DIRECCIÓN GENERAL DE UNIVERSIDADES TECNOLÓGICAS y POLITÉCNICAS

Universidad Tecnológica del Norte de Guanajuato
Organismo Público Descentralizado del Gobierno del Estado
"Educación y progreso para la vida"

INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN ÁREA ENTORNOS VIRTUALES Y NEGOCIOS DIGITALES.

"Programación de videojuegos 1"

Tutorial 1:

Documentación del tutorial 1

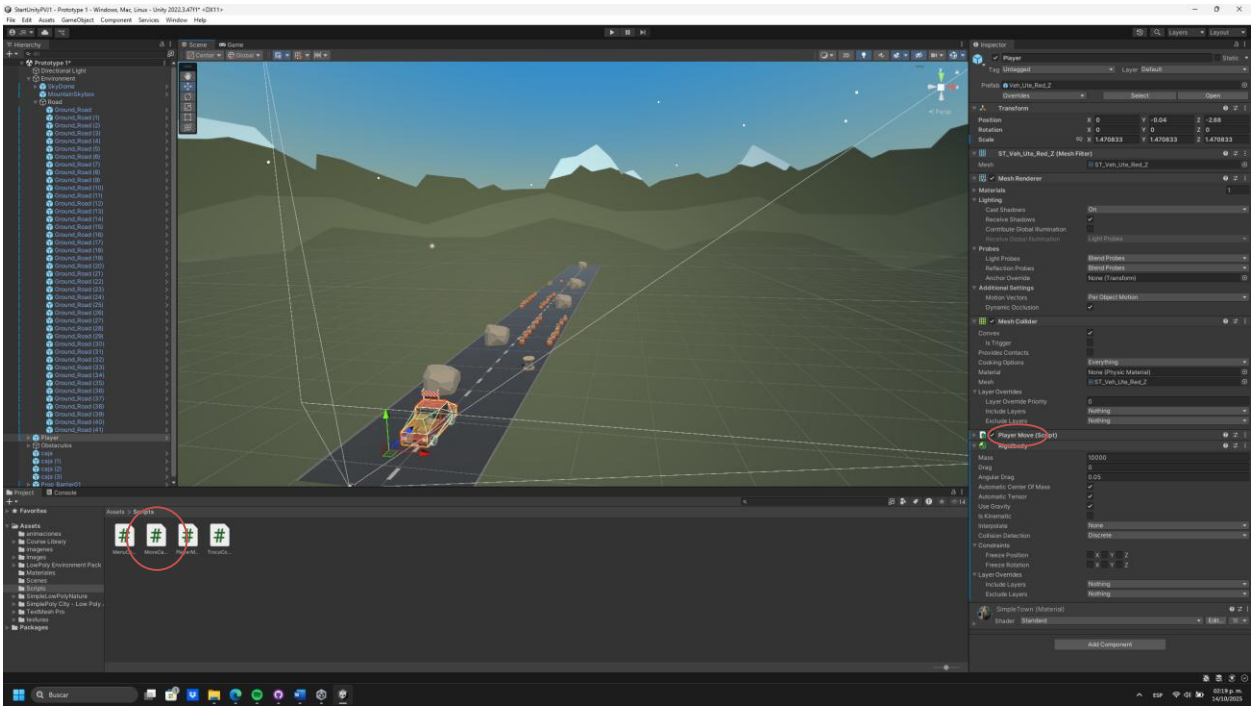
Profesor: Gabriel Barrón Rodríguez.

