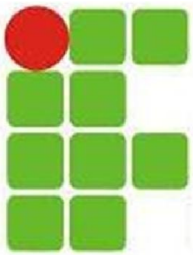
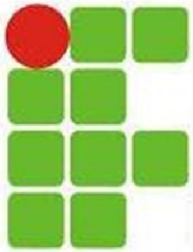
Informática

Professor: Diego Oliveira

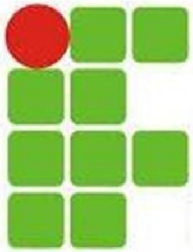


Conteúdo 04:
Orientação a Objetos



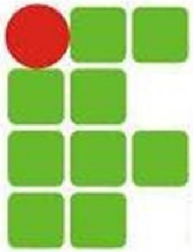
Conteúdo da Aula

- Introdução à Programação Orientada a Objetos
- Linguagem Java
- Classes
- Objetos
- Atributos
- Métodos e Construtores
- Parâmetros
- Visibilidade e Encapsulamento
- Herança e Polimorfismo
- Interfaces



O que é Programação Orientada a Objetos?

- É um paradigma de programação
- Baseia-se em objetos
- É um dos paradigmas mais utilizados
- Possui diversas linguagens que o usam:
  - Java
  - C++
  - Object Pascal
  - Python
  - VB.NET

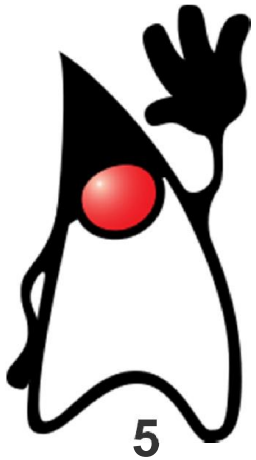


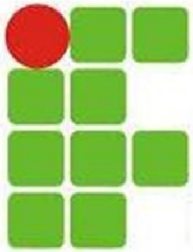
Orientação a Objetos

- A Orientação a Objetos se baseia em alguns princípios:
  - Abstração
  - Encapsulamento
  - Composição
  - Herança
  - Polimorfismo

Linguagem Java

- Linguagem de Programação mais utilizada
- Orientada a Objetos
- Multiplataforma
- Possui Várias Versões
  - **Java Card**
  - **Java ME**
  - **Java SE**
  - **Java EE**
  - **Java TV**
  - **Java FX**

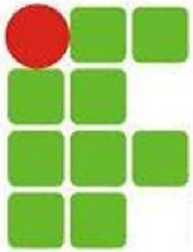




Classe

- A Classe é o molde, a planta, o esquema, o modelo a ser seguido pelos objetos
- A planta da casa é o modelo que as casas construídas terão
- Porém não é possível morar na planta da casa, apenas na casa já construída
- A Classe define as características da casa e as funções que ela terá: parte elétrica, hidráulica, saneamento e etc.

Classe



Objeto

- Objetos são utilizados para representar conceitos do mundo real
- Objetos seguem fielmente as especificações de suas Classes
- Os Objetos são instâncias concretas das Classes
- As casas são instâncias concretas das plantas que lhes deram origem

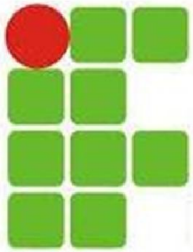
Objeto

Atributos Existentes na Classe Pessoa

- Definem o estado de uma classe
- Pessoa tem:
  - Nome (texto)
  - Idade (inteiro)
  - Peso (real)
  - Altura (real)
  - Profissão (texto)

Valores dos Atributos no Objeto

- Objeto Diego:
  - Nome: Diego Oliveira
  - Idade: 27
  - Peso: 70.0
  - Altura: 1.70
  - Profissão: Professor



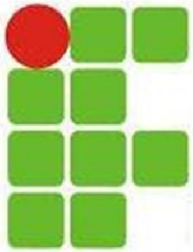
Atributos da Classe Pessoa e Objeto 'Diego' em Java

Pessoa.java | TestePessoa.java

```
1  public class Pessoa {
2      String nome;
3      int idade;
4      double peso;
5      double altura;
6      String profissao;
7  }
```

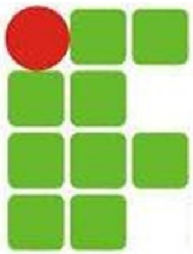
Pessoa.java | TestePessoa.java

```
1  public class TestePessoa {
2      public static void main(String[] args) {
3          Pessoa Diego = new Pessoa();
4          Diego.nome = "Diego Oliveira";
5          Diego.idade = 27;
6          Diego.peso = 70.0;
7          Diego.altura = 1.70;
8          Diego.profissao = "Profess'or";
9      }
10 }
```



