

Integração com Roblox - Guia Completo

Visão Geral

Este guia explica como usar o Motor Procedural Educacional em um projeto Roblox. O fluxo é:

1. **Gerar mapa** no motor (TypeScript)
2. **Exportar JSON** do mapa
3. **Carregar em Roblox** via script Luau
4. **Construir** no Workspace

Quick Start

Passo 1: Gerar Mapa no Motor

TypeScript

```
import { generateDungeonForStudent } from "../src/edu/api/educationalApi";

const resultado = generateDungeonForStudent("aluno_001");

// Obter JSON do mapa
const mapaJson = resultado.codigoGerado;

// Salvar em arquivo (ou enviar para Roblox)
console.log(mapaJson);
```

Passo 2: Adicionar Módulo ao Roblox

1. Abra seu projeto Roblox no Roblox Studio
2. Vá para **ServerScriptService**
3. Crie um novo **ModuleScript** chamado `RobloxMapaModule`
4. Copie o conteúdo de `examples/RobloxMapaModule.lua` para o script
5. Salve

Passo 3: Adicionar Script de Servidor

1. Crie um novo **Script** em **ServerScriptService** chamado `MapBuilder`

2. Copie o conteúdo de `examples/RobloxServerScript.lua`
3. Salve

Passo 4: Testar

1. Clique em **Play** no Roblox Studio
2. Você deve ver mensagens no Output:

Plain Text

```
[MapaBuilder] Iniciando servidor de construção de mapas...
[MapaBuilder] Testando com mapa de exemplo...
[MapaBuilder] Mapa ID: mapa_teste_001, Seed: seed_123
[MapaBuilder] ✓ Mapa construído com sucesso: 2 setores, 8 tiles
```



Estrutura do Mapa JSON

O motor gera mapas em JSON com esta estrutura:

JSON

```
{
  "id": "mapa_dungeon_aluno_001_1768092246025",
  "seed": "9vcfi8",
  "dimensoes": {
    "largura": 64,
    "altura": 64
  },
  "setores": [
    {
      "id": "setor_bsp_0",
      "bounds": {
        "x": 0,
        "y": 0,
        "largura": 32,
        "altura": 32
      },
      "tipo": "sala"
    }
  ],
  "tiles": [
    {
      "tileId": "chao_normal",
      "x": 0,
      "y": 0,
```

```

        "metadados": {}
    },
    {
        "tileId": "parede_pedra",
        "x": 1,
        "y": 0,
        "metadados": {}
    }
],
"metadados": {
    "autorId": "aluno_001",
    "criadoEm": "2026-01-10T20:10:00Z",
    "stats": {
        "numSetores": 30,
        "numTiles": 4096,
        "densidade": 1.0,
        "tempoGeracaoMs": 595
    }
}
}

```

API do RobloxMapaModule

MapaModule.FromJSON(json)

Desserializa JSON para tabela Lua.

Parâmetros:

- `json` (string): JSON do mapa

Retorno: Tabela com estrutura do mapa

Exemplo:

Lua

```

local MapaModule =
require(game.ServerScriptService:WaitForChild("RobloxMapaModule"))
local mapa = MapaModule.FromJSON(mapaJson)

```

MapaModule.BuildFromMapa(workspace, mapa, options)

Constrói o mapa no Workspace.

Parâmetros:

- `workspace` (Instance): Onde construir (geralmente `workspace`)

- `mapa` (table): Estrutura do mapa
- `options` (table, opcional):
 - `baseFolderName` (string): Nome da pasta raiz (default: "GeneratedMaps")
 - `maxParts` (number): Limite de partes (default: 5000)
 - `tileSize` (number): Tamanho de cada tile em studs (default: 1)
 - `colorScheme` (table): Cores customizadas por tipo

Retorno: Tabela com resultado

Lua

```
{
  success = true,
  partCount = 4096,
  setorCount = 30,
  mapaId = "mapa_...",
  seed = "9vcfi8",
  message = "Mapa construído com sucesso: 30 setores, 4096 tiles"
}
```

Exemplo:

Lua

```
local resultado = MapaModule.BuildFromMapa(workspace, mapa, {
  baseFolderName = "MeuMapa",
  maxParts = 10000,
  tileSize = 2,
})

if resultado.success then
  print(resultado.message)
end
```

MapaModule.ClearGeneratedMaps(workspace, baseFolderName)

Remove mapas gerados anteriores.

