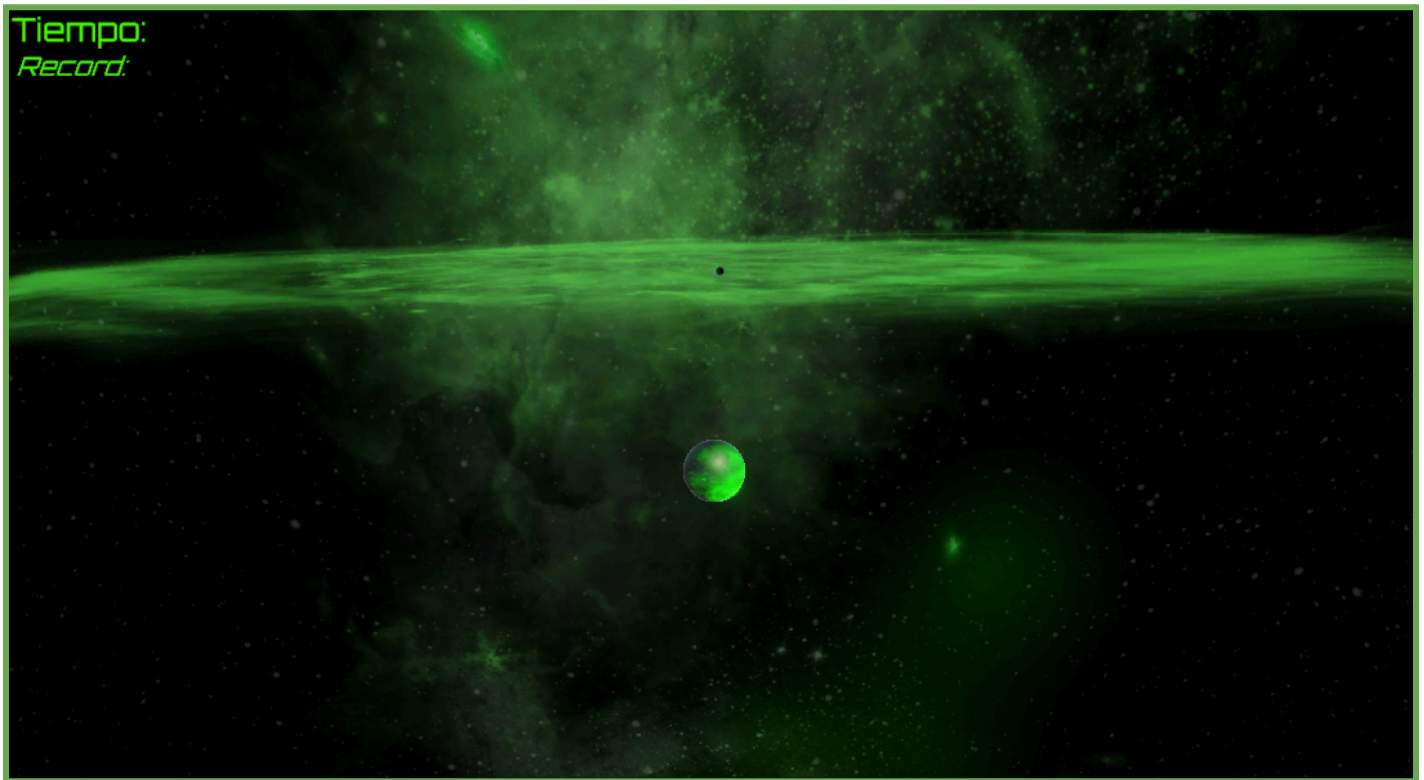


Desafío Esférico

“Simulador de Física Simple”



Iván Wilfrido Delgado Chaparro
2ºDAM

ÍNDICE:

Introducción.....	3
Objetivo del Juego:.....	3
Inspiración:.....	3
Diseño del Juego.....	4
Menús:.....	4
Sonido:.....	5
Fonts:.....	5
Jugador:.....	5
Rampas:.....	5
Obstáculos:.....	6
Entorno del juego:.....	6
Desarrollo Técnico.....	7
Escena y Jerarquía:.....	7
Scripts:.....	8
Problemas.....	9
Cambio de idea:.....	9

Introducción

Objetivo del Juego:

El principal objetivo de este proyecto es crear un videojuego funcional que permita al jugador controlar una bola en su descenso por una rampa infinita mientras esquiva obstáculos y trata seguir con vida hasta que sea eliminado. A través de este proyecto, también se busca:

- Desarrollar habilidades en el uso de Unity y el lenguaje de programación C#.
- Implementar mecánicas de juego dinámicas.
- Crear una experiencia visual atractiva mediante el uso de efectos de partículas y diseño futurista neón.
- Aplicar conocimientos de física para simular movimientos realistas.

