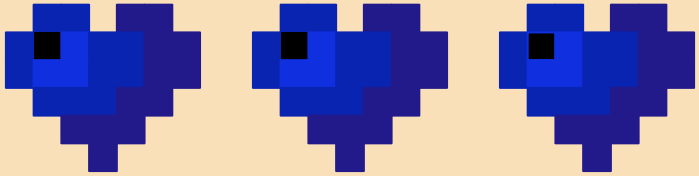


Endless Runner

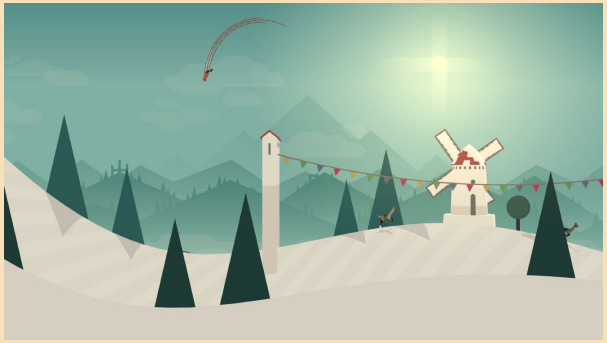
Pintos, Micaela

¿Qué es?

Para que un juego sea del género endless runner debe cumplir con las siguientes características:

- El jugador debe avanzar en una dirección
- Escapar de algún enemigo o peligro (generalmente)
- El objetivo principal es avanzar lo máximo posible antes de morir.

Ejemplos 2D



Alto's Adventure

Canabalt HD

Super Mario Run

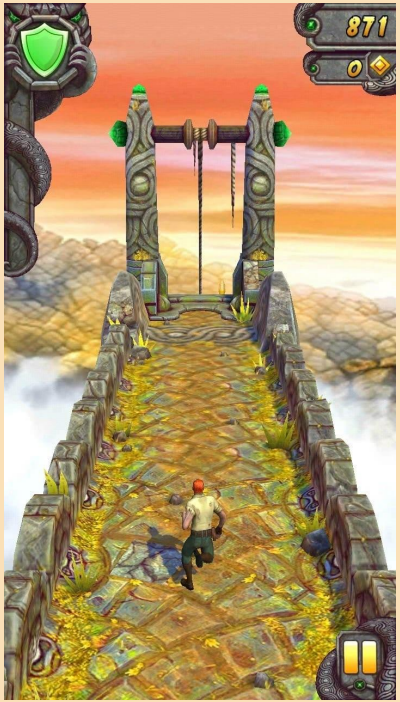
Dino de chrome

Tiny Wings

Ejemplos 3D

Subway Surfers

Minions Rush



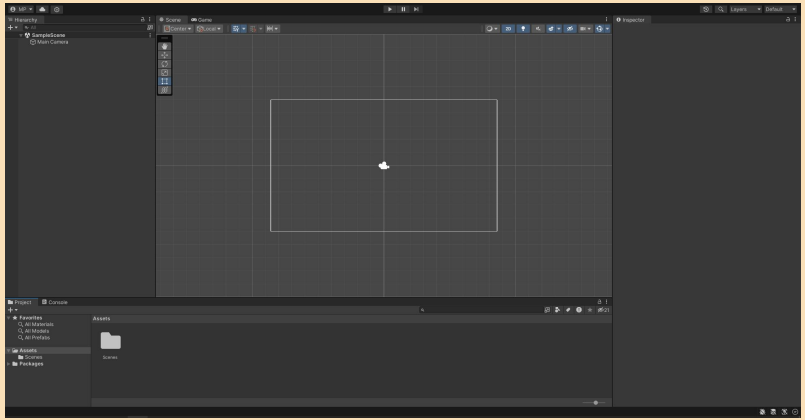
Temple Run

¿Qué necesitamos...

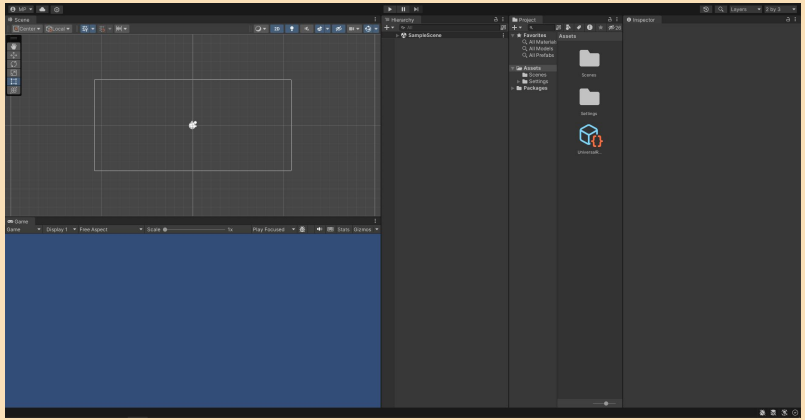
... para realizar nuestro propio endless runner?

