

TI-220 Java Orientado a Objetos

ANTONIO CARVALHO - TREINAMENTOS

A solid blue horizontal bar spanning the width of the slide, located at the bottom.

Polimorfismo

Sobrecarga (Overload)

A sobrecarga permite a criação de métodos com o mesmo nome porém com parâmetros diferentes, de modo que serão invocados conforme o tipo e a quantidade de parâmetros informados.

Sobrecarga (Overload)

Exemplo :

```
int somaNumeros( int numero1, int numero2 ) {  
    System.out.println("Método 1");  
    int soma = numero1 + numero2;  
    return (soma)  
}
```

```
int somaNumeros( int numero1, int numero2, int numero3 ) {  
    System.out.println("Método 2");  
    int soma = numero1 + numero2 + numero3;  
    return (soma)  
}
```


Sobrecarga (Overload)

Ao ser invocado o método somaNumeros com dois números inteiros, então será invocado o **método 1**, caso a quantidade de parâmetros seja 3, então o método invocado será o **método 2**

Sobrescrita (Overriding)

A sobrescrita permite que uma subclasse modifique o comportamento herdado de uma super classe.

Dessa forma uma subclasse (classe mais específica) pode conter um comportamento diferenciado daquele que foi herdado.

A solid blue horizontal bar spanning the width of the slide at the bottom.

Sobrescrita (Overriding)

Exemplo de sobrescrita:

```
public class Pessoa {  
    public void comer() {  
        System.out.println("Pessoa comendo");  
    }  
    public void andar( int qtdPassos) {  
        System.out.println("Pessoa andando " + qtdPassos + " passos");  
    }  
}
```

```
public class Atleta extends Pessoa {  
    public void comer() {  
        System.out.println("Atleta comendo");  
    }  
    public void andar( int qtdPassos ) {  
        System.out.println("Atleta andando " + qtdPassos +  
            " passos rapidamente");  
    }  
}
```

Sobrescrita (Overriding)

No exemplo anterior a classe **Atleta** herda da classe **Pessoa** e modifica os comportamentos **comer** e **andar**, de forma que o Atleta execute estes comportamentos de maneira diferente.

O comportamento é submetido ao polimorfismo e será acessado conforme o tipo da instância.

```
public class Campo {  
    public static void main(String[] args) {  
        Pessoa p1 = new Pessoa();  
        p1.comer(); // Mostrará Pessoa comendo  
        Atleta a = new Atleta();  
        a.comer(); // Mostrará Atleta comendo  
        Pessoa p2 = new Atleta();  
        p2.comer(); // Mostrará Atleta comendo  
        // Porque o método comer é invocado conforme a instância  
    }  
}
```


Sobrescrita (Overriding)

A sobrescrita tem algumas regras são elas:

- O tipo de retorno pode ser o mesmo tipo ou um subtipo dele
- Não pode haver redução de visibilidade
- Não pode lançar mais ou menos exceptions que o método que está sendo sobrescrito, como exemplo um método que lança **NullPointerException** não pode ser sobrescrito e lançar outra exception como **IOException**.
 - Não é porque o método está sendo sobrescrito que ele apresentará menos riscos à excessões.
- Métodos marcados com **final** ou **static** não podem ser sobrescritos
- Um método somente poderá ser sobrescrito, caso a subclasse consiga acessá-lo.

Sobrescrita (Overriding)

Exemplo das regras aplicadas

```
public class Pessoa {  
    public void comer() {  
        System.out.println("Pessoa comendo");  
    }  
    public void andar( int qtdPassos) {  
        System.out.println("Pessoa andando " + qtdPassos + " passos");  
    }  
    private void pensar() {  
        System.out.println("Pessoa pensando");  
    }  
}
```

```
public class Atleta extends Pessoa {  
    private void comer() { // Não compila pois reduz a visibilidade  
        System.out.println("Atleta comendo");  
    }  
    public void andar( int qtdPassos ) throws IOException {  
        // Não compila pois aumenta a quantidade de exceptions  
        System.out.println("Atleta andando " + qtdPassos +  
            " passos rapidamente");  
    }  
    private void pensar() {  
        // Não compila pois o método pensar não é visível nesta classe  
        System.out.println("Atleta comendo");  
    }  
}
```

Dúvidas



Bibliografia

BARNES, DAVID J. Programação Orientada a Objetos com Java

DEITEL, Java - Como Programar - 6ª Edição, Pearson Education

BEZERRA, EDUARDO Princípio de Análise e Projeto de Sistemas com UML, Campus

SIERRA, KATHY e BATES BERT, Use a Cabeça Java, Alta Books

SIERRA, KATHY e BATES BERT, OCA/OCP Java SE 7 Programmer I & II Study Guide

