

DCC192

2025/2



# Desenvolvimento de Jogos Digitais

A10: Sistemas de Partículas

Prof. Lucas N. Ferreira

## Avisos

- ▶ A entrega do **TP1: Pong** é hoje (15/09) às 23:59!

## Última Aula

- ▶ Otimizações para Detecção de Colisão
  - ▶ Sort, sweep and prune
  - ▶ Particionamento Espacial:
    - ▶ Haching Espacial
    - ▶ Quadtree/Octree

# Plano de Aula



- ▶ Sistemas de Partículas
  - ▶ Partícula, Emissores de Partículas e Sistemas de Emissores
  - ▶ Detalhes de Implementação: Pool de Memória
- ▶ Exemplos
  - ▶ Fogo
  - ▶ Projéteis

# Sistemas de Partículas em Jogos Digitais



Geralmente, em jogos digitais queremos **modelar fenômenos naturais** orgânicos — aqueles em que uma malha sólida não consegue representar bem:



Por exemplo:

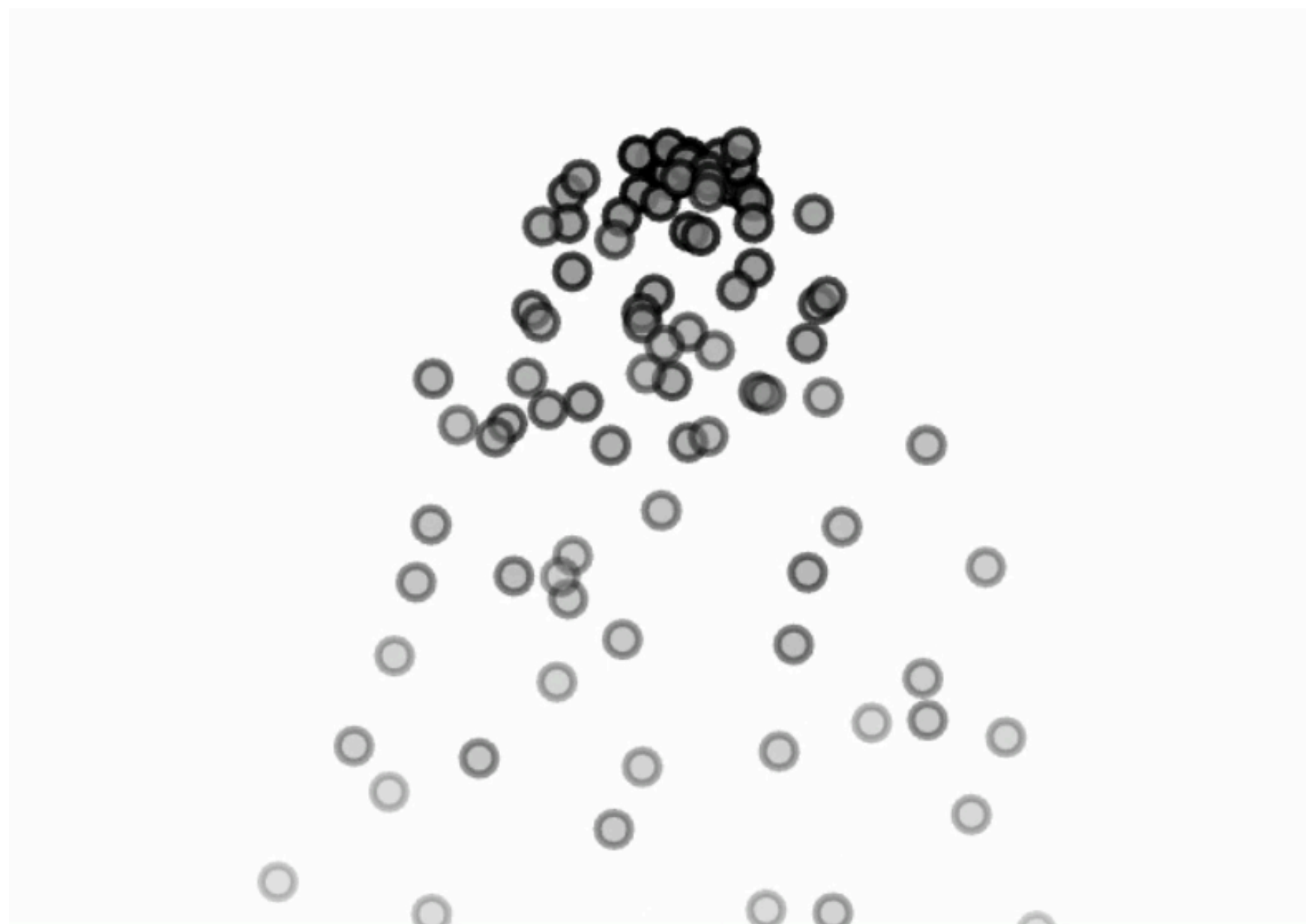
- ▶ Fogo
- ▶ Fumaça
- ▶ Explosões
- ▶ Tiros
- ▶ Estilhaços
- ▶ ...



# Partículas, não malhas



Sistemas de partículas **modelam fenômenos naturais** complexos com uma coleção de pontos ou polígonos simples texturizados. Dessa forma, um sistema de partículas é composto por:



- ▶ **Partícula**

Objeto rígido sem rotação com tempo de vida limitado, i.e. morre após um certo tempo

