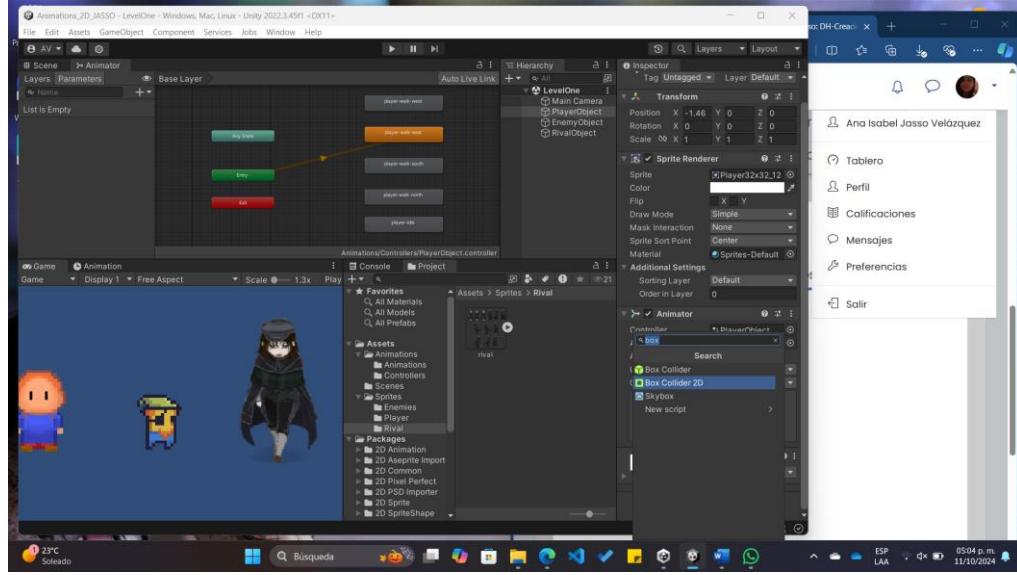
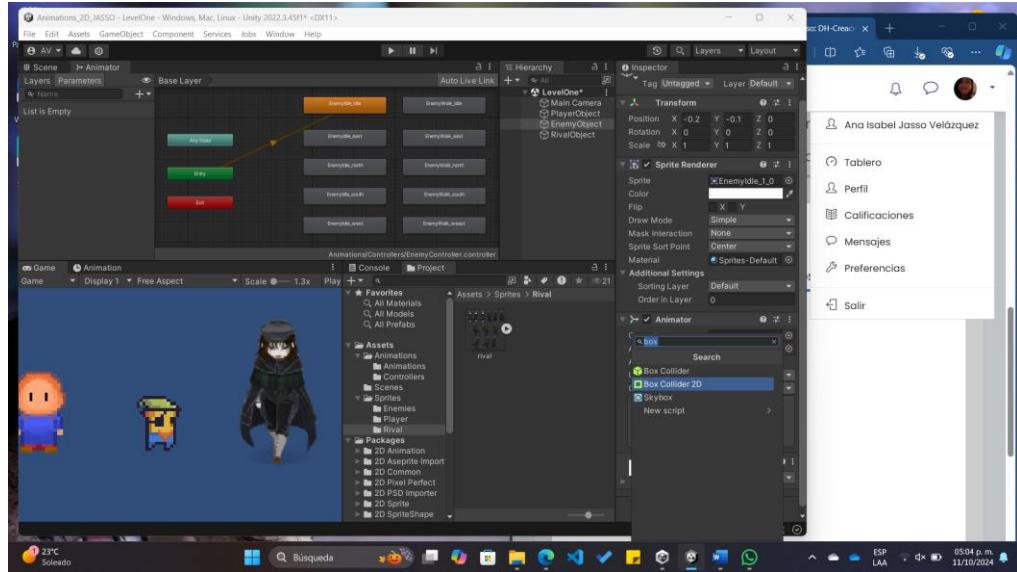


ANIMATIONS 2D PARTE 2

Seleccione PlayerObject y luego seleccione el botón Add Component en la ventana del inspector. Busque y seleccione "Box Collider 2D" para agregar un Box Collider 2D al PlayerObject.

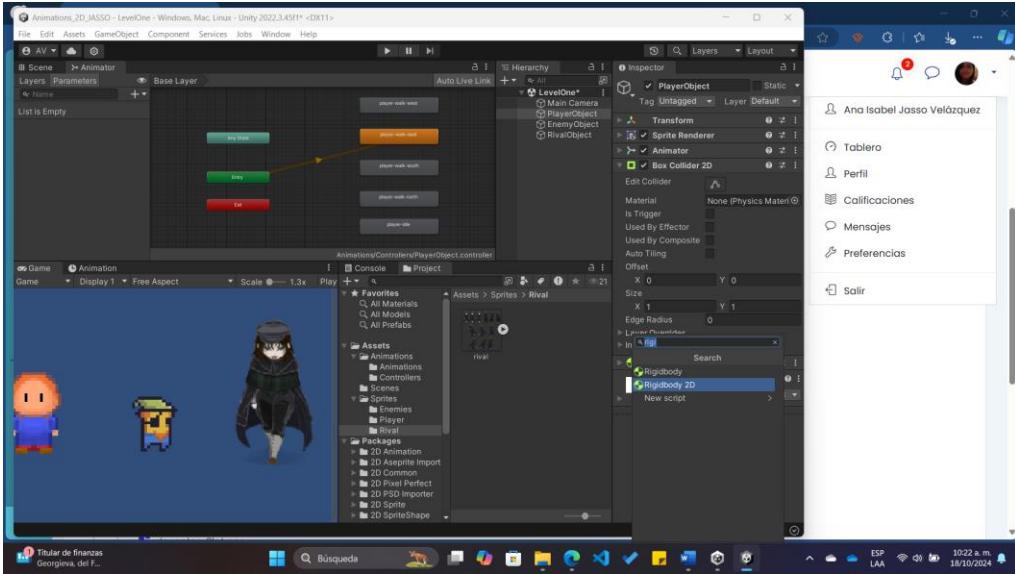


Necesitaremos saber cuándo el jugador choca con un enemigo, así que agregue un Box Collider 2D al EnemyObject también



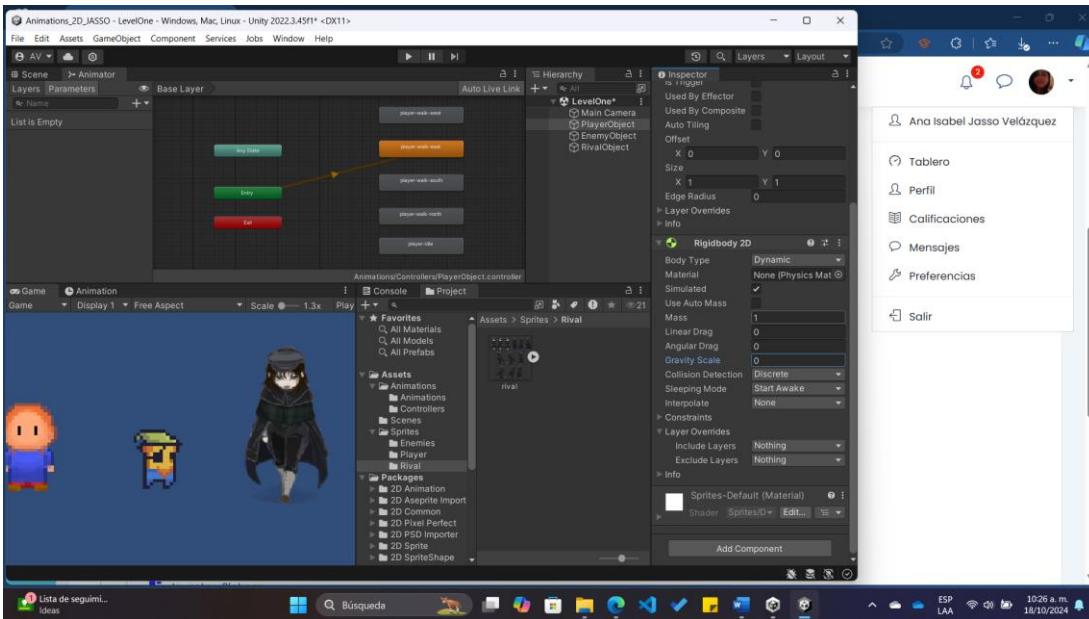
RigidBody

Con PlayerObject seleccionado, haga clic en el botón “Add component” en la ventana Inspector, busque “Rigidbody 2D” y agréguese al PlayerObject.

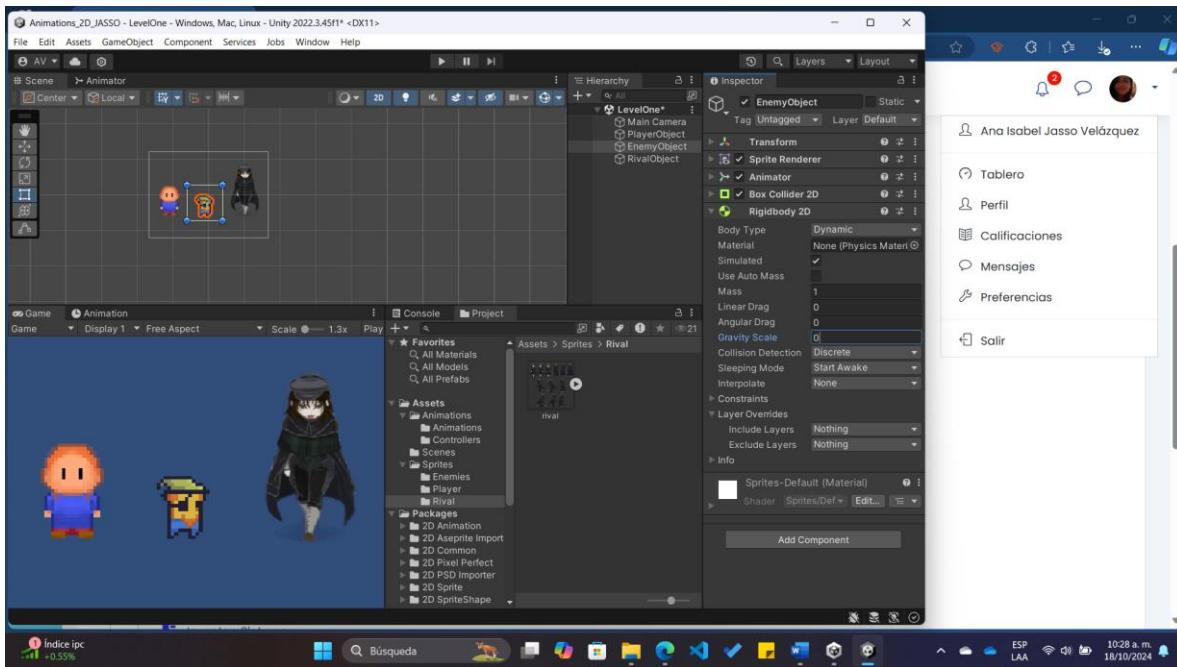
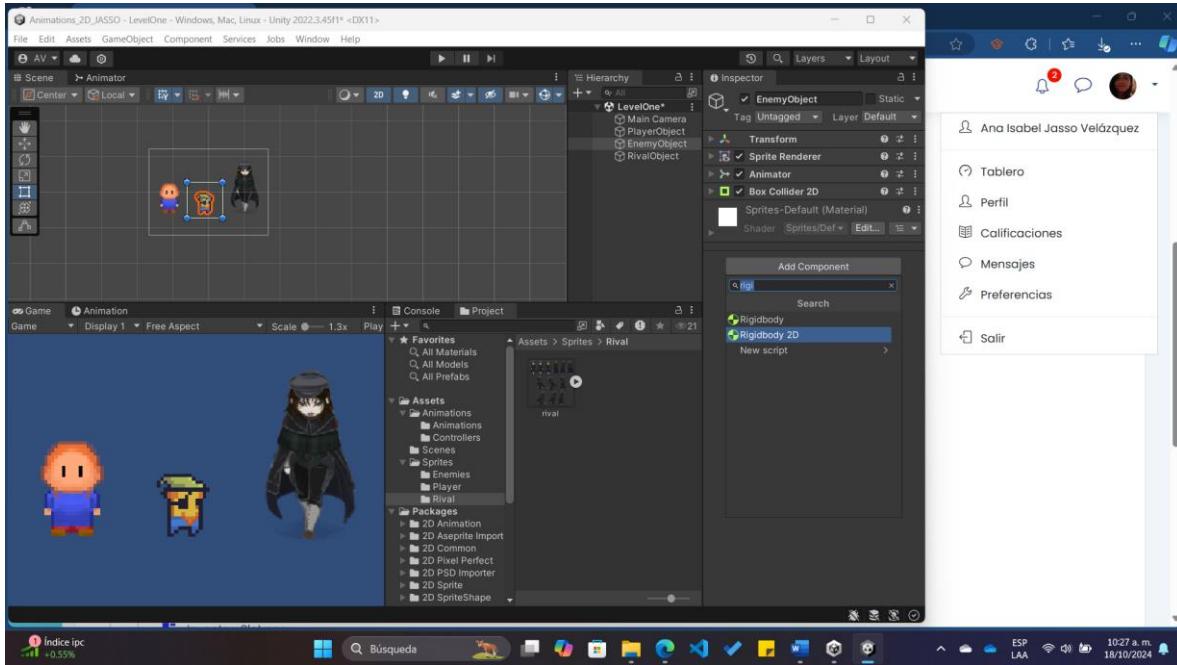


En el menú desplegable establezca los valores a las siguientes propiedades:

- **Body Type Dynamic**
- **Mass 1**
- **Linear Drag 0**
- **Angular Drag 0**
- **Gravity Scale 0**

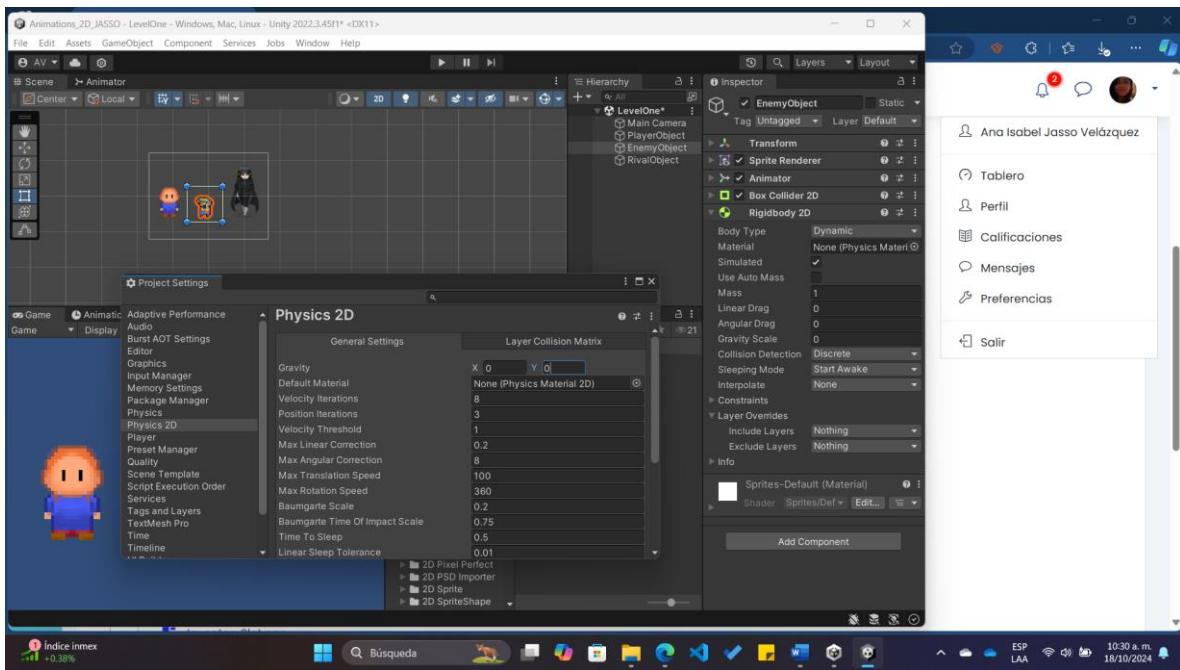


Seleccione el EnemyObject y agregue un Componente Rigidbody 2D de tipo Dinámico para él también.



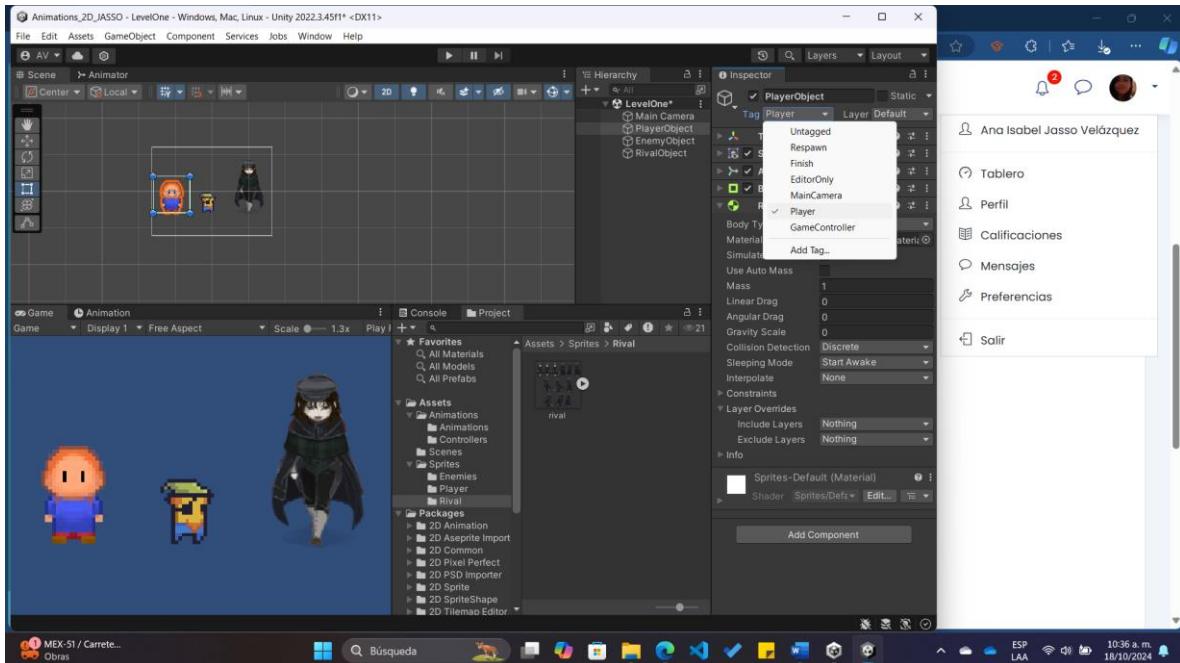
Ahora que agregamos un Rigidbody 2D a nuestro jugador y enemigo, se verán afectados por la gravedad. Debido a que nuestro juego usa una perspectiva top-down, apaguemos la gravedad para que nuestro jugador no salga volando en la pantalla.