Inspiración:

Slopegame.online es un juego conocido por su simplicidad y capacidad para mantener al jugador enganchado durante largos periodos. La inspiración principal radica en su capacidad para generar una sensación de velocidad y riesgo constante, combinada con controles intuitivos. Este proyecto toma esa base y busca enriquecerla mediante:

- La inclusión de efectos visuales adicionales, como partículas de polvo que la bola desprende mientras se desliza por la rampa.
- Una curva de dificultad progresiva que mantenga el interés del jugador durante partidas prolongadas.
- Elementos de personalización y mejoras en el sistema de puntuación.

Diseño del Juego

El juego está basado en un diseño futurista con colores neón y simulando que estamos en el espacio. Los elementos principales son:

Menús:

En el menú inicial contamos con diferentes opciones en el cuál podemos cambiar en opciones el volumen general del juego. Para hacer esto se creó una nueva escena.

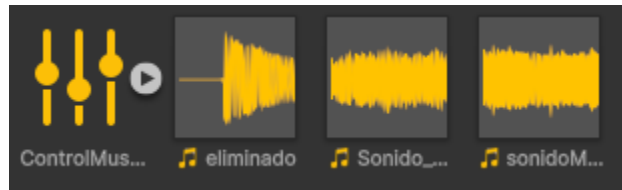


También consta con un menú de pausa el cuál hará las siguientes funciones:



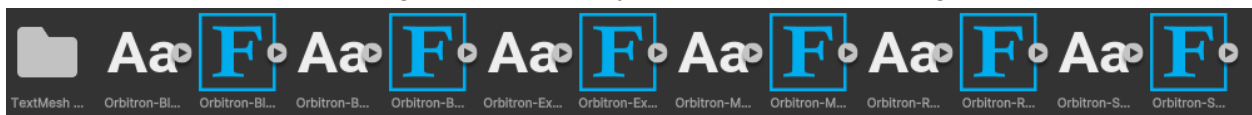
Sonido:

Tanto el menú principal como en el propio juego en acción tienen músicas asociadas a ellos. Para agregar dinamismo hay un sonido el cual se reproduce una vez el jugador es eliminado.



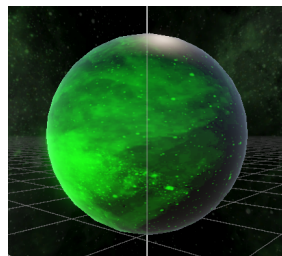
Fonts:

El formato de letra fue descargado de internet y la intención es que tenga esa estética futurista.



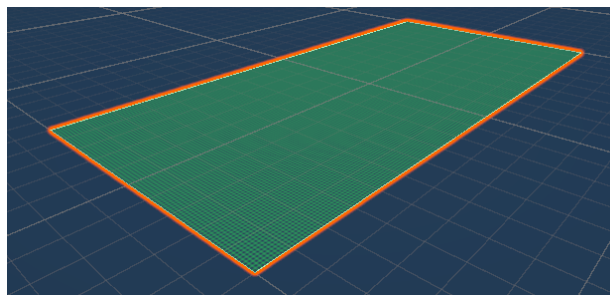
Jugador:

Representado por una bola que el jugador controla. Se mueve automáticamente hacia adelante y puede girar a izquierda o derecha mediante entradas del usuario. Se le asignan físicas realistas, como gravedad y colisiones, para simular deslizamientos y rebote



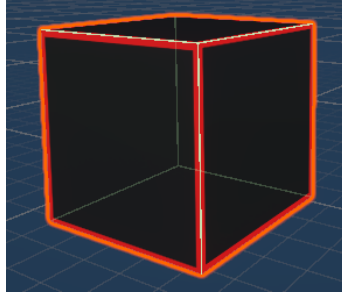
Rampas:

Son estructuras largas y anchas generadas proceduralmente. Están diseñadas con colisionadores para detectar caídas y mantener a la bola sobre su superficie.

