Universidad Politécnica Salesiana

UNIDAD 3 - Entornos para desarrollo y programación de videojuegos para plataformas de escritorio y móviles.

Diseño de Videojuego 3D

Docente: Ing. Ángel Pérez

Carrera de Diseño Multimedia
NIVEL 5
G1 - DIURNO

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR

Desarrollo de videojuego 3D y assets de apoyo

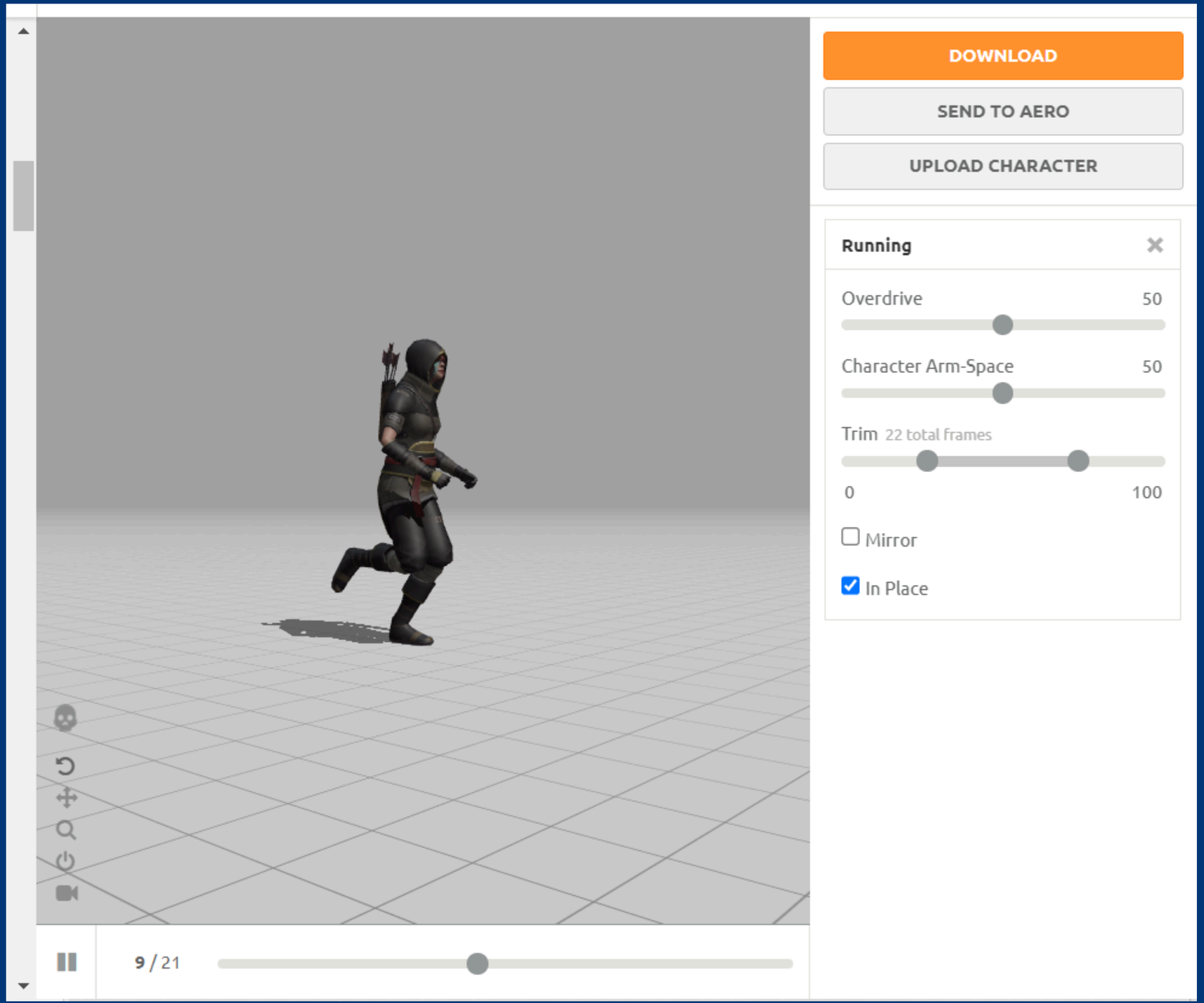
Carrera de Diseño Multimedia

Contenido

1 Movimiento del personaje y animación

1

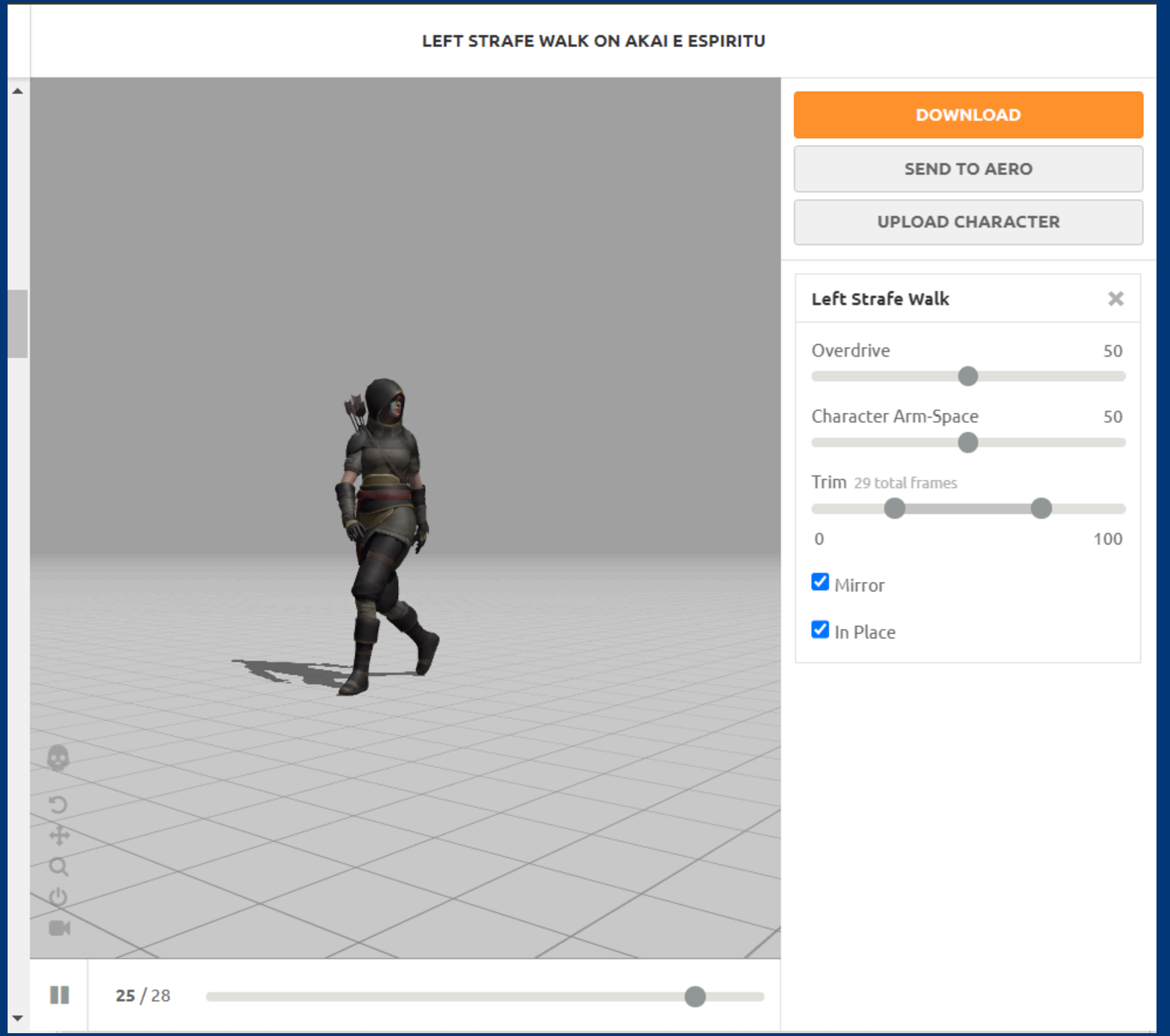
Descargamos el personaje y las animaciones necesarias



1

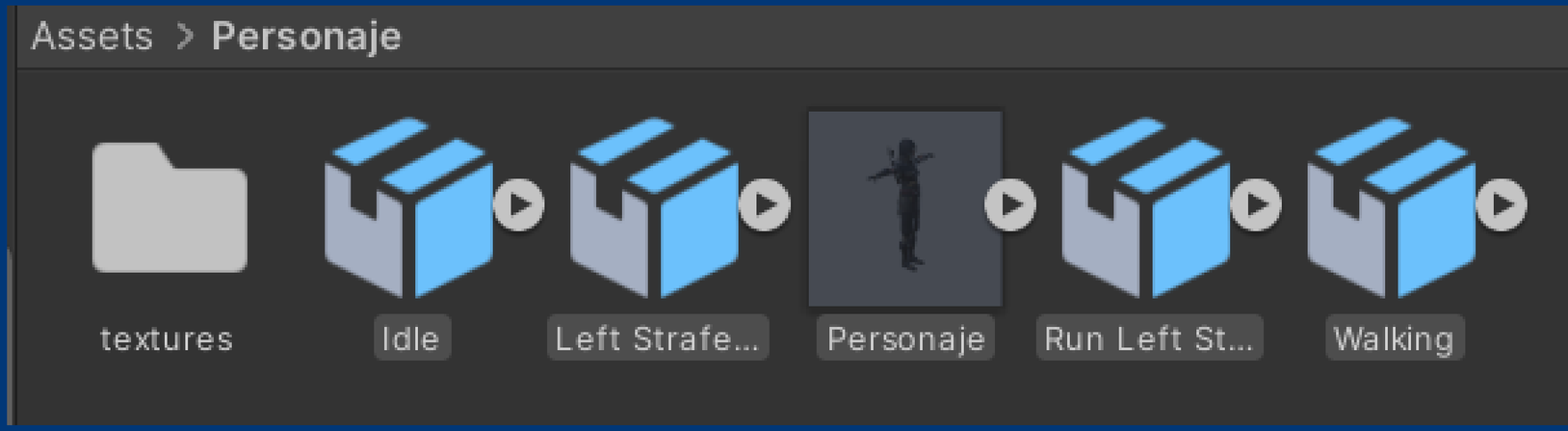
Las animaciones deben estar en su propio sitio (No de deben mover), en mixamo activamos **In Place**

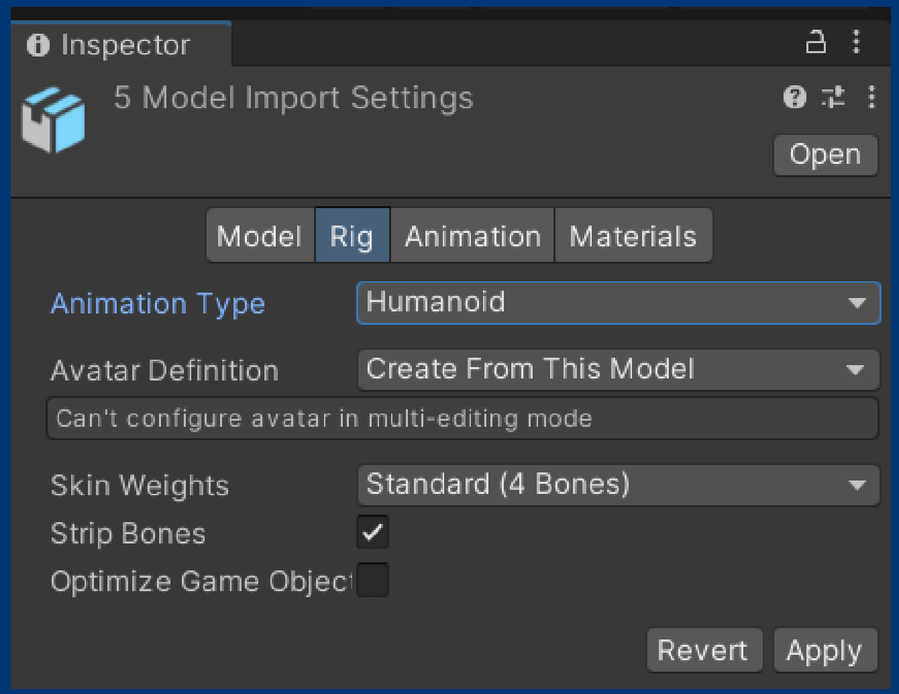
Si hay una animacion que va a la izquierda y queremos que sea a la derecha activamos **Mirror,** tambien se puede hacer en Unity



1

Agrego los objetos descargados al espacio de Assets de Unity

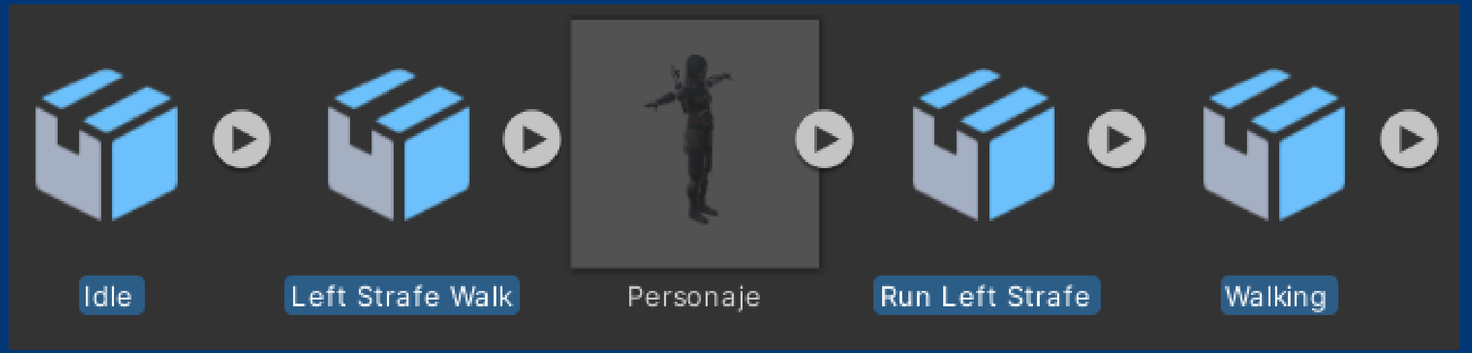




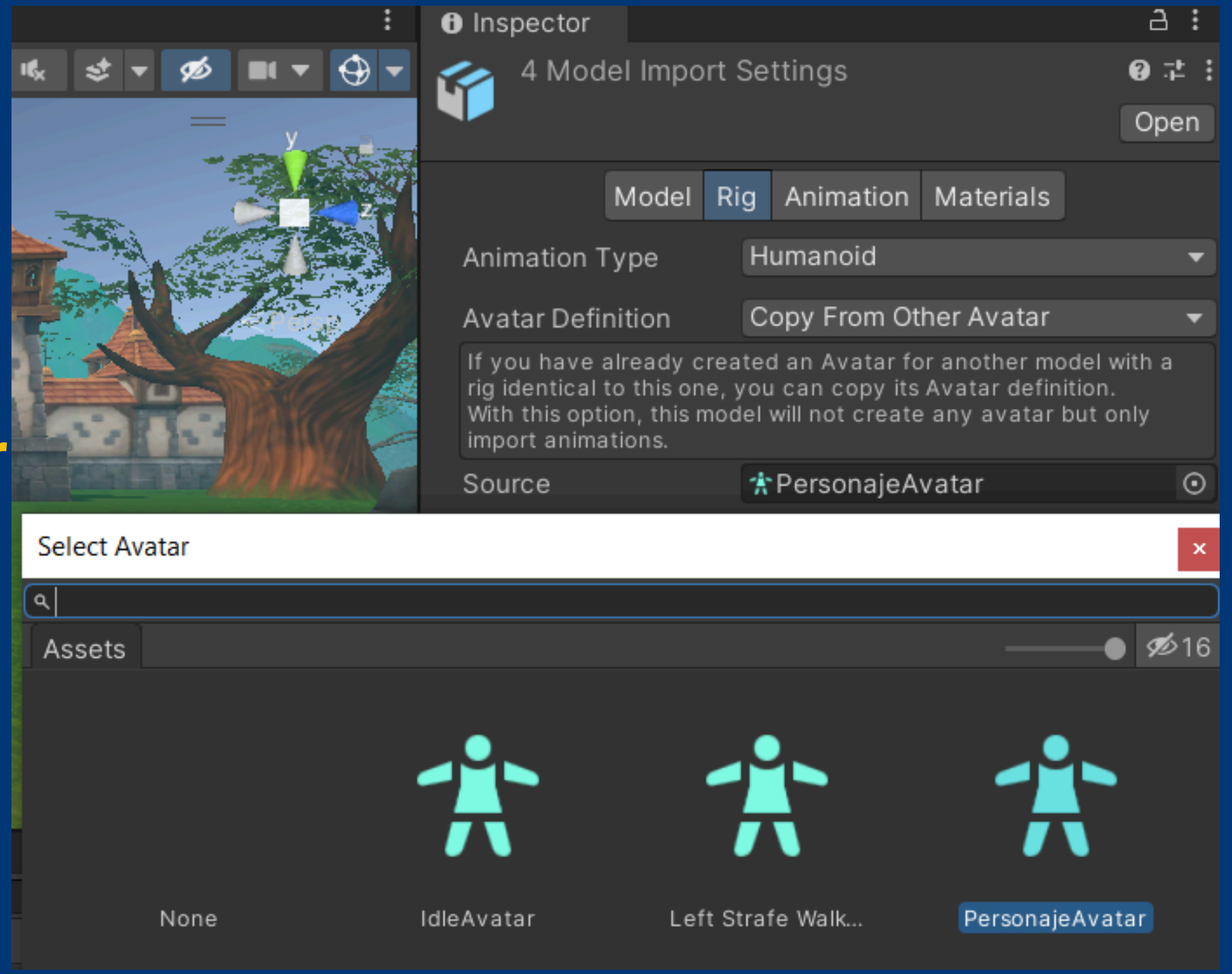
- Selecciono el personaje y las animaciones
- En el inspector navego a **Rig**
- En Animation type cambio a **Humanoid**
- Click en el boton **Apply**

1

Seleccionamos únicamente las Animaciones



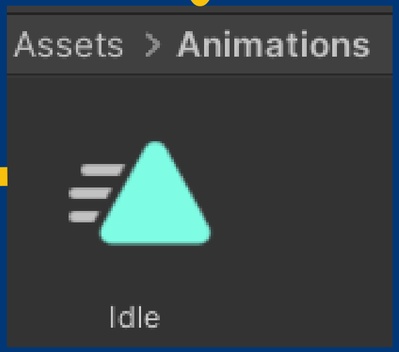
En la opción de Avatar Definition
Seleccionamos Copy From Other Avatar
Donde seleccionamos o arrastramos el PersonajeAvatar



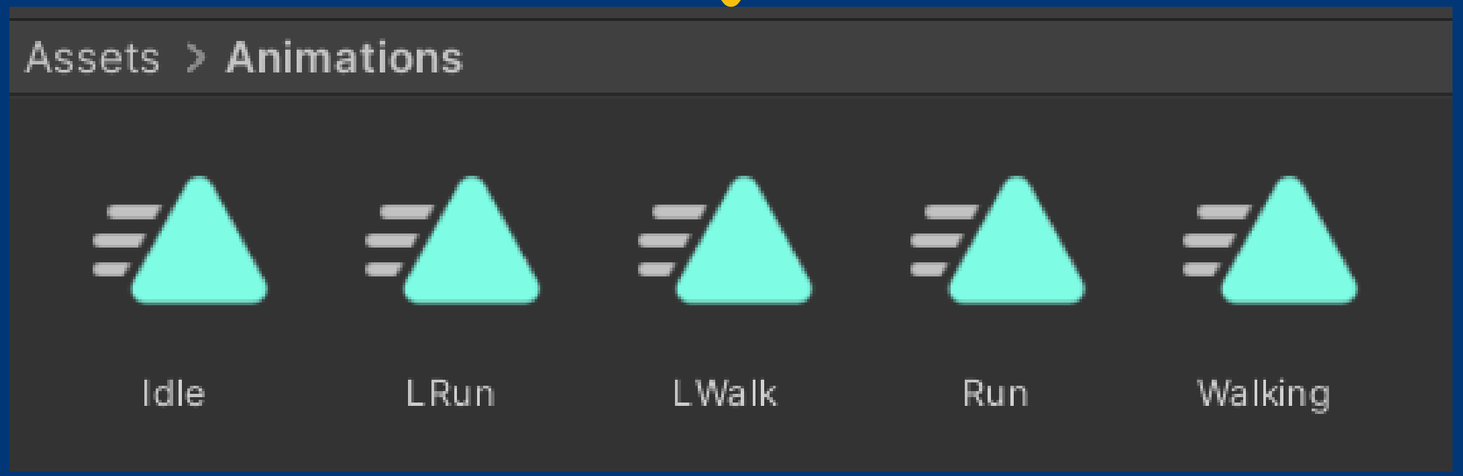
1

Seleccionamos una animación

Con Ctrl + D duplico la animación, le cambio el nombre y la muevo a la carpeta de animaciones

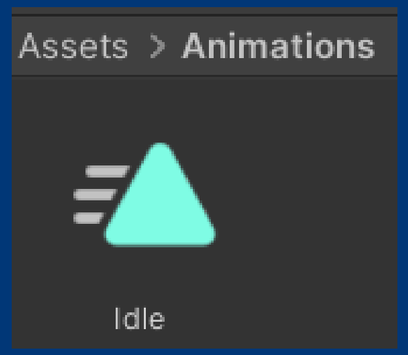


Lo realizo para todas las animaciones



1

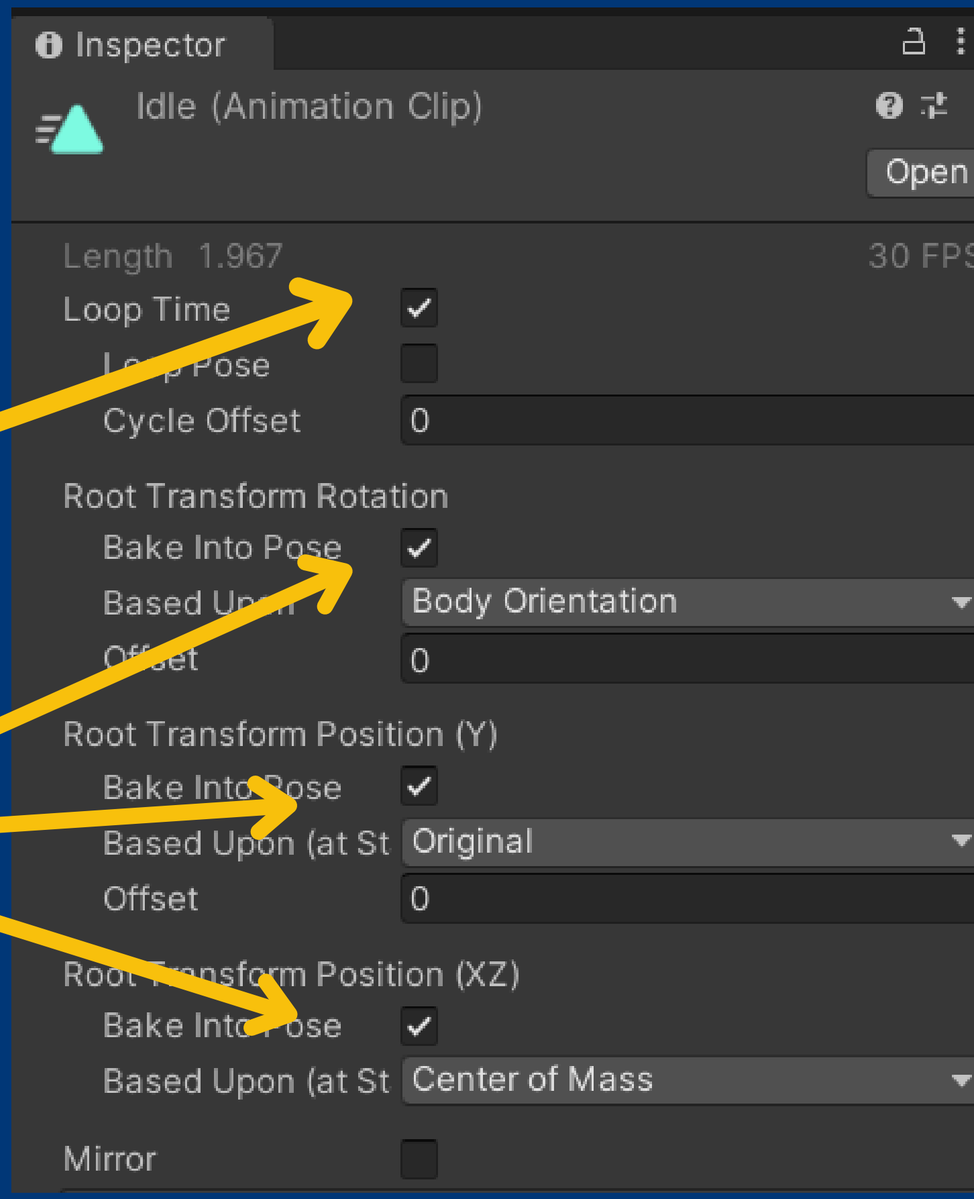
Seleccionamos una animación



En el inspector

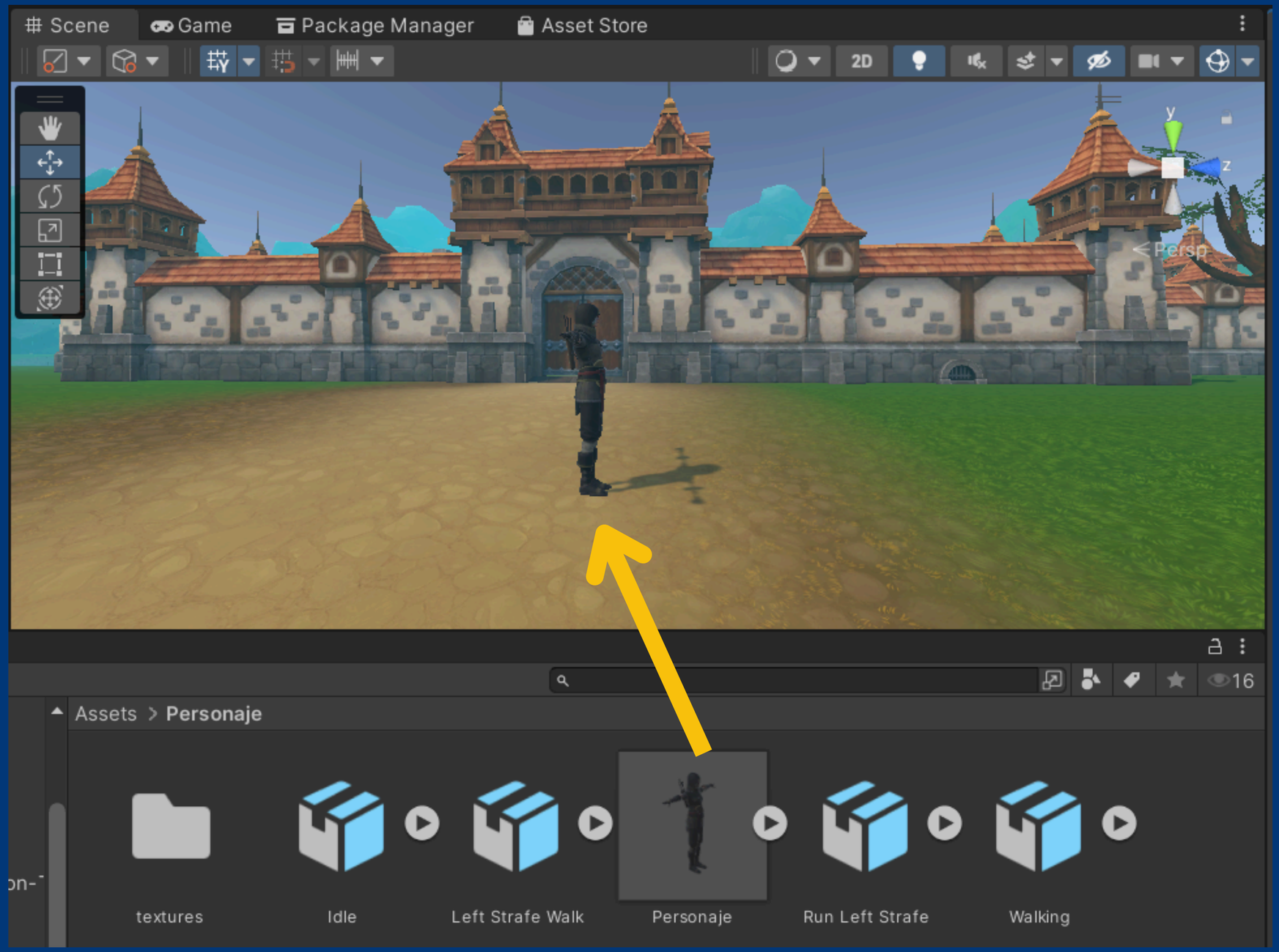
- **Ponemos en Loop Time**

Activamos el Bake Into Pose



1

Arrastramos el personaje a la escena

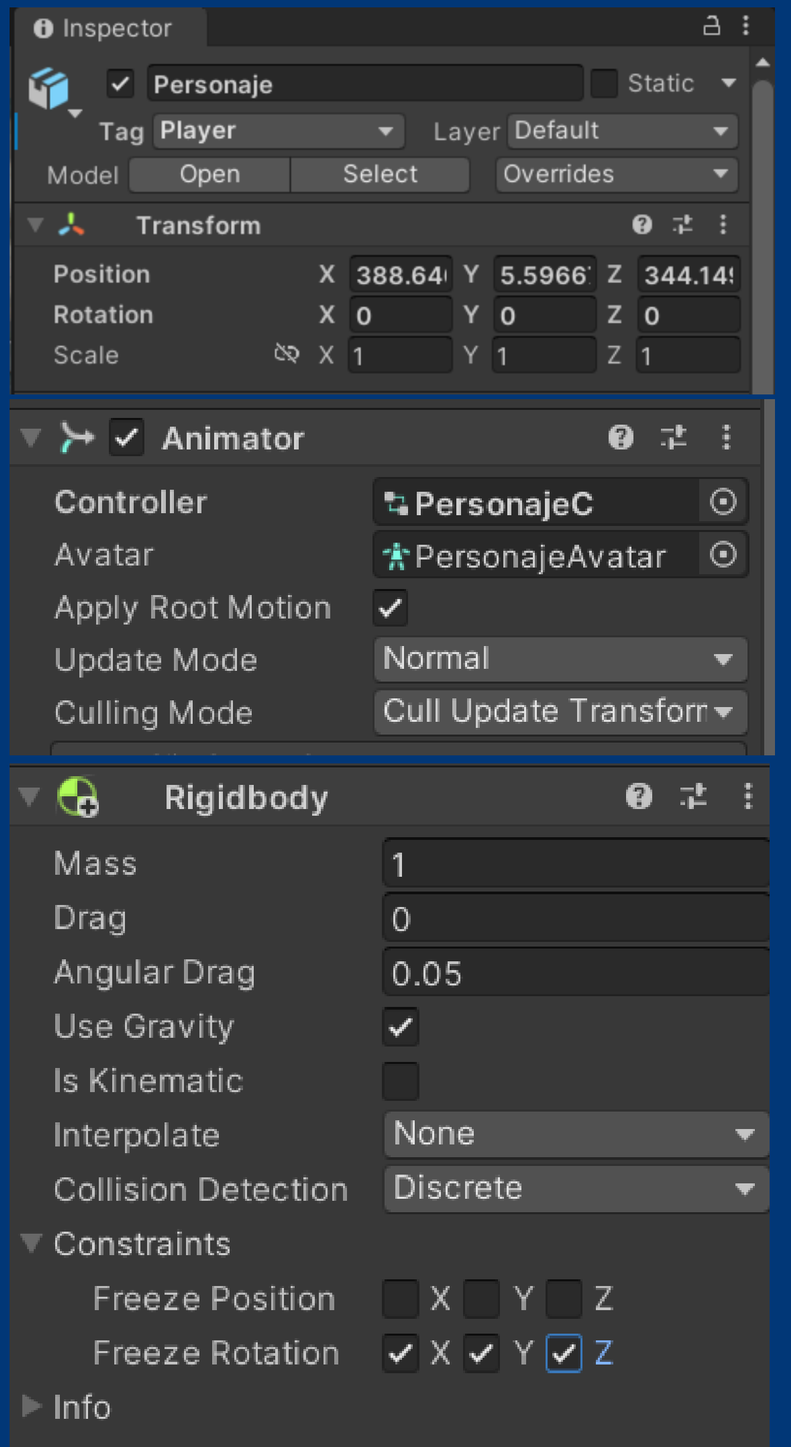


1

Cambiamos el Tag a Player

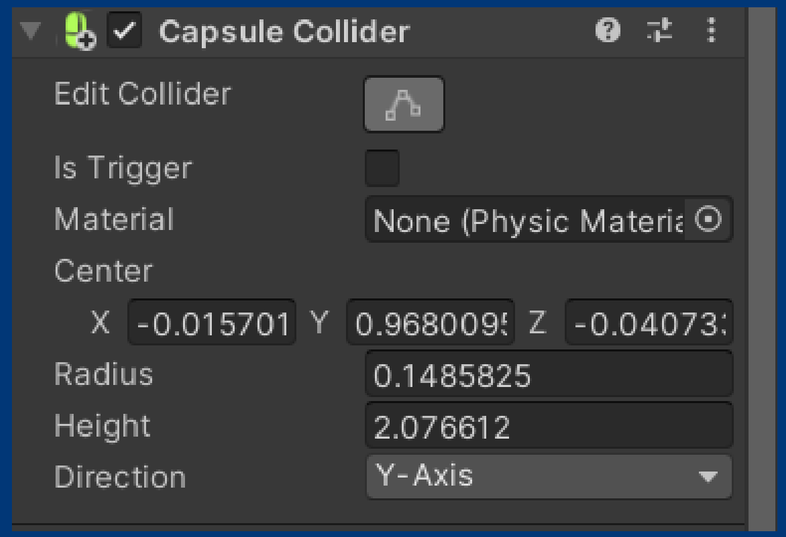
Creamos el Animator Controler y lo agregamos al Personaje

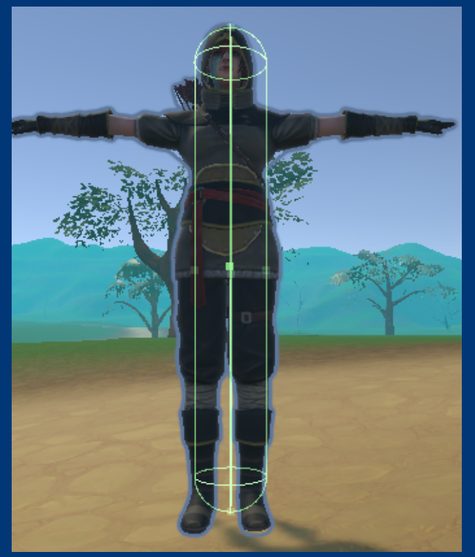
Agregamos un Rigidbody
Con las restricciones



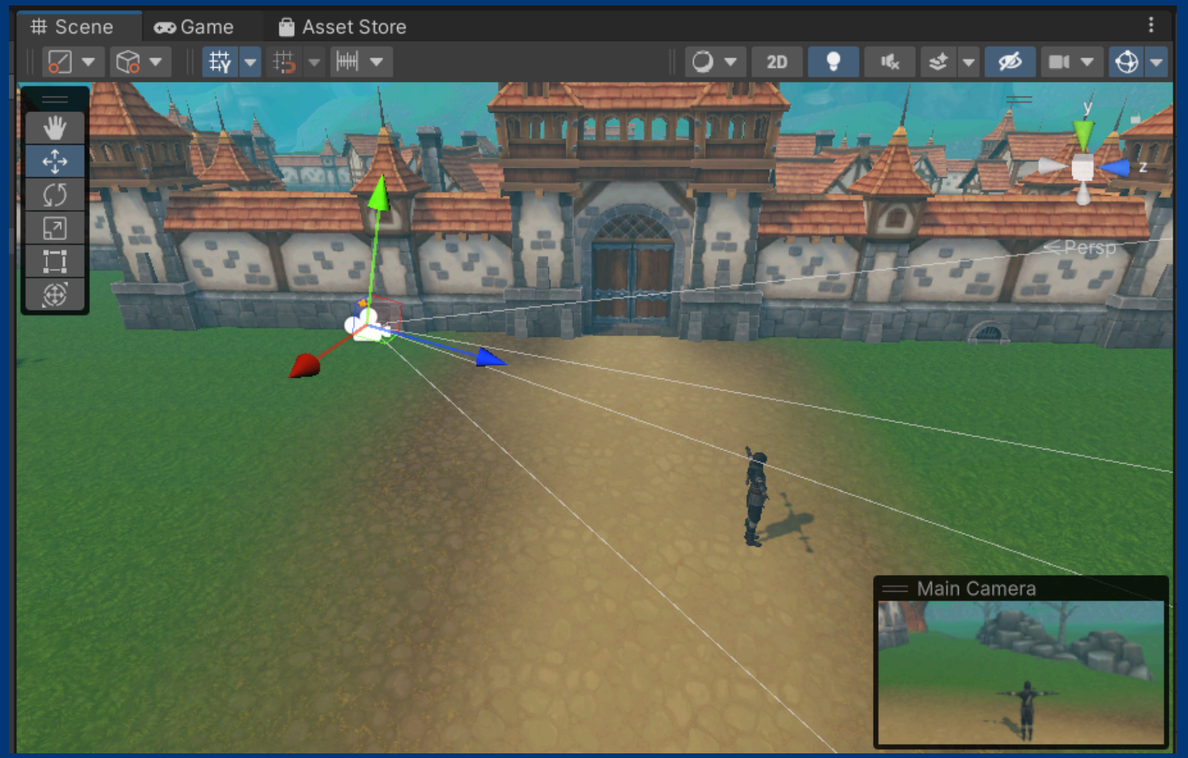
1

Agregamos un Capsule Collider adaptandolo al personaje

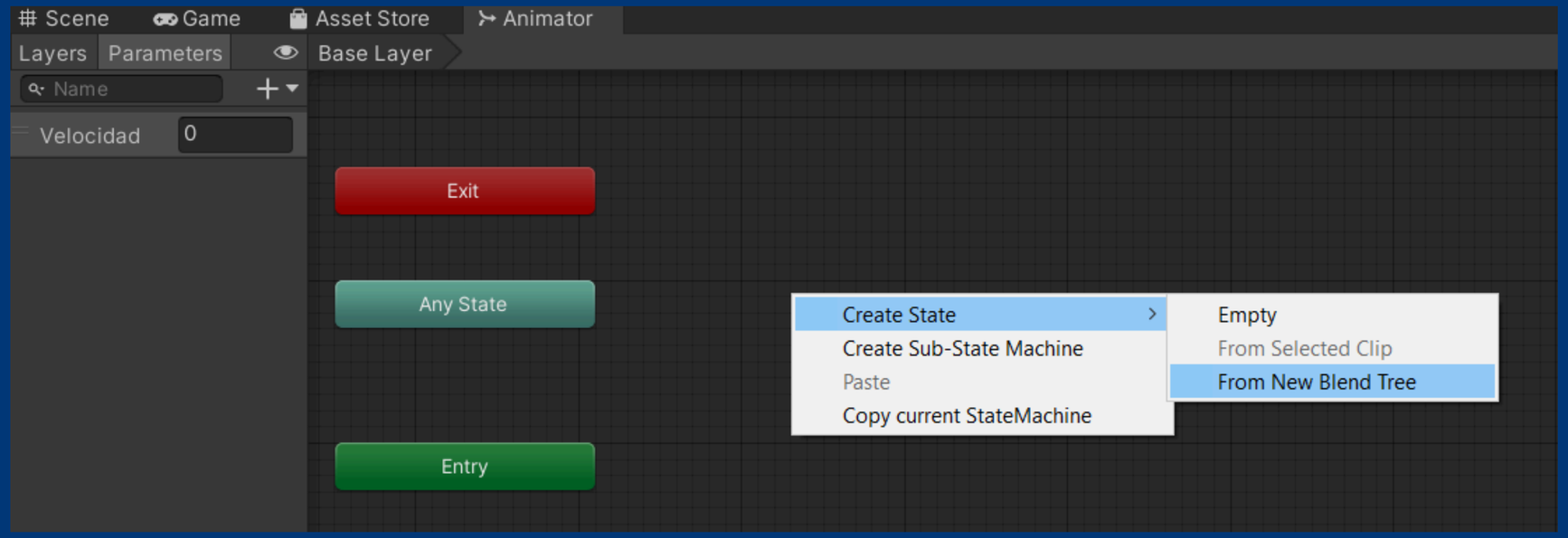




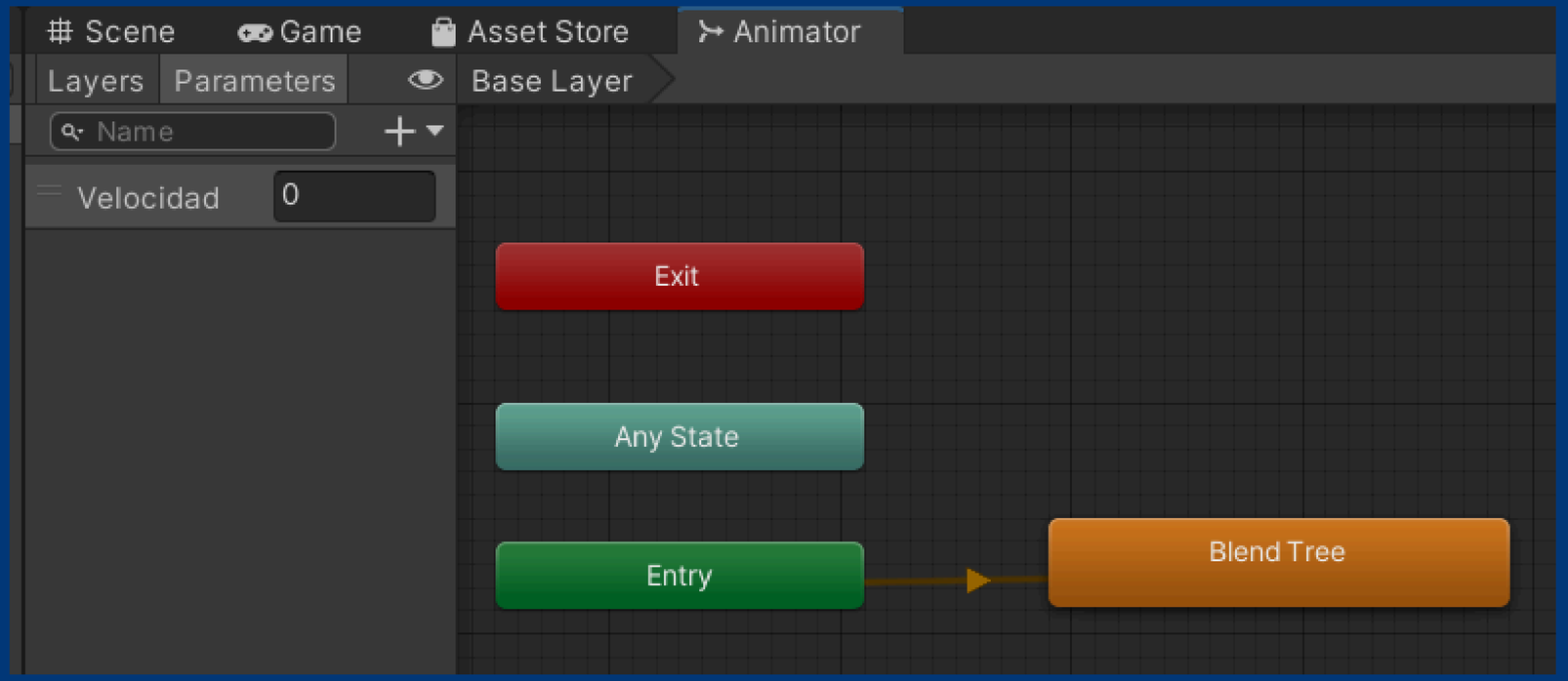
Posiciono la camara en un plano del jugador