Vaya a la opción Edit > Project settings > Physics 2D y cambie el valor para la gravedad Y de -9,81 a 0.

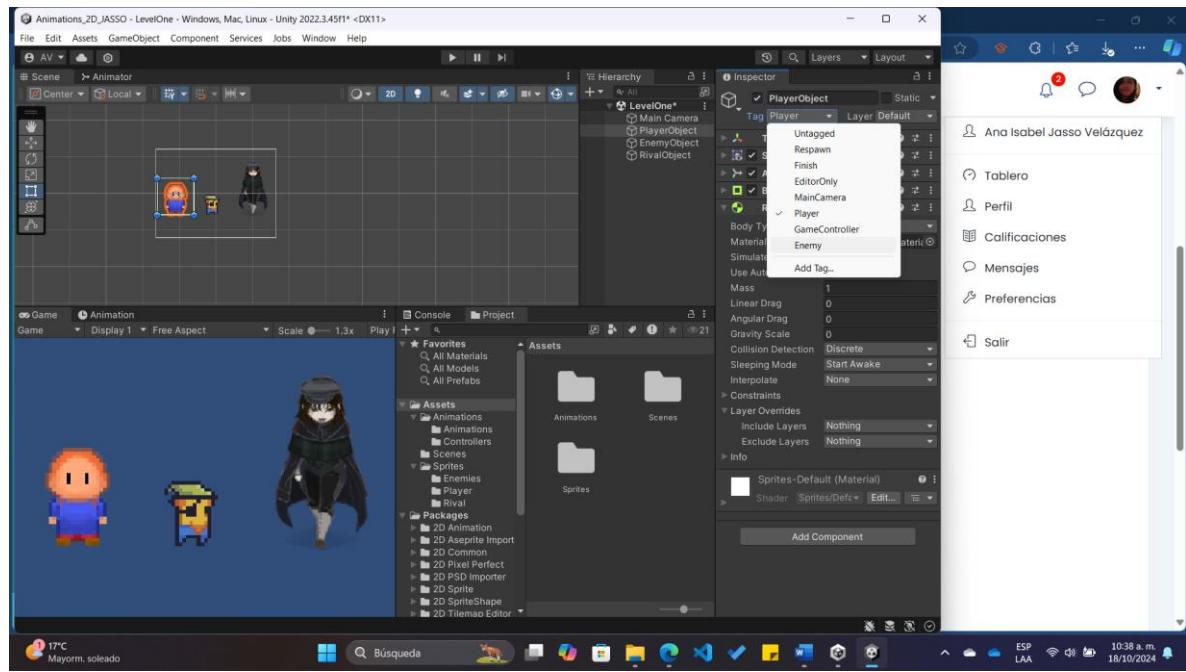
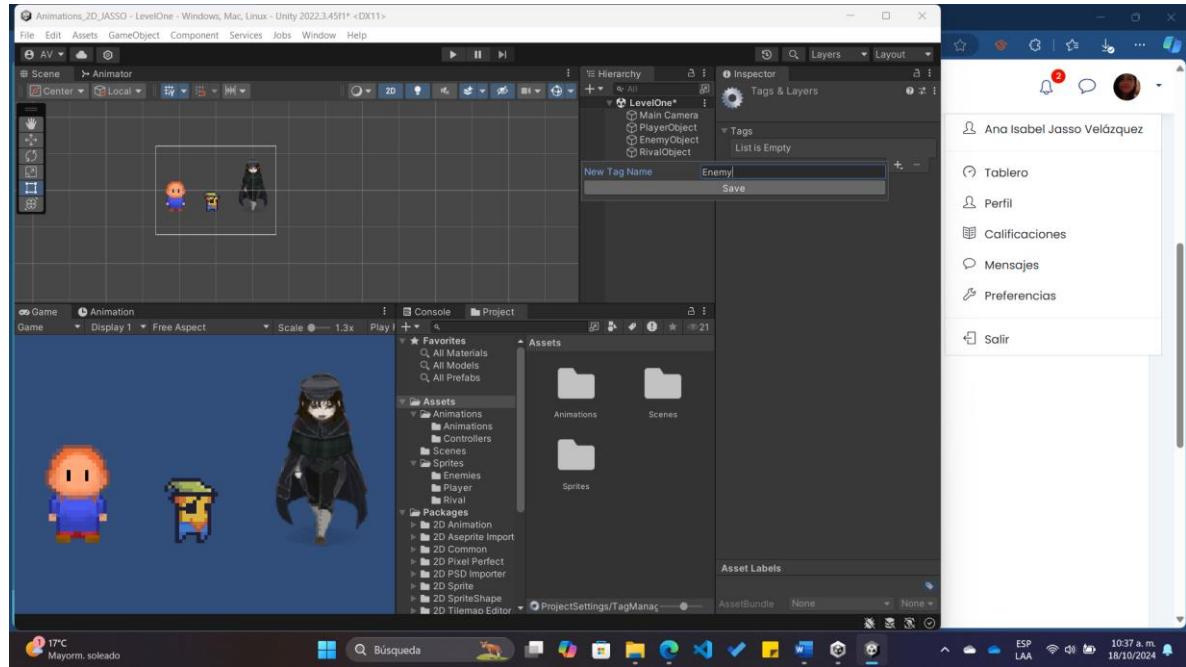


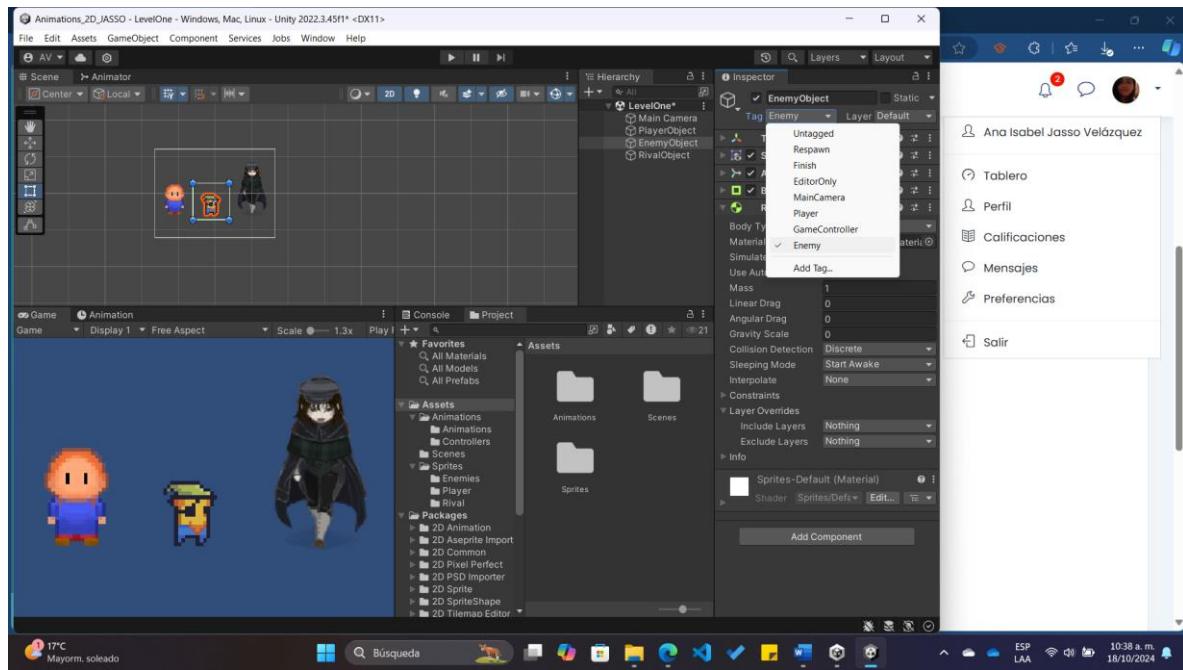
Etiquetas y Capas

Selecciona el PlayerObject. En el menú desplegable Tag en la parte superior izquierda del Inspector, seleccione la etiqueta de Jugador para agregar una etiqueta a nuestro PlayerObject.



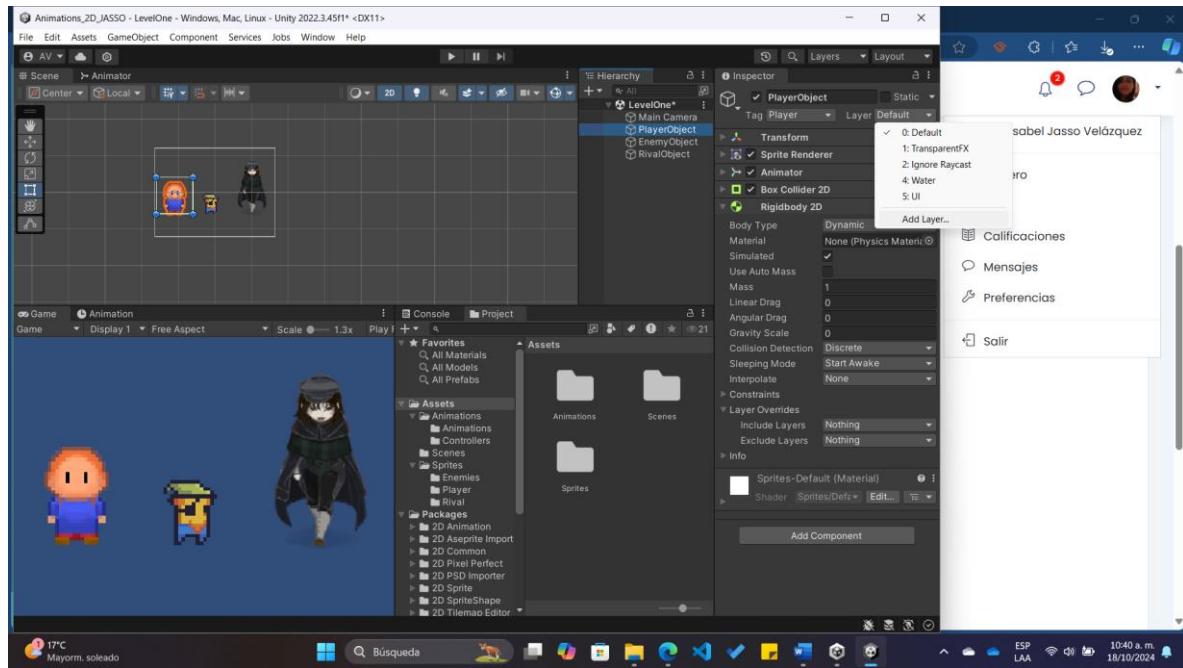
Crear una nueva etiqueta llamada "Enemy" y úsala para configurar la etiqueta del objeto EnemyObject

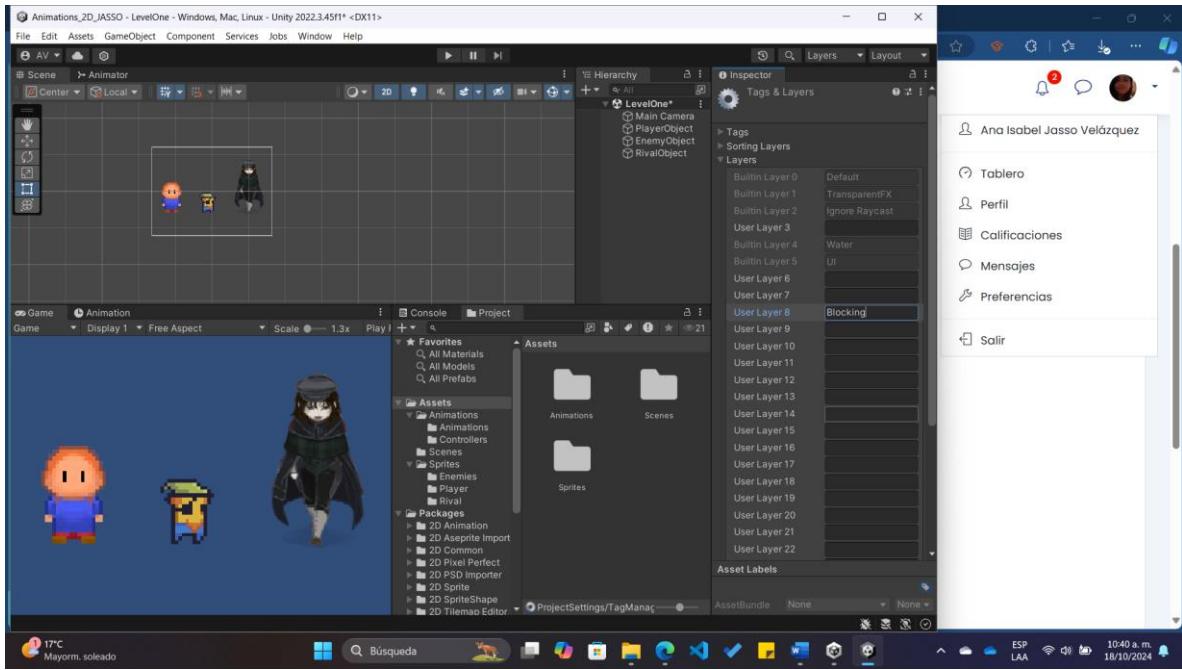




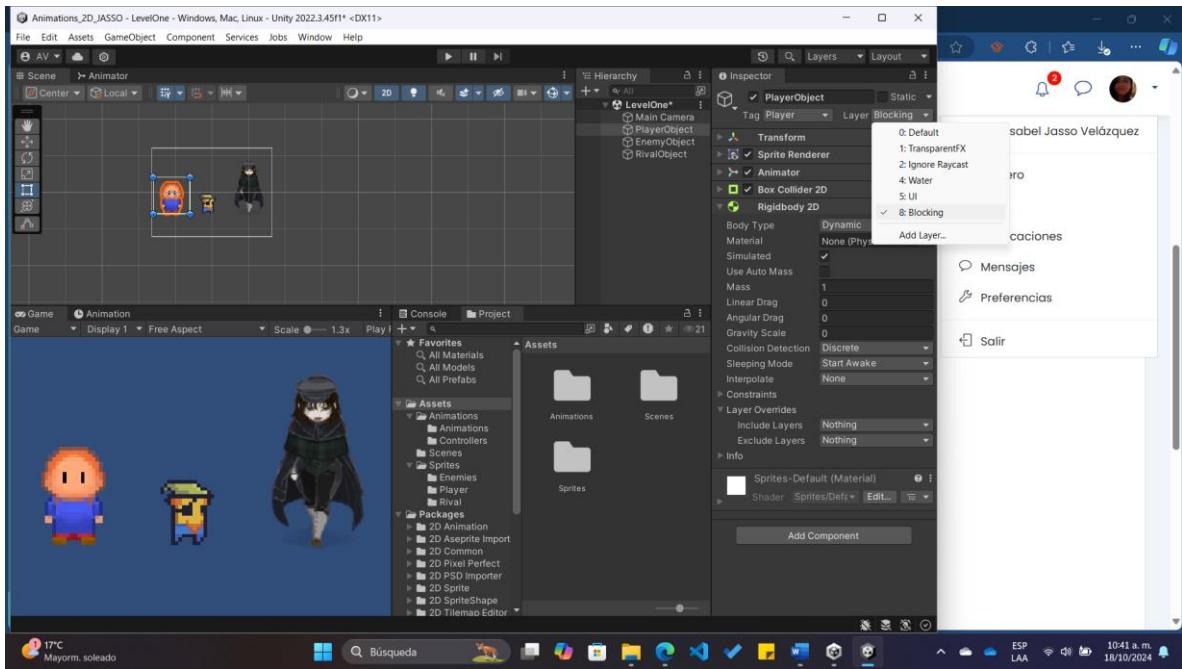
Capas

Seleccionar en el menú desplegable Layers y seleccionar "Add layer". Debería ver la ventana Layers



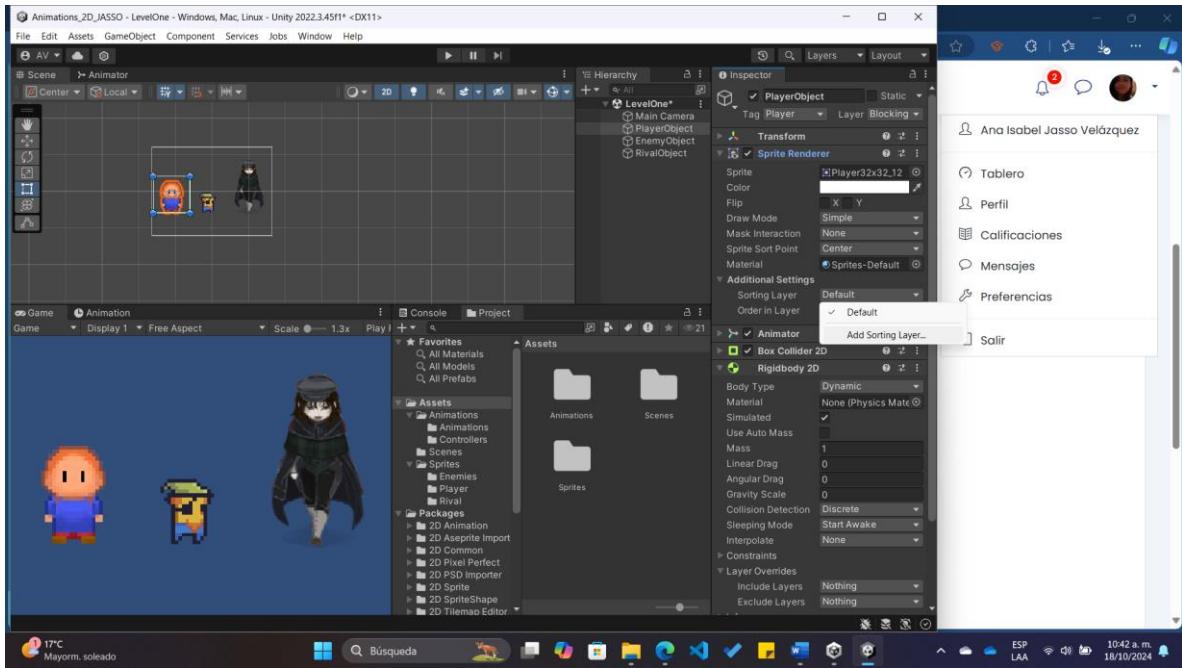


Ahora seleccione el PlayerObject nuevamente para ver sus propiedades en el Inspector. Seleccione la capa de bloqueo que acabamos de crear en el menú desplegable.

