

Sistema
FIRJAN




INFORMA, FORMA, TRANSFORMA.

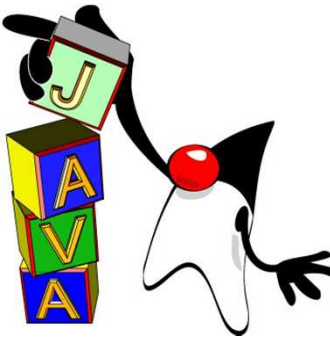
Programação Desktop

Fabício Curvello Gomes



Sistema
FIRJAN




INFORMA, FORMA, TRANSFORMA.




Estrutura de Código Java

INFORMA, FORMA, TRANSFORMA.



Tipos Primitivos



Tipo	Tamanho	Características
byte	8 bits	Numéricos sem casa decimal
short	16 bits	
int	32 bits	
long	64 bits	
float	32 bits	Numéricos com casa decimal
double	64 bits	
char	16 bits	Caracter da tabela unicode
boolean	JVM	true/false



3

INFORMA, FORMA, TRANSFORMA.


Tipos Inteiros

byte, short, int e long



A diferença entre eles está no intervalo de valores que cada um pode suportar:

Exemplos:

```
byte menor = 10; // 1 byte
short pequeno = 456; // 2 bytes
int normal = 10252; // 4 bytes
long muitoGrande = 6263732239L; // 8 bytes
```



4

INFORMA, FORMA, TRANSFORMA.


Tipos Ponto Flutuante

Float



Precisão simples (7 dígitos) que utiliza 32 bits de armazenamento. Tornam-se imprecisas para valores muito grandes ou muito pequenos. Úteis quando precisamos de um valor fracional sem grande necessidade de precisão.

Exemplo: Reais e Centavos.

```
float numeroReal = 10.9f; // 4 bytes
```



5

INFORMA, FORMA, TRANSFORMA.


Tipos Ponto Flutuante (Cont.)

double



Precisão dupla (15 dígitos) que utiliza 64 bits de armazenamento.

Exemplo:

```
double numero = 6745.9E13; // 8 bytes
```



6

INFORMA, FORMA, TRANSFORMA.


Tipo Textual

char - 16 bits - 2 bytes



Exemplos:

```
char meuCaracter = 'L';  
char meuCharUnicode = '\u0058';
```

A contrabarra indica uma sequência de escape.
Neste exemplo em específico, indica a utilização de um caractere da tabela Unicode (no caso X).




7



INFORMA, FORMA, TRANSFORMA.

Sequências de Escape

<code>\b</code>	<i>backspace</i>
<code>\t</code>	<i>tab</i>
<code>\f</code>	<i>form feed</i>
<code>\n</code>	<i>line feed</i>
<code>\r</code>	<i>carriage return</i>
<code>\'</code>	<i>aspas simples</i>
<code>\"</code>	<i>aspas duplas</i>
<code>\\</code>	<i>contrabarra</i>



8

INFORMA, FORMA, TRANSFORMA.


Tipo Lógico

boolean



Exemplos:

```
boolean status = true;  
boolean continuar = false;
```

Os literais do tipo boolean são escritos em letra minúscula.




9

INFORMA, FORMA, TRANSFORMA.



Exercício *05_TiposPrimitivos*

```
package controller;  
public class ExemploInteiro {  
    public static void main(String[] args){  
        int numero1, numero2, soma;  
  
        numero1 = 12;  
        numero2 = 3;  
        soma = numero1 + numero2;  
  
        System.out.println("Valor da Soma: " + soma);  
    }  
}
```

Classe
ExemploInteiro



10


  INFORMA, FORMA, TRANSFORMA.

Exercício *05_TiposPrimitivos*



```
package controller;
public class ExemploFlutuante {
    public static void main(String[] args) {
        double salario, aumento, novoSalario;

        salario = 2000.00;
        aumento = 0.15;
        novoSalario = salario + (aumento * salario);
        System.out.println("Novo Salário R$ " +
                           novoSalario);
    }
}
```

Classe
ExemploFlutuante



11


  INFORMA, FORMA, TRANSFORMA.

Exercício *05_TiposPrimitivos*

```
package controller;
public class ExemploEscape {
    public static void main(String[] args) {


        System.out.println("\t Utilizando TAB");
        System.out.println("\n\n2x Quebra de linha");
        System.out.println("\\ Contra-Barra");
        System.out.println("'Aspas Simples'");
        System.out.println("\"Aspas Duplas\"");
    }
}
```

Classe
ExemploEscape



12

Sistema
FIRJAN



INFORMA, FORMA, TRANSFORMA.