1

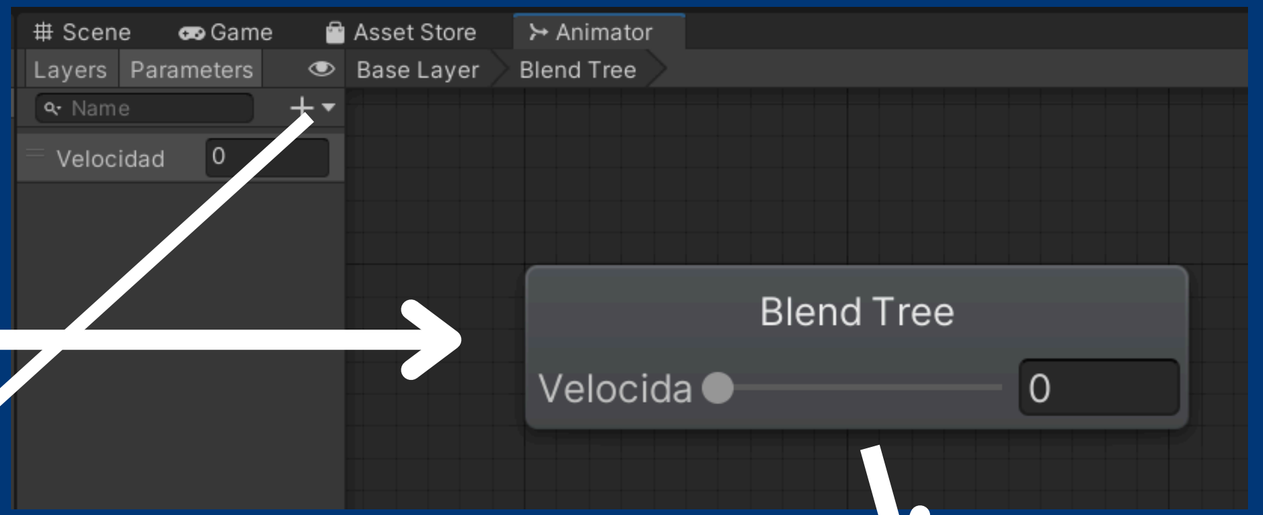


Creo un Blend Tree (Para mesclar entre dos o mas animaciones similares) Idle-Caminar-Correr

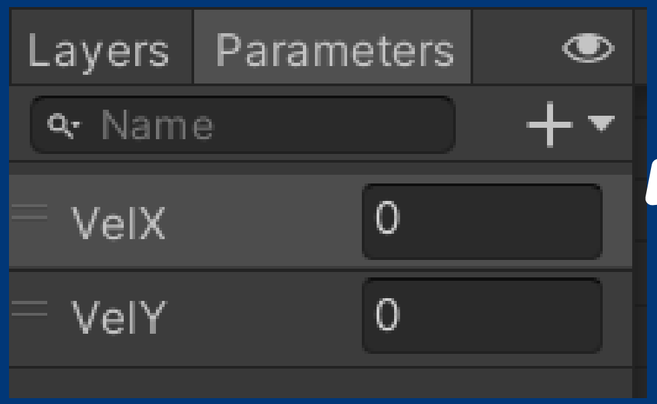


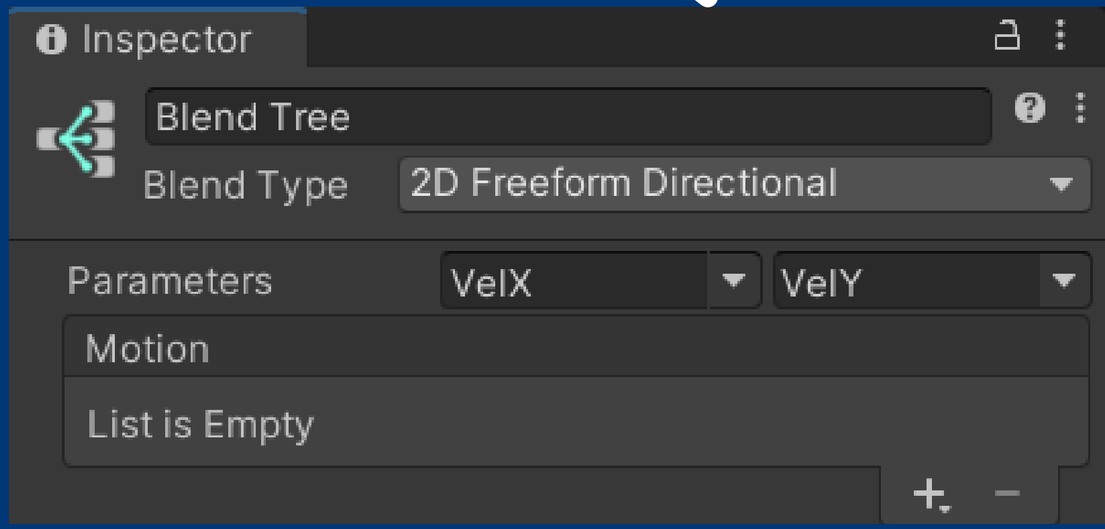
1

Doble Click



Creo dos parámetros tipo Float

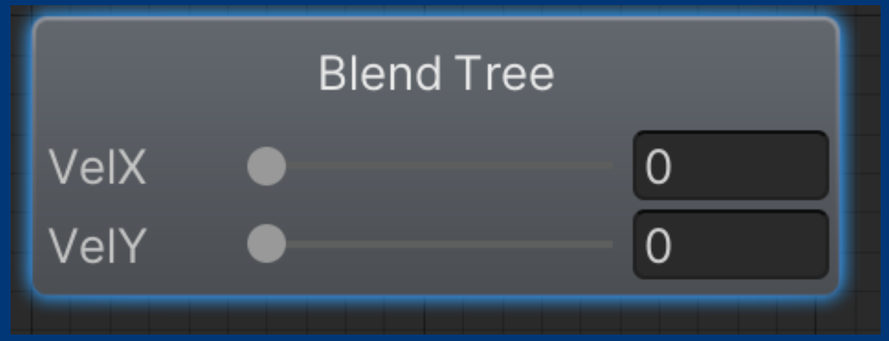




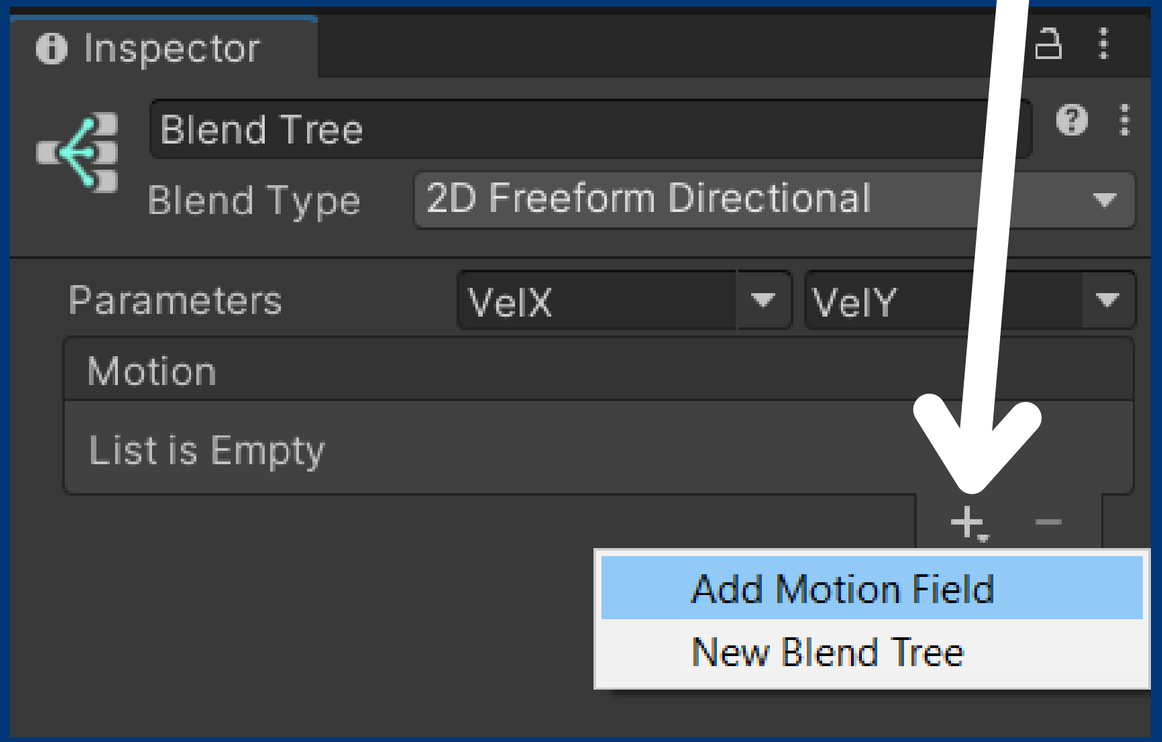
- **Seleccionando el BlendTree en el inspector cambio en el BlendType a 2D FreeformDirectiona**
- **En parámetros selecciono los creados VelX y VelY**

1

Vamos a crear las transiciones de las animaciones por la velocidad (valores de las flechas) que se tenga



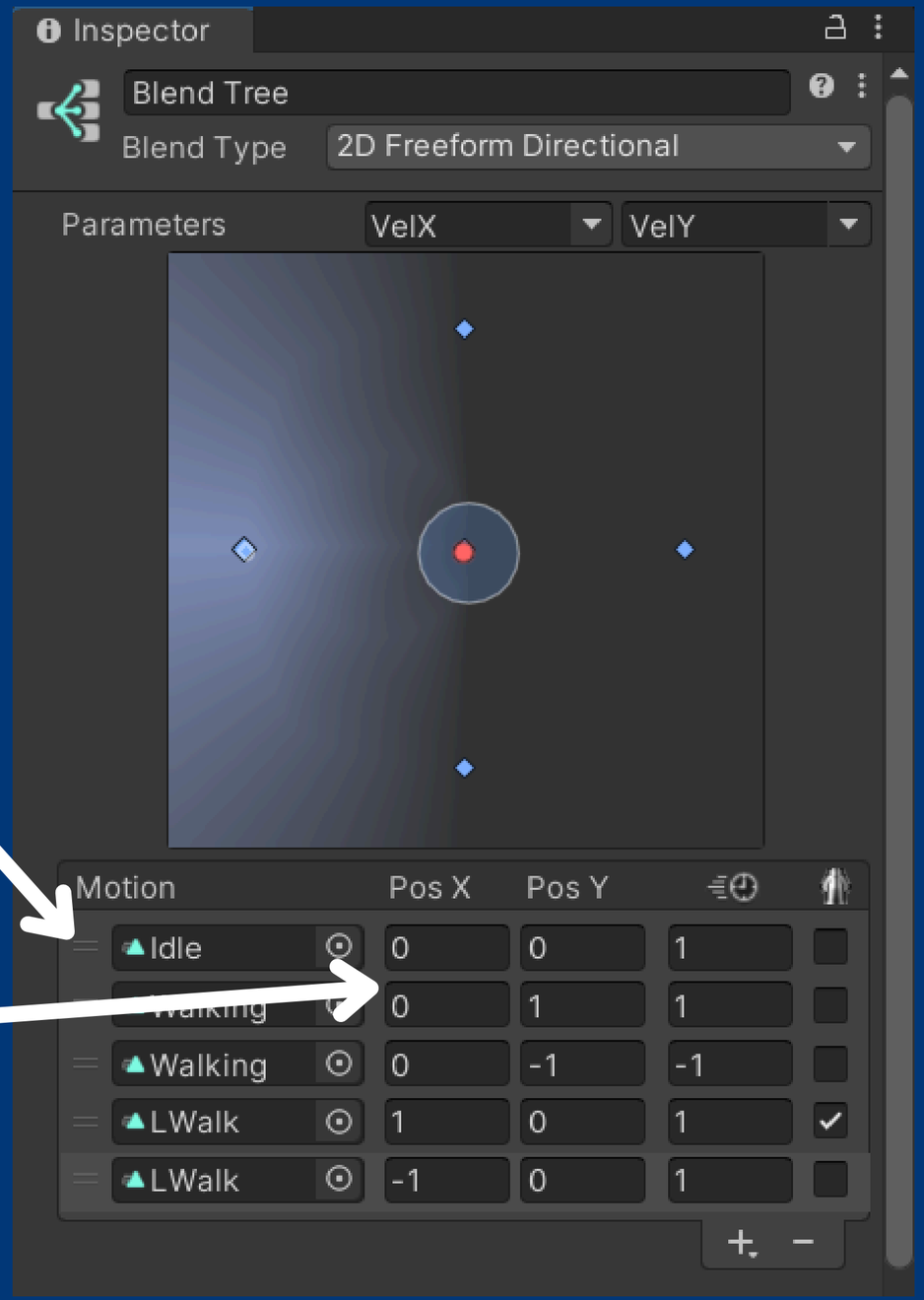
Creamos 5 Motion Field



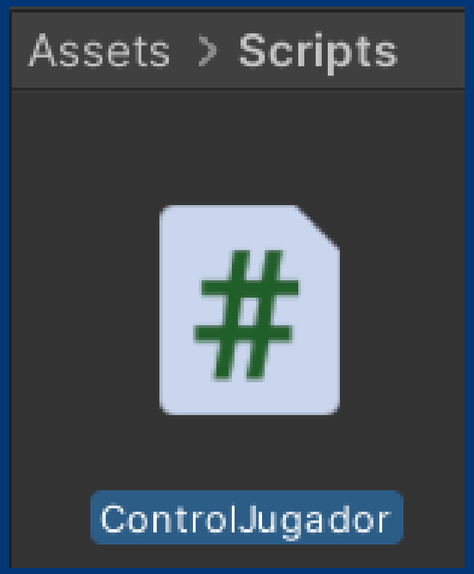
1

Agrego la animaciones

X Y representa las flechas del teclado

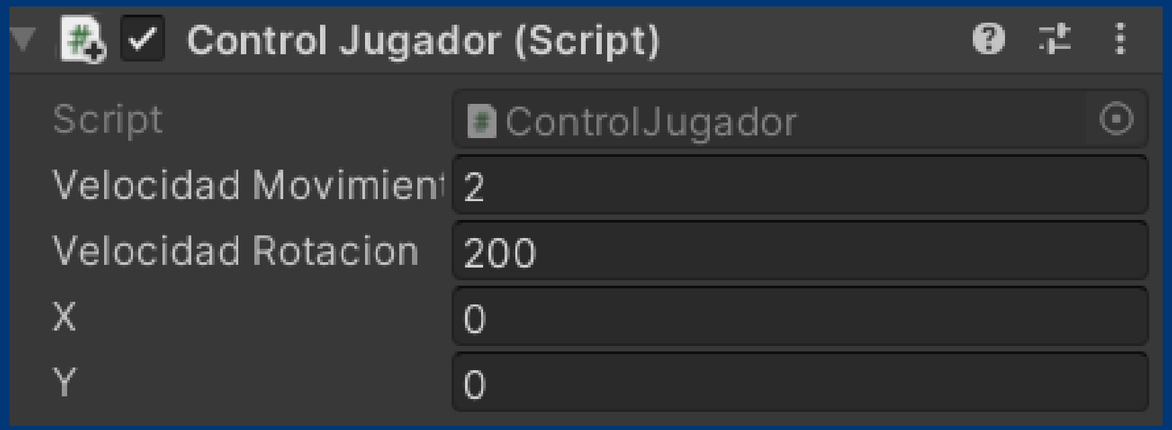


1



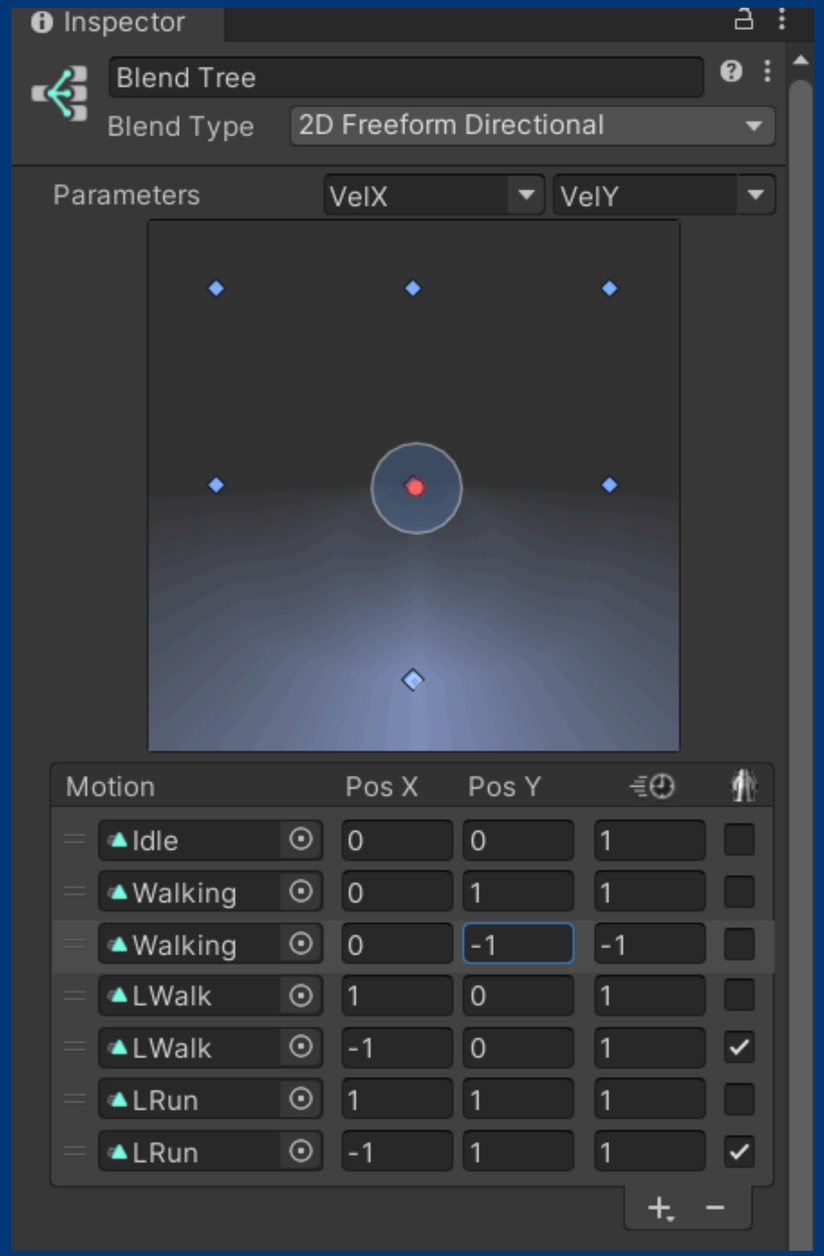
Script #1 ControlJugador.cs

Control básico del personaje y animaciones



1

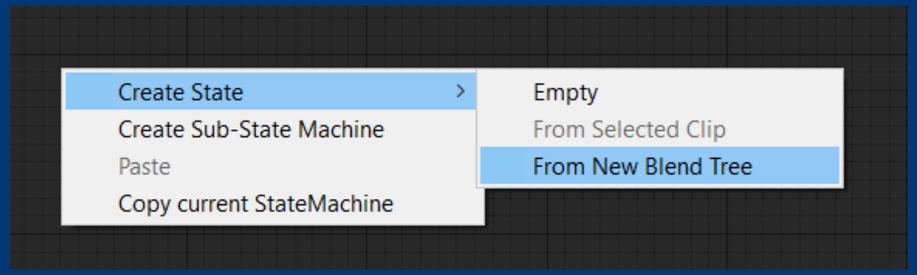
Probamos el los lados

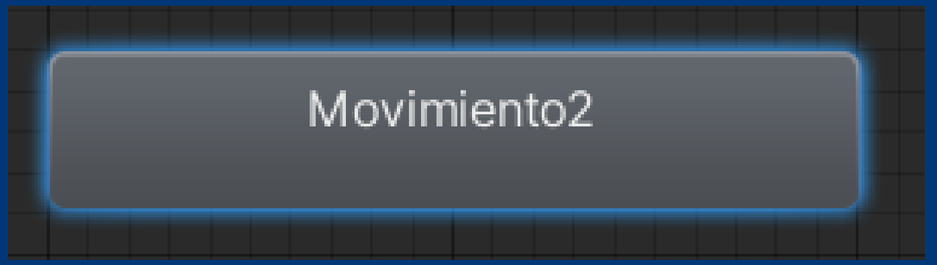


1

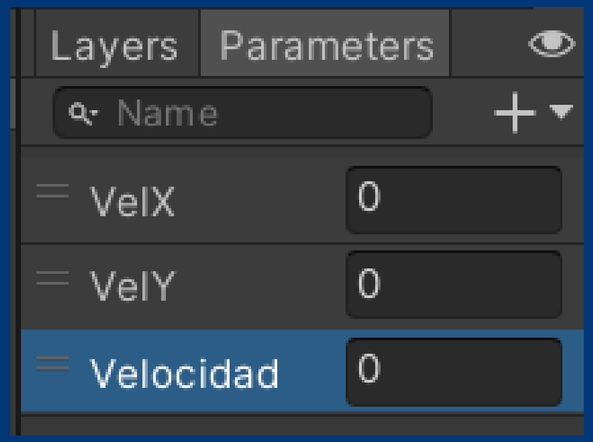
Ejemplo 2

Creamos otro Blend Tree y renombramos Movimientos 2





Creamos otro un nuevo parametro tipo fload de nombre Velocidad

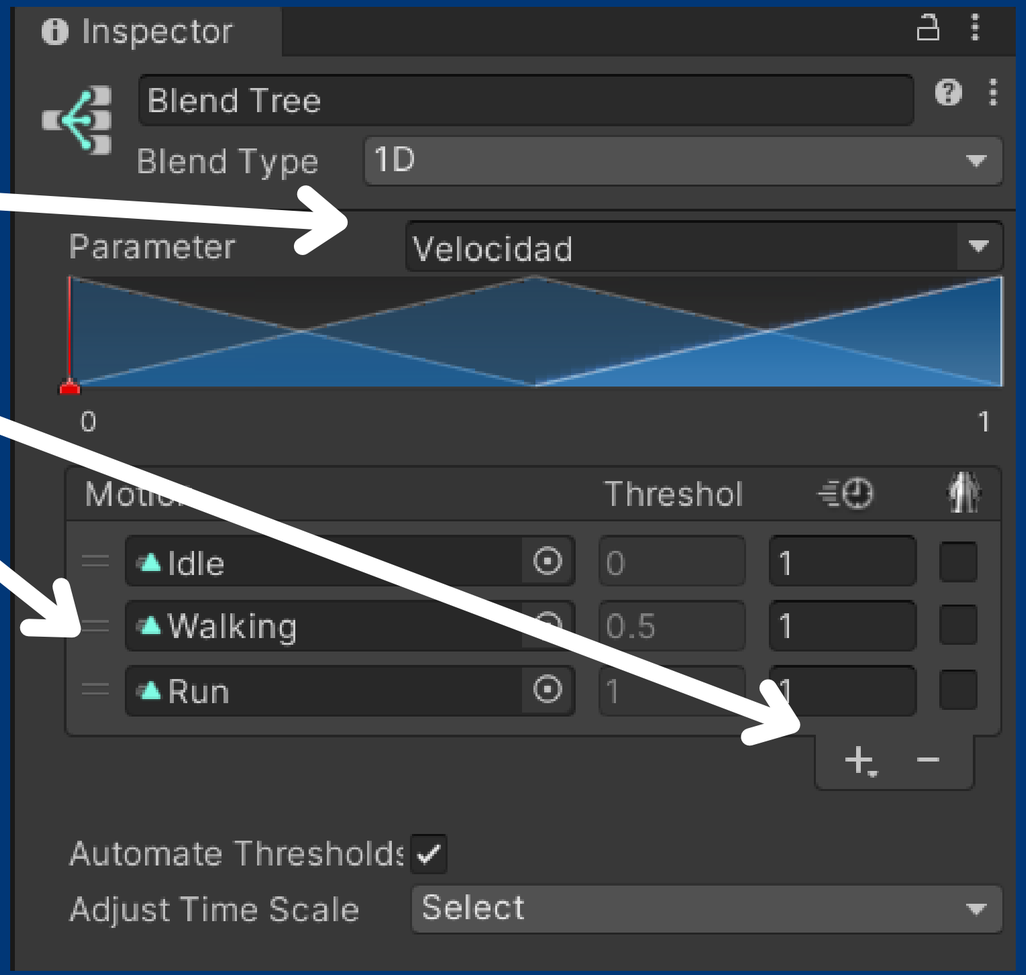


1

En el inspector del Blend Tree

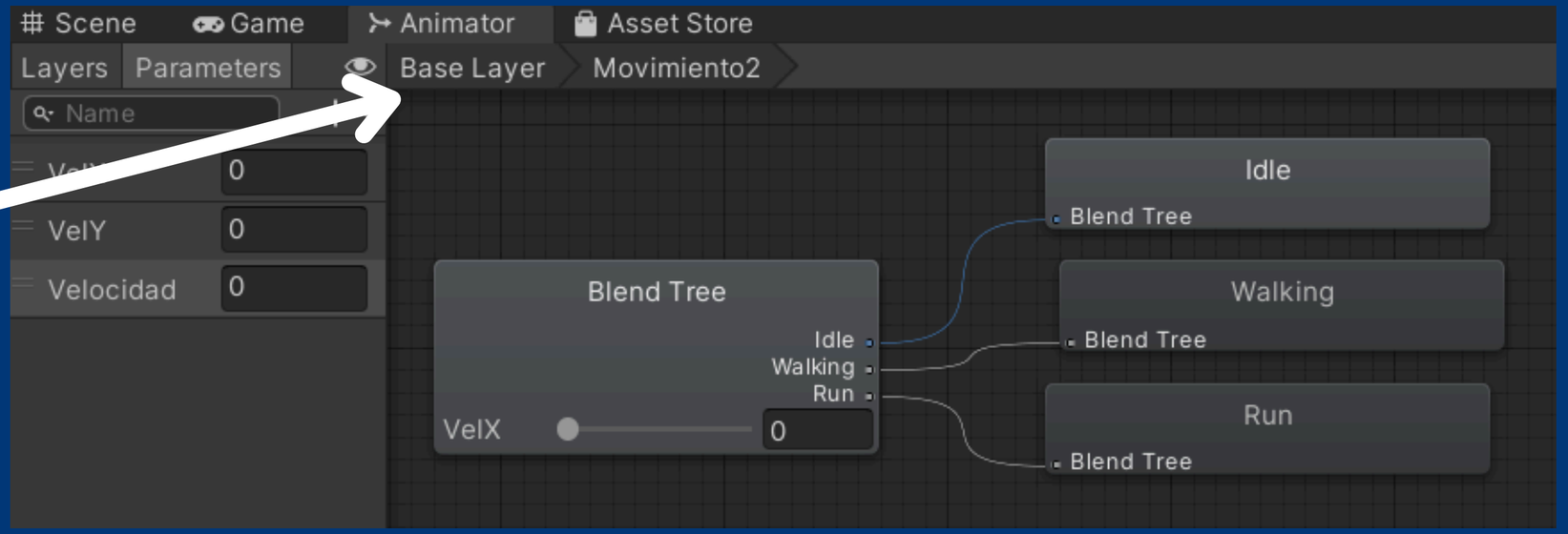
- **En Parametro selecciona Velocidad**
- **Creamos 3 Add Motion Field**

- **Arrastramos las 3 animaciones**
  - **Depende la velocidad con la que estemos pasara de Idle, Walking y Run (Considerando que para run Debe estar presionada la tecla Shift)**

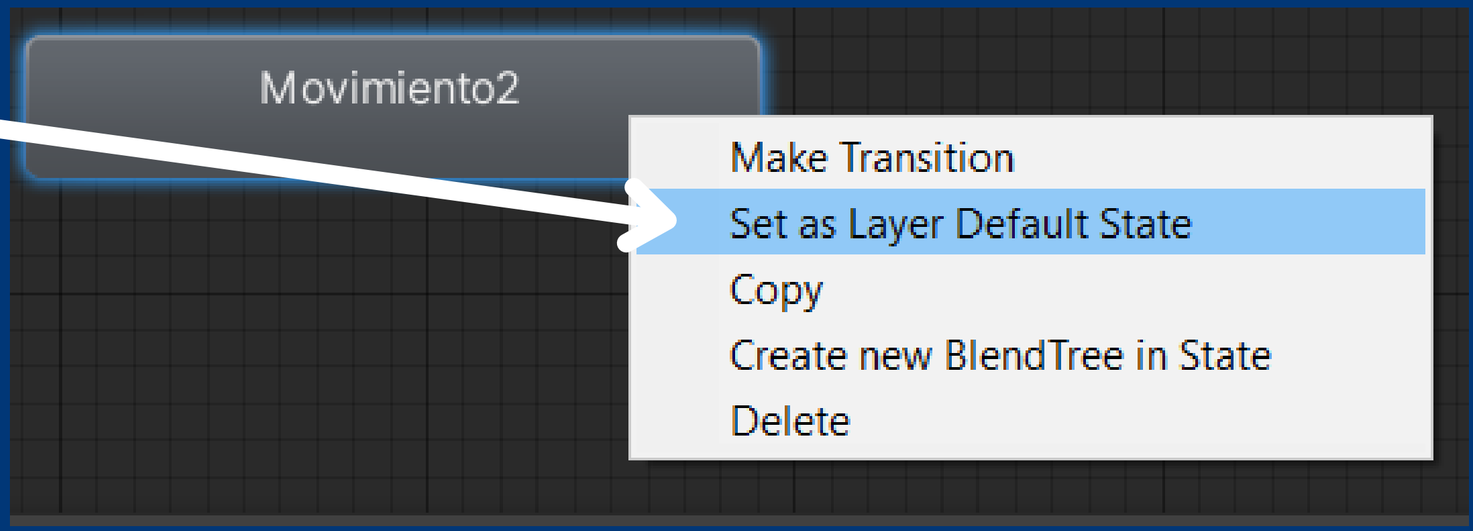


1

Regresamos a la Base Layer



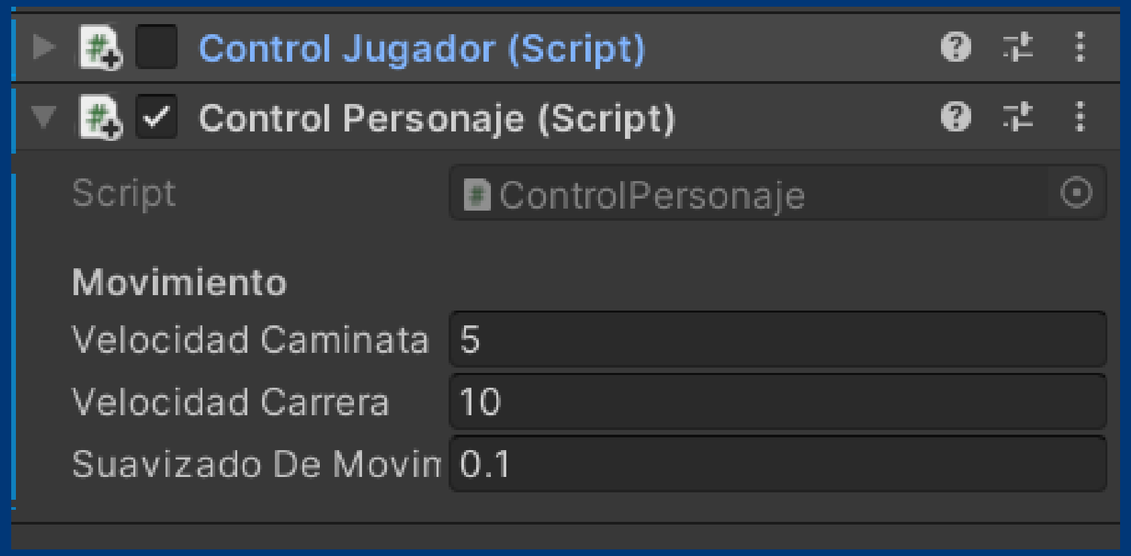
Click derecho en el estado y seleccionamos