- ▶ **Emissor de Partículas**

Uma estrutura de dados que gerencia add/del de partículas

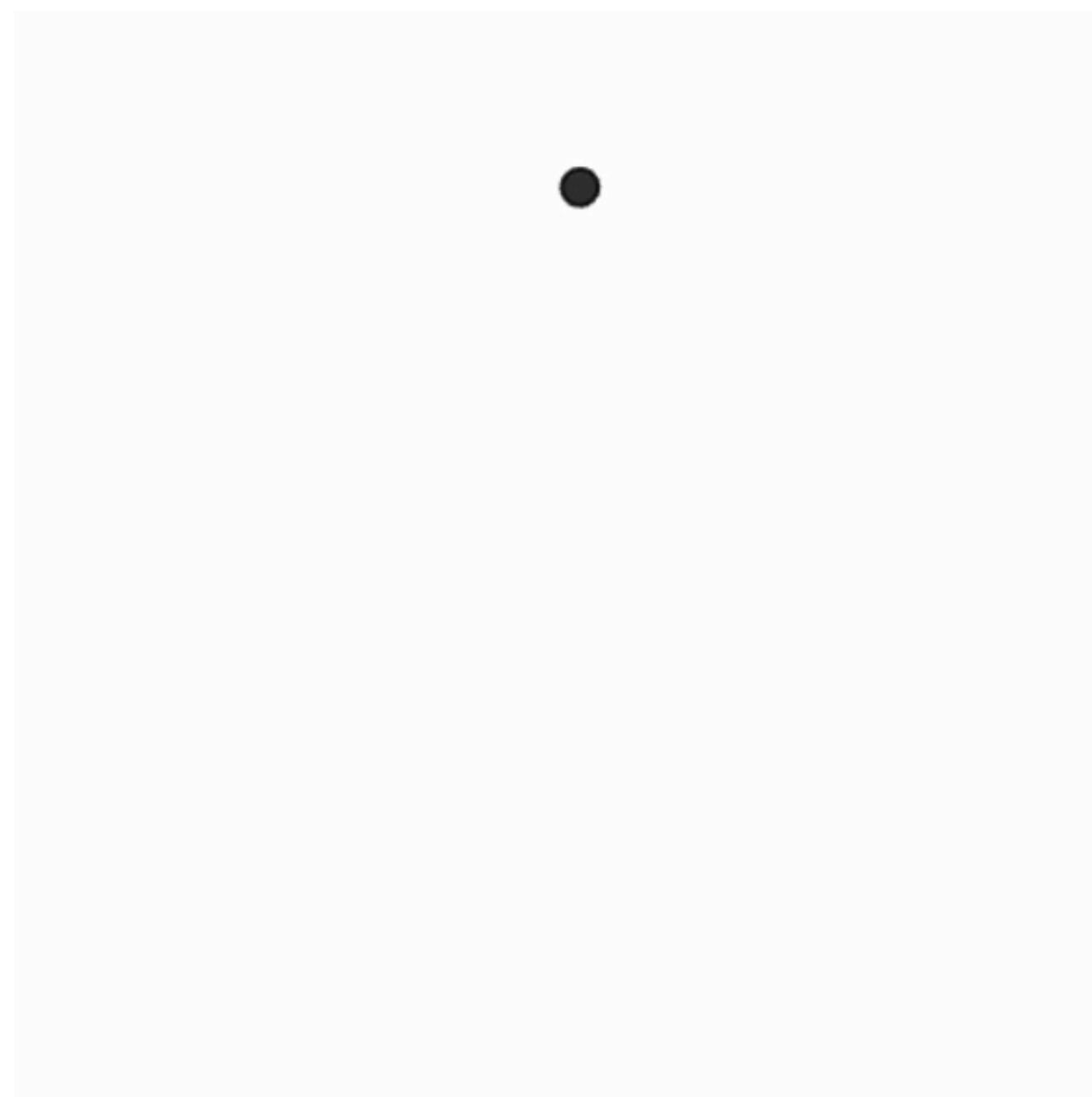
- ▶ **Sistemas de Emissores**

Uma estrutura de dados que gerencia múltiplos emissores

# Partícula



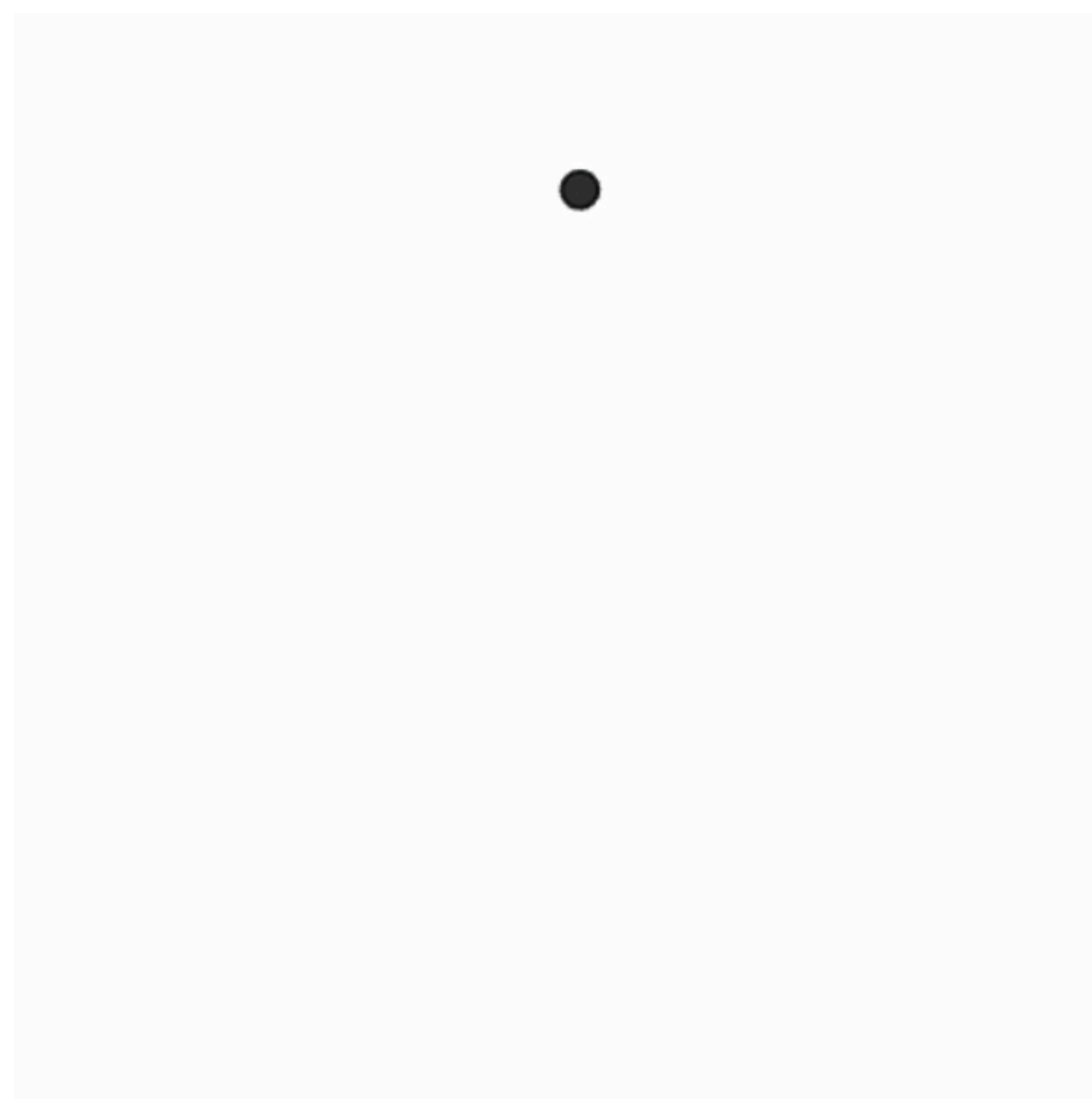
Uma **partícula** é um objeto rígido sem rotação com tempo de vida limitado, i.e. morre após um certo tempo.