Parâmetros:

- `workspace` (Instance): Workspace
- `baseFolderName` (string): Nome da pasta a remover

Retorno: `true` se removido, `false` se não encontrado

Exemplo:

Lua

```
MapaModule.ClearGeneratedMaps(workspace, "GeneratedMaps")
```

MapaModule.GetMapStats(mapa)

Obtém estatísticas do mapa.

Parâmetros:

- `mapa` (table): Estrutura do mapa

Retorno: Tabela com stats

Lua

```
{  
  id = "mapa_...",  
  seed = "9vcfi8",  
  dimensoes = { largura = 64, altura = 64 },  
  numSetores = 30,  
  numTiles = 4096,  
  stats = { ... }  
}
```



Customizar Cores

Você pode customizar as cores dos tiles passando `colorScheme` :

Lua

```
local resultado = MapaModule.BuildFromMapa(workspace, mapa, {  
  colorScheme = {  
    chao = BrickColor.new("Bright green"),  
    parede = BrickColor.new("Dark stone grey"),  
    porta = BrickColor.new("Bright orange"),  
    spawn = BrickColor.new("Bright blue"),  
    boss = BrickColor.new("Bright red"),  
    loja = BrickColor.new("Bright yellow"),  
  }  
})
```



Usar com RemoteFunction

Para enviar mapas do cliente para o servidor:

Cliente:

Lua

```
local remoteFunction =  
game.ServerScriptService:WaitForChild("BuildMapaRemote")  
local resultado = remoteFunction:InvokeServer(mapaJson)  
print(resultado.message)
```

Servidor (já implementado em `RobloxServerScript.lua`):

Lua

```
local remoteFunction = Instance.new("RemoteFunction")  
remoteFunction.Name = "BuildMapaRemote"  
remoteFunction.Parent = game.ServerScriptService  
  
function remoteFunction.OnServerInvoke(player, mapaJson)  
    return construirMapa(mapaJson)  
end
```



Fluxo Completo de Exemplo

1. Gerar no Motor

TypeScript

```
// motor.ts  
import { generateDungeonForStudent } from "../src/edu/api/educationalApi";  
import { serializeMapa } from "../src/core/models/serialization";  
  
const resultado = generateDungeonForStudent("aluno_001");  
const mapaJson = serializeMapa(resultado.mapa);  
  
// Salvar em arquivo  
import fs from "fs";  
fs.writeFileSync("dungeon.json", mapaJson);
```

2. Copiar para Roblox

Copie o conteúdo de `dungeon.json` para uma StringValue em Roblox:

Lua

```
-- Em ServerScriptService
local mapaJsonValue = Instance.new("StringValue")
mapaJsonValue.Name = "MapaJSON"
mapaJsonValue.Value = [[{...json aqui...}]]
mapaJsonValue.Parent = game.ServerScriptService
```

3. Carregar e Construir

Lua

```
local MapaModule =
require(game.ServerScriptService:WaitForChild("RobloxMapaModule"))
local mapaJsonValue = game.ServerScriptService:WaitForChild("MapaJSON")

local mapa = MapaModule.FromJSON(mapaJsonValue.Value)
local resultado = MapaModule.BuildFromMapa(workspace, mapa)

print(resultado.message)
```

Limitações e Considerações

Performance

- **Limite de Partes:** Roblox tem limite de ~10k partes por workspace. O padrão é 5000.
- **Rendering:** Muitas partes podem impactar FPS. Considere usar terrain ao invés de parts.
- **Serialização:** Mapas grandes (~4096 tiles) geram JSON de ~200KB.

Compatibilidade

- **Versão Roblox:** Requer Roblox Studio recente (2023+)
- **Luau:** Usa apenas Luau padrão, sem dependências externas
- **HttpService:** Necessário para desserializar JSON



Troubleshooting

Erro: "HttpService is not allowed to access roblox.com"

Solução: Ative HttpService em **Game Settings** → **Security**

Erro: "Contradição em WFC"

Solução: Reduzir complexidade da intenção (menos áreas, dificuldade menor)

Mapa não aparece

Solução: Verificar console para erros, validar JSON, aumentar `maxParts`



Próximos Passos

1. **Customizar Tiles:** Adicione novos tipos de tiles em `educationalApi.ts`
2. **Adicionar Lógica de Jogo:** Scripts Luau para interação com tiles
3. **Otimizar Performance:** Usar terrain ao invés de parts
4. **Integrar com UI:** Mostrar stats do mapa na interface



Referências

- [Roblox Scripting Documentation](#)
- [Luau Language](#)
- [Motor Procedural Educacional - README](#)
- [Arquitetura do Motor](#)