Alumno: Jesús Alberto Arriaga Ramírez
No. de Control: 1223100850

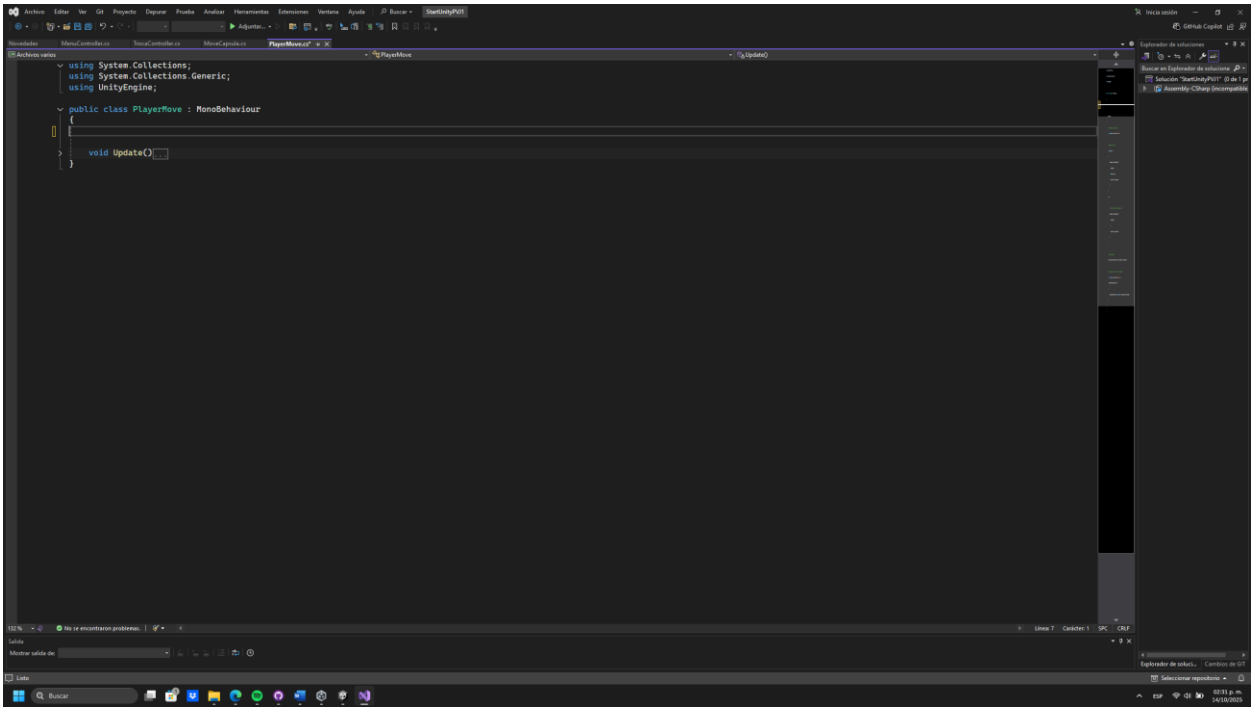
14 de octubre del 2025.

