10

Para saber mais: Threads e Java 8

Sabemos que, para criar uma thread, é preciso criar dois objetos:

- a tarefa do tipo `Runnable`
- a thread em si

Por exemplo, segue uma implementação mais simples possível de um `Runnable`:

```
class Tarefa implements Runnable {

    public void run() {
        System.out.println("rodando");
    }
}

//no método main, para inicializar a tarefa
Runnable tarefa = new Tarefa();
new Thread(tarefa).start();
```

Ok, temos nenhuma novidade até aqui. Você também já viu, quando a tarefa está muito simples, podemos usar uma classe anônima para representar a tarefa em vez de criar uma classe dedicada:

```
new Thread(new Runnable() {

    public void run() {
        System.out.println("rodando");
    }
}).start();
```

Já é mais enxuto sem a necessidade de criar uma classe dedicada, no entanto, com o Java 8, surgiram formas que permitem escrever um código mais sucinto ainda:

```
new Thread( () -> { System.out.println("rodando");} ).start();
```

Repare que não foi necessário usar a interface `Runnable` explicitamente, passamos apenas um *pedaço* de código: `() -> {Sytem.out.println("rodando");}`

Esse pedaço é chamado de **lambda expressions** e faz parte do Java 8! Já ouviu falar?