# Partícula: Propriedades



Uma **partícula** é um objeto rígido sem rotação com tempo de vida limitado, i.e. morre após um certo tempo. Além das **propriedades** física (`pos`, `vel` e `acc`), uma partícula também possui:

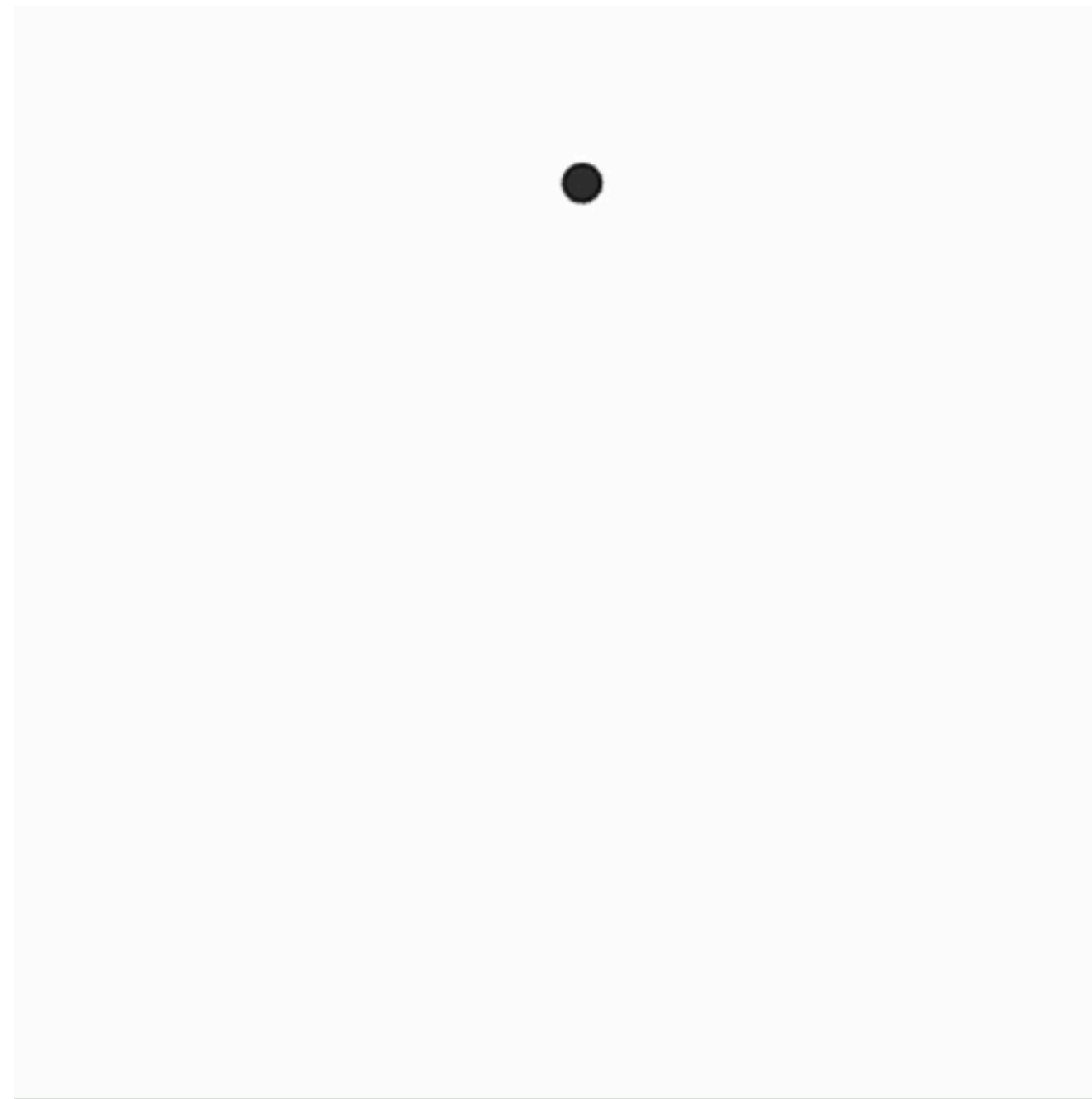


- ▶ **Lifetime**: tempo que a partícula permanecerá ativa
- ▶ **Color**: cor em (RGBA) para renderização
- ▶ **Size**: tamanho da geometria da partícula
  - ▶ Circunferência → raio
  - ▶ Retângulo → (largura e altura)
  - ▶ ...
- ▶ **IsDead**: estado de vida atual da partícula

# Partícula: Métodos



Uma **partícula** é um objeto rígido sem rotação com tempo de vida limitado, i.e. morre após um certo tempo. Os principais **métodos** de uma partícula são:



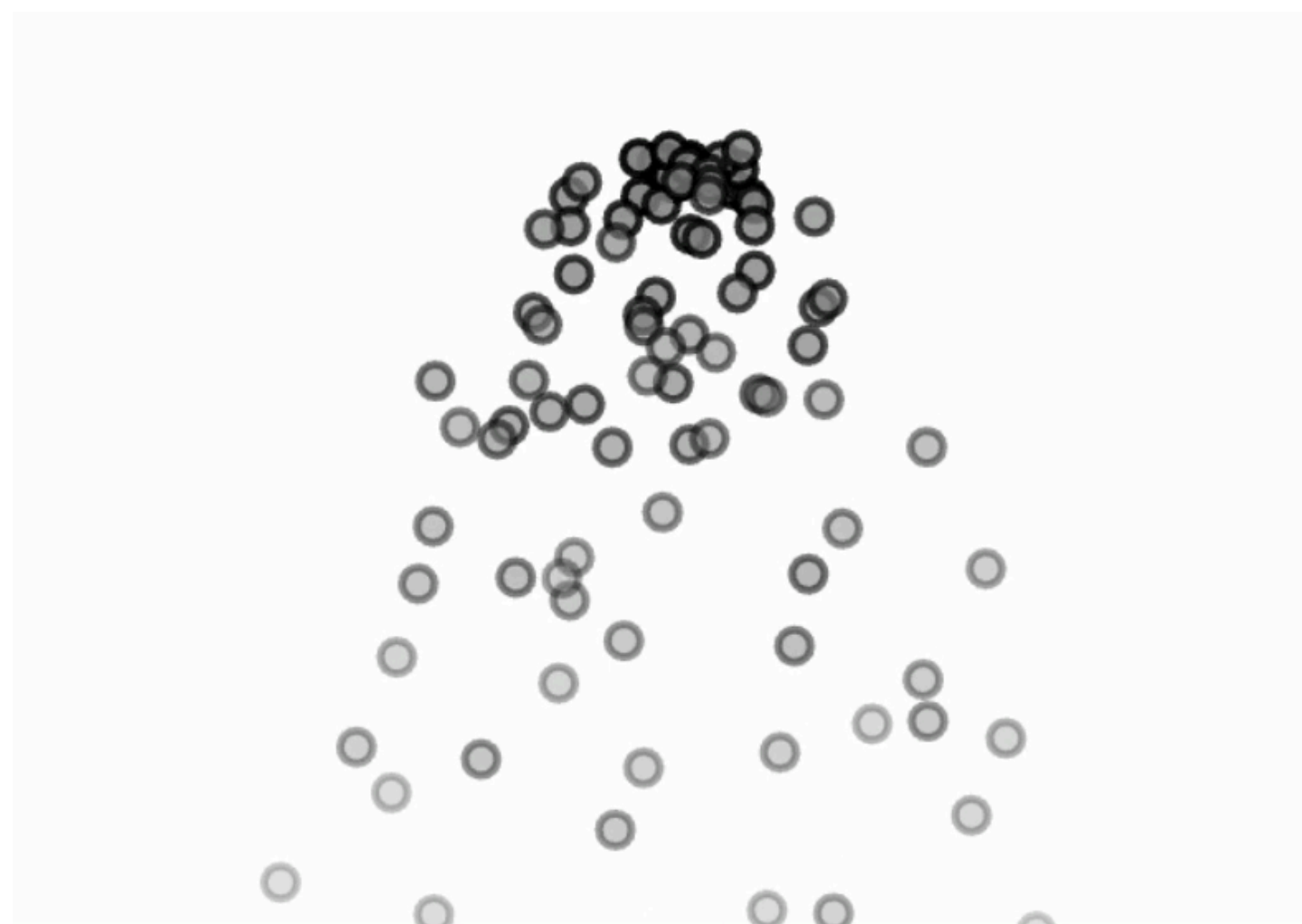
- ▶ **Update ( ) :**
  - ▶ Atualiza simulação física se ainda estiver viva
  - ▶ Decrementa tempo de vida
  - ▶ Caso tenha chegado em zero:
    - ▶ Marca como morta
- ▶ **Draw ( ) :**
  - ▶ Desenha a partícula
  - ▶ Considerar alteração de cor em função do tempo



# Emissor de Partículas



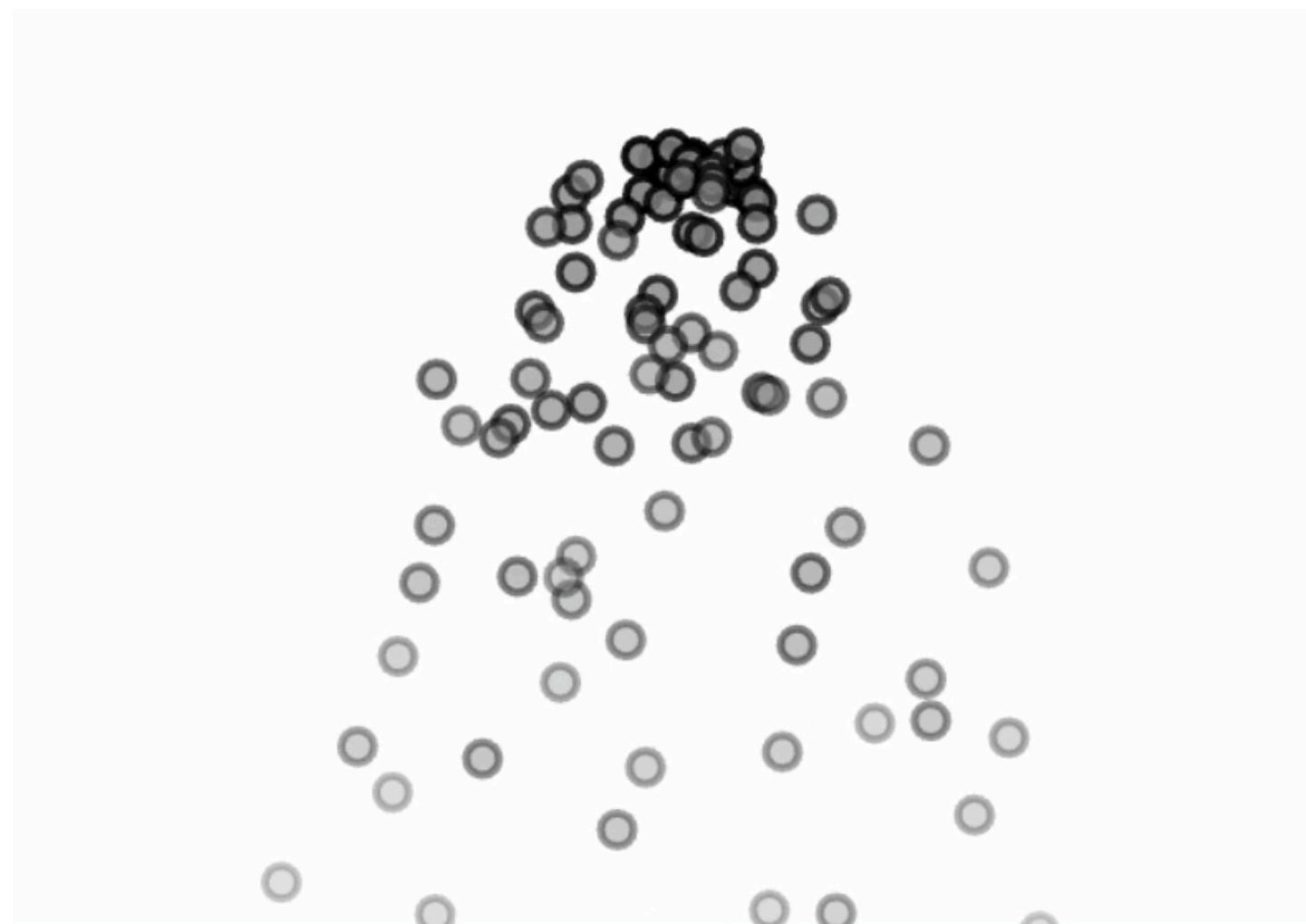
Um **emissores de partículas** é uma estrutura de dados que gerencia a criação de remoção de partículas.



