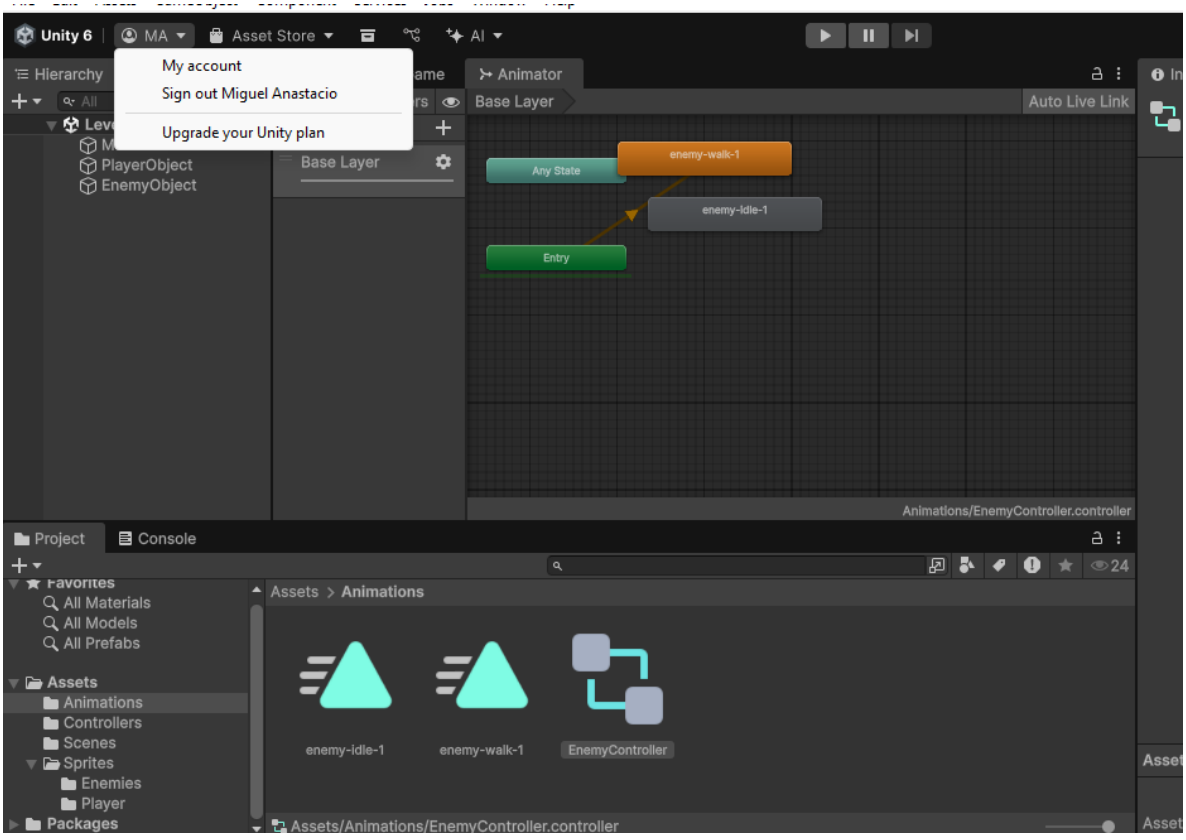




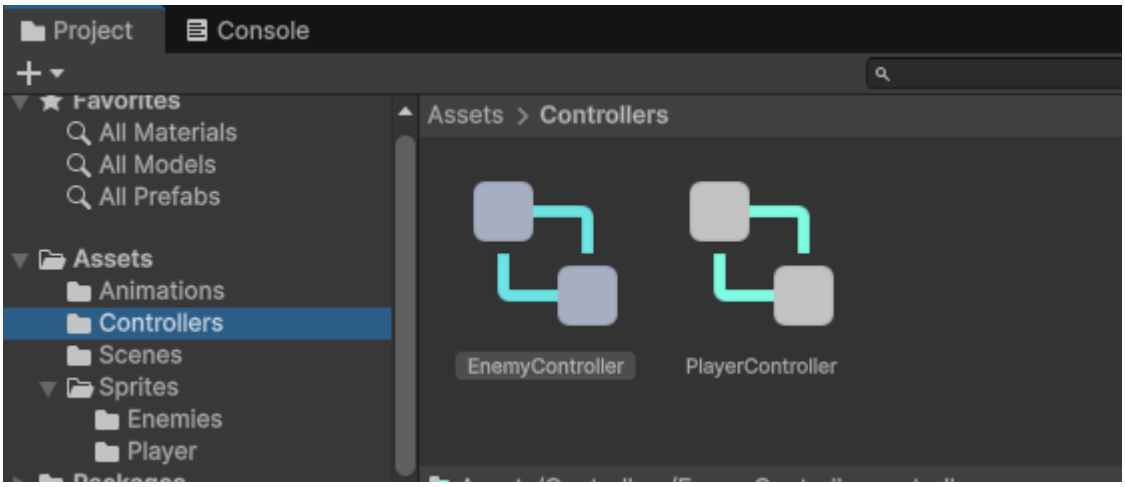
Hacer lo mismo para enemigo



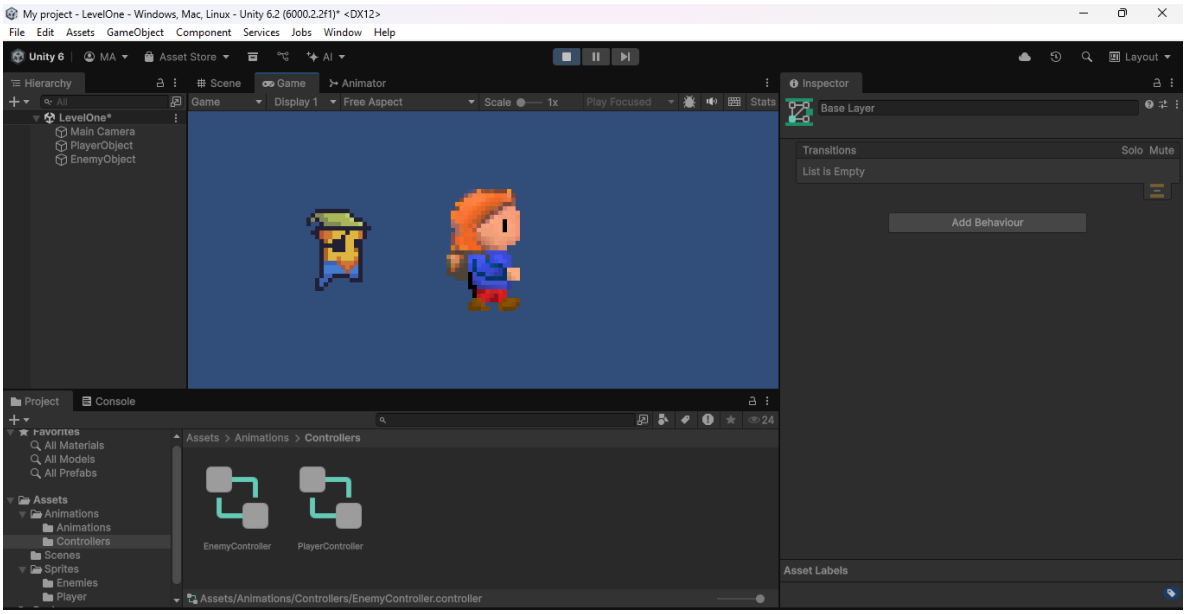

Create New Animation
Assets › Animations
Buscar en Animations
Organizar
Nueva carpeta
Galería
Escritorio
Descargas
Documentos
Imágenes
Música
Nombre
Fecha de modificación
Tipo
enemy-walk-1.anim
22/09/2025 07:24 p. m.
Archivo ANIM
Nombre: enemy-idle-1.anim
Tipo: anim (*.anim)
Ocultar carpetas
Guardar
Cancelar
Inspector
5 Sprites
Name
Pivot
X:0.5, Y:0.5
Border
L:0 B:0 R:0 T:0
ScriptableObjects
Previewing 4 of 5 Objects
Assets/Sprites/Enemies/EnemyIdle_1.png


Unity 6
MA
Asset Store
AI
My account
Sign out Miguel Anastacio
Upgrade your Unity plan
Hierarchy
PlayerObject
EnemyObject
Animator
Base Layer
Auto Live Link
Any State
enemy-walk-1
enemy-idle-1
Entry
Animations/EnemyController.controller
Project
Console
Favorites
All Materials
All Models
All Prefabs
Assets
Animations
Controllers
Scenes
Sprites
Enemies
Player
Packages
Assets > Animations
enemy-idle-1
enemy-walk-1
EnemyController
Assets/Animations/EnemyController.controller

Arrastra el EnemyController a la carpeta Animations → Controllers



Darle play para ver como esta el estado de los personajes



Preguntas y Respuestas

 ¿Qué vista te permite modificar propiedades de un objeto seleccionado? Respuesta: La vista **Inspector**. Cuando seleccionas cualquier GameObject en la Hierarchy o Scene, el Inspector muestra todos sus componentes y propiedades, permitiendo modificar valores, añadir o quitar componentes.

 ¿Dónde ves el resultado de tu juego en tiempo real? Respuesta: En la vista **Game**. Esta ventana muestra exactamente lo que ve el jugador a través de la cámara principal, permitiendo probar la jugabilidad y ver el resultado final del juego.

 ¿Qué vista muestra los errores y mensajes del sistema? Respuesta: La vista **Console**. Aquí aparecen todos los errores de compilación, warnings, mensajes de Debug.Log() y información del sistema que ayuda en la depuración.