Leccion 1.2 Del pedal al metal

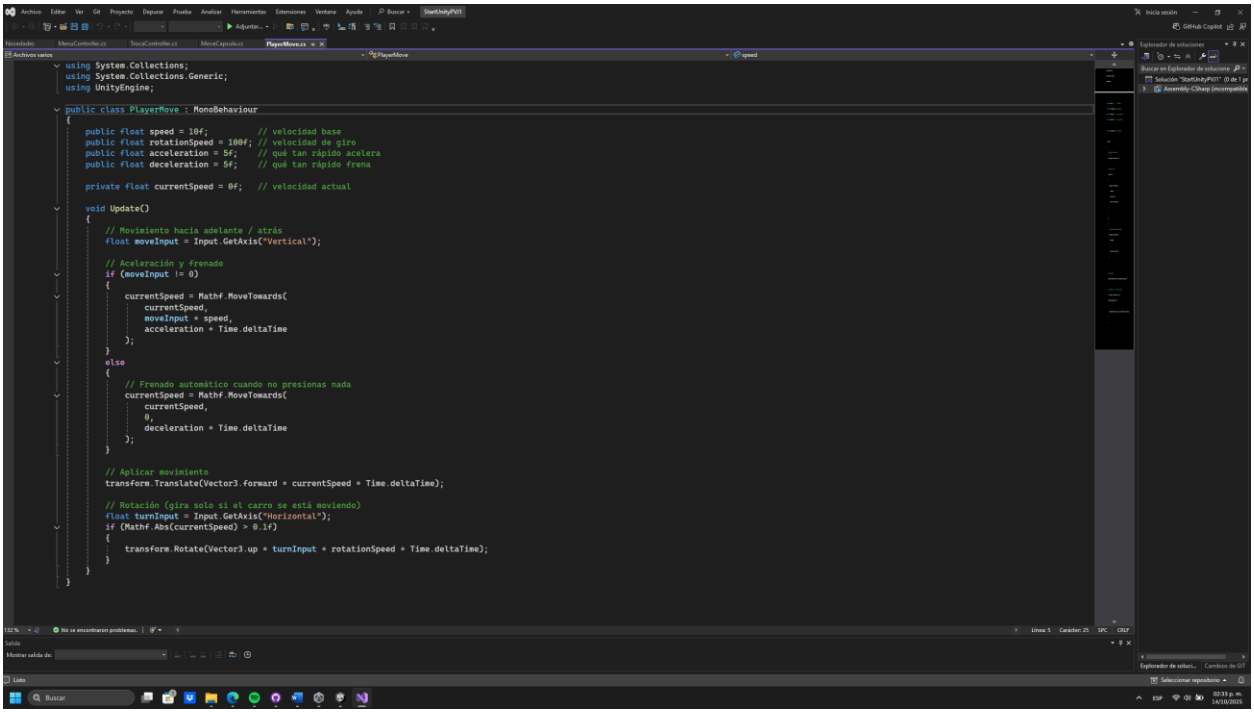
Para comenzar esta lección, crearemos nuestro primer Script de C# que controlará el movimiento del vehículo.



Para lograr que el vehículo se mueva, primero tenemos que abrir nuestro nuevo script y familiarizarnos con el entorno de desarrollo.



Ahora que contamos con el comentario que dice lo que VAMOS a programar, tenemos que escribir una línea de código que haga que nuestro vehículo avance.



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerMove : MonoBehaviour
{
    public float speed = 10f;          // velocidad base
    public float rotationSpeed = 100f; // velocidad de giro
    public float acceleration = 5f;    // qué tan rápido acelera
    public float deceleration = 5f;    // qué tan rápido frena

    private float currentSpeed = 0f;   // velocidad actual

    void Update()
    {
        // Movimiento hacia adelante / atrás
        float moveInput = Input.GetAxis("Vertical");

        // Aceleración y frenado
        if (moveInput != 0)
        {
            currentSpeed = Mathf.MoveTowards(
                currentSpeed,
                moveInput * speed,
                acceleration * Time.deltaTime
            );
        }
        else
        {
            // Frenado automático cuando no presionas nada
            currentSpeed = Mathf.MoveTowards(
                currentSpeed,
                0,
                deceleration * Time.deltaTime
            );
        }

        // Aplicar movimiento
        transform.Translate(Vector3.forward * currentSpeed * Time.deltaTime);

        // Rotación (gira solo si el carro se está moviendo)
        float turnInput = Input.GetAxis("Horizontal");
        if (Mathf.Abs(currentSpeed) > 0.1f)
        {
            transform.Rotate(Vector3.up * turnInput * rotationSpeed * Time.deltaTime);
        }
    }
}
```

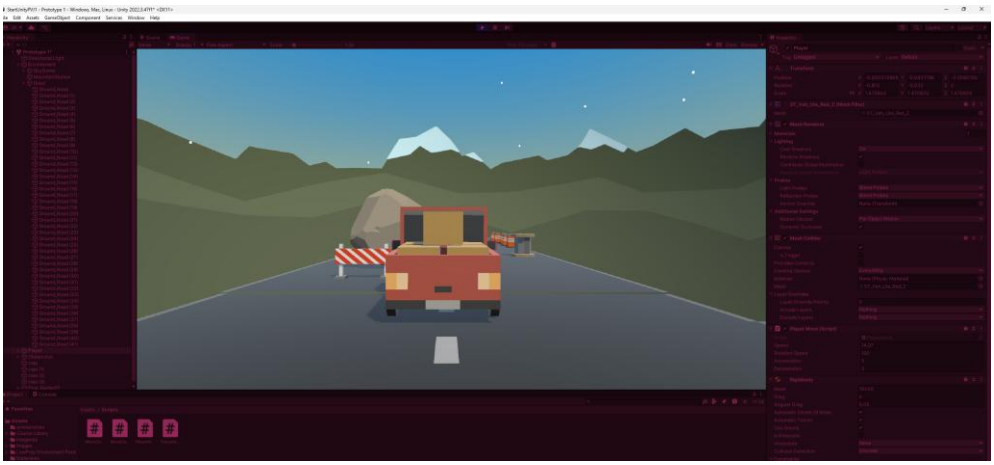
1. **Elimina** 0, 0, 1 del código que escribiste y autocomplétalo para **reemplazarlo** con ***Vector3.forward***.