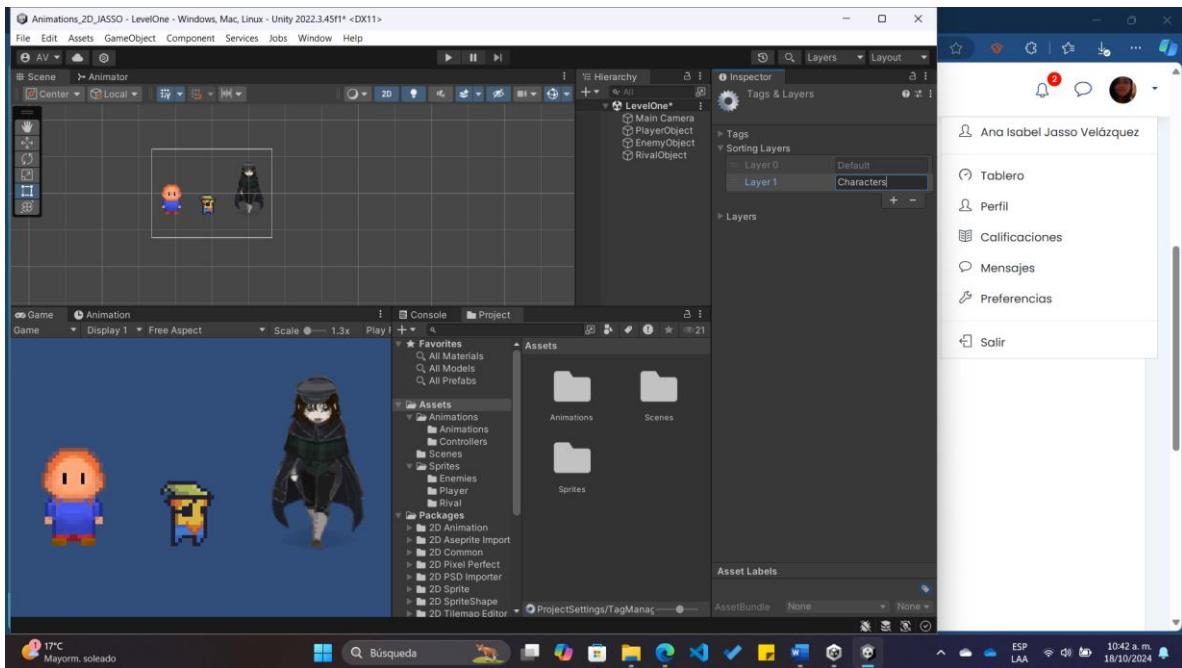


Sorting layers

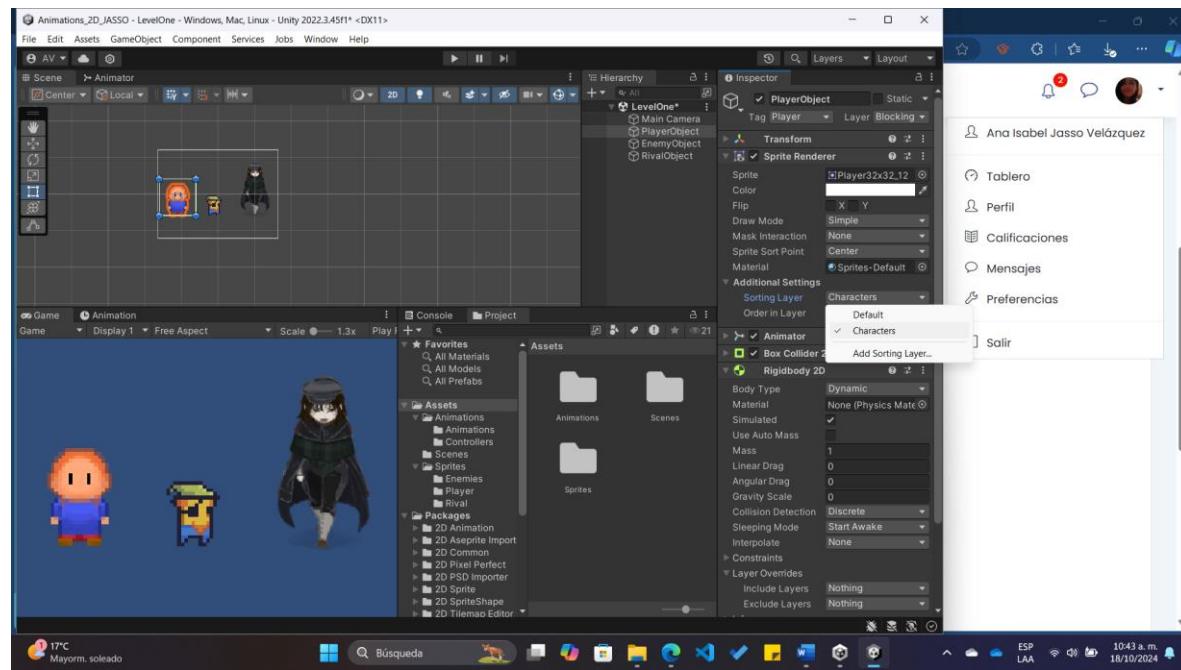
Vamos a agregar una capa de ordenamiento llamada "Caracteres" que usaremos para nuestro jugador y todos los enemigos.



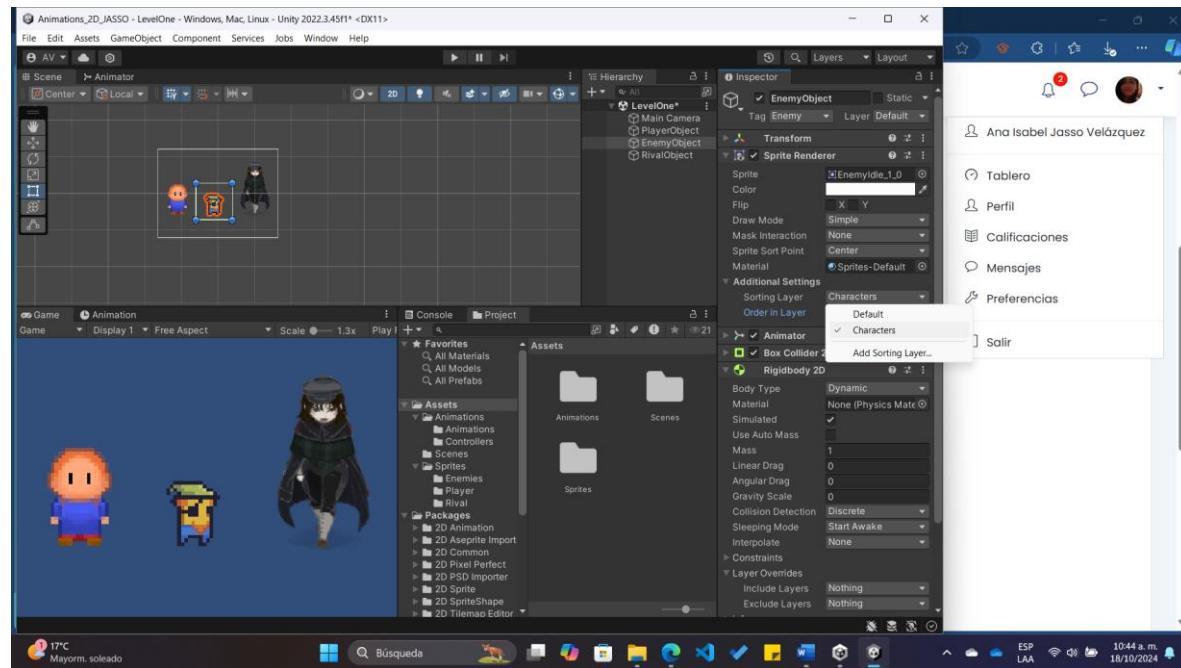
En el componente Sprite Renderer en la ventana Inspector, seleccione la opción **Sorting layer** Menú desplegable de capa y seleccione "Add Sorting Layer".

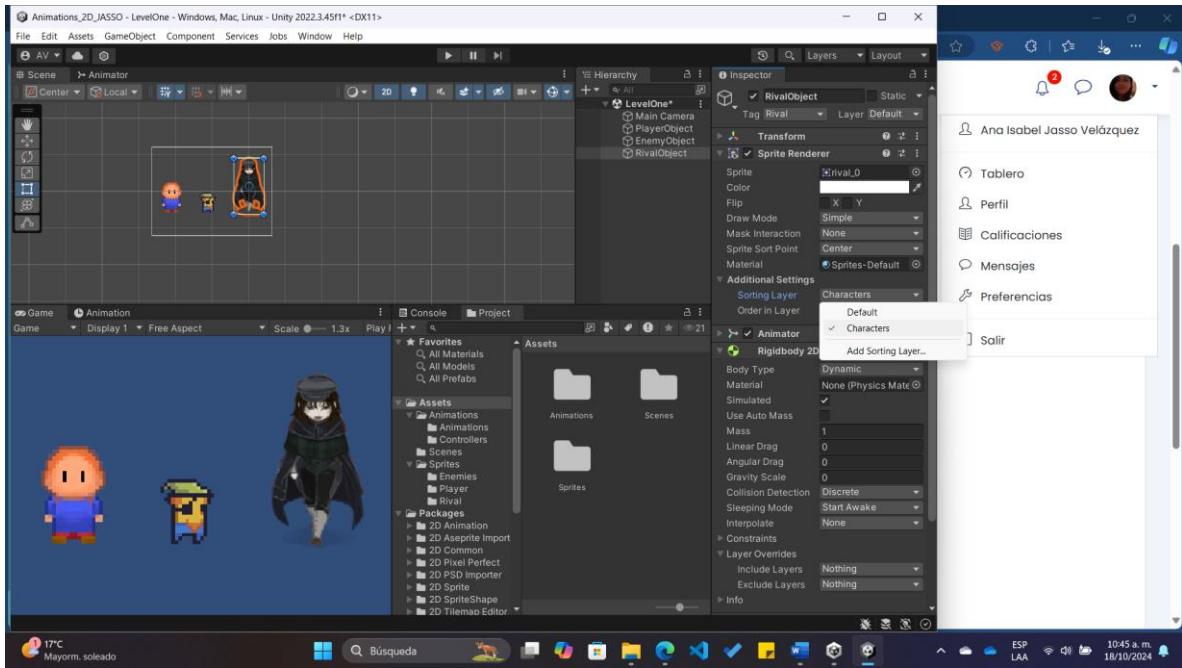


Agregue una capa de ordenamiento llamada "Characters", y luego haga clic en PlayerObject nuevamente para ver su Inspector y seleccione la nueva Capa de ordenamiento de caracteres del menú desplegable.



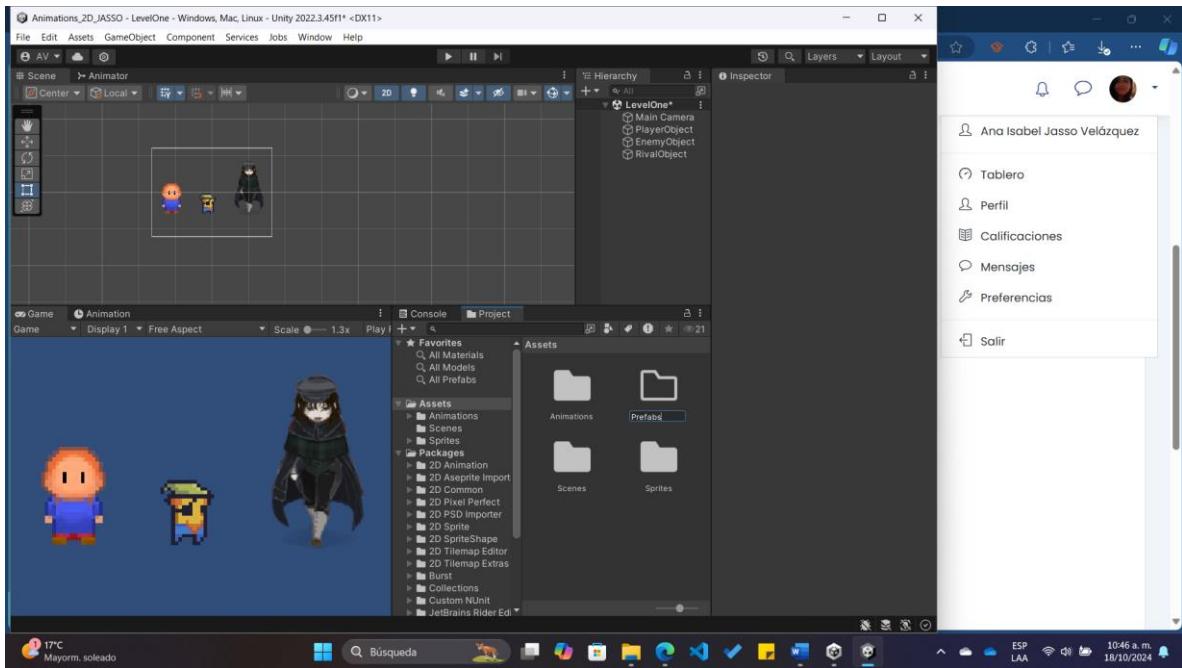
Seleccione nuestro EnemyObject y establezca su Capa de ordenamiento en Characters, porque queremos que los enemigos también aparezcan en la parte superior de las cosas.



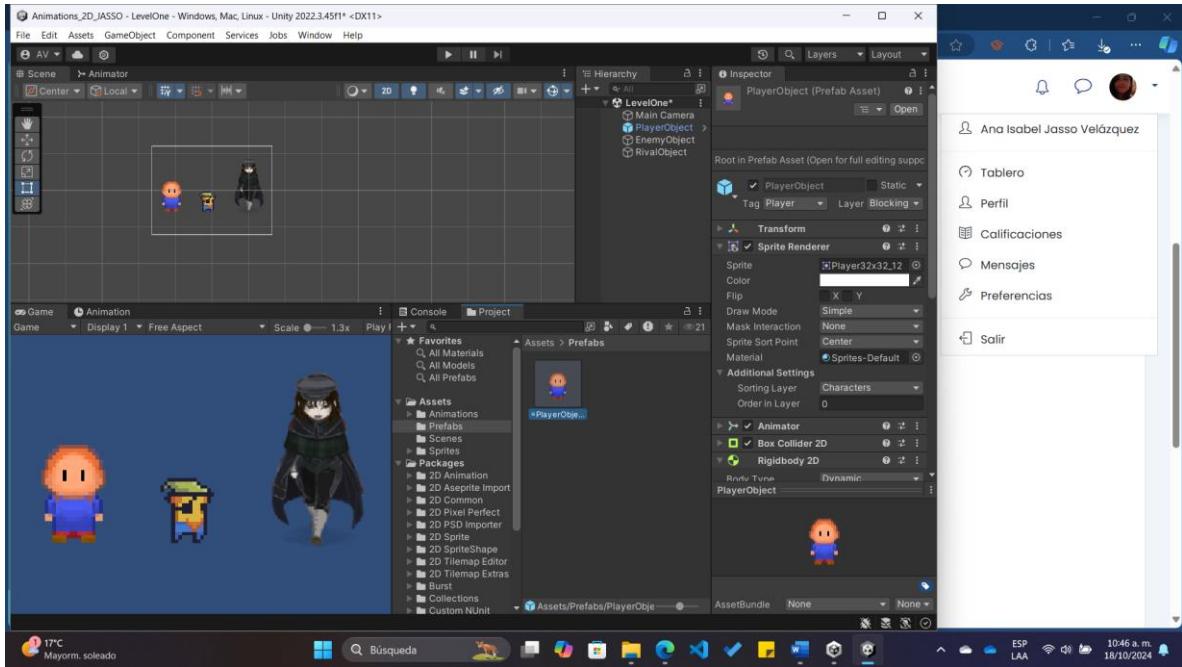


Prefs

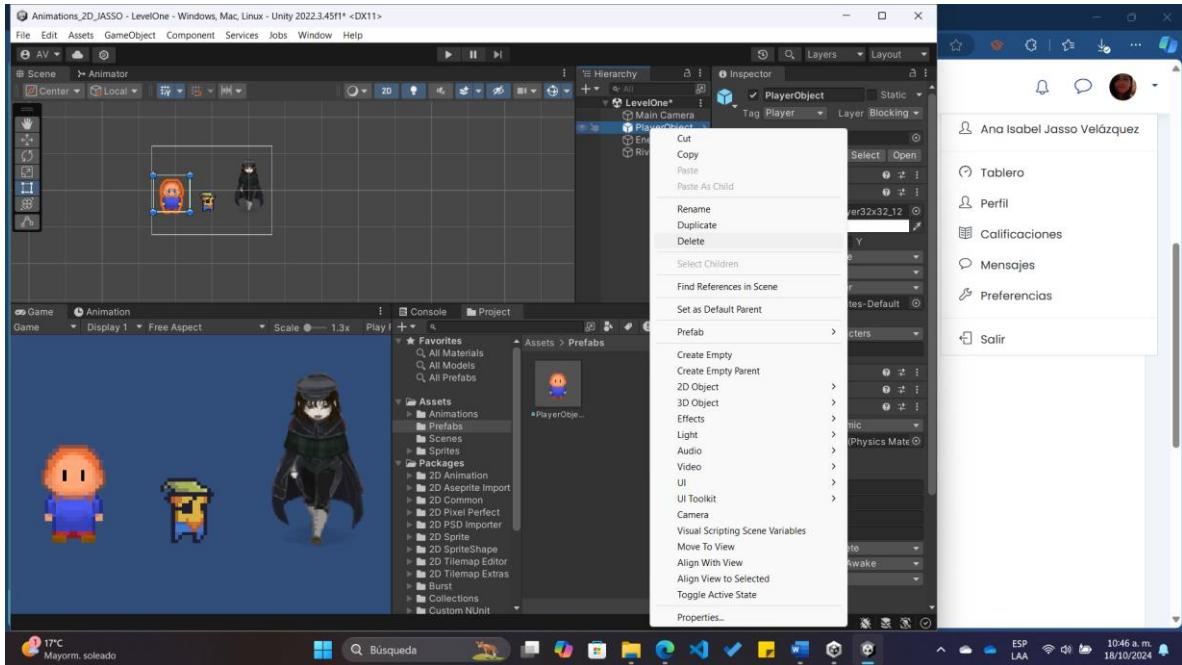
Primero, crea una carpeta llamada “Prefabs” en nuestra carpeta Assets en la vista Proyecto.

