Singleton

por Matheus Anzzulin

Oque é

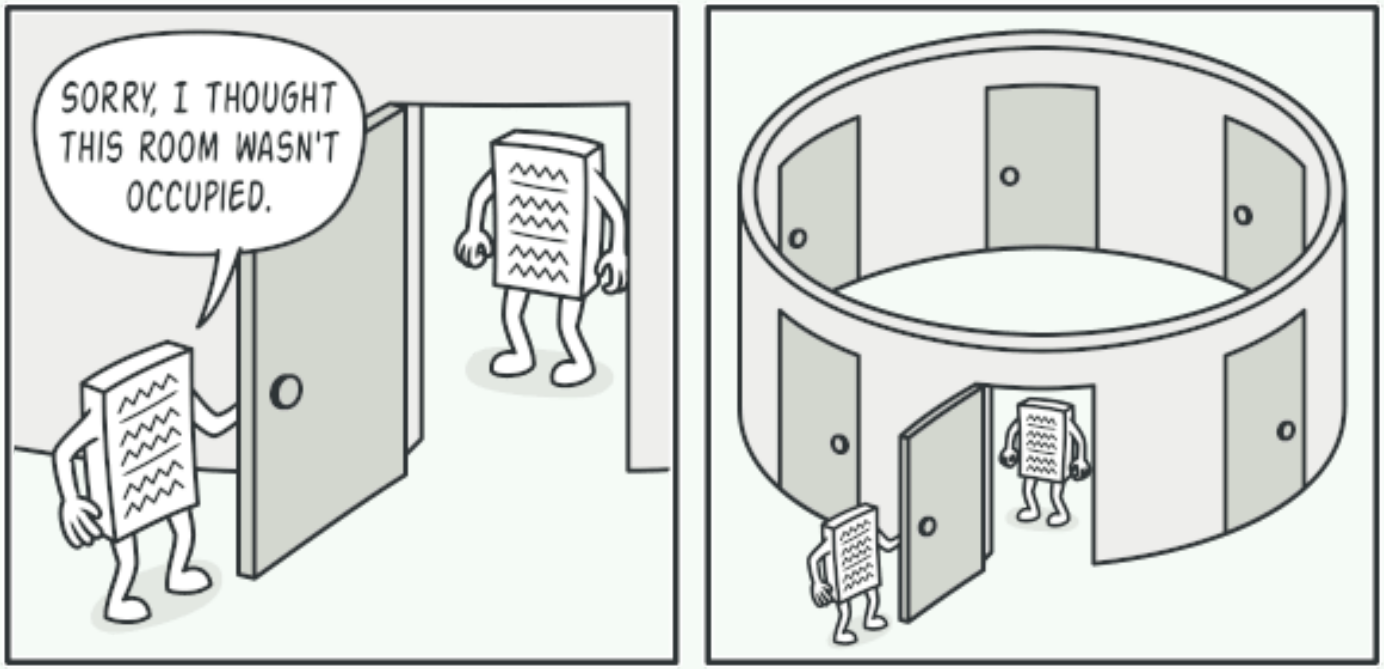
É uma pattern muito utilizada em linguagens de orientação a objeto, aonde restringimos a inicialização de uma classe para uma unica instancia. Geralmente utilizada quando precisamos de um estado global na aplicação para coordenar ações pelo sistema.

Pontos Chave

- Controle mais estrito sobre o estado do sistema
- Inicialização unica
- Estado Global (geralmente ruim)

Casos de uso

Quanto uma parte do sistema deve ter apenas uma instancia disponivel para todos os clientes, por exemplo, um unico banco de dados compartilhado por diferentes partes do sistema.



Como utilizamos esse conceito em elixir

Como elixir não é uma linguagem orientada a objetos, se quisermos ter um estado global que pode ser lido e modificado em diversas partes do sistema, temos algumas opções:

- ETS (Erlang Term Storage)
- Mensagens entre processos
- Integração com sistemas externos

Agents em elixir

Segundo o guia getting-started de elixir (https://elixir-lang.org/getting-started)

> Agents are simple wrappers around state. If all you want from a process is to keep state, agents are a great fit.

```elixir
defmodule KV.Bucket do
  use Agent

  @doc """
  Starts a new bucket.
  """
  def start_link(_opts) do
    Agent.start_link(fn -> %{} end)
  end

  @doc """
  Gets a value from the `bucket` by `key`.
  """
  def get(bucket, key) do
    Agent.get(bucket, &Map.get(&1, key))
  end

  @doc """
  Puts the `value` for the given `key` in the `bucket`.
  """
  def put(bucket, key, value) do
    Agent.update(bucket, &Map.put(&1, key, value))
  end
end
```