```
        );
    }

    // Aplicar movimiento
    transform.Translate(Vector3.forward * currentSpeed * Time.deltaTime);
```

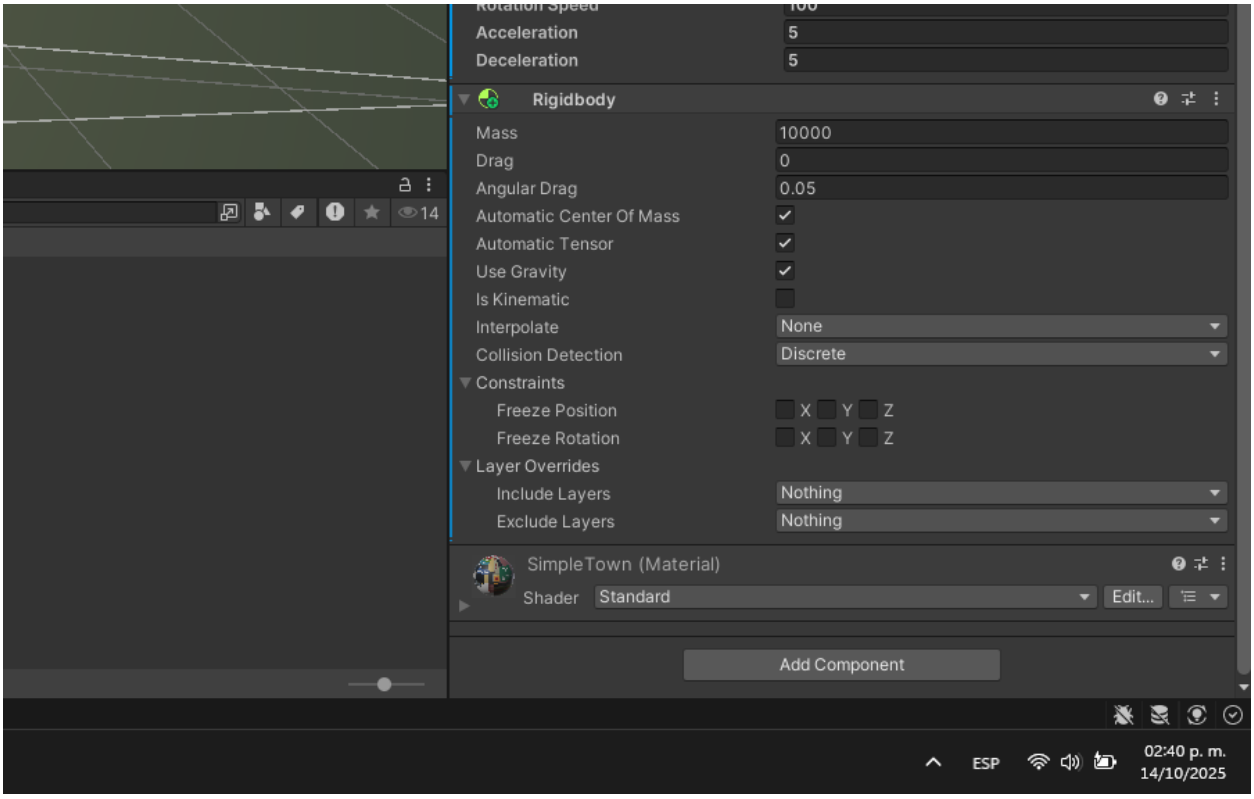
Ahora, la velocidad del vehículo está fuera de control. Debemos cambiar el código para arreglarlo.

1. Agrega **** Time.deltaTime*** y ejecuta el juego.
2. Añade **** 20*** y ejecuta el juego.



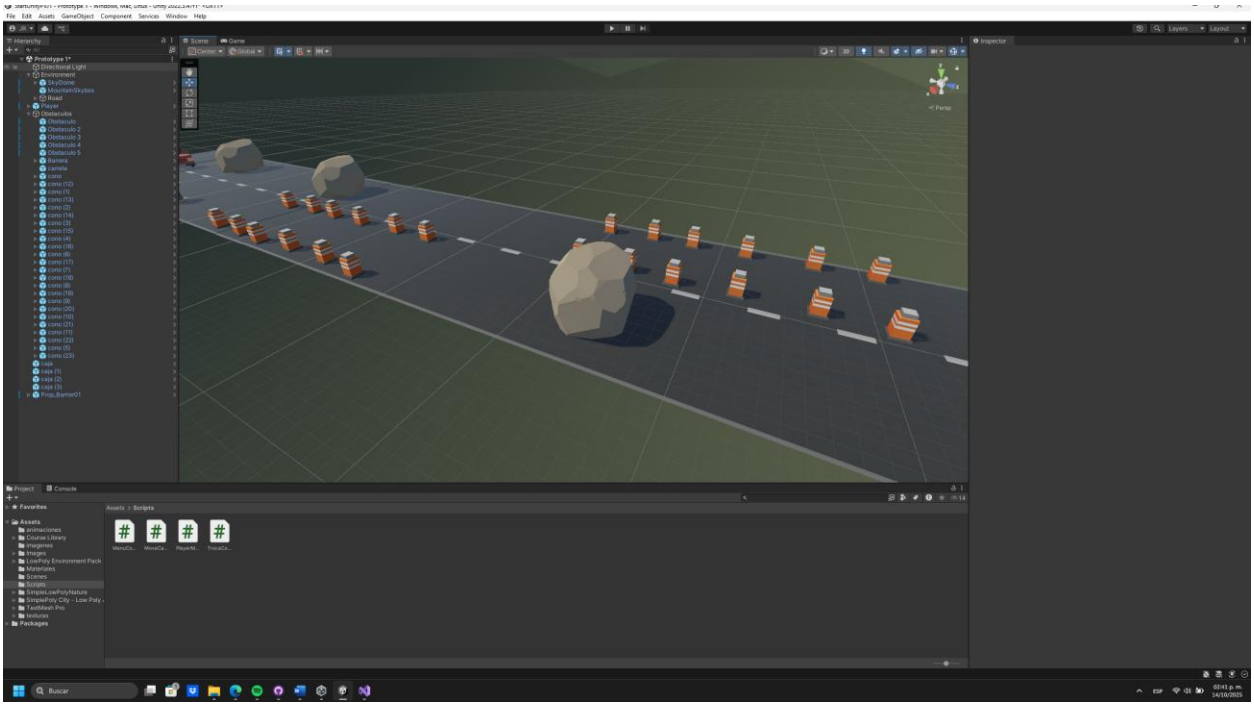
Ahora, el vehículo pasa a través la caja. Si queremos que sea más realista, tenemos que añadir física.

1. Selecciona el **Vehicle** y, en Hierarchy, haz clic en **Add Component (Agregar componente)** y selecciona **RigidBody.**
2. Selecciona el **Obstacle (Obstáculo)** y, en Hierarchy, haz clic en **Add Component** y selecciona **RigidBody.**