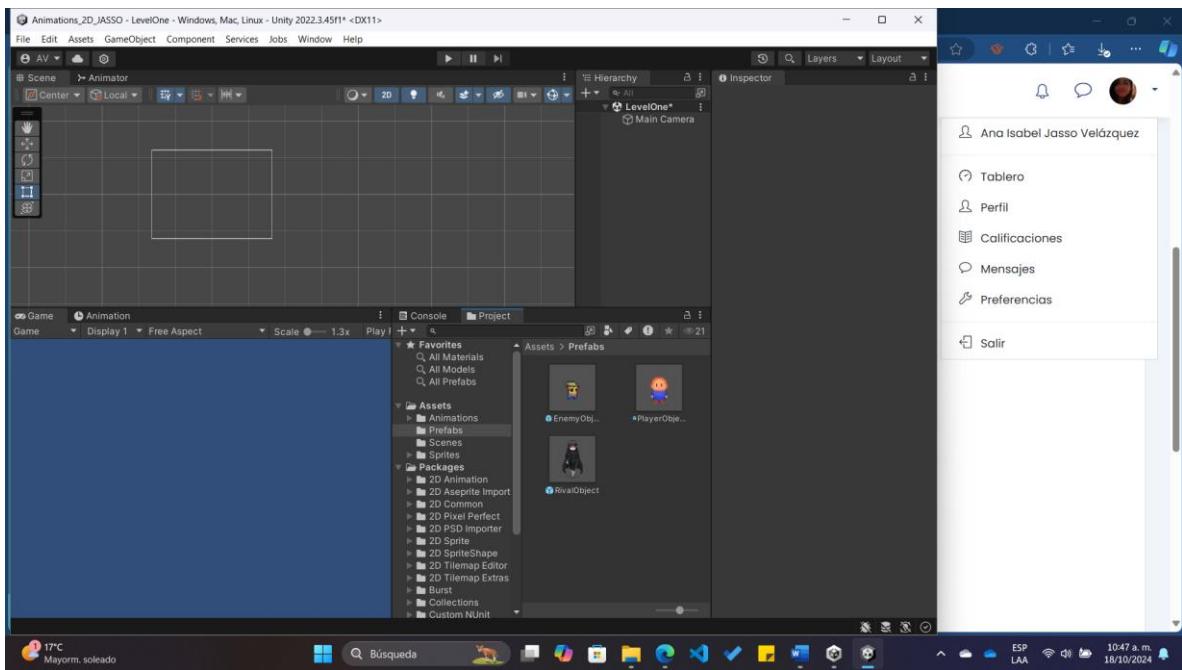
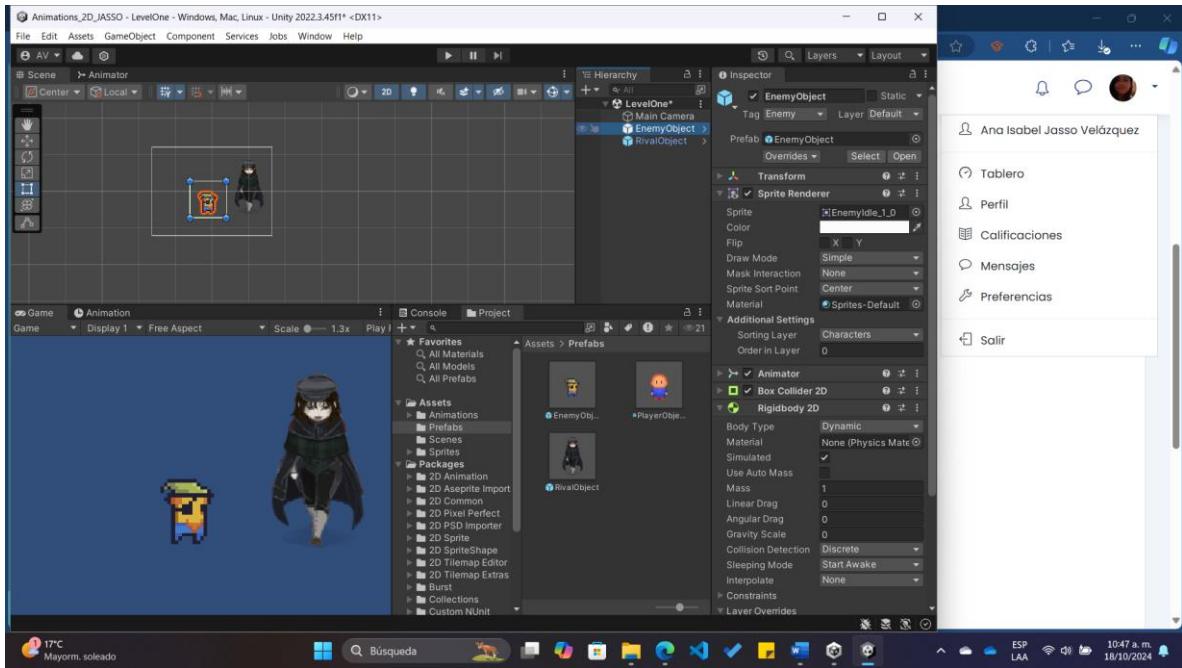


Luego seleccione nuestro PlayerObject de la vista de Hierarchy y simplemente arrástralolo a la carpeta Prefabs



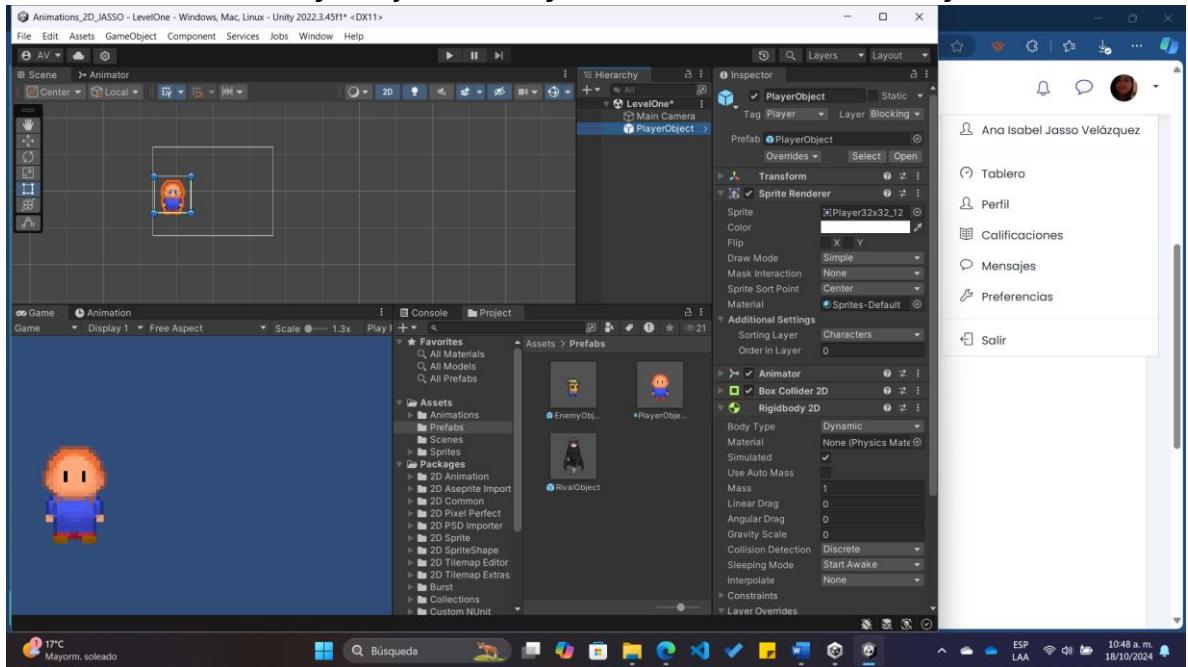
Ahora puedes eliminar de forma segura el PlayerObject de la vista de Hierarchy. Haga lo mismo con EnemyObject: arrástralolo a la carpeta Prefabs y eliminar el EnemyObject original de la vista Hierarchy



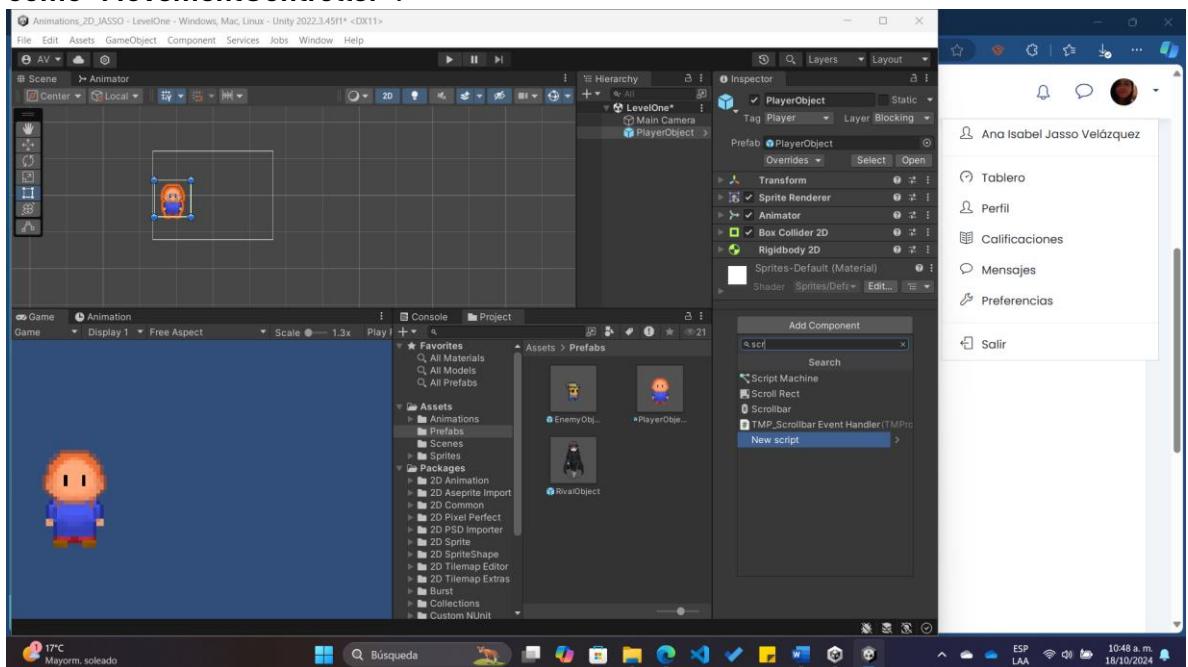


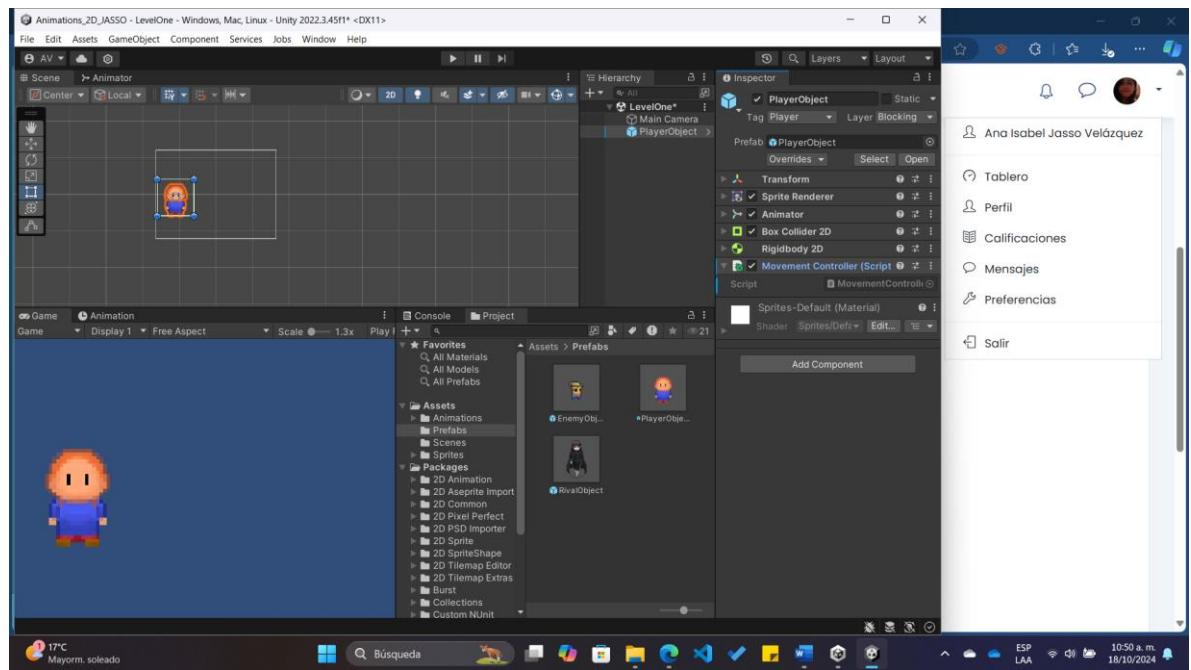
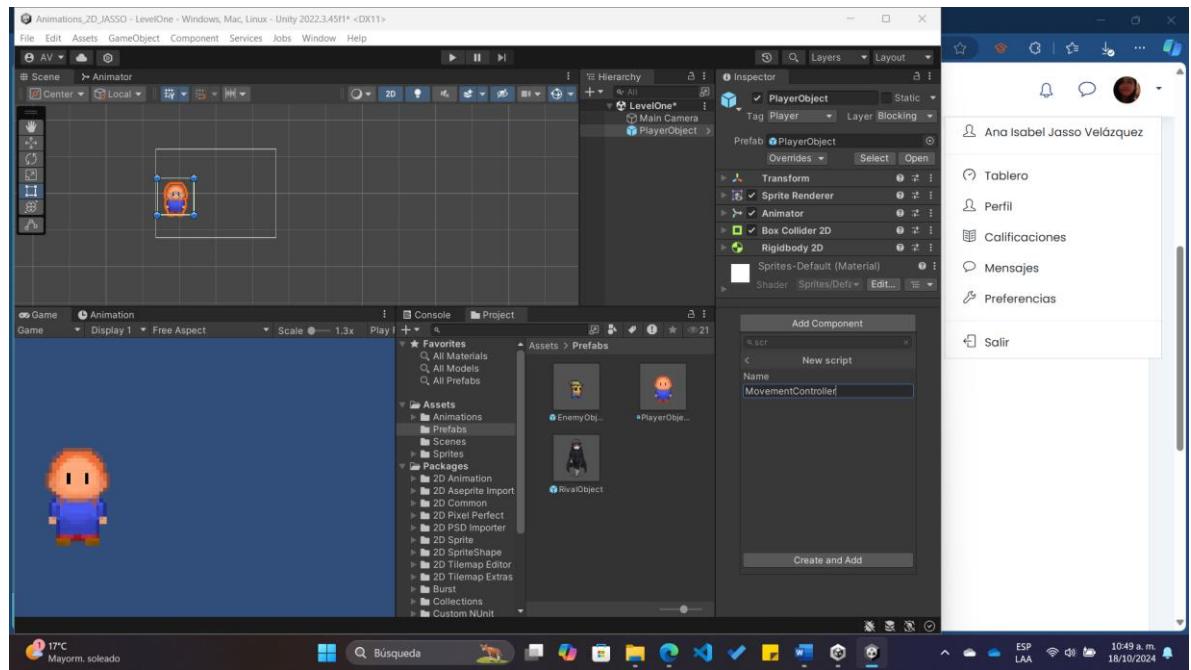
Scripts: Lógica para componentes

- Seleccione nuestro PlayerObject Prefab y arrástrelo a la vista Hierarchy.

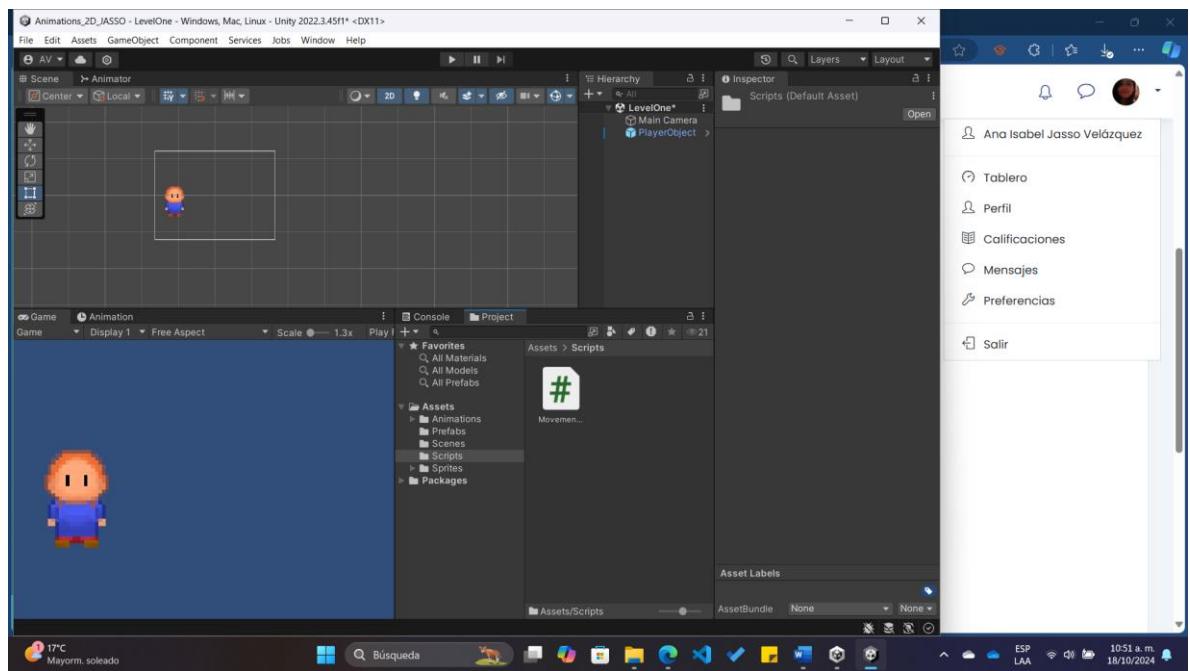
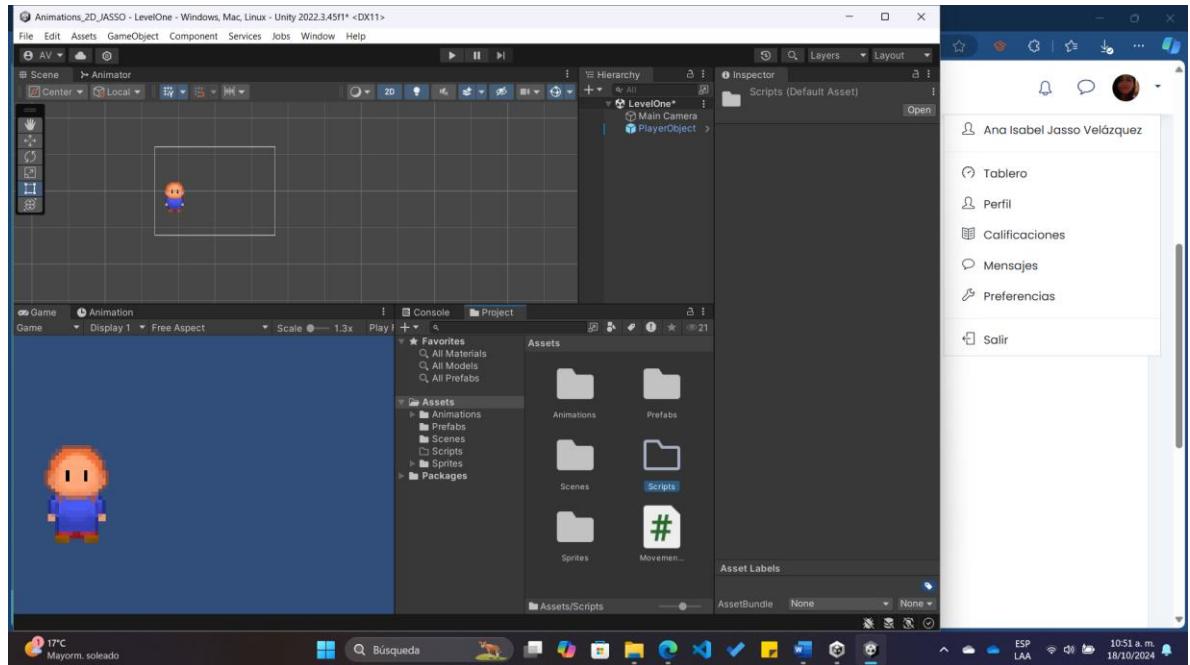


- Desplácese hasta la parte inferior del Inspector y presione el botón Add Component. Escriba la palabra Script y seleccione "New Script" y nombrarlo como "MovementController".