- Un entorno de desarrollo como Unity o Godot.
- La temática, es decir, sprites, musica, fondo, etc.

Nuevo proyecto

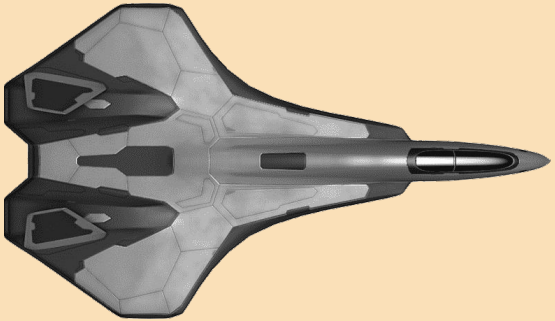


Cuando creamos un nuevo proyecto nos aparece así.

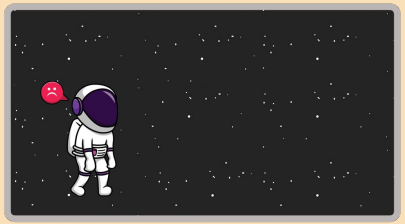


Recomiendo cambiar el layout “2 by 3” para poder visualizar todo al mismo tiempo.

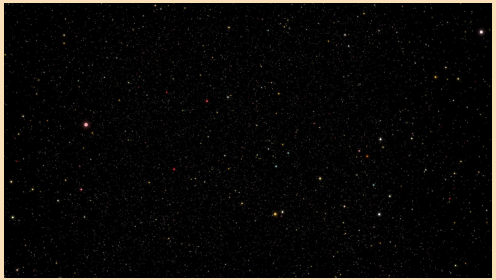
Sprites



Jugador



Panel Game Over



Fondo

Obstaculo

¿Qué necesitamos...

... para realizar nuestro propio endless runner?

1. Controles del jugador
2. Movimiento de cámara
3. Fondo infinito
4. Enemigos
5. Bordes
6. Game Over
7. Puntaje

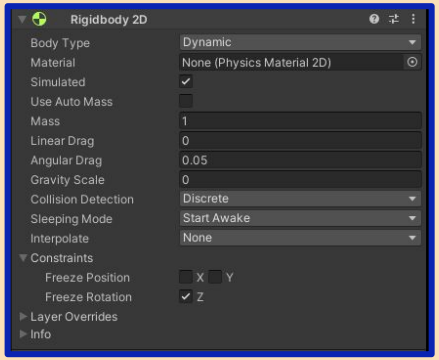
01

Controles del jugador

Controles del jugador

Creamos un GameObject vacío y le agregamos el sprite del jugador como hijo.

Al objeto "*Player Controller*" le tenemos que agregar físicas, es decir, el componente "*RigidBody 2D*".



Modificar la gravedad a 0 y no permitir que en el eje z.

Controles del jugador

Creamos el script "*Jugador*":

```
public float playerSpeed; //La velocidad de movimiento
private Rigidbody2D rb; //La fisica
private Vector2 playerDirection; //Direccion de movimiento
void Start()
{
    rb = GetComponent<Rigidbody2D>();
}
```

Update vs FixedUpdate

Update

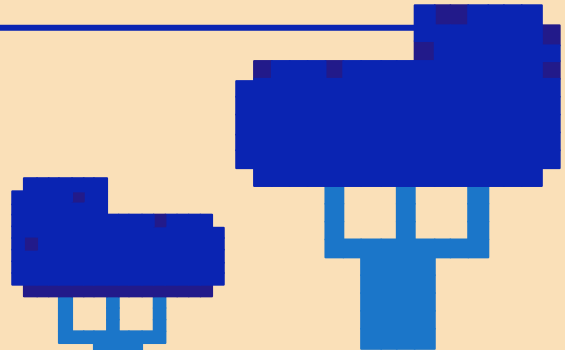
Se llama una vez por frame

Ejemplos: timers, Detecciones de entradas

FixedUpdate

Se llama una vez por frame físico

Ejemplos: Rigidbody



Controles del jugador

```
void Update()
{
    float direccionY = Input.GetAxisRaw("Vertical"); //Determino las teclas
    playerDirection = new Vector2(0, direccionY).normalized; //Normalizo para que el movimiento sea consistente
}

void FixedUpdate()
{
    rb.velocity = new Vector2(0, playerDirection.y * playerSpeed);
}
```