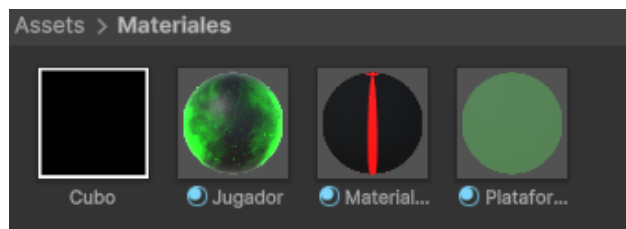


Obstáculos:

Elementos que caen en las rampas desde el cielo y obligan al jugador a maniobrar.. Están estratégicamente ubicados para aumentar la dificultad progresivamente y se generan de manera aleatoria.

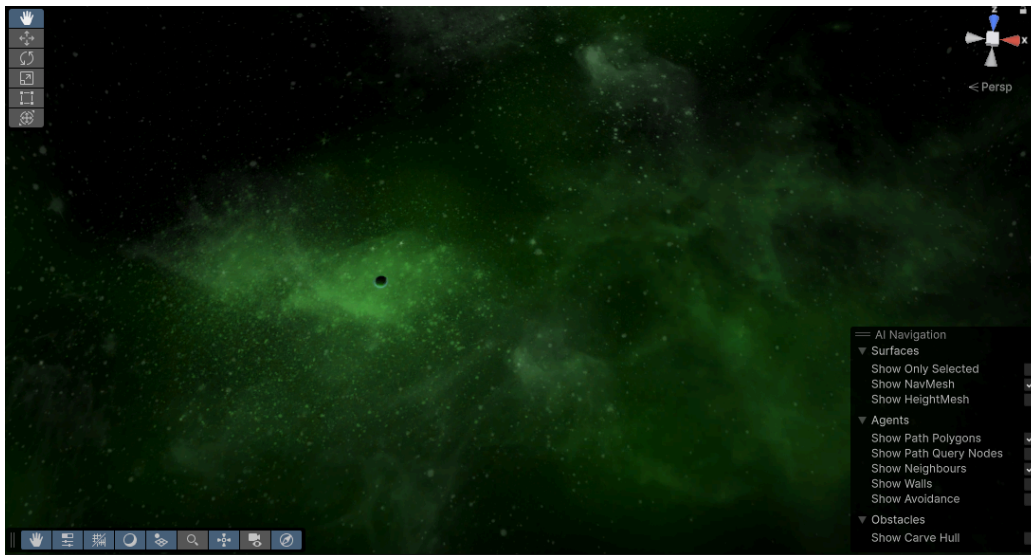


El diseño de las rampas y obstáculos fueron diseñados para que tengan colores neón deseados con Materiales:



Entorno del juego:

Descargado de la asset store hace bastante armonía con el juego pues el color base es el verde y negro. GalacticGreen.

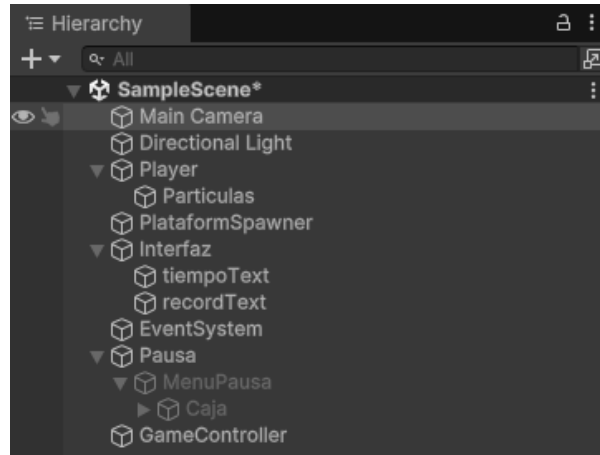


Estos componentes trabajan juntos para crear una experiencia inmersiva que desafía los reflejos y habilidades del jugador.