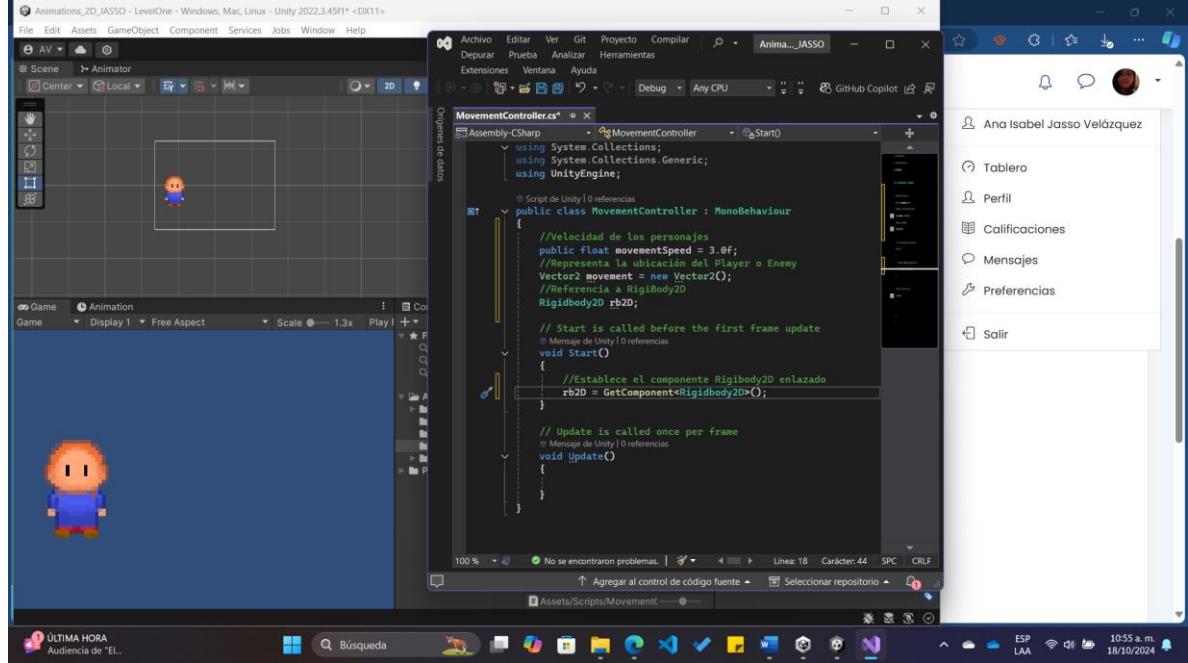


- Cree una nueva carpeta llamada "Scripts" en la vista Project. El nuevo script se habrá creado en la carpeta Assets de nivel superior en la vista Project. Arrastre el script MovementController a la carpeta Scripts y luego haga doble clic en él para abrirlo en Visual Studio.

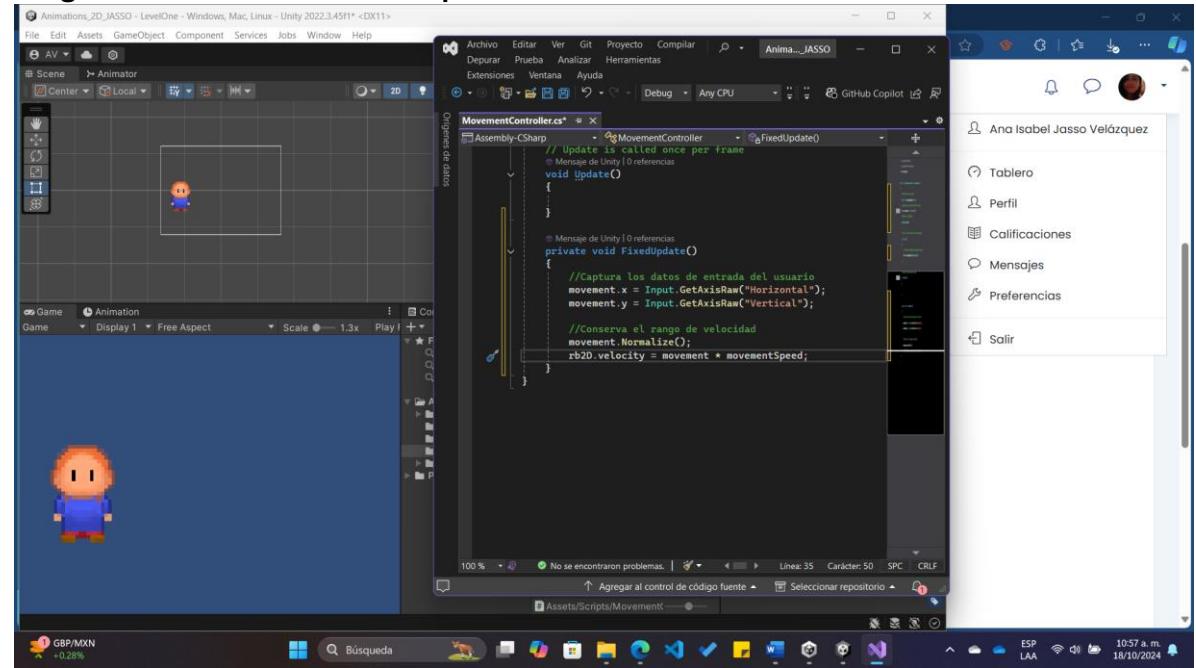


Es hora de programar nuestro primer Script. Los scripts en Unity están escritos en un lenguaje llamado C#. El espacio de nombres UnityEngine contiene muchas clases específicas de Unity como MonoBehaviour, GameObject, Rigidbody2D y BoxCollider2D.

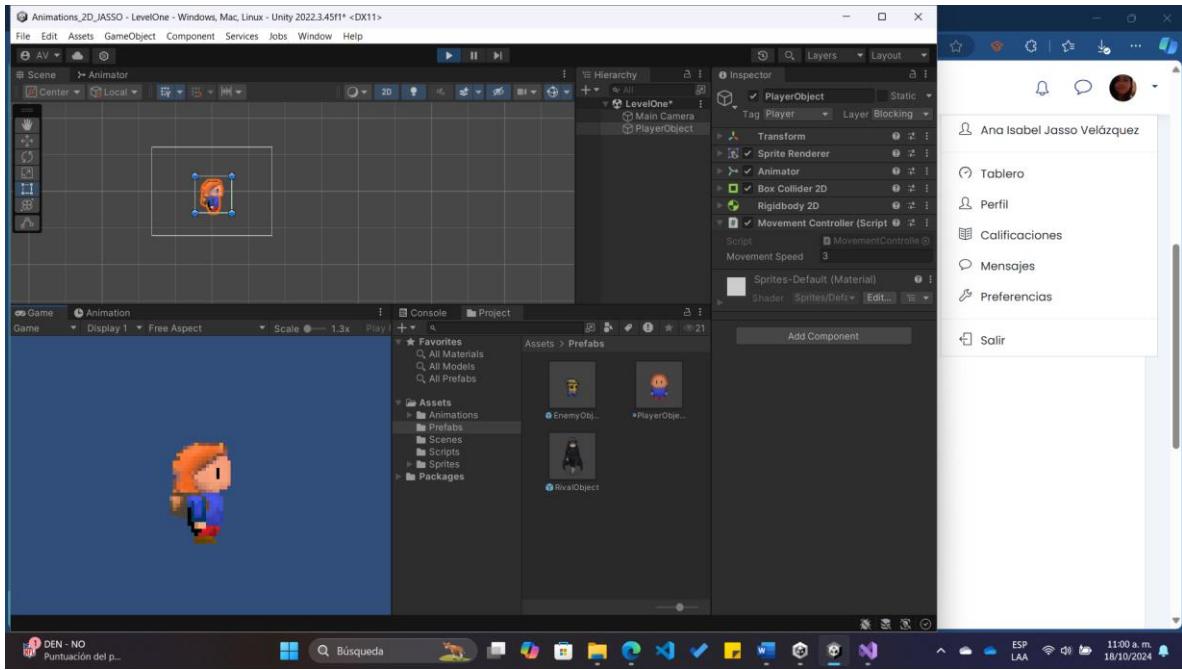
- ← Agreguemos las siguientes propiedades a la clase y Establezcamos la referencia RigidBody2D



- ← Programemos el método FixedUpdate

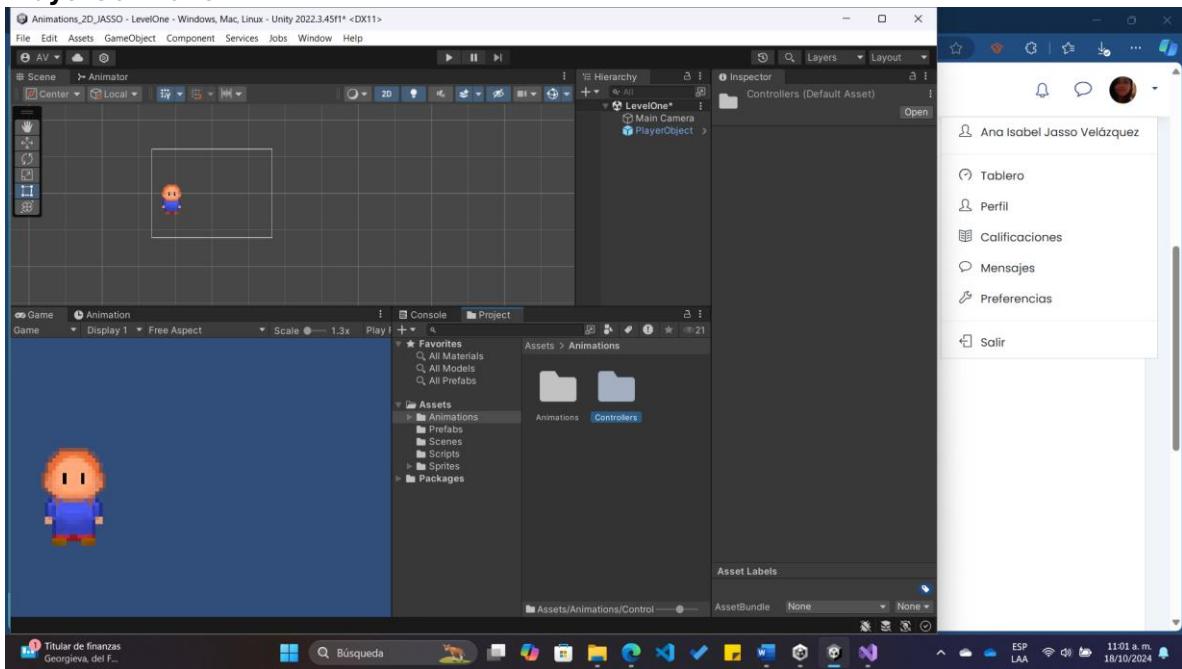


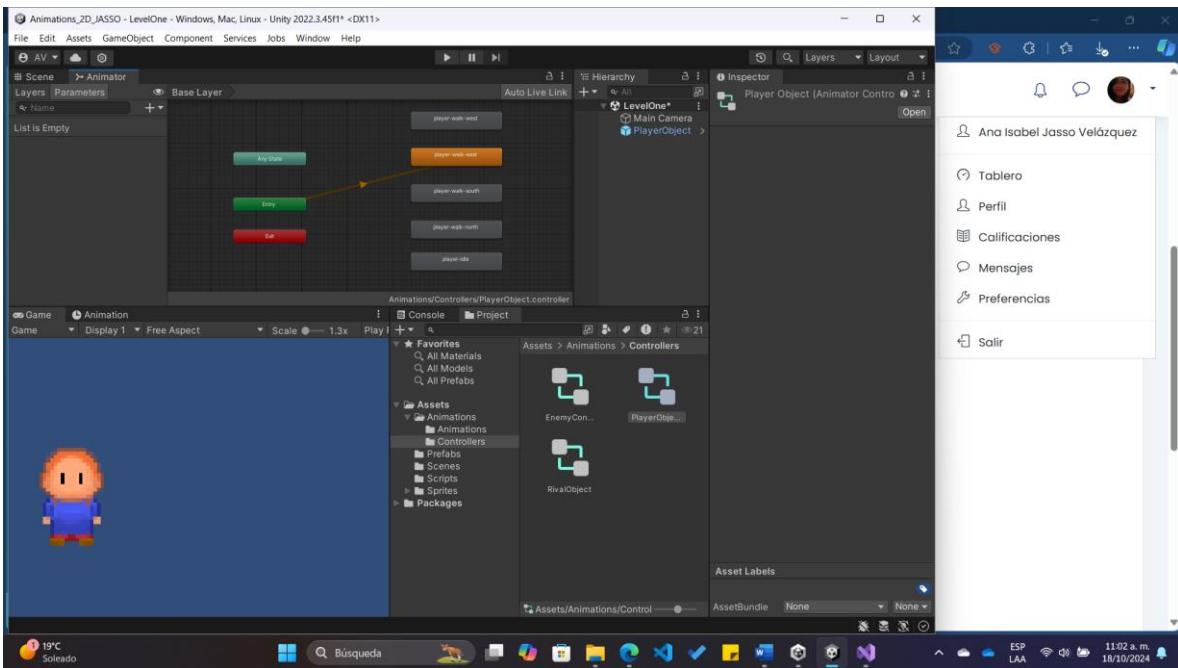
- ← Ahora presione el botón Play. Deberías ver a nuestro personaje de jugador caminando en su lugar. Presione las teclas de flecha o W, A, S, D en su teclado y mírala moverse.



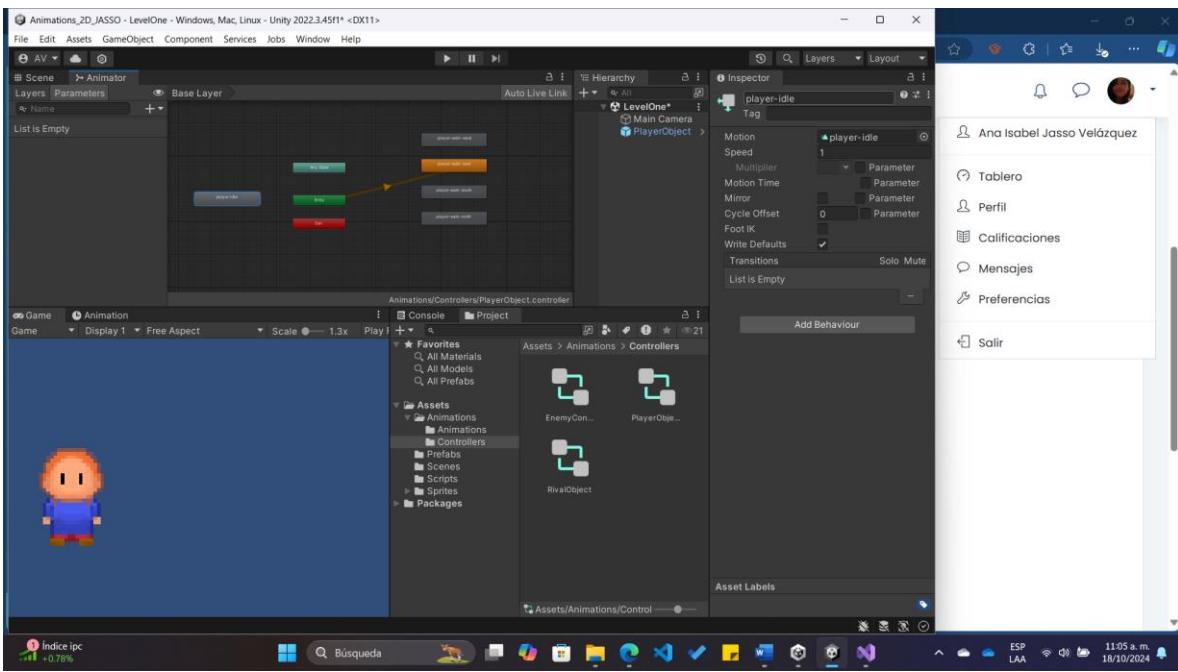
Estados y Animaciones

Vaya a la carpeta Animations > Controllers y haga doble clic en el Objeto PlayerController.