Controles del jugador

02

Movimiento de cámara

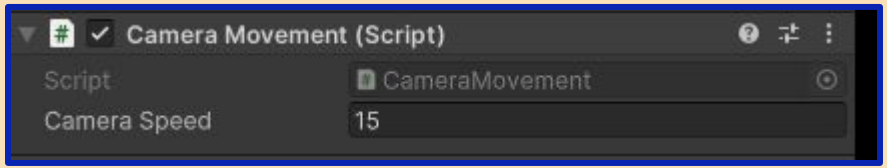
Movimiento de cámara

Creamos el script "*MovimientoCarama*":

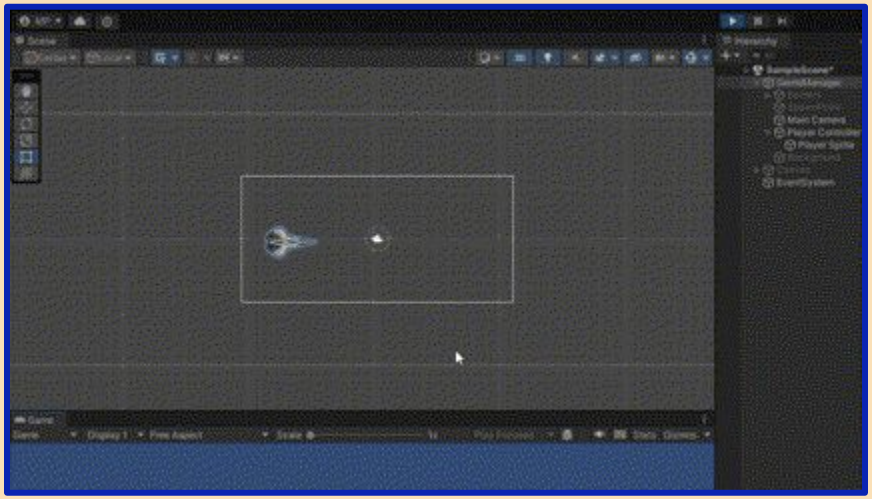
```
public float cameraSpeed; //Velocidad de la camara
void Update()
{
    transform.position += new Vector3(cameraSpeed * Time.deltaTime, 0, 0);
}
```

Movimiento de cámara

Volvemos a crear un GameObject vacío con el nombre "*Game Manager*".
A esté le agregamos el componente con el script que creamos de la cámara.
Por último le asignamos alguna velocidad a la cámara.



Movimiento de cámara

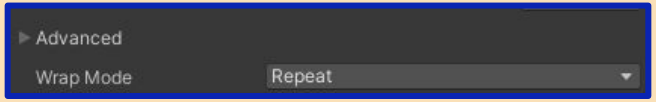


03

Fondo infinito

Fondo infinito

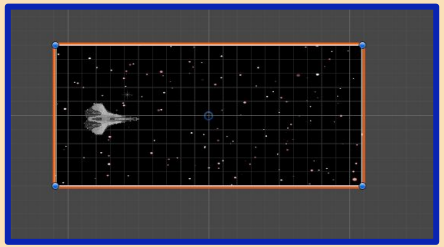
Seleccionamos el sprite de fondo y modificarle el "*wrap mode*"



Esto permite que la imagen se repita y haga el efecto de que estamos navegando por el espacio.

Fondo infinito

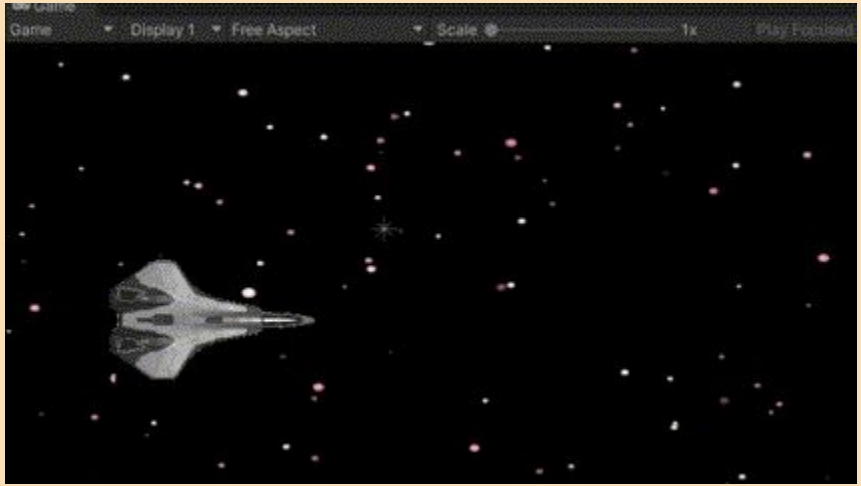
1. Creamos un GameObject 3D Quad con el nombre "*Fondo*"
2. Lo adaptamos a nuestra pantalla, es decir, que ocupe toda la misma.
3. Al material, le agregamos el sprite de fondo
4. Por último, creamos un script con el nombre "*FondoInfinito*"

