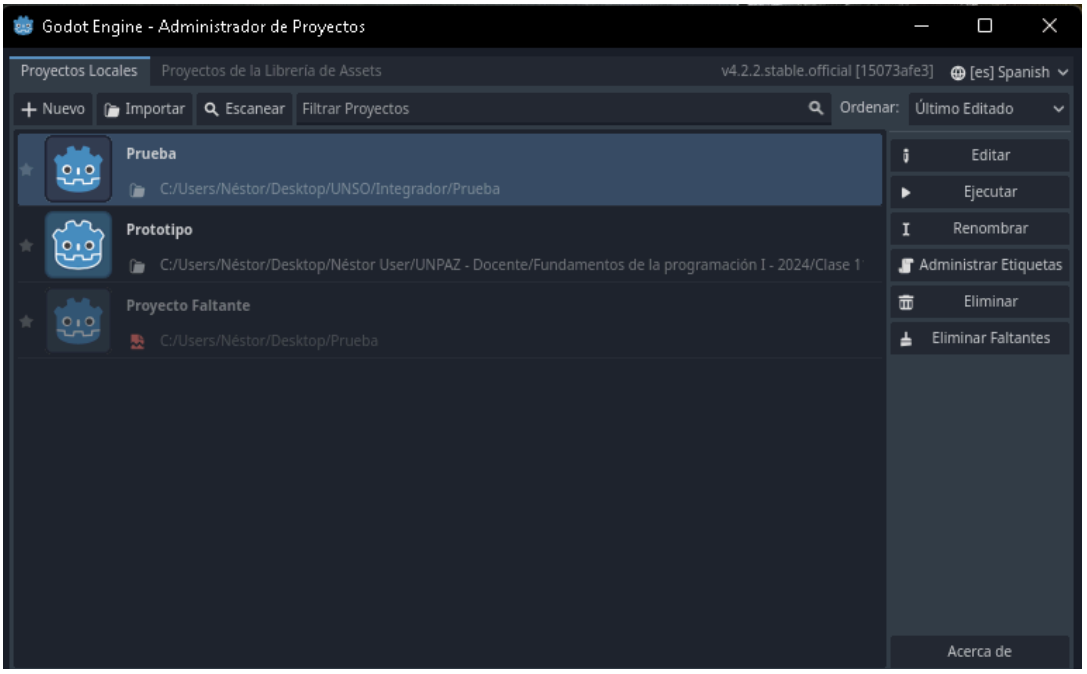
TUTORIAL 3: Endless Runner

Creación de un proyecto

Utilizaremos GODOT 4.

Hacemos doble click en el ícono de Godot para ejecutar el programa. La primera pantalla que veremos es el gestor de proyectos. Hacemos click en el botón “Nuevo”, nombramos el proyecto, seleccionamos o creamos una carpeta para alojar el proyecto, seleccionamos el renderizador y hacemos click en el botón “Crear y editar”.

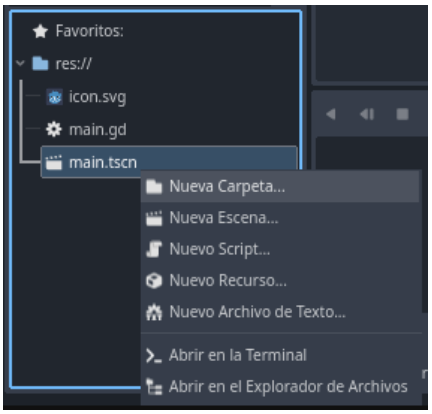


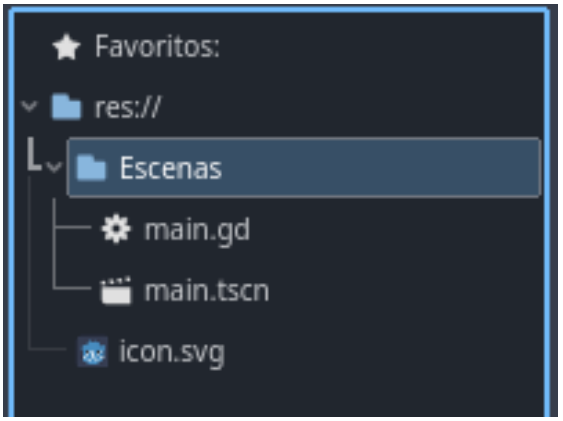
Creación de una escena principal

Ahora estamos ubicados en el área de trabajo de Godot. Por defecto viene predeterminada la vista 3D. Haremos click en el botón 2D que se encuentra en la barra de visualización.

