

Contextos da Aplicação

Profa. Karen Selbach Borges



Introdução



- Uma classe java possui dois contextos: um dinâmico e um estático.
- No contexto dinâmico estão as instâncias da classe (os objetos) e suas variáveis, que mudam de valor conforme a execução dos métodos, mudando, assim, o estado do objeto.
- No contexto estático está o código que não pertence a um objeto específico, mas a todas às instâncias de uma classe.



Variáveis de Instância



- Os atributos de um objeto também são chamados de variáveis de instância
- Cada objeto possui valores distintos para estas variáveis.

Execução 1



```
Nome Ana
Idade 23
Numero 1
```

```
public class Pessoa {
   String nome;
   int idade;
   int numero;

   public Pessoa (String nome, int idade){
      this.nome = nome;
      this.idade = idade;
      numero ++;
   }
```

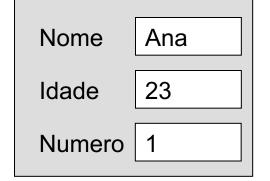
Variáveis de Instância



 Repare que a variável número, na segunda execução, continua como um.

public class Pessoa { String nome; int idade; int numero; public Pessoa (String nome, int idade) { this.nome = nome; this.idade = idade; numero ++; Execução 2

Execução 1



Nome

Idade

Numero

Luis

51



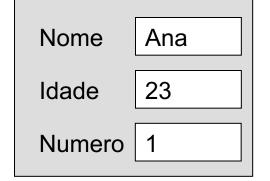
Variáveis de Instância



 O valor de número não muda porque cada instância de Pessoa possui sua própria variável número.

```
public class Pessoa {
 String nome;
  int idade;
  int numero;
 public Pessoa (String nome, int idade) {
     this.nome = nome;
    this.idade = idade;
     numero ++;
  Execução 2
                           Execução 3
```

Execução 1



Nome Luis Idade 51

Numero

Nome João Idade 35 Numero 1

Variáveis de Classe

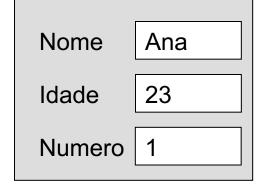


 São as variáveis definidas como static.

```
public class Pessoa {
   String nome;
   int idade;
   static int numero;

   public Pessoa (String nome, int idade) {
      this.nome = nome;
      this.idade = idade;
      numero ++;
   }
}
```







Variáveis de Classe



 A variável estática será a mesma para todas as instâncias.

```
public class Pessoa {
 String nome;
 int idade;
 static int numero;
 public Pessoa (String nome, int idade) {
    this.nome = nome;
    this.idade = idade;
    numero ++;
 Execução 2
```

Execução 1



Nome

Idade

Numero

Luis

51



Variáveis de Classe

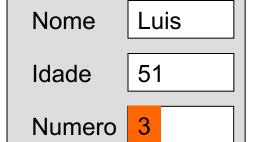


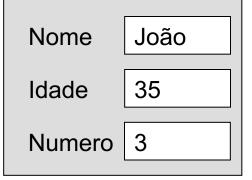
 Alterações no valor da variável estática irão refletir em todas as instâncias.

```
public class Pessoa {
 String nome;
 int idade;
 static int numero;
 public Pessoa ( String nome, int idade) {
    this.nome = nome;
    this.idade = idade;
    numero ++;
 Execução 2
                          Execução 3
```

Execução 1









Métodos Estáticos



- Os métodos declarados com static não precisam de objetos para serem executados.
- O método main é um método estático: public static void main (String[] args)
- Ele é executado pela JVM sem que seja necessário instanciar a classe em que foi declarado.

Métodos Estáticos



 Geralmente os métodos criados como estáticos são métodos utilitários

```
package util;
import javax.swing.*;
public class Console {
  public static String leDados(String mensagem){
    String s = JOptionPane.showInputDialog(null, mensagem);
    return s;
  public static void exibeDados(String mensagem, int tipoMensagem){
    JOptionPane.showMessageDialog(null, mensagem, tipoMensagem);
```

Métodos Estáticos



 Os métodos estáticos são invocados a partir do nome da própria classe onde foram declarados

```
import util.Console;
Import javax.swing.*;
public class Testa {
   public static void main (String[] args){
      String s = Console.input ("Digite alguma coisa");
      Console.output ("Você digitou "+ s, JOptionPane. INFORMATION_MESSAGE);
}
```



Foi declarado como static dentro de JOptionPane Está escrito com maiúsculas por que é uma constante

Constantes



- São variáveis estáticas que não podem sofrer alteração de valor.
- Devem ser declaradas como static final.

```
public class Matematica{
   public static final PI = 3.14;
}
```





- Um bloco de inicialização estático é executado quando a classe é carregada pela JVM
- Serve para inicializar alguma informação necessária antes mesmo de se ter acesso a uma instância da classe.





```
class Test {
  static int i;
  int j;
  static {
     i = 10;
     System.out.println("static block called ");
public class Main {
  public static void main(String args[]) {
        System.out.println(Test.i);
```

Saída: static block called 10



- O uso de inicializadores estáticos e de instância pode tornar seu código muito mais difícil de ler. Evite-os!
- Existe um caso comum para usar um inicializador estático: quando você precisa inicializar um campo estático e o código para isso requer mais de uma linha.





```
public class Main {
  static Integer[] dezenas = new Integer[10];
  static {
     for (int i = 0; i < 10; i + +){
        dezenas[i] = (i+1)*10;
  public static void main(String args[]) {
      for (int i = 0; i < 10; i + +){
        System.out.println(Main.dezenas[i]);
```