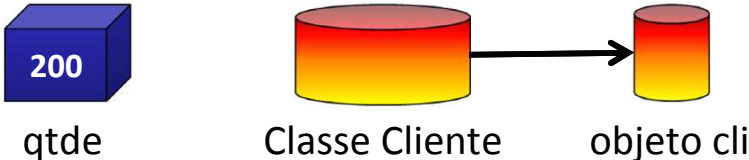
Valor e Referência


Em Java existem dois tipos de variáveis:

- Valor:** variáveis de tipos primitivos
- Referência:** variáveis de classes

Exemplos:


```
int qtde = 200;  
Cliente cli = new Cliente();
```





13

Sistema
FIRJAN




INFORMA, FORMA, TRANSFORMA.



Type Cast

Java não faz conversão implícita quando um tipo não “cabe” no outro.

A conversão deve ser explícita.



14




 INFORMA, FORMA, TRANSFORMA.


Type Cast (Cont.)



Exemplos:

```
long grande = 890L; // inicialmente c 64 bits
int pequeno = (int)(grande); // conversão explícita
char letra = (char) 87 // Letra 'W'
```

Sempre que possível é feita a conversão implícita.
Algumas conversões implícitas permitidas:




 15



 INFORMA, FORMA, TRANSFORMA.


Exercício *06_TypeCast* (Parte 1)



Exibir
Código
Inteiro

```
package controller;
public class ExemploTypeCast {
    public static void main(String[] args) {
        int a = 5, b = 2;
        int c;

        c = a / b;
        System.out.println("Valor de C: " + c);


        double d;
        d = a / b;
        System.out.println("Valor de D: " + d);
    }
}
```


 16



  **INFORMA. FORMA. TRANSFORMA.**

Exercício *06_TypeCast* (Parte 2) Exibir Código Inteiro

```
double e;  
//conversão explícita  
e = (double) a / b;  
System.out.println("Valor de E: " + e);  
  
float f = 14.5f;  
//conversão implícita.  
e = f;  
System.out.println("Valor de E: " + e);  
}  
}
```



17

  **INFORMA. FORMA. TRANSFORMA.**


Métodos

O comportamento invocável de objetos são os métodos.



Um método é algo que se pode pedir para um objeto de uma classe fazer.

Objetos da mesma classe tem os mesmos métodos.

Métodos são definidos ao nível de classe, enquanto que a invocação de uma operação é definida ao nível de objeto.




18



INFORMA, FORMA, TRANSFORMA.

Exemplos de Métodos


Classe Carro





→

Métodos:

- Cadastrar
- Consultar
- Alterar
- Excluir



19



INFORMA, FORMA, TRANSFORMA.


Exercício *07_Metodo*

Objetivos:

- Apresentar exemplo com métodos
- Estudar o comportamento das variáveis em relação aos métodos

Esta tarefa está descrita num documento específico com o seguinte nome:

[*PD - TI - 03.2 - Instruções Exercício 07 Metodo \(Escopo de Variável\).pdf*](#)



20

Exercício *07_Metodo* (Cont.)

```
package controller;
public class ChamadaMetodos {
    public static void main(String[] args) {
        System.out.println("Iniciando Programa");
        primeiro();
        System.out.println("Continuando Programa");
        terceiro();
        System.out.println("Terminando Programa");
    }
    public static void primeiro(){
        System.out.println("Iniciando método 1");
        segundo();
        System.out.println("Terminando método 1");
    }
    public static void segundo(){
        System.out.println("Iniciando método 2");
        System.out.println("Terminando método 2");
    }
    public static void terceiro(){
        System.out.println("Iniciando método 3");
        System.out.println("Terminando método 3");
    }
}
```

Ainda no projeto *07_Metodo*, crie a Classe *Metodos* dentro do pacote *controller*.

21



Sobrecarga de Métodos

São métodos com mesmo nome e assinaturas diferentes.

A assinatura é composta pelo nome do método com seus parâmetros.

Algumas vezes, precisamos que métodos por alguma questão de modelagem tenham o **mesmo nome**, mas implementem **comportamentos diferentes** de acordo com o argumento que é passado.


22





INFORMA, FORMA, TRANSFORMA.

Sobrecarga de Métodos (Cont.)

A sobrecarga pode ser de:

1. Quantidade de argumentos (quantidades diferentes para métodos diferentes).
2. Tipos de dados.
3. Retorno de valores diferentes.


23



INFORMA, FORMA, TRANSFORMA.

Exemplo de Sobrecarga de Métodos


1 – No Projeto
07_Metodo,
criar Classe
SobrecargaMetodo



```

package controller;

public class SobrecargaMetodo {
    int idade ;
    String nome;

    public static void main(String[] args) {
    }
}
    
```


24

INFORMA. FORMA. TRANSFORMA.

