# Guia Space Shooter – Parte 7

## Montando o Mundo (Cena Principal - *Main*)

### Objetivo desta parte

• Montar a cena principal Mundo (mundo.tscn).  
• Adicionar os nós necessários (Timers, Jogador, HUD).  
• Conectar os sinais dos Timers.  
• Colocar o script completo do Mundo.  
• Entender como os grupos Player e Inimigos são usados para colisões e limpeza.

### 1. Criar a Cena Mundo

1. Abra a cena mundo.tscn.  
2. Dentro do nó Mundo, adicione:  
- Timer\_Asteroide (spawn dos asteroides).  
- Timer\_Inimigo (spawn dos inimigos).  
- Jogador (adicione a cena jogador.tscn como filho).  
- HUD (adicione a cena hud.tscn como filho).  
  
👉 Configure os timers:  
- Timer\_Asteroide: Wait Time = 1.5s, Autostart = OFF.  
- Timer\_Inimigo: Wait Time = 2.5s, Autostart = OFF.  
  
⚡ Conecte os sinais timeout() dos timers para o script do Mundo.  
  
👉 Lembre-se: o Jogador deve estar no grupo Player. Inimigos e Asteroides devem estar no grupo Inimigos.  
Assim o script consegue identificar corretamente os objetos.

### 2. Script do Mundo

extends Node2D  
  
@onready var cena\_asteroide = preload("res://Cenas/asteroide.tscn")  
@onready var cena\_inimigo = preload("res://Cenas/inimigo.tscn")  
  
var vidas := 3  
var score := 0  
  
func \_ready():  
 randomize()  
 $HUD.connect("start\_game", Callable(self, "iniciar\_jogo"))  
 $HUD.update\_lives(vidas)  
 $HUD.update\_score(score)  
  
func iniciar\_jogo():  
 vidas = 3  
 score = 0  
 $HUD.update\_lives(vidas)  
 $HUD.update\_score(score)  
   
 if not has\_node("Jogador"):  
 var cena\_jogador = preload("res://Cenas/jogador.tscn")  
 var j = cena\_jogador.instantiate()  
 j.position = Vector2(240, 400)  
 add\_child(j)  
   
 $Timer\_Asteroide.start()  
 $Timer\_Inimigo.start()  
  
func \_on\_timer\_asteroide\_timeout():  
 var a = cena\_asteroide.instantiate()  
 a.position = Vector2(randi\_range(30, 480), -20)  
 add\_child(a)  
  
func \_on\_timer\_inimigo\_timeout():  
 var i = cena\_inimigo.instantiate()  
 i.position = Vector2(randi\_range(30, 480), -20)  
 add\_child(i)  
  
func add\_score(valor: int) -> void:  
 score += valor  
 $HUD.update\_score(score)  
   
func perder\_vida(qtd := 1):  
 if vidas <=0:  
 return  
 vidas -= qtd  
 $HUD.update\_lives(vidas)  
 if vidas <= 0:  
 game\_over()  
  
func game\_over():  
 $Timer\_Asteroide.stop()  
 $Timer\_Inimigo.stop()  
  
 for n in get\_tree().get\_nodes\_in\_group("Inimigos"):  
 n.queue\_free()  
  
 if has\_node("Jogador"):  
 $Jogador.queue\_free()  
   
 $HUD.show\_game\_over()

### 3. O que esse script faz

• \_ready() → conecta o HUD e inicializa vidas e score.  
• iniciar\_jogo() → reinicia vidas, score e recria o Jogador.  
• \_on\_timer\_asteroide\_timeout() → cria asteroides.  
• \_on\_timer\_inimigo\_timeout() → cria inimigos.  
• add\_score() → aumenta a pontuação.  
• perder\_vida() → tira vidas quando o jogador é atingido.  
• game\_over() → para os timers, remove todos os nós do grupo Inimigos e o Jogador, e mostra Game Over.  
  
👉 Note como o uso dos grupos Player e Inimigos é essencial: sem eles, o jogo não conseguiria detectar colisões nem limpar corretamente os objetos.