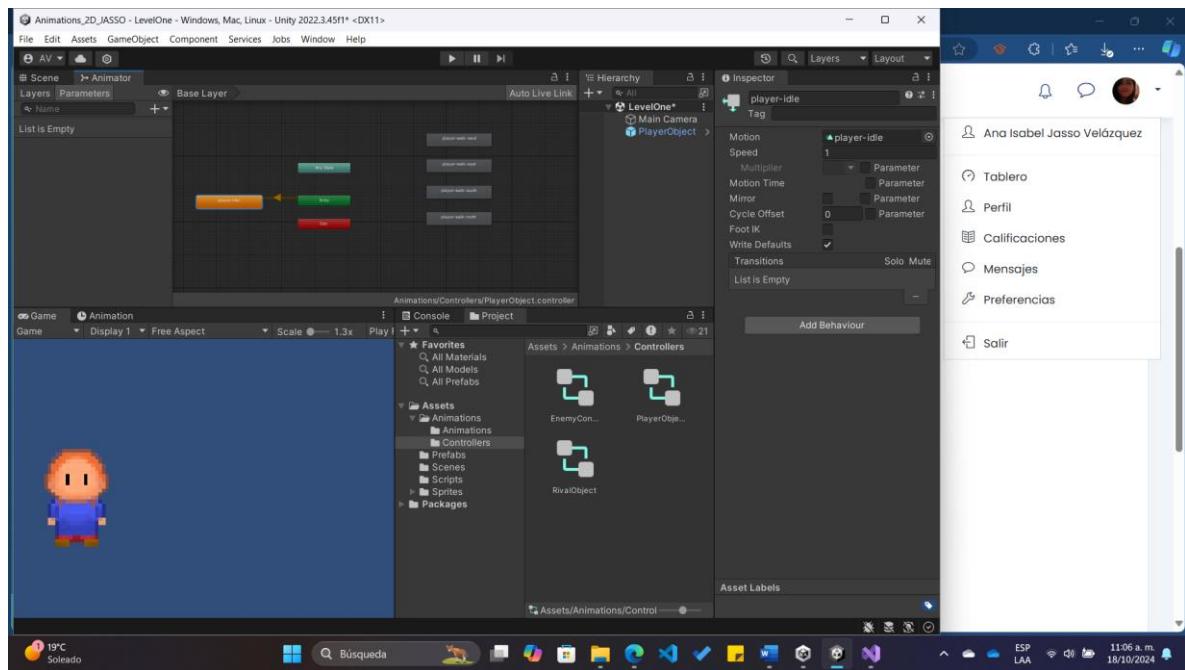
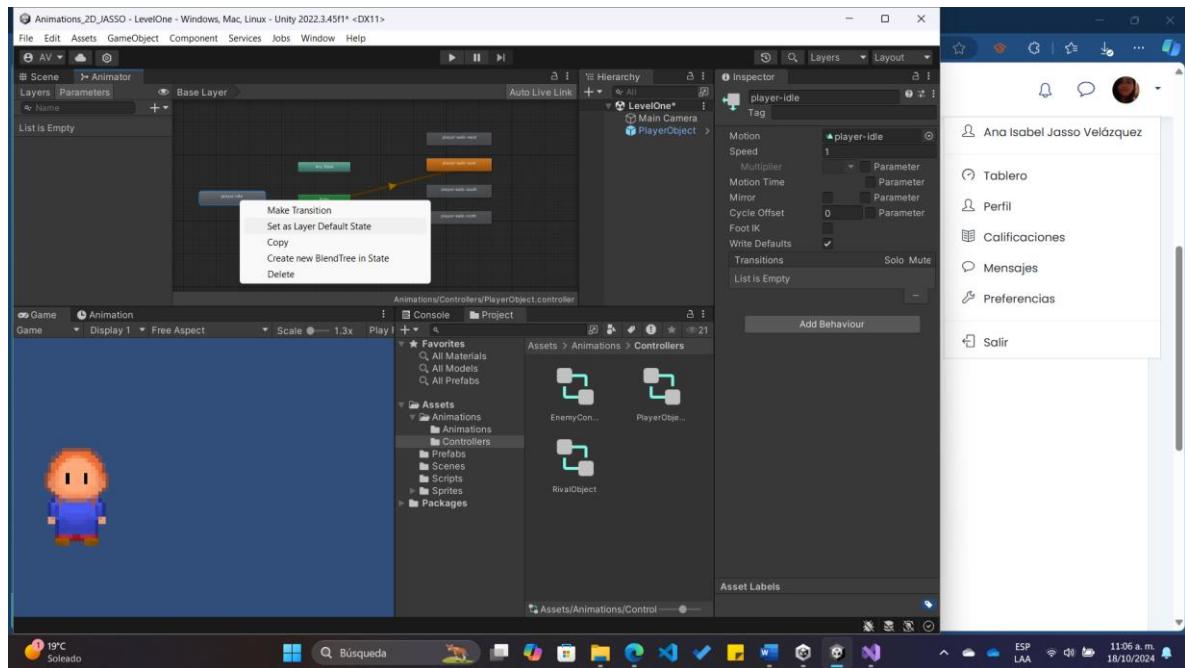




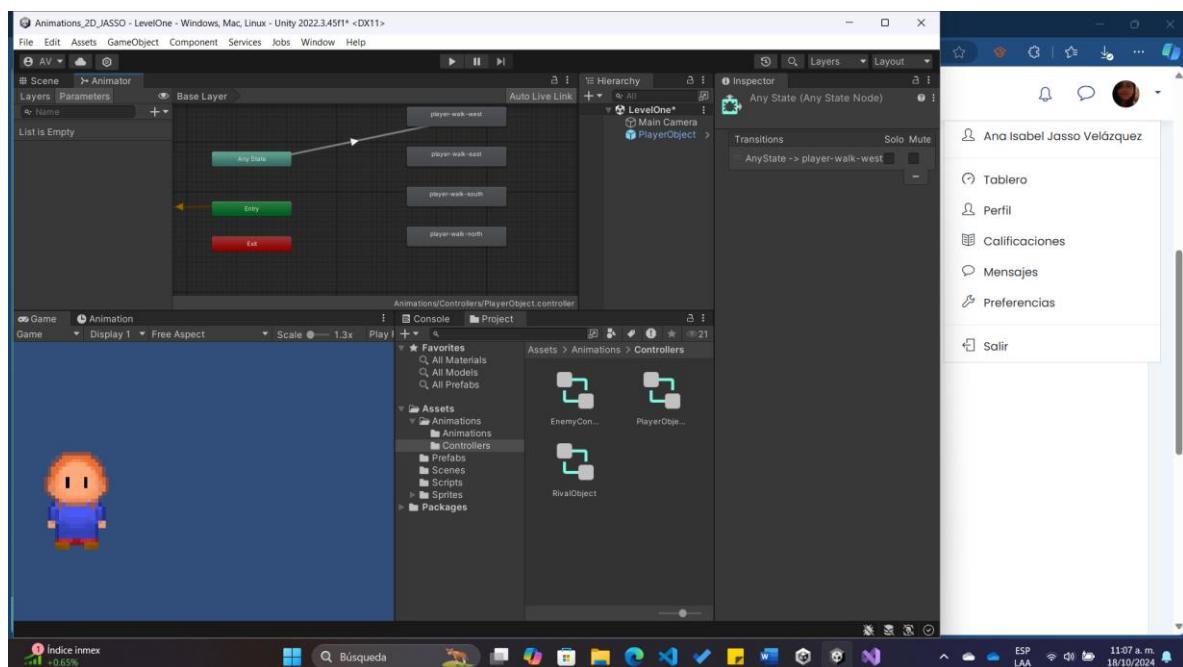
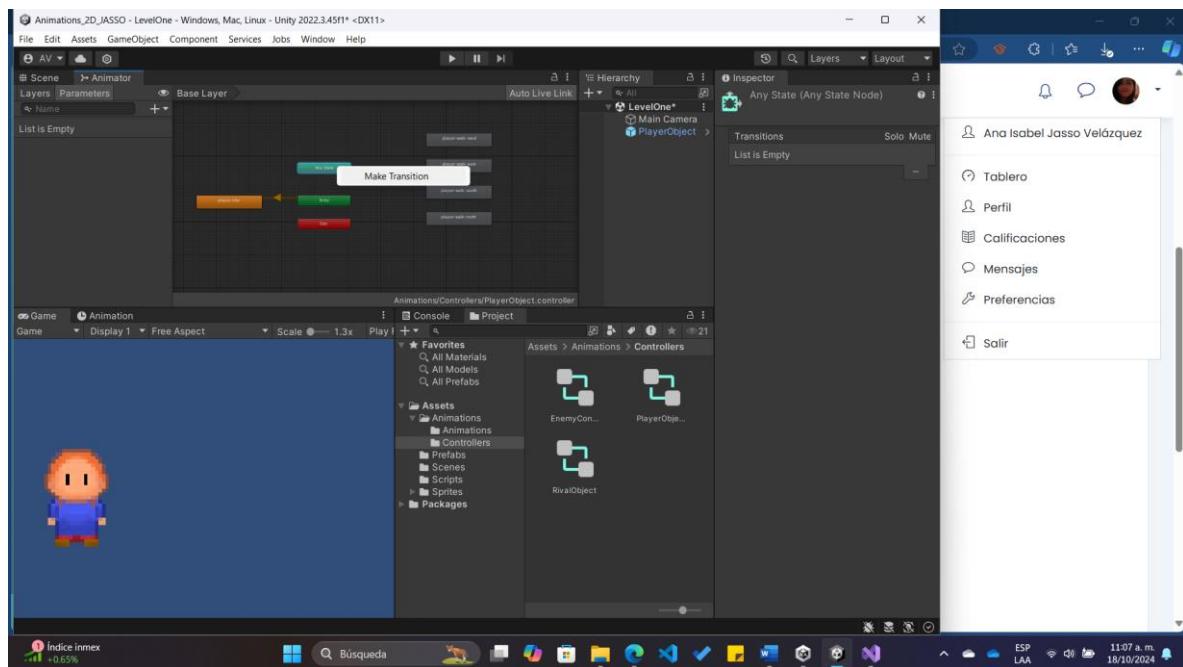
Haga clic y arrastre sus objetos de estado de animación hasta que aparezcan en la pantalla, con el reproductor inactivo a un lado, y las animaciones de caminatas de jugador agrupadas



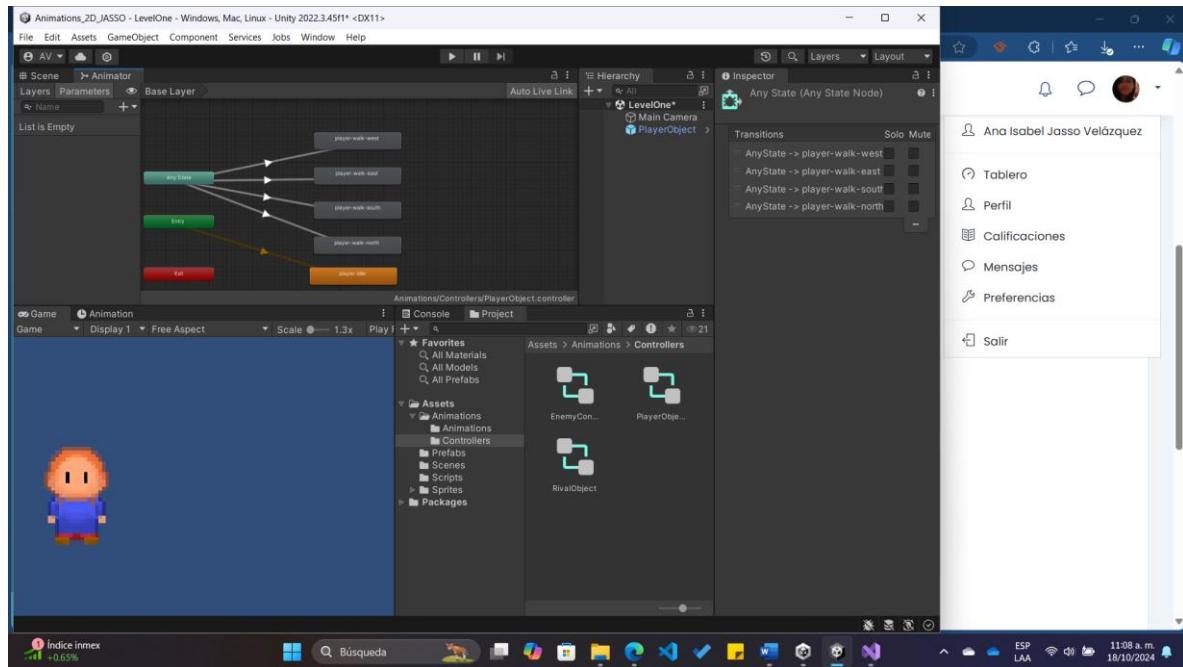
El color naranja indica que es el estado predeterminado para este Animator. Seleccione y, a continuación, haga clic con el botón derecho en el estado de animación "player-idle" y seleccione "Set as Layer Default State"



Ahora seleccione y haga clic con el botón derecho en "Any State" y seleccione "Make Transition" Aparecerá una línea con una flecha adjunta y siguiendo alrededor de su ratón. Haga click en "player-walk-east" para crear una transición entre el objeto Any State y player-walk-east.

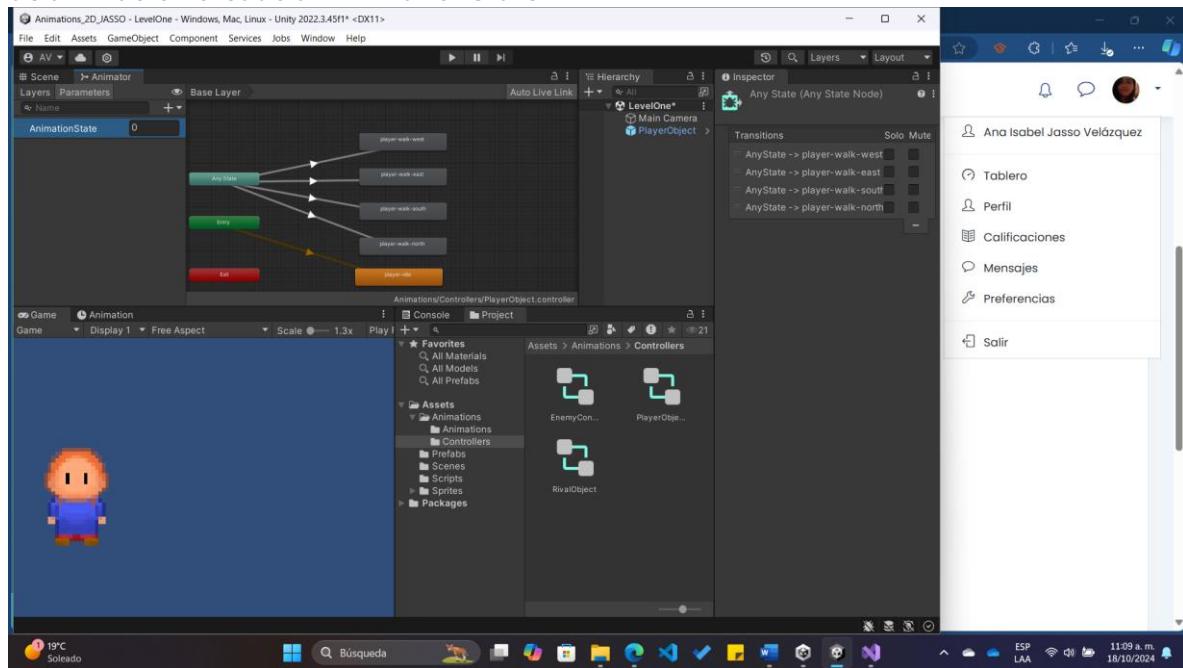


Debe crear un total de cinco flechas de transición blancas que apunten desde Any State a los cuatro estados de animación de caminata del jugador y al jugador inactivo player-idle. También debería haber un estado predeterminado de color naranja flecha desde el estado de animación de entrada, que conduce a la animación de jugador inactivo Estado.

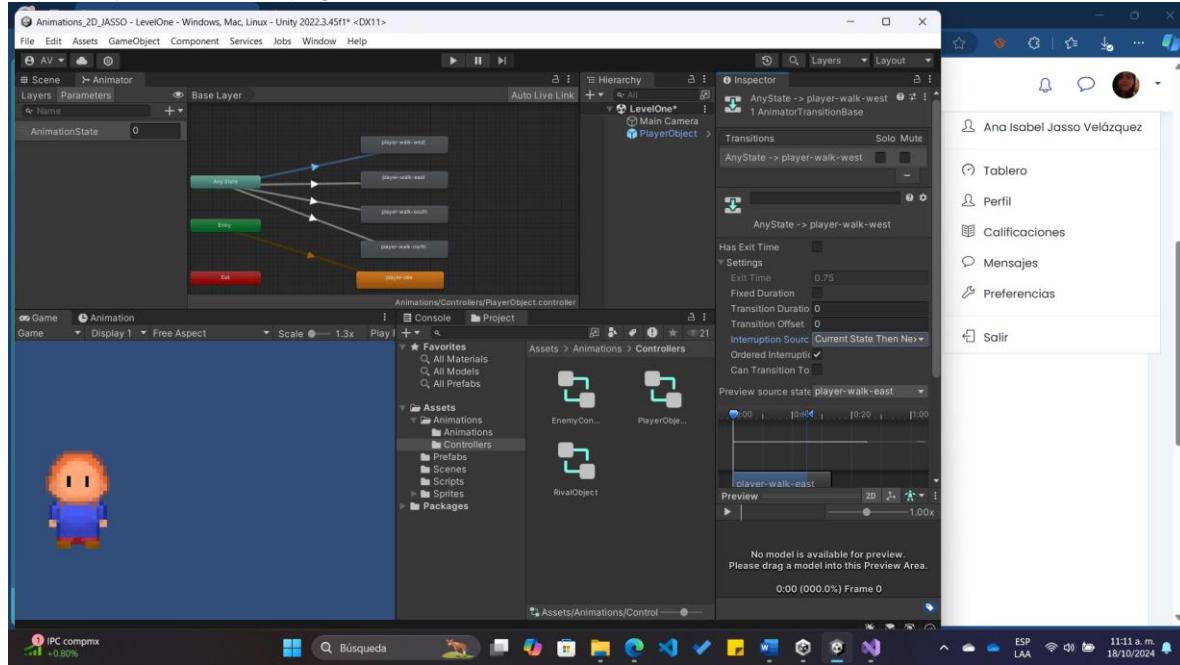


Parámetros de estado de Animación

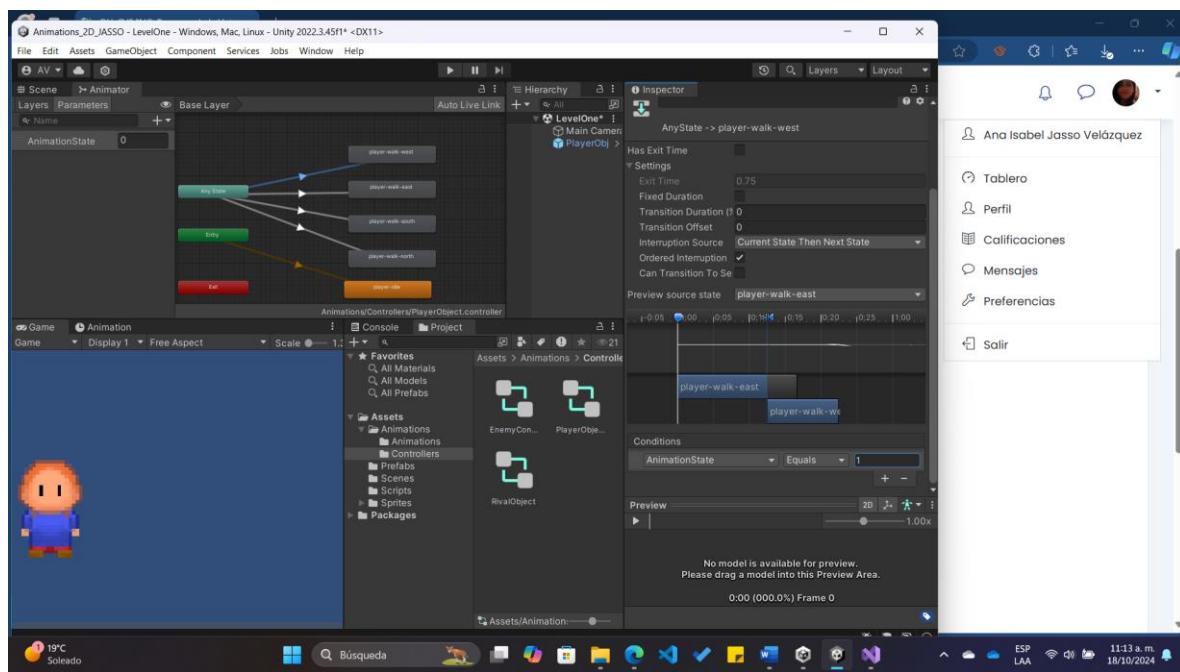
Seleccione la pestaña Parámetros en el lado izquierdo de la Ventana Animator. Presione el símbolo + y seleccione "Int" en la lista desplegable. Cambie el nombre del parámetro de animación creado a "AnimationState".



