Aula 4

Programação Orientada a Objetos

Prof. Leonardo Gomes

1

Conversa Inicial

2

Programação orientada a objetos

1. **Herança**
2. **Herança na linguagem Java**
3. **Construtores e herança**
4. **Palavras reservadas *super* e *this***
5. **Herança e UML**

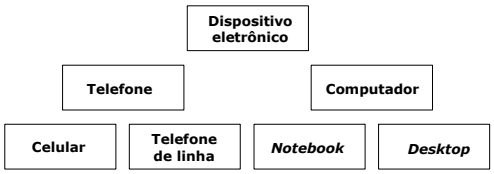
3

Herança

4

- **Útil para evitar replicação de códigos**
- **Classes semelhantes**
- **Uma classe baseada em outra**

5



6

Herança – nomes

Classe herdada	Classe herdeira
Classe mãe	Classe filha
Superclasse	subclasse
Classe base	Classe específica
Classe original	Classe derivada

7

Herança na linguagem Java

8

Herança Java

```
package uninter.com;
public class Livro {

   public String autor;
   public float custoProducao;
   public float precoVenda;
   public String titulo;
   public int paginas;

   //Cálculo do lucro
   public float lucro(){
         return (precoVenda - custoProducao);
   }
   //Cálculo do imposto de 20%
   public float imposto(){
         return (0.2f*this.lucro());
    }
   //Título do livro
   public void imprimirTitulo(){
         System.out.print("O titulo : "+ titulo);
   }
}
```

9

```
package uninter.com;
public class LivroDigital {
   public String autor;
   public float custoProducao;
   public float precoVenda;
   public String titulo;
   public int paginas;
   public String linkDownload;
   public int tamanhoMB;

   //Calculo do lucro
   public float lucro(){
         return (precoVenda - custoProducao);
   }

   //Tamanho do arquivo por página
   public float tamanhoPorPagina(){
         return ((float)tamanhoMB/(float)paginas);
   }

   //Título do livro
   public void imprimirTitulo(){
         System.out.print("O titulo : "+ titulo);
   }

   //Calculo do imposto de 20% + R$ 2 por livro
   public float imposto(){
         return (0.2f*this.lucro() + 2);
   }
}
```

10

```
package uninter.com;
public class LivroDigital extends Livro {
   public String linkDownload;
   public int tamanhoMB;

   //Tamanho do arquivo por página
   public float tamanhoPorPagina(){
         return ((float)tamanhoMB/(float)paginas);
   }
   //Calculo do imposto de 20% + R$ 2 por livro
   public float imposto(){
         return (0.2f*this.lucro() + 2);
   }
}
```

11

Construtores e herança

12

- **Construtores semelhantes aos métodos**
- **Sem retorno**
- **Só é chamado no momento da instanciação**
- **Classes filhas não herdam construtores, mas podem invocá-los**

13

Construtor herança

14

```
01.  public Livro(String autor, float custoProducao, float
   precoVenda, String titulo, int paginas) {
02.      this.autor = autor;
03.      this.custoProducao = custoProducao;
04.      this.precoVenda = precoVenda;
05.      this.titulo = titulo;
06.      this.paginas = paginas;
07.  }


08.  public LivroDigital(String autor, float
    custoProducao, float precoVenda, String titulo, int
    paginas, String linkDownload, int tamanhoMB) {

09.     super(autor,custoProducao,precoVenda,titulo,paginas);
10.     this.linkDownload = linkDownload;
11.     this.tamanhoMB = tamanhoMB;
12.  }
```

15

Palavras reservadas *Super* e *This*

16

Super

- **Palavra reservada**
- **Faz referência explícita à superclasse**
- **Invoca o construtor da superclasse**

17

This

- **Palavra reservada**
- **Faz referência explícita ao próprio objeto/à própria classe**
- **Utilizado para resolver ambiguidades de nome**

18

Instanceof

- **Compara tipagem de instâncias**
- **Classes filhas herdam tipagem das classes mães**

19

```
01.  class Animal {}
02.  class Mamifero extends Animal {}
03.  class Reptil extends Animal {}
04.  public class Cachorro extends Mamifero {
05.
06.     public static void main(String args[]) {
07.        Animal a = new Animal();
08.        Mamifero m = new Mamifero();
09.        Cachorro c = new Cachorro();
10.
11.        System.out.println(m instanceof Animal);
12.        System.out.println(c instanceof Mamifero);
13.        System.out.println(c instanceof Animal);
14.     }
15.  }
```

20

Herança e UML

21

Relações

- **Associação**
- **Agregação**
- **Composição**
- **Herança**
- **Dependência**

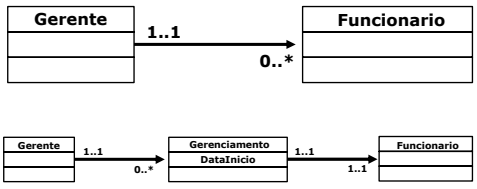
22

Multiplicidade

Código	Descrição
0..1	Zero ou um objeto. Não é obrigatório e tem no máximo um único objeto da classe na relação
1..1	Apenas um objeto, nunca mais ou menos
0..*	Muitos objetos. De zero até um número qualquer de objetos na relação
1..*	Pelo menos um. Podem existir mais objetos na relação, mas ao menos um
2..4	Na presença de valores específicos, a relação está limitada aos valores apresentados

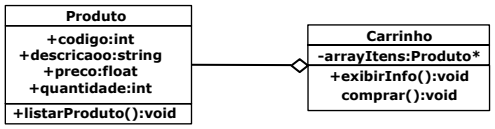
23

Associação



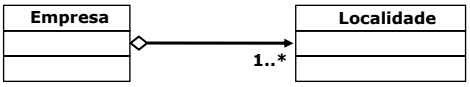
24

Agregação



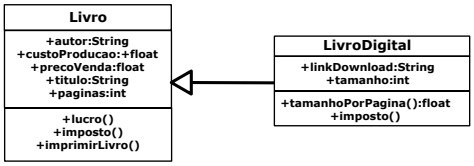
25

Composição



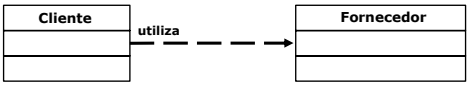
26

Herança

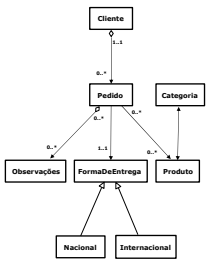


27

Dependência



28



29