# Emissor de Partículas



Um **emissores de partículas** é uma estrutura de dados que gerencia a criação de remoção de partículas. As principais **propriedades** de uma emissor são as seguintes:

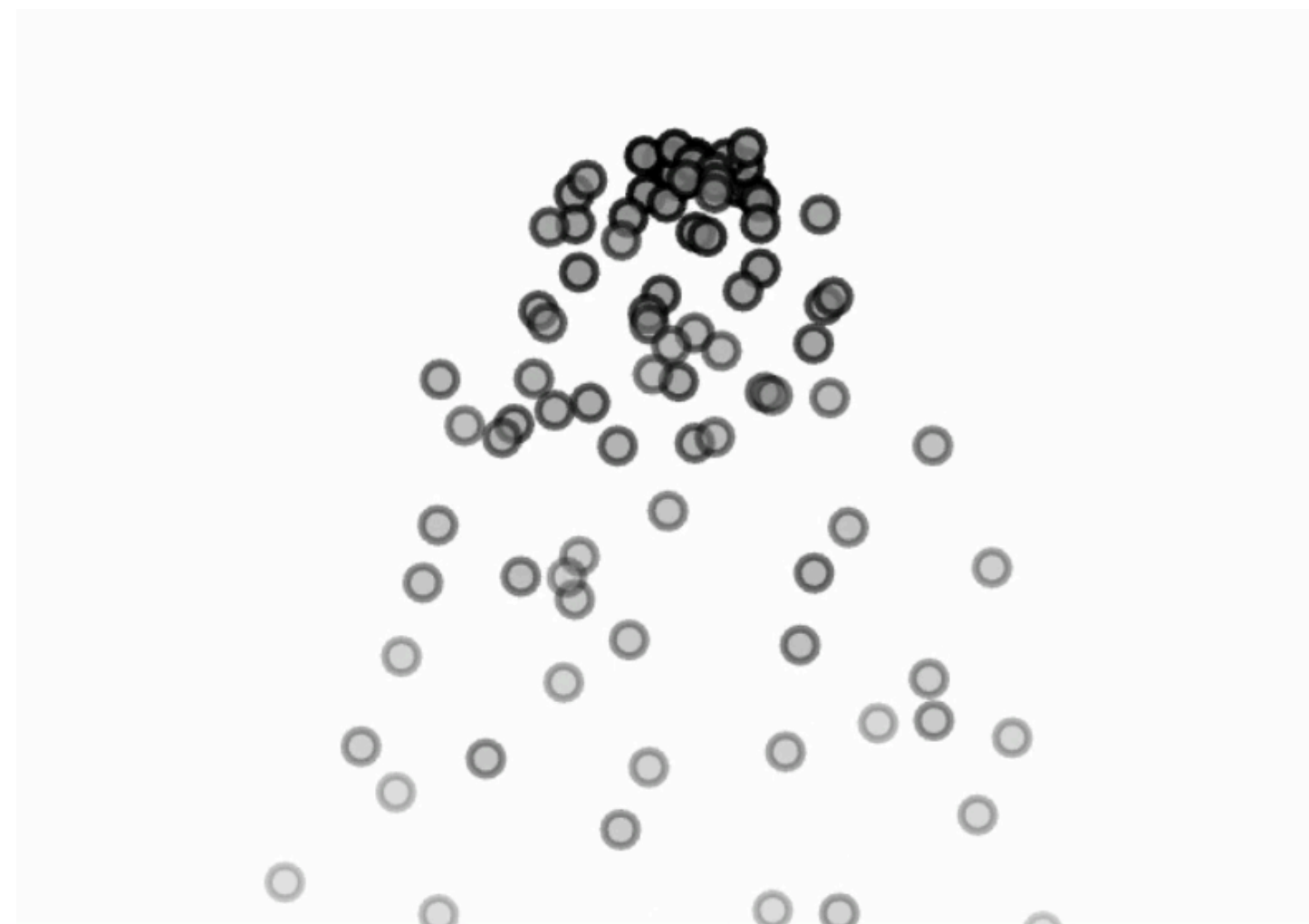


- ▶ **Particles:** lista de partículas do sistema
- ▶ **Origin:** posição onde as partículas irão nascer

# Emissor de Partículas



Um **emissor de partículas** é uma estrutura de dados que gerencia a criação de remoção de partículas. Os principais métodos de uma emissor são as seguintes:

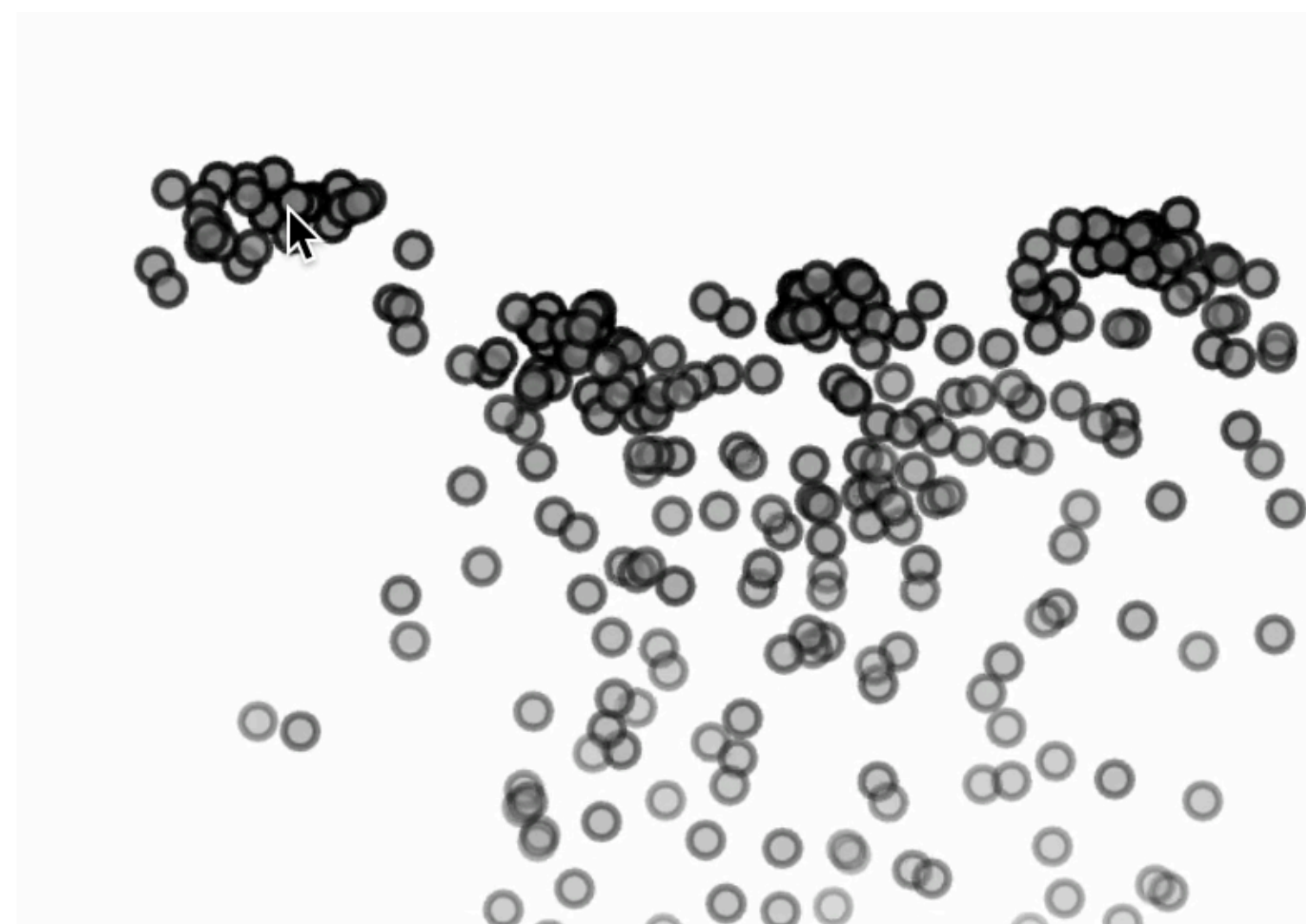


- ▶ **AddParticle()** : adiciona uma partícula na origem do emissor
- ▶ **DelParticle(p)** : deleta uma partícula **p**
- ▶ **Update()** : para cada partícula **p** em **Particles**:
  - ▶ Atualiza **p** e verifica se ela morreu
  - ▶ Caso **p** tenha morrido: **DelParticle(p)**
- ▶ **Draw()** : Desenha cada partícula **p** em **Particles**:

# Sistemas de Emissores



Um **sistema de emissores** é uma estrutura de dados que gerencia a criação de remoção de emissores de partículas. As propriedades de uma sistema de emissores são as seguintes:



- ▶ **Emitters**: lista de emissores do sistema
- ▶ **AddEmitter()**: adiciona um emissor
- ▶ **DelEmitter(p)**: deleta um emissor

Normalmente, haverá uma série de sistemas de partículas trabalhando juntos para criar um único efeito visual. O sistema de emissores representa um único efeito visual.

# Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
  - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
  - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

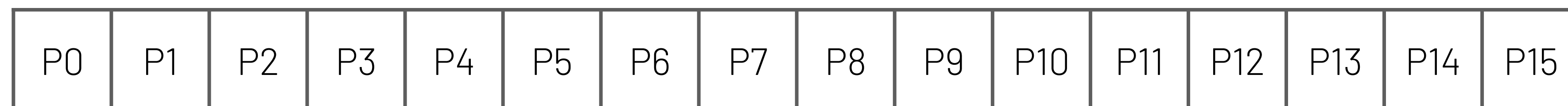


# Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
  - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
  - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las



Lifetime      -      -      -      -      -      -      -      -      -      -      -      -      -      -      -

# Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
  - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
  - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

# Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
  - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
  - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	3	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-

# Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
  - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
  - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	3	1	3	-	-	-	-	-	-	-	-	-	-	-	-	-

# Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
  - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
  - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	2	-	2	-	-	-	-	-	-	-	-	-	-	-	-	-



# Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
  - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
  - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	1	3	1	-	-	-	-	-	-	-	-	-	-	-	-	-

# Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
  - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
  - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

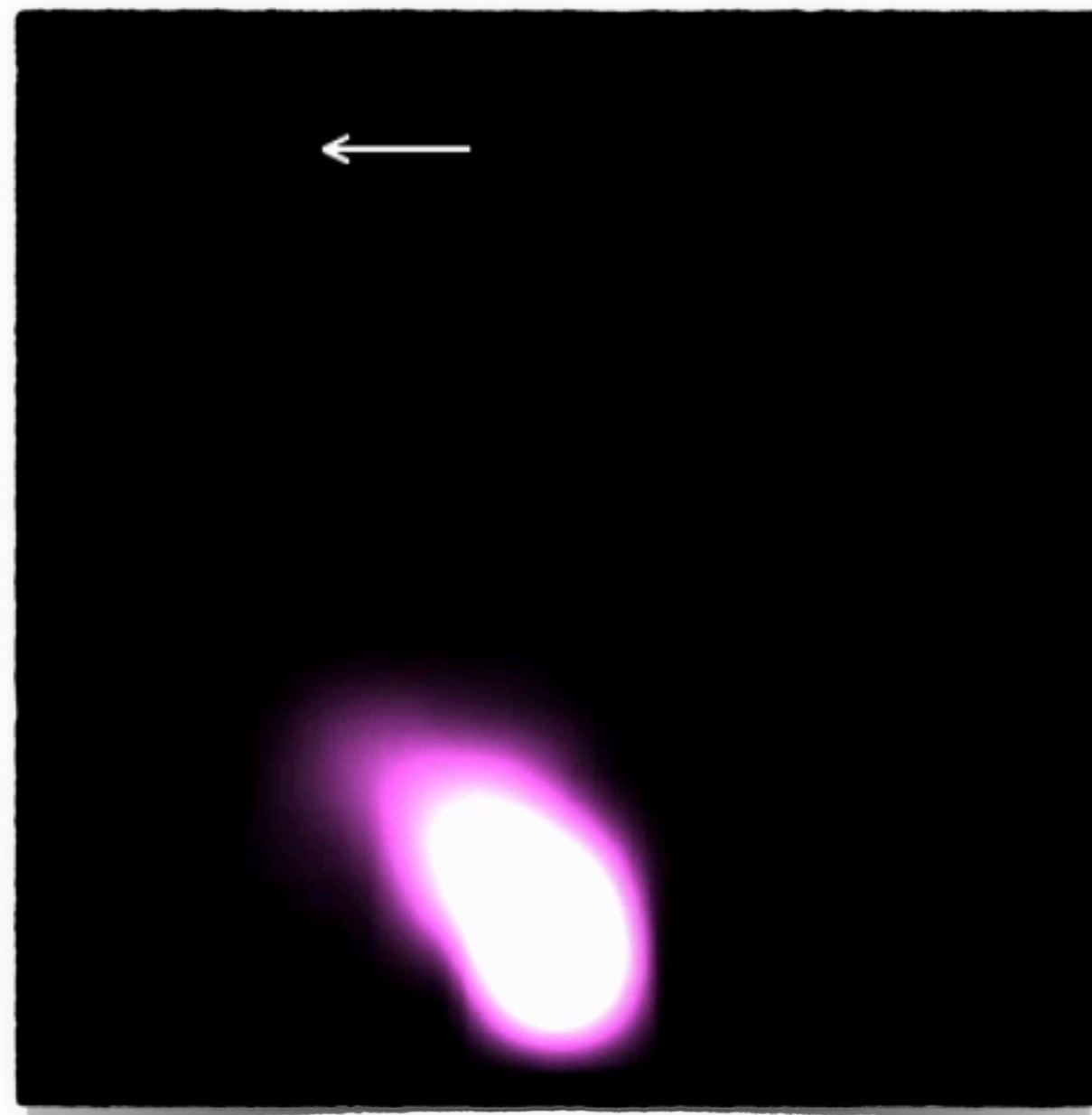
	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	1	3	1	2	4	2	3	6	2	3	3	1	3	1	2	1