1

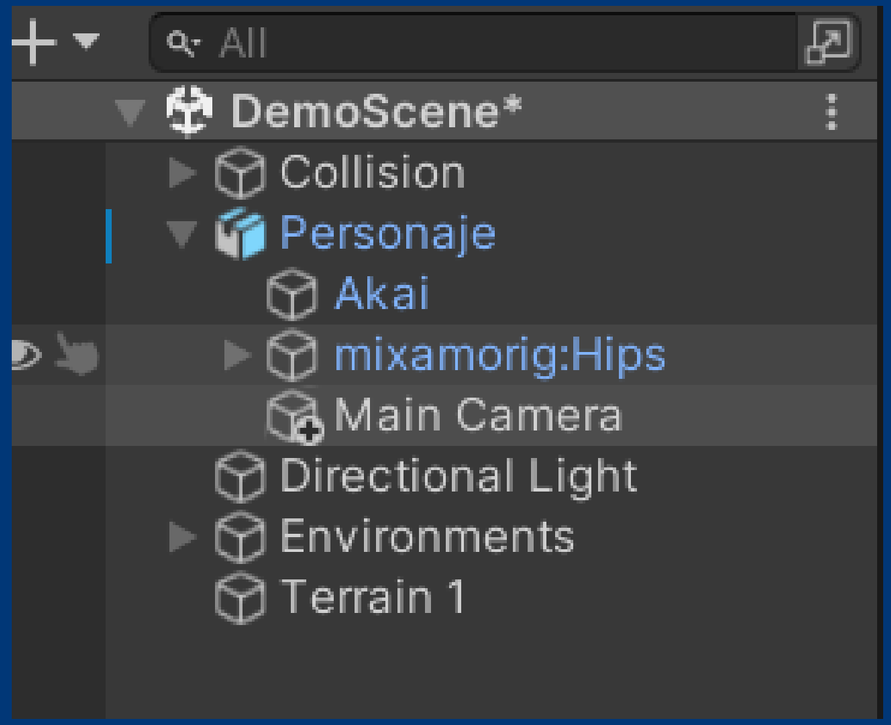
Script #2
ControlPersonaje.cs

Control básico del personaje y animaciones

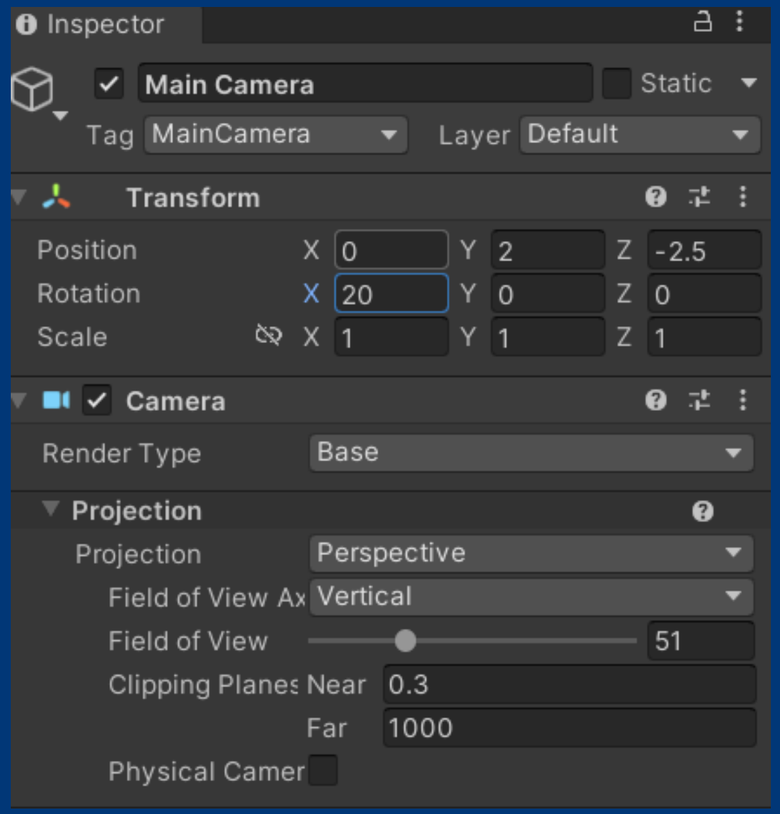


1 Control de Camara

Podemos poner la cámara dentro del personaje



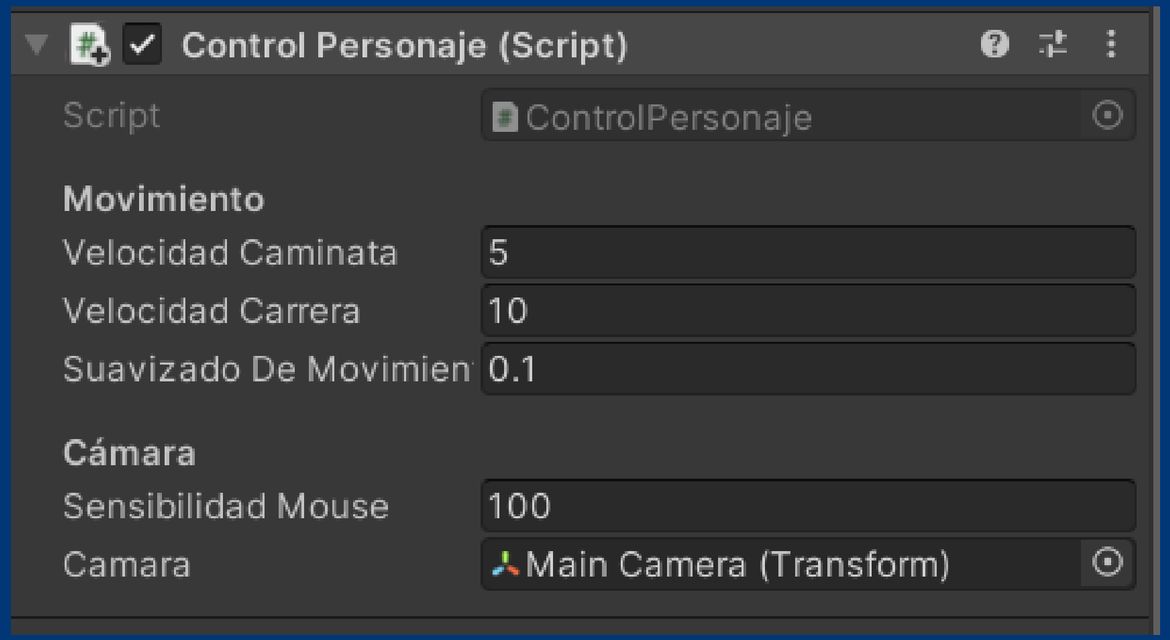
Posicionando en un plano al jugador



1

Actualizar Script #3 ControlPersonaje.cs

Control básico del personaje y movimiento de la camara con el raton



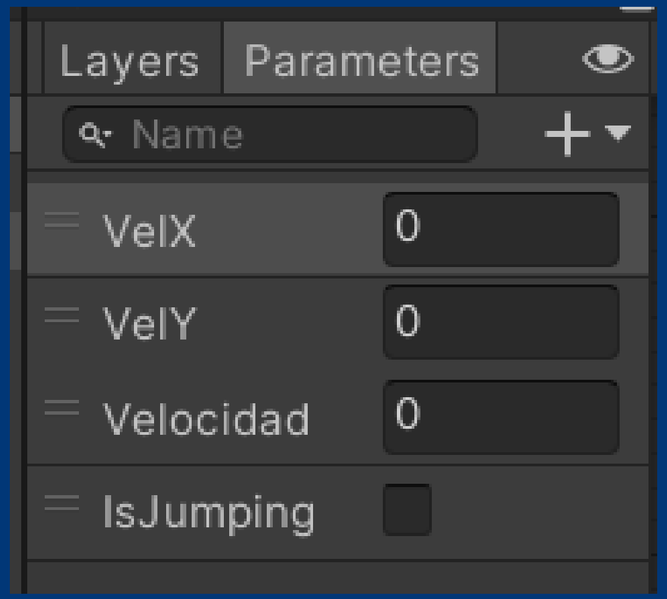
2 Acciones y animaciones adicionales

2

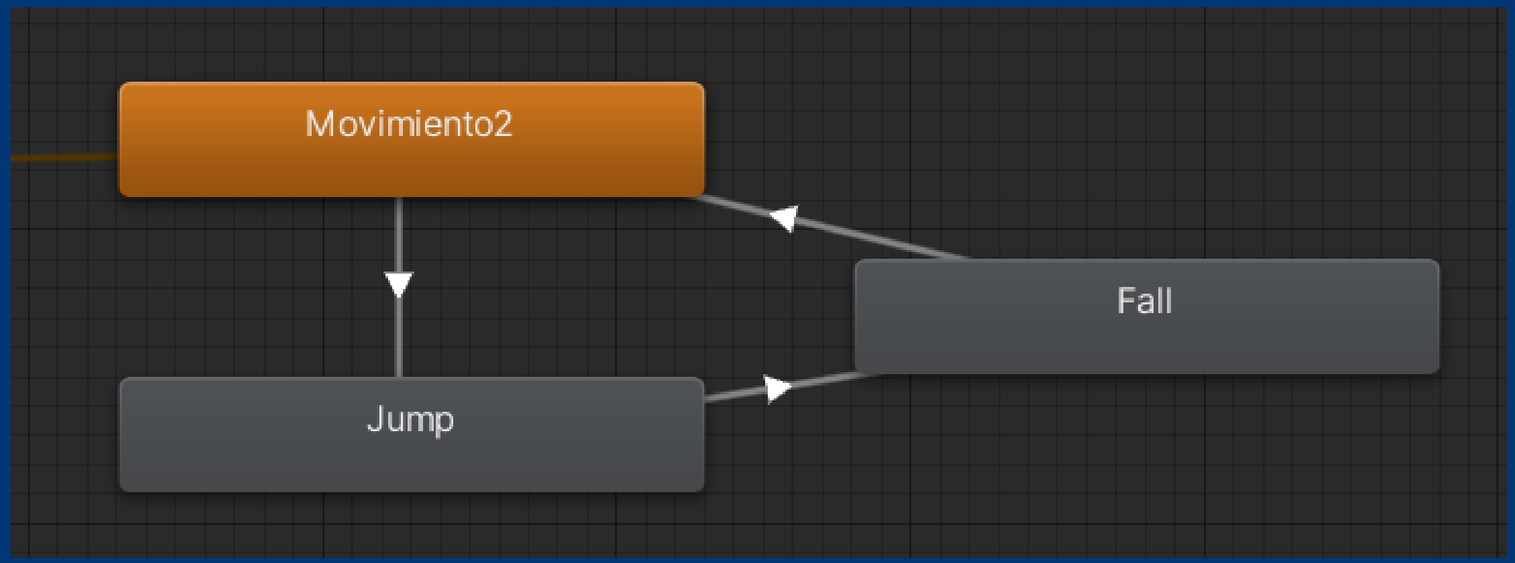
Salto

Preparo las animaciones

Creo la Variable de tipo bool IsJumping

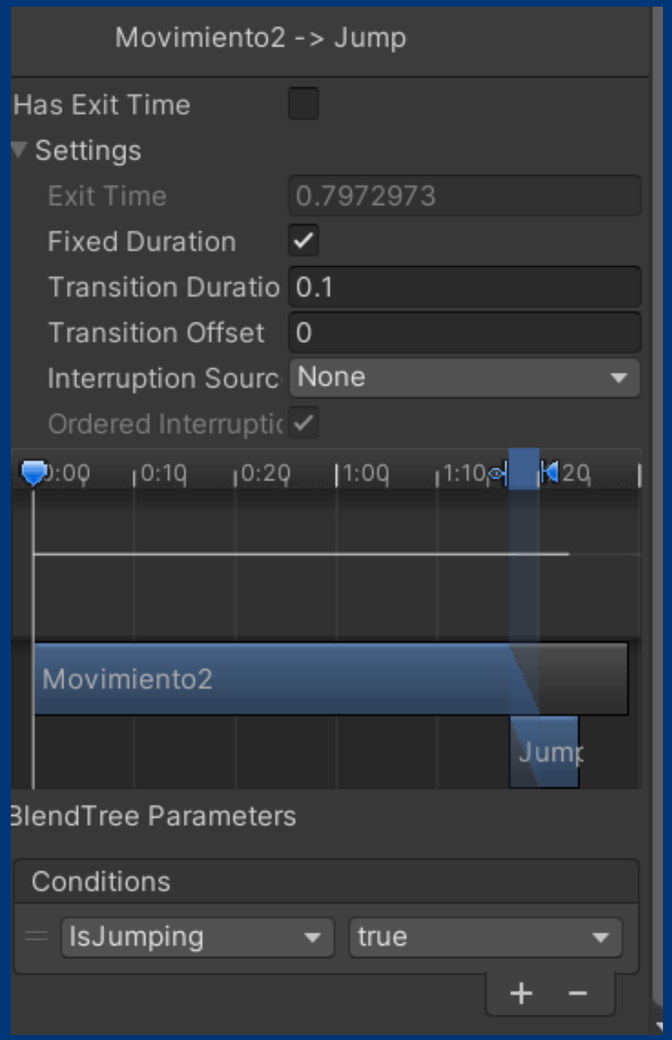


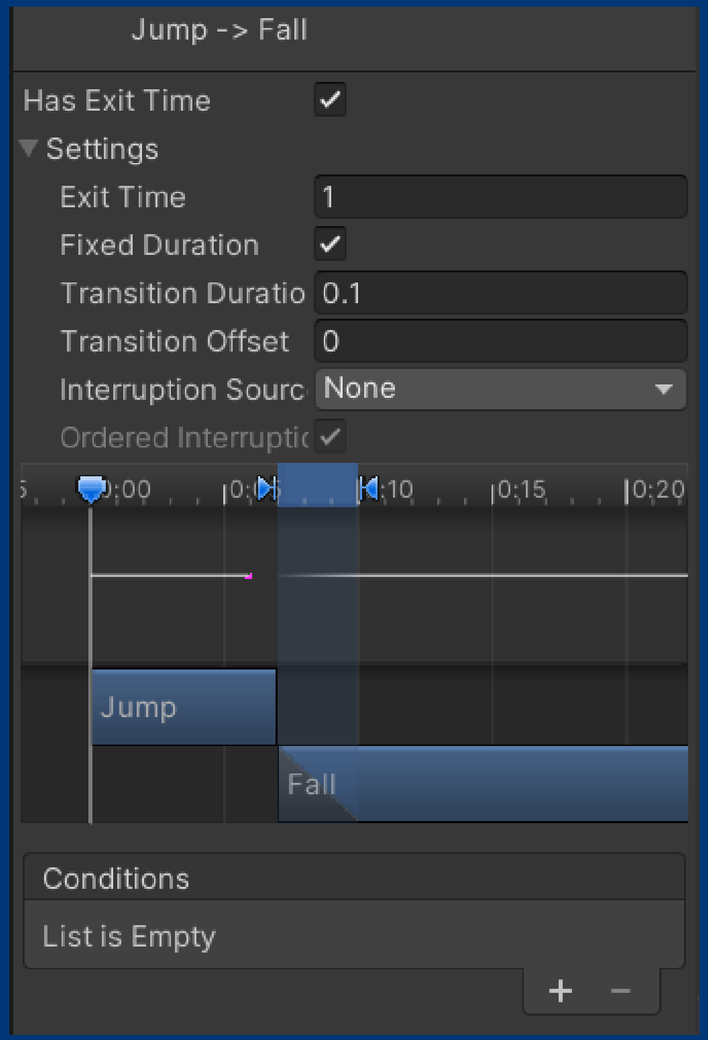
Transiciones hacia Movimiento2

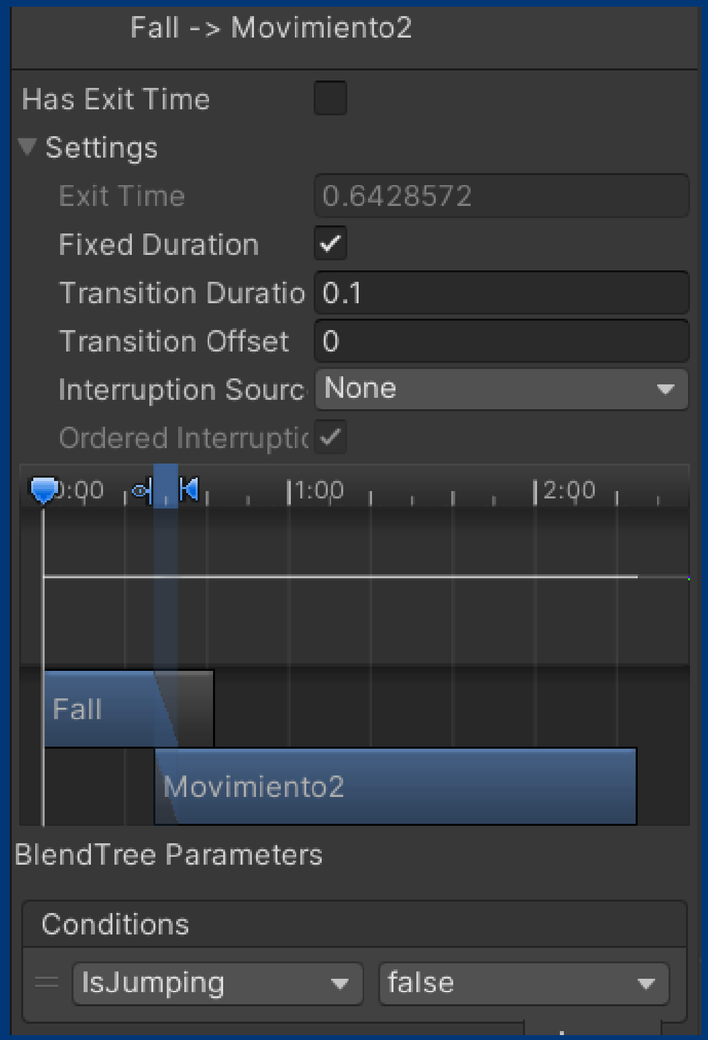


2

Transiciones

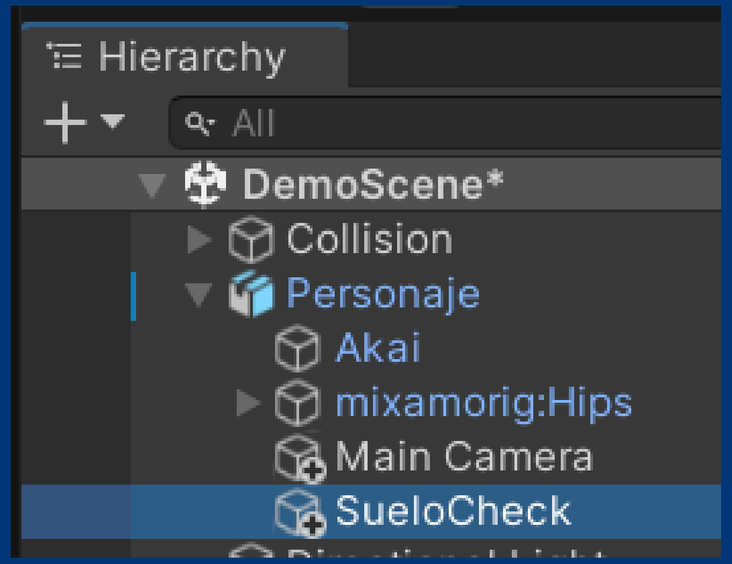




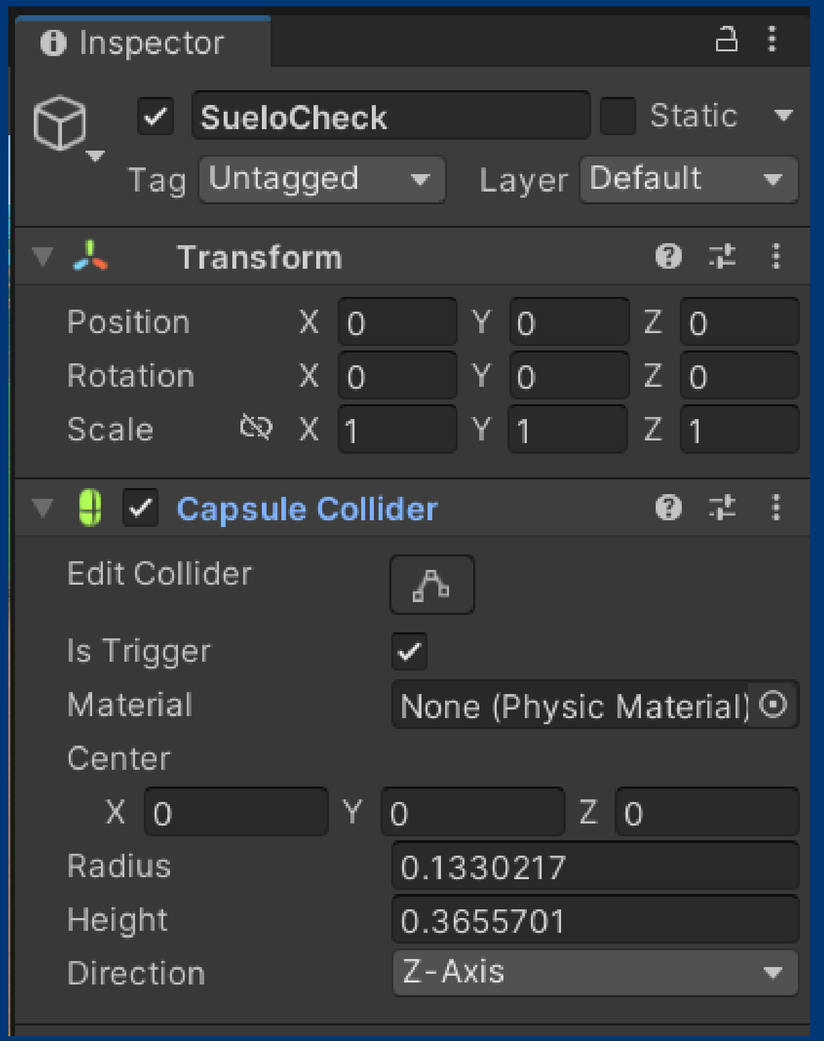


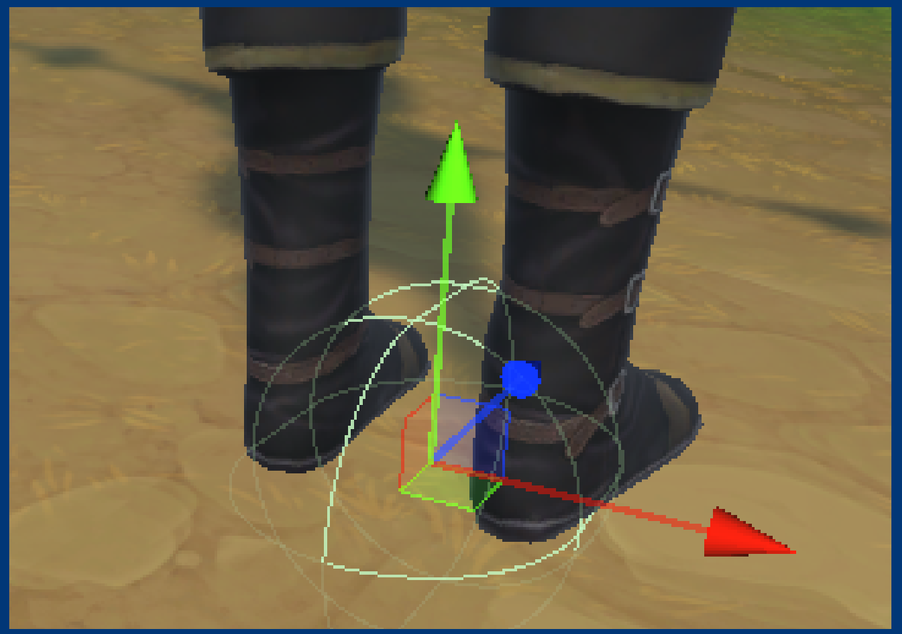
2

Agrego un objeto vacío



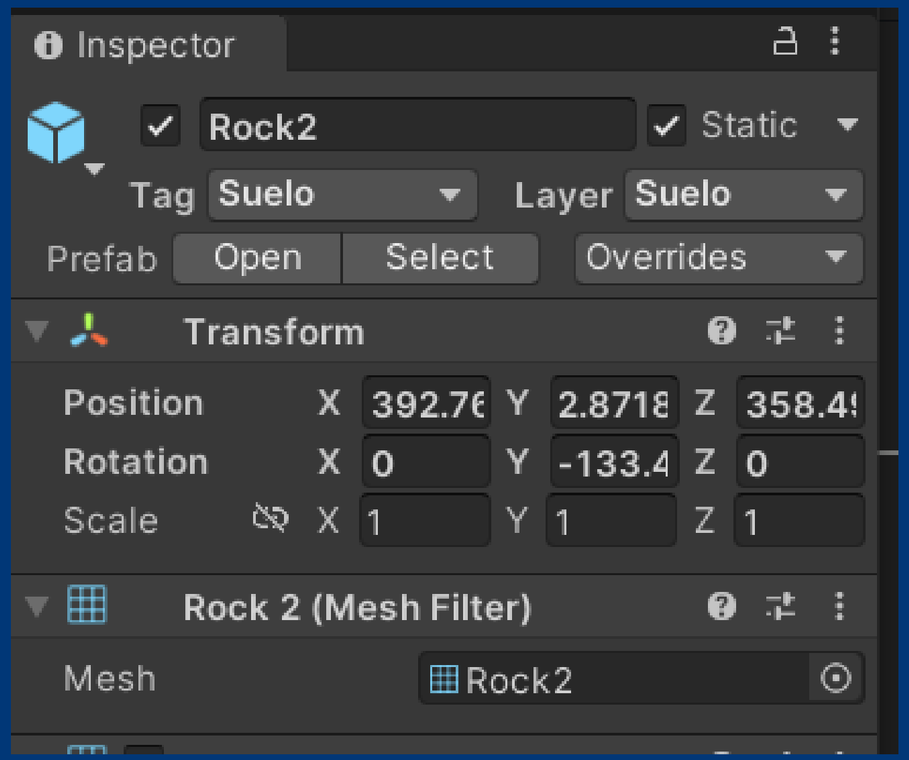
Agrego un Colisionador en trigger

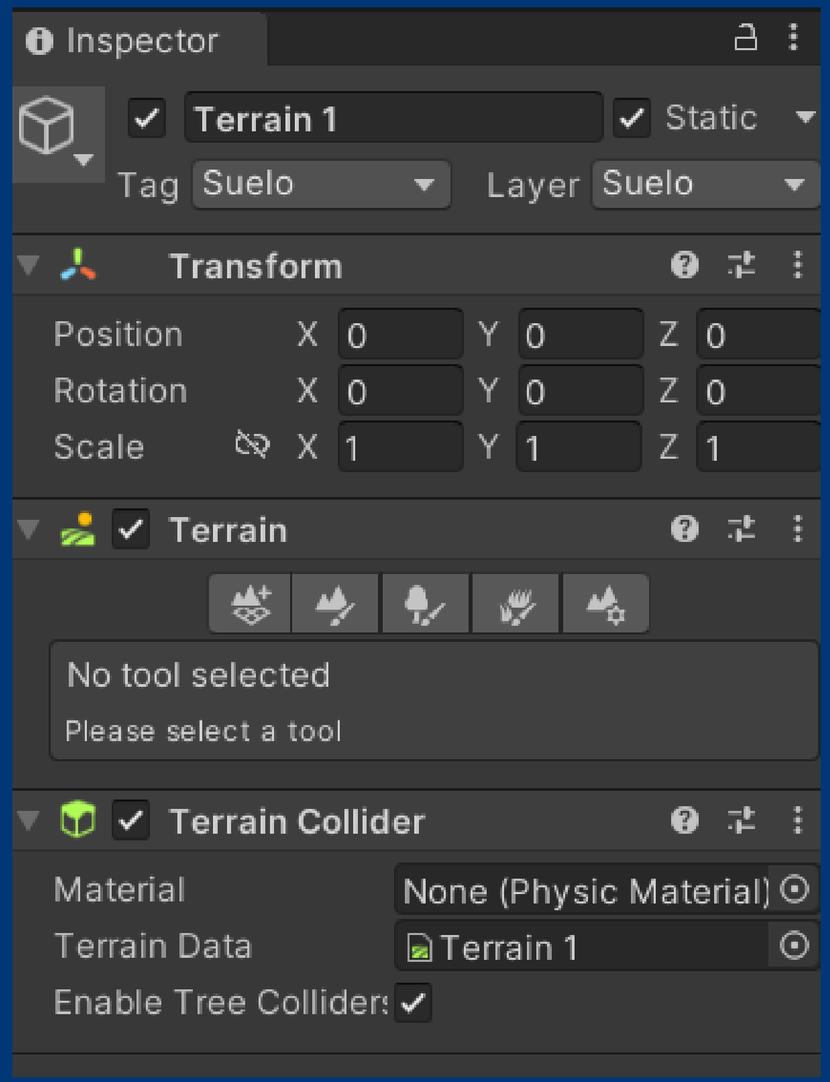




2

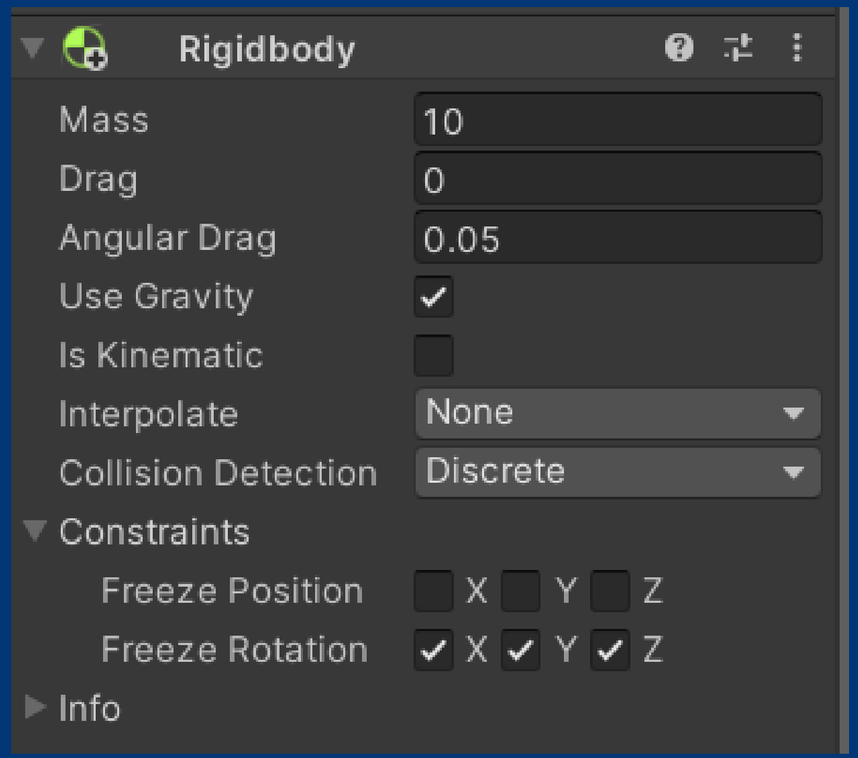
Crear Layer de nombre Suelo para el terreno y objetos de suelo



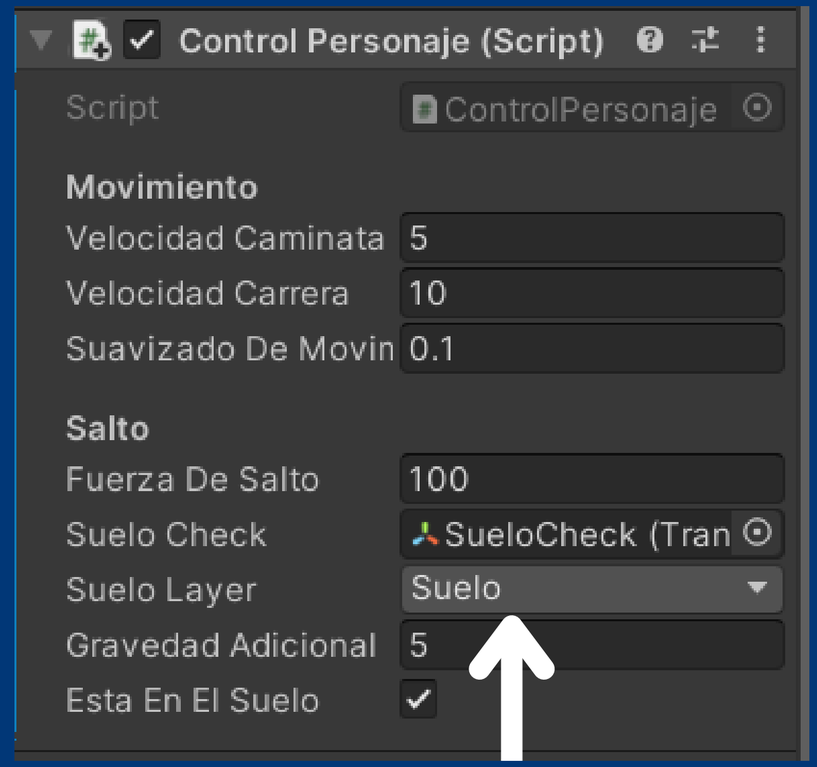


2

Pruebo configuraciones de Masa del personaje



Actualizar Script #4 ControlPersonaje.cs

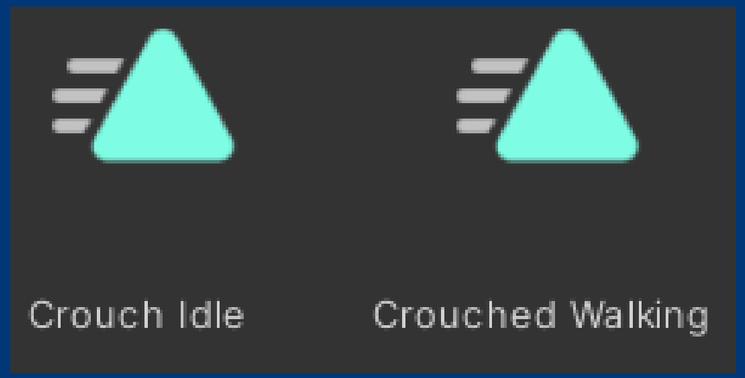


Agrego el objeto Suelo check y el Layer

2

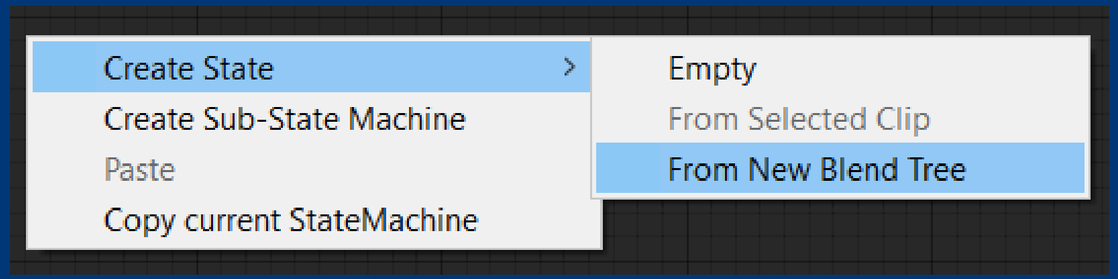
Agacharse

Preparo las animaciones

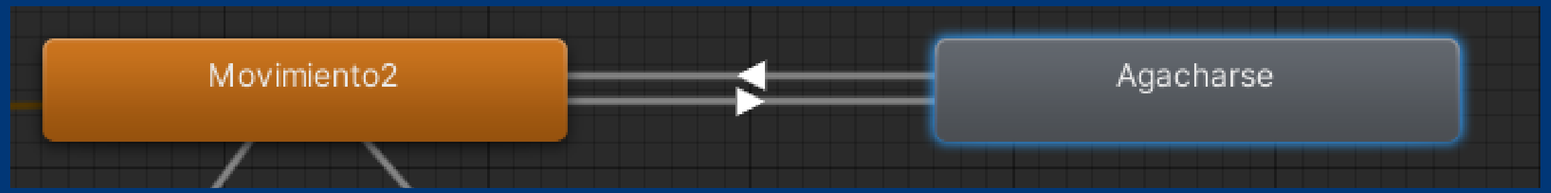


Creo un parametro IsCroucg tipo Bool

Creo un Blend tree



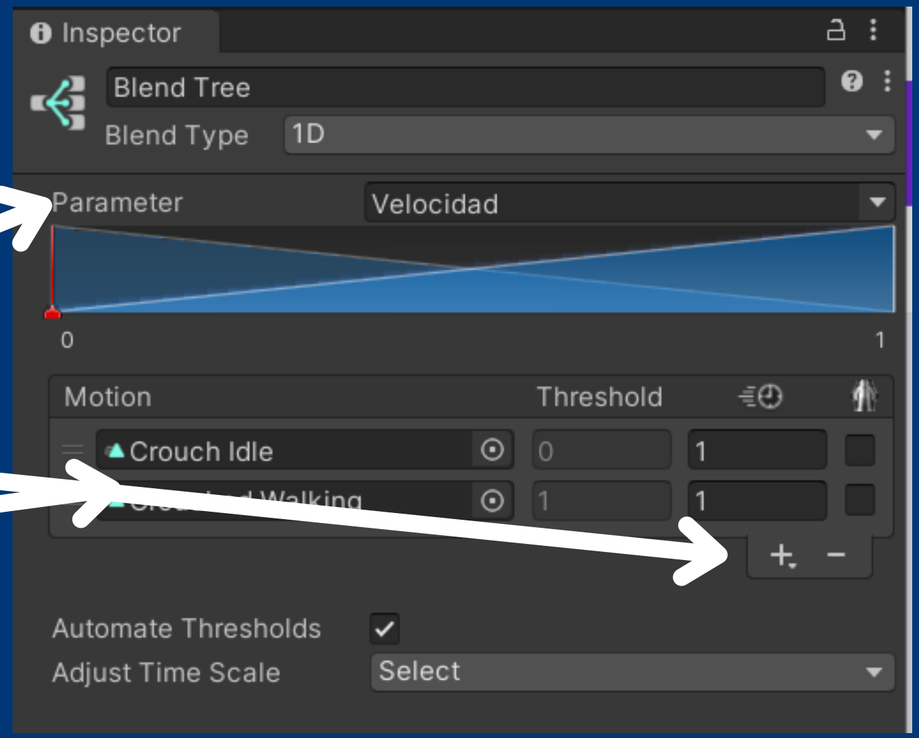
Realizo las transiciones hacia movimiento2

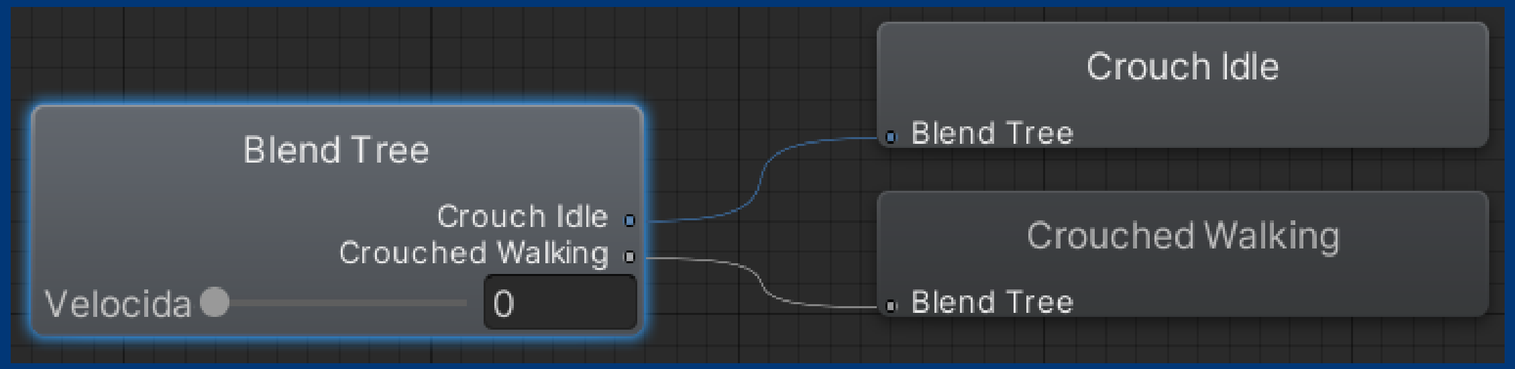


2

Seleccionando el Blend Tree

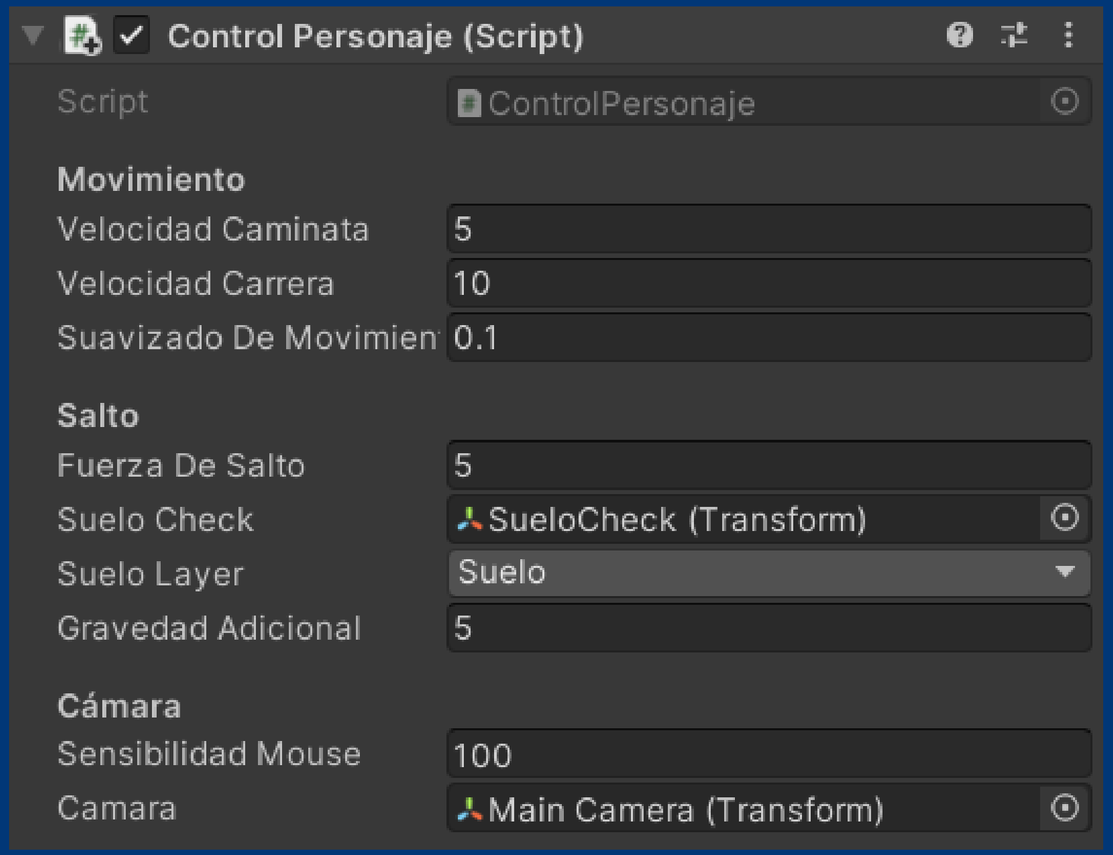
- En el inspector cambio el parametro a Velocidad
- Creo dos Motion
- Agrego sus animaciones





2

Actualizar Script #5 ControlPersonaje.cs



El script controla el agacharse con la tecla V

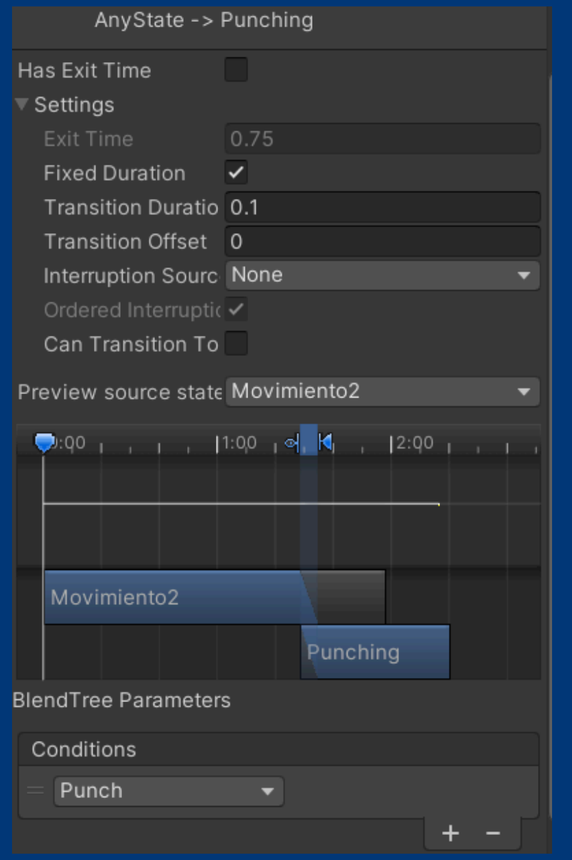
2

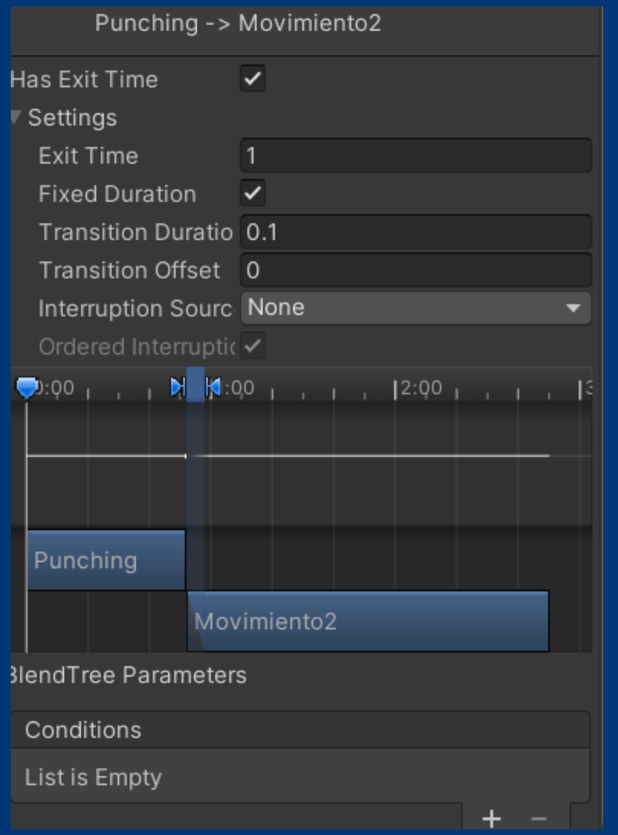
Golpe

Creo la Variable de tipo trigger Punch

Realizo las transiciones

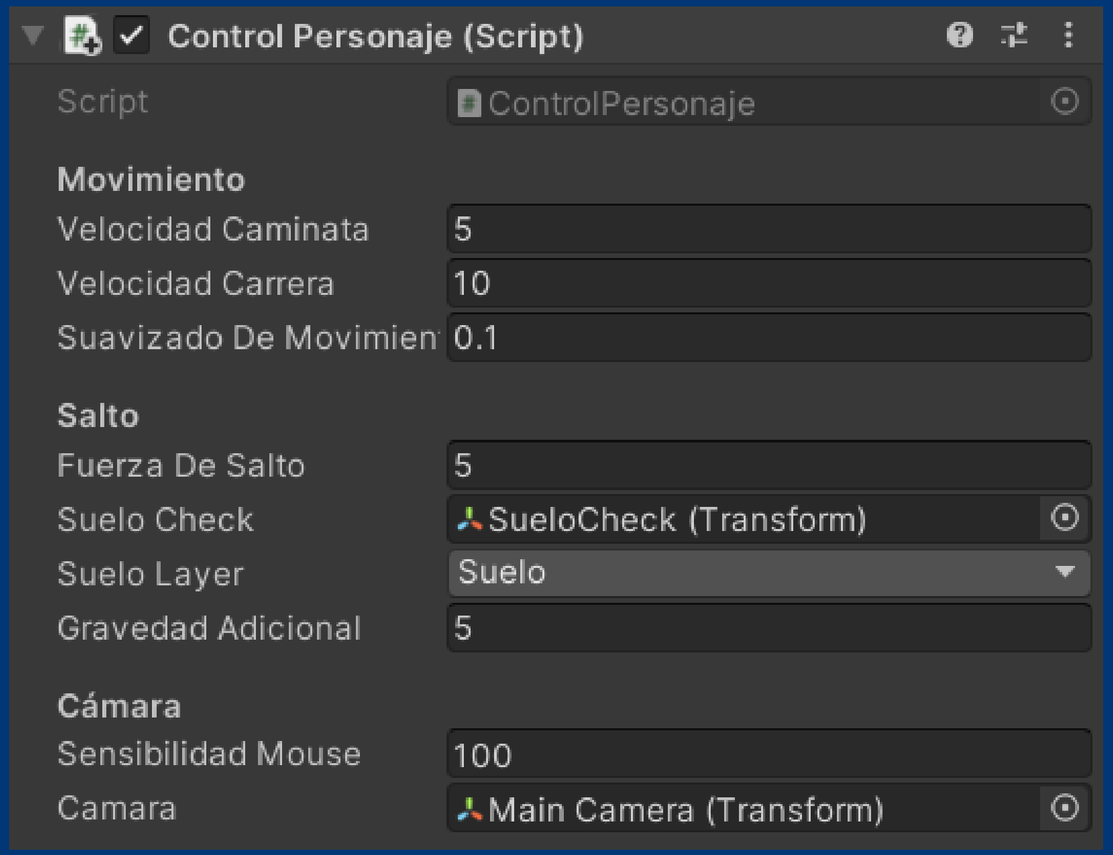
Configuración de transiciones





2

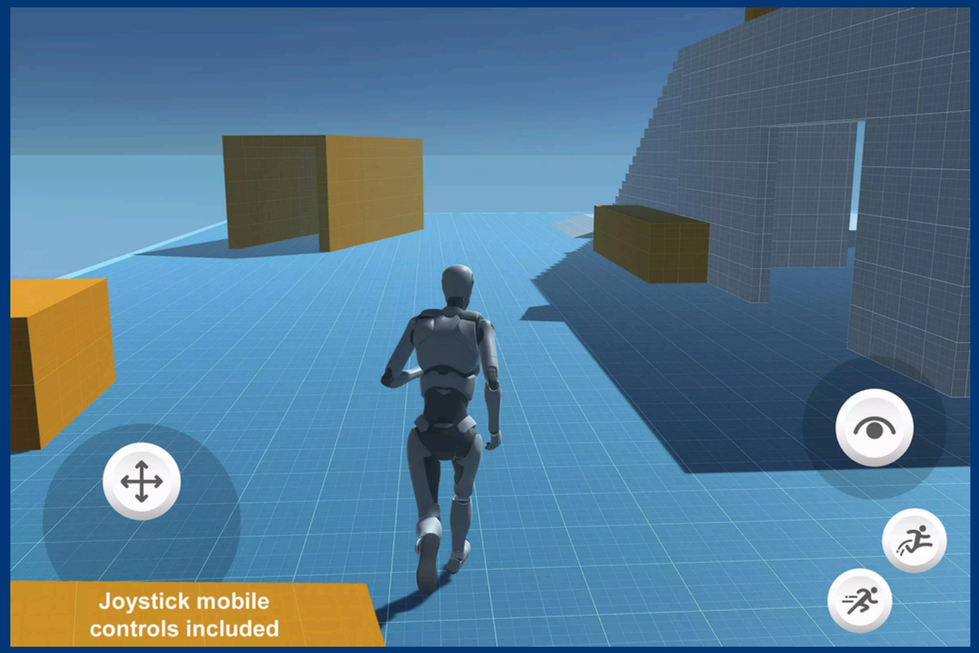
Actualizar Script #6
ControlPersonaje.cs

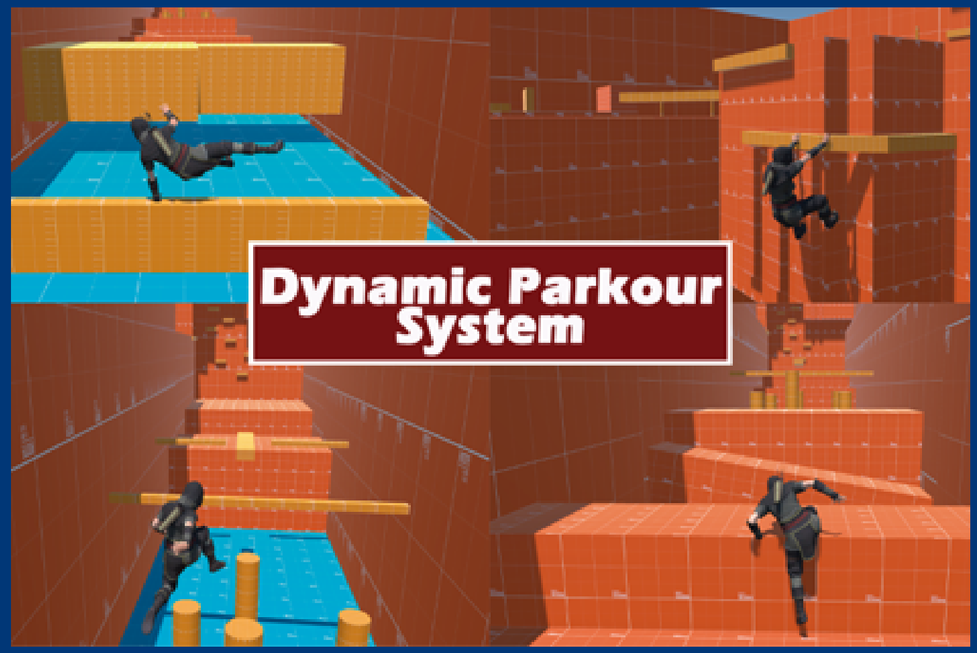


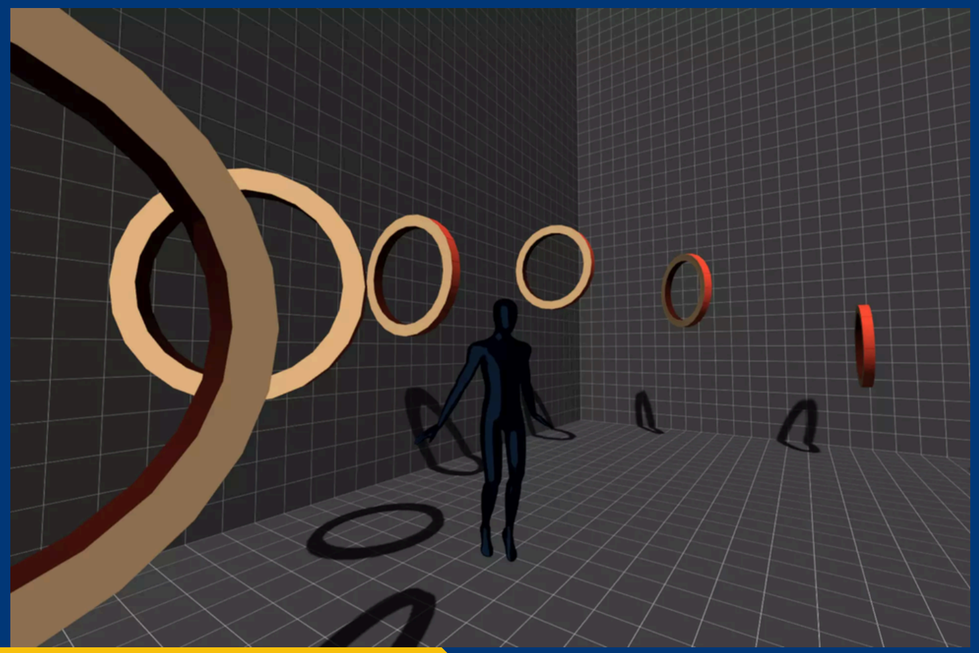
El script controla el golpe con la tecla Q

3 Assets: Character Controller

3


Joystick mobile
controls included


Dynamic Parkour
System

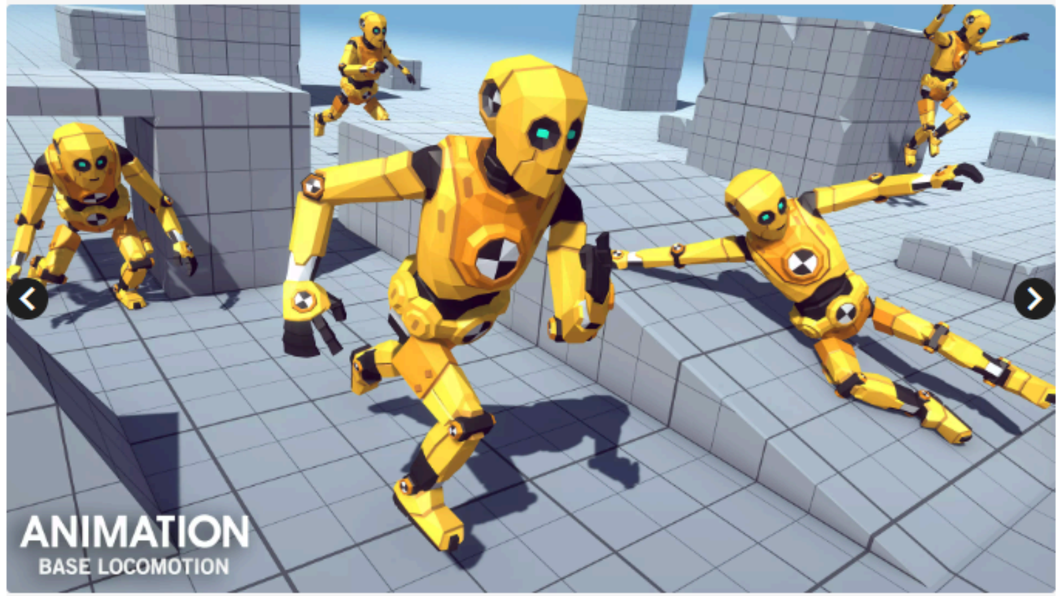



ANIMATION
BASE LOCOMOTION

3

Synty ANIMATION

Home > 3D > Animations > Synty ANIMATION - Base Locomotion - Character Animset



4/12

Overview | Package Content | Releases | Reviews | Publisher info | Asset Quality

Synty ANIMATION is here! Compatible with Synty POLYGON and Simple characters. This pack contains 247 hand crafted animations and a demo character controller crafted with Synty's trademark quality.