3. En las propiedades del componente RigidBody, aumenta la **Mass (Masa)** del Vehicle y el Obstacle a lo que deberían pesar en **kilogramos** y vuelve a probar.



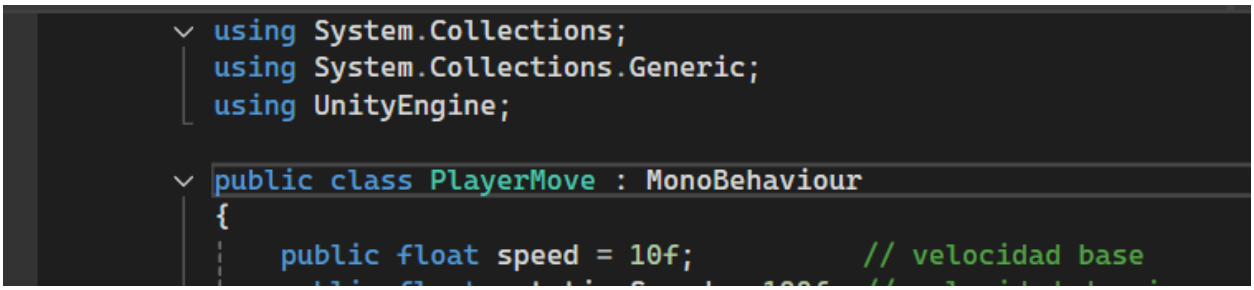
Por último, debemos duplicar el obstáculo para hacer que la carretera sea más desafiante para el vehículo.

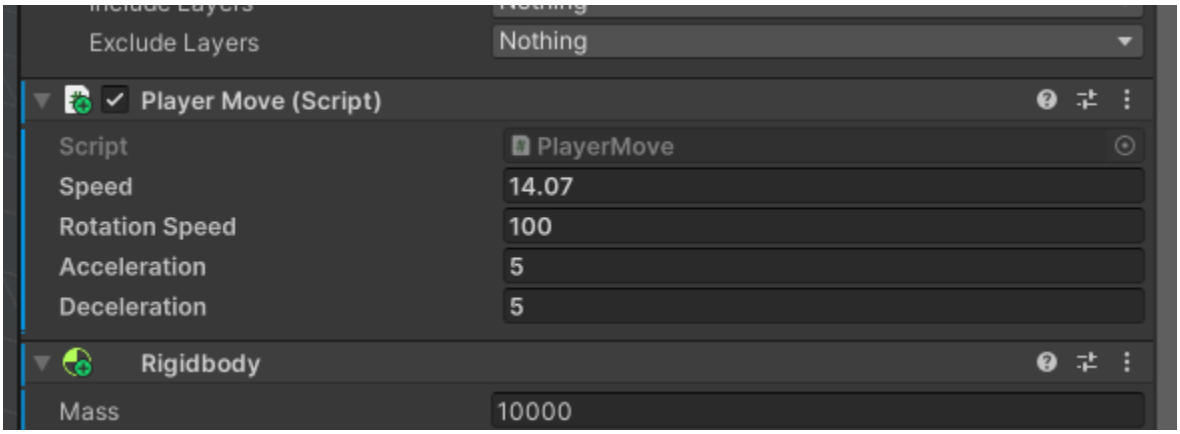
- Haz clic y arrastra el Obstacle hasta el final de la lista en Hierarchy.
- Presiona Ctrl/Cmd + D para duplicar el Obstacle y moverlo en el eje Z.
- Repite el proceso unas cuantas veces para crear más obstáculos.
- Luego de duplicarlo unas cuantas veces, selecciona uno en Hierarchy y mantén presionado Ctrl + clic para seleccionar múltiples Obstacles, luego duplícalos.



