

Classes e Objetos - Implementação

Orientação a Objetos - DCC025

Prof. Edmar Welington Oliveira  
[edmar.oliveira@ufjf.edu.br](mailto:edmar.oliveira@ufjf.edu.br)

Universidade Federal de Juiz de Fora - UFJF  
Departamento de Ciência da Computação - DCC

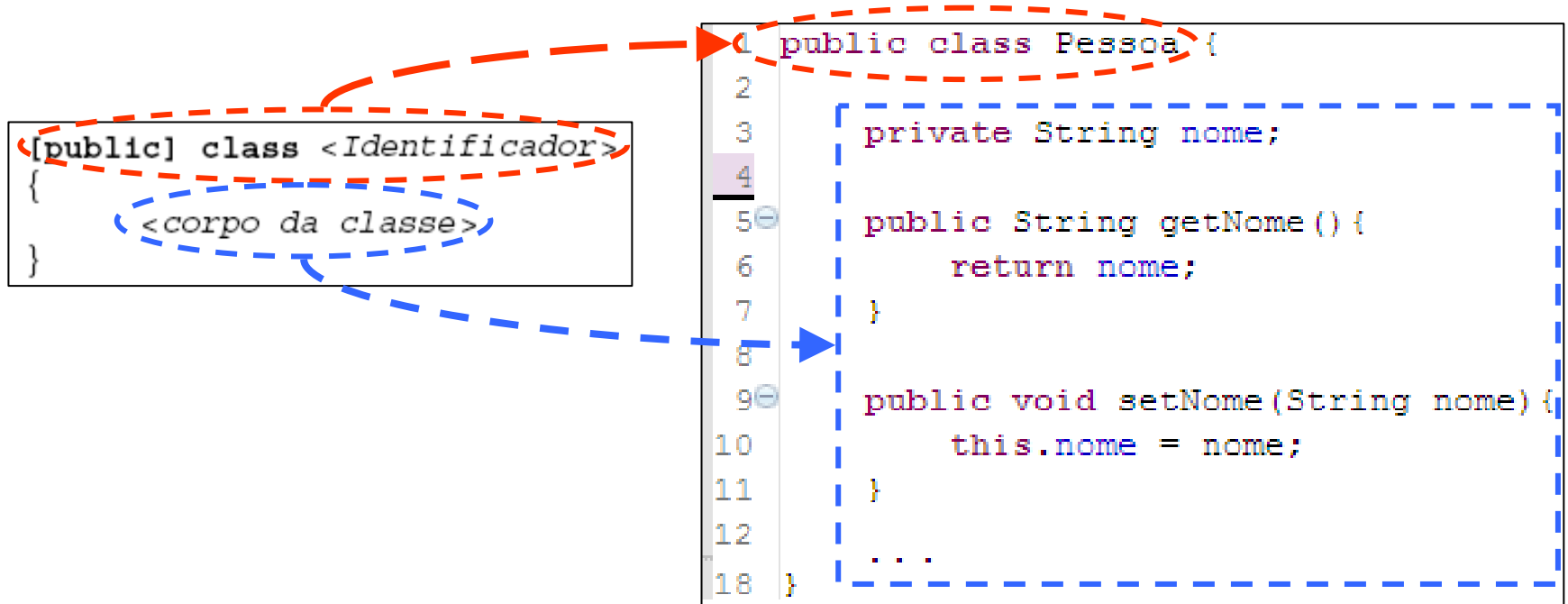
# Implementação de Classes

```
[public] class <Identificador>
{
    <corpo da classe>
}
```



```
1 public class Pessoa {
2
3     private String nome;
4
5     public String getNome() {
6         return nome;
7     }
8
9     public void setNome(String nome) {
10         this.nome = nome;
11     }
12     ...
13 }
14
15
16
17
18 }
```

# Implementação de Classes



# Programação Java - Classe

```
<modificador> <tipo> <identificador>(<parâmetros>)  
{  
    <corpo do método>  
}
```

## ■ Métodos

- Métodos que pertencem ao objeto são chamados de métodos de instância. Analisando a definição padrão de métodos, temos:
  - <modificador> é um modificador de acesso
  - <tipo> é o tipo do valor de retorno do método
  - <identificador> é o identificador do método
  - <parâmetros> é uma lista de parâmetros.
- Além disso, o corpo do método é composto de comandos, expressões e declarações de variáveis locais ao método.

