**1. MÉTODOS getters() e setters()**

**1.1 Definição de cada um deles**

“Basicamente, os métodos getters e setters são funções que manipulam as variáveis que estão encapsuladas como privadas. Segundo Horstmann e Cornell (2010), há uma diferença conceitual entre o método getters e o método setters. O getters só consulta o estado do objeto e informa sobre isso, ou seja, esse método acessa os campos de instância e é chamado de método de acesso. O setters modifica o estado do objeto, ou seja, altera os campos de instância e é chamado de método modificador.” (MASIERO, OBERLEITER, 2011).

“\*Métodos get

É o método utilizado para retornar o valor de cada um dos atributos de uma classe. Para cada atributo é utilizado um método getX, substituindo o X pelo nome do atributo. Vale ressaltar que o tipo do retorno do método deve ser exatamente o mesmo tipo do atributo.

\*Método set

É o método utilizado para definir o valor de cada um dos atributos de uma classe. Para cada atributo é utilizado setX, substituindo o x pelo nome do atributo. Vale ressaltar que o tipo de retorno do método sempre será void e o mesmo sempre receberá como parâmetro um objeto exatamente o mesmo tipo do atributo.” (MATTOS, 2007, p. 86)

**1.2 Em que contexto esses métodos devem existir? Por que devem ser criados?**

“São métodos criados para cada um dos atributos com o objetivo de permitir a interação com outras classes, sem que tenha que utilizar atributos públicos.” (MATTOS, 2007, p. 86).

“Os métodos de acesso consultam uma informação e a retorna, precisamos indicar qual é o tipo de retorno dessa informação. Isso é indicado pelo tipo da variável retornada pelos métodos getters. No caso do getCPF(), por exemplo, o método vai consultar o valor da variável CPF e retornar um número inteiro (representado pelo int). getCPF(): int.

Já os métodos modificadores, que alteram o campo de instância, precisam da informação para alterar os campos da instância ou os atributos do objeto. Essa informação é passada por parâmetro dentro dos parênteses () do método. Como esse método não retorna nenhuma informação, indicamos com a palavra void.” (MASIERO, OBERLEITER, 2011).

**Referências:**

OBERLEITNER, Allen; MASIERO, Andrey. **PROGRAMAÇÃO ORIENTADA A OBJETOS**. Editora Senac:São Paulo, 2011.

MATTOS, Erico Casella Tavares. **Programação de softwares em Java**. São Paulo: Digerati Books, 2007.