Andres Juan Giraldo Vargas – Joel Esteban Peña Pinzón

1. El Legacy input system (Input manager) es el input system tradicional y nativo de Unity, por lo que no necesita instalación adicional. Este sistema se caracteriza por requerir una configuración manual de entradas y botones en las configuraciones del proyecto.

Este input system brinda facilidad para los inputs en proyectos simple y pequeños, ya que su configuración manual es sencilla y no conlleva mucho tiempo. Por ser el input system tradicional de Unity, al momento de configurarlo, se podrán encontrar mucha documentación oficial y tutoriales en internet para sacarle el mayor provecho.

Este sistema presenta algunas limitaciones, más en específico con múltiples entradas y con características avanzadas (como gamepads o rampeo de controles).

No es una buena opción cuando el proyecto es grande y complejo, porque sus configuraciones no son tan amplias y avanzadas para casos dones los jugadores quieran usar diferentes dispositivos de entradas diferentes a los típicos botones del teclado, además de que tampoco presenta tanto rendimiento ni fidelidad en las entradas.

El new input system, presenta un sistema de entradas más moderno, fiel y avanzado para unity, pero este no es nativo al entorno, por lo que tendrá que ser instalado por aparte. Este nuevo sistema es perfecto para proyectos más grandes y complejos pues permite la utilización de múltiples entradas y de características más avanzadas (gamepads, por ejemplo) de manera nativa.

Aunque este sistema presenta más fidelidad en las entradas y en los eventos a comparación del input manager, este también conlleva más aprendizaje para su correcto uso y más tiempo para su configuración debido a que tiene más versatilidad. El hecho de que este sistema no sea el sistema nativo de unity, hace que algunos assets y complementos de terceros no sean compatibles con este.

A)

* + 1. Crear un action map desde “assets” desde el panel del proyecto.
    2. Desde el panel inspector, hay que clickear en el botón “add action” que se encuentra en la sección “actions”.
    3. En la sección de “bindings” ,de la acción creada, hay que clickear en el botón “add binding”. Aquí hay que configurar el botón o tecla que activara la acción.
    4. En la sección “action properties”, de la acción, se puede configurar características adicionales de esta (como la sensibilidad)
    5. Con las características ya configuradas, se deben de guardar y aplicar en el proyecto desde el botón “apply” en “file”. Ahora hay que implementar los inputs en el código, empezando por usar la referencia correcta en las clases necesarias (como por ejemplo: public InputActionAsset “nombreInput”;)
    6. Ahora sólo falta obtener las acciones especificas desde el código (InputAction moveAction = playerInput.FindAction("Move"); ejemplo de una acción llamada “move”) y escribir el código que les dará los valores referentes a la(s) variable(s) que modifiquen (ejes/axis pueden ser algunas variables con las que interactúen) desde los eventos respectivos para las entradas que hayan.

B & C)

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.InputSystem;

public class PlayerMovement : MonoBehaviour

{

public InputActionAsset playerInput;

public Rigidbody rb;

public InputActionAsset playerJump;

private float vertical;

private float speed = 8f;

Vector2 moveDirection = Vector2.zero;

private void OnEnable()

{

playerInput.Enable();

playerJump.Enable();

}

private void OnDisable()

{

playerInput.Disable();

playerJump.Disable();

}

void Update()

{

moveDirection = playerInput.ReadValue<Vector2>();

rb.velocity = new Vector3(vertical \* speed, rb.velocity.y);

}

public void jump(PlayerInput context)

{

if (context.performed)

{

rb.velocity = new Vector3(rb.velocity.y \* 0.5f);

}

}

}