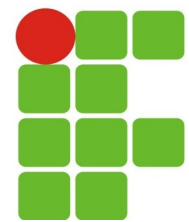


# Classes e Objetos (Cont.)

---

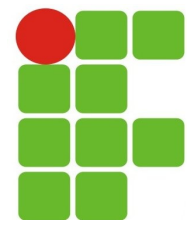
**João Paulo Q. dos Santos**  
**[joao.queiroz@ifrn.edu.br](mailto:joao.queiroz@ifrn.edu.br)**





# Classes

- Para que a JVM crie objetos:
  - Ela precisa saber qual classe o objeto pertence
    - Na classe estão definidos os **atributos** e **métodos**
- Programamos classes e depois as usamos
  - Tudo em JAVA são classes
    - Menos os tipos primitivos
  - Uma classe define um novo tipo
- Componentes da classe
  - São os **membros** da classe



# Classe

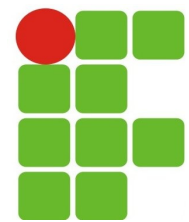
## Declaração da Classe

```
[Opções] class NomeClasse {
```

```
[atributos]  
[construtores]  
[métodos]
```

```
}
```

Corpo



# Exemplo

```
package br.edu.ifrn.exemplos;
```

```
public class Lampada {
```

```
private boolean ligada;
```

```
public Lampada(){
```

```
    ligada = false;
```

```
}
```

```
public void ligar(){
```

```
    ligada = true;
```

```
}
```

```
public void desligar(){
```

```
    ligada = false;
```

```
}
```

```
public boolean estaLigada(){
```

```
    return ligada;
```

```
}
```

```
}
```

Atributos

Construtor

Métodos

<<public>>

**Lampada**

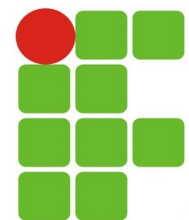
-ligada: boolean = false

+<<construtor>> Lampada()

+ligar()

+desligar()

+estaLigada(): boolean



# Usando a classe Lampada

```
package br.edu.ifrn.exemplos;

import java.util.Scanner;

public class PrincipalLampada {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        Lampada lampada = new Lampada();

        int opcao = 0;

        do {

            System.out.println("0) Sair 1) Ligar 2) desligar 3) Mostrar ");
            System.out.print("Digite opção: "); opcao = sc.nextInt();

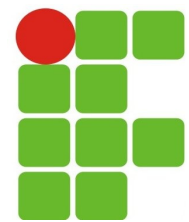
            switch (opcao){

                case 1: lampada.ligar(); break;
                case 2: lampada.desligar(); break;
                case 3:
                    if (lampada.estaLigada())
                        System.out.println("Ligada");
                    else
                        System.out.println("Desligada");
                    break;

            }

        } while (opcao != 0);

        System.out.println("Fim do programa!");
    }
}
```

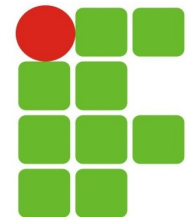


# Atributos

- São as variáveis de instância
  - Fazem parte de cada objeto (instância)
- Declarada fora dos métodos
- "Vivem" enquanto o objeto "viver"
- São pré-inicializadas:

boolean	false
Tipo aritmético	0
char	'\u0000'
class	null

Obs: Todo objeto possui um identificador chamado **this**, que é uma referência para o próprio objeto.



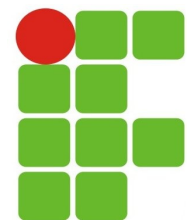
# Atributo

- Todo objeto possui um atributo que é uma referência a ele mesmo
  - Usado para acesso a membros do próprio objeto

`this.membro`

- Evita conflito
  - Com parâmetros de métodos, por exemplo
- Exemplo:

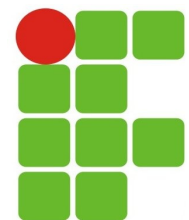
```
public class Qualquer {  
    int x, y;  
    public void mover(int x,int y){  
        this.x = x;  
        this.y = y;  
    }  
}
```



# Métodos

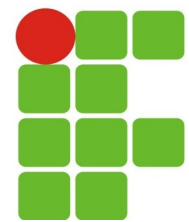
- Declaração de método:
  - `<opções> tipoRetorno nomeMetodo (parâmetros)`
    - **public int** metodo(**float** x)
- Passagem de parâmetros:
  - Deve ser informados o tipo e identificador dos parâmetros.
  - Funciona no método como uma variável normal
- Passam o valor do identificador





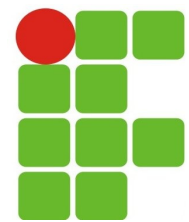
# Métodos

- Corpo do método:
  - Implementa as operações do método
  - Fica entre chaves ( { } )
  - Variáveis podem ser criadas
    - Ela é dita local
    - Não é pré-inicializada.
    - Só existe enquanto o método está em execução



# Construtores

- Mesmo nome da classe;
- Não possui retorno;
- Podem ser **vários (sobrecarga) parâmetros**
  - Diferença na **quantidade** e **tipo** dos parâmetros;
  - Eles são especificados nos parênteses após o nome da classe
- O compilador Java fornece um construtor padrão caso não seja definido nenhum outro construtor;
- É chamado na execução do **new**.

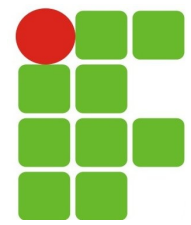


# Exemplo

- Plano cartesiano
  - Coordenadas X e Y
  - Pode ser movido de lugar
  - Podemos saber sua distância da origem

<code>&lt;&lt;public&gt;&gt;</code> <b>Ponto</b>
-x: double -y: double
+distanciaOrigem(): double +getX(): double +getY(): double +movePara(x:double,y:double): void +toString(): String

```
public class Ponto{  
  
    private double x,y;  
  
    public Ponto(){  
        x = 0; y = 0;  
    }  
  
    public Ponto(double x, double y) {  
        this.x = x; this.y = y;  
    }  
  
    public void moverPara(double x, double y){  
        this.x = x; this.y = y;  
    }  
  
    public double getX(){  
        return x;  
    }  
  
    public double getY(){  
        return y;  
    }  
  
    public double distanciaOrigem(){  
  
        double distancia;  
  
        distancia = (double)Math.sqrt(x*x +y*y);  
  
        return distancia;  
    }  
  
    public String toString(){  
        return "Ponto (" +this.x+", "+this.y+")";  
    }  
}
```



# Dúvidas

