**Padrão de Projeto Singleton**

Singleton é um padrão de projeto criacional que tem como objetivo garantir que um objeto seja instanciado apenas uma vez durante o ciclo de vida da aplicação.

Suas **principais características** são:

* Um construtor padrão privado, permitindo apenas que a própria classe crie uma instância de si mesma;
* Um método estático que funciona como um construtor, chamando “por trás das cortinas” o construtor privado e retornando a instância única da classe, atribuída a um campo/atributo estático.

**Vantagens**:

* Garantia de instância única da classe;
* Ganho de um único ponto de acesso global à instância;
* O objeto é inicializado apenas quando chamado pela primeira vez.

**Desvantagens**:

* Como resolve 2 problemas de uma só vez (garantia de acesso a única instância da classe e ponto de acesso global a essa única instância), o padrão fere o Princípio de Responsabilidade Única (Open Closed Principle – OCP) do S.O.L.I.D.;
* Pode mascarar problemas com design;
* Pode requerer tratamento especial para ambientes multithreaded (múltiplas threads podem criar múltiplos objetos singletons, caso não sejam tratados);
* Dificuldade em realizar testes unitários. Pode requerer soluções criativas para simulação do Singleton ou simplesmente inviabilizar o teste.

**Referências**:

* [Singleton (refactoring.guru)](https://refactoring.guru/pt-br/design-patterns/singleton)
* [Padrão Singleton: Como funcionam? Onde vivem? Do que se alimentam? | by Christian Lima | Medium](https://medium.com/@christianmellolima/padrão-singleton-como-funcionam-onde-vivem-do-que-se-alimentam-6291fb72b22d)