

AfroDev

Java Básico - Aula 2 - Parte 1

Modificadores de Acesso

- Tem dois níveis de opção de proteção: classe (top level) e membros (variáveis de classe)
- Para classe tem 2 opções: public e visibilidade de pacote (sem declarar)
- Para membros tem 4 opções: public, private, protected, ou visibilidade de pacote (sem declarar)

Modificadores de Acesso - Tabela

Access Levels

Modifier	Class	Package	Subclass	World
<code>public</code>	Y	Y	Y	Y
<code>protected</code>	Y	Y	Y	N
<i>no modifier</i>	Y	Y	N	N
<code>private</code>	Y	N	N	N

Modificadores de Acesso - Exemplos

Private

```
1 package org.basic.java;
2
3 public class Cafe {
4
5     private String nome;
6
7 }
8
```

```
1 package org.basic.java;
2
3 public class CafeDemo {
4
5     public static void main(String[] args) {
6         Cafe cafe = new Cafe();
7         cafe.nome;
8     }
9
10 }
11
```

The field Cafe.nome is not visible

2 quick fixes available:

- ➔ [Change visibility of 'nome' to 'package'](#)
- ➔ [Create getter and setter for 'nome'...](#)

Press 'F2' for focus

Modificadores de Acesso - Exemplos

Public

```
1 package org.basic.java;  
2  
3 public class Cafe {  
4  
5     public String nome;  
6  
7 }  
8
```

```
1 package org.basic.java;  
2  
3 public class CafeDemo {  
4  
5     public static void main(String[] args) {  
6         Cafe cafe = new Cafe();  
7         System.out.println(cafe.nome);  
8     }  
9  
10 }  
11
```

Modificadores de Acesso - Exemplos

Protected - Mesmo pacote

```
1 package org.basic.java;  
2  
3 public class Cafe {  
4  
5     protected String nome;  
6  
7 }  
8
```


```
1 package org.basic.java;  
2  
3 public class CafeDemo {  
4  
5     public static void main(String[] args) {  
6         Cafe cafe = new Cafe();  
7         System.out.println(cafe.nome);  
8     }  
9  
10 }  
11
```

Modificadores de Acesso - Exemplos



Protected - Outro pacote

```
1 package org.basic.java;  
2  
3 public class Cafe {  
4  
5     protected String nome;  
6  
7 }  
8
```

```
1 package org.basic.cafe;  
2  
3 import org.basic.java.Cafe;  
4  
5 public class CafeProtegido {  
6  
7     public static void main(String[] args) {  
8         Cafe cafe = new Cafe();  
9         cafe.nome;  
10    }  
11  
12 }  
13
```

 The field Cafe.nome is not visible

2 quick fixes available:

-  [Change visibility of 'nome' to 'public'](#)
-  [Create getter and setter for 'nome'...](#)

Press 'F2' for focus

Modificadores de Acesso - Exemplos

Package-private na Classe

```
1 package org.basic.java;  
2  
3 class Cafe {  
4  
5     protected String nome;  
6  
7 }  
8
```

```
1 package org.basic.cafe;  
2  
3 import org.basic.java.Cafe;  
4  
5 public class Cafe {  
6  
7     public static  
8         Cafe cafe  
9         cafe.nome  
10 }  
11  
12 }  
13
```

The type org.basic.java.Cafe is not visible

1 quick fix available:

- [Change visibility of 'Cafe' to 'public'](#)

Press 'F2' for focus

Modificadores de Acesso - Dicas

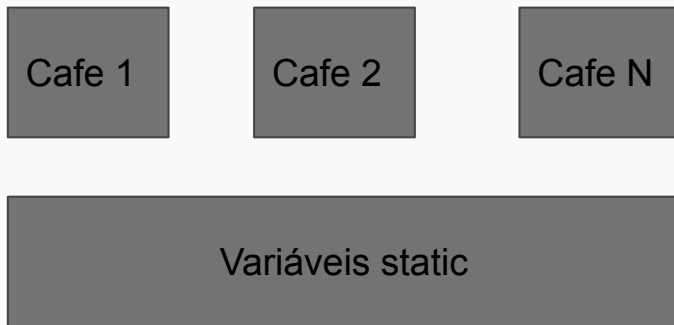
- Ao escolher um modificador de acesso baseado na situação específica em questão (não copiar e não tem receita de bolo, precisa entender a tabela)
- Evite ao máximo MEMBROS públicos
- Escolha um, ou seja, evite a visibilidade padrão. Para deixar explícito a sua ideia/pensamento ao criar o código

