02

Teste a igualdade entre Strings e outros objetos usando == e equals() - Parte 1

Observe o seguinte código que cria duas Strings:

```
String nome1 = new String("Mario");
String nome2 = new String("Mario");
```

COPIAR CÓDIGO

Como já estudamos anteriormente, o operador `==` é utilizado para comparação. Neste caso, como se tratam de objetos, irá comparar as duas referências e ver se apontam para o mesmo objeto:

```
String nome1 = new String("Mario");
String nome2 = new String("Mario");

System.out.println(nome1 == nome2); // imprime false
```

COPIAR CÓDIGO

Até aqui tudo bem. Mas vamos alterar um pouco nosso código, mudando a maneira de criar nossas Strings, e rodar novamente:

```
String nome1 = "Mario";
String nome2 = "Mario";
```

```
System.out.println(nome1 == nome2); // o que imprime?
```

COPIAR CÓDIGO

Ao executar o código, vemos que ele imprime `true`. O que aconteceu?

Pool de Strings

O Java mantém um ::pool:: de objetos do tipo `String`. Antes de criar uma nova String, primeiro o Java verifica neste pool se uma String com o mesmo conteúdo já existe; caso sim, ele a reutiliza, evitando criar dois objetos exatamente iguais na memória. Como as duas referências estão apontando para o mesmo objeto do pool, o `==` retorna `true`.

Mas por que isso não aconteceu antes, com nosso primeiro exemplo? O Java só coloca no pool as Strings criadas usando **literais**. Strings criadas com o operador `new` não são colocadas no pool automaticamente.

```
String nome1 = "Mario"; //será colocada no pool
String nome2 = new String("Mario");
/*
"Mario" é colocado, mas nome2 é outra
referência, não colocada no pool
*/
```

COPIAR CÓDIGO

Sabendo disso, temos que ter cuidado redobrado quando comparando Strings usando o operador `==`:

```
String s1 = "string";
String s2 = "string";
String s3 = new String("string");

System.out.println(s1 == s2); // true, mesma referencia
System.out.println(s1 == s3); // false, referências diferentes
System.out.println(s1.equals(s3)); // true, mesmo conteúdo
```

COPIAR CÓDIGO

Repare que, mesmo sendo instâncias diferentes, quando comparadas usando o método `equals`, o retorno é `true`, caso o conteúdo das Strings seja o mesmo.