Synty ANIMATION - Base Locomotion - Character Animset

Synty Studios (not enough ratings) | (156)

$69.99

Taxes/VAT calculated at checkout

451 views in the past week

License type: Restricted Single Entity

Refund policy

Buy Now

Secure checkout:

Mekrx
★★★★★ 3 months ago

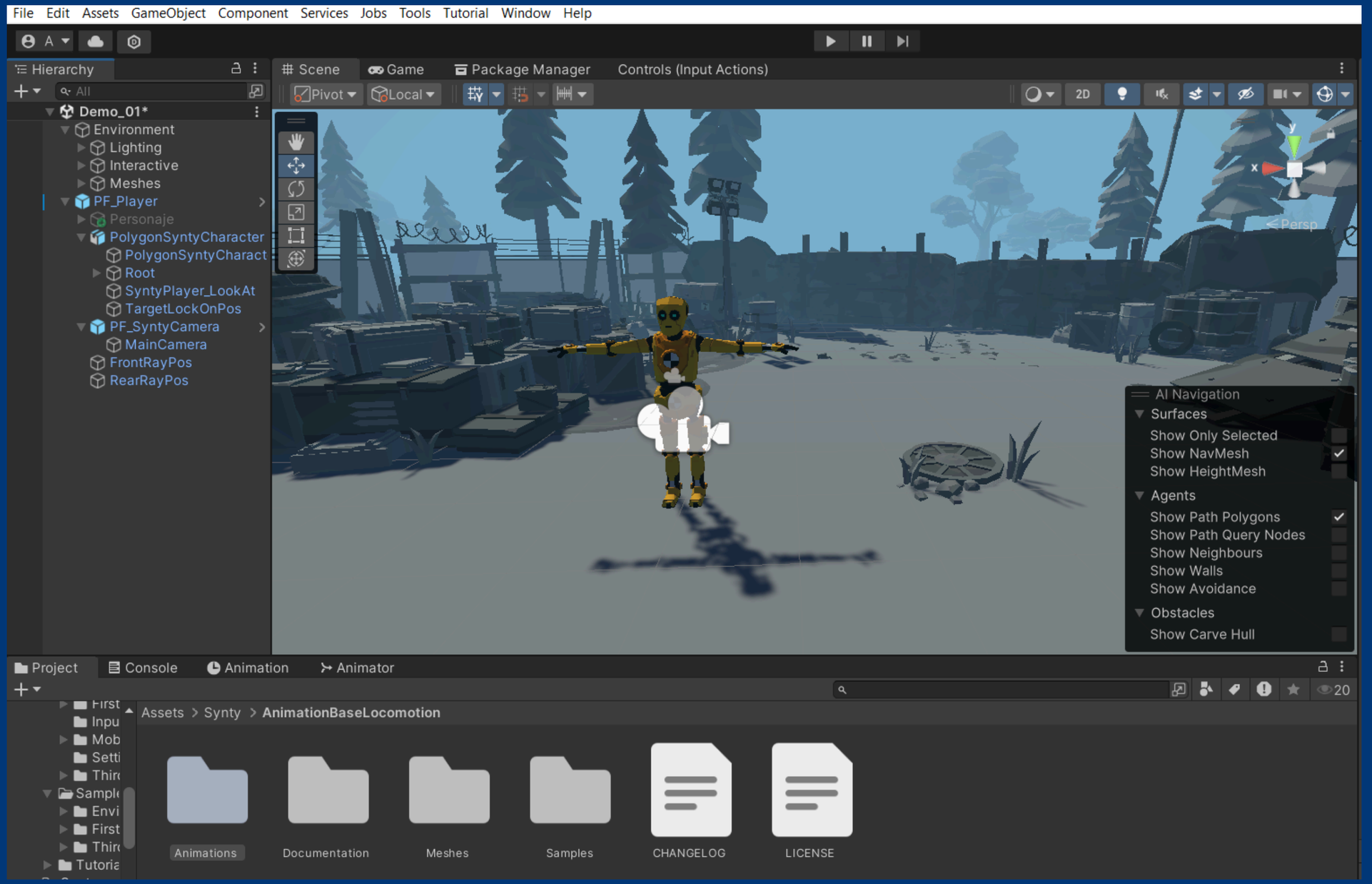
It's a good asset

Very good animations and character controller. I would really like to see more animations with weapons made from polygon assets.
Read more reviews

License agreement	Standard Unity Asset Store EULA
License type	Restricted Single Entity
File size	49.9 MB
Latest version	1.0.4
Latest release date	Apr 2, 2024
Original Unity version	2021.3.17 or higher

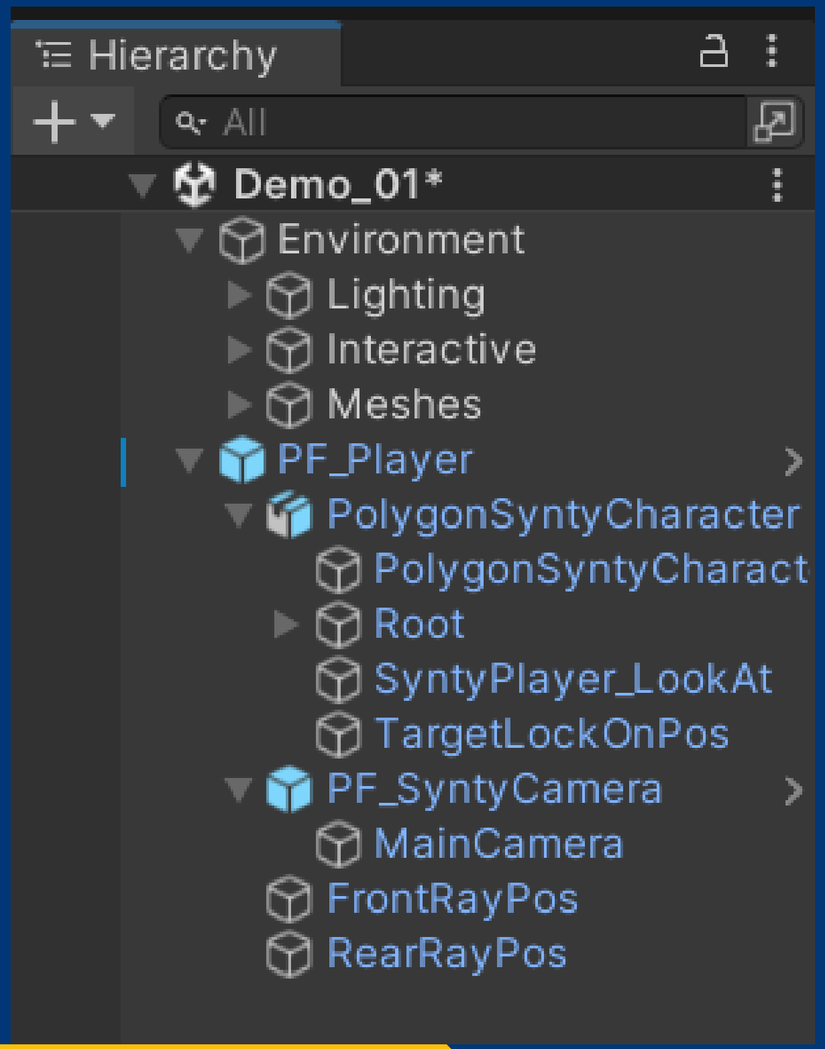
3

Abrimos la Scena

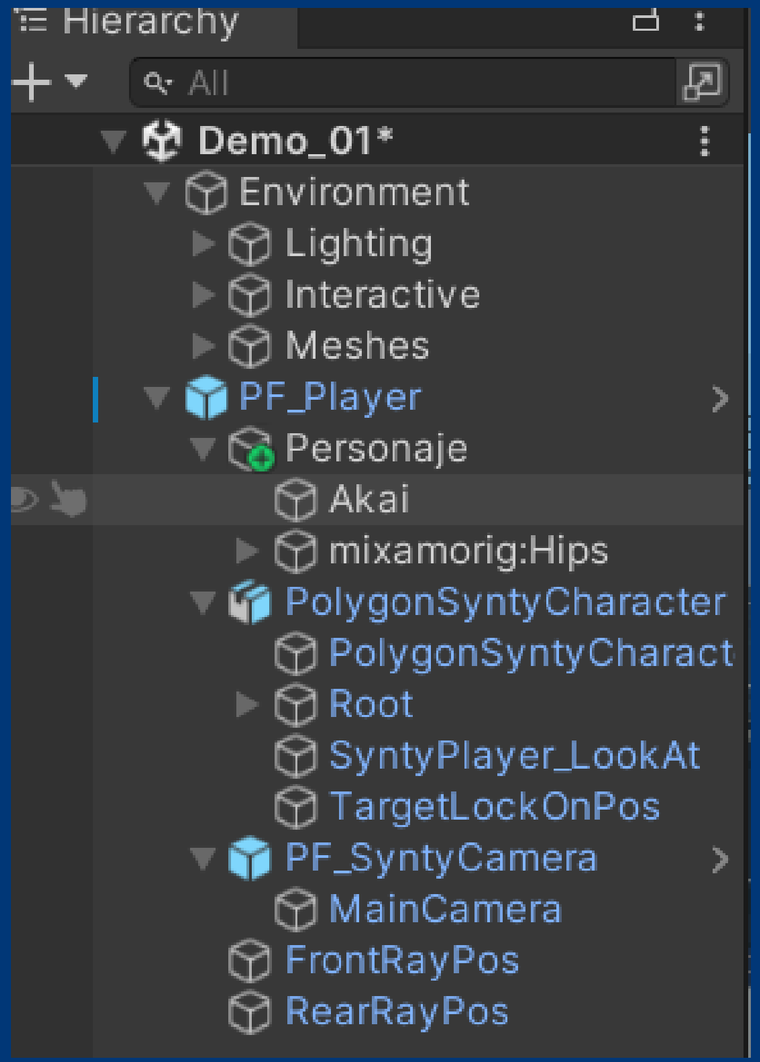


3

Identificamos los elementos del personaje

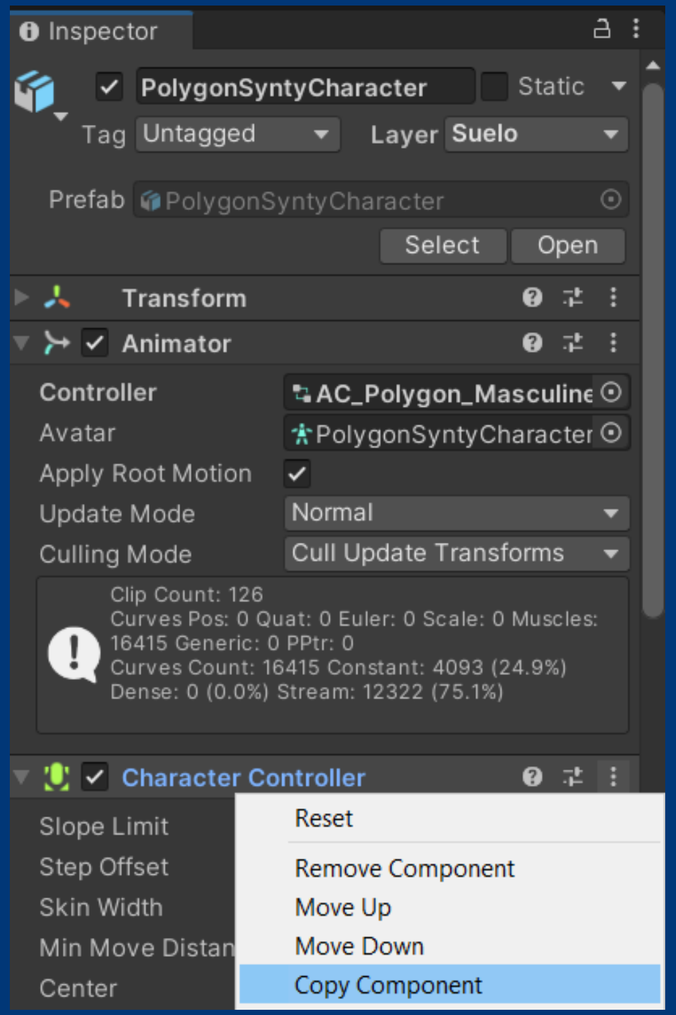


Agregamos al Personaje dentro del PF_Player

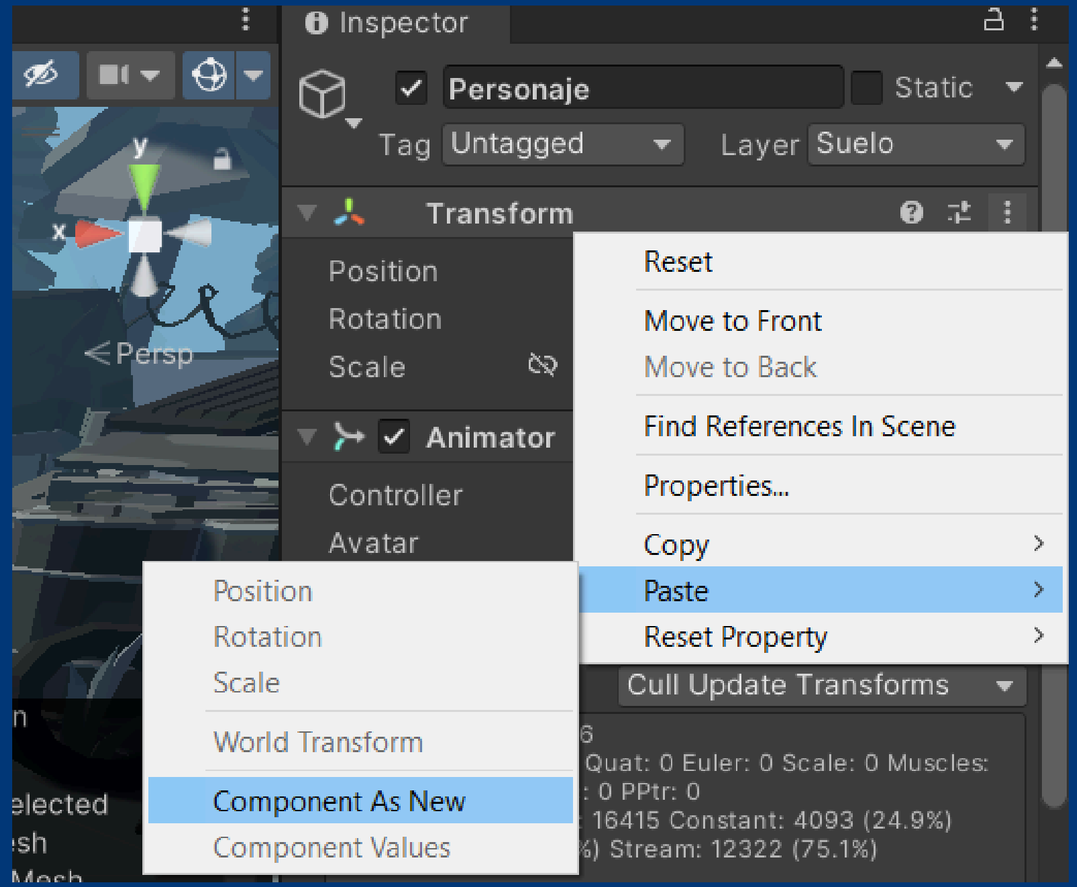


3

En los tres puntos en la propiedad el personaje del objeto le damos en CopyComponent



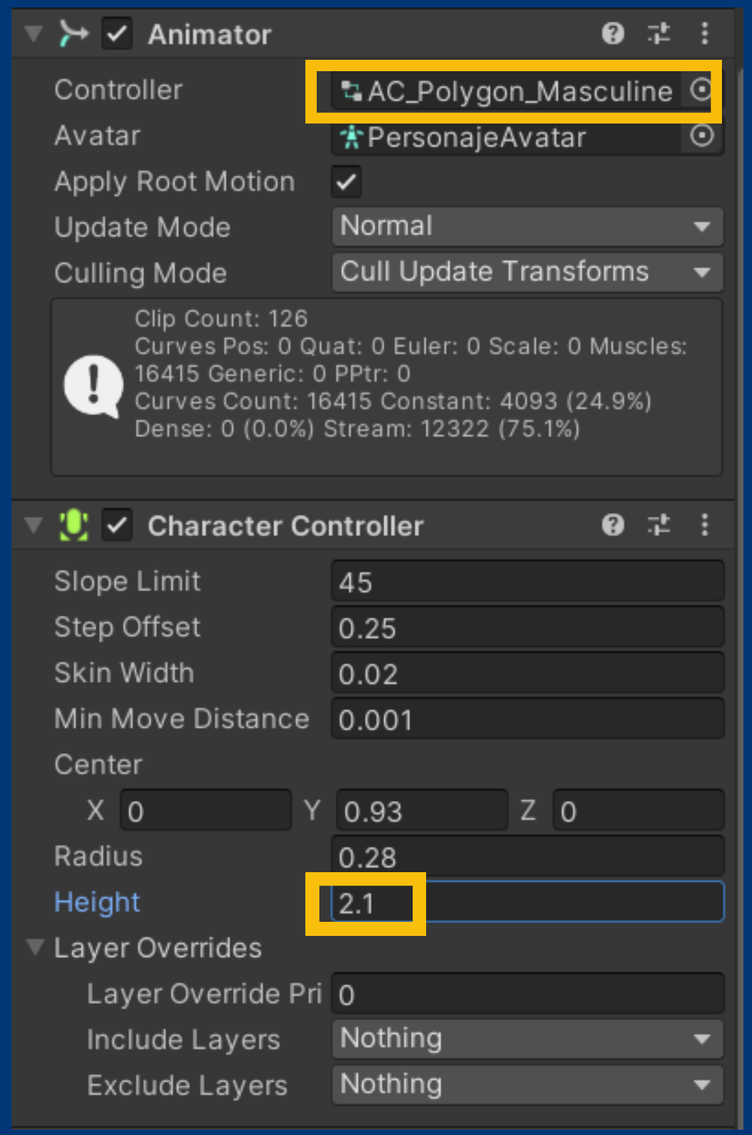
En el personaje de damos en los tres puntos (Cualquier propiedad) - Paste y Component As New

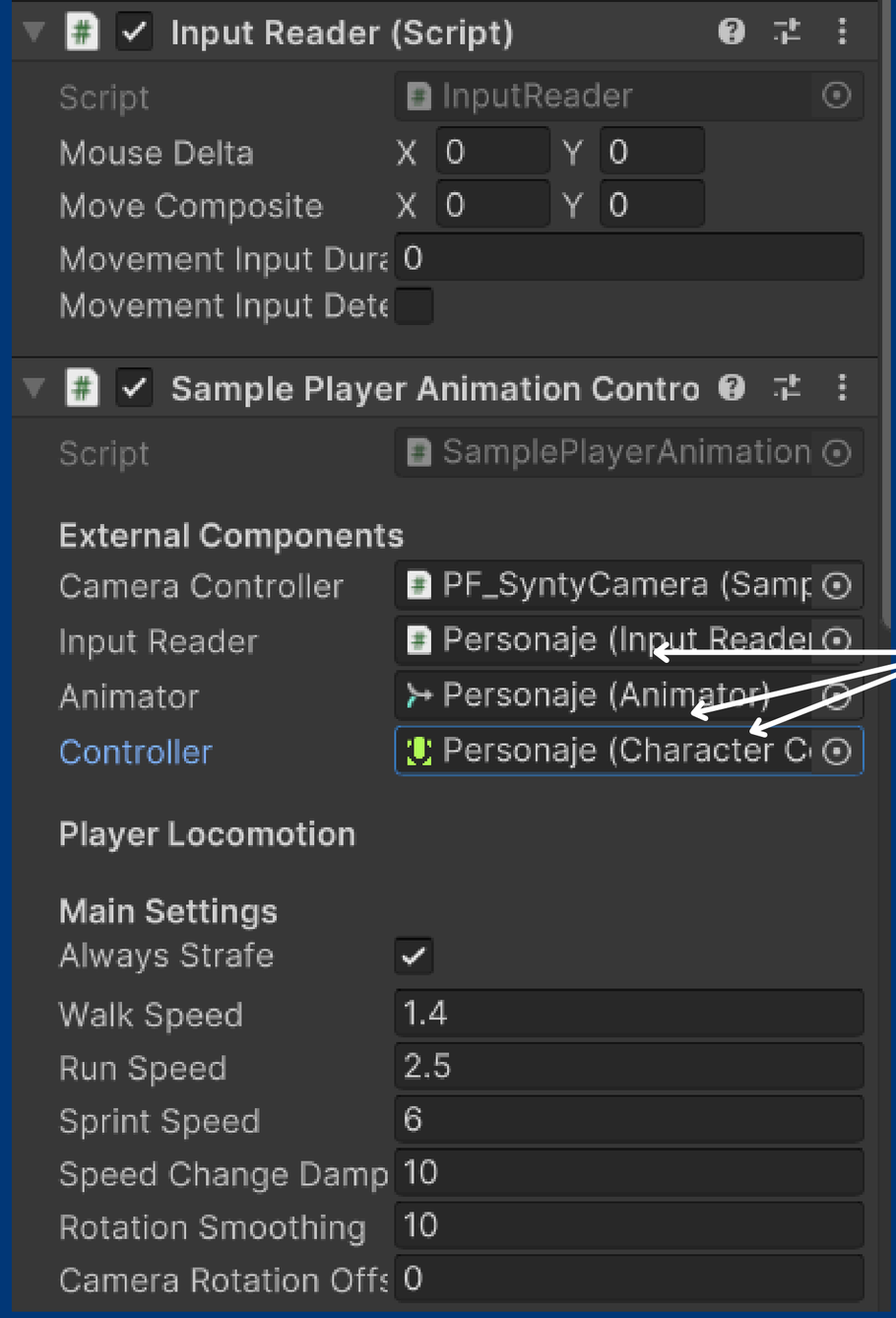


Va copiar el componente y sus propiedades al nuevo

3

Realizo la copia de todas las propiedades



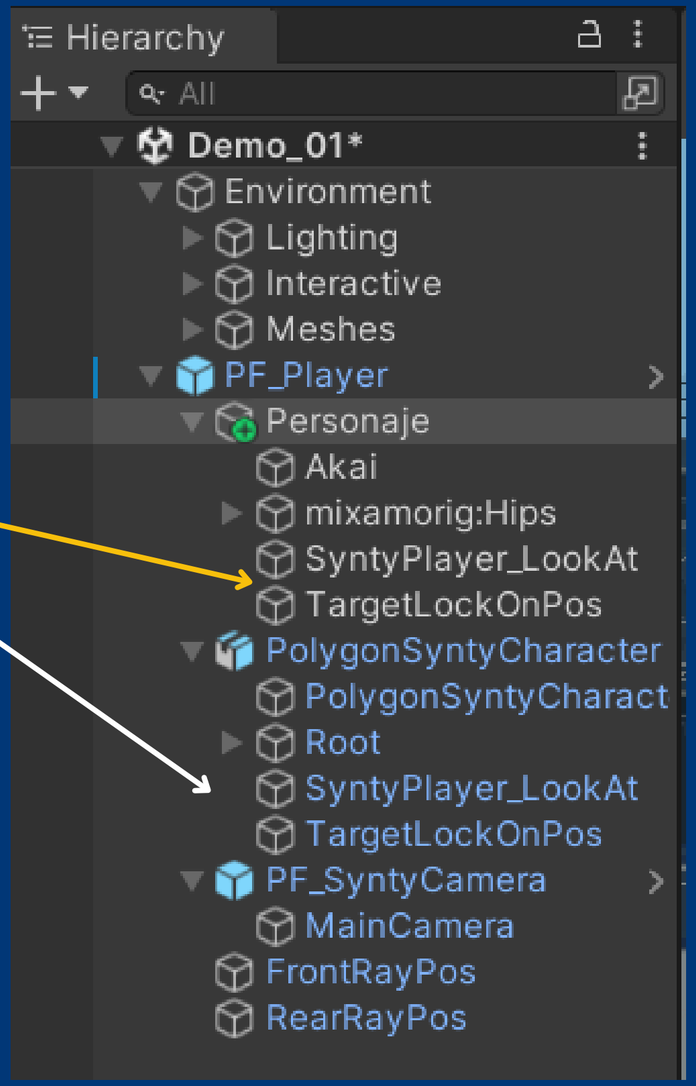


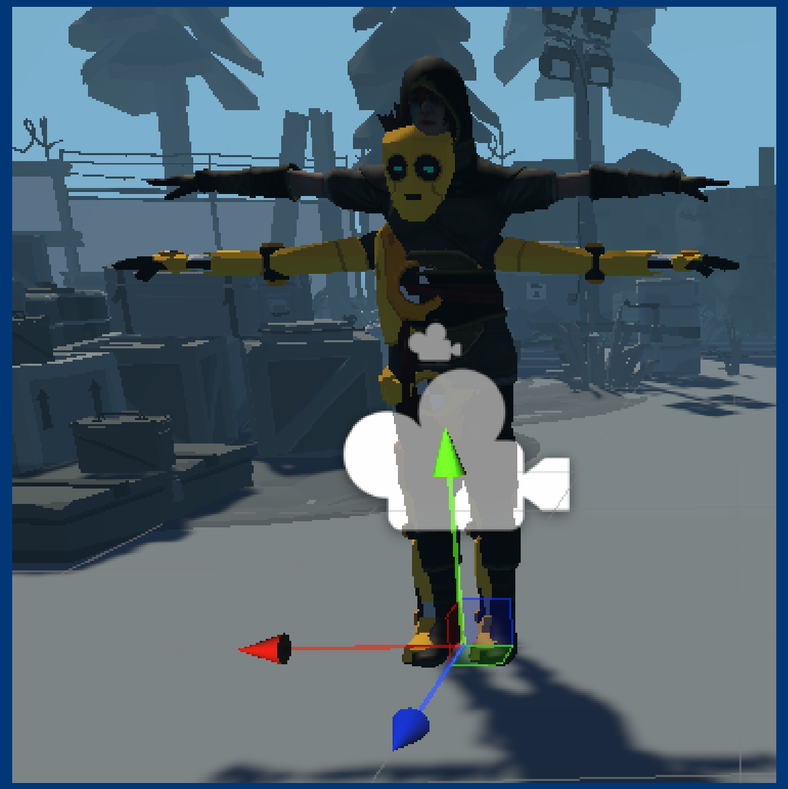
Arrastro al mismo objeto Personaje

Camera controller dejo el mismo

3

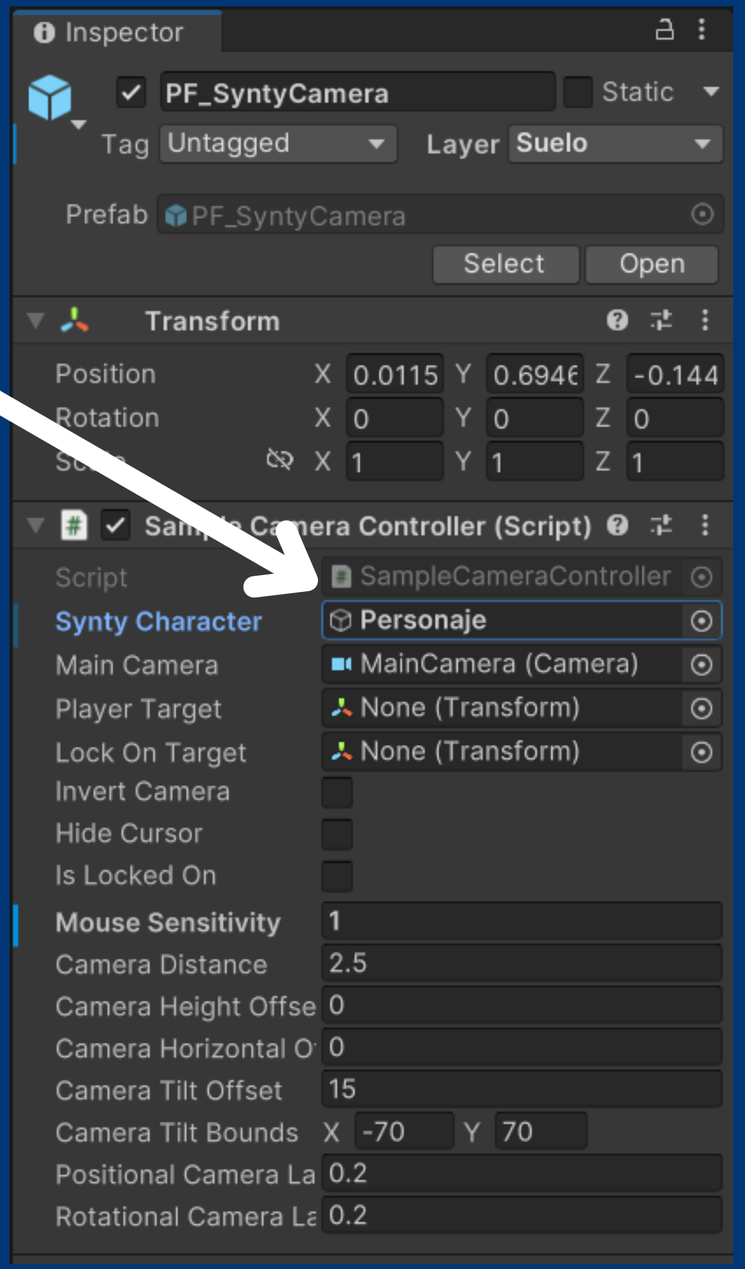
Copio los objetos SyntyPlayer_LookAt y TargetLockOnPos del personaje de ejemplo al Personaje propio





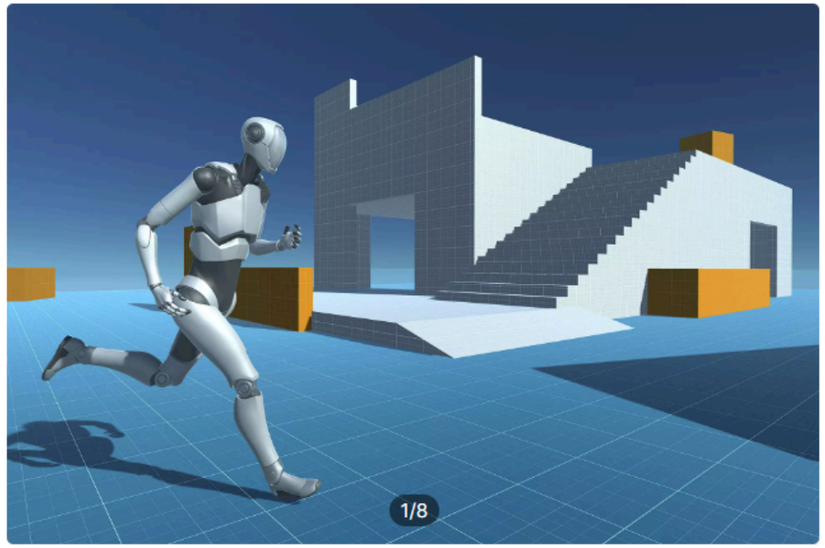
3

En el objeto PF_SyntyCamera arrastro al personaje a Synty Caracter



3

Starter Assets ThirdPerson



Starter Assets - ThirdPerson | Upd...

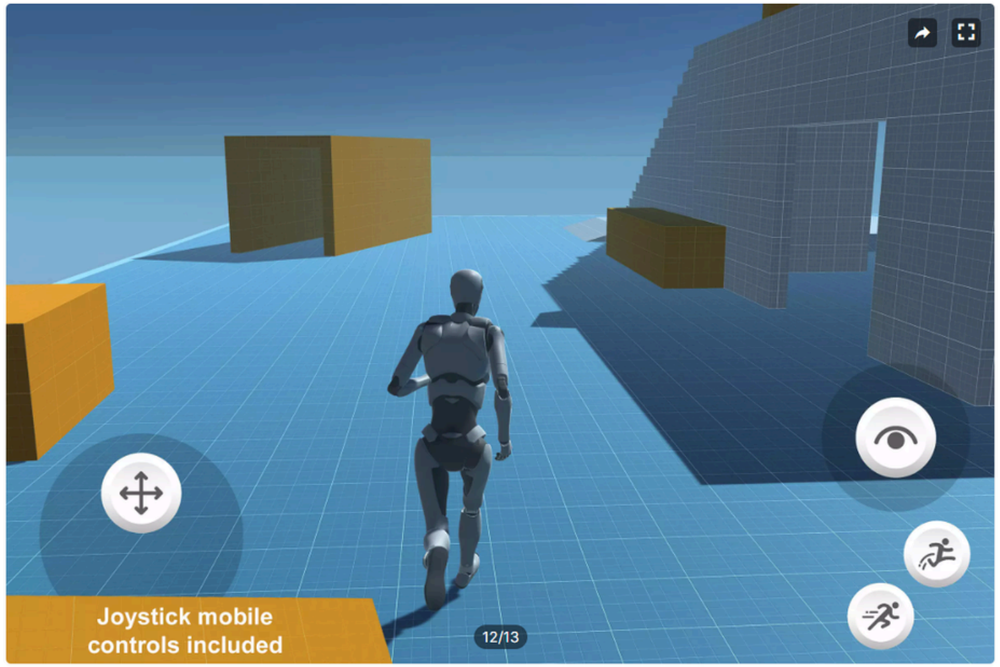
Unity Technologies ★★★★☆ (265) | ♥ (11322)

Taxes/VAT calculated at checkout

Product Information

License agreement	Non standard EULA
File size	69.9 MB
Latest version	1.1.5
Latest release date	Jun 16, 2023
Original Unity version	2020.3.0 or higher
Support	Visit site

Reviews ★★★★☆



Starter Assets: Character Controllers | URP

Unity Technologies ★★★☆☆ (19) | ♥ (587)

FREE

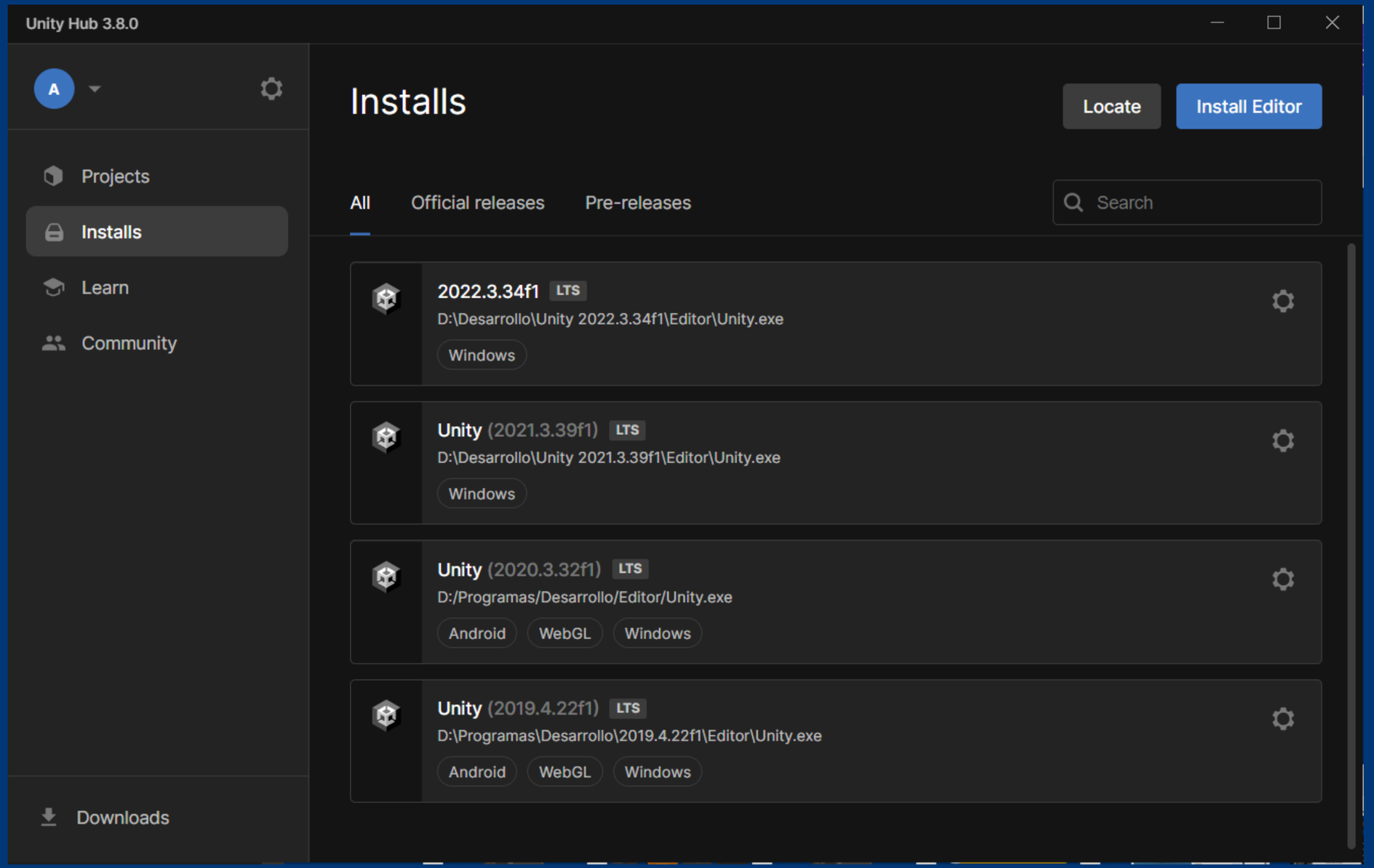
3937 views in the past week

Add to My Assets

License agreement	Standard Unity Asset Store EULA
License type	Extension Asset
File size	70.3 MB
Latest version	1.1.0
Latest release date	Mar 29, 2024
Original Unity version	2022.3.0 or higher
Support	Visit site

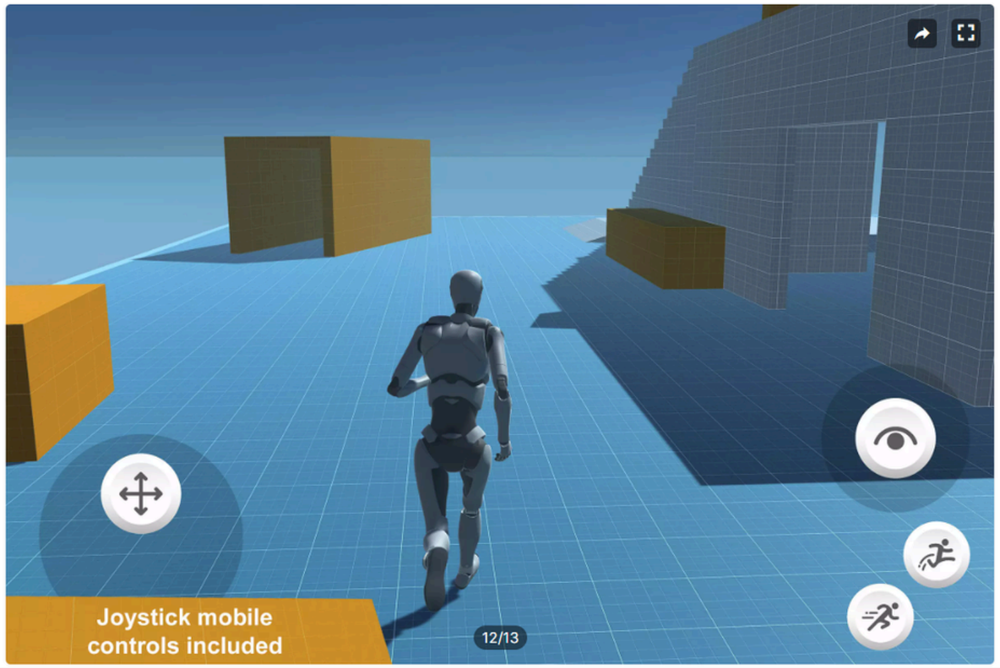
3

Instalo la versión de Unity 2022 LTS



3

Descargar Assets



Starter Assets: Character Controllers | URP

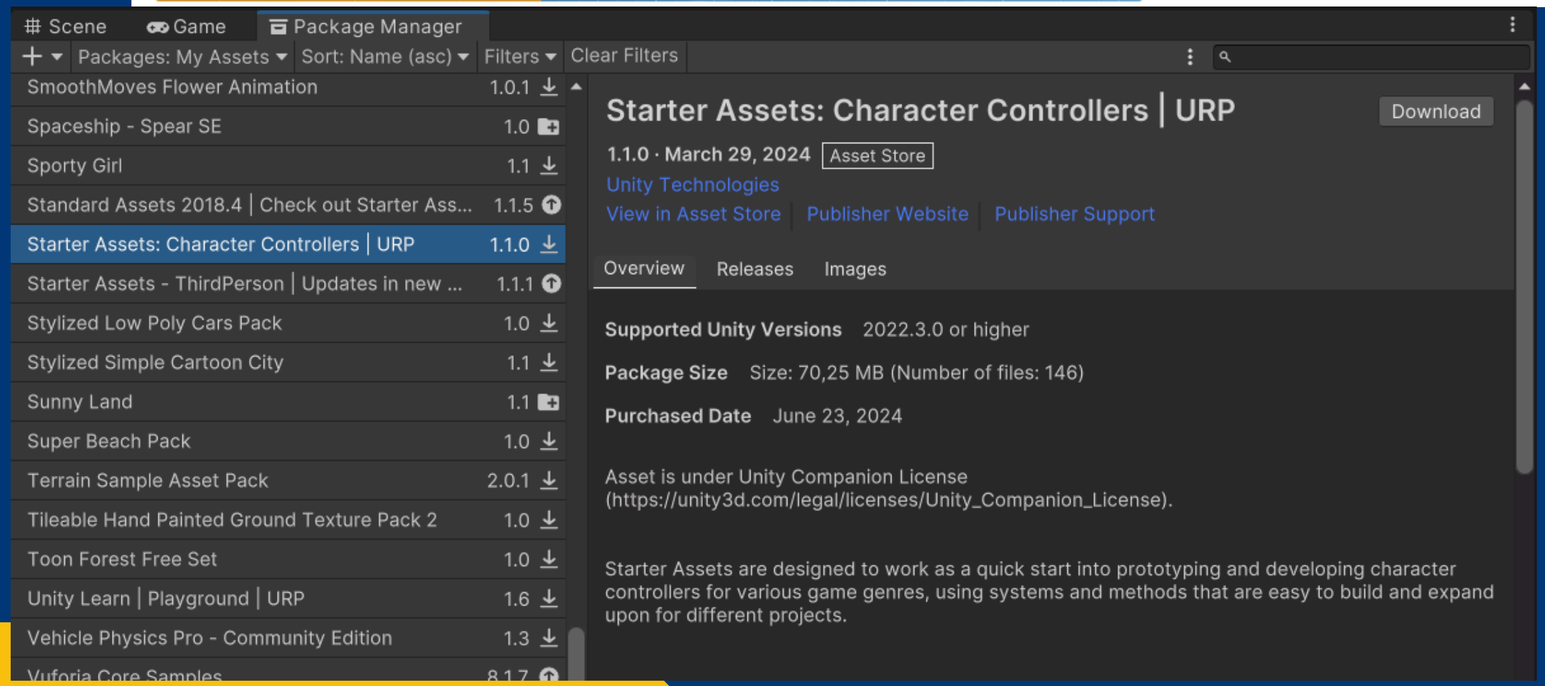
Unity Technologies ★★★☆☆ (19) | ♥ (587)