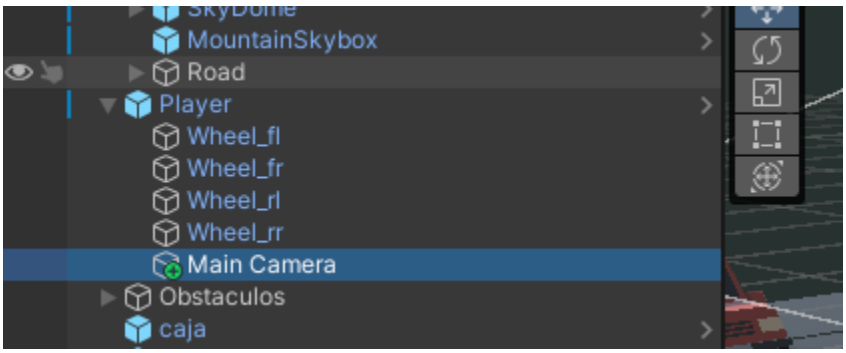
Necesitamos una forma más fácil de cambiar la velocidad del vehículo y hacer que sea accesible desde el Inspector. Para hacerlo, necesitamos algo que se denomina variable.

1. En PlayerController.cs, añade ***public float speed = 5.0f;*** en la parte superior de la **clase.**
2. Reemplaza el **valor speed** en el método Translate con la **variable speed**, y haz una prueba.
3. **Guarda** el *script* y edita el valor de speed en el **Inspector** para conseguir la velocidad que quieras.

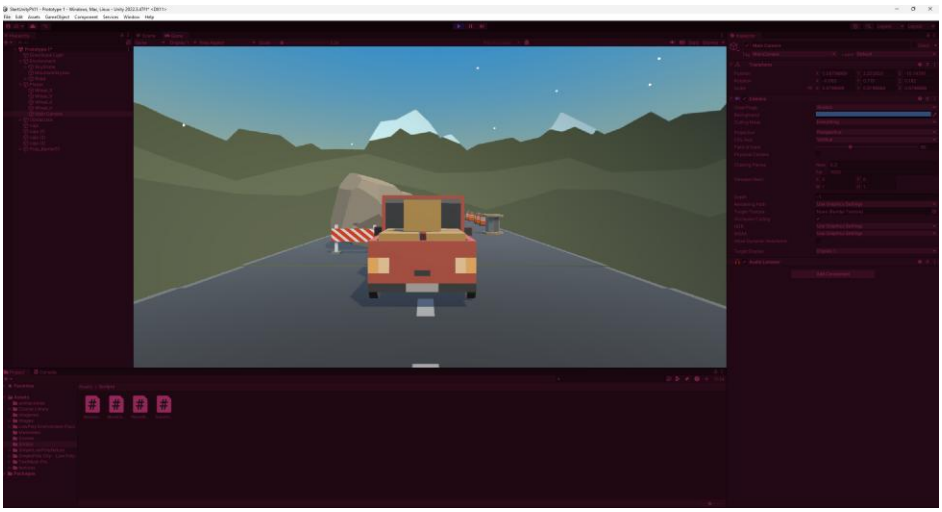




La cámara se encuentra bloqueada en una posición. Si queremos que siga al jugador, tenemos que hacer un nuevo script *para la cámara.*