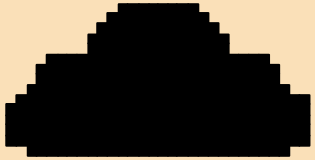


Fondo infinito

```
public float backgroundSpeed; //Velocidad del fondo
public Renderer backgroundRenderer; //Mesh del fondo

void Update()
{
    backgroundRenderer.material.mainTextureOffset += new Vector2(backgroundSpeed * Time.deltaTime, 0f);
}
```

Fondo infinito





04

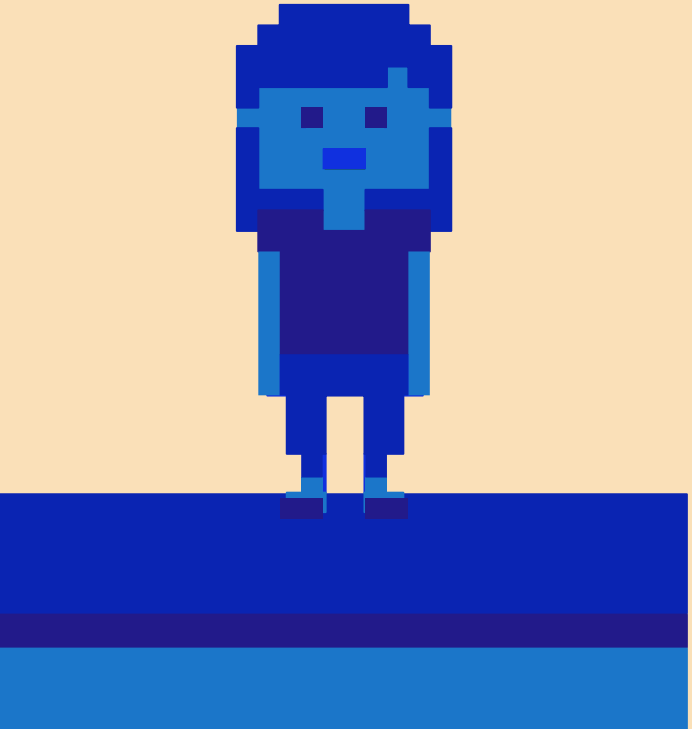
Enemigos

Enemigos

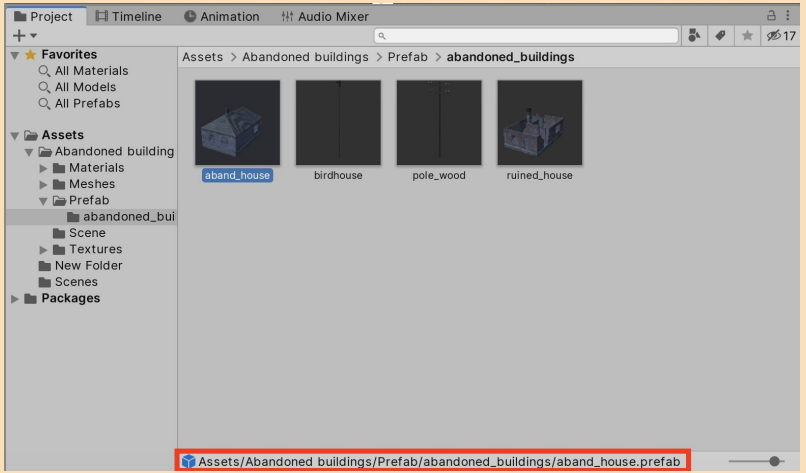
Creamos un GameObject vacío con el nombre "*Obstaculo*"
Le agregamos de hijo el sprite obstáculo (similar a como hicimos con el jugador)

Con este componente, creamos un Prefab.

Prefab



El prefab actúa como una plantilla a partir de la cual se pueden crear nuevas instancias del objeto en la escena.



Enemigos

Creamos el script "*Spawn Enemigos*":

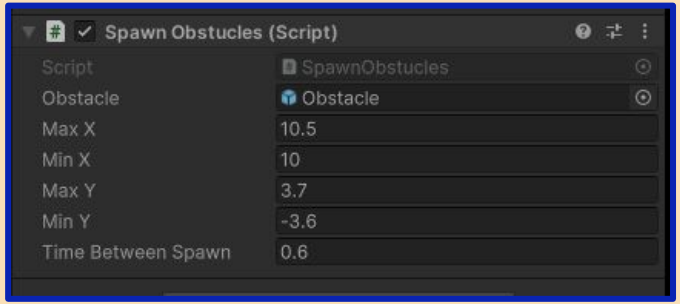
```
public GameObject obstacle;
public float maxX;
public float minX;
public float maxY;
public float minY;
public float timeBetweenSpawn;
private float spawnTime;
```