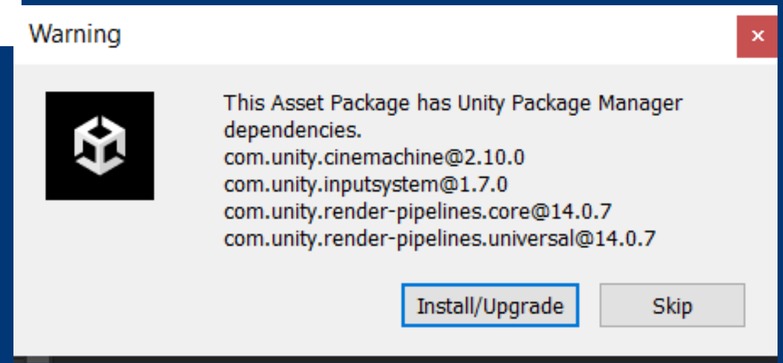
FREE

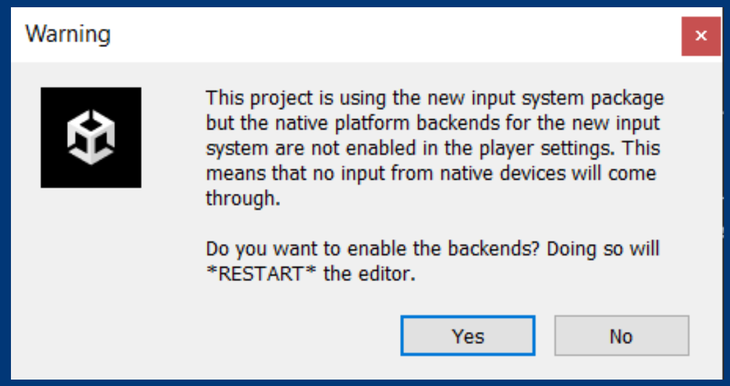
3937 views in the past week

Add to My Assets

License agreement	Standard Unity Asset Store EULA
License type	Extension Asset
File size	70.3 MB
Latest version	1.1.0
Latest release date	Mar 29, 2024
Original Unity version	2022.3.0 or higher
Support	Visit site

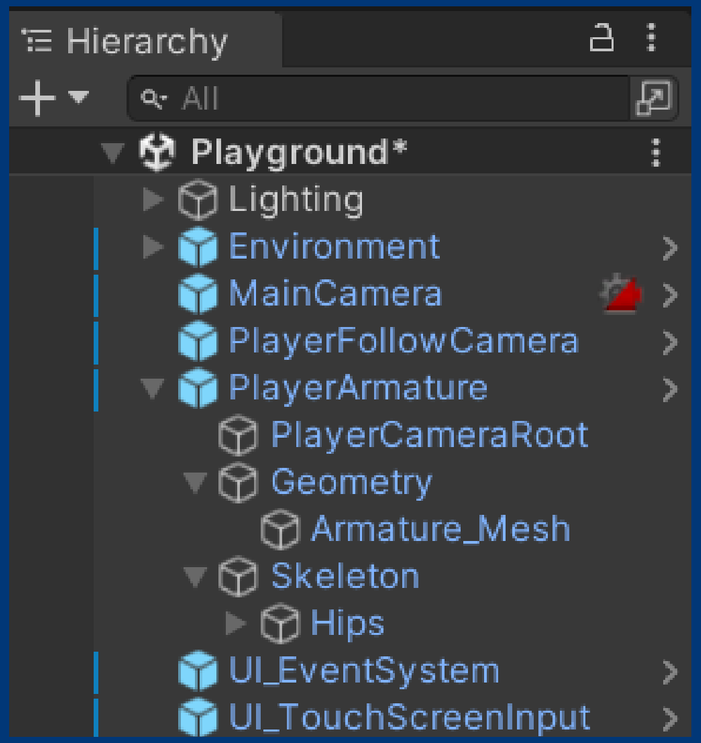




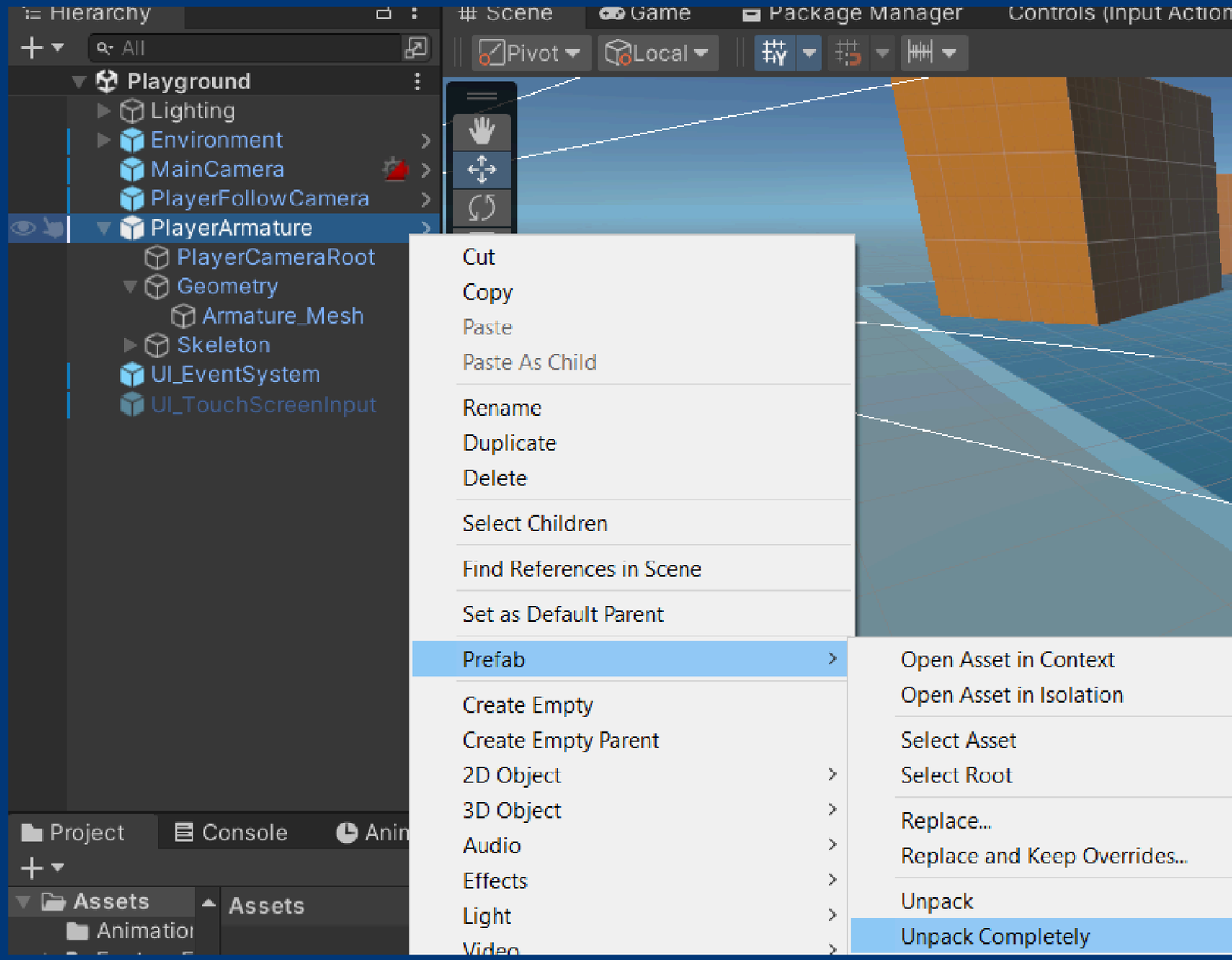


3

Reviso los objetos del modelo de ejemplo

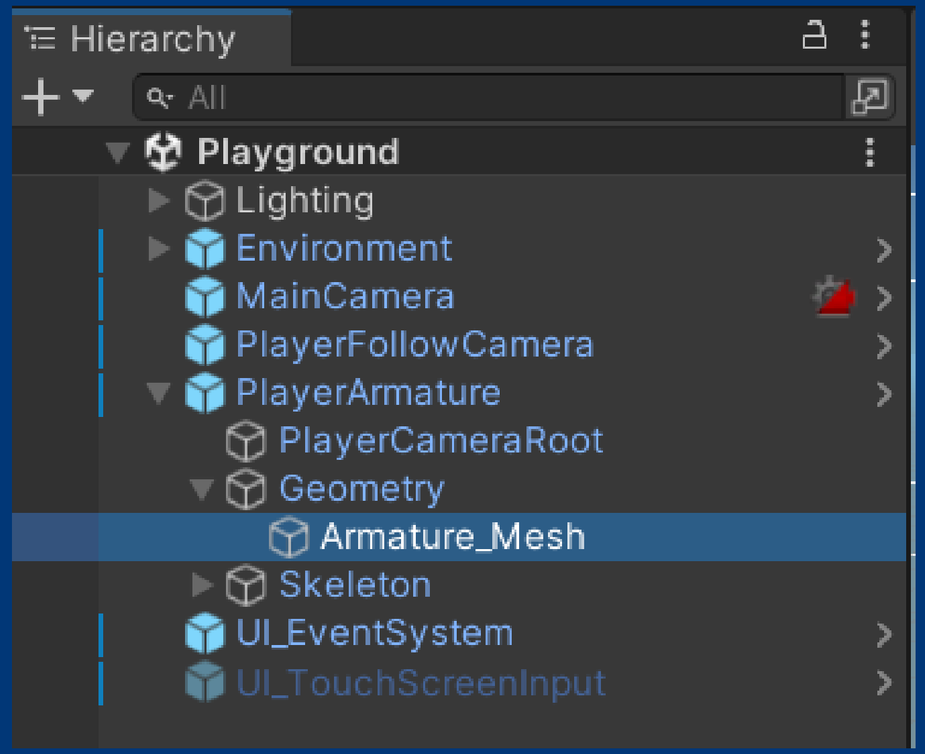


Desempaqueto el Prefab

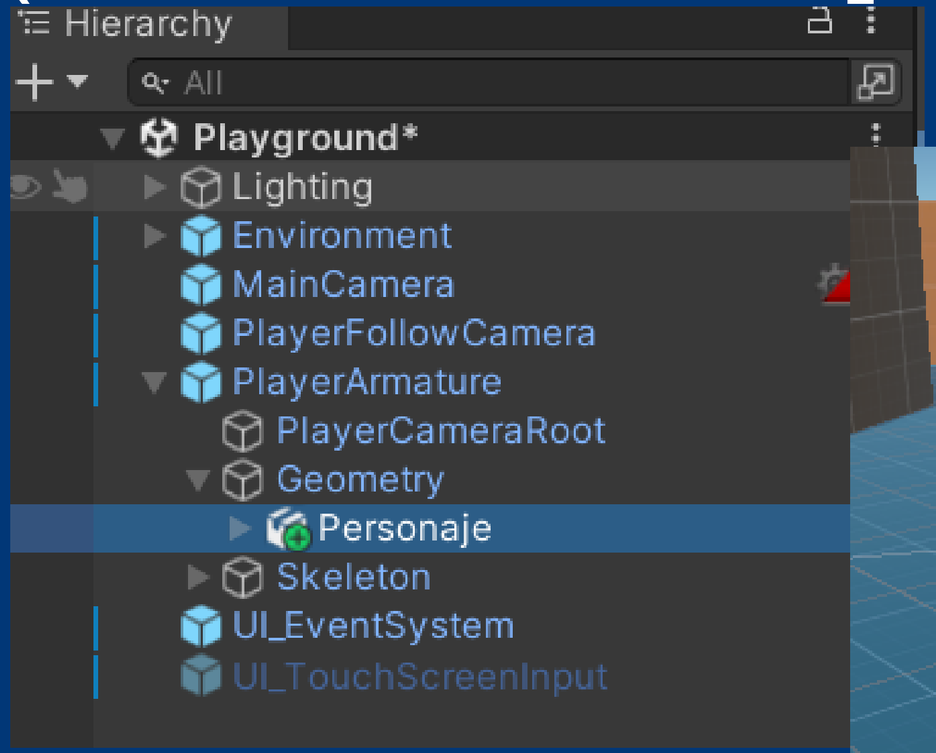


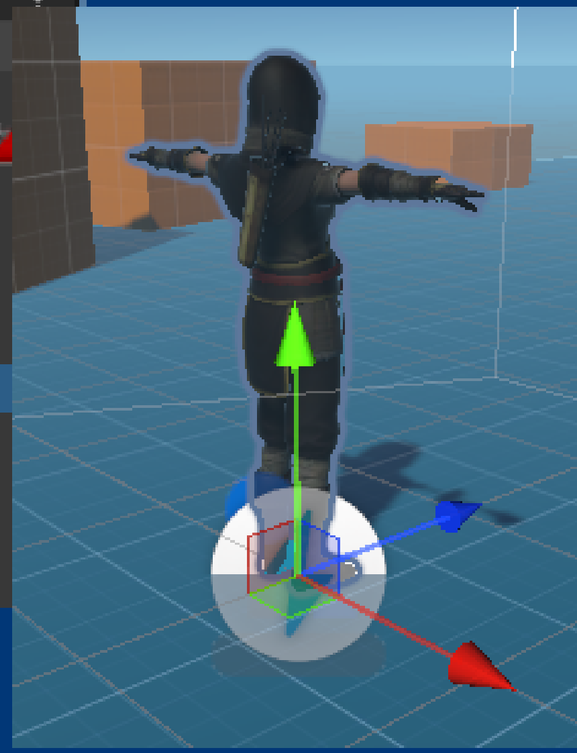
3

Elimino el Armature_Mesh "el robot"



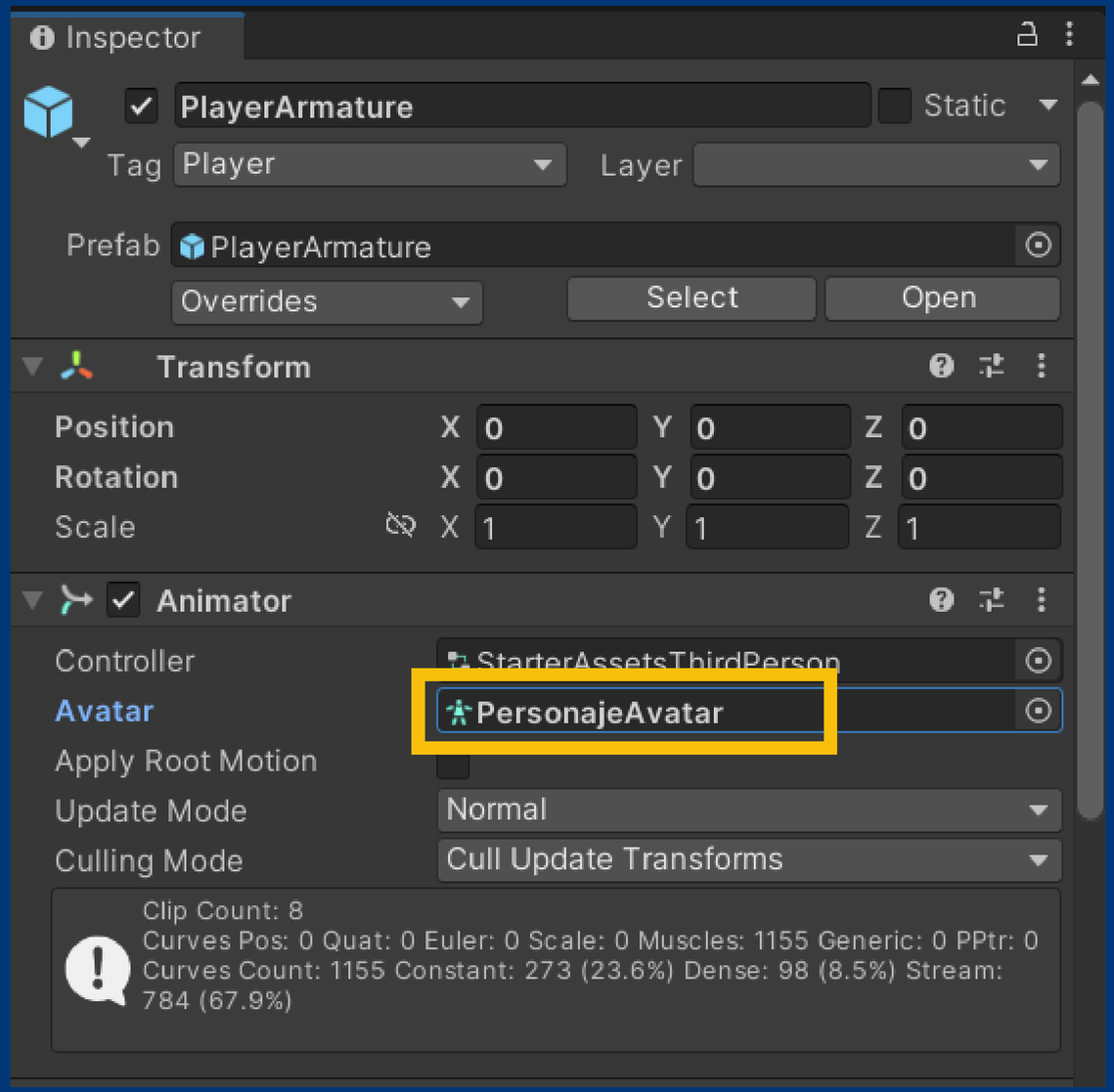
Arrastro el personaje hacia Geometry (donde estaba el Armature_Mesh)





3

En el Objeto PlayerArmadure selecciono la armadura del personaje

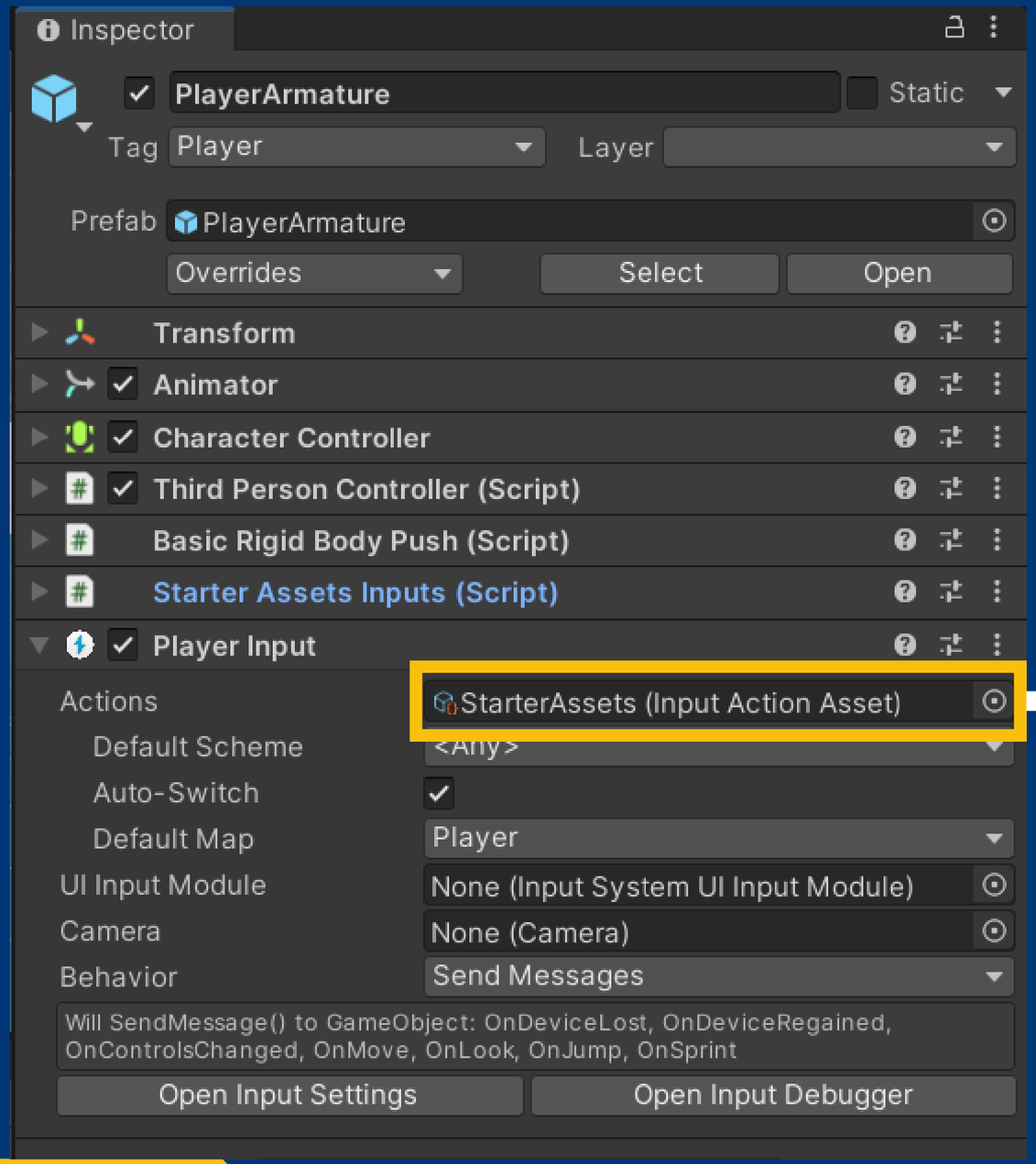


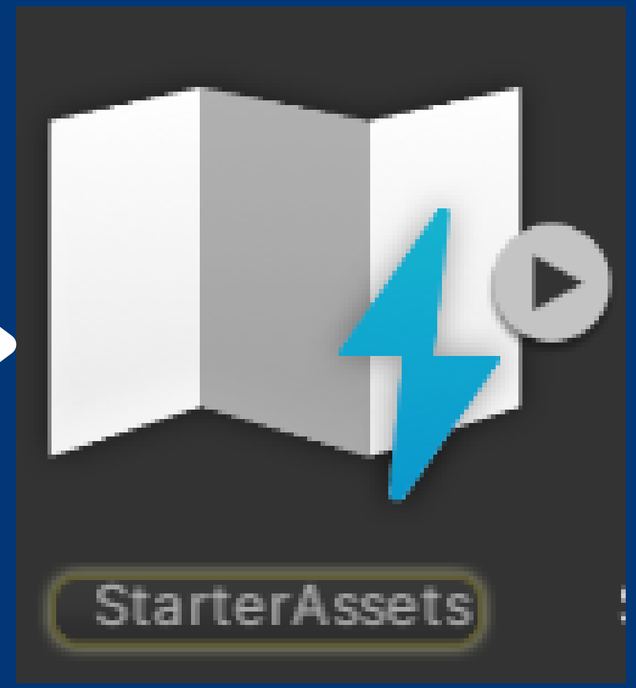
Probar

4 Imput System y acciones adicionales

4

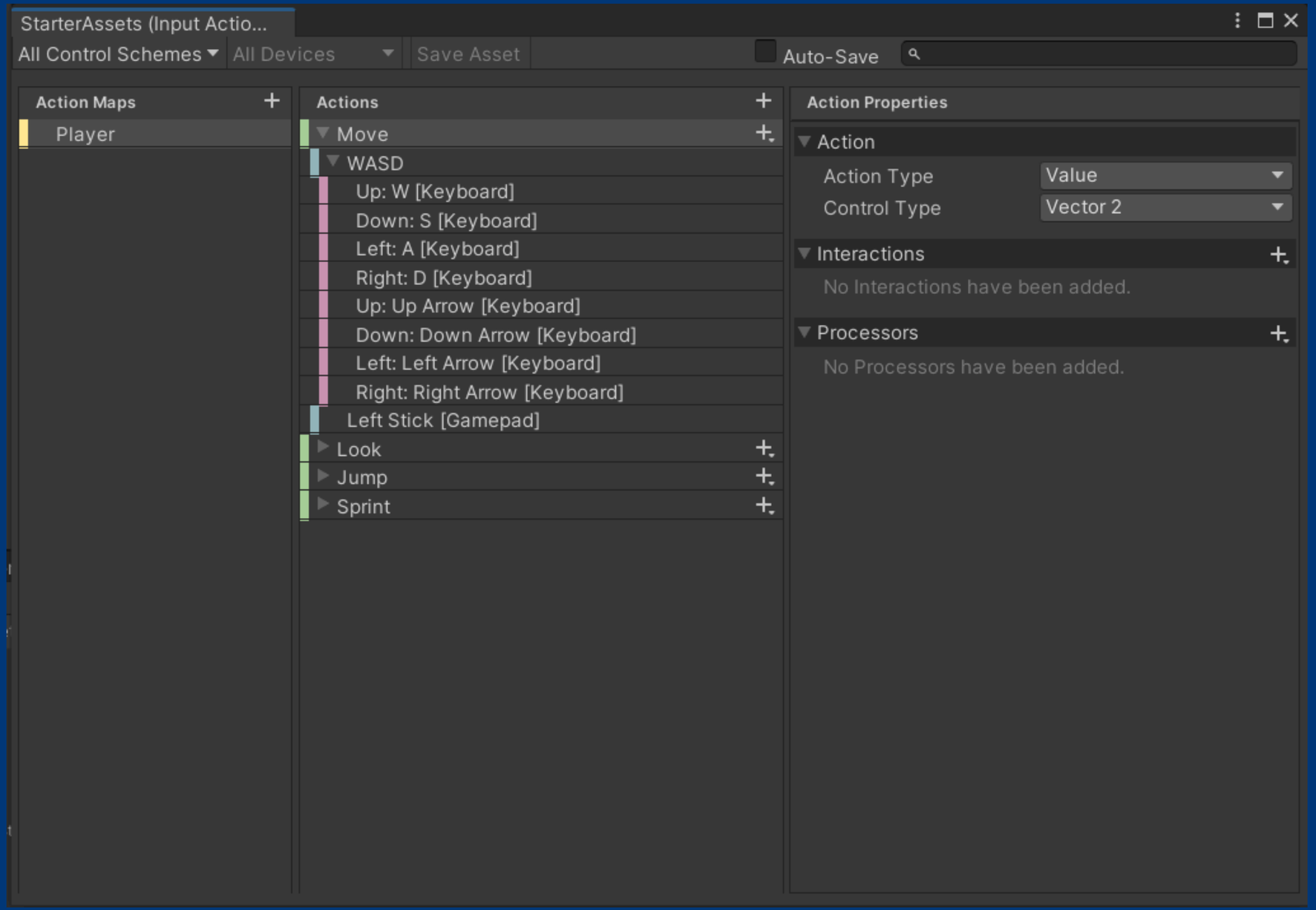
Busco el archivo de Input Action Asset





4

En Unity, el "Input Action Asset" es una parte del sistema de entrada (Input System) que permite definir y gestionar las entradas del usuario de una manera más estructurada y flexible en comparación con el antiguo sistema de entrada basado en Input.GetAxis y Input.GetButton.



4

Funciones del Input Action Asset

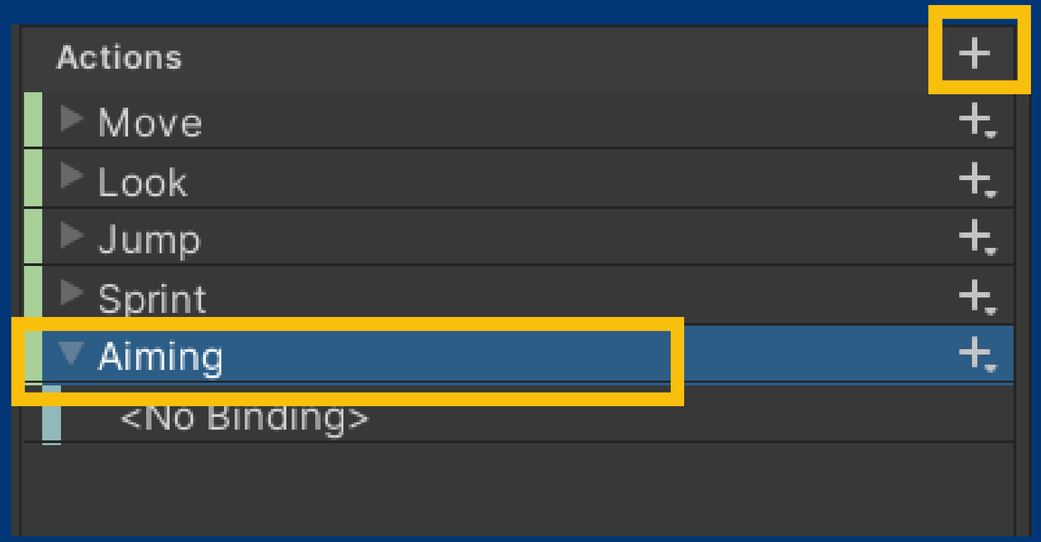
- Definición de Acciones
- Asignación de Controles
- Mapeo Flexible
- Soporte Multiplataforma
- Eventos y Callbacks
- Control de Estado de Entrada

4

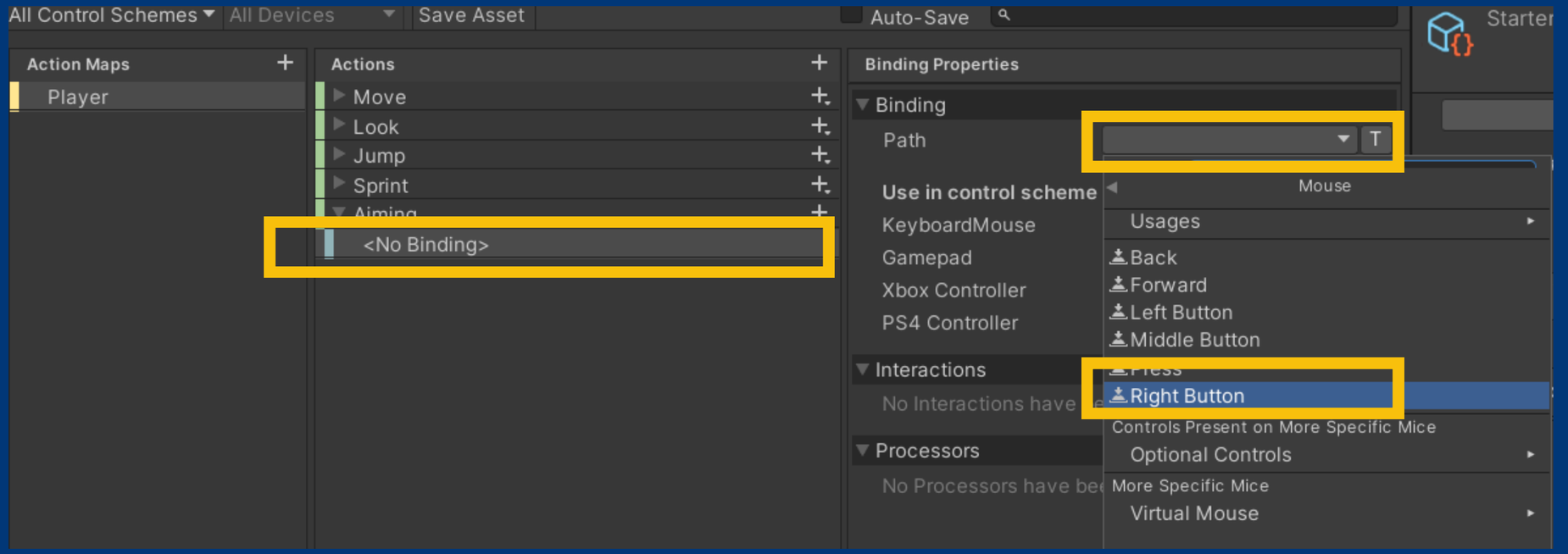
Definir acciones y controles con el Sistema Imput

4

Creo una nueva acción "AIMING"

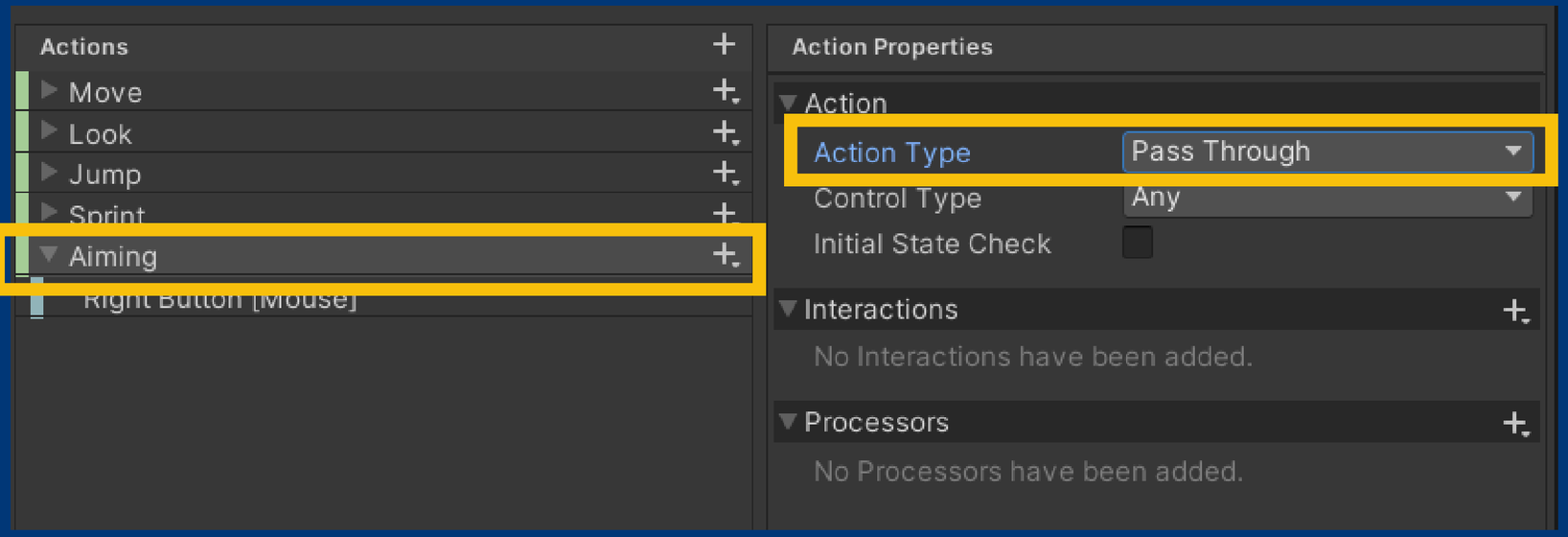


Vinculamos el Path a Mouse - Right Button



4

Seleccionando la acción Aiming



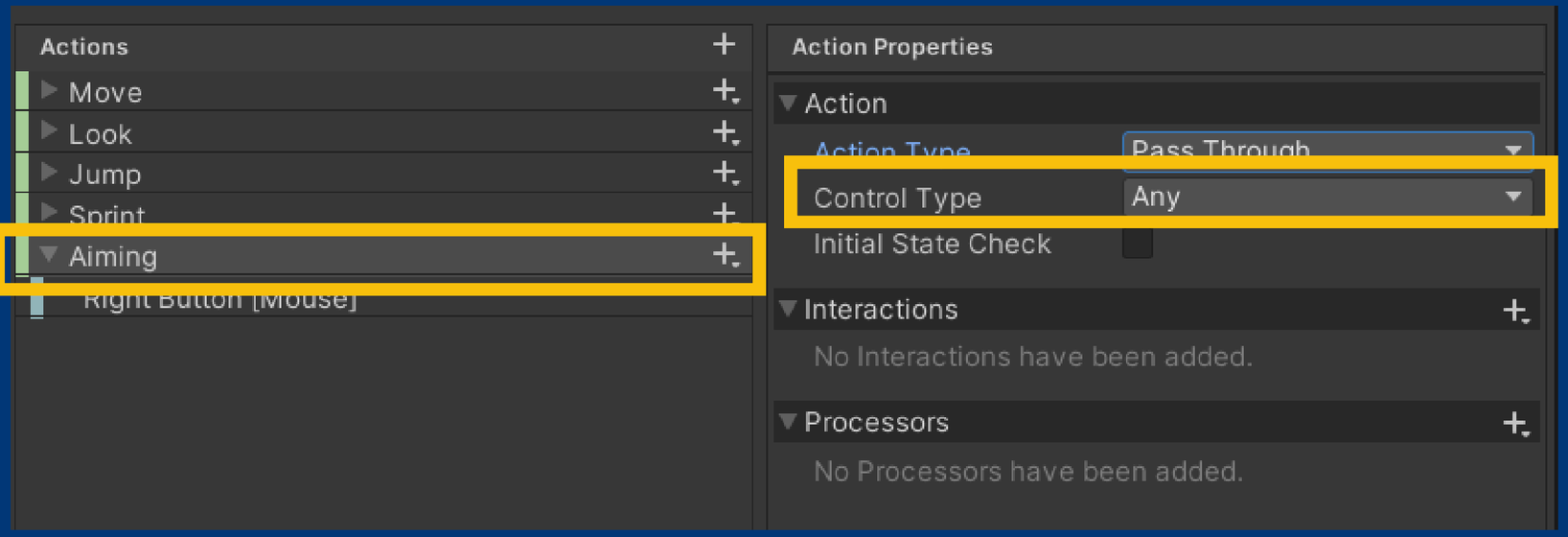
En **Action Type** seleccionamos **Pass Through**

Action Type determina el tipo de interacción que una acción de entrada representa

- **Button:** Representa una acción binaria que puede estar en estado activado o desactivado (por ejemplo, pulsar un botón o una tecla). Se usa para acciones como "saltar" o "disparar".
- **Value:** Representa una acción que produce un valor continuo o discreto (por ejemplo, un eje de un joystick o la presión de un gatillo). Se usa para acciones como "moverse" o "mirar alrededor".
- **Pass-Through:** Similar a "Value", pero diseñado para casos en los que múltiples entradas pueden afectar la misma acción simultáneamente. Por ejemplo, si tienes múltiples controladores que pueden mover un personaje.
- **Vector2 y Vector3:** Representan acciones que producen vectores bidimensionales o tridimensionales, como los movimientos en un joystick o la entrada de movimiento en un espacio 3D.

4

Seleccionando la acción Aiming



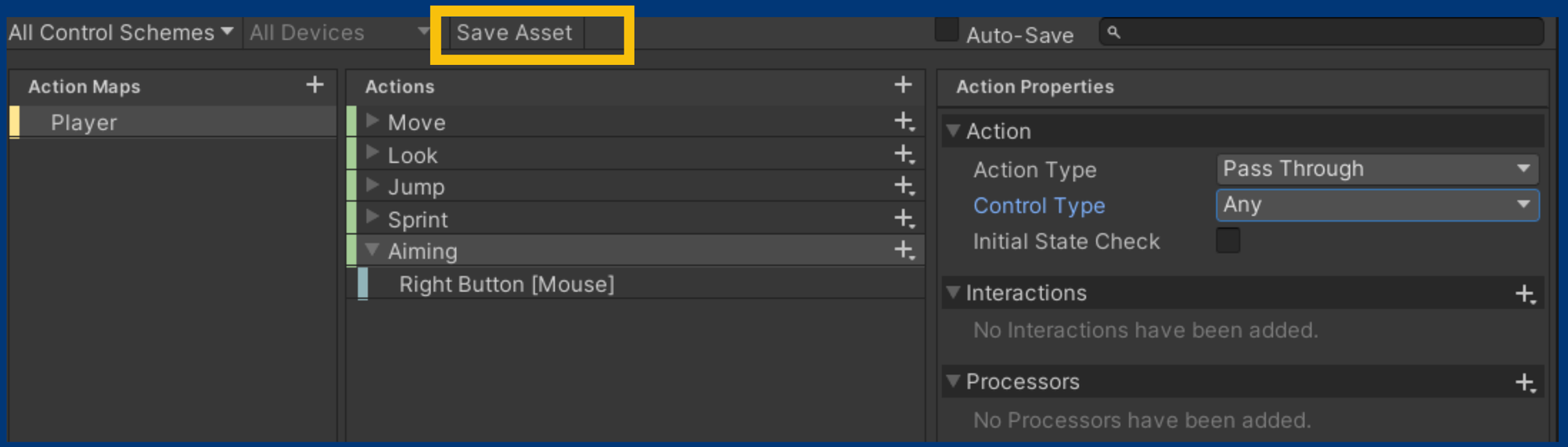
En **Control Type** seleccionamos **Any**

Control Type especifica qué tipo de control de entrada puede activar la acción.

- **Button:** Se utiliza para entradas digitales simples, como botones de teclado o botones de un controlador.
- **Stick:** Para entradas de joystick que proporcionan valores de dirección.
- **Trigger:** Para entradas de gatillo analógicas que proporcionan un valor en un rango (por ejemplo, los gatillos de un controlador).
- **Axis:** Para entradas de eje, como las ruedas de desplazamiento o ejes de joystick.
- **Dpad:** Para entradas de un pad direccional.
- **Mouse:** Para entradas específicas del ratón, como movimiento, clics y desplazamiento de la rueda.
- **Touchscreen:** Para entradas táctiles, como toques y gestos en una pantalla táctil.

