09

Voltando a limpar

Transcrição

Agora podemos pensar que nossos convidados são mais exigentes ainda. Assim que alguém usou o banheiro, ele fica sujo novamente, razão suficiente para outros convidados negarem o serviço.

Vamos alterar o booleano na classe `Banheiro`. Nos métodos `fazNumero1()` e `fazNumero2` colocamos a valor `ehSujo = true`.

Segue, como exemplo, o primeiro método:

```
public void fazNumero1() {

    // restante do código omitido

    this.ehSujo = true;

    System.out.println(nome + " dando descarga");
    System.out.println(nome + " lavando a mao");
    System.out.println(nome + " saindo do banheiro");
}
```

Vamos testar novamente o código e rodar a classe `Principal`. Para nossa surpresa tudo continua funcionando, aparentemente os convidados não se preocupam mais com a limpeza!

Na verdade é um problema da nossa implementação. Não basta testar uma vez apenas se o banheiro está sujo ou não. Se um convidado ficou esperando, é justo verificar novamente se banheiro está limpo ou não. Isso é muito fácil de implementar, basta trocar o `if` com `while`. Ou seja, enquanto o banheiro está sujo, ficamos esperando:

```
public void fazNumero1() {

    String nome = Thread.currentThread().getName();

    System.out.println(nome + " batendo na porta");

    synchronized (this) {

        System.out.println(nome + " entrando no banheiro");

        while (this.ehSujo) {
            esperaLaFora(nome);
        }

        // restante do código omitido
}
```

Vamos executar e observar a saída:

```
Pedro batendo na porta
João batendo na porta
Limpeza batendo na porta
Pedro entrando no banheiro
Pedro, eca, banheiro está sujo
Limpeza entrando no banheiro
Limpeza limpando o banheiro
Limpeza saindo do banheiro
João entrando no banheiro
João fazendo coisa rapida
João dando descarga
João lavando a mao
João saindo do banheiro
Pedro, eca, banheiro está sujo
```

E novamente criamos um outro problema. Repare que na minha saída o *Pedro* ficou esperando. Novamente a JVM não termina de executar! O problema é que a nossa limpeza limpa apenas uma vez o banheiro, mas ela precisa voltar e repetir o serviço!

Repetindo a execução

Vamos fazer que a limpeza volte periodicamente, por exemplo a cada 15 segundos. Vamos criar um laço infinito dentro da classe `TarefaLimpeza` :

```
public class TarefaLimpeza implements Runnable {

    //construtor e atributo omitido

    @Override
    public void run() {
        while(true) {
            this.banheiro.limpa();
            try {
                Thread.sleep(15000);//limpando cada 15s
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}
```

Agora teste novamente no seu código. Tudo dele está funcionando e cada convidado só faz as coisas em um banheiro limpo. Segue uma possível saída com apenas dois convidados:

```
Limpeza batendo na porta
Limpeza entrando no banheiro
Limpeza limpando o banheiro
Pedro batendo na porta
João batendo na porta
Limpeza saindo do banheiro
João entrando no banheiro
João fazendo coisa rapida
```

```
João dando descarga
João lavando a mao
João saindo do banheiro
Pedro entrando no banheiro
Pedro, eca, banheiro está sujo
Limpeza batendo na porta
Limpeza entrando no banheiro
Limpeza limpando o banheiro
Limpeza saindo do banheiro
Pedro fazendo coisa demorada
Pedro dando descarga
Pedro lavando a mao
Pedro saindo do banheiro
Limpeza batendo na porta
Limpeza entrando no banheiro
Limpeza limpando o banheiro
Limpeza saindo do banheiro
```

Usando Daemon

Por último, vamos melhorar ainda mais o nosso código. Uma pergunta que pode surgir é: Faz sentido a limpeza continuar verificando o banheiro quando não há mais nenhum convidado com necessidades? Talvez não, né? Vamos alterar o thread da limpeza para funcionar dessa maneira.

Você pode definir que threads dependem de outros. Em outras palavras, há threads que só deveriam rodar enquanto outros existem. Queremos que a máquina virtual automaticamente termine o thread de limpeza quando não existir mais nenhum convidado. Esse tipo de thread se chama **Daemon**.

Podemos definir o thread de limpeza como *Daemon*, através do método `setDaemon(true)`:

```
Thread limpeza = new Thread(new TarefaLimpeza(banheiro), "Limpeza");
limpeza.setDaemon(true);
```

Ao executar, podemos ver que a máquina virtual realmente termina quando todos os convidados foram embora! Perfeito!