# Métodos

```
1 public class Pessoa {  
2  
3     private String nome;  
4  
5     public String getNome() {  
6         return nome;  
7     }  
8  
9     public void setNome(String nome) {  
10         this.nome = nome;  
11     }  
12     ...  
18 }
```

`<modificador> <tipo> <identificador>(<parâmetros>)`  
{  
 <corpo do método>  
}

# Classes e Método Principal

```
1 public class Pessoa {  
2  
3     private String nome;  
4  
5     public String getNome() {  
6         return nome;  
7     }  
8  
9     public void setNome(String nome) {  
10        this.nome = nome;  
11    }  
12  
13    public static void main(String args[]) {  
14  
15        Pessoa p1 = new Pessoa();  
16        System.out.println(p1.getNome());  
17    }  
18 }
```

Método main() dentro da classe Pessoa

# Classes e Método Principal

```
1 public class Pessoa {  
2  
3     private String nome;  
4  
5     public String getNome() {  
6         return nome;  
7     }  
8  
9     public void setNome(String nome) {  
10         nome = nome;  
11     }  
12 }
```

```
public class Principal {  
  
    public static void main(String args[]) {  
  
        Pessoa p1 = new Pessoa();  
        System.out.println(p1.getNome());  
    }  
}
```

Prefira construir uma classe de execução, que, quando necessário, irá chamar os métodos da classe Pessoa

# Modificadores - Introdução



# Modificadores

<i>Modificador</i>	<i>Descrição</i>
default	Somente classes do mesmo package possuem acesso
public	Todos possuem acesso
protected	Apenas os membros da classe e subclasse
private	Apenas os membros da classe

# Modificadores em Classes

```
[public] class <Identificador>
{
    <corpo da classe>
}
```

## ■ Palavra reservada public

- É opcional e indica que a classe pode ser referenciada por qualquer outra classe além do próprio pacote. Se a declaração da classe não for precedida de public, então ela só poderá ser referenciada por outras classes do mesmo pacote (**visibilidade default**)
  - OBS: Os modificadores de acesso **private** e **protected** não podem ser aplicados às classes.
- O **<Identificador>** é usado para referenciar a classe.
- No **<corpo da classe>** são definidos os atributos e métodos da classe.

# Modificadores em Classes

```
1 public class Pessoa {  
2  
3     private String nome;  
4  
5     public String getNome() {  
6         return nome;  
7     }  
8  
9     public void setNome(String nome) {  
10        this.nome = nome;  
11    }  
12    ...  
18 }
```

Regra Geral:

Definir como "private" todos os atributos da classe

Regra Geral:

Definir como público os métodos da classes. OBS: alguns métodos deverão, em circunstância específicas, ser definidos como privados. Contudo, como geral geral, utilize "public"

# Exemplo de Acesso

```
public class Aluno {  
  
    public int matricula;  
    public Date dataNascimento;  
    public String nome;  
  
}
```

Tentativa de acesso 01 - T01

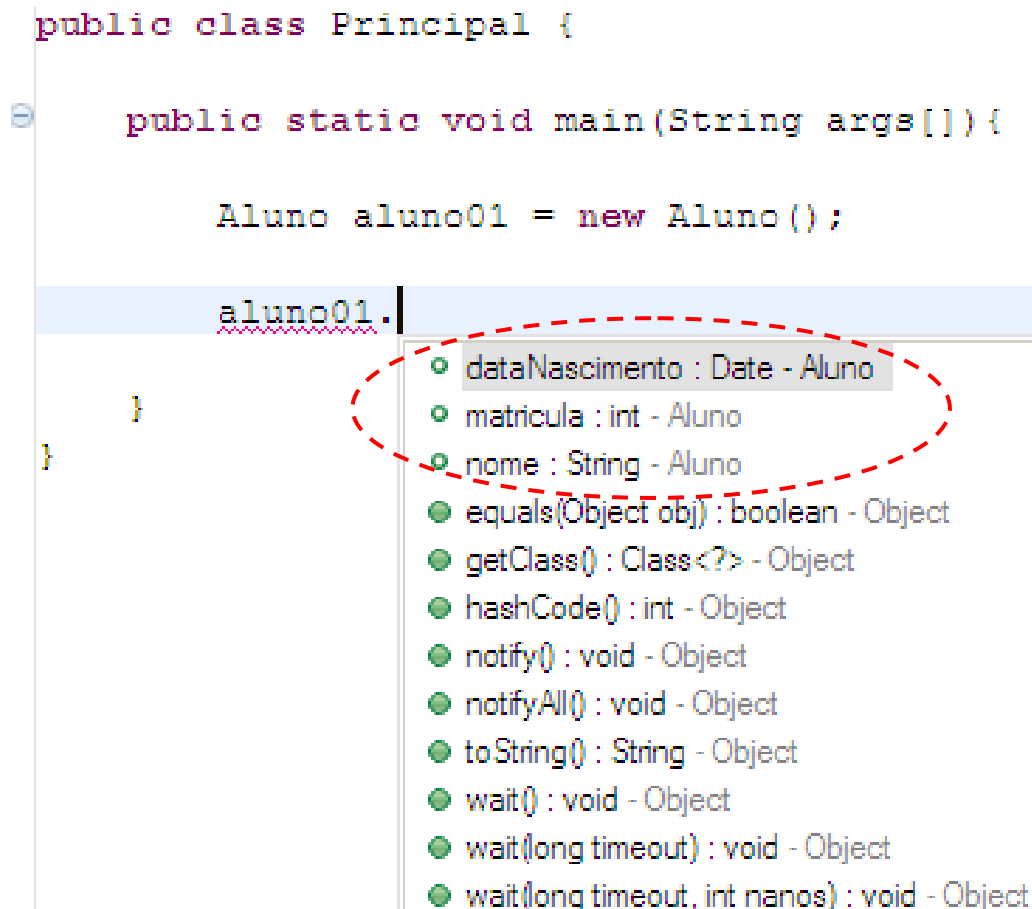
```
public class Aluno {  
  
    private int matricula;  
    private Date dataNascimento;  
    private String nome;  
  
}
```

```
public class Principal {  
  
    public static void main(String args[]){  
  
        Aluno aluno01 = new Aluno();  
  
    }  
  
}
```