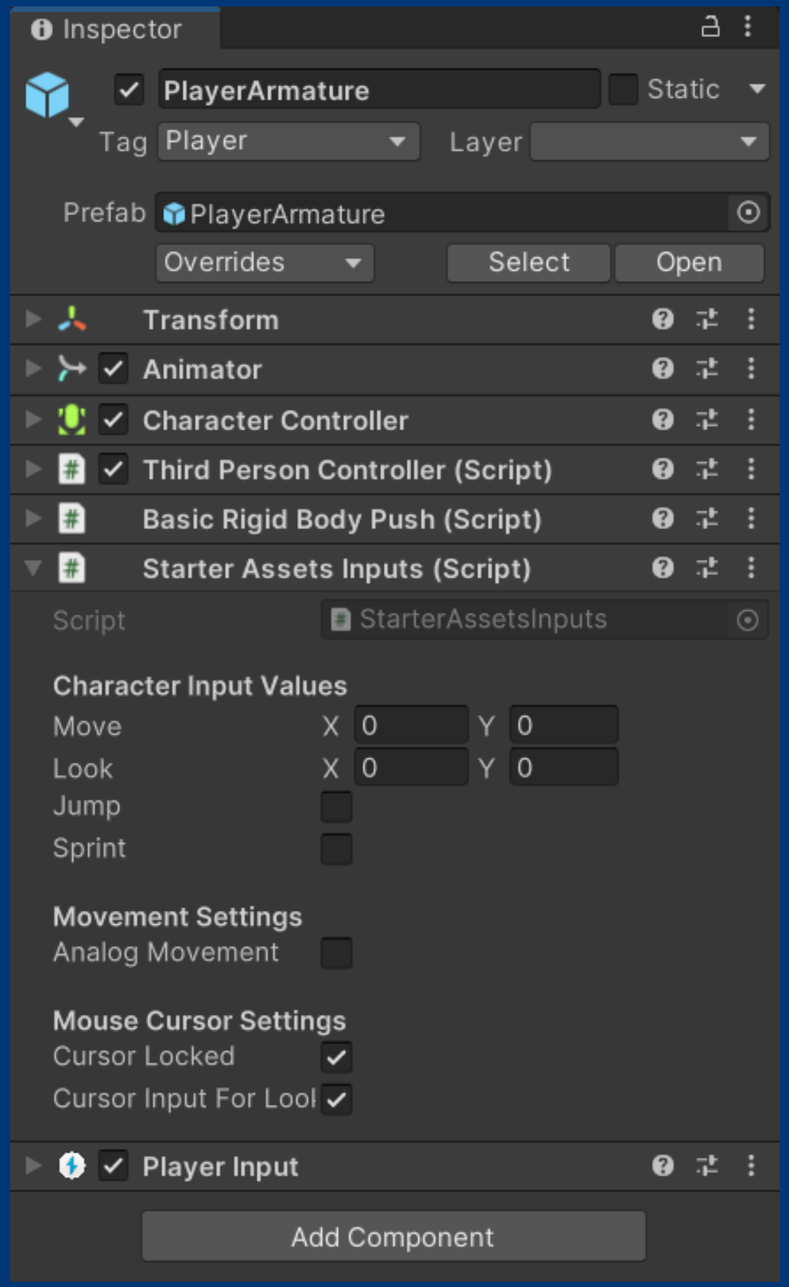
4

Guardamos los cambios

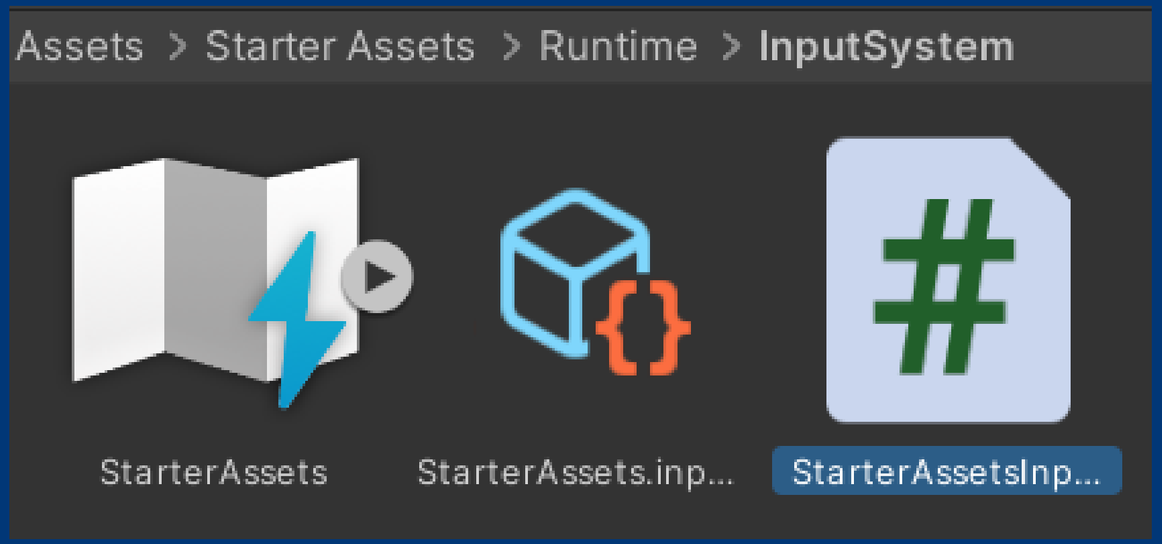


4

Buscamos el Script Standar Assets Inputs



Abro el Script



El script proporcionan implementaciones predefinidas para gestionar diferentes tipos de entradas de usuario (Variables y metodos para controlar las acciones del personaje)

4

Linea 15

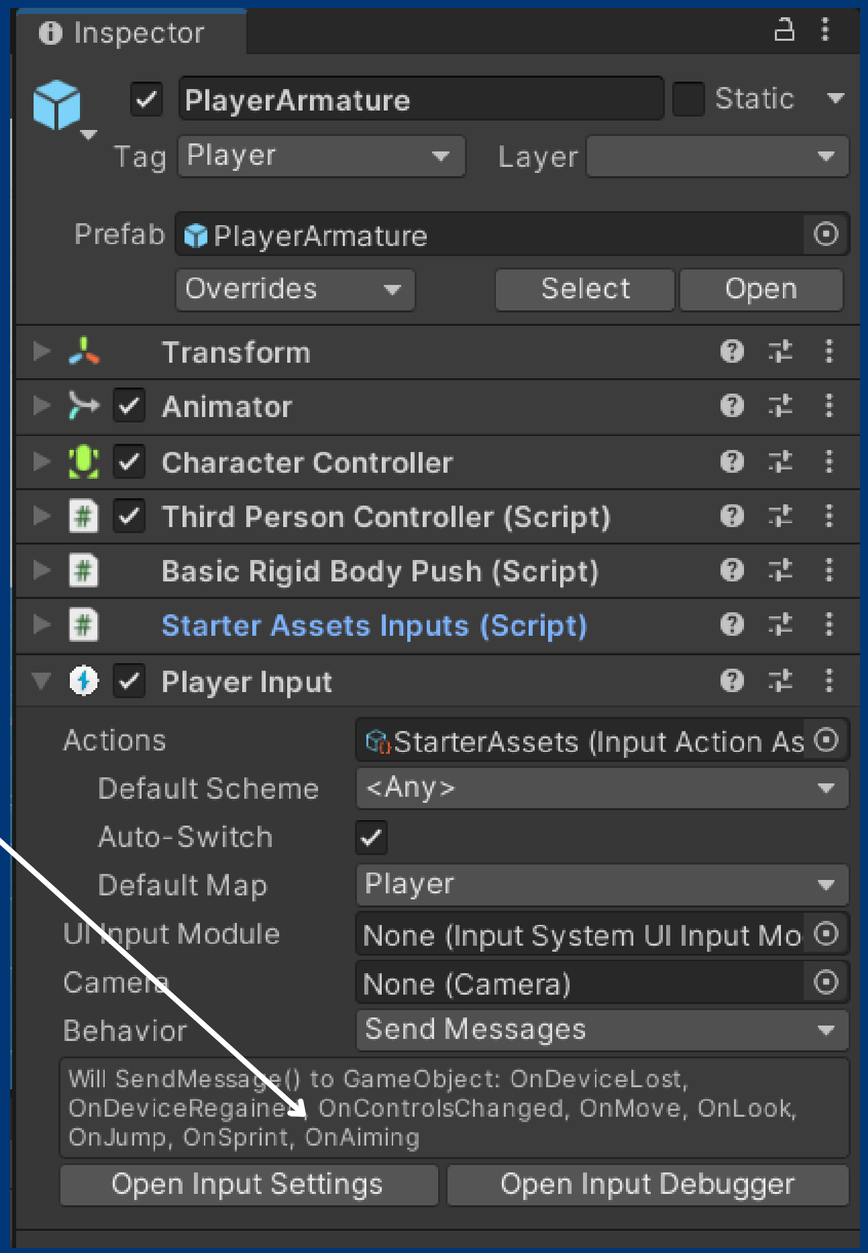
Linea 48

```
13          public bool jump;
            1 referencia
14          public bool sprint;
            0 referencias
15          public bool isAiming; //
16
17          [Header("Movement Settings")]
```

```
45              SprintInput(value.isPressed);
46          }
47
48          public void OnAiming(InputValue value)
49          {
50              isAiming = value.isPressed;
51          }
52
53
54
55  #endif
```

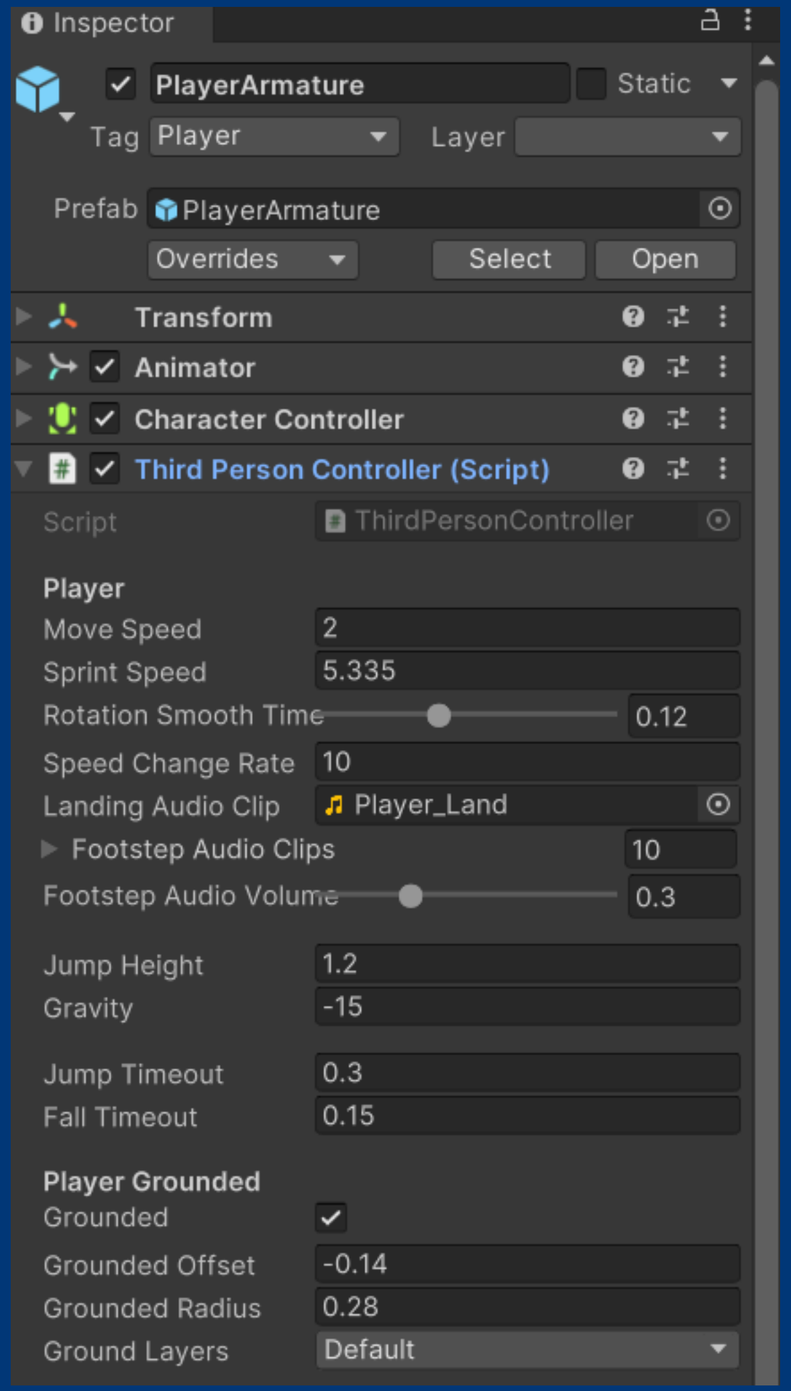
4

El componente Player Input cuenta ya con el metodo OnAiming



4

Buscamos el Script Third Person Controller

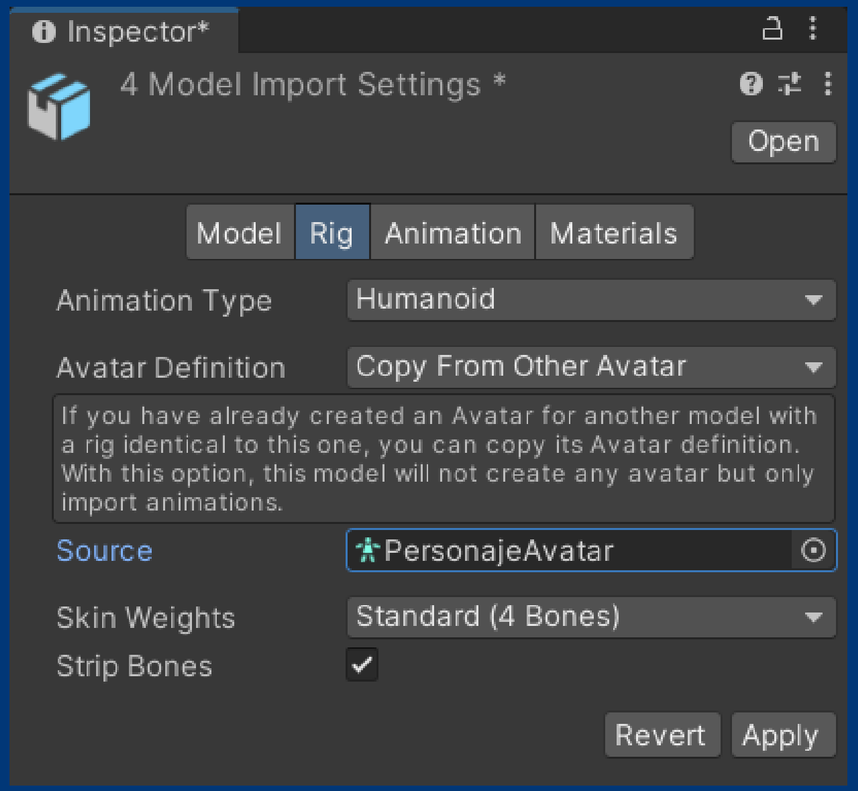


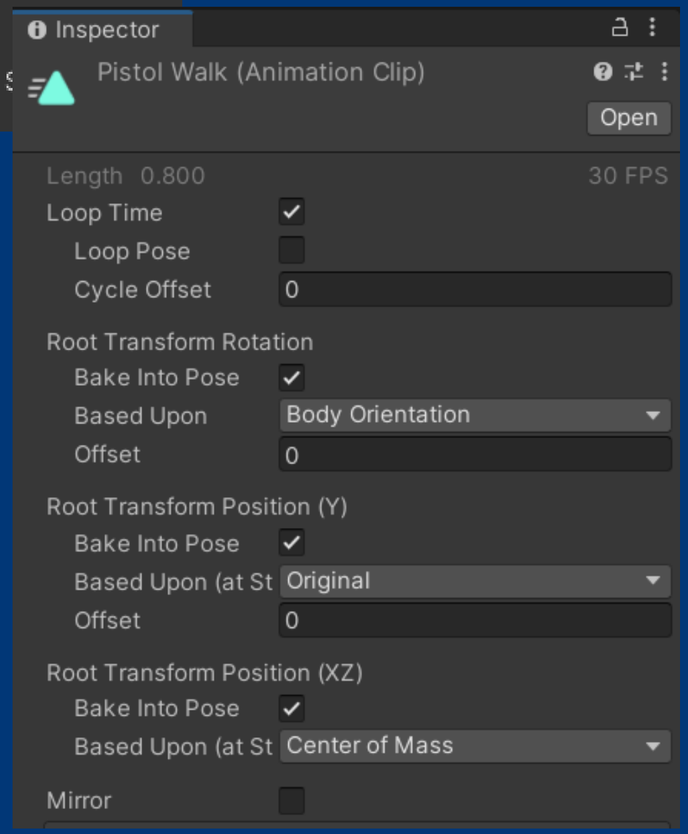
Abrimos

Se utiliza para implementar y gestionar el control de un personaje en tercera persona

4

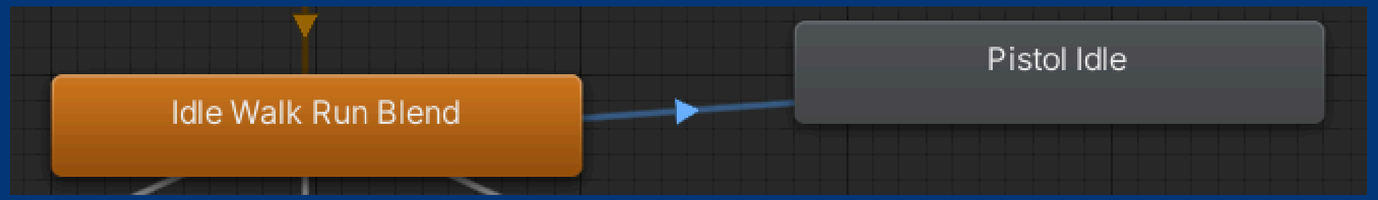
Preparo las animaciones

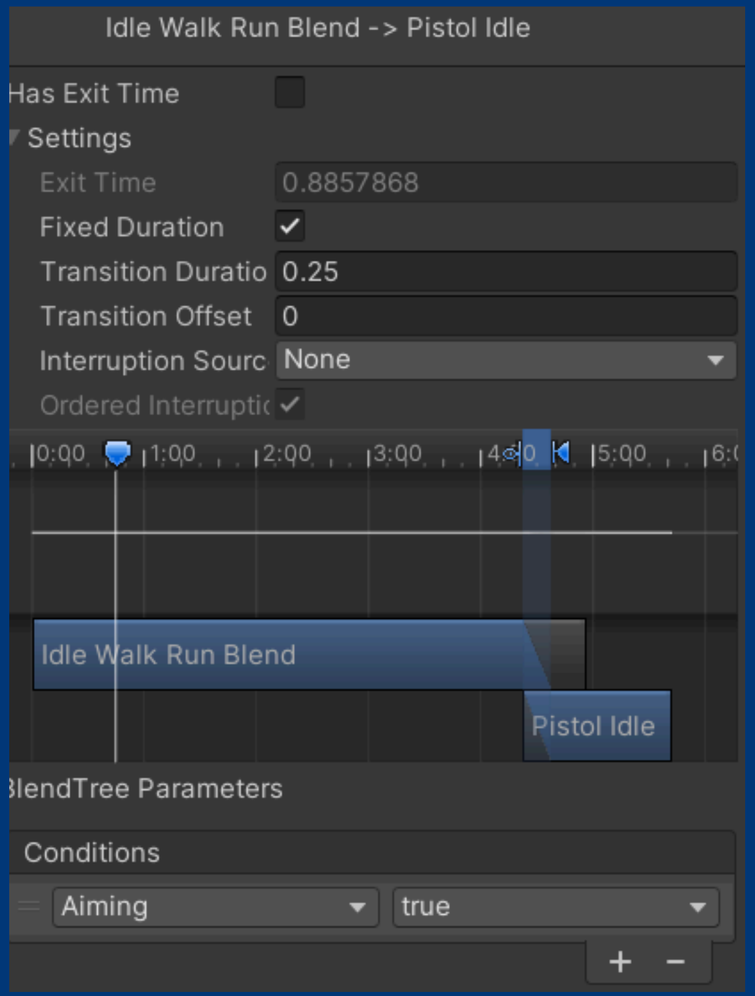




4

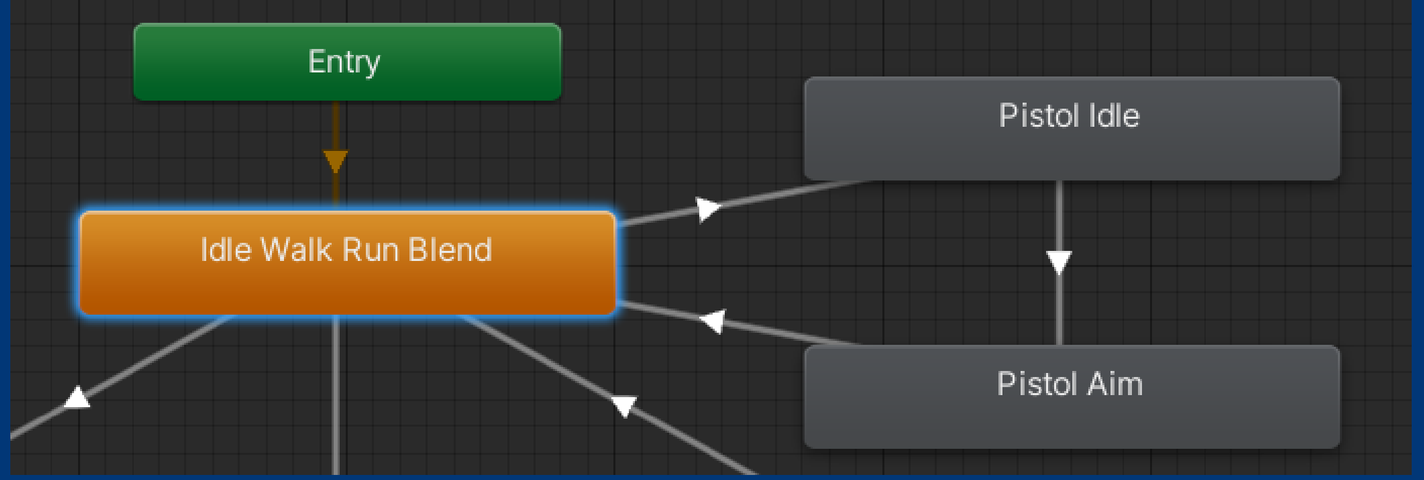
Agrego la animación de Pistol Idle

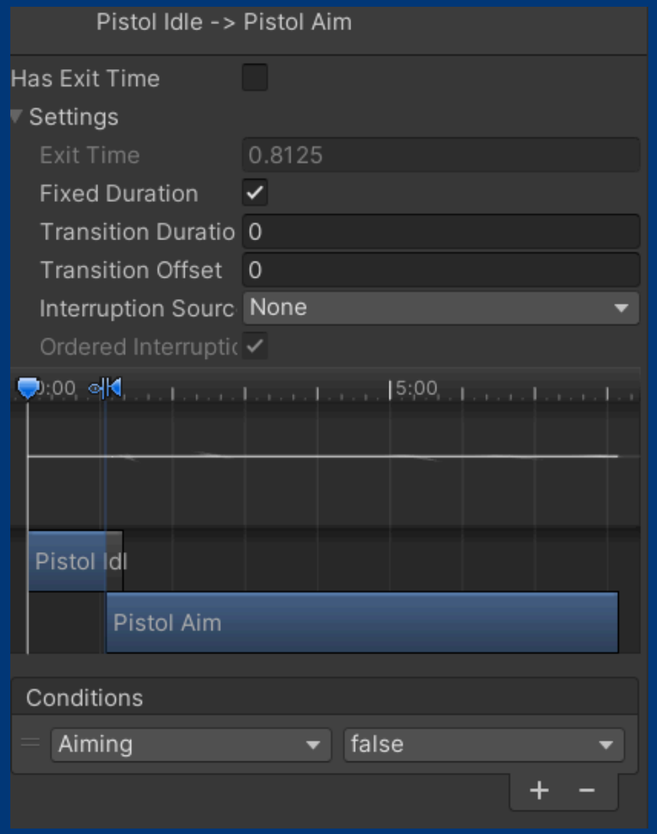


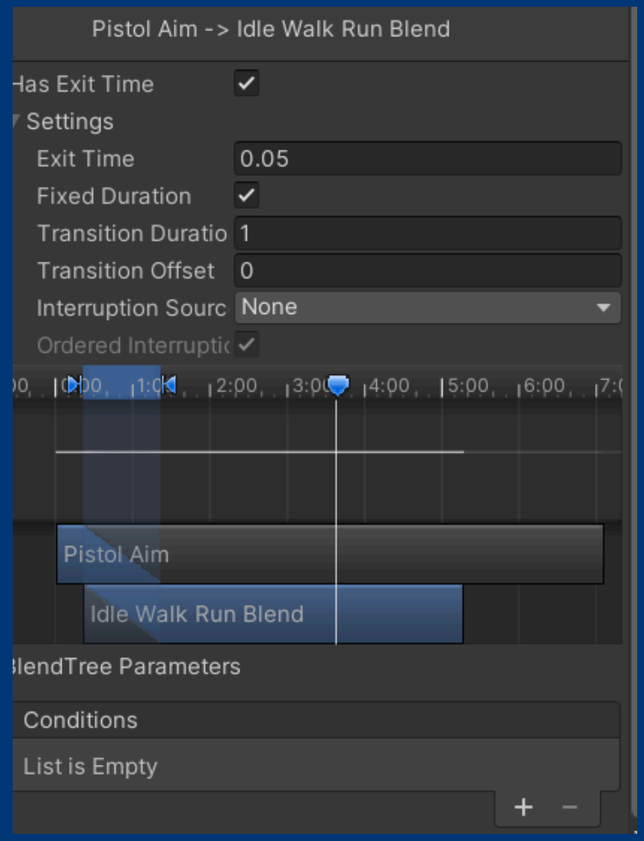


4

Agrego la Animacion de Pistol Aim







4

Codigo en script lineas 162 y 165

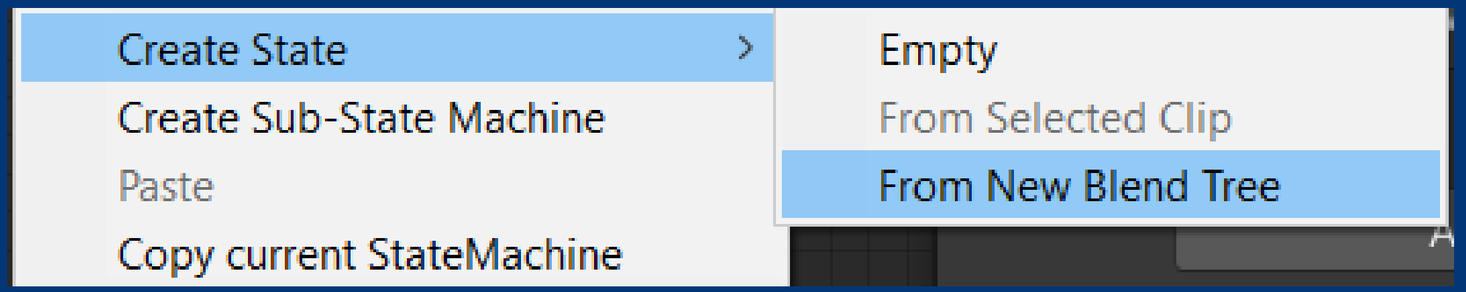
```
0 referencias
155     private void Update()
156     {
157         _hasAnimator = TryGetComponent(out _animator
158
159         JumpAndGravity();
160         GroundedCheck();
161         Move();
162         AimShoot();
163     }
```

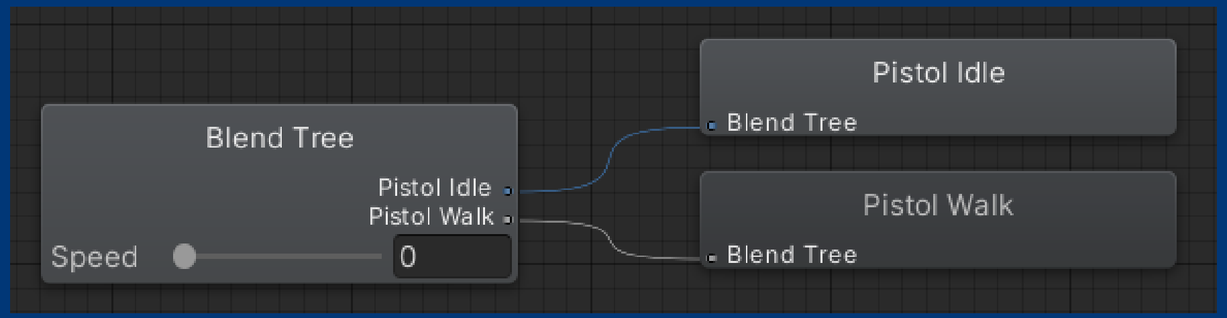
```
165     private void AimShoot()
166     {
167         if(_input.isAiming && Grounded && !_input.sp
168         {
169             _animator.SetBool("Aiming",_input.isAimi
170         }
171         else
172         {
173             _animator.SetBool("Aiming",false);
174         }
175     }
176
```