 ¿Cuál es la diferencia entre Scene y Game? Respuesta: La vista **Scene** es el espacio de trabajo del desarrollador donde puedes editar, posicionar y manipular objetos libremente con herramientas de desarrollo. La vista **Game** muestra únicamente lo que ve el jugador a través de la cámara, sin herramientas de edición visibles.

 ¿Qué es un objeto CreateEmpty? ¿Qué características tiene? Respuesta: Es un GameObject vacío que solo tiene el componente Transform por defecto. Sus características son: no tiene representación visual, sirve como contenedor para otros objetos o componentes, es útil para organizar la jerarquía, y puede actuar como objeto padre para agrupar otros GameObjects.

==========

 ¿Qué es un SpriteSheet y para qué se utiliza en el desarrollo de videojuegos? Respuesta: Un SpriteSheet es una imagen única que contiene múltiples sprites individuales organizados en una cuadrícula. Se utiliza para optimizar el rendimiento del juego, reducir el número de archivos de imagen, y crear animaciones fluidas mediante la secuencia rápida de sprites.

 ¿Cuál es la ventaja de usar un SpriteSheet en lugar de imágenes individuales? Respuesta: Las ventajas incluyen: menor uso de memoria RAM, reducción de llamadas de renderizado (draw calls), organización más eficiente de assets, menor tiempo de carga, y optimización del espacio en disco al eliminar duplicación de metadatos.

 ¿Qué tipo de animaciones se pueden crear con un SpriteSheet? Respuesta: Se pueden crear animaciones de: movimiento (caminar, correr, saltar), estados emocionales (idle, atacar, morir), efectos visuales (explosiones, magia), transformaciones de personajes, y cualquier secuencia de frames que represente cambio visual.

 ¿Qué formato de archivo es común para SpriteSheets (PNG, JPG, etc.) y por qué? Respuesta: El formato **PNG** es el más común porque soporta transparencia (canal alfa), mantiene la calidad sin compresión con pérdida, es compatible con Unity, y preserva los detalles necesarios para sprites de juegos 2D.

 ¿Cómo se relaciona el Sprite Renderer con un SpriteSheet en Unity? Respuesta: El Sprite Renderer es el componente que muestra visualmente un sprite en la pantalla. Toma un sprite

individual (extraído del SpriteSheet) y lo renderiza en la posición del GameObject. Es el puente entre los datos del sprite y su representación visual.

 ¿Qué herramienta de Unity permite dividir un SpriteSheet en múltiples sprites? Respuesta: El **Sprite Editor**, accesible desde Window → 2D → Sprite Editor. Esta herramienta permite cortar automática o manualmente un SpriteSheet en sprites individuales usando la función "Slice".

 ¿Qué parámetros puedes ajustar al importar un SpriteSheet en Unity? Respuesta: Los parámetros principales son: Texture Type (Sprite 2D and UI), Sprite Mode (Single/Multiple), Pixels Per Unit (PPU), Filter Mode (Point/Bilinear), Compression (None/Low Quality/Normal Quality), y configuraciones de corte en el Sprite Editor.

==========

 ¿Qué es una animación en Unity y cómo se representa en el editor? Respuesta: Una animación en Unity es una secuencia de cambios en las propiedades de un GameObject a lo largo del tiempo. Se representa como un archivo de Animation Clip (.anim) que contiene keyframes y curvas de animación, visible en la ventana Animation donde se pueden editar los frames y propiedades.

 ¿Qué es el componente Animator y para qué se utiliza? Respuesta: El componente Animator es el encargado de controlar y reproducir animaciones en un GameObject. Utiliza un Animator Controller para gestionar múltiples clips de animación, transiciones entre estados, y parámetros que determinan qué animación se reproduce según las condiciones del juego.

 ¿Qué es un Animator Controller y cómo se relaciona con el componente Animator? Respuesta: El Animator Controller es un asset (.controller) que define la máquina de estados de animación. Contiene los estados (animaciones), transiciones entre ellos, parámetros, y la lógica de control. El componente Animator lo utiliza como "cerebro" para saber qué animación reproducir y cuándo cambiar entre ellas.

 ¿Qué función cumple el panel Animator en la gestión de estados? Respuesta: El panel Animator (ventana Animator) permite visualizar y editar la máquina de estados de forma gráfica. Muestra los estados como nodos conectados, permite crear transiciones arrastrando entre estados, configurar condiciones de cambio, y ajustar parámetros de cada estado de animación.

 ¿Qué es un "State" y qué representa dentro del Animator? Respuesta: Un "State" (estado) representa una situación específica del objeto (como caminar, estar idle, atacar, etc.). Cada estado está asociado a un clip de animación y define qué se reproduce cuando el objeto se encuentra en esa condición particular. Los estados se conectan mediante transiciones que determinan cuándo cambiar de uno a otro.