01. Introdução à Threads

Date	@24/08/2022
Categoria	Java
Curso	Curso Threads em Java 1: programação paralela

Tópicos

1. Introdução às Threads
2. Off Topic: O Thread, A Thread
3. Visualizando a primeira Thread
4. Threads da JVM
5. Criando a primeira Thread
6. A tarefa da Thread
7. Mãos à obra: Thread simples
8. Mãos à obra: Calculador demorado
9. Mãos à obra: Calculador rápido
10. Para saber mais: Estendendo a classe Thread
11. Para saber mais: Classes anônimas
12. O que aprendemos?

Para saber mais: Estendendo a classe Thread

Há uma outra forma de criar uma tarefa da thread, sem precisar da interface `Runnable` explicitamente.

A classe `Thread` na verdade já implementa a interface `Runnable`. Ou seja, a thread também é um `Runnable`!

Sabendo disso, você pode criar uma subclasse dela e sobrescrever o método `run`. Assim, você não precisaria da classe separada para implementar a tarefa a executar.

Por exemplo, veja a classe `Multiplicador` que estende a classe `Thread`:

```
public class Multiplicador extends Thread {
  public void run () {
// calculo demorado
  }
}COPIAR CÓDIGO
```

E, como nossa classe é uma `Thread`, podemos usar o `start()` diretamente:

```
Multiplicador multiplicador = new Multiplicador();
multiplicador.start();COPIAR CÓDIGO
```

Parece ser interessante, mas você pode enxergar uma desvantagem disso?

Apesar de ser um código mais simples, você está usando ou abusando a herança. Essa forma de herança também é chamada de "herança por preguiça".

Repare que, ao estender a classe `Thread`, herdamos um monte de métodos mas usamos apenas o `run`. Além disso, não estamos querendo aproveitar o polimorfismo que a herança traz.

Hoje em dia, apesar de ser funcional, essa forma de criar uma thread é considerada um mau exemplo de herança.

Prefira sempre implementar o `Runnable` a herdar de Thread. Separando as responsabilidades de *ser uma thread* da *definição da tarefa*, seguimos as boas práticas do mundo OO. Neste treinamento sempre usaremos uma tarefa (`Runnable`) separada da classe `Thread`, nunca usaremos herança.

No blog da Caelum, o Paulo Silveira fala um pouco mais sobre esse assunto e dá outros exemplos de mau uso da herança:

Como não aprender orientação a objetos - Herança

Boa leitura :)

O que aprendemos?

- Através das Threads podemos executar tarefas em paralelo;
- Uma classe que implementa a interface `Runnable` define a tarefa que o Thread executará;
- O construtor da classe `Thread` recebe esse `Runnable`;
- Devemos inicializar uma Thread explicitamente através do método start();
- Através do `Thread.sleep(millis)` podemos mandar uma thread dormir.