Singleton

Prof. Igor Avila Pereira
igor.pereira@riogrande.ifrs.edu.br

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
Câmpus Rio Grande
Divisão de Computação

Agenda

Introdução

Nossa próxima parada é o Padrão Singleton, nosso passaporte para criar objetos únicos para os quais há apenas uma instância.

Há muitos objetos dos quais precisamos apenas uma instância:

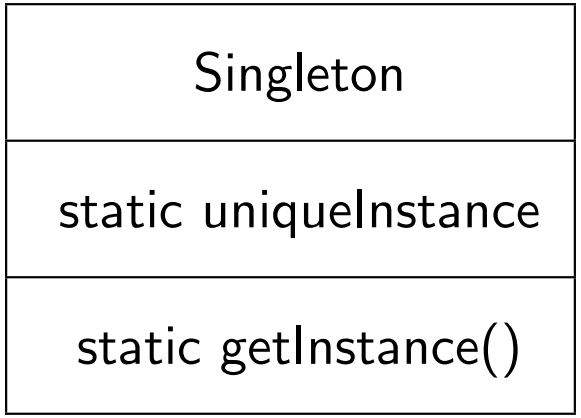
1. caixas de diálogo, objetos que cuidam de preferências e,
2. configurações de registro
3. objetos usados para registro e;
4. objetos que agem como drivers de dispositivos para dispositivos como impressoras e placas gráficas.

Introdução

Na verdade, para muitos desses tipos de objetos, se instanciássemos mais uma vez, encontraríamos todos os tipos de problema, como comportamento inadequado do programa, uso excessivo de recursos e resultados inconsistentes

De muitas maneiras, o Padrão Singleton é uma convenção para garantir que um e apenas um objeto seja instanciado para uma dada classe.

UML



- A variável de classe uniqueInstance contém nossa única instância de Singleton
- Uma classe implementando o Padrão Singleton é mais que um Singleton: é uma classe de fins gerais com seu conjunto de dados e métodos
- O método getInstance() é estático, o que significa que é um método de classe, então você pode acessá-lo, convenientemente, a partir de qualquer lugar de seu código usando Singleton.getInstance()

Código

```
public class Singleton {
        private static Singleton uniqueInstance;
        private Singleton() {
        }
        public static synchronized Singleton getInstance() {
                if (uniqueInstance == null)
                        uniqueInstance = new Singleton();
                return uniqueInstance;
        }
}
```

Acima temos a implementação do padrão Singleton.

- Nota-se a presença do *synchronized*, isso se deve pois se tirássemos o *synchronized* e tentássemos criar duas instâncias da classe num determinado momento verificaríamos que isso é possível.
- Utilizando *synchronized* tem-se a certeza que o método nunca será acessado por duas *threads* ao mesmo tempo.

Código

Também existem outras abordagens para a criação da instância da classe Singleton.

- Por exemplo, se uma determinada classe Singleton sempre é criada e usada, pode-se usar o código abaixo:

```
public class Singleton {
        private static Singleton uniqueInstance = new Singleton();
        private Singleton() {
        }
        public static Singleton getInstance() {
                return uniqueInstance;
        }
}
```

Prefira a abordagem da implementação acima caso o método *getInstance()* também seja muito acessado, pois usar *synchronized* pode diminuir a performance da aplicação.

Conclusão

O Padrão Singleton é utilizado quando necessita-se de um ponto único para criação de uma instância de classe e quando precisamos de apenas uma instância de uma classe.

Tem-se diversas formas de implementar o padrão Singleton e deve-se optar pela implementação que melhor atende aos requisitos da aplicação.

Singleton

Prof. Igor Avila Pereira
igor.pereira@riogrande.ifrs.edu.br

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
Câmpus Rio Grande
Divisão de Computação