



# { Singleton }



```
main() {
    int a[2][3];
    cout << "Enter elements of first matrix ";
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 3; j++) {
            cin >> a[i][j];
        }
    }
    cout << endl;
    int b[3][2];
    cout << "Enter elements of second matrix ";
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 2; j++) {
            cin >> b[i][j];
        }
    }
    cout << endl;
    cout << "Product is : ";
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            int sum = 0;
            for (int k = 0; k < 3; k++) {
                sum += a[i][k] * b[k][j];
            }
            cout << sum << " ";
        }
        cout << endl;
    }
}
```

# { Singleton }

- Foi criado para economizar memória.
- Garante a existência de apenas uma instancia de uma classe, mantendo um ponto global de acesso a esse objeto.

## Quando usar ?

- Se 'n' lugares da tua aplicação precisam acessar a mesma informação, e essa informação não pode ser diferente para mais de um acesso.

## CUIDADO!

- Se você tem um objeto que é pouco requisitado, e você implementa o padrão singleton, ele vai ser um objeto carregado em memória sem necessidade.

# { Singleton }

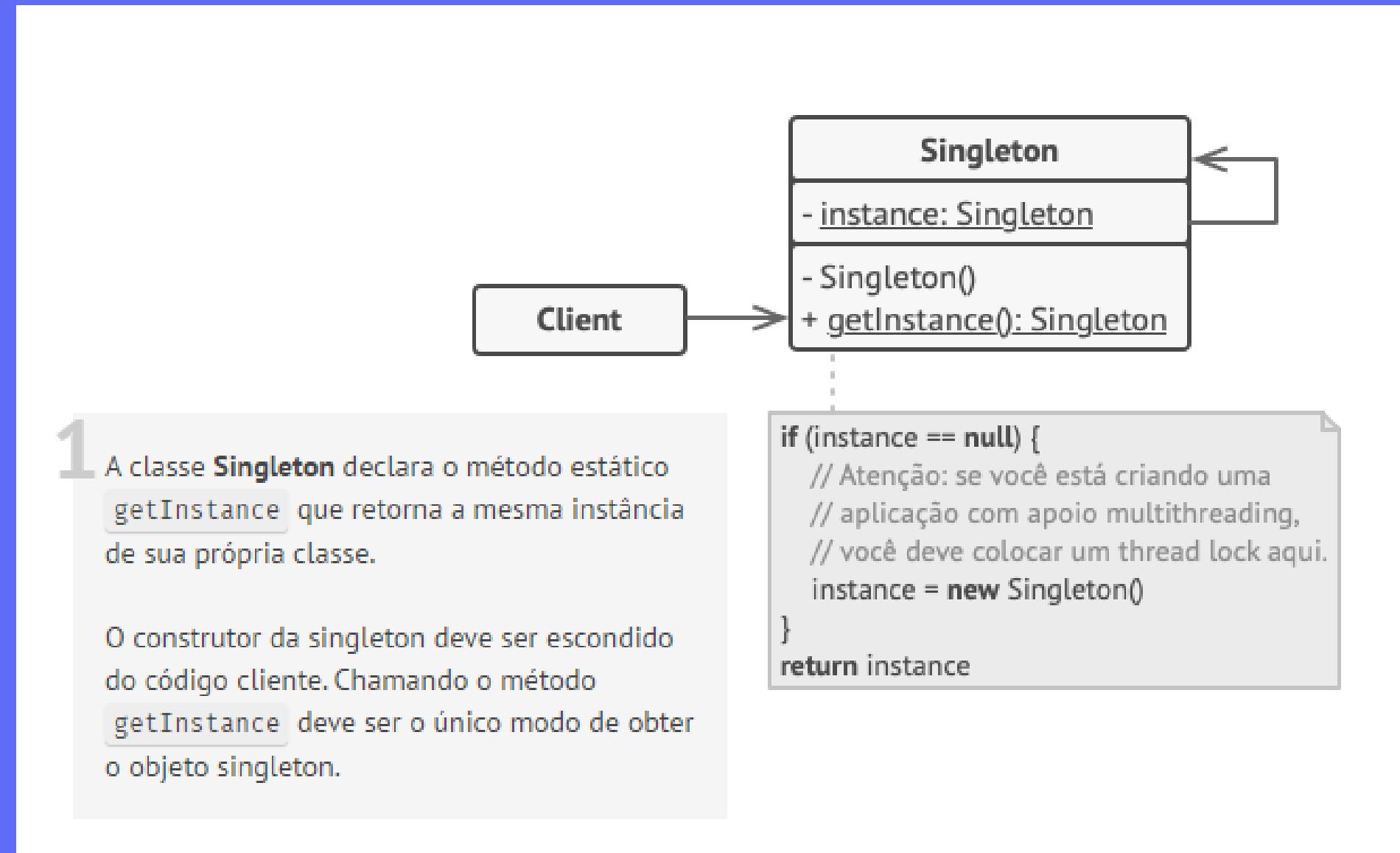
## Analogia com o mundo real!

- O governo é um excelente exemplo de um padrão Singleton. Um país pode ter apenas um governo oficial. Independentemente das identidades pessoais dos indivíduos que formam governos, o título, “O Governo de X”, é um ponto de acesso global que identifica o grupo de pessoas no comando.



# { Singleton }

- Estrutura



# Show me the code!