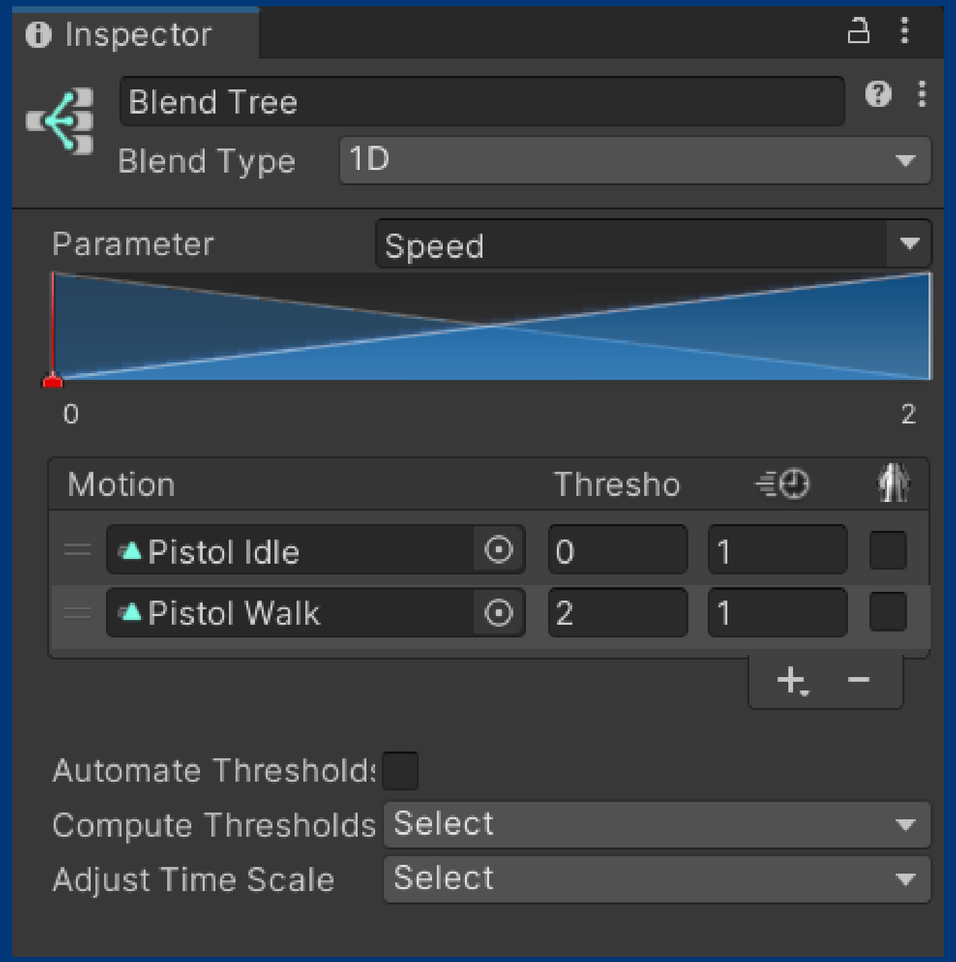
Probamos

4

Creo un Blend Tree

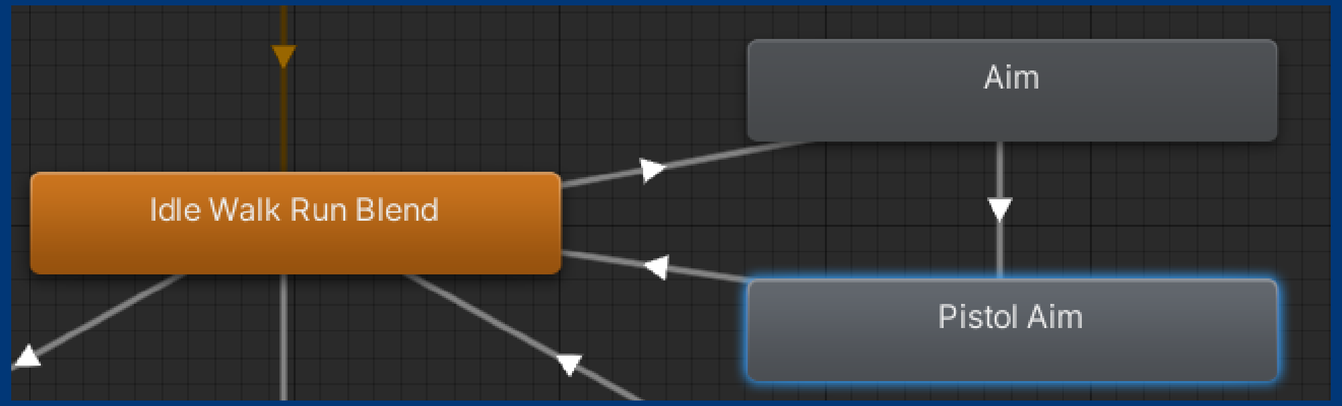


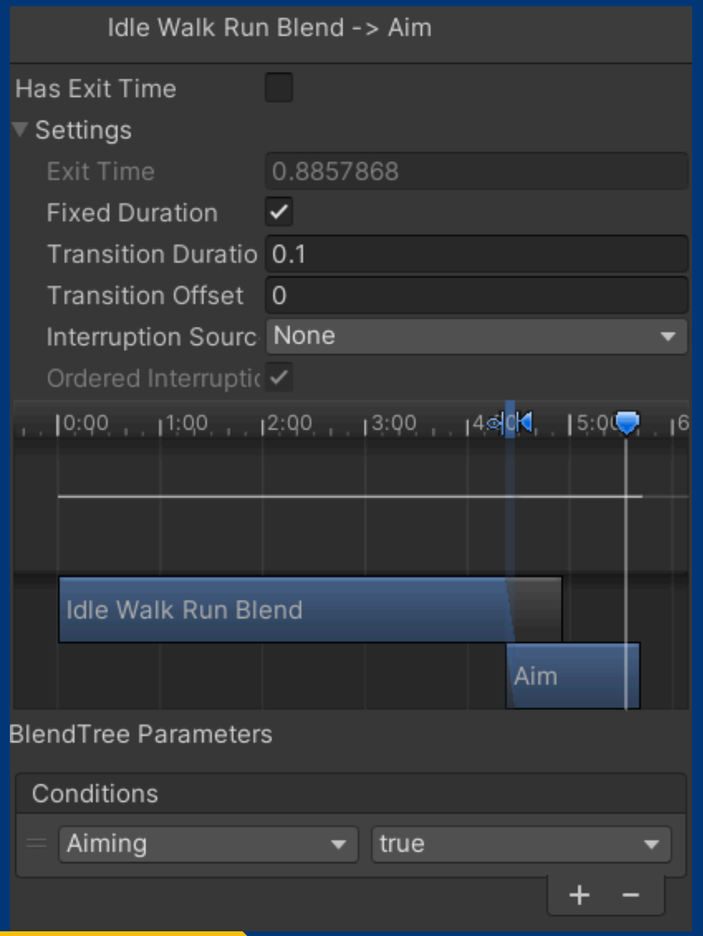


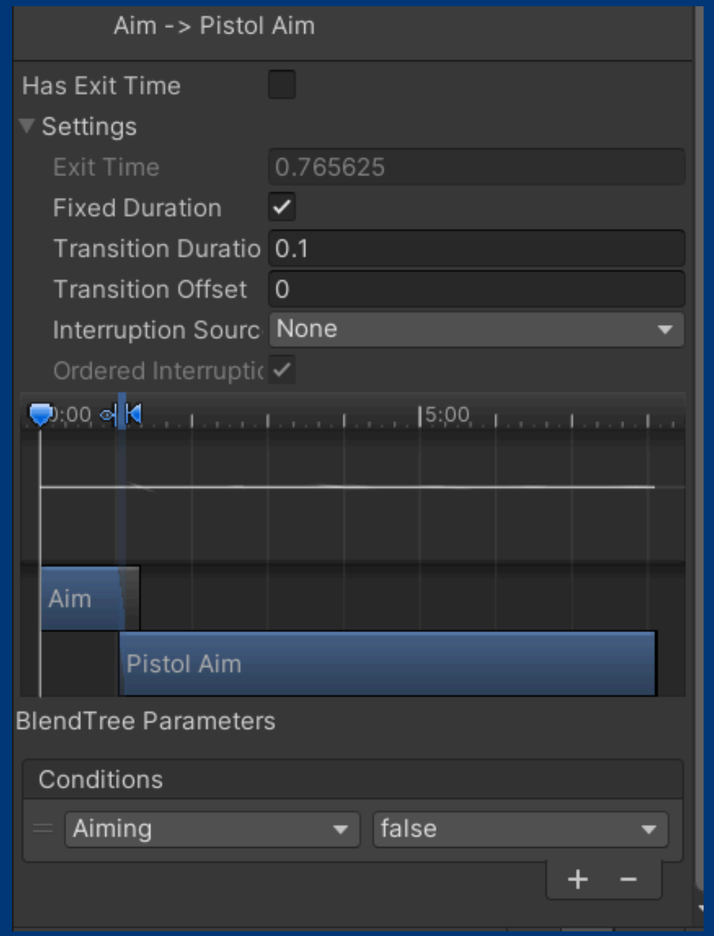


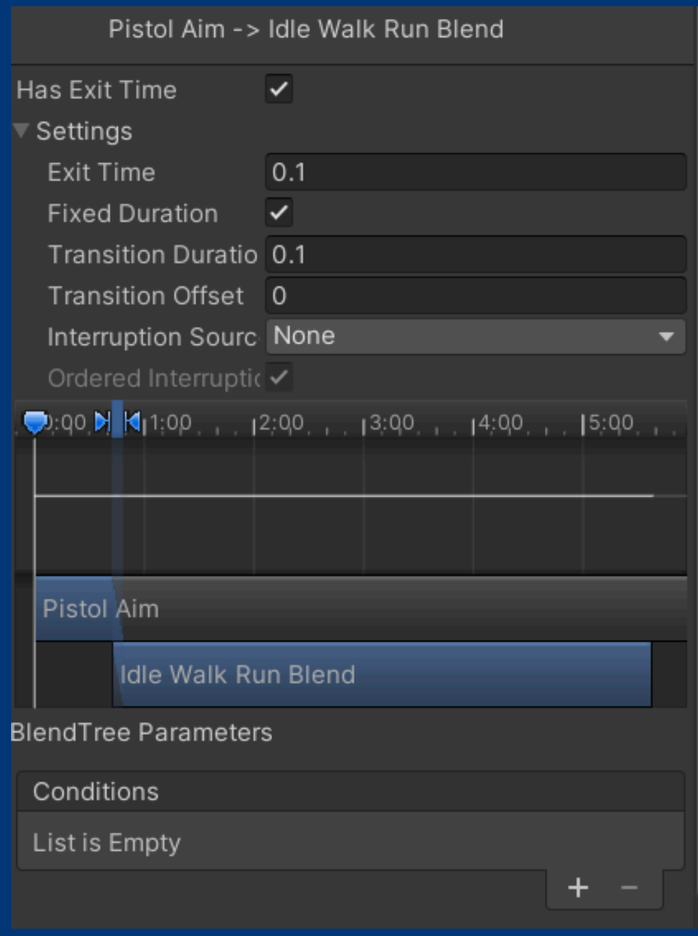
4

Realizo las nuevas transiciones



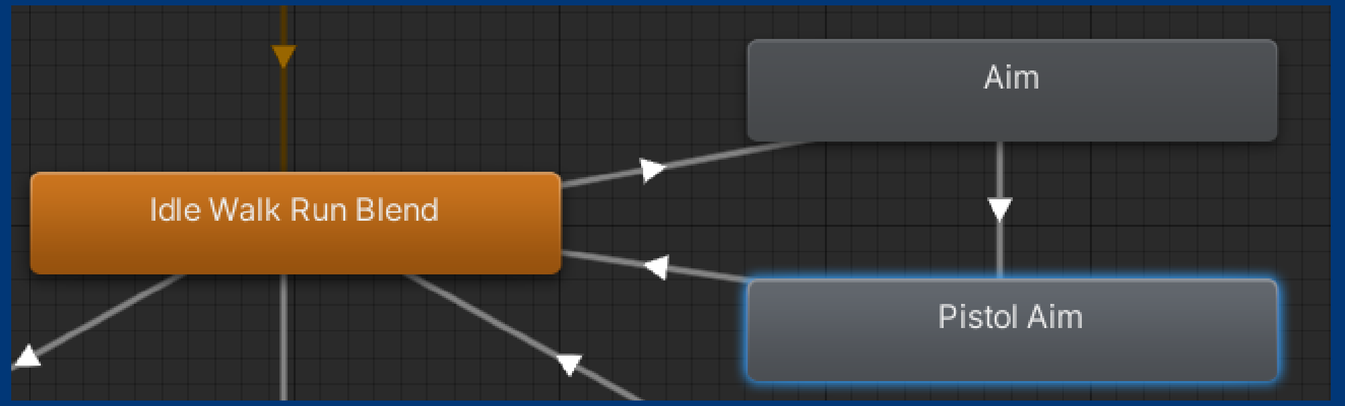


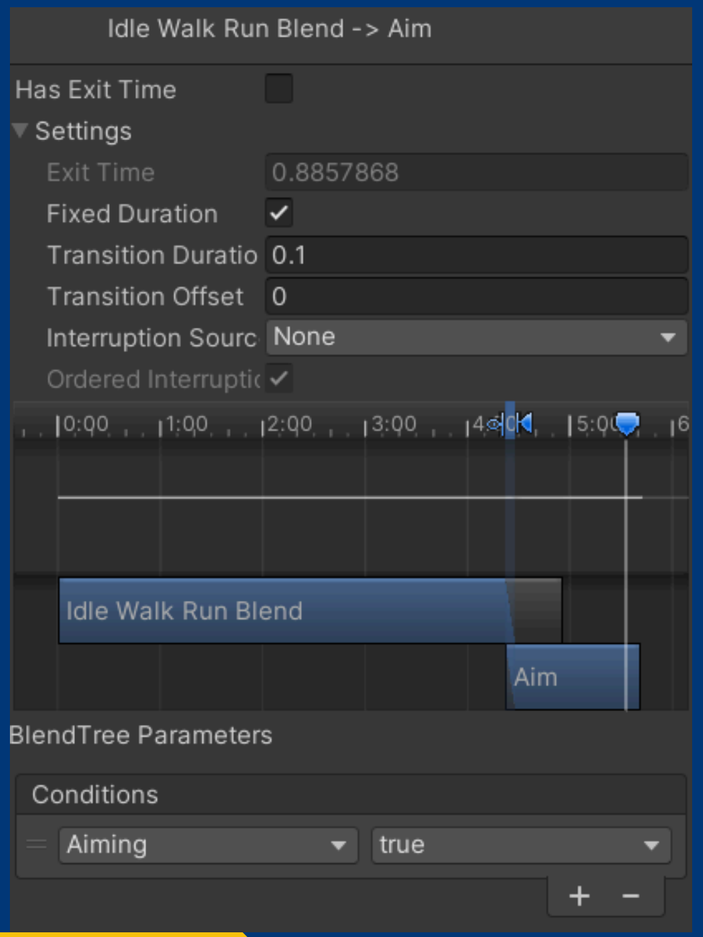


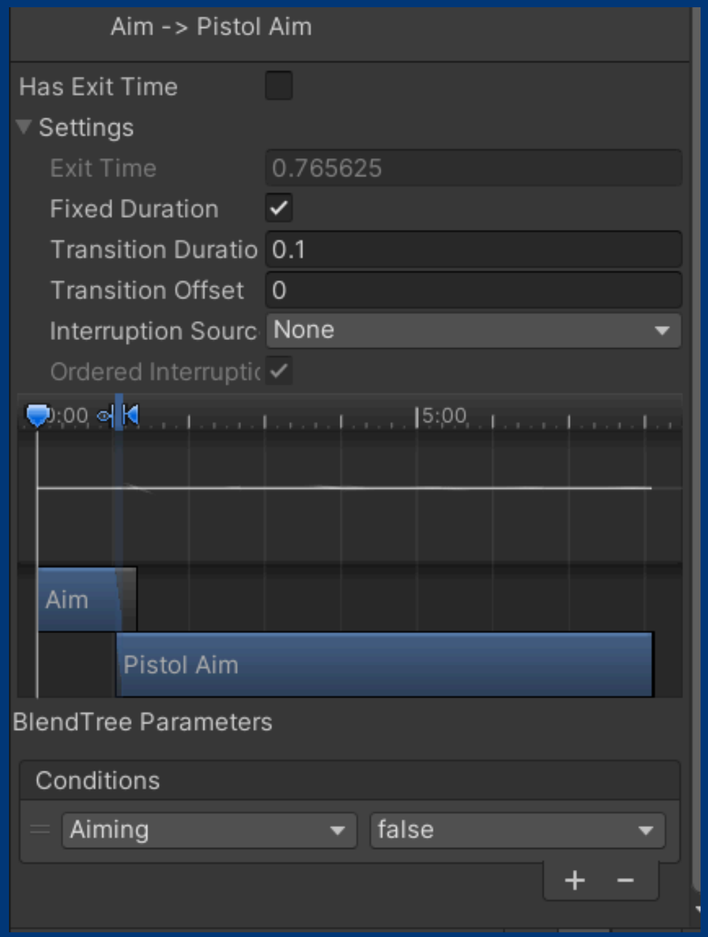


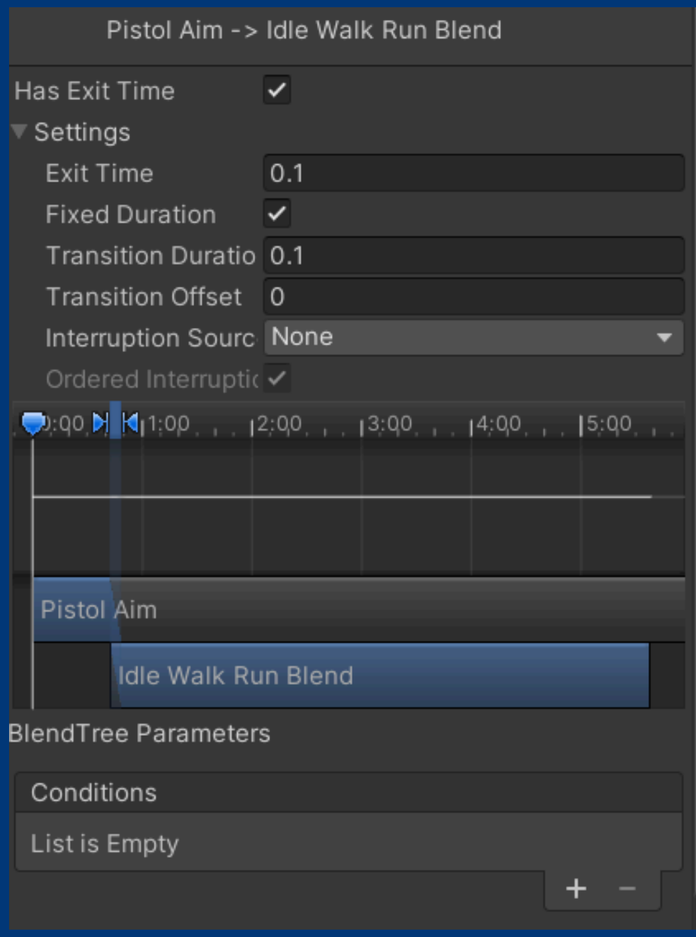
4

Realizo las nuevas transiciones







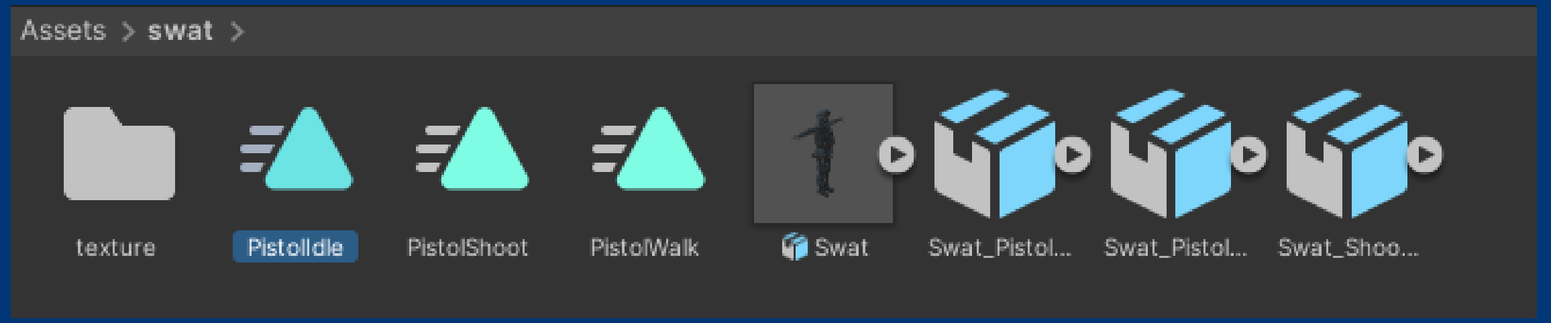


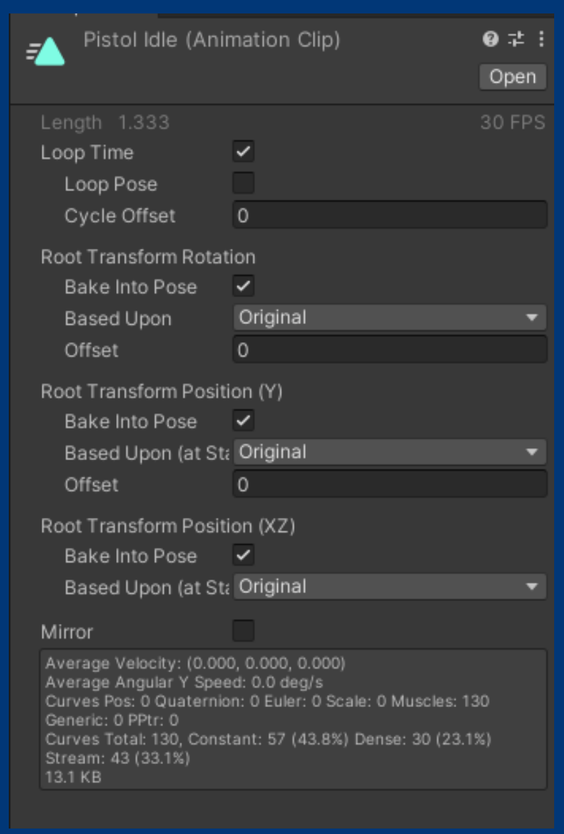
4

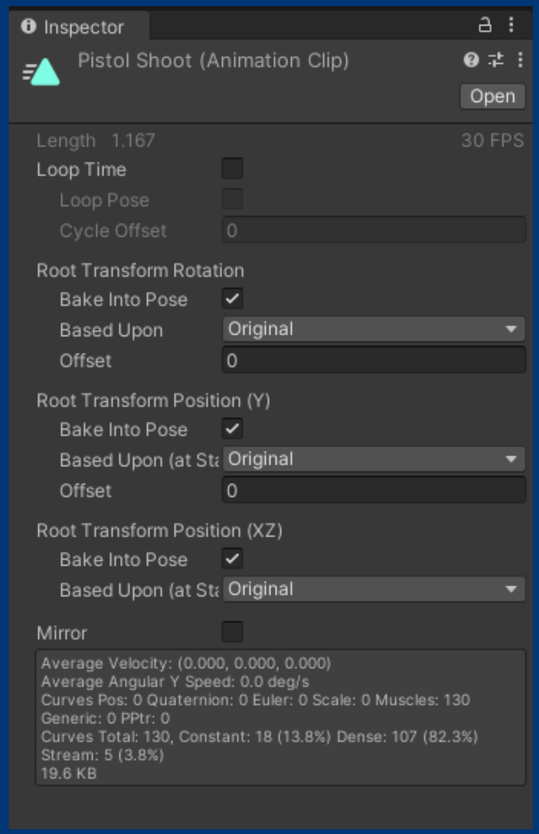
Solución Animación Shoot

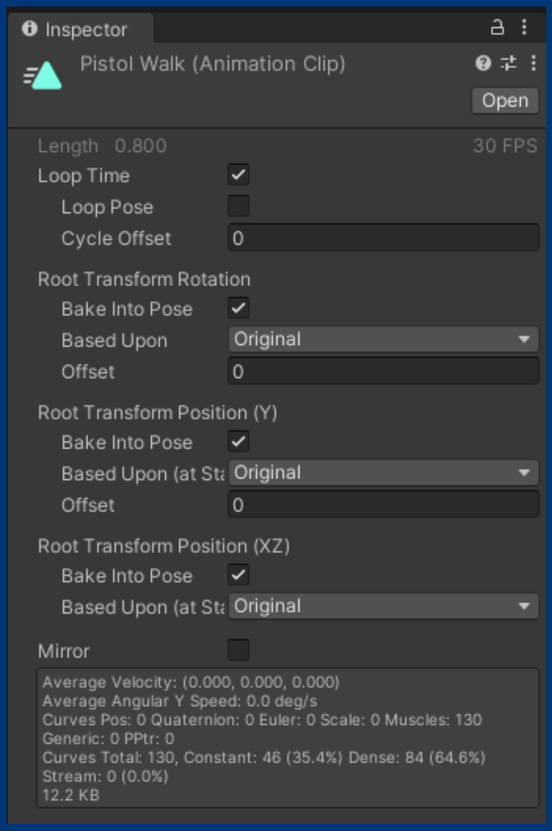
4

Animaciones



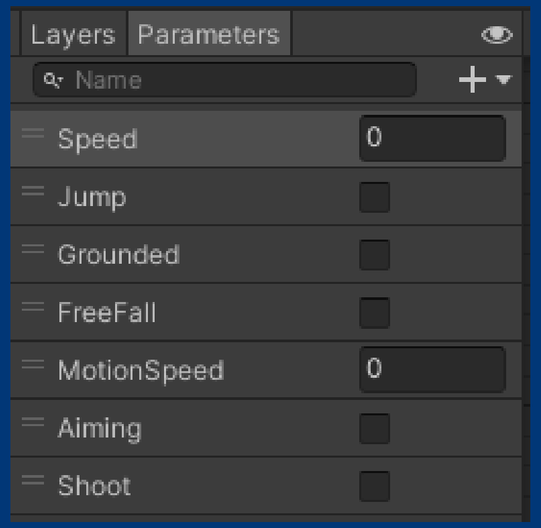


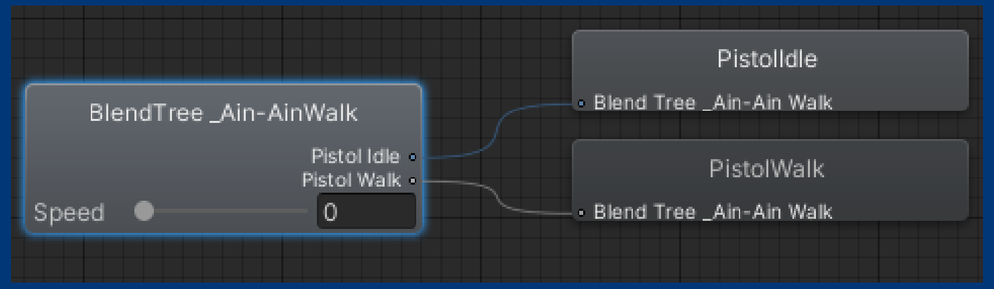


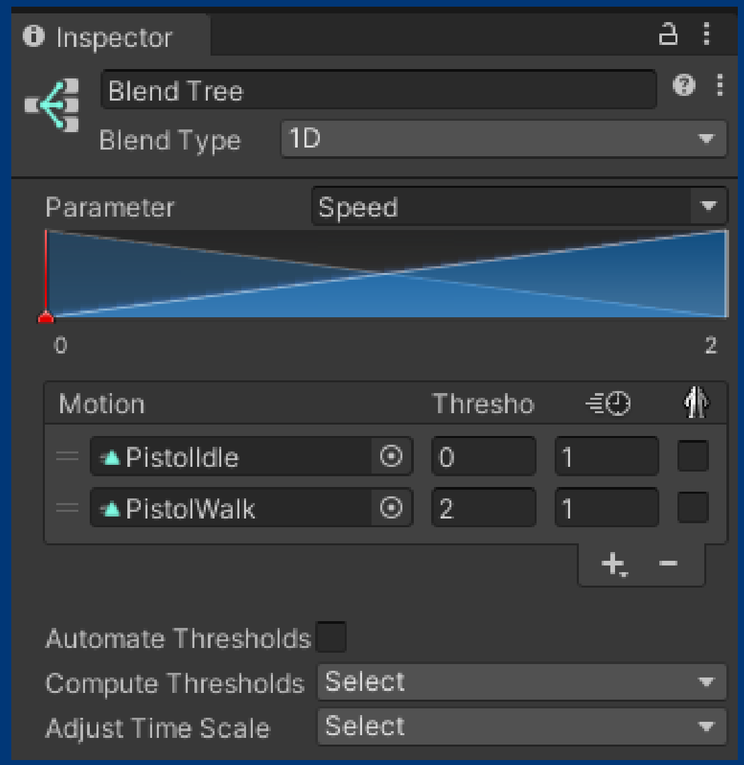


4

Animaciones

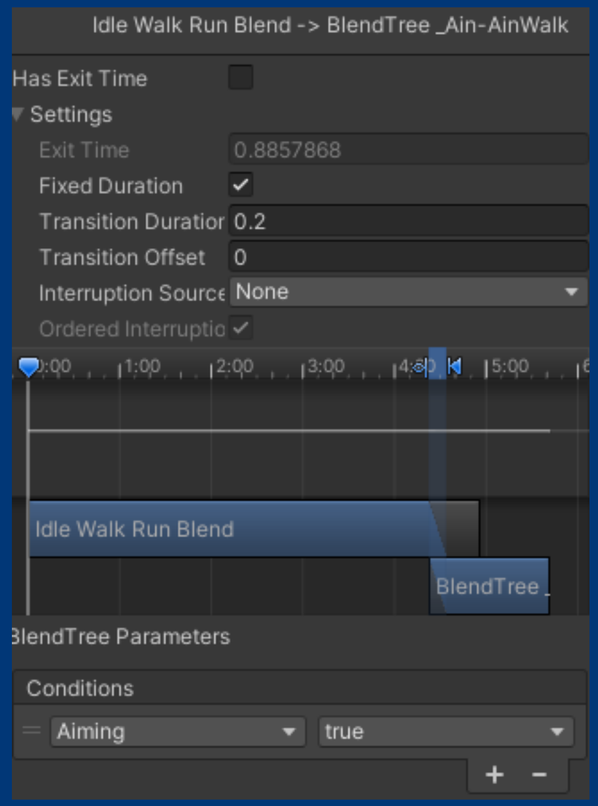


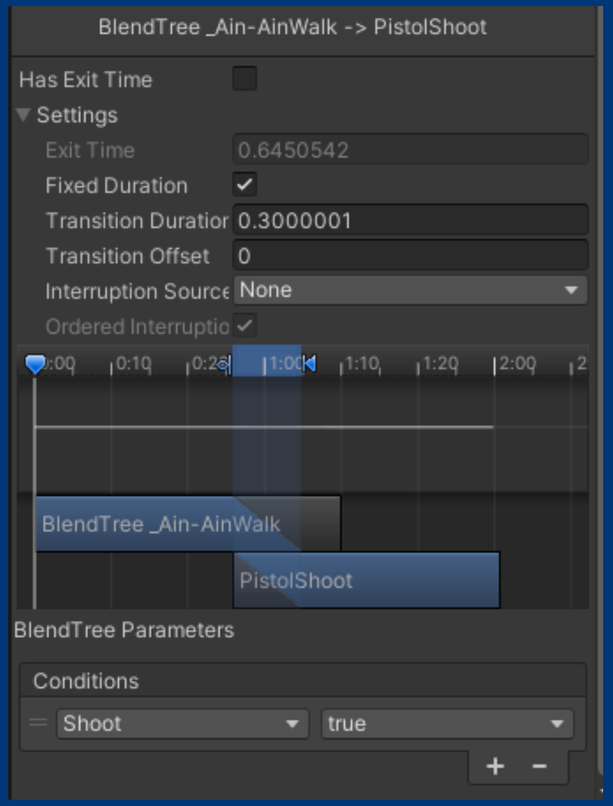


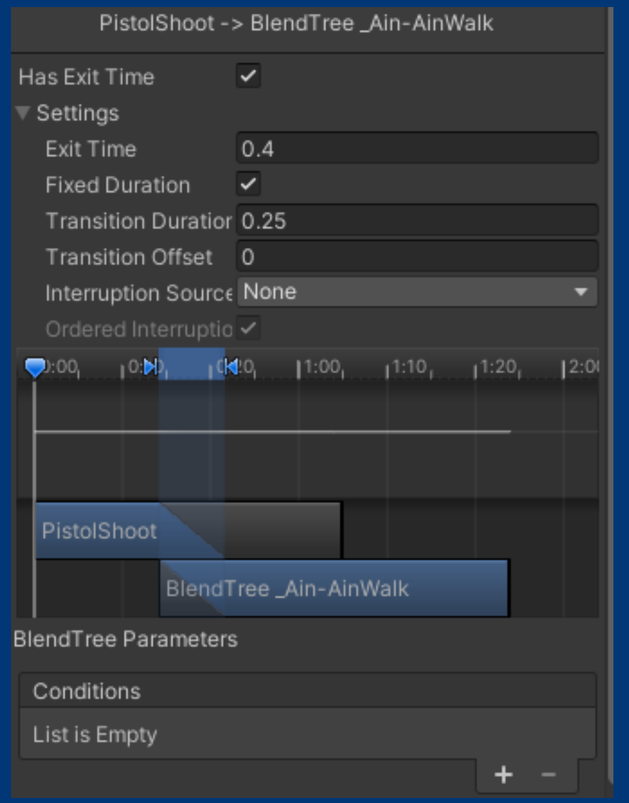


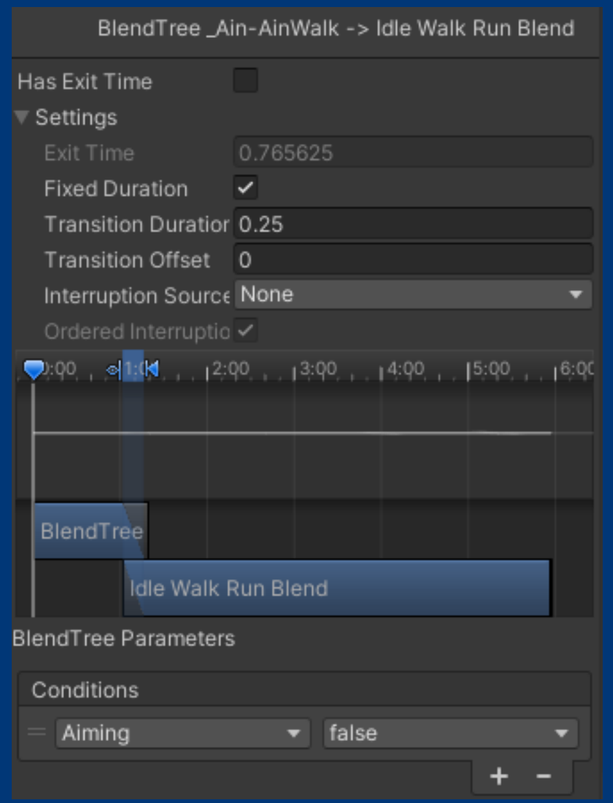
4

Animaciones



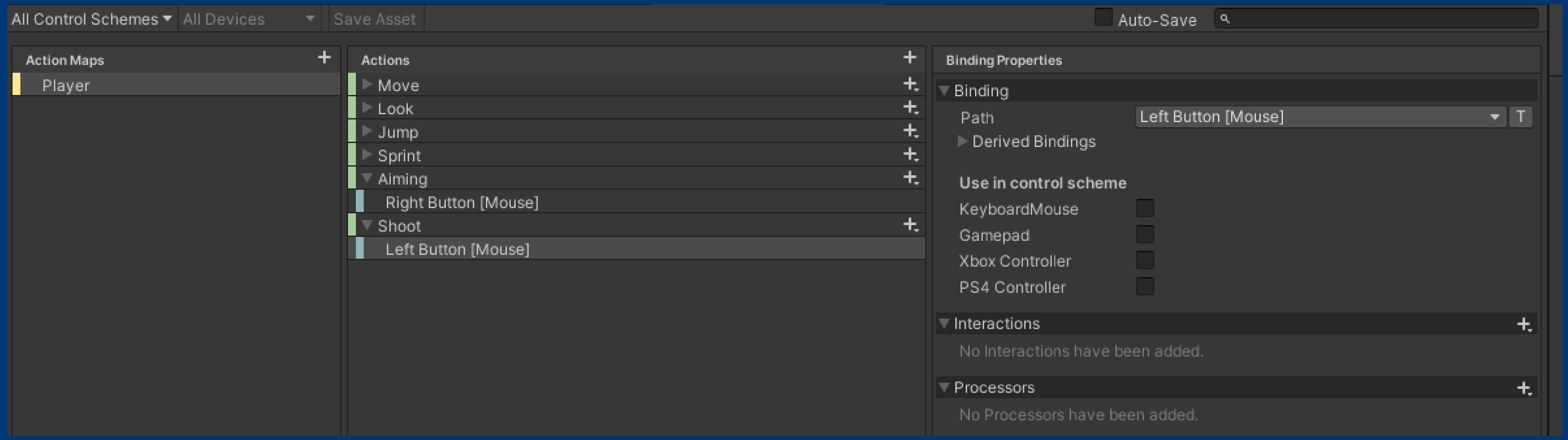






4

Input Action



4

Codigo

StarterAssetsInputs.cs

```
15          public bool isAiming;
            0 references
16          public bool isShoot;
```

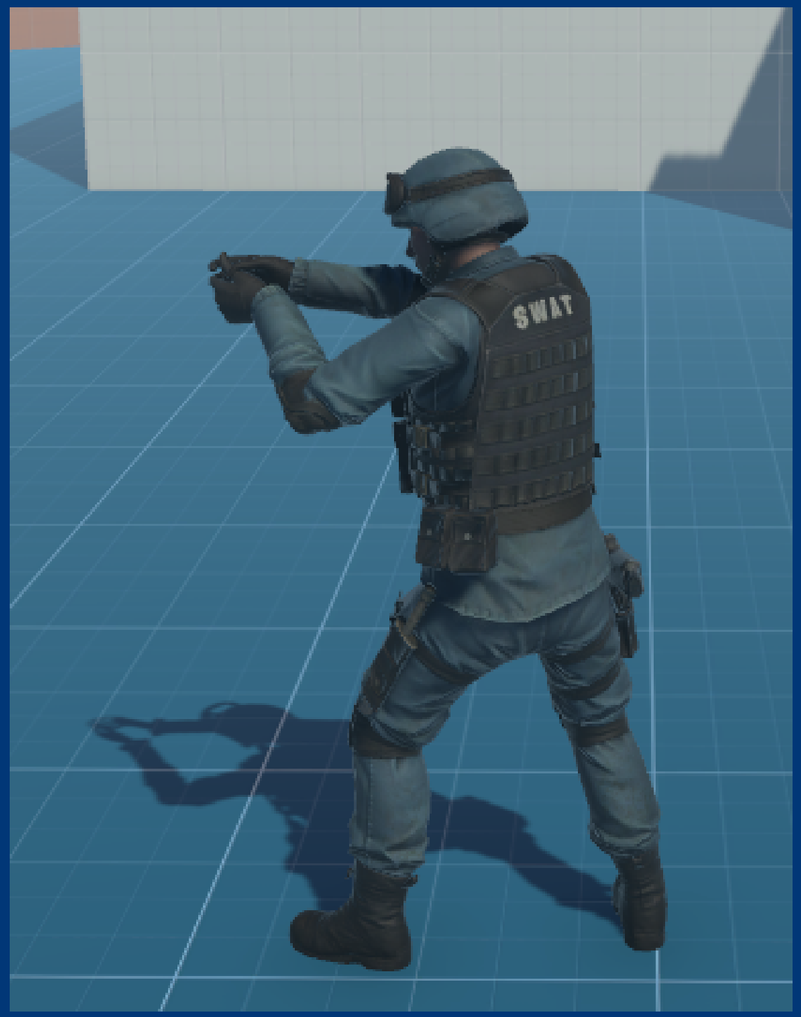
```
48
49          public void OnAiming(InputValue value)
50          {
51              isAiming = value.isPressed;
52          }
53
54          public void OnShoot(InputValue value)
55          {
56              isShoot = value.isPressed;
57          }
58
59
60      #endif
```

ThirdPersonControlle.cs

```
167         private void AimShoot()
168         {
169             if(_input.isAiming && Grounded && !_input.sprint)
170             {
171                 _animator.SetBool("Aiming",_input.isAiming);
172             }
173             else
174             {
175                 _animator.SetBool("Aiming",false);
176             }
177         }
178
            1 reference
179         private void Shoot()
180         {
181             if(_input.isShoot && Grounded && !_input.sprint && _input.isAiming && _animationBlend<0.1)
182             {
183                 _animator.SetBool("Shoot",_input.isShoot);
184             }
185             else
186             {
187                 _animator.SetBool("Shoot",false);
188             }
189         }
```

4

Prueba

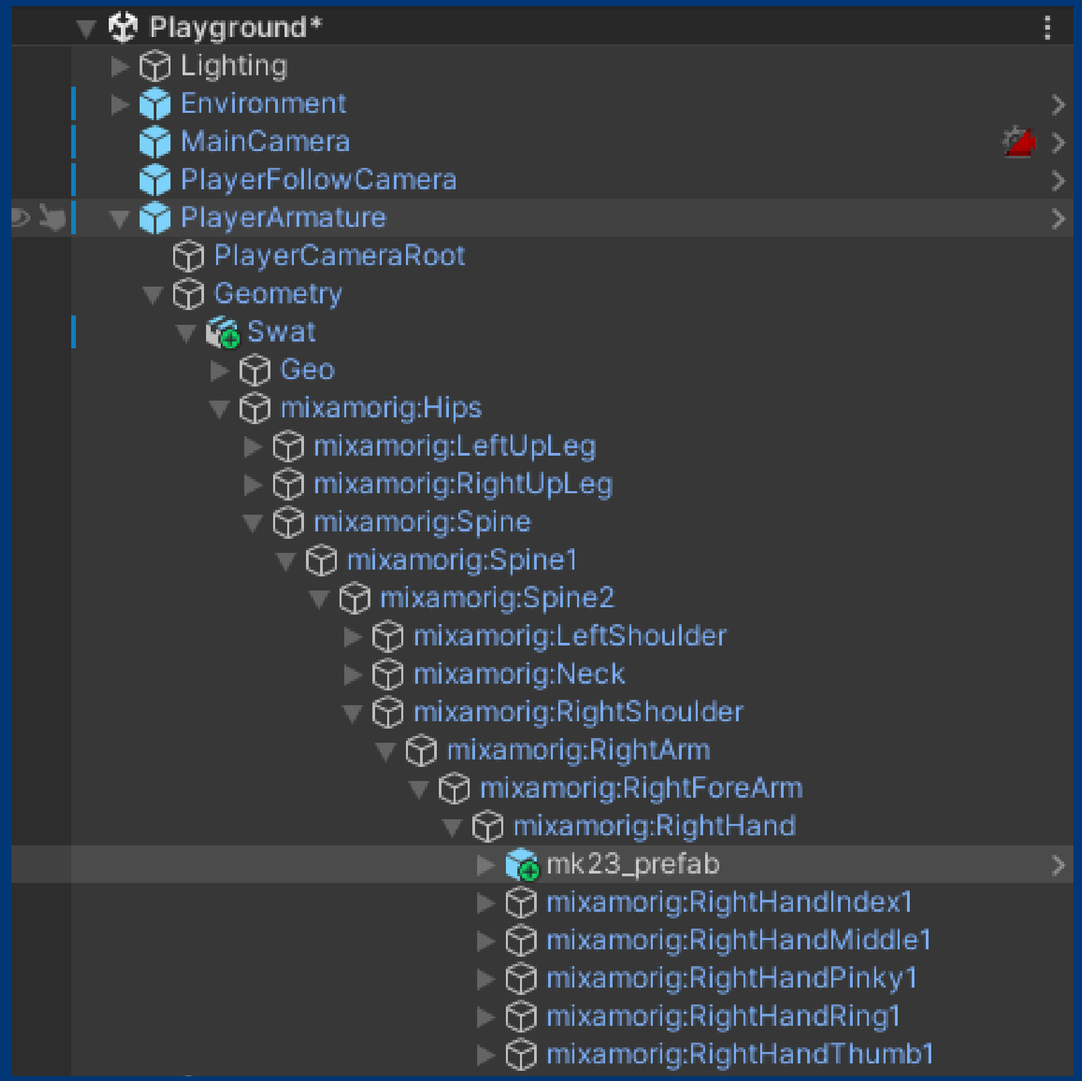


4

Agregar Objetos

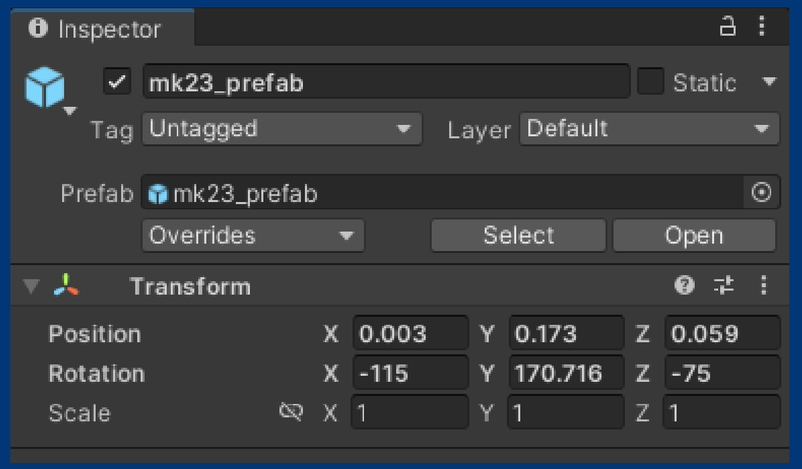
4

Objeto “MK23” pistola en la mano derecha del personaje



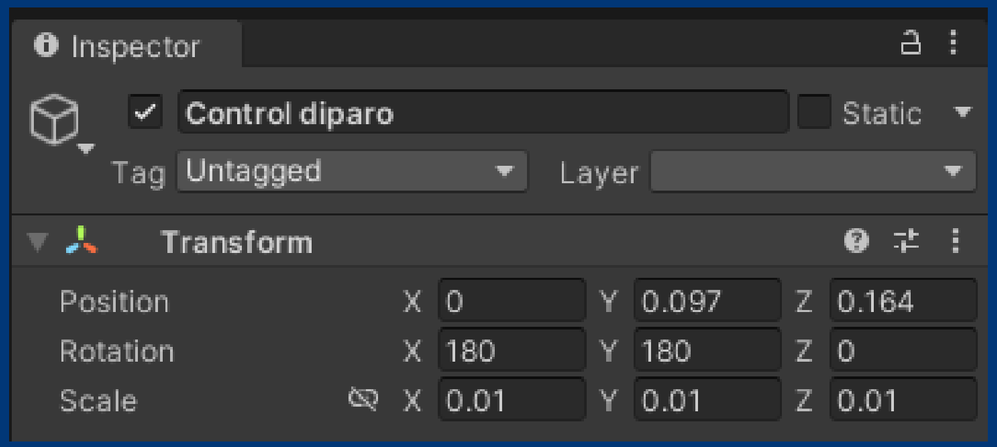
4

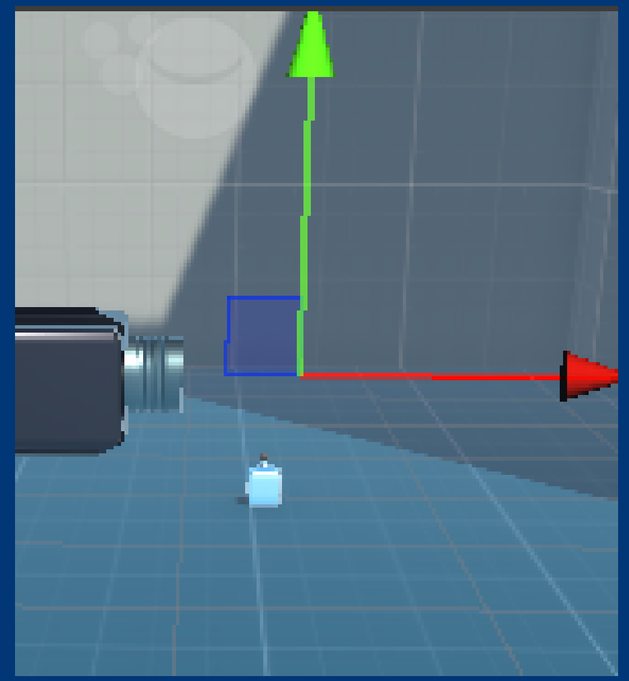
Posiciono objeto



4

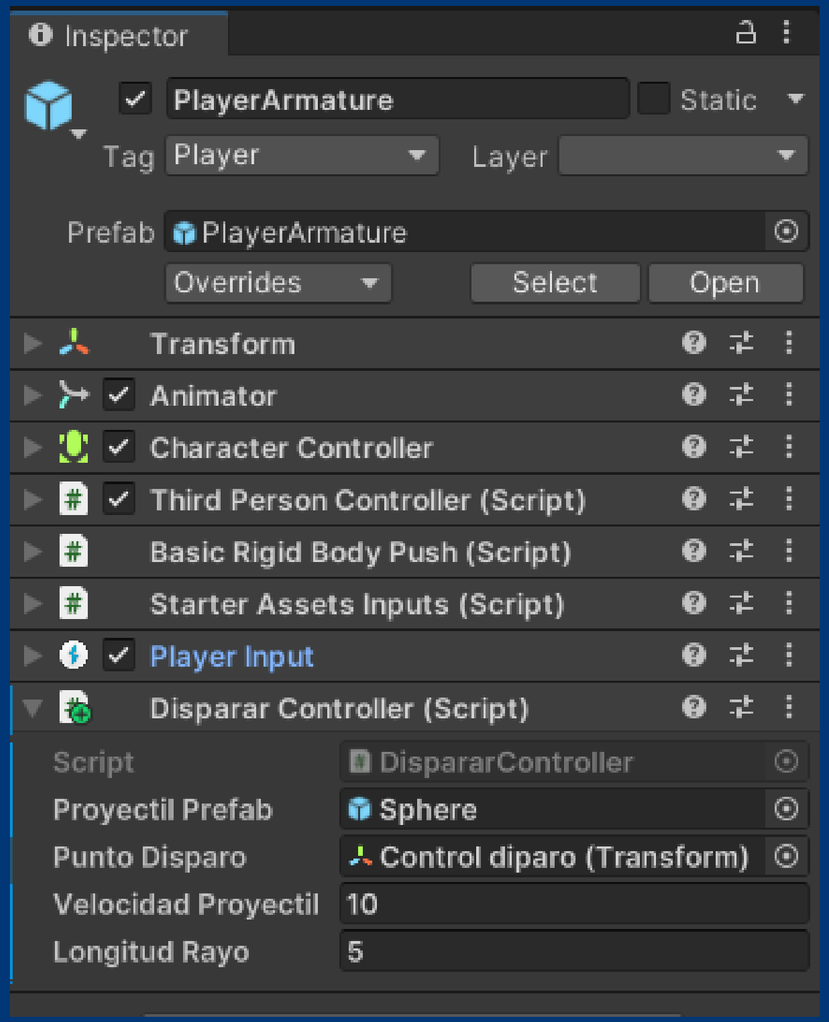
Objeto vacío en pistola





4

Script #2 para disparar DispararController.cs

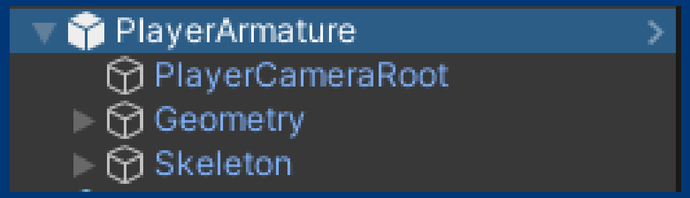


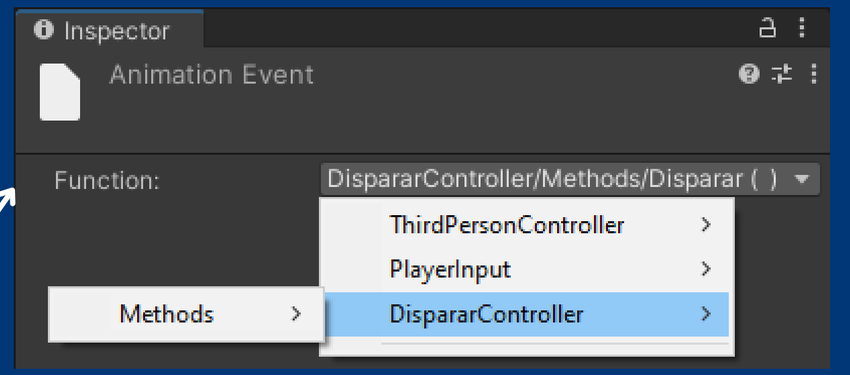
Agrego el script al Player Armature

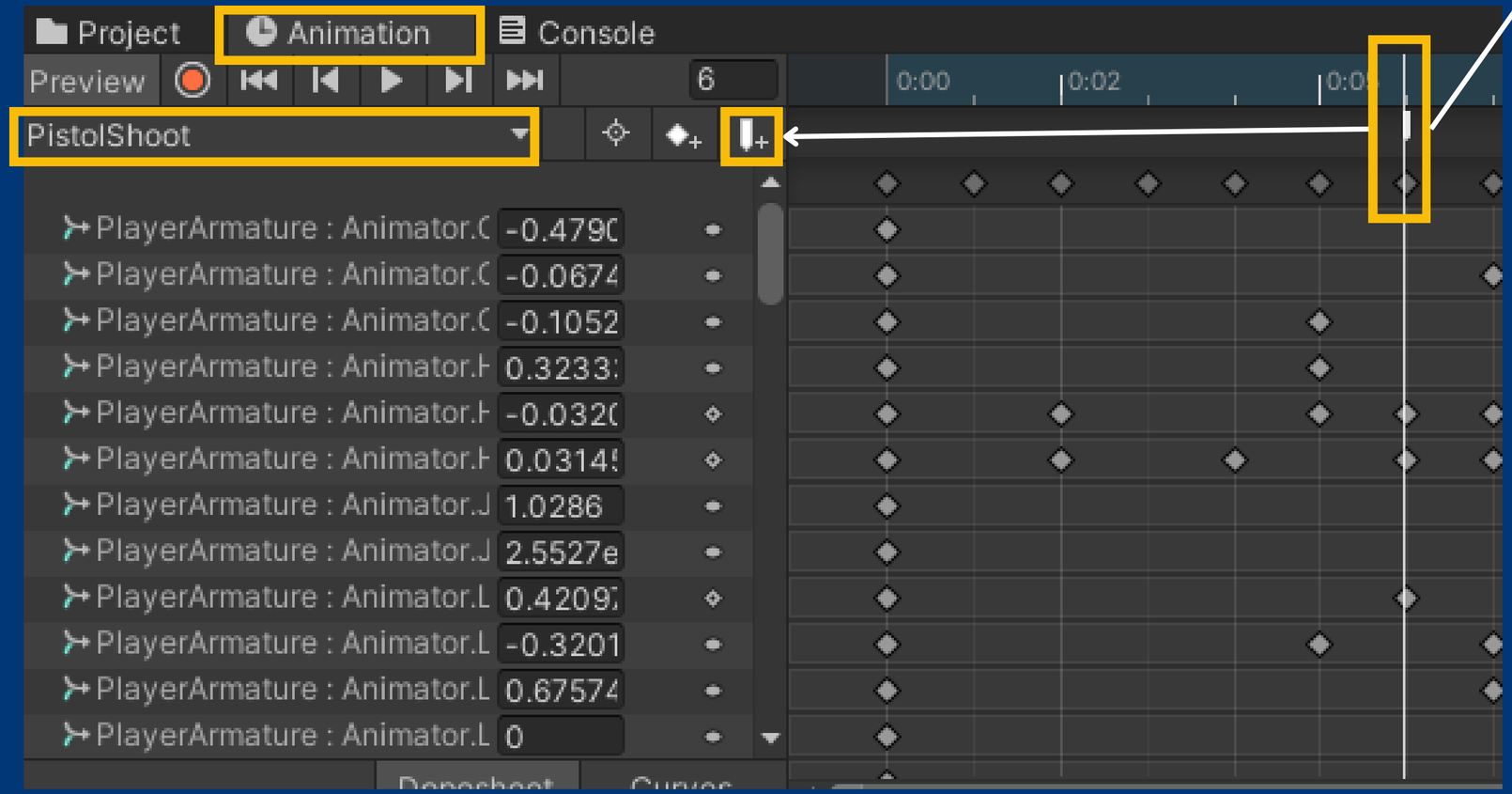
- Agrego el prefab que sera la bala
- Agrego el controlador disparo