Modificadores de Acesso - Exercício

- Crie uma classe que represente uma conta corrente (pode ser o mais simples possível. dicas de campos: correntista, saldo, senha, etc)
- Defina modificadores de acessos para os campos, dos tipos: private, public e package
- Crie uma classe no mesmo pacote de teste/demo para verificar o que acontece quando se tenta acessar os vários campos
- Crie uma classe em outro pacote de teste/demo para verificar o que acontece quando se tenta acessar os vários campos

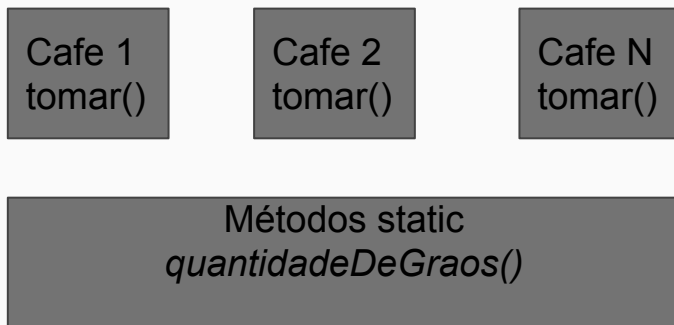
Variáveis e métodos static

- Uma variável static é compartilhada por todos os objetos de uma determinada classe. Variáveis marcadas com palavra static são chamadas de “membros estáticos” ou “variáveis de classe”.



Variáveis e métodos static

- Métodos marcados como static podem ser invocados no contexto da classe e não de um objeto específico. Também chamados de métodos de classe



Variáveis e métodos static - Regras

- Métodos static podem acessar outras variáveis e métodos static
- Métodos static NÃO podem acessar variáveis ou métodos de instância (não static)
- Métodos static NÃO podem fazer uso da palavra **this**

Variáveis e métodos static - Uso

- Variáveis static são boas para valores compartilhados entre objetos e constantes
- Métodos static servem para acessar variáveis static

Variáveis e métodos static - Exemplo

```
private static int numberOfBicycles = 0;

public Bicycle(int startCadence,
               int startSpeed,
               int startGear) {
    gear = startGear;
    cadence = startCadence;
    speed = startSpeed;

    id = ++numberOfBicycles;
}

public int getID() {
    return id;
}

public static int getNumberOfBicycles() {
    return numberOfBicycles;
}
```

Variáveis e métodos static - Exercício

- Acesse o link: <https://docs.oracle.com/javase/tutorial/java/javaOO/classvars.html>
- Execute o código exemplo, criando 2 ou mais bicicletas e informe o que aconteceu quando método `getNumberOfBicycles` é invocado

Construtores

- Para criar um objeto usamos a palavra chave **new** (instanciar um objeto)
- Para criar um objeto com atributos “obrigatórios”, usamos um construtor (instanciar um objeto já inicializado)

```
public class Point {  
    public int x = 0;  
    public int y = 0;  
    //constructor  
    public Point(int a, int b) {  
        x = a;  
        y = b;  
    }  
}
```

Construtores

- Essencialmente são métodos
- O nome do “método”/construtor é o nome da classe
- A assinatura do “método”/construtor são os atributos a serem inicializados
- Quando NÃO informamos um construtor, o compilador “coloca” um sem parâmetros automaticamente (construtor default).
- Construtores podem ser compostos
- É um método... vale tudo.

Construtores - Exercício

- Acesse o site: <https://docs.oracle.com/javase/tutorial/java/javaOO/constructors.html>
- Copie o construtor da classe “Bicicleta” e Tente criar um objeto dessa classe:
 - SEM informar um argumento qualquer
 - Informando mais argumentos que o exigido
 - Informando algum argumento nulo
 - É possível verificar que algum argumento é nulo?

Revisão

- Aprendemos a importância e os tipos de modificadores de acesso. Os mais importantes são: `private`, `protected` e `public`
- Vimos o que são variáveis e métodos `static`, bem como suas regras de utilização e usos comuns: compartilhar um mesmo dado entre objetos
- O compilador sempre disponibiliza um construtor default (sem argumentos) e criamos o nosso construtor quando queremos explicitar a necessidade de um determinado objeto já “nascer” com determinados valores