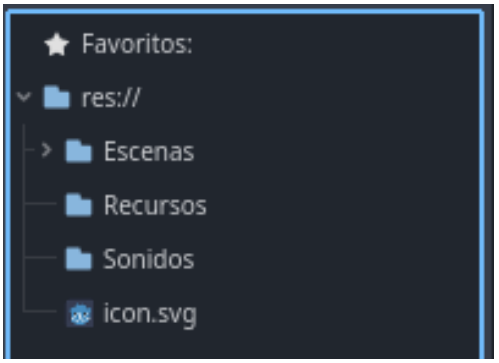
Crearemos una nueva escena con un nodo de control Node2D, lo renombramos Main y adjuntamos un script. Luego guardaremos el proyecto con Ctrl+S.

Una vez que hayamos guardado la escena iremos al sistema de archivos y con click derecho seleccionaremos “Nueva carpeta” de esta forma crearemos una carpeta a la que llamaremos “Escenas” y allí dentro arrastraremos nuestra escena “Main” junto con su script.





Vamos a adelantarnos algo en la organización y crearemos dos carpetas más para guardar nuestros recursos gráfico y audios, llamaremos a estas carpetas “Recursos” y “Sonidos”.



Creamos un suelo y fondo.

Dentro del Main crearemos el suelo, un nodo StaticBody2D que tenga como hijos a un Sprite2D y a un CollisionShape2D.

Renombramos al StaticBody2D Ground y cargamos el sprite en la propiedad texture bajando y centrando la imagen.

Modificamos y ajustamos el CollisionShape2D y por último, convertimos al Ground en una escena independiente que quede instanciada dentro del Main.

Crearemos un nodo de ParallaxBackground, lo renombraremos “Background” y le añadimos dos nodos hijos de tipo ParalaxLayer, cada uno con un Sprite2D.

Modificamos la propiedad Mirroring en su eje X de cada ParalaxLayer en 1152. Luego la propiedad scale del motion del ParalaxLayer en 0.5.

Cargamos el recurso sky.png en la textura del Sprite2D del ParalaxLayer y desmarcamos la propiedad centered. Lo mismo hacemos con la textura del Sprite2D del ParalaxLayer2 cargando el recurso Tree.png.

Por último, guardamos la escena y la instanciamos dentro del Main.

Main Character, animación y movimiento.

Crearemos un nodo de tipo CharacterBody2D, lo renombraremos “MainCharacter” y le añadimos como nodos hijos un Sprite2D, un collisionSWhape2D, una Camera2D y un AnimationPlayer.

Seleccionamos el nodo AnimationPlayer en el área de escenas y esto abrirá un sector de trabajo de animaciones en la parte inferior de la pantalla.

Hacemos clic en “Animación” → “Nuevo” y le ponemos un nombre a la animación que vamos a crear, la llamaremos “Run”.

Seleccionamos el nodo Sprite2D y en la propiedad “texture” cargamos el recurso run_000. Luego hacemos click en la imagen de la llave que está al lado. Hacemos clic en “Aceptar” en la ventana de creación de llave y proseguimos a terminar de cargar todos los recursos para la creación de la animación.

Activamos la Reproducción Automática al Cargar y el Loop de Animación.

En este punto ya hemos creado la animación con la cual nuestro personaje podrá correr hacia la derecha.

Hacemos click derecho en el nodo MainCharacter y hacemos click en el botón “Añadir nuevo script”. En la ventana que se ha abierto, ubicaremos la propiedad “Plantilla” y le sacaremos la tilde, luego hacemos click en el botón “Crear”.

Una vez abierta la hoja de script, debajo del “extends CharacterBody2D” agregaremos:

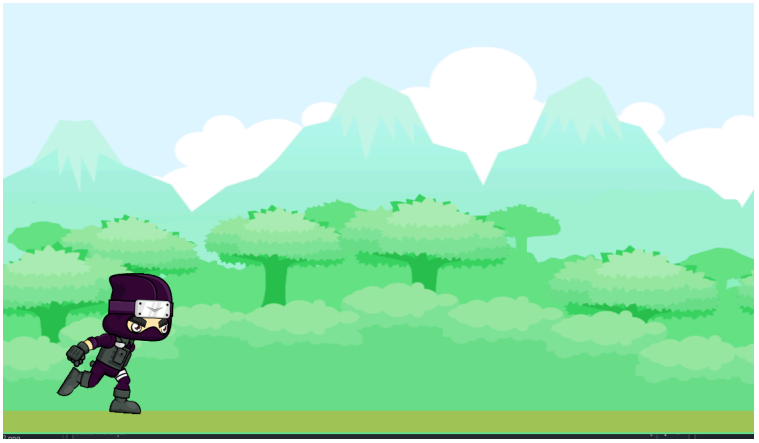
```
extends CharacterBody2D

var speed = 300
const JUMP_VELOCITY = -400
var gravity = ProjectSettings.get_setting("physics/2d/default_gravity")

func _physics_process(delta):
	Move(delta)
	pass

func Move(delta):
	Apply_Gravity(delta)
	Jump()
	Run()
	move_and_slide()
	pass

func Apply_Gravity(delta):
	if not is_on_floor():
		velocity.y += gravity * delta
		pass

func Jump():
	if Input.is_action_just_pressed("ui_accept") and is_on_floor():
		velocity.y = JUMP_VELOCITY
		pass

func Run():
	velocity.x = speed
	pass
```

Límites de la cámara, y suelo procedural.

Modificamos la propiedad sclae del MainCharacter en 0.5.

Seleccionamos el nodo Camera2D y en el inspector ubicamos y desplegamos la propiedad Limit. En Left seteamos a 0, en Top a 0, en Right lo dejamos por default y en Bottom a 648.

El offset lo seteamos a 400 y ajustamos la ubicación del MainCharacter en la escena Main.



Al script del Main agregamos el siguiente código:

```
extends Node2D
var screen_size

func _ready():
	screen_size = get_window().size
	pass

func _process(delta):
	ProcessGround()
	pass

func ProcessGround():
	if $MainCharacter.position.x > $Ground.position.x:
		$Ground.position.x += screen_size.x - 400
		pass
```

Creación de un obstáculo y un punto de spawn.

Crearemos un nodo de tipo Area2D, lo renombraremos “Box” y le añadimos como nodos hijos un Sprite2D, un collisionShape2D y un VisibleOnScreenNotifier2D.

Añadimos un script al Box con el siguiente código:

```
extends Area2D

signal hit

func _process(delta):
	position.x -= 10
	pass
```

Estando seleccionado el Box, desplegamos la pestaña Nodos que se encuentra a un lado del inspector. Allí podremos ver las señales que el nodo posee.

Conectaremos la señal body_entered(body:Node2D) al script del Box, y a la función creada le añadiremos el siguiente script:

```
func _on_body_entered(body):
if body.name == "MainCharacter":
	hit.emit()
pass
```

Estando seleccionado el VisibleOnScreenNotifier2D, desplegamos las señales nuevamente y conectaremos la señal screen_exited() al script del Box, y a la función creada le añadiremos el siguiente script:

```
func _on_visible_on_screen_notifier_2d_screen_exited():
	await get_tree().create_timer(1).timeout
	queue_free()
	pass
```

Crearemos un nodo de tipo Marker2D, lo renombraremos “SpawnPosition” y le añadimos como nodo hijo un Timer.

Seteamos la propiedad Wait Time del Timer a 2.

Añadimos un script al SpawnPosition.

Estando seleccionado el Timer, desplegamos las señales y conectaremos la señal timeout() al sript del SpawnPosition, y a la función creada la modificaremos de manera que nos quede el siguiente sript:

```
extends Marker2D

var box_scene = preload("res://Escenas/box.tscn")
signal hit

func _on_timer_timeout():
	var box = box_scene.instantiate()
	add_child(box)
	box.hit.connect(Won)
	pass

func Won():
	emit_signal("hit")
	pass
```

Lo siguiente es instanciar el SpawnPosition en el main modificando su posición para que siempre se encuentre por delante del MaiCharacter y por fuera de la pantalla.

Hacemos esto agregando al _process(delta) del Main la función Process_SpawnPosition() que se define de la siguiente manera:

```
func Process_SpawnPosition():
	$SpawnPosition.position.y = 553
	$SpawnPosition.position.x = $MainCharacter.position.x + 1300
	pass
```

Al instanciar el SpawnPosition en el main podremos conectar la señal hit al script del main, hacemos esto y modificamos la función creada para que cuando el MainCharacter toque una caja el juego se pause:

```
func _on_spawn_position_hit():
	get_tree().paused = true
	pass
```

Creación de un menú principal, un HUD y una condición de derrota.

Crearemos un nuevo script desde el visor de script Archivo Nuevo script.

Renombramos el script a “Global.gd” y agregamos la siguiente variable booleana:

var first_game: bool = true

Ahora haremos que el script “Global.gd” sea un contenedor de variables globales.

