06

Mãos à obra: Implementando o banheiro

Vamos testar o acesso ao banheiro com várias threads (*convidados*), igual ao video.

1) Crie um novo projeto **banheiro** (simples *Java Project*)

2) Crie uma classe `Banheiro` no pacote `br.com.alura.banheiro` com os dois métodos abaixo:

```
public class Banheiro {

    public void fazNumero1() {

        System.out.println("entrando no banheiro");
        System.out.println("fazendo coisa rapida");

        try {
            Thread.sleep(8000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

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

    public void fazNumero2() {

        System.out.println("entrando no banheiro");
        System.out.println("fazendo coisa demorada");

        try {
            Thread.sleep(15000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

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

3) Crie uma nova classe `Principal` com um método `main` que instância o `Banheiro` :

```
public class Principal {

    public static void main(String[] args) {

        Banheiro banheiro = new Banheiro();
```

```
    }
}
```

4) Ainda na classe `Principal` crie duas threads, cada uma com a sua tarefa importante:

```
public class Principal {

    public static void main(String[] args) {

        Banheiro banheiro = new Banheiro();

        //Passando a tarefa e o nome do Thread
        Thread convidado1 = new Thread(new TarefaNumero1(banheiro), "João");
        Thread convidado2 = new Thread(new TarefaNumero2(banheiro), "Pedro");

        convidado1.start();
        convidado2.start();
    }
}
```

5) Implemente a `TarefaNumero1` :

```
public class TarefaNumero1 implements Runnable {

    private Banheiro banheiro;

    public TarefaNumero1(Banheiro banheiro) {
        this.banheiro = banheiro;
    }

    @Override
    public void run() {
        this.banheiro.fazNumero1();
    }
}
```

6) E a classe `TarefaNumero2` :

```
public class TarefaNumero2 implements Runnable {

    private Banheiro banheiro;

    public TarefaNumero2(Banheiro banheiro) {
        this.banheiro = banheiro;
    }

    @Override
    public void run() {
        this.banheiro.fazNumero2();
    }
}
```

7) Como nomeamos as threads, podemos aproveitar esse nome e imprimir nos métodos do banheiro. Assim nós sabemos qual convidado está acessando o banheiro. Para pegar o nome do thread, basta acessar o thread atual (`Thread.currentThread()`) .

Chame o método `getName()` dentro do método `fazNumero1()` e `fazNumero2()` . Concatene o nome nas camadas `Syso` :

```
public void fazNumero1() {

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

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

    try {
        Thread.sleep(8000);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }

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

8) Se tudo estiver compilando, execute a classe `Principal` . Fique atento na saída no console. Tem algo estranho?