**Se todas as partículas estiverem ocupadas, não será possível criar partículas por um tempo!**

# Modelando Comportamentos Orgânicos



Para **modelar comportamentos orgânicos**, podemos configurar diferentes parâmetros do sistema de partículas:



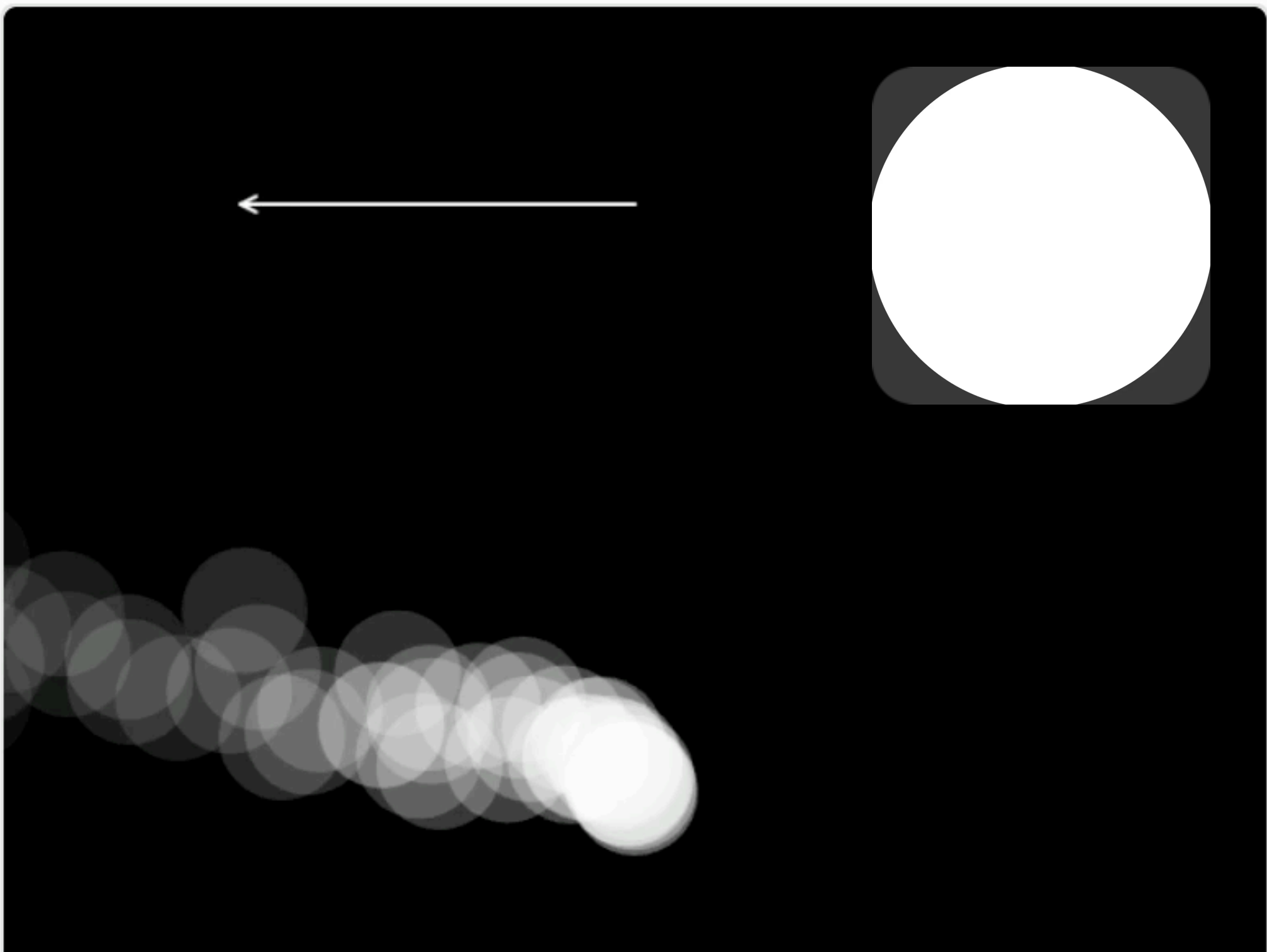
- ▶ Número, posição e tempo de vida de partículas
  - ▶ Tempo aleatório em um intervalo [min, max]
- ▶ Número e posição de emissores
- ▶ Frequência de emissão dos emissores
- ▶ ...

