Orientação a Objetos em Java

Classe e Objetos

- Objetos (ou instância) - **''O bolo''**
- Classe - **''A receita''**
- Variável de Instância: **Atributos do objeto**

Aplicando Orientação a Objetos Básica

Situação

''*Você foi contratado como programador júnior e, em seu primeiro trabalho, seu gerente pediu para você criar um programa em Java para gerenciar dados de funcionários. Como você é iniciante, ele solicitou que você fizesse um programa simplificado que imprima o nome, cargo e idade dos funcionários.*''

Construtor

```
class Funcionario {
    String nome;
    String cargo;
    int idade;

    // Este é o construtor
    Funcionario(String nome, String cargo, int idade) {

        this.nome = nome;
        this.cargo = cargo;
        this.idade = idade;

    }

    public static void main(String[] args) {

        Funcionario funcionario = new Funcionario("João",
"Desenvolvedor", 30);

        System.out.println("Nome: " + funcionario.nome);
        System.out.println("Cargo: " + funcionario.cargo);
        System.out.println("Idade: " + funcionario.idade);
    }
}
```

Pacotes java

Pontos importantes:

1. **Organização**
2. **Prevenção de conflitos de nomes**
3. **Reutilização de código**
4. **Controle de acesso**

Exemplo de como criar um pacote em Java:

```
package empresa.dados; // Declaração do pacote



public class Funcionarios {

    // Seu código aqui

}
```

Modificadores de acesso

1. **public**
2. **private**
3. **protected**
4. **default** (*sem modificador*)

Exemplo:

```
package com.empresa;



public class Funcionarios {

    private String nome; // Só pode ser acessado dentro da classe
Funcionarios



    // Seu código aqui

}
```

```java
package com.empresa.folhadepagamento;


import com.empresa.Funcionarios;


public class FolhaDePagamento {

    public static void main(String[] args) {

        Funcionarios funcionario = new Funcionarios();

        // Não podemos acessar funcionario.nome aqui porque é private

    }

}
```

Métodos em Java

Declaração de Métodos

A declaração de um método inclui o modificador de acesso, o tipo de retorno, o nome do método e os parâmetros.

Exemplo:

```
public String obterInfo() {
    return "Nome: " + nome + ", Salario: " + salario;
}
```

Neste exemplo:

- `public` é o modificador de acesso.
- `String` é o tipo de retorno.
- `obterInfo` é o nome do método.
- Não há parâmetros.

Este método retorna uma string que contém o nome e o salário do funcionário.

Métodos em Java

Passagem por Valor

Exemplo:

```
public void aumentarSalario(double aumento) {
    salario += aumento;
    //aumento += 1000;
}
```

Neste método:

- `public` é o modificador de acesso.
- `void` é o tipo de retorno, void não retorna nenhum valor
- `aumentarSalario` é o nome do método.
- `double aumento` é o parâmetro.

Este método aumenta o salário do funcionário pelo valor fornecido.

Passagem por Referência

Exemplo:

```
// Método para alterar o salário
public void alterarQualquerSalario(Funcionario func, double
novoSalario){
    func.salario = novoSalario;
}
```

Neste método:

- `public` é o modificador de acesso.
- `void` é o tipo de retorno.
- `alterarQualquerSalario` é o nome do método.
- `Funcionario func, double novoSalario` são os parâmetros.

Este método altera o salário do funcionário especificado (`func`) para o novo salário fornecido (`novoSalario`).

Sobrecarga de Métodos em Java

```
class NomeDaClasse {

    tipoRetorno nomeMetodo(tipo1 param1, tipo2 param2, ...) {

        // corpo do método

    }


    tipoRetorno nomeMetodo(tipoA paramA, tipoB paramB, ...) {

        // corpo do método

    }

    // e assim por diante...
}
```

Exemplo

```
    public void aumentarSalario(int porcentagem) {
        salario += salario * porcentagem / 100.0;
    }
```

Métodos Estáticos em Java

- Pertencem à classe
- Palavra-chave `static`.

Declaração de Métodos Estáticos

A declaração de um método estático é semelhante à de um método normal, mas inclui a palavra-chave `static`.

Exemplo:

```
public class TestStatic {
    // Método estático sobrecarregado #1
    public static void exibirValor(int a) {
        System.out.println("O valor do inteiro é: " + a);
    }

    // Método estático sobrecarregado #2
    public static void exibirValor(String str) {
        System.out.println("A string é: " + str);
    }
}
```

Neste exemplo, `exibirValor` é um método estático. Ele pode ser chamado diretamente sem a necessidade de criar uma instância da classe `TestStatic`.

Membros Estáticos em Java

1. **Pertencem à classe, não ao objeto**
2. **Podem ser acessados pelo nome da classe**
3. **Compartilham o mesmo membro estático**

```
class NomeDaClasse {

    static tipo nomeDoMembro;

}
```