Criação

Conceituação de métodos

"É uma porção de código (sub-rotina) que é disponibilizada por uma classe. Este é executado quando é feita uma requisição a ele. São responsáveis por definir e realizar um determinado comportamento. "

Criação

Padrão de definição:

<?visibilidade?> <?tipo?> <?modificador?> retorno nome (<?parâmetros?>) <?exceções?> corpo

Onde:

- **Visibilidade:** "public", "protected" ou "private"
- **Tipo:** concreto ou abstrato
- **Modificador:** "static" ou "final"
- **R:** tipo de dado ou "void"
- **Nome:** nome que é fornecido ao método
- **Parâmetros**: parâmetros que pode receber
- **Exceções:** exceções que pode lançar
- **Corpo:** código que possui ou vazio

Exemplos:

public String getNome() {…}

public double calcularTotalNota() {…}

public int verificarDistancia(int cordenda1, int cordenada2) {…}

public abstract void executar() ;

public void alterarFabricante(Fabricante fabricante) {…}

public Relatorio gerarDadosAnaliticos(Cliente cliente, List <Compras> compras) {…}

public static R N(P) {…}

Utilização

Passa-se uma mensagem através de uma classe ou objeto.

nome_da_classe.nome_do_metodo(); ou nome_da_classe.nome_do_metodo(…);

nome_do_objeto.nome_do_metodo(); ou nome_do_objeto.nome_do_metodo(…);

Math.random(); ou Math.sqrt(4);

usuario.getEmail(); ou usuario.alterarEndereco(endereco);

Conceitos Inerentes aos métodos e boas práticas na sua criação

Particularidades

- **Assinatura:** é a forma de identificar unicamente o método

  Ass = nome + parâmetros

  Método:

```
public double calcularTotalVenda(double
precoItem1, double precoItem2, double precoItem3)
{...}
```

  Assinatura:

```
calcularTotalVenda(double precoItem1,
double precoItem2, double precoItem3)
```

- **Construtor e Destrutor:** são métodos especiais usados na Orientação a Objetos.
- **Mensagem:** é o ato de solicitar ao método que o mesmo execute. Esta pode ser direcionada a um objeto ou a uma classe.
- **Passagem de parâmetros:**
  - Por valor (cópia)

```
int i = 10;
public void fazerAlgo(int i) {

   i = i + 10;
   System.out.println("Valor de i dentro: " + i);
}
System.out.println("Valor de i fora: " + i);
```

  - Por referência (endereço)

Boas Práticas

- Nomes devem ser descritivos, mas curtos
- Notação camelo
  - verificarSaldo (); executarTransferenncia (…); existeDebito ();
- Deve possuir entre 80 e 120 linhas
- Evite lista de parâmetros longas
- Visibilidade adequadas