Si vamos a crear y editar variables, debemos asegurarnos de no hacer cambios accidentalmente en el «Modo juego»

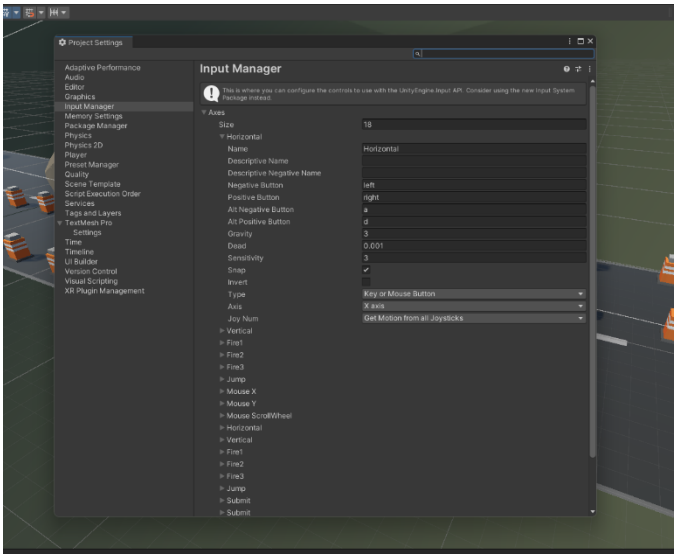


Hasta ahora, el vehículo solo podía avanzar en línea recta por la carretera. Necesitamos que se pueda mover de izquierda a derecha para evitar los obstáculos.

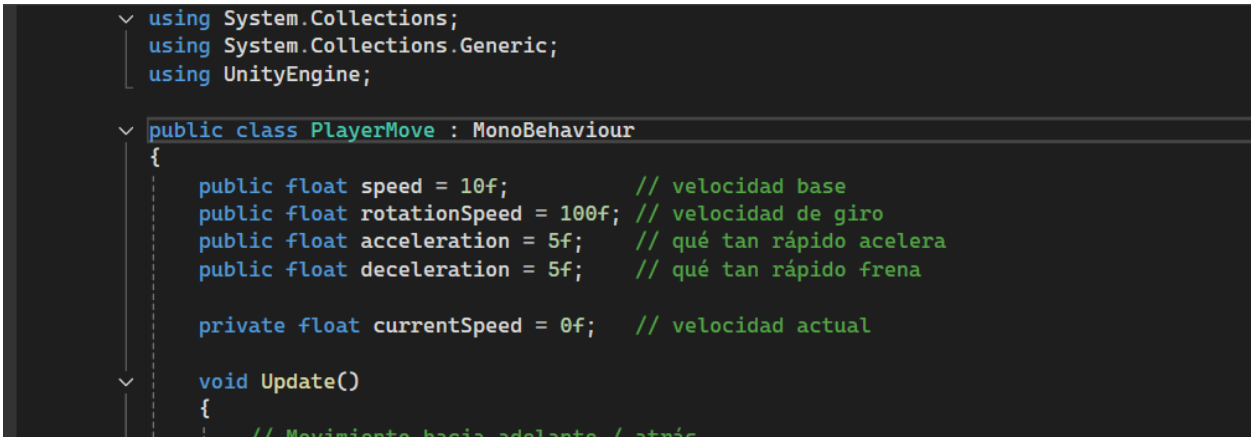
1. En la parte superior de PlayerController.cs, añade ***public float turnSpeed;*** como variable.
2. En ***Update()***, añade ***transform.Translate(Vector3.right * Time.deltaTime * turnSpeed);*.**
3. Inicia el juego y utiliza el ***deslizador de la variable*** **turnSpeed** para mover el vehículo de izquierda a derecha.

