Singleton

Intenção: Garantir que uma classe tenha somente uma instância no programa e fornecer um ponto de acesso global para a mesma.

Tipo: creational

Visão Geral

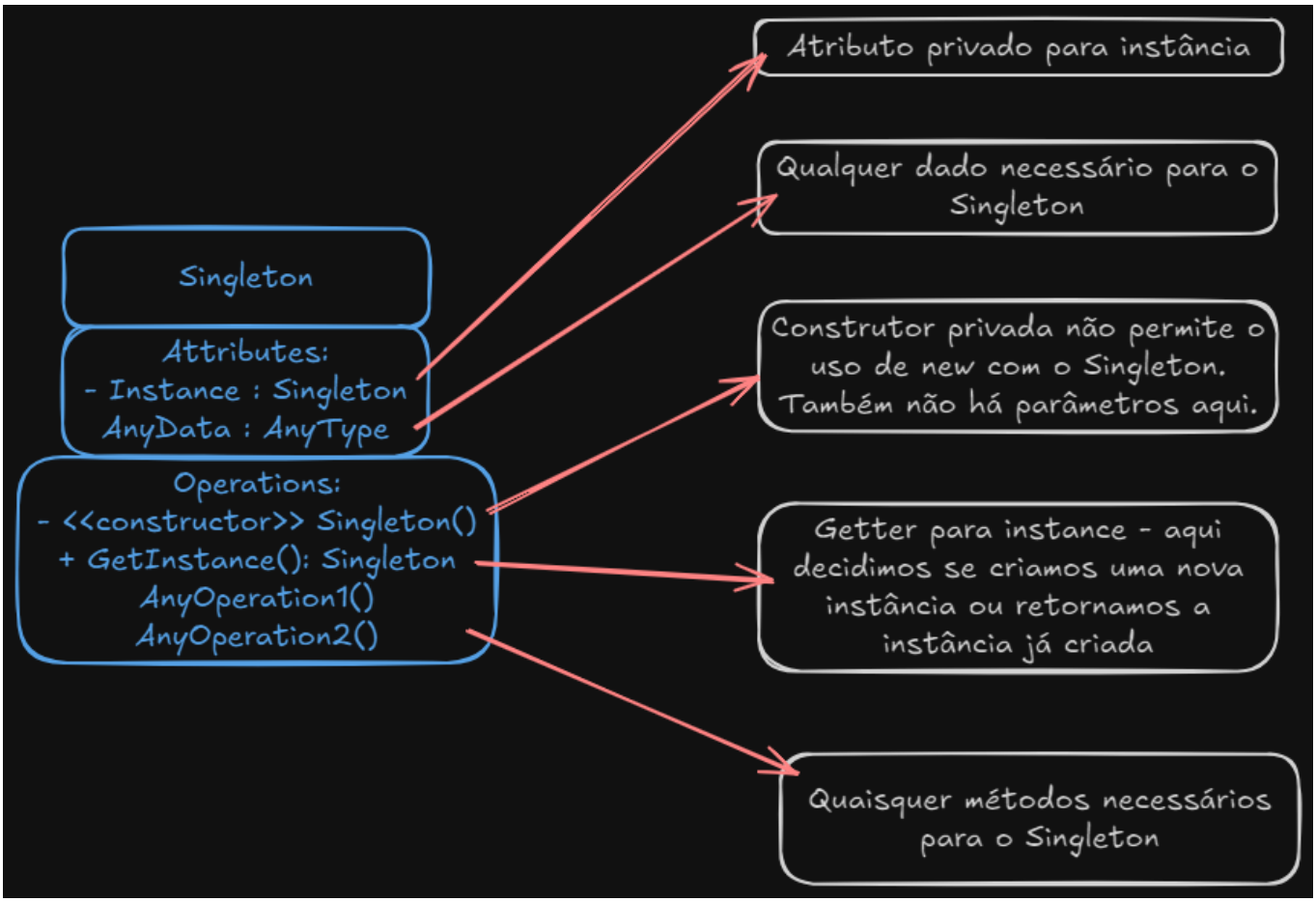
Somente uma instância?

- Geralmente usado para acesso a **recursos compartilhados**, como acesso à base de dados, interfaces gráficas, sistemas de arquivos, servidores de impressão, logger e mais.
- Também usado para substituir variáveis globais, como em casos de uso de **objetos de configuração do sistema** como um todo.

Ponto de acesso global?

- Você pode permitir acesso global ao *Singleton* em toda sua aplicação, assim como fazíamos (ou fazemos) com variáveis globais.
- Uma vantagem do *Singleton* é que podemos proteger a instância com encapsulamento, evitando que outro código sobrescreva seu valor.

Estrutura



Implementação

```
public sealed class Singleton
{
    private static Singleton? _instance;
    private Singleton()
    {
        // Construtor privado para impedir instância externa
    }

    public static Singleton Instance
    {
        get
        {
            // Verifica se a instância já foi criada
            if(_instance == null)
            {
                _instance = new Singleton();
            }
            return _instance;
```

```
        }
    }
}

var instance1 = Singleton.Instance;
var instance2 = Singleton.Instance;

Console.WriteLine(typeof(instance1 == instance2));
```

`instance1` e `instance2` são o **mesmo objeto**.

Aplicabilidade

- Use o *Singleton* quando uma classe precisa ter somente uma instância disponível em todo o seu programa.
- Use o *Singleton* quando perceber que está usando variáveis globais para manter partes importantes do programa, como variáveis de configuração que são usadas por toda a aplicação.

Consequências

Bom:

- Acesso controlado à instância única
- É fácil permitir um número maior de instâncias caso mude de ideia
- Usa *lazy instantiation*, o *Singleton* só é criado no momento do uso
- Substitui variáveis globais

Ruim:

- É mais difícil de testar
- Viola o princípio da responsabilidade única
- Requer tratamento especial em caso de concorrência
- Erich Gamma (autor) descreveu que este seria o único padrão que ele removeria se fosse refatorar o livro