**Métodos Especiais** 

Métodos() podem ser criados com qualquer nome...

Contudo, há uma padronização de nomenclatura entre os programadores quando se tratando dos **Métodos Especiais**:

- getter
- setter
- constructor



Métodos Especiais

#### Forma atual:

```
Para instanciar um novo objeto:

Estante e = nova Estante;

//atributos:
  qtdPortas
  qtdGavetas
  qtdDocumentos

//métodos():
  abrir() / fechar()
  inserirDoc() / removerDoc()

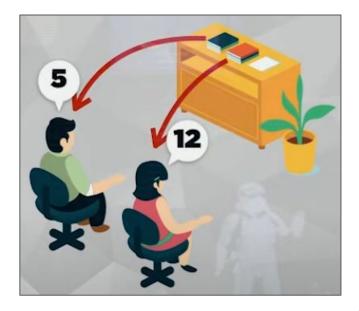
//estado:
  aberta/fechada
  nova/quebrada
```

### <u>Para acessar informações/atributos:</u> int t = e.qtdDocumentos;

<u>Para modificar informações/atributos:</u> e.qtdDocumentos = e.qtdDocumentos + 1

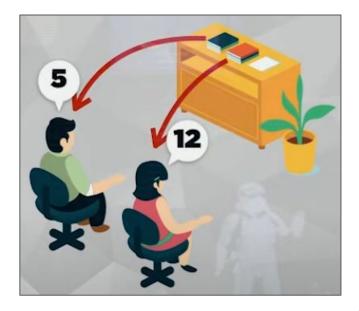
Métodos Especiais

Contudo, várias consultas e modificações podem ser feitas ao mesmo tempo, por pessoas diferentes, podendo haver bagunça e insegurança, gerando inconsistência de dados.



Métodos Especiais

Contudo, várias consultas e modificações podem ser feitas ao mesmo tempo, por pessoas diferentes, podendo haver bagunça e insegurança, gerando inconsistência de dados.



#### **Métodos Acessores**

getters

#### **Métodos Acessores**:

**Getters**: Pegar ou <u>Acessar informações</u> getQtdDocumentos() getSaldoBancario() getModeloVeiculo()

```
//Antes:
    Estante e = new Estante();
    int t = e.qtdDocumentos;

//Agora:
    Estante e = new Estante();
    int t = e.getQtdDocumentos();
```

#### **Métodos Modificadores**

setters

#### **Métodos Modificadores**:

**Setters**: Adicionar ou <u>Modificar informações</u> setQtdDocumentos() setSaldoBancario() setModeloVeiculo()

```
//Antes:
    Estante e = new Estante;
    e.qtdDocumentos = e.qtdDocumentos + 1;

//Agora:
    Estante e = new Estante;
    e.setQtdDocumentos(doc);
```

#### **Assessores / Modificadores**

- **setters** (pegar dados)
- setters (alterar dados)

```
Classe Caneta
    //atributos
   privado modelo: String
    privado ponta: double
    //metodos() especiais: GETters e SETters
    publico Metodo getModelo()
        retorne this modelo
    FimMetodo
    publico Metodo setModelo(mudaModelo: String)
        this.modelo = mudaModelo
    FimMetodo
    publico Metodo getPonta()
        retorne this ponta
    FimMetodo
    publico Metodo setPonta(mudaPonta: double)
        this.ponta = mudaPonta
    FimMetodo
FimClasse
caneta01 = new Caneta
caneta01.setModelo("BIC Comum")
caneta01.setPonta(0.5)
modeloAtual = caneta01.getModelo()
Escreva("O modelo é: " + modeloAtual)
Escreva("A ponta é:" + caneta01.getPonta())
```

**Métodos Construtores** 

constructor

#### **Método Construtor (Construct):**

Serve para <u>fazer coisas de forma</u> <u>automática ao instanciar um novo objeto</u>.

#### Pode ser:

- atribuir valores
- chamar métodos
- atribuir regras
- etc

Posso criar vários métodos construtores, com ou sem parâmetros.

O método construtor tem o mesmo nome que a Classe.

O Java irá reconhecer qual método construtor deve utilizar, de acordo com os parâmetros utilizados no momento da instanciação do objeto.

```
//pseudo codigo - PRIMEIRA FORMA, SEM ATRIBUTOS
Classe Caneta
    // "aqui estão os atributos..."
    // Tudo o que estiver dentro desse método acontece automaticamente
   // sempre que uma nova caneta for instanciada/criada
   Metodo construtor()
       tampar()
       this.cor = "Azul"
   FimMetodo
   // "aqui estão os outroas métodos..."
FimClasse
// Instanciando uma nova caneta
Caneta novaCaneta = new Caneta()
```



```
//pseudo codigo - SEGUNDA FORMA, COM ATRIBUTOS
Classe Caneta
   Metodo construtor(m: String, c: String, p: double)
        setModelo(m)
        setCor(c)
        setPonta(p)
       tampar()
    FimMetodo
FimClasse
// Como meu construtor espera atributos...
// Agora, ao instanciar a caneta, já passo os valores
//dela no momento da criação!
Caneta novaCaneta = new Caneta("BIC", "Azul", 0.5)
// Agora, o Metodo construtor() já foi executado
```

# Atividade avaliativa n1.3

- a. Crie uma Classe/molde para um novo objeto do seu cotidiano (recomendação: um objeto comum do seu trabalho);
- b. Defina os atributos desta classe (recomendação: deixe todos os atributos como privados);
- c. Crie os métodos especiais (getters e setters) para cada atributo da sua Classe;
- d. Crie um método construtor para sua Classe (com ou sem atributos, a depender da sua lógica. Seu objeto DEVE possuir tal atributo ao ser Instanciado/criado?);

- e. Crie alguns objetos utilizando o método construtor
- f. Mude alguns atributos utilizando os setters
- g. Exiba alguns atributos utilizando os getters