```
        // Rotación (gira solo si el carro se está moviendo)
        float turnInput = Input.GetAxis("Horizontal");
        if (Mathf.Abs(currentSpeed) > 0.1f)
        {
            transform.Rotate(Vector3.up * turnInput * rotationSpeed * Time.deltaTime);
        }
    }
}
```

1. Desde el menú superior, haz clic en **Edit** > **Project Settings**, selecciona **Input Manager** en la barra lateral y luego despliega **Axes** para explorar las entradas.
2. En **PlayerController.cs**, añade ***public float horizontalInput*** como variable.
3. En ***Update***, asigna ***horizontalInput = Input.GetAxis("Horizontal");***, luego haz una prueba para verlo en el Inspector.
4. Añade la variable ***horizontalInput*** a la izquierda y derecha del **método Desplazar** para ganar control del vehículo.
5. En el Inspector, edita las variables ***turnSpeed*** y ***speed*** para corregir la sensibilidad.



1. Define ***forwardInput*** como una nueva variable «public».
2. En ***Update***, asigna ***forwardInput = Input.GetAxis("Vertical");*.**
3. Añade la variable ***forwardInput*** al método **forward Translate**, y luego haz una prueba.



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerMove : MonoBehaviour
{
    public float speed = 10f;          // velocidad base
    public float rotationSpeed = 100f; // velocidad de giro
    public float acceleration = 5f;    // qué tan rápido acelera
    public float deceleration = 5f;    // qué tan rápido frena

    private float currentSpeed = 0f;   // velocidad actual

    void Update()
    {
        // Movimiento hacia adelante / atrás
```