Enemigos

```
void Update()
{
    if(Time.time > spawnTime) //Si el tiempo transcurrido es mayor al tiempo de aparicion
    {
        Spawn();
        spawnTime = Time.time  + timeBetweenSpawn;
    }
}

void Spawn()
{
    float randomX = Random.Range(minX, maxX);
    float randomY = Random.Range(minY, maxY);

    Instantiate(obstacle, transform.position + new Vector3(randomX, randomY, 0), transform.rotation);
}
```

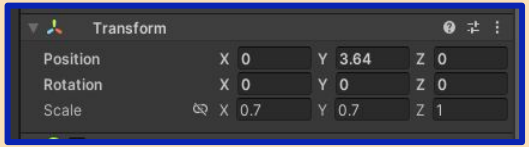
Enemigos

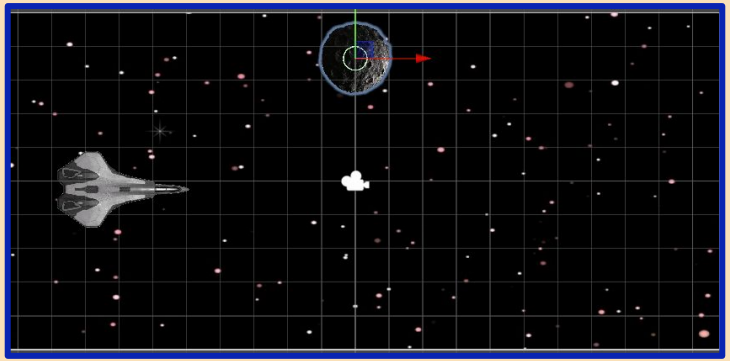
Creamos un GameObject vacío con el nombre "*Punto de aparición*" y le agregamos el script que hicimos recién.



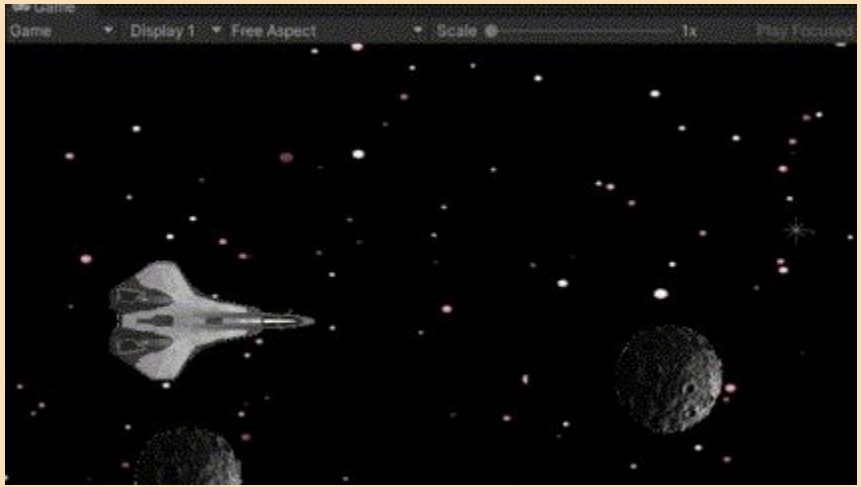
Enemigos

Para determinar el *MaxY* o *MinY*





Enemigos

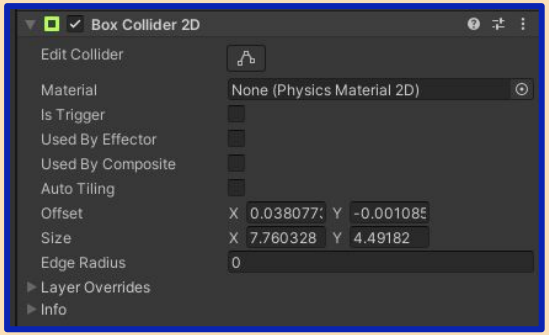


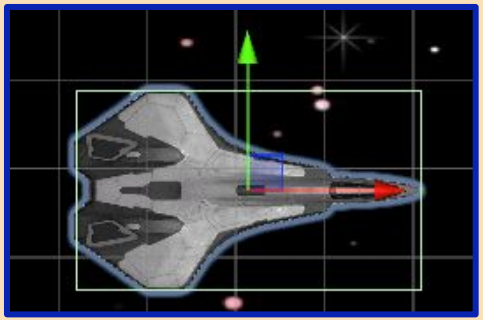
05

Bordes

Bordes

Para evitar que el jugador pase los bordes de la cámara, debemos agregar colisión que también funcionara contra los enemigos.





