



Prof. Dr. Fernando Almeida
proffernando.almeida@fiap.com.br

DDD (Domain Driven Design) Métodos em Java

O QUE VAMOS APRENDER HOJE?

1

Métodos (definição)

2

Métodos sem retorno

3

Métodos com retorno

4

Passagem de parâmetros

5

Variáveis locais e globais

6

Método printf()

Métodos em Java

Definição

Métodos

Conjunto de declarações
que são agrupadas para
executar uma operação.



```
System.out.println("Algoritmos")
```

Quando chamamos este comando,
diversas operações são realizadas
até que a mensagem é impressa na
tela



DIVERSAS OPERAÇÕES



Métodos em JAVA

- Também conhecidos como **funções**, **ações de execuções** ou **procedimentos**
- Responsáveis por ajudar no **design** do sistema (separação em blocos)
- Permite a **reutilização** e **organização** do código, podendo ser reaproveitados na construção de um novo sistema
- Facilita a **manutenção** do código
- O nome do método acrescido de seus parâmetros é denominado **assinatura do método**
- As classes que armazenam os métodos são conhecidas como **projetos de objetos** (atributos e métodos)
- Um método por ser **Integrado**, **importado** ou **definido**
- Exemplos: **print()**, **println()**, **next()**, **nextInt()**, **meuMetodo()**, **calculaMedia()**

Sintaxe de um método

nome da função lista de parâmetros

tipo de retorno

public
private
protected

<modificador>

```
int soma (int a, int b){
```

```
// corpo da função  
return (a + b);
```

```
}
```

retorno

Modificadores de Acesso

- São utilizados pelo Java para controlar o acesso aos **atributos e métodos**
- **public** - **menos restritivo**
 - Atributos e métodos definidos em uma classe (com **public**) podem ser acessados pelos métodos da própria classe, por classes derivadas desta e por qualquer outra classe em qualquer outro pacote
- **protected**
 - Atributos e métodos definidos em uma classe (com **protected**) são acessíveis pelos métodos da própria classe e pelas classes derivadas
- **private** - **mais restritivo**
 - Atributos e métodos definidos em uma classe (com **private**) só podem ser acessados pelos métodos da própria classe

Modificadores de Acesso

- Quando nenhum modificador é definido (tipo “**package**”), os atributos e métodos podem ser acessados pelos métodos da própria classe, pelas classes derivadas e por qualquer outra classe **dentro do mesmo pacote**
- **IMPORTANTE:** esse controle **não** se aplica às **variáveis locais** aos métodos

Métodos em Java

Métodos **sem** retorno

Métodos sem retorno em JAVA

- Executa apenas o código que tem dentro dele, não retornando nenhum resultado
- Identificado com a palavra-chave **void**

```
| <modificador> void imprime (String x) {  
|     System.out.println(x);  
| }  
+ . . . . .
```

Métodos sem retorno em Java

Exemplo

```
1 class Metodo2{  
2     public void escrever()  
3     {  
4         System.out.println("Método sem Retorno - VOID ");  
5     }  
6 }  
7  
8 public class Metodos_Sem_Retorno {  
9  
10    public static void main(String[] args) {  
11  
12        Metodo2 m = new Metodo2();  
13        m.escrever();  
14    }  
15 }
```

Métodos em Java

Métodos **com** retorno

Métodos com retorno em JAVA

- Executa o código que tem dentro dele e retorna algum resultado
- Identificado com o tipo de dado com o qual foi declarado - **int, byte, short, float, double, char, booleano, String...**

```
public int minimo (){  
  
    int min = 0;  
    int a, int b;  
  
    if (a < b){  
        min = a;  
    }  
    else {  
        min = b;  
    }  
    return min;  
}
```

Métodos com retorno em Java

Exemplo de método com retorno de uma String

```
1 class Metodo3{  
2  
3     String nome = "João Silva";  
4  
5     public String retornaNome()  
6     {  
7         return nome;  
8     }  
9 }  
10  
11 public class Metodo_Com_Retorno {  
12  
13     public static void main(String[] args) {  
14  
15         Metodo3 m3 = new Metodo3();  
16         System.out.println(m3.retornaNome());  
17  
18     }  
19 }  
20 }
```

