Índice

[Estrutura de dados 1](#_butpk9qu101k)

[Métodos 1](#_o2kx7f9ofetl)

[Funções e métodos 2](#_rgl3zz40g9gx)

[Comandos 3](#_stud4j8xcw5o)

[Pacotes 3](#_qusu5yomam12)

[Pacotes API Java 4](#_lb707fjjmj55)

[Projeto guia (banco) 4](#_sfr4h7270yds)

[Comandos inusitados 4](#_sf5lzqrjbpyv)

[Links e afins 4](#_v49ndfn8hwjl)

[Dúvidas 5](#_huow9eiz5of6)

# 

**Assuntos: JAVA Estrutura de dados, JAVA métodos.**

**Abertura Alunos:**

Reflexão diária: Relacionamento Interpessoal.

Marina Pereira e Lauane Gonçalves.

# [Estrutura de dados](https://docs.google.com/document/u/0/d/1TDc-PhSJj_uCvjUdQUvQEvacjqszZzkN_YgW2ALQoqk/edit)

(Ao clicar no título você será direcionado as anotações anteriores de estrutura de dados - queue e stack)

# Métodos

Para maiores informações, acesse o Cookbook MD09 na aba de links.

Um método é uma função associada à Classe, ou seja uma ação.

Os Métodos permitem que a pessoa desenvolvedora modularize (divida) os programas, separando suas tarefas em unidades menores autocontidas.

Outra grande vantagem do uso de Métodos é que eles são reutilizáveis em futuros programas, evitando a repetição desnecessária de código.

Para promover a capacidade de reutilização de software, todos os Métodos devem estar limitados à realização de uma única tarefa bem definida.

O nome do método também deve ser assertivo e expressar essa tarefa efetivamente.

Os métodos tornam mais fácil as tarefas de escrever, depurar, manter e modificar programas, pelo simples fato de um método que realiza apenas uma tarefa é mais fácil de testar e depurar do que um método maior que realiza muitas tarefas.

Os métodos são essencialmente procedimentos que podem manipular atributos de objetos para os quais o método foi definido e receber parâmetros por valor através da lista de argumentos presentes na sua assinatura.

#### Funções e métodos

Função é um grupo de código reutilizável, que pode ser invocado em qualquer lugar do seu programa.

Ela pode receber 0 ou mais argumentos que correspondam a diferentes dados de entrada (input).

Esses dados podem ser processados ou utilizados dentro do código de uma função.

Uma função nula é um tipo especial que não retorna um valor.

Funções de classes da API Java também são chamadas de métodos, e podem ser invocadas diretamente quando a classe é instanciada, ou se os métodos são estáticos, diretamente pelo nome da classe.

**Exemplo:**

**static void *meuMetodo* (String *palavra*) {**

**system.out.println(*palavra* + *“ 71”*);**

**}**

**public static void main (String [] args) {**

**meuMetodo(*“Generation”*);**

**}**

Todas as informações em itálico são informações de nomes substituíveis.

Primeiro, nós criamos uma função que vai guardar uma linha de código.

Ela vai funcionar quase que como uma variável, ou seja, ela pode ser invocada a qualquer momento, e quantas vezes quiser, sem precisar repetir todas as linhas de código para cada vez que quiser usá-la.

Então, para códigos longos, que precisam ser repetidos em diversas partes, você pode só citar a função, e colocar os dados a serem substituídos nela.

É como quando usamos o java.útil para pegar a função Scanner ou semelhantes.

**No exemplo acima, vemos que:**

1. Se cria uma função.
2. Essa função se chama ***meuMetodo***, é por esse nome que podemos invocá-la.
3. Ela tem dentro de si uma variável **String** chamada ***palavra***, essa variável é uma parte de entrada do código quando a função for invocada.
4. Dentro da função ***meuMetodo*** criamos um sysout que vai colocar na tela o que quer que se coloque na variável ***palavra*** + a frase “ 71”
5. Depois, prosseguimos o código com a função comum de sempre, public static…
6. Dentro dessa função, invocamos ***meuMetodo***, e nele colocamos a informação “Generation”, nesse caso, ele irá dar entrada com essa frase no lugar de ***palavra***.
7. No console, seríamos capazes de ver: Generation 71

## Comandos

**static void *nomeMétodo* (*tipo* *nome*) { }**

Comando para definir um método

**[modificador de acesso] [modificados non-access] tipo nome (argumentos) {**

**corpo do método**

**}**

Sintaxe método

**void** É um método sem retorno. Pode ser usado caso o Método não necessite ter um tipo de dado de retorno, ou seja, um valor que deverá ser retornado ao final da execução do Método. Neste caso NÃO se usa return.

**static** Usado quando se cria um método na classe main. Todos os métodos nesta classe devem ser do tipo static.

Um Método **static** é um Método da Classe, ou seja, são Métodos que não dependem de um Objeto. Quando eles são invocados, executam uma função sem a dependência de um objeto ou instância de uma classe, conseguindo chamar diretamente qualquer Método da classe.

**return**  Quando um Método recebe os dados, e deve retornar o resultado de algum comando feito com os dados. Neste caso NÃO se usa void.

## Pacotes

Conjunto de classes. Pacotes organizam e guardam uma série de classes em si, como uma espécie de “pasta”.

### Pacotes API Java

| **Pacote API** | **Suporta** |
| --- | --- |
| **java.awt** | Recursos gráficos: botão, barra, caixas de texto, janela, etc. |
| **javax.swing** | Complemento ao pacote awt. |
| **java.io** | input e output (entrada e saída)  de dados. |
| **java.lang** | Recursos de linguagem. |
| **java.math** | Operações matemáticas. |
| **java.text** | Recursos de texto. |
| **java.util** | Miscelânea de recursos utilitários. |

# Projeto guia (banco)

## Comandos inusitados

**System.exit(*valor*)** Fecha o sistema.

**while (true)**  Loop infinito

# 

# Links e afins

**Cookbook MD18, Estrutura de Dados, Queue e Stack:**

<https://github.com/conteudoGeneration/cookbook_java_fullstack/blob/main/01_java/18.md>

**Cookbook MD09, Métodos:**

<https://github.com/conteudoGeneration/cookbook_java_fullstack/blob/main/01_java/09.md>

**Cookbook PR01, Projeto Conta bancária, Criação MENU:**

<https://github.com/conteudoGeneration/cookbook_java_fullstack/blob/main/01_java/pr01.md>

**Cookbook PR02, Projeto Conta bancária, Colorindo o MENU:**

<https://github.com/conteudoGeneration/cookbook_java_fullstack/blob/main/01_java/pr02.md>

# 

# Dúvidas