Esto lo hacemos desde Proyecto Configuración del Proyecto Autoload

En ruta agregamos el “Global.gd” y presionamos “Añadir”, nos aseguramos de que quede tildada como variable global y cerramos.

Crearemos un nodo de tipo Node2D, lo renombraremos “Menu” adjuntándole un script y le añadimos como nodo hijo un ColorRec y a este tres Button.

Modificamos la propiedad del inspector Process → Mode de Inherit a Always.

Ajustamos el tamaño y la posición del ColorRec para que quede en medio de la pantalla.

Cambiamos los nombres de los botones a “Play”, “Play_again” y “Exit”; además seteamos la propiedad text con sus correspondientes nombres y conectamos la señal pressed() de cada uno con el script de Menu para que al ser presionados emitan una señal.

El código debe quedar de esta manera:

```
extends Node2D

signal play
signal play_again
signal exit

func _process(delta):
	if Global.first_game:
		$ColorRect/Play_again.visible = false
		$ColorRect/Play_again.disabled = true
```

```
                $ColorRect/Play.visible = true
                $ColorRect/Play.disabled = false
        else:
                $ColorRect/Play_again.visible = true
                $ColorRect/Play_again.disabled = false
                $ColorRect/Play.visible = false
                $ColorRect/Play.disabled = true
        pass

func _on_play_pressed():
        emit_signal("play")
        pass

func _on_play_again_pressed():
        emit_signal("play_again")
        pass

func _on_exit_pressed():
        emit_signal("exit")
        pass
```

Lo siguiente es instanciar el Menu en el Main, ubicarlo en el medio de la pantalla y conectar las señales.

Modificaremos el scrip del Main de manera que el _ready() quede de esta manera:

```
func _ready():
        screen_size = get_window().size
        Process_MenuPosition()
        StartGame()
        pass

func Process_MenuPosition():
        $Menu.position.y = $MainCharacter.position.y - 300
        $Menu.position.x = $MainCharacter.position.x + 100
        pass

func StartGame():
        if Global.first_game:
                get_tree().paused = true
        else:
                _on_menu_play()
```

```
            pass
        pass
```

Modificaremos también el scrip del _on_spawn_position_hit() y el _on_menu_play() para que queden de esta manera:

```
func _on_spawn_position_hit():
    $Menu.visible = true
    Process_MenuPosition()
    get_tree().paused = true
    pass

func _on_menu_play():
    get_tree().paused = false
    $Menu.visible = false
    Global.first_game = false
    pass
```

Para terminar con el Menu debemos conectar las señales _on_menu_play_again() y _on_menu_exit() para luego agregar el siguiente scrip a cada una de las funciones creadas:

```
func _on_menu_play_again():
    get_tree().reload_current_scene()
    pass

func _on_menu_exit():
    get_tree().quit()
    pass
```

Por último crearemos un HUD que informe en todo momento al jugador cuántas cajas lleva saltando.

Para esto crearemos una nueva escena con un nodo CanvasLayer, lo renombramos HUD y añadimos un nodo hijo de tipo Label, a este lo renombramos Score.

En la propiedad Text del Score escribimos SCORE para tener referencia visual y lo posicionamos en el extremo superior izquierdo de la pantalla.

Instanciamos el HUD en el Main.

Para administrar la cantidad de cajas que lleva saltando el MaiCharacter vamos a crear una variable global llamada score y vamos a trabajarla desde las cajas haciendo que cada caja que desaparezca, es decir, que el MainCharacter haya saltado exitosamente, antes de desaparecer, modifique la variable global score sumando 1 al total.

Necesitamos entonces agregar un script al HUD con el siguiente código:

```
extends CanvasLayer

func _process(delta):
    $Score.text = "SCORE: " + str(Global.score)
    pass
```

En la función _on_visible_on_screen_notifier_2d_screen_exited() agregamos en primer lugar esta línea de código:

```
Global.score += 1
```

Lo último que haremos es agregar un tema de fondo.

Para esto agregamos un nodo de tipo AudioStreamPlayer2D al Main y:

- en la propiedad Stream cargamos el recurso Airwolf_Theme
- activamos la propiedad Autoplay
- modificamos la propiedad Max Distance al valor 9223372036854775800.

Podemos hacer algo similar para agregar un sonido de salto al Maincharacter.

En este caso agregamos un nodo de tipo AudioStreamPlayer2D al Maincharacter, en la propiedad Stream cargamos el recurso Jump_sound y modificamos la función Jump() agregando una línea de código más:

```
$AudioStreamPlayer2D.play()
```