Métodos com retorno em Java

Exemplo de método com retorno de um int (inteiro)

```
1  class ValoresInt
2  {
3      public int calculadora()
4      {
5          int a = 10;
6          int b = 20;
7          int c = a + b;
8
9          return c;
10     }
11 }
12
13 public class Metodo_Com_Retorno_Inteiro {
14
15     public static void main(String[] args) {
16
17         ValoresInt valores = new ValoresInt();
18
19         System.out.print(valores.calculadora());
20     }
21 }
```

Métodos em Java

Passagem de parâmetros (argumentos)

Métodos com parâmetros em JAVA

- São funções (métodos) que recebem **parâmetro(s)** para executar uma tarefa específica
- Podem receber diferentes **tipos** de dados

```
public int minimo (int a, int b){  
    int min = 0;  
    if (a < b){  
        min = a;  
    }  
    else {  
        min = b;  
    }  
    return min;  
}
```

3 x
 ↓
 2
 ↓
 4

```
public static void main (String [] args){  
    ...  
    int x = 11;  
    int y = 6;  
    int c = <objeto>.minimo(x, y);  
    System.out.println("Mínimo: " + c);  
}
```

Um método pode ter zero ou mais argumentos

Exemplo de uma classe em Java

Classe com atributo, métodos com e sem retorno, e passagem de parâmetros

```
1 public class Conta {  
2     private double saldo;  
3  
4     public void setSaldo( double saldo) {  
5         this.saldo = saldo;  
6     }  
7  
8     public double getSaldo() {  
9         return saldo;  
10    }  
11  
12    public void depositar( double valor){  
13        saldo += valor;  
14    }  
15  
16    public void verificaSaldo(){  
17        System.out.println("Valor do Saldo: "+getSaldo());  
18    }  
19}  
20 }
```

Vamos praticar!

Métodos com retorno em Java

Invocação dos métodos da Classe

```
1 public static void main(String[] args) {  
2  
3     //INSTANCIA A CLASSE  
4     Conta conta = new Conta();  
5  
6     //DEFINE UM VALOR DE SALDO  
7     conta.setSaldo(633.00);  
8  
9     //DEFINE VALOR PARA DEPOSITAR  
10    conta.depositar(555.0);  
11  
12    //RESGATA VALOR  
13    conta.verificaSaldo();  
14  
15 }
```

Adicione um método que permita a entrada de dados pelo teclado

Métodos em Java

Variáveis Locais

Variáveis Locais

- São variáveis definidas dentro de um determinado escopo (ou bloco)

Exemplo 1

```
public class VerificaAprov {  
  
    public void imprimirNota (double nota){  
        String status;  
        if (nota >= 6){  
            status = "Aprovado";  
        }  
        else {  
            status = "Reprovado";  
        }  
        System.out.println(status);  
    }  
  
    public static void main (String [] args){  
        VerificaAprov va = new VerificaAprov();  
        va.imprimirNota(6.7);  
    }  
}
```

Exemplo 2

```
public class VerificaAprov {  
    public void imprimirNota (double nota){  
        String status;  
        if (nota >= 6){  
            status = "Aprovado";  
        }  
        else {  
            status = "Reprovado";  
        }  
    }  
  
    public static void main (String [] args){  
        VerificaAprov va = new VerificaAprov();  
        va.imprimirNota(6.7);  
        System.out.println(status);  
    }  
}
```

erro! variável local em imprimirNota()

Métodos em Java

Variáveis Globais

Variáveis Globais

- São variáveis definidas dentro da Classe (visto em todo o código)

```
public class VerificaAprov {  
    String status;  
    public void imprimirNota (double nota){  
        if (nota >= 6){  
            status = "Aprovado";  
        }  
        else {  
            status = "Reprovado";  
        }  
    }  
  
    public static void main (String [] args){  
        VerificaAprov va = new VerificaAprov();  
        va.imprimirNota(6.7);  
        System.out.println(status);  
    }  
}
```



Métodos em Java

método `print()`, `println()` e `printf()`

Métodos print(), println() e printf()

- Métodos muito utilizados para exibir uma mensagem (ou resultado) do processamento no console

System.out.



```
print();  
println();  
printf();
```

Three method names are listed vertically. Blue arrows point from the identifier `System.out.` to each of these three methods.