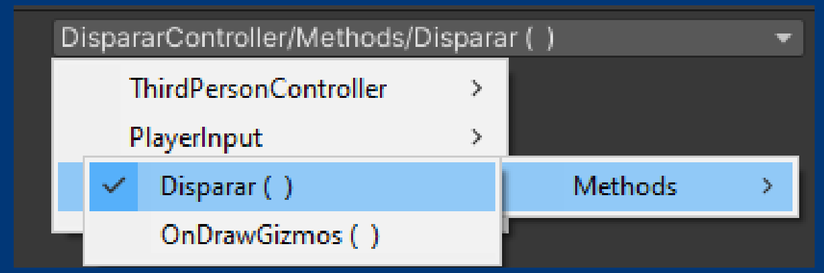
4

Selecciono el PayerArmadure



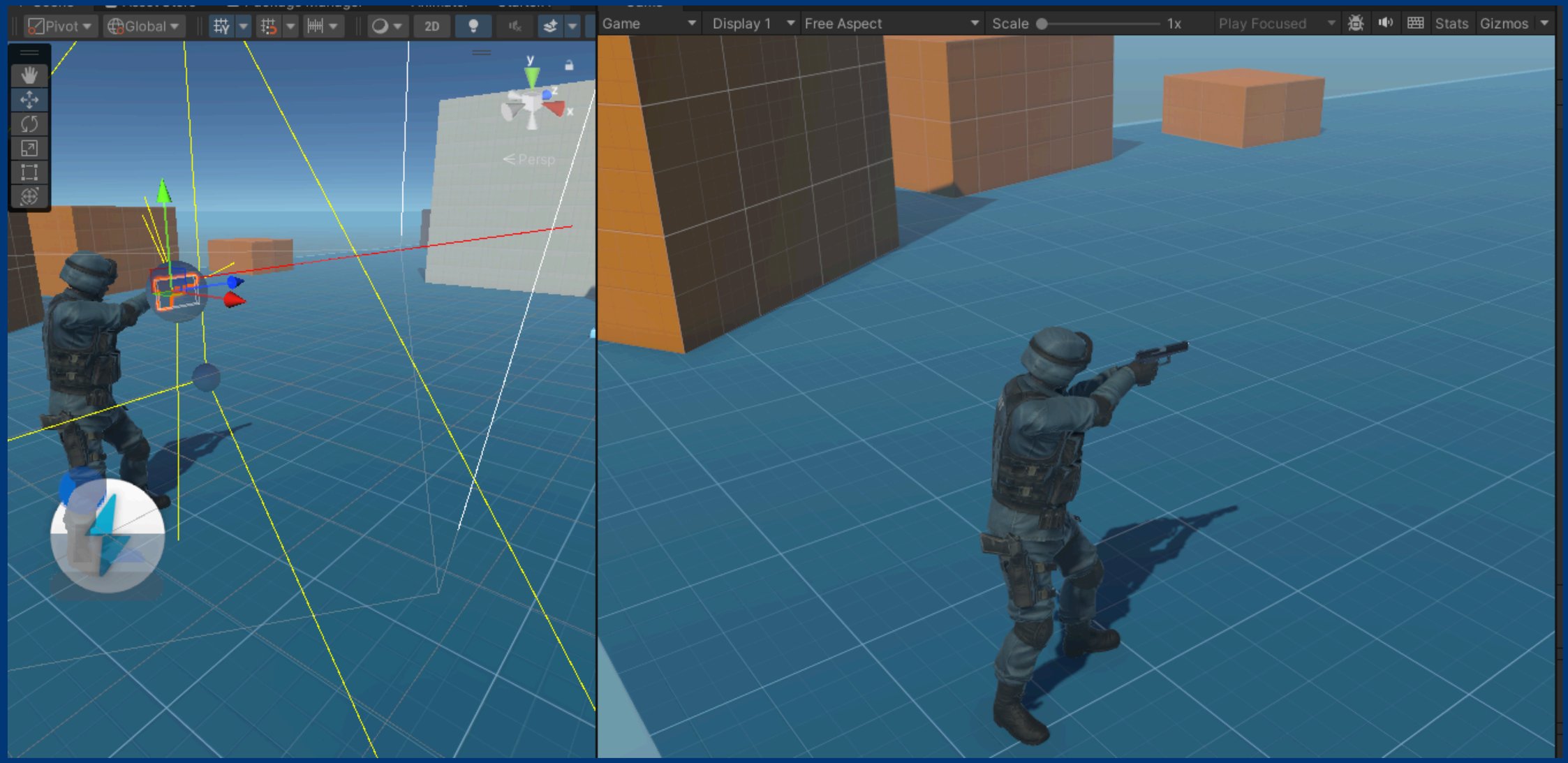






4

Pruebo

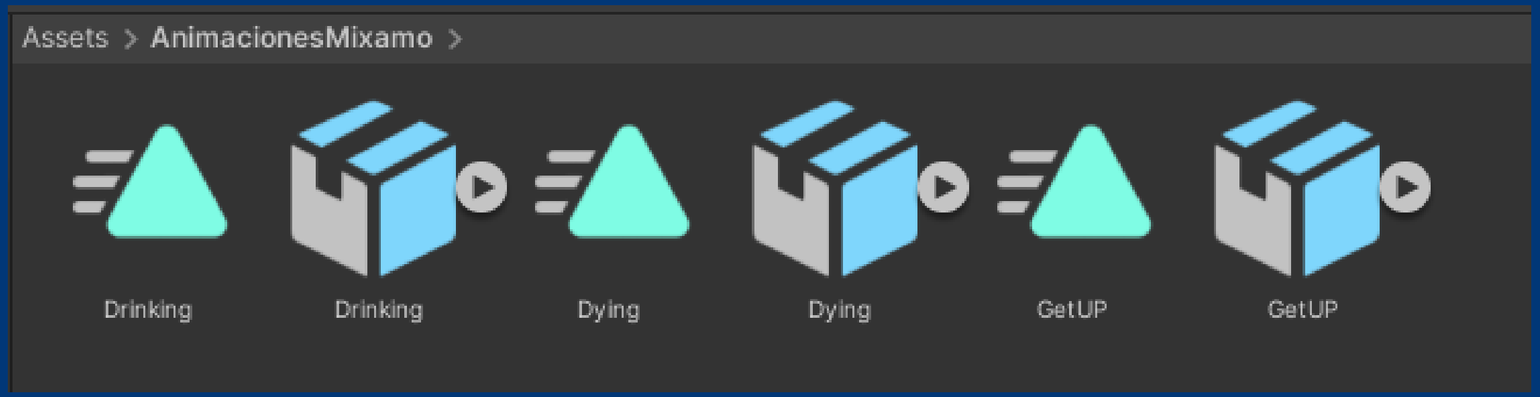

Pivot
Global
2D
Persp
Game
Display 1
Free Aspect
Scale
1x
Play Focused
Stats
Gizmos

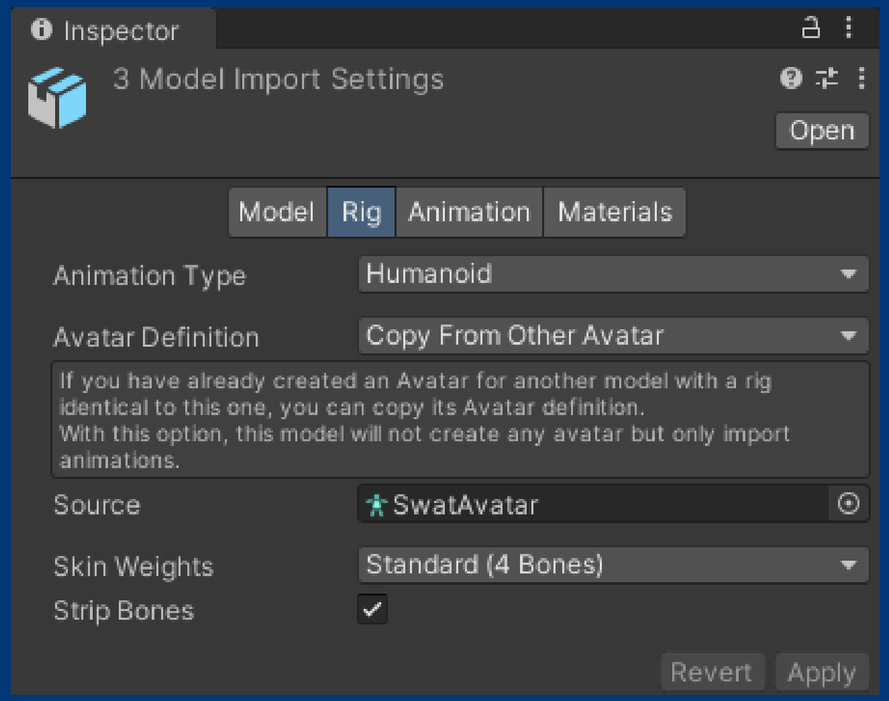
4

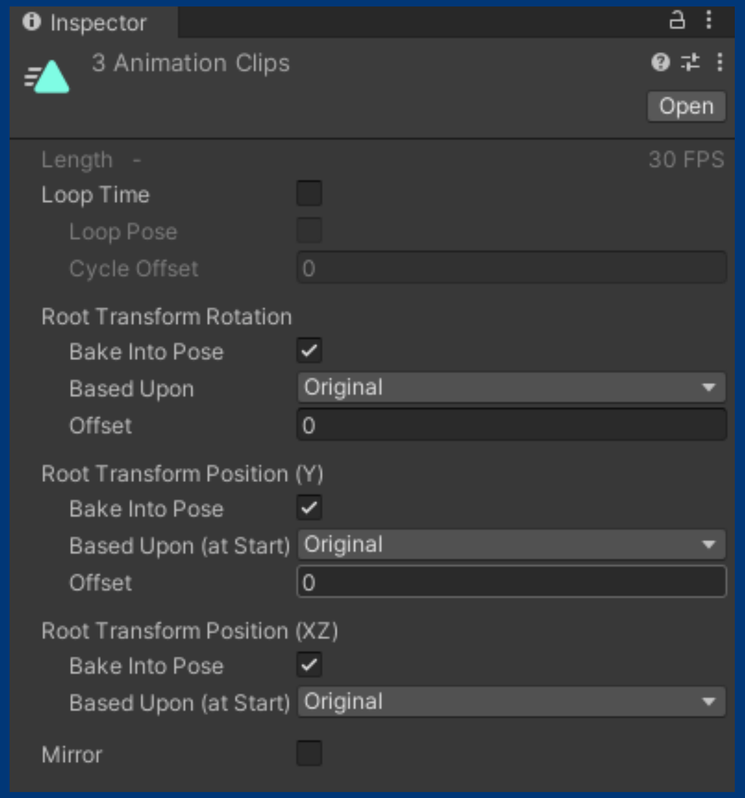
Tomar un objeto del piso

4

Preparo animaciones

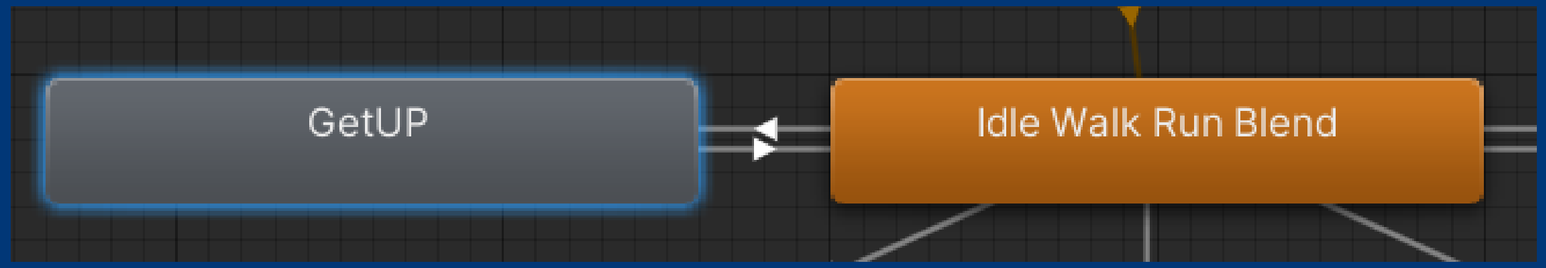


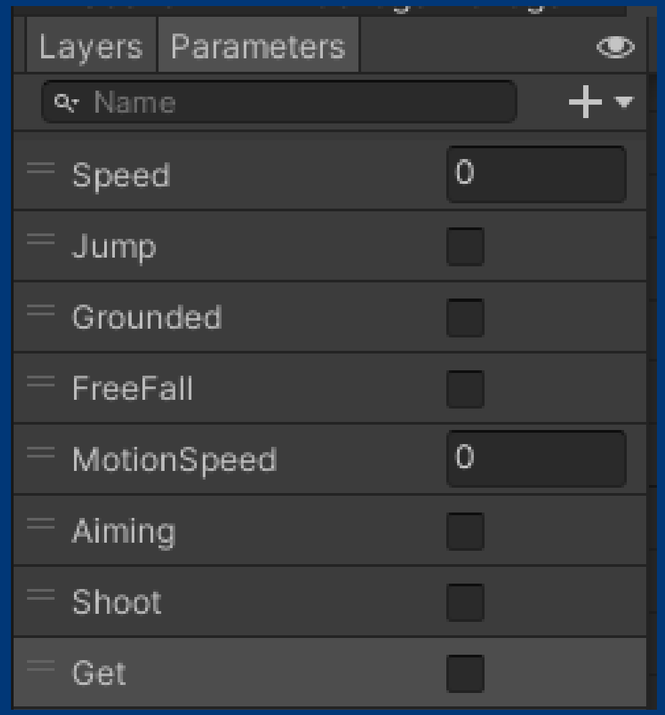


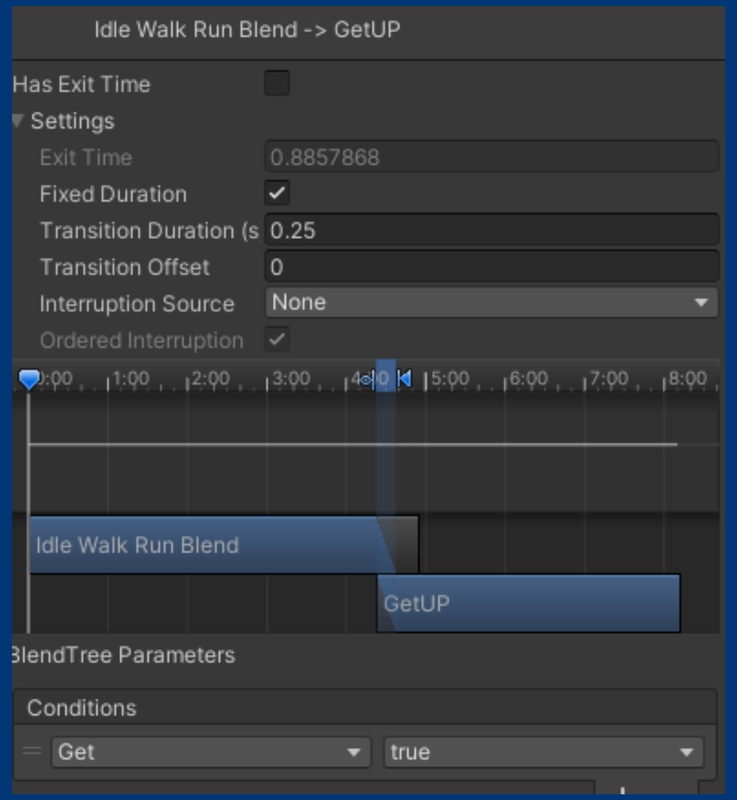


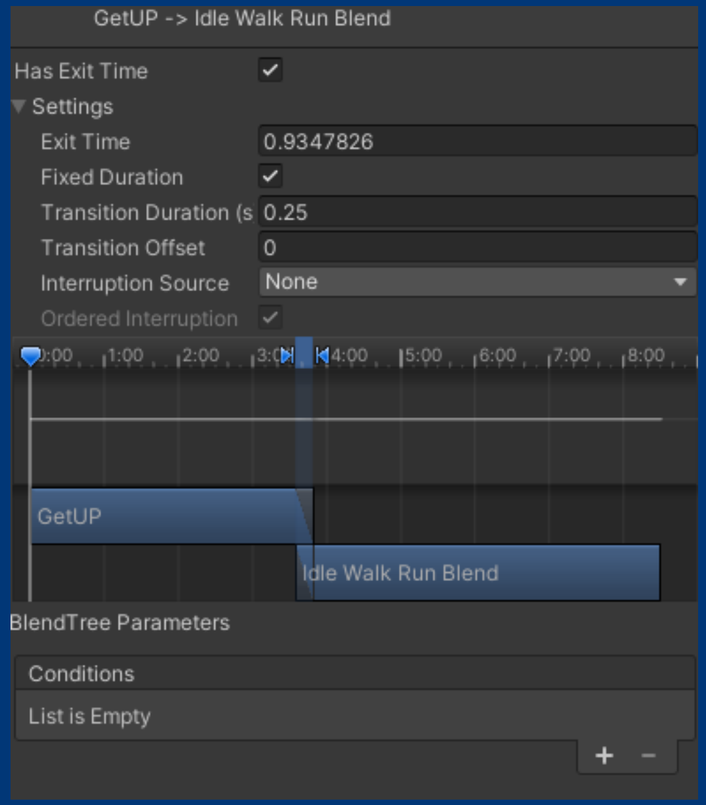
4

Transiciones



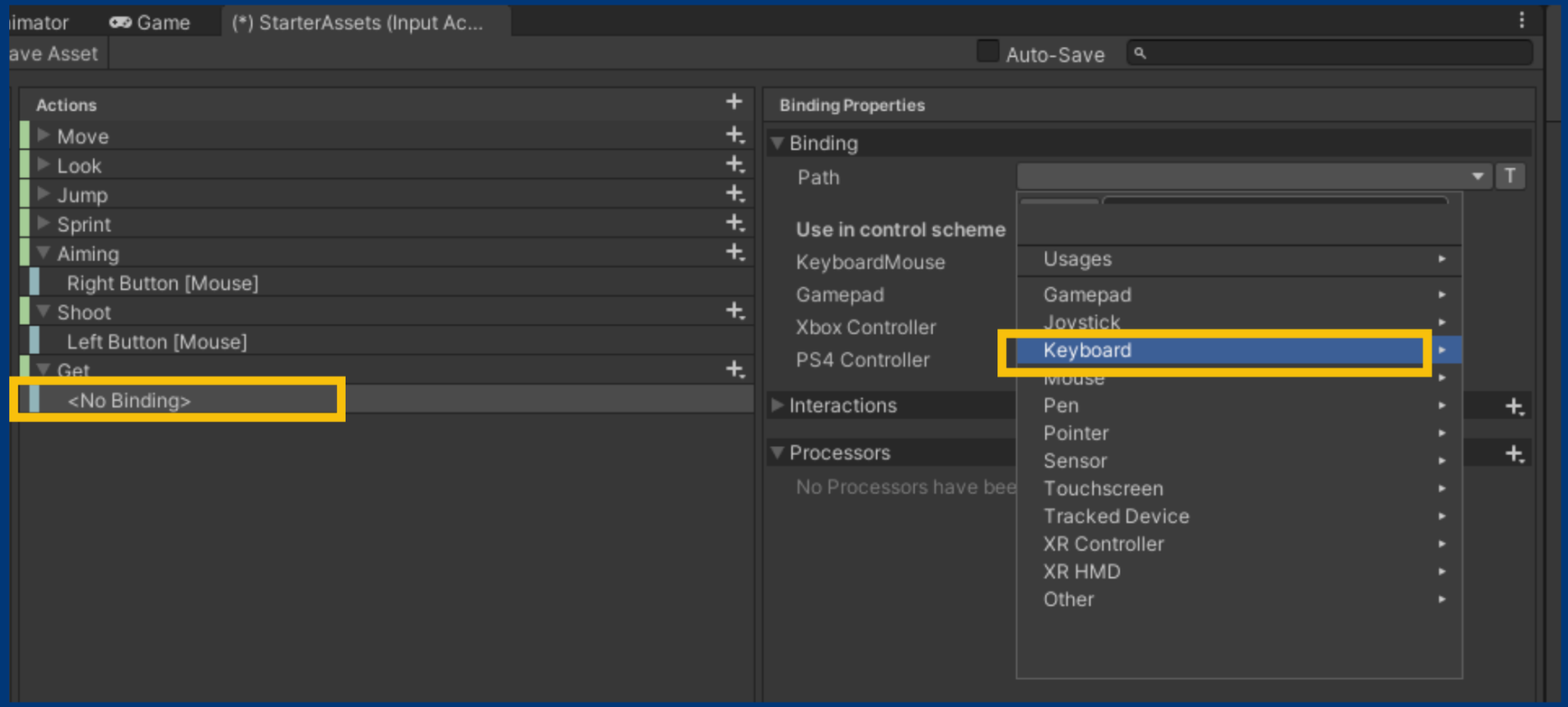


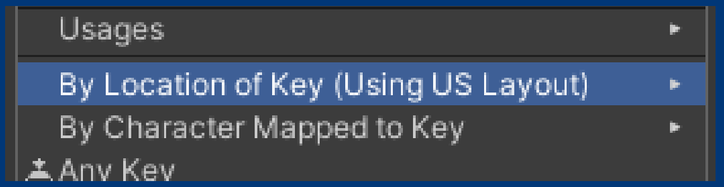


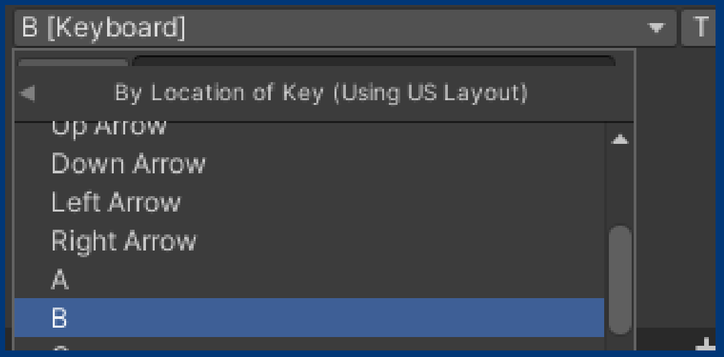


4

Teclas

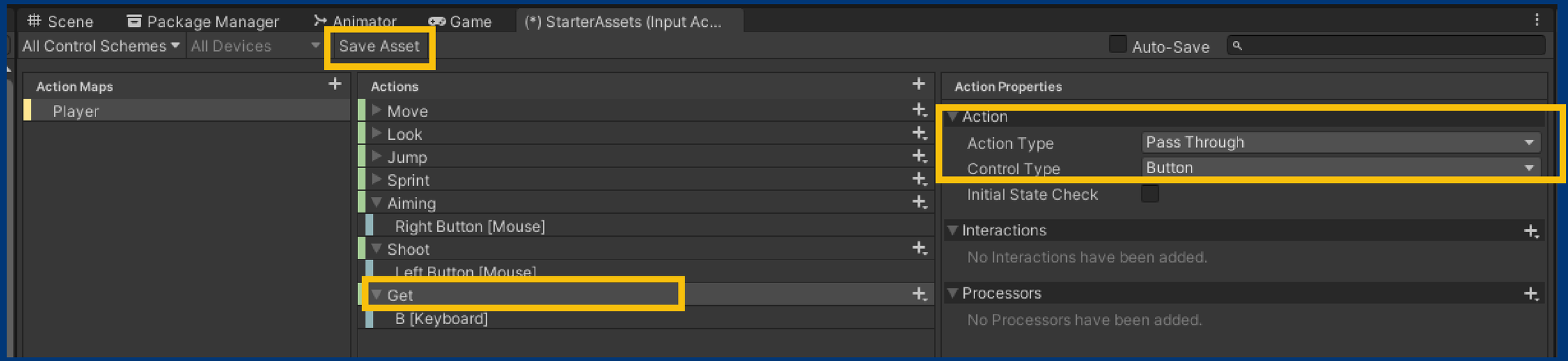

imator
Game
(*) StarterAssets (Input Ac...
ave Asset
Auto-Save
Actions
Move
Look
Jump
Sprint
Aiming
Right Button [Mouse]
Shoot
Left Button [Mouse]
Get
<No Binding>
Binding Properties
Binding
Path
Use in control scheme
KeyboardMouse
Gamepad
Xbox Controller
PS4 Controller
Interactions
Processors
No Processors have bee
Usages
Gamepad
Joystick
Keyboard
Pen
Pointer
Sensor
Touchscreen
Tracked Device
XR Controller
XR HMD
Other


Usages
By Location of Key (Using US Layout)
By Character Mapped to Key
Any Key


B [Keyboard]
By Location of Key (Using US Layout)
Down Arrow
Left Arrow
Right Arrow
A
B

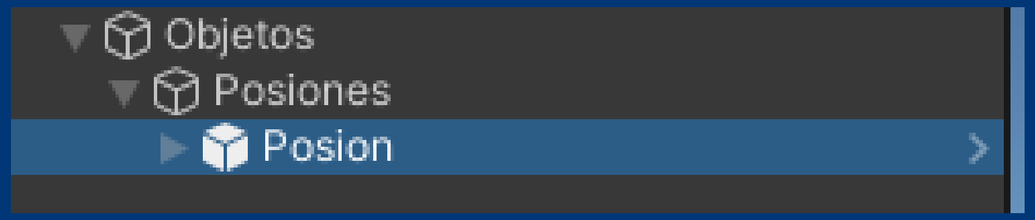
4

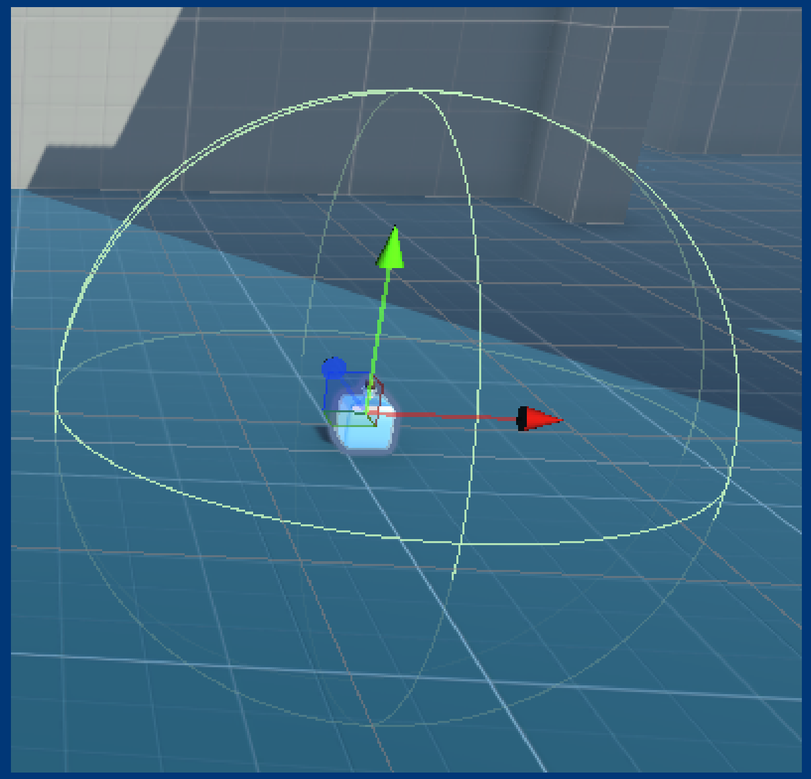
Teclas

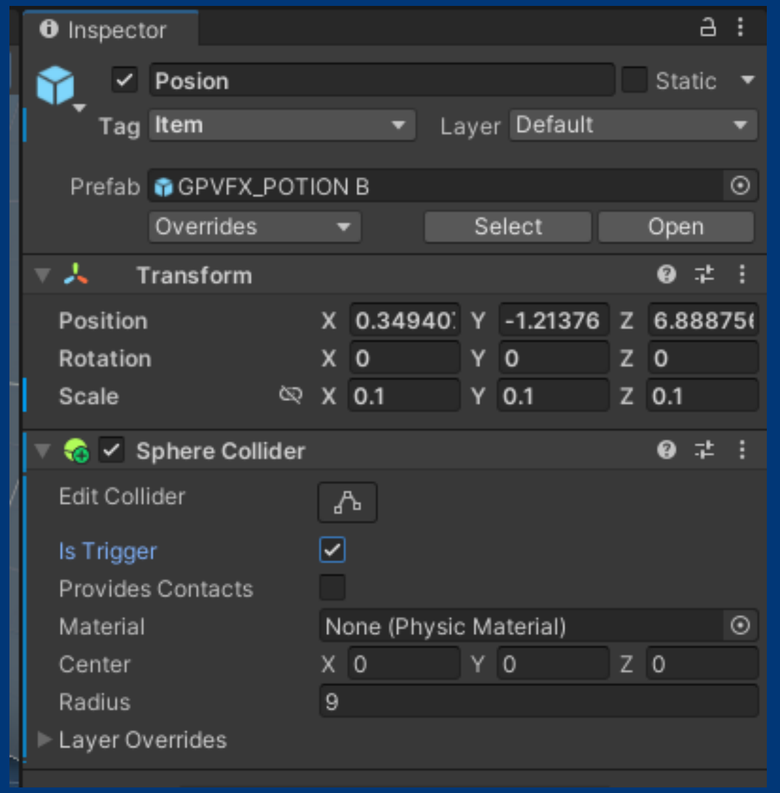

Scene
Package Manager
Animator
Game
(*) StarterAssets (Input Ac...
All Control Schemes
All Devices
Save Asset
Auto-Save
Action Maps
Player
Actions
Move
Look
Jump
Sprint
Aiming
Right Button [Mouse]
Shoot
Left Button [Mouse]
Get
B [Keyboard]
Action Properties
Action
Action Type
Pass Through
Control Type
Button
Initial State Check
Interactions
No Interactions have been added.
Processors
No Processors have been added.

4

Posiciono objeto Posion







4

Codigo

StarterAssetsInputs.cs

```
17          public bool isGet;
```

```
60          public void OnGet(InputValue value)
61          {
62              isGet = value.isPressed;
63          }
```

4

ThirdPersonControlle.cs

```
99          private int _animIDMotionSpeed;
            0 references
100         private bool _isInItemCollision = false;
101
102     #if ENABLE_INPUT_SYSTEM
```

```
156         private void Update()
157         {
158             _hasAnimator = TryGetComponent(out _animator);
159
160             JumpAndGravity();
161             GroundedCheck();
162             Move();
163             AimShoot();
164             Shoot();
165             Getting();
166         }
```

```
193         private void Getting()
194         {
195             if (_input.isGet && Grounded && !_input.sprint && !_input.isAiming && _animationBlend < 0.1f && _isInItemCollision)
196             {
197                 _animator.SetBool("Get", _input.isGet);
198             }
199             else
200             {
201                 _animator.SetBool("Get", false);
202             }
203         }
```

4

ThirdPersonControlle.cs

```
                0 references
206             private void OnTriggerEnter(Collider other)
207             {
208                 if (other.CompareTag("Item"))
209                 {
210                     _isInItemCollision = true;
211                 }
212             }
213
                0 references
214             private void OnTriggerExit(Collider other)
215             {
216                 if (other.CompareTag("Item"))
217                 {
218                     _isInItemCollision = false;
219                 }
220             }
221
```

4

Pruebo

4

ThirdPersonControlle.cs

```
3 references
100     private bool _isInItemCollision = false;
101
102     [Header("Hand")]
103     [Tooltip("HandTransform representa la posicion de la mano Izquierda")]
0 references
104     public Transform HandTransform;
0 references
105     private GameObject _currentItem;
106
```

```
228     // Método para ser llamado desde la animación
0 references
229     public void AttachItemToHand()
230     {
231         if (_currentItem != null)
232         {
233             _currentItem.transform.SetParent(HandTransform);
234             _currentItem.transform.localPosition = Vector3.zero;
235             _currentItem.transform.localRotation = Quaternion.identity;
236         }
237     }
238
239     // Método para ser llamado al final de la animación
0 references
240     public void DetachAndDestroyItem()
241     {
242         if (_currentItem != null)
243         {
244             _currentItem.transform.SetParent(null);
245             Destroy(_currentItem);
246             _currentItem = null;
247         }
248     }
```

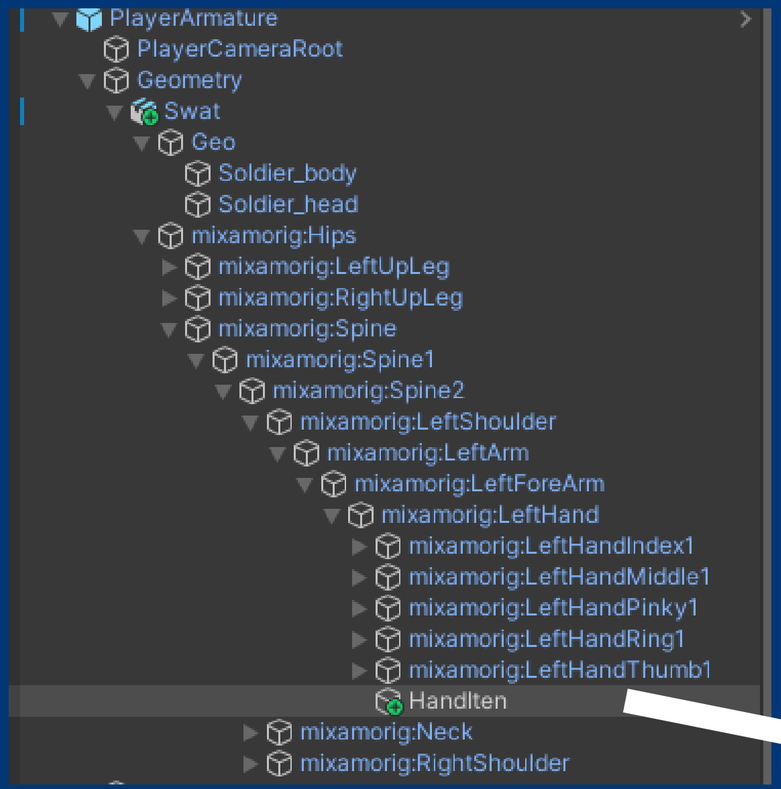
4

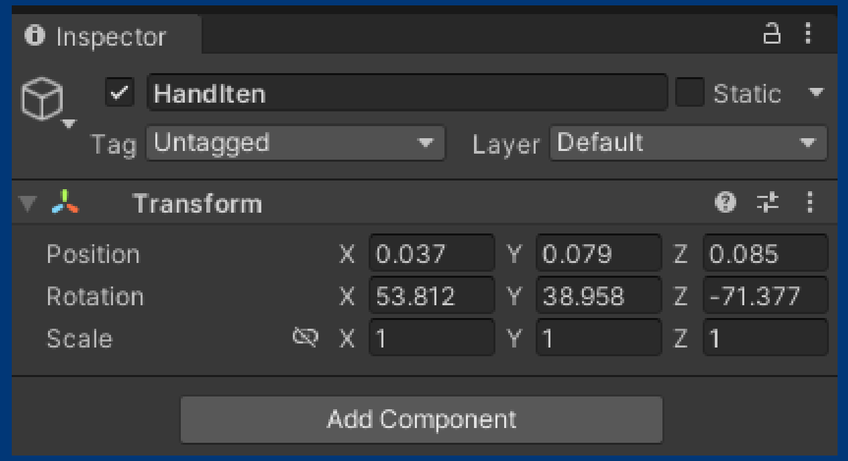
ThirdPersonControlle.cs

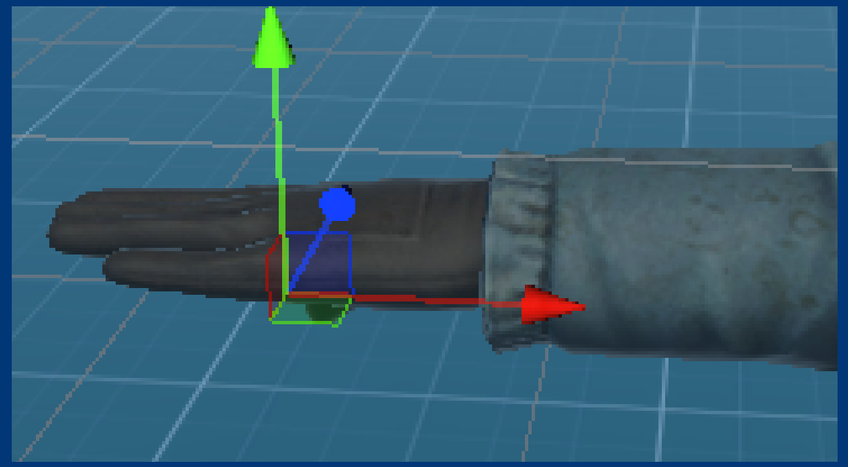
```
0 references
212     private void OnTriggerEnter(Collider other)
213     {
214         if (other.CompareTag("Item"))
215         {
216             _isInItemCollision = true;
217             _currentItem = other.gameObject;
218         }
219     }
220
0 references
221     private void OnTriggerExit(Collider other)
222     {
223         if (other.CompareTag("Item"))
224         {
225             _isInItemCollision = false;
226
227             _currentItem = null;
228         }
229     }
230
```

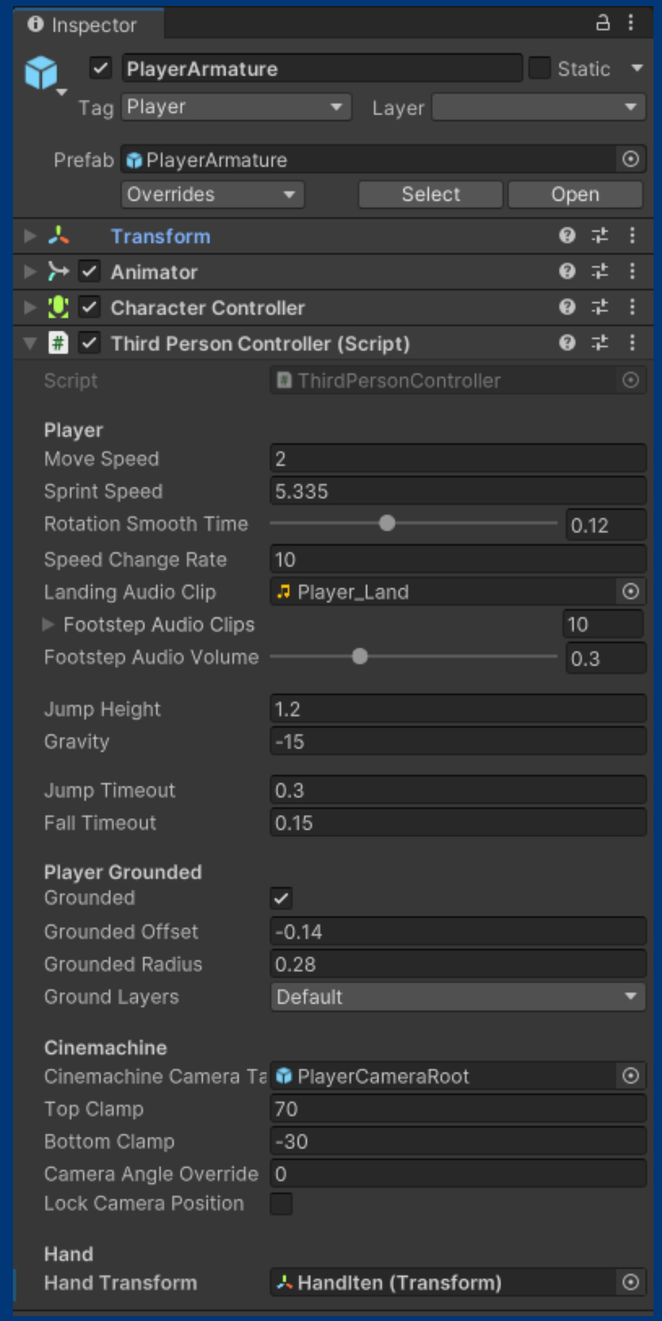
4

ThirdPersonControlle.cs



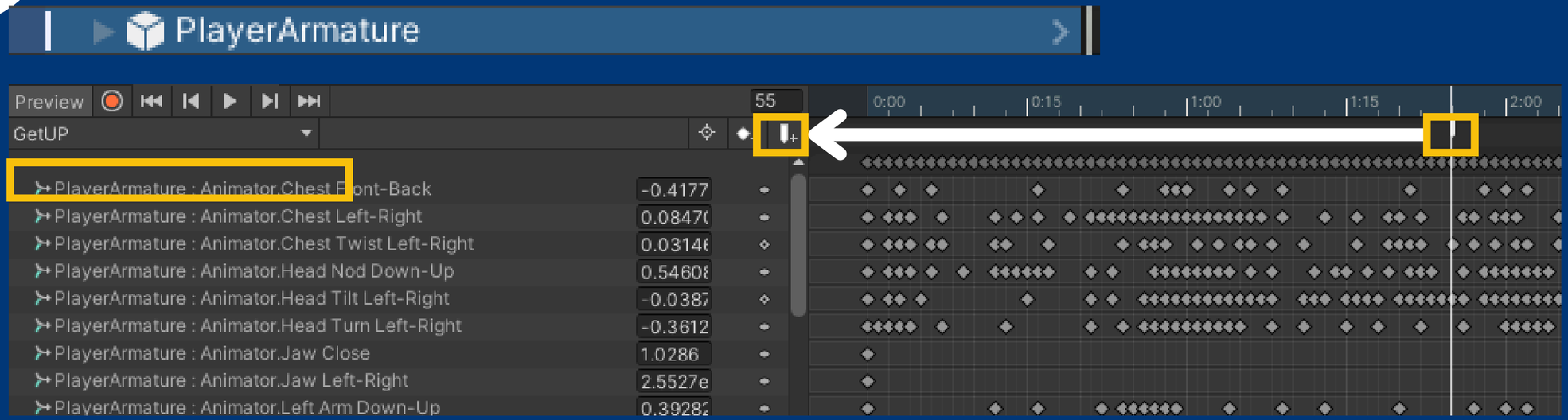


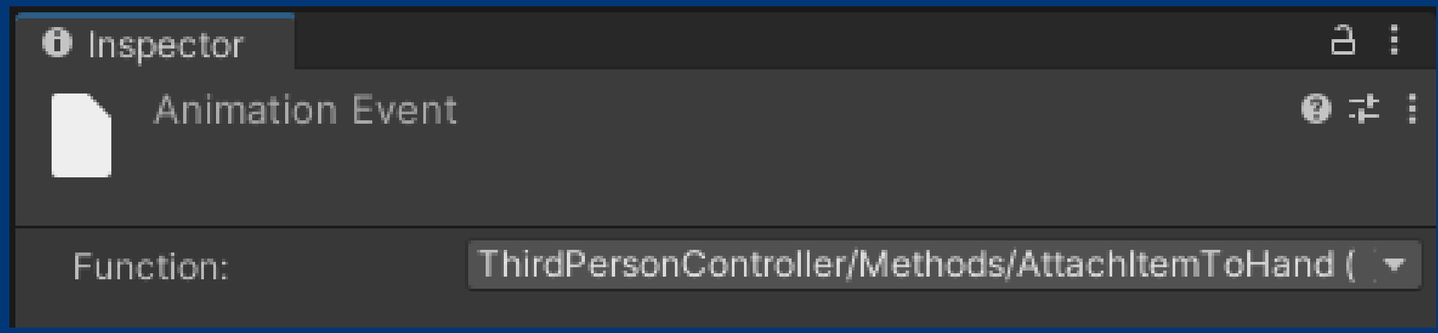




4

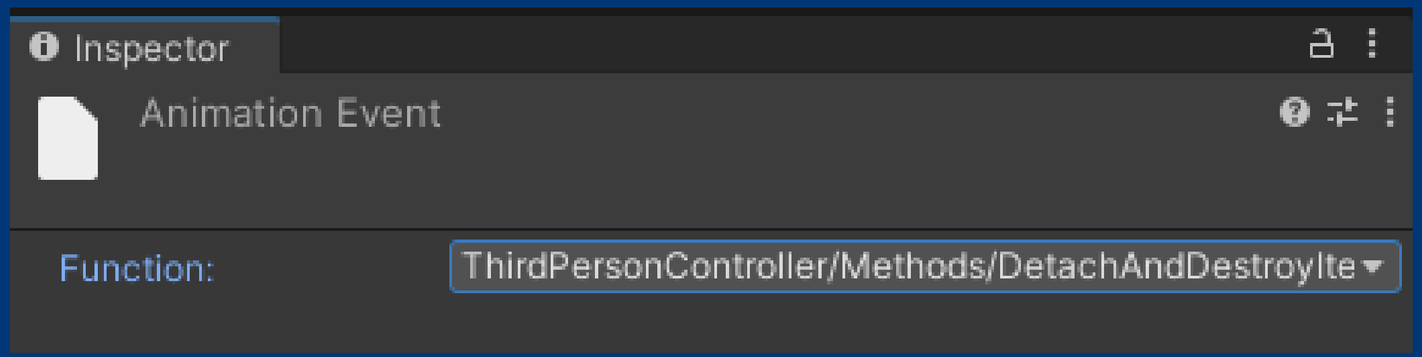
Metodos llamados desde animaciones





4

StarterAssetsInputs.cs



4

StarterAssetsInputs.cs

4

StarterAssetsInputs.cs