Desmarque Has Exit Time porque queremos interrumpir una animación si nuestro usuario presiona una tecla diferente. Si dejamos marcado Has Exit Time, entonces la animación tendría que terminar de reproducirse hasta que el % ingresado en la salida Exit time, antes de que pueda comenzar la siguiente, y eso daría como resultado una mala experiencia del jugador.



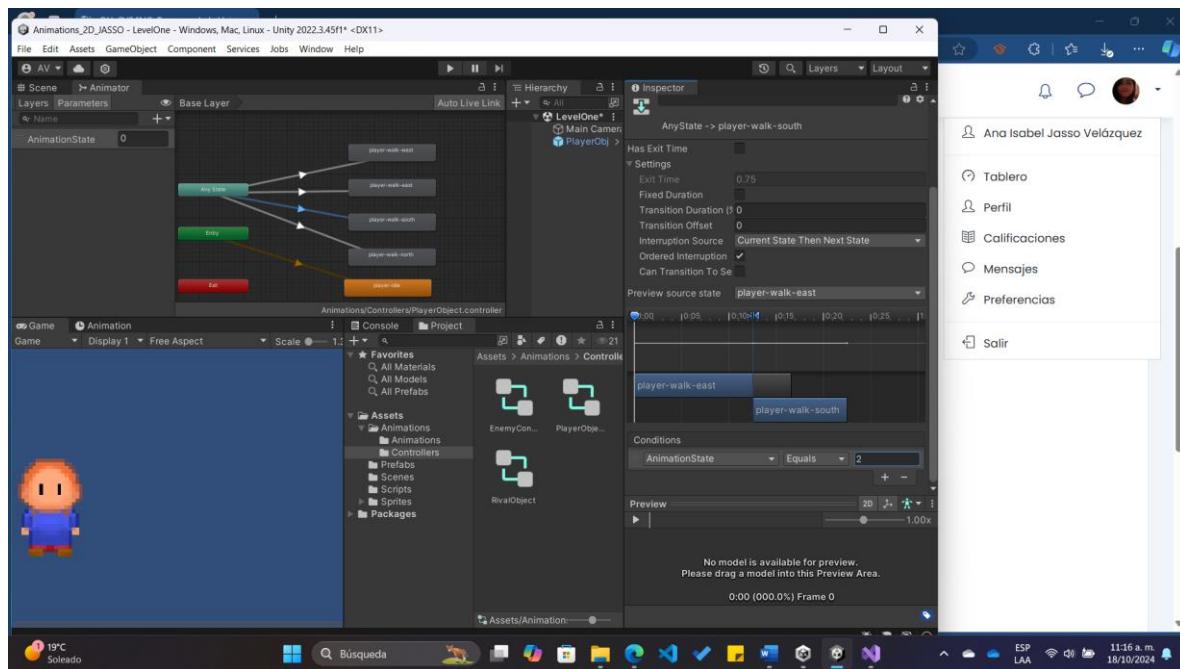
En la parte inferior del inspector, verá un área titulada "Conditions". Haga clic en el símbolo + en la parte inferior derecha y seleccione AnimationState ► Equals, e ingrese 1

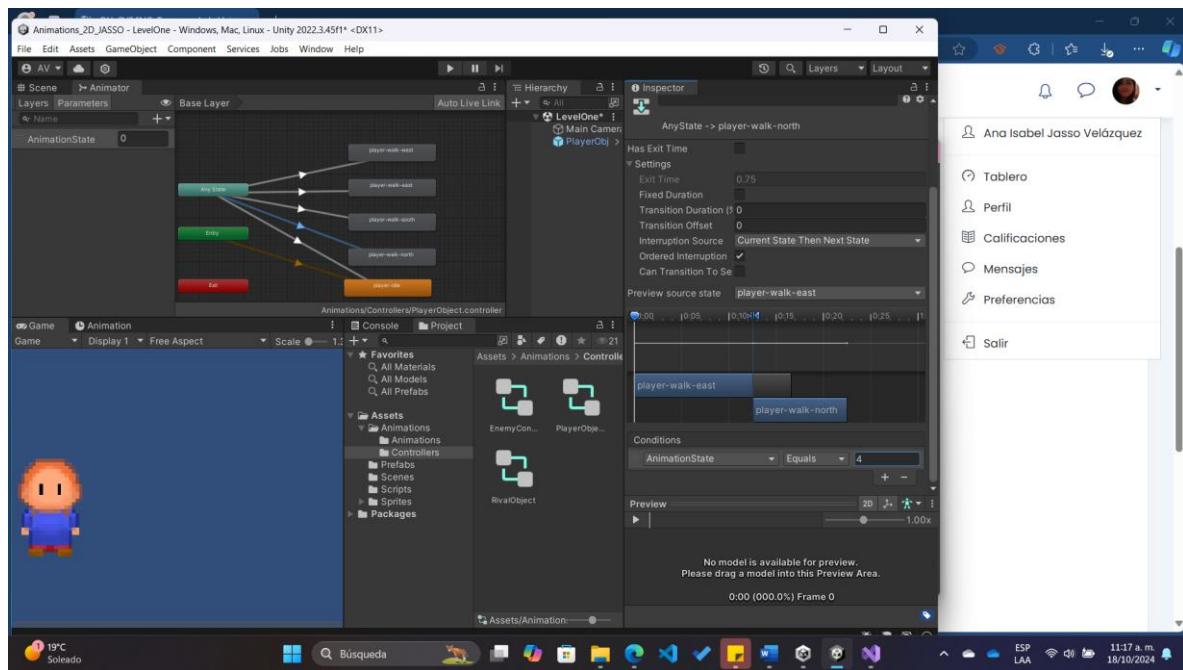
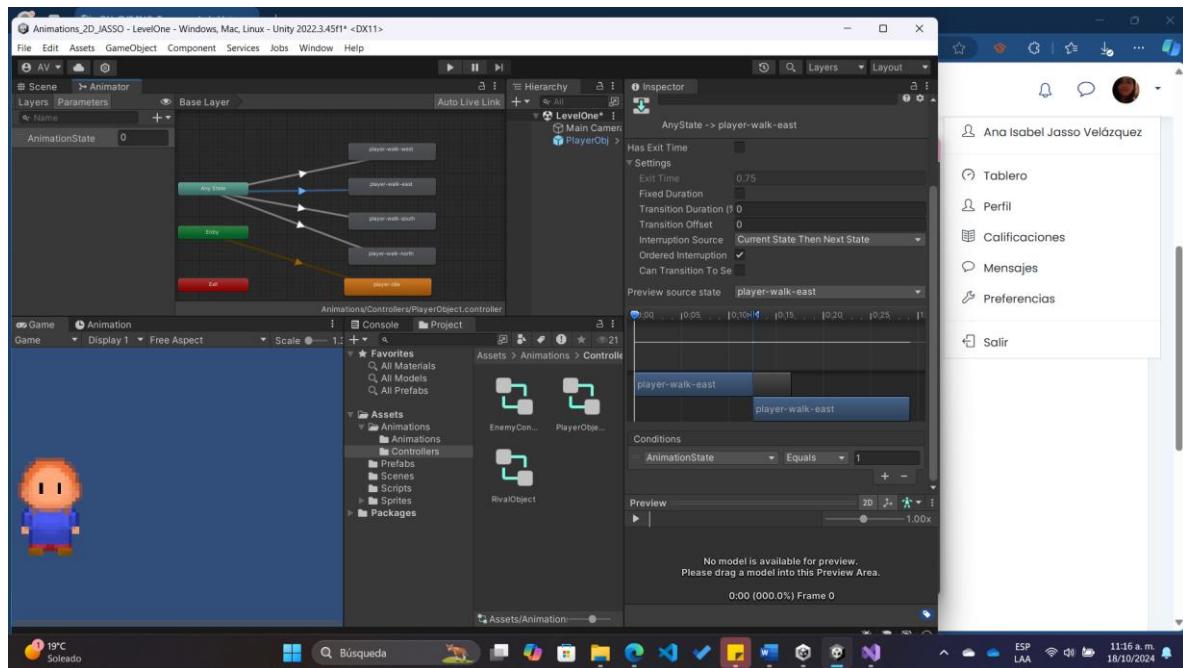


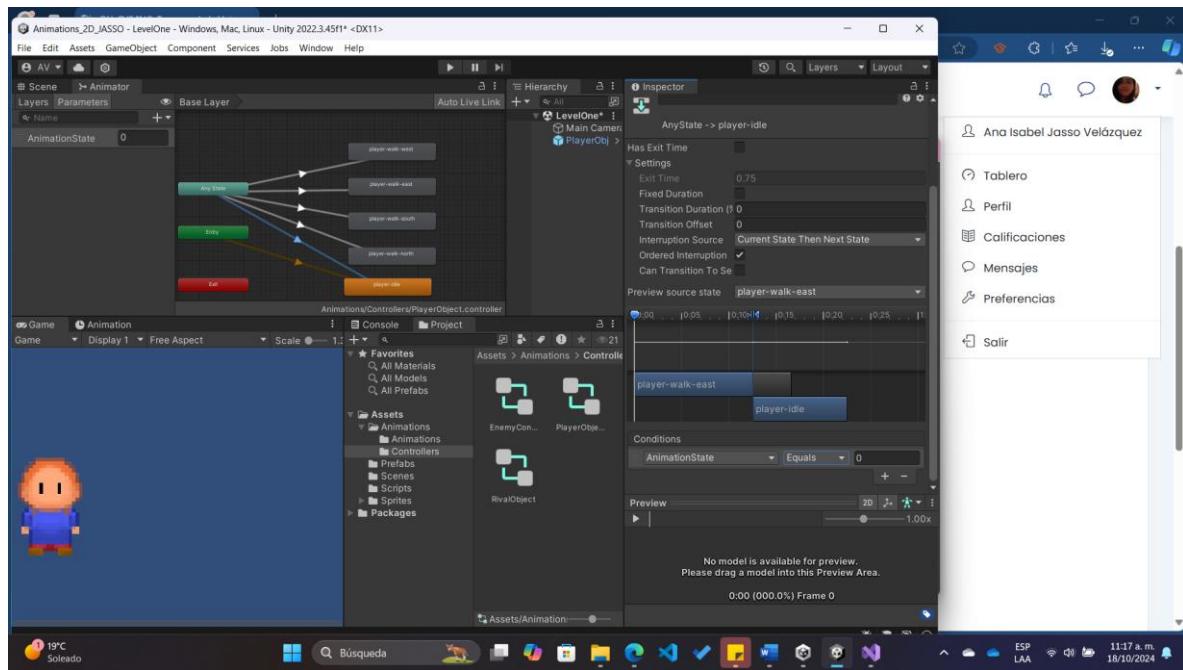
Transición	Condición
Any State to player-walk-east	AnimationState Equals = 1
Any State to player-walk-west	AnimationState Equals = 3
Any State to player-walk-north	AnimationState Equals = 4
Any State to player-walk-south	AnimationState Equals = 2
Any State to player-idle	AnimationState Equals = 0

Lo siguiente que vamos a hacer es establecer que el parámetro AnimationState igual a 1 en nuestro script. Regrese a Visual Studio y abra nuestro Script MovementController.cs.

Estados restantes:

