02

Diferenciar entre variáveis de referências a objetos e tipos primitivos

As variáveis de tipos primitivos de fato armazenam os valores (e não ponteiros/referências). Ao se atribuir o valor de uma variável primitiva a uma outra variável, o valor é copiado, e o original não é alterado:

```
int a = 10;
int b = a; // copiando o valor de a para b
b++; // somando 1 em b
System.out.println(a); // continua com 10.
```

COPIAR CÓDIGO

Os programas construídos com o modelo orientado a objetos utilizam, evidentemente, objetos. Para acessar um atributo ou invocar um método de qualquer objeto, é necessário que tenhamos armazenada uma **referência** para o mesmo.

Uma variável de referência é um ponteiro para o endereço de memória onde o objeto se encontra. Ao atribuirmos uma variável de referência a outra, estamos copiando a referência, ou seja, fazendo com que as duas variáveis apontem para o mesmo objeto, e não criando um novo objeto:

```
class Objeto {
    int valor;
}
```

```
class Teste{
    public static void main(String[] args){
        Objeto a = new Objeto();
        Objeto b = a; // agora b aponta para o mesmo objeto de a

        a.valor = 5;

        System.out.println(b.valor); // imprime 5
    }
}
```

COPIAR CÓDIGO

Duas referências são consideradas iguais somente se elas estão apontando para o mesmo objeto. Mesmo que os objetos que elas apontem sejam iguais, ainda são referências para objetos diferentes:

```
Objeto a = new Objeto();
a.valor = 5;

Objeto b = new Objeto();
b.valor = 5;

Objeto c = a;

System.out.println(a == b); // false
System.out.println(a == c); // true
```

COPIAR CÓDIGO

Veremos bastante sobre comparação de tipos primitivos e de referências mais à frente.