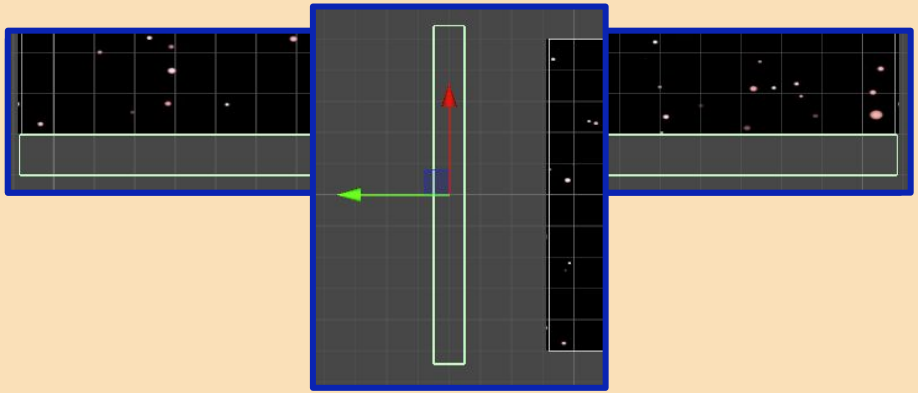
Bordes

Creamos un GameObject vacío con el nombre de "*Bordes*" y esté será padre de los siguientes objetos:

- Borde de arriba
- Borde de abajo
- Borde de lado

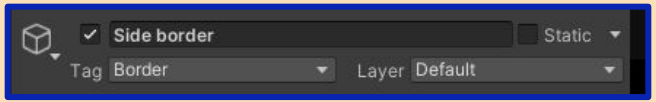
Bordes

Cada borde será un GameObject vacío y con el componente Box Collider 2D



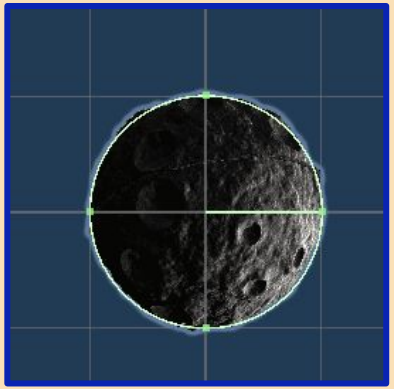
Bordes

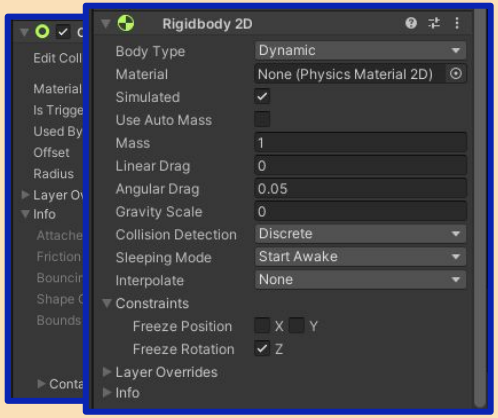
Para todos los bordes, creamos el "*tag*" Borde y se lo agregamos



Bordes

Al prefab obstáculo tenemos que agregarle el componente Circle Collider



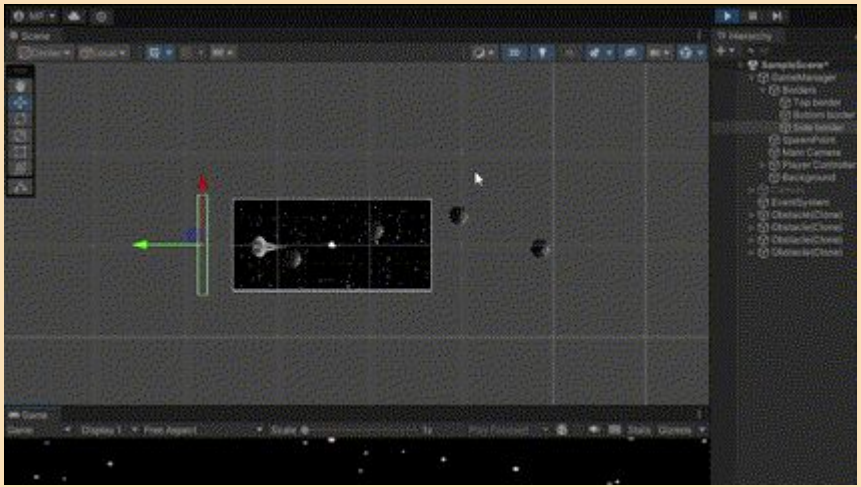


Bordes

Creamos un script con el nombre "*Obstaculo*" y se lo agregamos al Prefab de obstáculo.