Métodos

- Definem o comportamento de uma classe
- Podem ser utilizados para:
  - realizar algum trabalho dentro da classe
  - modificar o valor de algum atributo
  - resgatar o valor de um atributo
  - ativar ações em outros objetos
  - enviar dados pela rede
  - iniciar eventos de interface gráfica
  - iniciar sons
  - outras ações



Exemplo de Métodos em Java

Matematica.java | TesteMatematica.java

```
1  public class Matematica {
2      int x = 10;
3      int y = 5;
4
5      public void mostrarX(){
6          System.out.println("O valor de X é " + x);
7      }
8
9      public void somarXY(){
10         System.out.println("O valor de X+Y é " + (x+y));
11     }
12 }
```

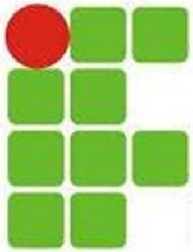
Matematica.java | TesteMatematica.java

```
1  public class TesteMatematica {
2      public static void main(String[] args) {
3          Matematica m = new Matematica();
4          m.mostrarX();
5          m.somarXY();
6      }
7  }
```

Problems | Javadoc | Outline | Task List | Declaration | Con

<terminated> TesteMatematica [Java Application] C:\Program Files\Java\jre7\bin

```
O valor de X é 10
O valor de X+Y é 15
```



Parâmetros

- Parâmetros são utilizados para passar valores para métodos
- São utilizados em casos em que o método precisa de um valor externo para realizar o seu trabalho
- Os parâmetros são passados entre parênteses logo após o nome do método
- Cada parâmetro tem um nome e um tipo

Exemplo de Parâmetros em Java

Matematica.java

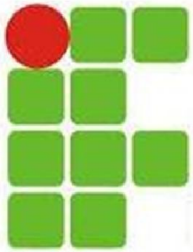
```
1  public class Matematica {
2      public void somar(int x, int y){
3          System.out.println("O valor de X+Y é " + (x+y));
4      }
5      public int multiplicar(int x, int y){
6          return x * y;
7      }
8  }
```

TesteMatematica.java

```
1  public class TesteMatematica {
2      public static void main(String[] args) {
3          Matematica m = new Matematica();
4          m.somar(5,10);
5          int resultado = m.multiplicar(5, 10);
6          System.out.println("O resultado de X*Y é " + resultado);
7      }
8  }
```

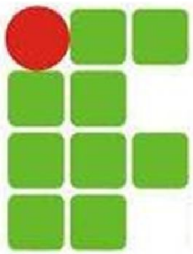
Problems | Javadoc | Outline | Task List | Declaration | Console

```
<terminated> TesteMatematica [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (Feb 8, 2014 4:39:15
O valor de X+Y é 15
O resultado de X*Y é 50
```



Visibilidade

- Definem quem pode visualizar atributos e métodos
- Modificadores de visibilidade do Java:
  - public
  - private
  - protected
  - “default”



Exemplos de Visibilidade em Java

Pessoa.java

```
2  public class Pessoa {
3      public String nome;
4      private int idade;
5      private double peso;
6      private double altura;
7      protected String profissao;
8  }
```

Pessoa.java | *TestePessoa.java

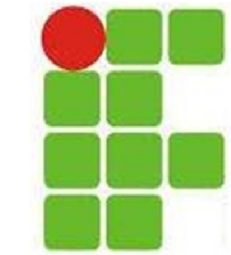
```
1  public class TestePessoa {
2      public static void main(String[] args) {
3          Pessoa Diego = new Pessoa();
4          Diego.nome = "Diego Oliveira";
5          Diego.profissao = "Professor";
6          Diego.idade = 27;
7          Diego.peso = 70.0;
8          Diego.altura = 1.70;
9      }
10 }
11
```

The field Pessoa.altura is not visible

2 quick fixes available:

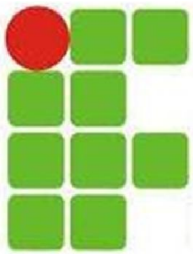
- Change visibility of 'altura' to 'default'
- Create getter and setter for 'altura'

Press 'F2' for focus



Encapsulamento

- Depende diretamente da Visibilidade
- Métodos Java para trabalhar com encapsulamento de dados:
  - **set**Atributo(parâmetros)
  - **get**Atributo()
- O encapsulamento garante maior segurança aos programas
- Encapsular dados é uma boa prática de programação e deve ser seguida