**Além disso, podemos aplicar forças às partículas!**

# Exemplo 1: Fogo



Cor sólida com transparência

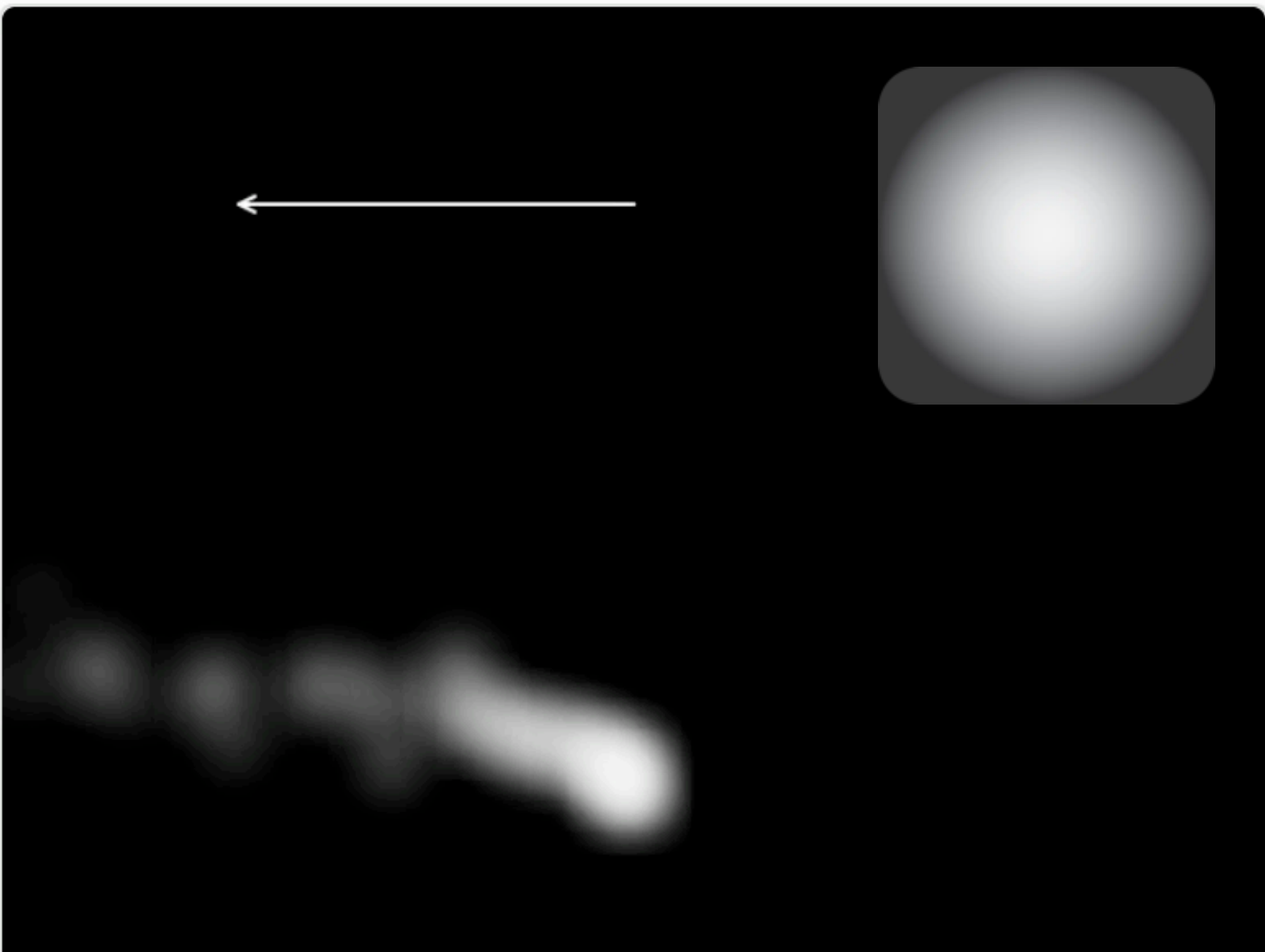


↺ Reset

⏸ Pause

[Open in Web Editor ↗](#)

Textura com fade out



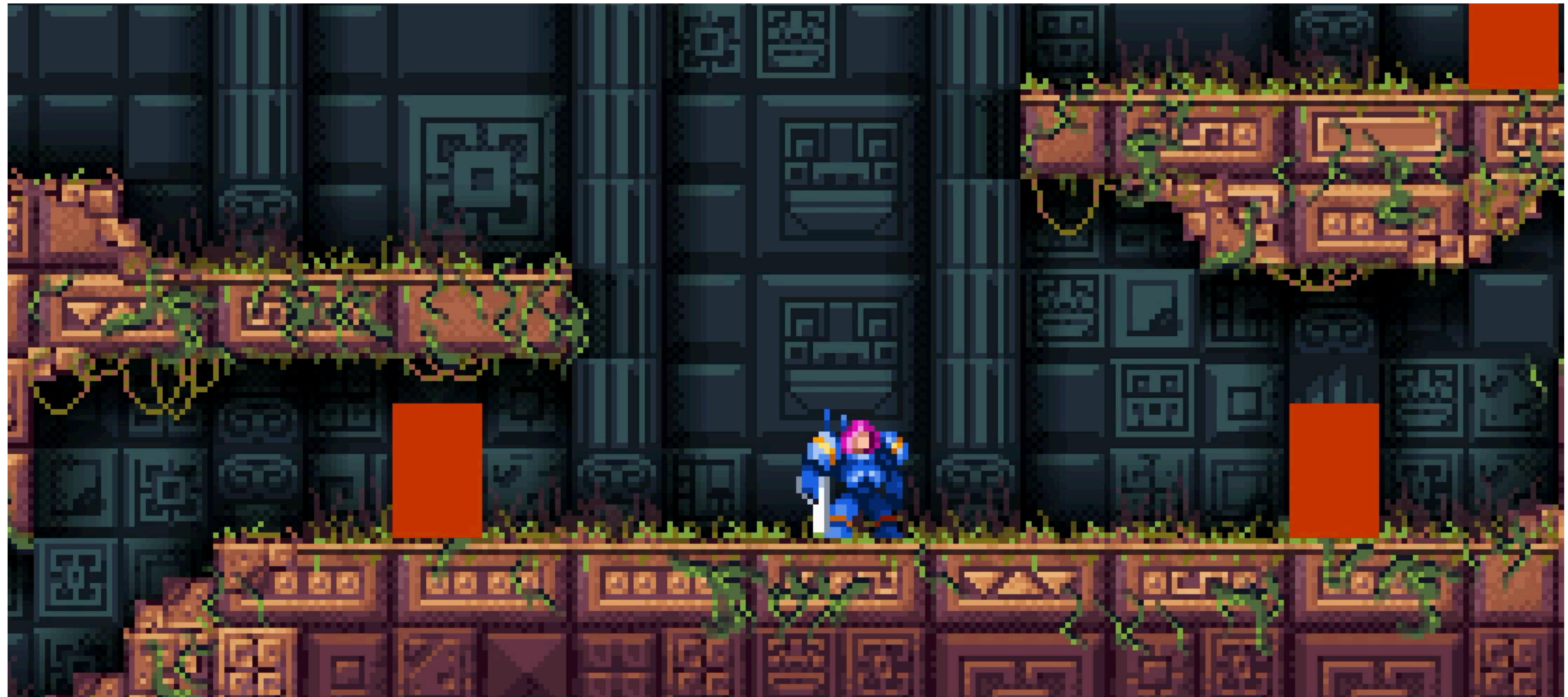
↺ Reset

⏸ Pause

[Open in Web Editor ↗](#)



# Exemplo 2: Projéteis





## A12: Animações e Câmeras 2D

- ▶ Animações 2D
  - ▶ Sprites e Spritesheets
- ▶ Tilemaps
- ▶ Câmera 2D
  - ▶ Controle de câmera
  - ▶ Efeitos de Paralaxe e Chacoalhão
  - ▶ Suavização de movimento