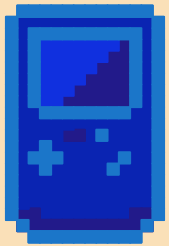
```
private void OnTriggerEnter2D(Collider2D collision)
{
    if(collision.tag == "Border")
    {
        Destroy(this.gameObject);
    }
}
```

Bordes



Bordes

Todo lo que hicimos hasta ahora pertenece al Game Manager

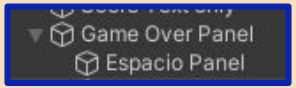


06

Game Over

Game Over

Creamos un panel con el nombre "Panel Game Over" y le determinamos el tamaño de la cámara. Le agregamos como hijo la imagen del panel.



Game Over

También le agregamos al panel un botón para reiniciar y un text "Game Over"

Game Over

En la clase "Obstáculo" tenemos que agregarle cuando el jugar colisiona con un obstáculo.

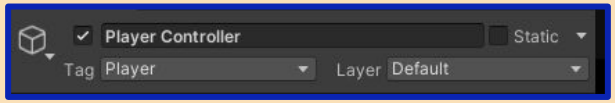
```
private GameObject player;
void Start()
{
    player = GameObject.FindGameObjectWithTag("Player");
}
```

Game Over

```
private void OnTriggerEnter2D(Collider2D collision)
{
    if(collision.tag == "Border")
    {
        Destroy(this.gameObject);
    }
    else if(collision.tag == "Player")
    {
        Destroy(player.gameObject);
    }
}
```

Game Over

Al jugador le agregamos el "tag Player"



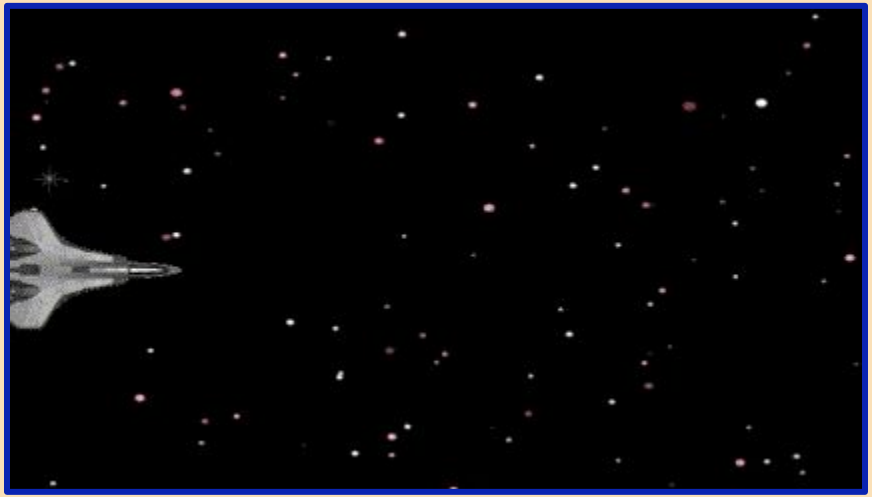
Game Over

Creamos un script con el nombre "Game Over":

```
using UnityEngine.SceneManagement;
```

```
public GameObject gameOverPanel;
void Update()
{
    if(GameObject.FindGameObjectWithTag("Player") == null)
    {
        gameOverPanel.SetActive(true);
    }
}

public void Restart()
{
    SceneManager.LoadScene(SceneManager.GetActiveScene().name);
}
```

Game Over




07
Puntaje

Puntaje

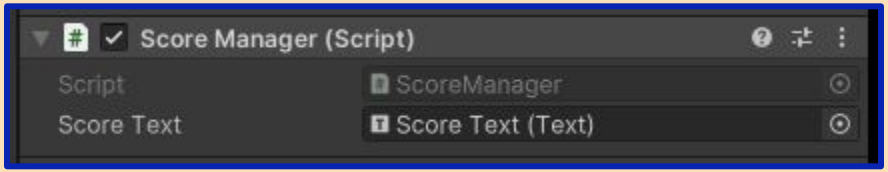
Creamos un script con el nombre "Puntaje"

```
using UnityEngine.UI;
```