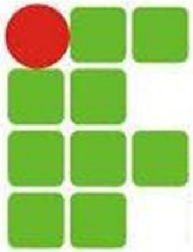
Exemplos de Encapsulamento em Java

Pessoa.java

```
1  public class Pessoa {
2      private String nome;
3      private int idade;
4      private double peso;
5      private double altura;
6      private String profissao;
7
8      public String getNome() {
9          return nome;
10     }
11     public void setNome(String nome) {
12         this.nome = nome;
13     }
```

Pessoa.java | TestePessoa.java

```
1  public class TestePessoa {
2      public static void main(String[] args) {
3          Pessoa Diego = new Pessoa();
4          Diego.setNome("Diego Oliveira");
5          Diego.setIdade(27);
6          Diego.setPeso(70.0);
7          Diego.setAltura(1.70);
8          Diego.setProfissao("Professor");
9      }
10 }
```

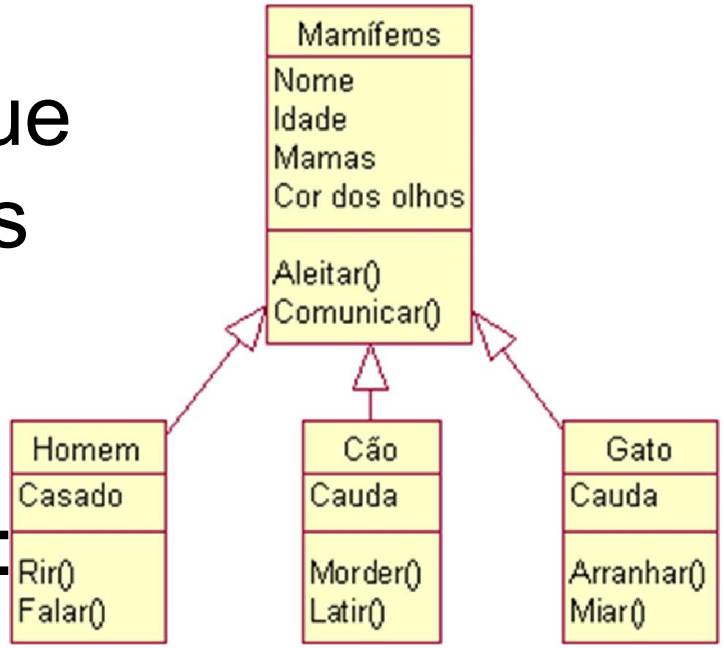


Herança

- A Herança é utilizada para o reaproveitamento de código em Java
- Uma classe herda de outra seus atributos e métodos, dependendo da visibilidade
- É uma boa prática de programação utilizar Herança para reduzir a replicação de código
- A Herança também ajuda na representação dos objetos e seus relacionamentos dentro do programa, de acordo com as necessidades

Herança

- Na figura ao lado temos a classe-mãe Mamíferos, que possui como classes filhas Homem, Cão e Gato
- Cada classe possui seus métodos de comunicação: Falar(), Latir() e Miar()
- Todos possuem os atributos nome, idade, mamas e cor dos olhos

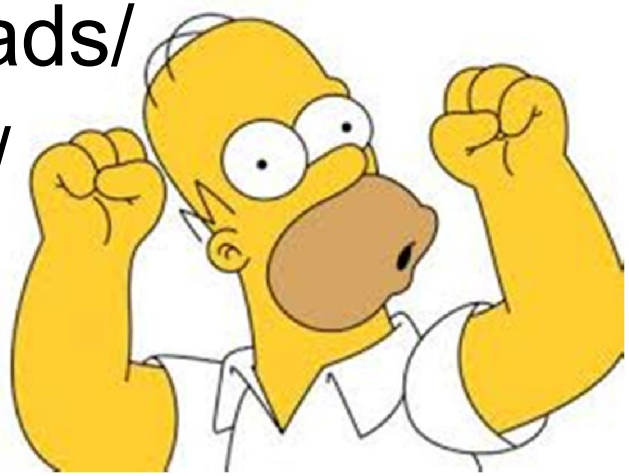


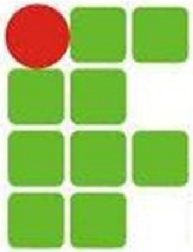
Polimorfismo

- O Polimorfismo está diretamente relacionado com a Herança
- Um método chamado em diferentes pontos da linha de Herança pode resultar em comportamentos diferentes

Indicações

- https://www.eclipse.org/downloads/
- https://netbeans.org/downloads/
- http://astah.net/download
- http://www.bluej.org/
- http://www.jcreator.com/
- http://www.oracle.com/technetwork/pt/java/javase/downloads/index.html?ssSourceSiteId=otnes





Perguntas?