Continuação do Exemplo

```


public void cadastrarPessoa (int valor){
    idade = valor;
    System.out.println("Idade: "+idade);
}



public void cadastrarPessoa (String valor){
    nome = valor;
    System.out.println("Nome: " + nome);
}

public void cadastrarPessoa (int valor1, String valor2){
    idade = valor1;
    nome = valor2;
    System.out.println("Idade: " + idade + " - Nome: " +
    nome);
}

```

2 – Criar
Métodos em
Sobrecarga


25

INFORMA. FORMA. TRANSFORMA.

Final do Exemplo


```



public static void main(String[] args) {
    Sobre cargaMetodo scm = new Sobre cargaMetodo();
    scm.cadastrarPessoa(58);
    scm.cadastrarPessoa("Jorge Nogueira");
    scm.cadastrarPessoa(35, "Fabrício Curvello");
}

```

3 – Chamar os
métodos criados,
no método
construtor.

4 – Agora analise todo o código, execute o programa e tire suas conclusões sobre a resposta apresentada na tela.



26

INFORMA, FORMA, TRANSFORMA.



Método Construtor

É um método utilizado para inicializar objetos da classe quando estes são criados.

Este método possui o mesmo nome da Classe e não tem nenhum tipo de retorno, nem mesmo void.



27

INFORMA, FORMA, TRANSFORMA.

Palavra Reservada *this*

Refere-se a variável de classe sobre o qual o método foi chamado.


É utilizada quando o nome da variável de classe for igual ao nome de um argumento passado pelo método de instância.

Exemplo: Método Construtor



```
public ItemDePedido(int qtde, double subtotal){  
  
    super();  
    this.qtde = qtde;  
    this.subtotal = subtotal;  
}
```

Argumento passado pelo método

Variável de classe



28

INFORMA, FORMA, TRANSFORMA.


Projeto *InfoNote_02*

Objetivos:



- Implementar métodos construtores
- Implementar método mostrar

Esta tarefa está descrita num documento específico com o seguinte nome:

[*PD - TI - 03.3 - Instruções Projeto InfoNote_02.pdf*](#)





29

INFORMA, FORMA, TRANSFORMA.



Visibilidade de Atributos e Métodos

- **Métodos Públicos:**
São métodos que podem ser visíveis externamente, ou seja, outras classes poderão acessar estes métodos sem restrições.
- **Atributos de Classes de Negócio:**
Por convenção estes atributos sempre possuem visibilidade privada.








30



INFORMA, FORMA, TRANSFORMA.

Visibilidade de Atributos e Métodos (Cont.)

Modificadores	Mesma Classe	Mesmo Pacote	SubClasses	Qualquer Lugar
private	•			
<package>	•	•		
protected	•	•	•	
public	•	•	•	•


31



INFORMA, FORMA, TRANSFORMA.


Encapsulamento e Ocultamento

Encapsulamento:

- Manter dentro da própria classe seus métodos e propriedades.
- Facilita a manutenção.

Ocultamento:

- Modificar a visibilidade de atributos e métodos conforme tabela do slide anterior.


32

Método Set e Get

Método Set: Entrada de dados no atributo da classe.

Método Get: Retorno do dado atribuído pelo método Set.

33

Projeto *08_ExemploGetSet*

1 – Criar pacote model e classe Pessoa:

```
package model;

public class Pessoa {
    private String nome;
    private String sexo;
    private int idade;

    public void setNome(String nome){
        this.nome = nome;
    }

    public String getNome(){
        return nome;
    }

    public String getSexo() {
        return sexo;
    }

    public void setSexo(String sexo) {
        this.sexo = sexo;
    }

    public int getIdade() {
        return idade;
    }

    public void setIdade(int idade) {
        this.idade = idade;
    }
}
```

34

Projeto **08_ExemploGetSet (Cont.)**

2 – Criar pacote controller e classe Cadastro:

```
package controller;

import model.Pessoa;

public class Cadastro {

    public static void main(String[] args) {
        Pessoa pessoa = new Pessoa();

        pessoa.setNome("Leandro Ferra");
        pessoa.setIdade(28);
        pessoa.setSexo("Masculino");

        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("Idade: " + pessoa.getIdade());
        System.out.println("Sexo: " + pessoa.getSexo());
    }
}
```

35

Sistema
FIRJAN

INFORMA. FORMA. TRANSFORMA.


Projeto **InfoNote_03**

Descrição:

- 1 – Copiar e colar o projeto InfoNote_02, renomeando-o para InfoNote_03.
- 2 – Mudar todas as visibilidades de atributos contidos em todas as classes de negócio (model) de *public* para *private* e gerar get e set.




36





INFORMA, FORMA, TRANSFORMA.

Dúvidas?




37




INFORMA, FORMA, TRANSFORMA.


Bibliografia



Java Como Programar 8ª Edição
Paul Deitel e Harvey Deitel
Ed. Pearson



Java 7 Ensino Didático
Sérgio Furgeri
Ed. Érica



Fundamentos de Computação e Orientação a Objetos Usando Java
Francisco A. C. Pinheiro
Ed. LTC

38