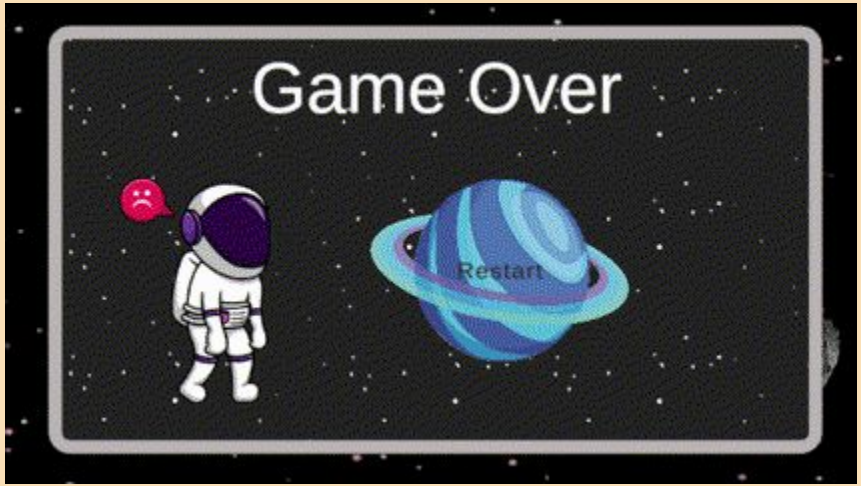
```
public Text scoreText;
private float score;
void Update()
{
    if(GameObject.FindGameObjectWithTag("Player") != null)
    {
        score += 1 * Time.deltaTime;
        scoreText.text = ((int)score).ToString(); //Actualizamos el valor

    }
}
```

Puntaje

Creamos un GameObject de tipo Text y le agregamos el script y linkeamos con el texto del puntaje.

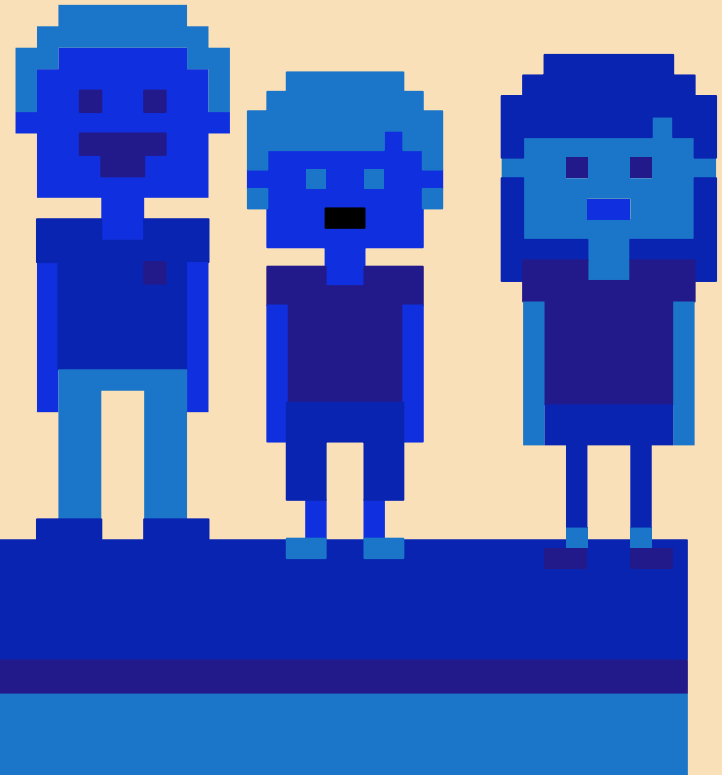


Puntaje

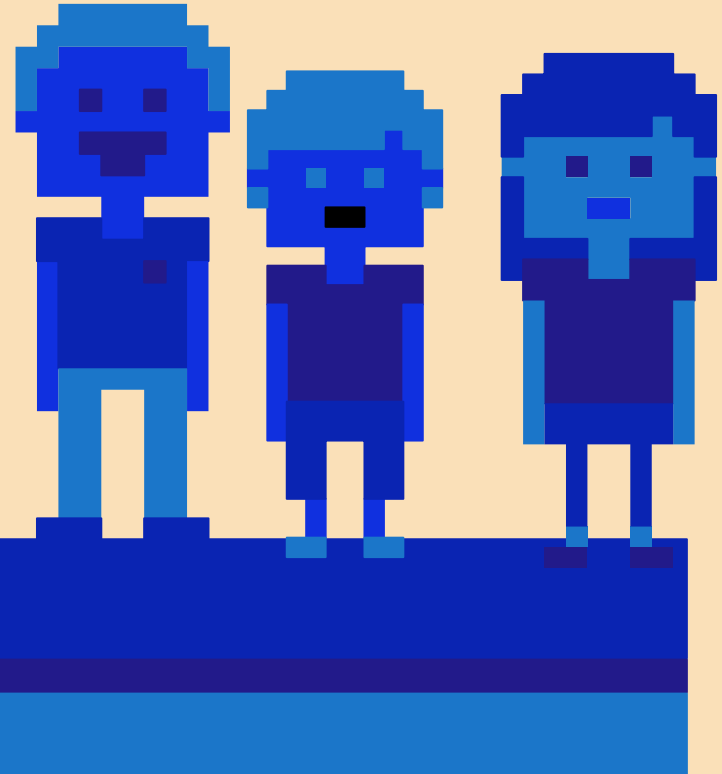


08

Juego Final



¿Preguntas?



Muchas Gracias