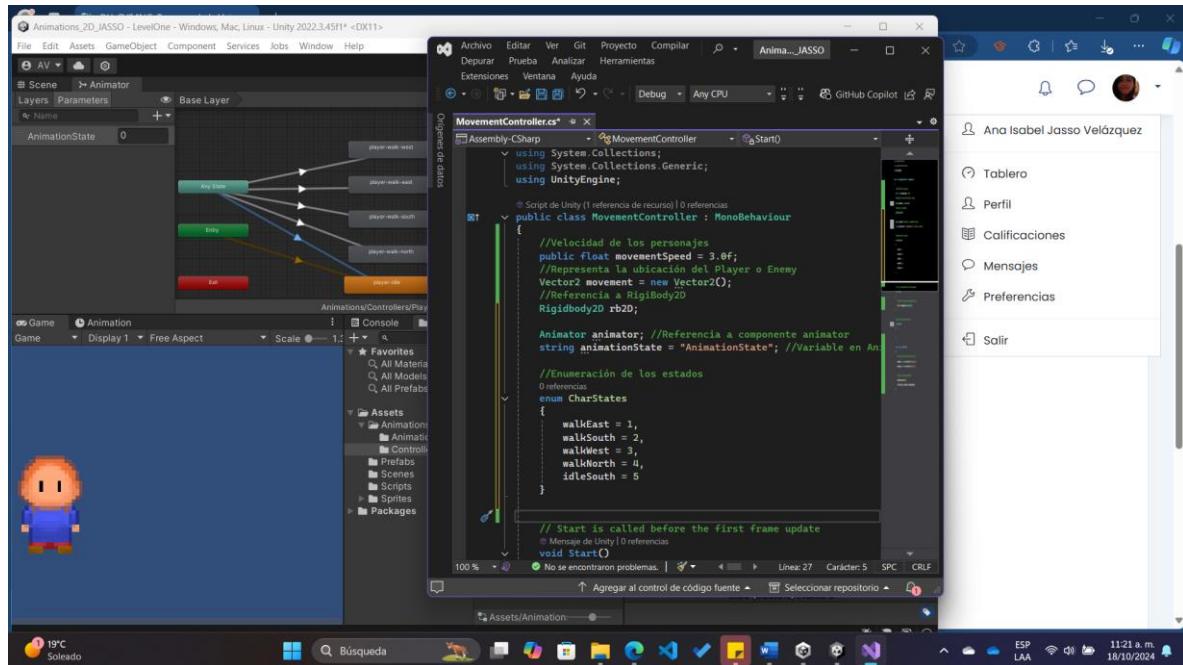




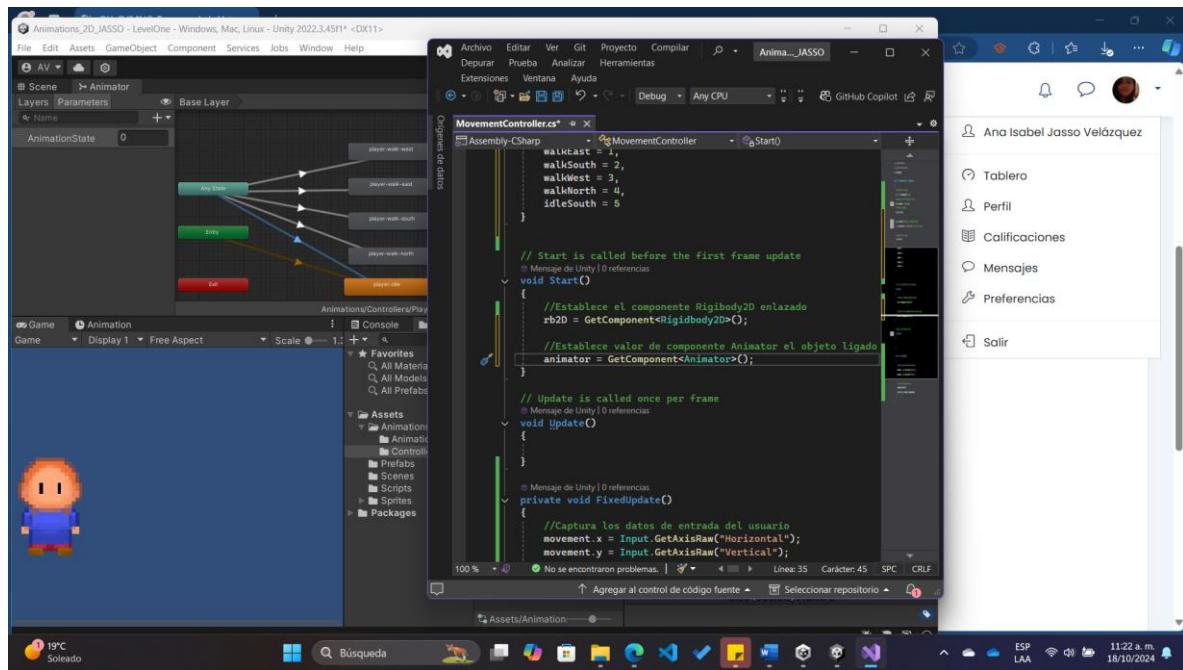


Regresar a Visual Studio MovementController.cs

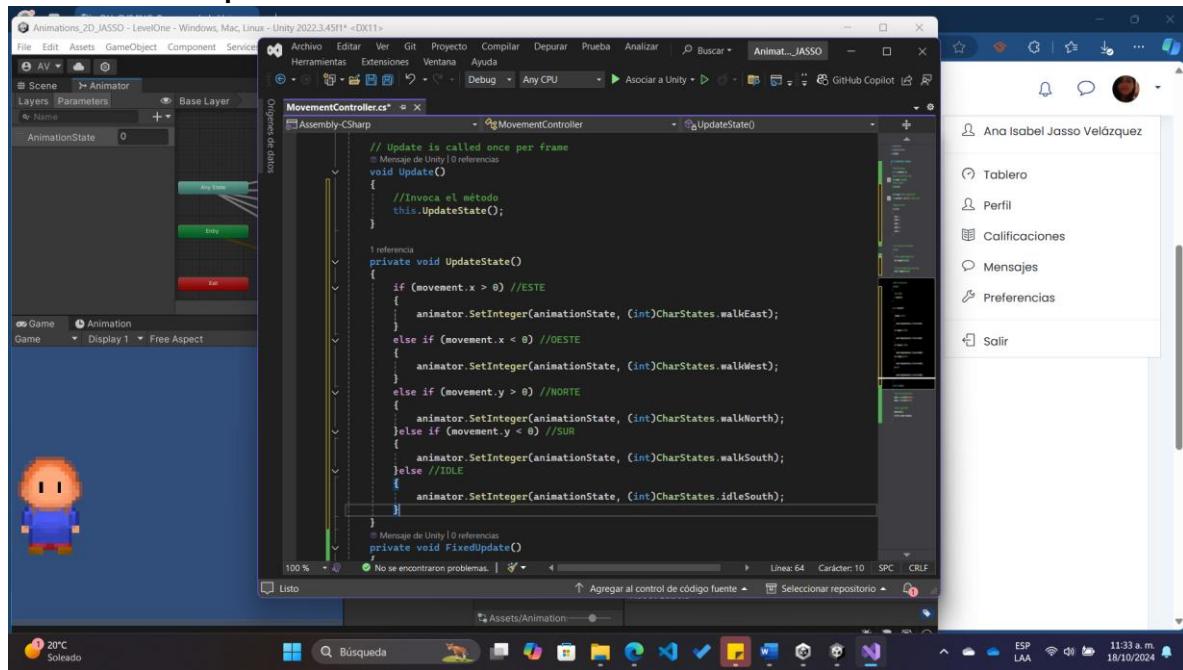
Agreguemos la referencia del objeto Animator; referencia de la variable qué guarda los estados y la definición de los estados



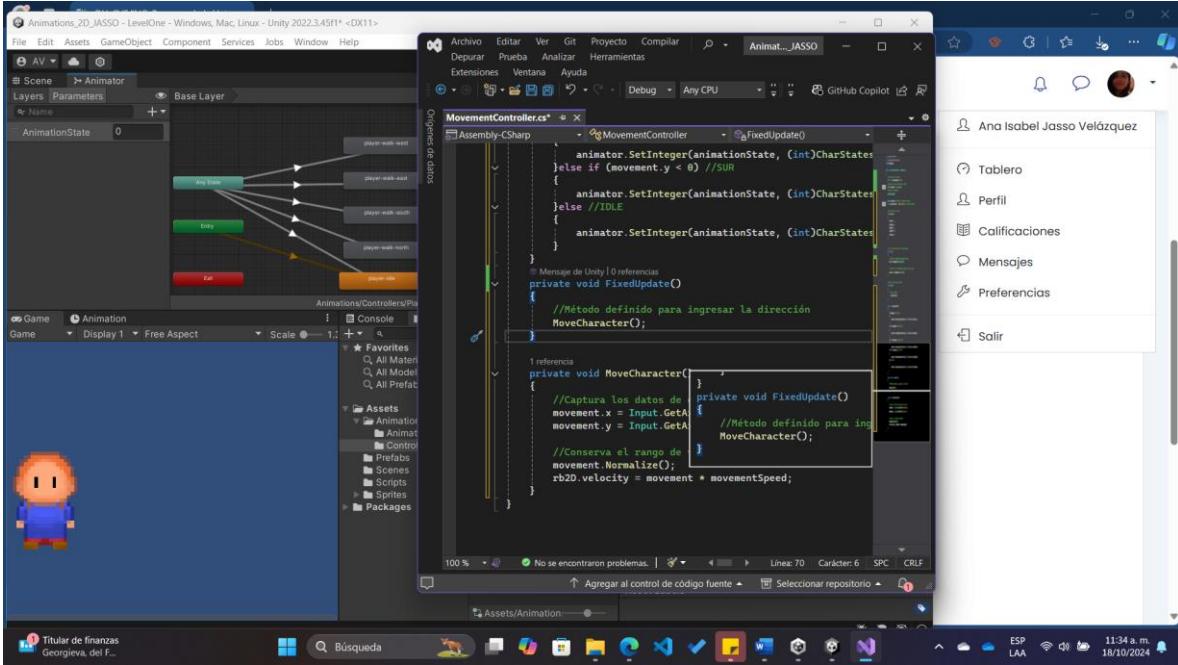
Inicializamos en el método start el valor del componente Animator del objeto enlazado



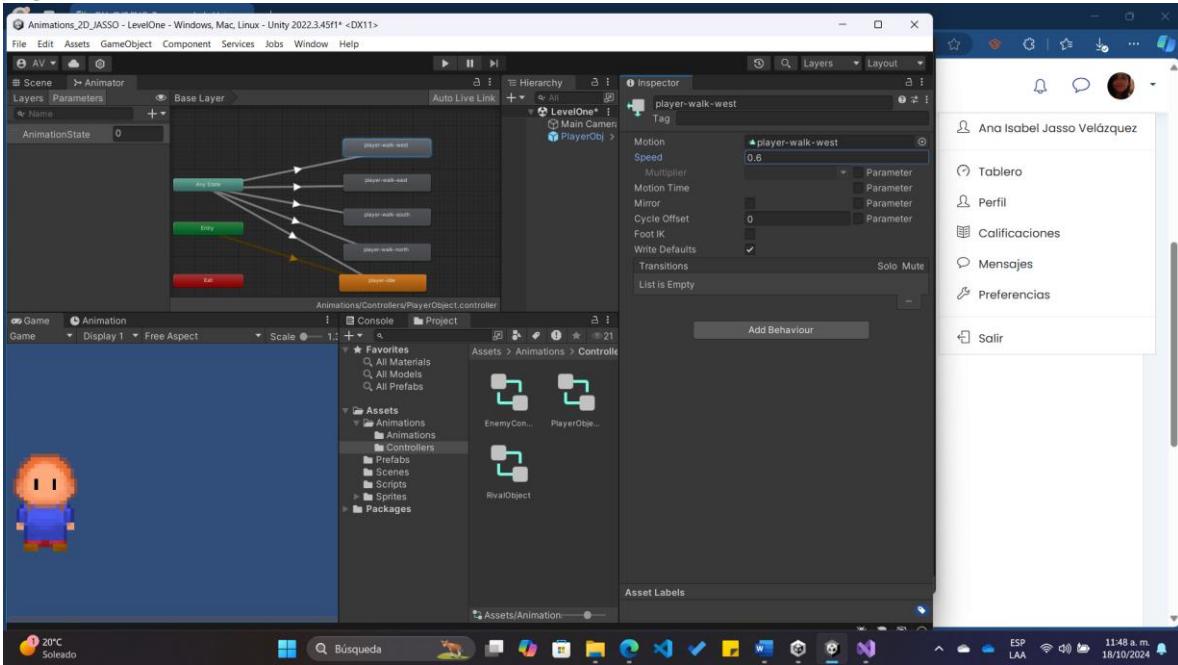
Modifiquemos el método Update donde se invoca a otro método qué define en base a las entradas WASD por el usuario

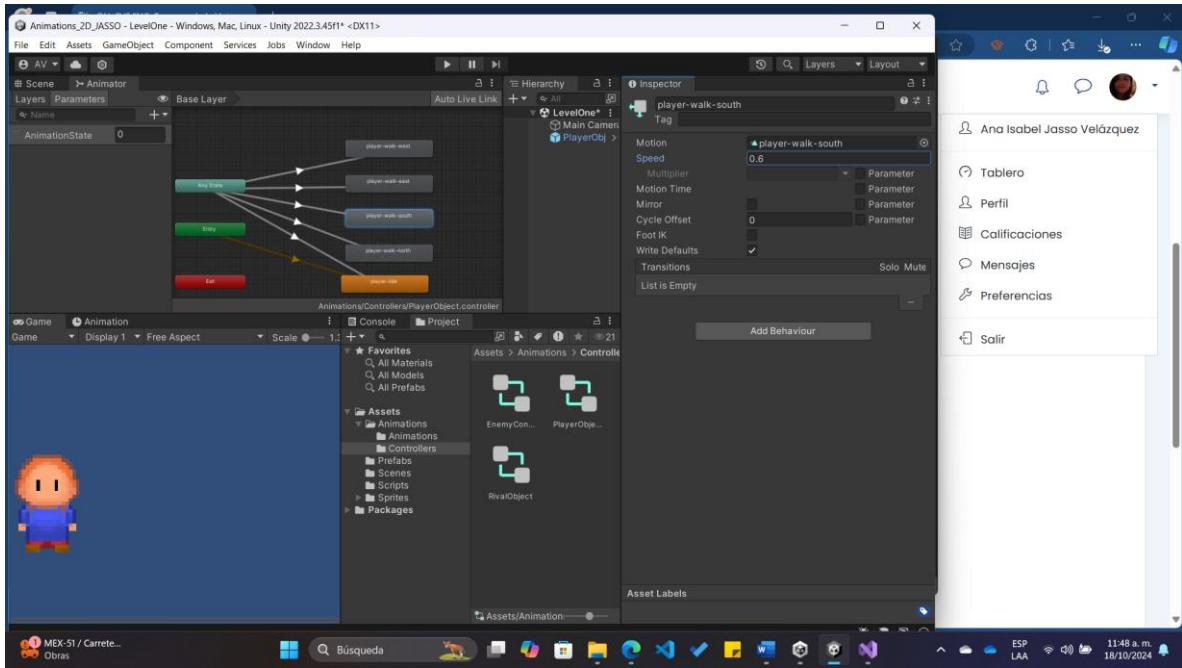
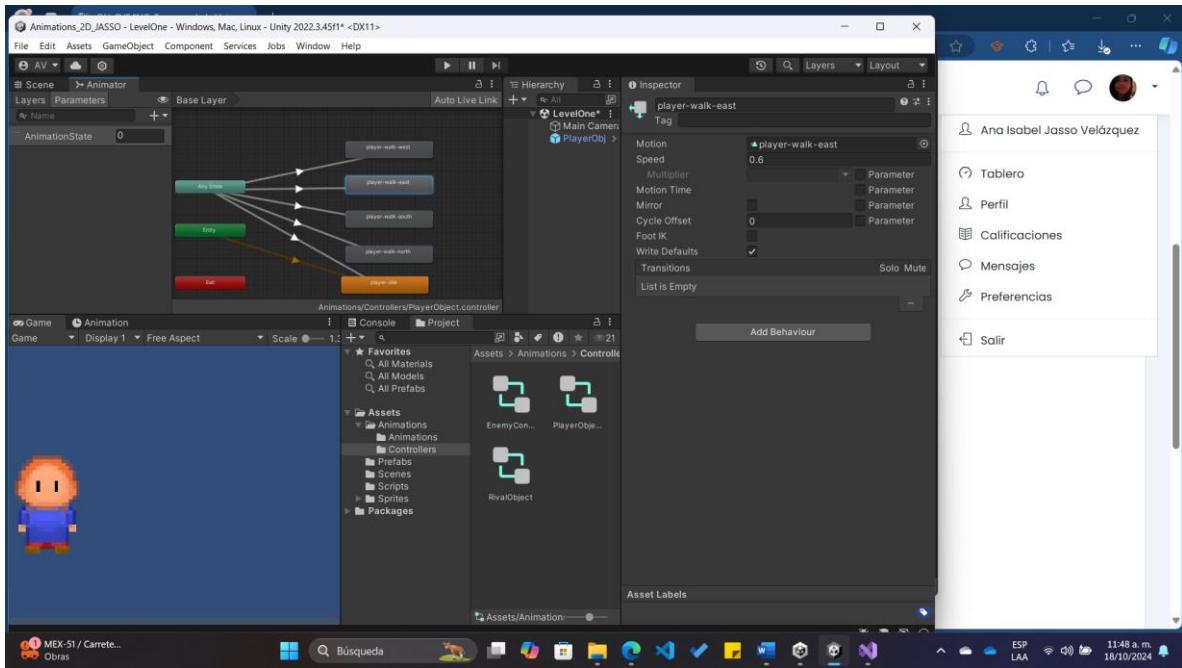


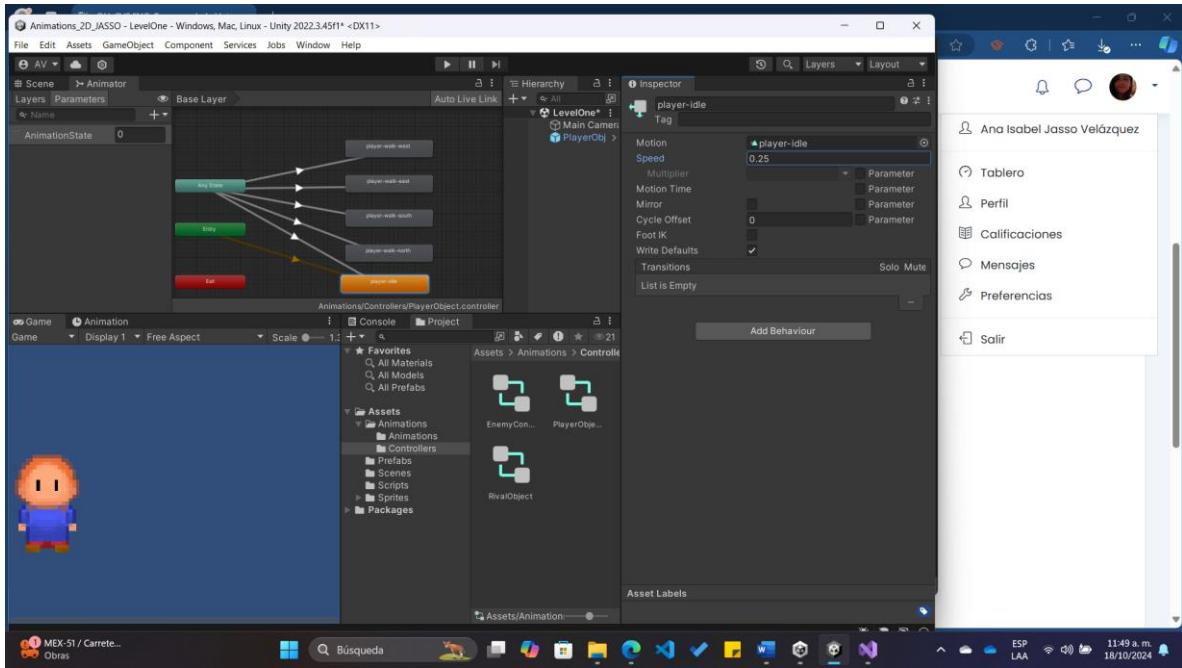
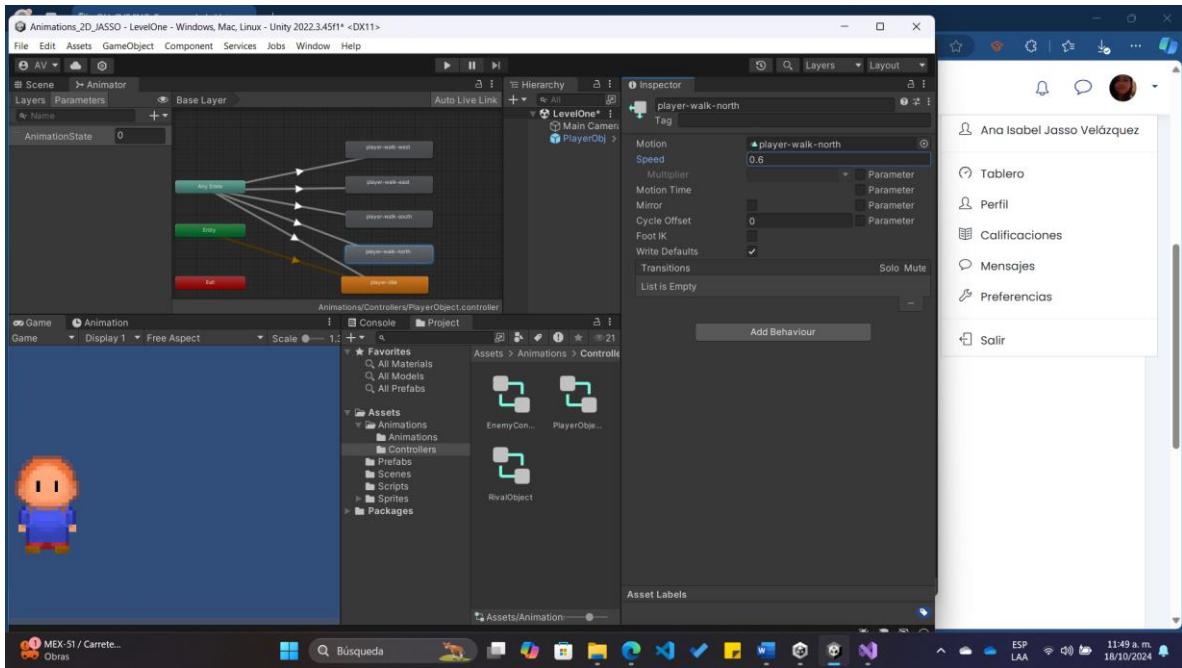
Modifiquemos el método FixedUpdate



Seleccione cada estado de animación de caminata del jugador objeto y ajuste la velocidad a 0,6, y ajuste player-idle a 0.25. Esto hará que las animaciones de nuestros jugadores se vean bien.







Ahora ha configurado una gran parte de las animaciones del reproductor necesarias para nuestro juego. Pulsamos el botón Play y movemos nuestro personaje por la pantalla con las teclas de flecha o W, A, S, D.