Métodos

O objeto **System.out** representa a saída padrão, permitindo exibir dados no console quando executamos uma aplicação em Java.

Métodos print(), println() e printf()

- O método **printf** permite apresentar os dados de saída formatados
- Utiliza especificados de formato (código que indica onde e que tipo de dado será mostrado)
- Inicia com o símbolo % seguido do tipo de dado que será mostrado

```
System.out.printf("Olá %s, sua idade é %d \n", nome, idade);
```

String

inteiro

String

inteiro

(ver tabela com os específicos)

• • • • • +

- Especificadores de formato

Especificador	Formato
%s	String de caracteres
%d	Número inteiro decimal
%u	Número inteiro decimal sem sinal
%o	Número inteiro octal sem sinal
%x, %X	Número inteiro hexadecimal sem sinal, minúsculo ou maiúsculo
%f	Float
%2f	Double
%e, %E	Número real, em notação científica, minúsculo ou maiúsculo
%b	Boolean
%c	Caractere (char)

Caracteres de escape

Caractere	Representa
\t	Tabulação
\b	Backspace
\n	Nova Linha
\r	Retorno de carro
\'	Aspa simples
\"	Aspa dupla
\\\	Barra invertida

Específico para o printf:

Caractere	Representa
%%	Símbolo de porcentagem

+



Exercício - método printf()

1) Escreva um programa que mostre na tela seu nome, seu endereço e seu email, como segue:

- 1) <seu nome>
- 2) <endereço>
- 3) <cep>
- 4) <cidade>
- 5) <enter>
- 6) <enter>
- 7) <enter>
- 8) <email>

Exercício método printf()

- 1) Resolução sugerida

```
System.out.printf("%s\n", "Domain Driven Design");
System.out.printf("%s %d\n", "FIAP - Av. Paulista, ", 1106);
System.out.println("CEP 01311-000");
System.out.printf("%s - %s", "São Paulo", "SP");
System.out.println("\n");
System.out.print("ddd@fiap.com.br");
```

Exercícios - Métodos

- 1) Modifique o método imprimirNota() do exemplo de aula para retornar uma String. Se o aluno aprovado, o método retorna “Aprovado”, caso contrário, retornar “Reprovado”.
- 2) Crie um método com dois parâmetros numéricos capaz de verificar o maior entre eles e retorná-lo.
- 3) Crie um método capaz de verificar se um número é PAR ou ÍMPAR (sem retorno)

Métodos em Java

Desafio

Calculadora Simples

- Utilizando o conceitos de métodos, crie uma Calculadora Simples contendo as 4 operações básicas: adição, subtração, multiplicação e divisão.

- 1) Cada função (matemática) deve receber apenas dois parâmetros e retornar o resultado da operação
- 2) Adicione um método Menu para mostrar as opções ao usuário
- 3) Adicione um método para imprimir uma mensagem com uma mensagem e o resultado da operação escolhida
- 4) Adicione um método para permitir a entrada de dados pelo teclado
- 5) Extra:
 - 1) Adicione um método para calcular a raiz quadrada de um número
 - 2) Adicione um método para calcular a potência de um número por outro

OBRIGADO

FIAP

Copyright © 2021 | Prof. Dr. Fernando Luiz de Almeida

Todos os direitos reservados. Reprodução ou divulgação total ou parcial deste documento, é expressamente
proibido sem consentimento formal, por escrito, do professor/autor.



Até a próxima aula