Desarrollo Técnico

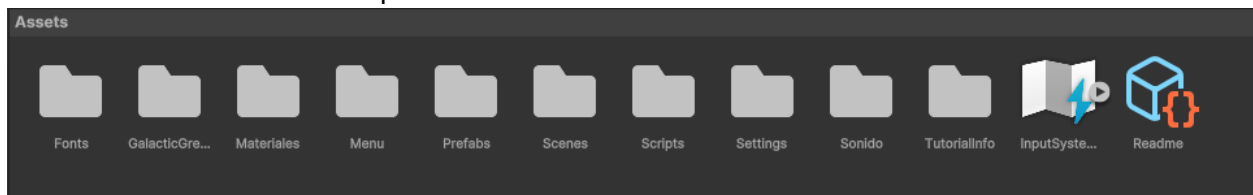
Escena y Jerarquía:

La jerarquía de la escena contiene los elementos principales:



- **Main Camera:** Cámara principal, configurada para seguir al jugador.
- **Directional Light:** Luz direccional para iluminar la escena.
- **Player:** Representa al jugador, una bola que se mueve en el espacio.
- **PlatformSpawner:** Responsable de generar plataformas de manera procedural.
- **Interfaz:** Gestiona la visualización de la UI (cronómetro, puntuación, etc.).
- **Pausa:** Gestiona el menú de pausa en el juego.
- **GameController:** Se encarga de eliminar los objetos que ya no son necesarios durante el desarrollo del juego,

La organización del proyecto fue hecho mediante diferentes carpetas donde se indican que contiene cada una y así poder acceder a ellos de manera más eficiente. El punto donde se desarrolla todo es en los scripts.



Scripts:

Trabajan en conjunto para proporcionar una experiencia de juego fluida, desde la gestión del jugador hasta los efectos visuales y sonoros, así como la generación de elementos en la escena.



- **ControlMusica:** Controla el volumen de la música en el juego ajustándose mediante un deslizador que interactúa con el AudioMixer.
- **DestroyPlayerIfFallen:** Detecta si el jugador cae por debajo de un umbral predefinido y, en tal caso, reproduce un sonido, muestra un mensaje, detiene el juego y regresa al menú principal después de un retraso.
- **DestroyPlayerOnCollision:** Detecta colisiones con objetos etiquetados como "Player", reproduce un sonido de eliminación, detiene el juego y regresa al menú principal tras el retraso del sonido.
- **FollowPlayer:** Mantiene la cámara siguiendo la posición del jugador con un desplazamiento fijo para simular un seguimiento dinámico.
- **GameTimerUI:** Gestiona y muestra el tiempo jugado y el récord del jugador. Detiene el cronómetro al final del juego y guarda un nuevo récord si se establece.
- **MenuIncial:** Proporciona opciones en el menú principal para iniciar el juego o salir del programa.
- **MenuPausa:** Permite pausar y reanudar el juego, reiniciar la escena o salir del juego desde un menú de pausa.
- **ObjectCleaner:** Elimina plataformas y obstáculos que se encuentran muy lejos detrás del jugador para optimizar el rendimiento del juego.
- **PlatformSpawner:** Genera rampas y obstáculos de manera procedural a medida que el jugador avanza, ajustando la frecuencia de los obstáculos según la distancia recorrida.
- **PlayerController:** Controla el movimiento del jugador (la bola) hacia adelante con aceleración progresiva y permite maniobras laterales según las entradas del jugador.

Problemas

El juego originalmente iba a contener diferentes tipos de rampas y estas iban aparecer aleatoriamente a medida que el jugador avanzaba pero no lo conseguí hacer después de dos días completos intentándolo, pues se generaban mal y no conseguía que fueran funcionales.

La solución implementada fue diseñar una única rampa que se genera dinámicamente a medida que el jugador alcanza la mitad de la rampa actual. Al iniciar el juego, se crean dos rampas consecutivas, ambas de gran tamaño, lo que permite al jugador moverse con cierta libertad, pero dentro de un límite que lo obliga a evitar caídas.

El desafío radica en que, mientras el jugador avanza, se generan obstáculos de forma aleatoria en la rampa restante. Estos obstáculos caen desde el cielo, aumentando la complejidad de mantenerse en la trayectoria correcta. A medida que el jugador continúa progresando, nuevas rampas y obstáculos(cada vez más con el tiempo) se generan continuamente, mientras que la velocidad del jugador incrementa con el paso del tiempo. Esto añade un nivel creciente de dificultad, poniendo a prueba la habilidad del jugador para esquivar obstáculos y mantenerse en movimiento en la rampa.

Cuando avanzas en el juego se van creando muchos elementos de tipo obstáculos y rampas, esto al principio funciona bien pero cuando avanzas mucho se crean demasiados y el ordenador va muy lento.

Su solución fue crear un script que eliminase los elementos que el jugador dejaba atrás, es decir tanto obstáculos como rampas y así mejorar el rendimiento del juego.