Tentativa de acesso 02 - T02

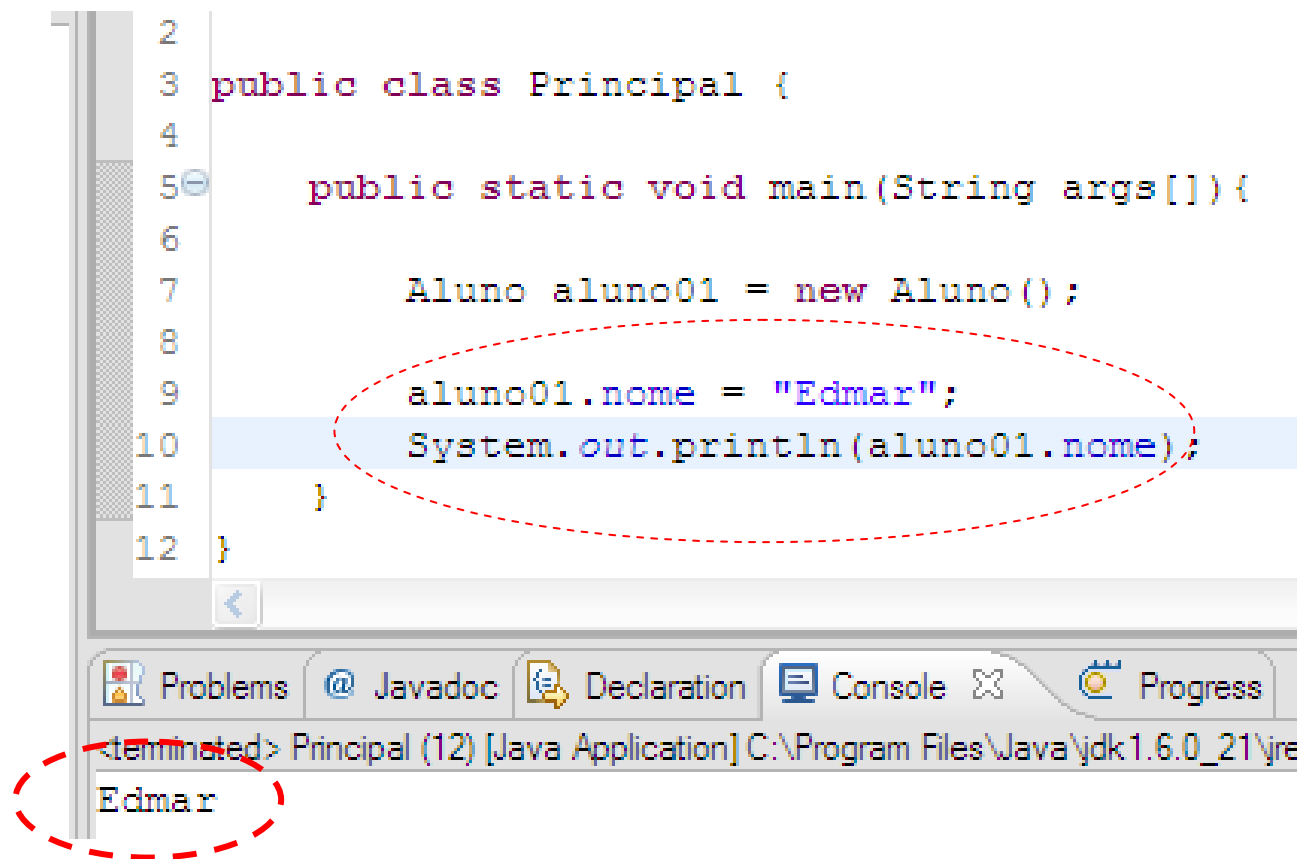
# Acesso Público - T01

```
public class Principal {  
    public static void main(String args[]) {  
        Aluno aluno01 = new Aluno();  
        aluno01.  
    }  
}
```



- dataNascimento : Date - Aluno
- matricula : int - Aluno
- nome : String - Aluno
- equals(Object obj) : boolean - Object
- getClass() : Class<?> - Object
- hashCode() : int - Object
- notify() : void - Object
- notifyAll() : void - Object
- toString() : String - Object
- wait() : void - Object
- wait(long timeout) : void - Object
- wait(long timeout, int nanos) : void - Object

## Acesso Público - T01



```
2
3 public class Principal {
4
5     public static void main(String args[]) {
6
7         Aluno aluno01 = new Aluno();
8
9         aluno01.nome = "Edmar";
10        System.out.println(aluno01.nome);
11    }
12 }
```

Problems Javadoc Declaration Console Progress

<terminated> Principal (12) [Java Application] C:\Program Files\Java\jdk1.6.0\_21\re  
Edmar

## Acesso Privado - T02

```
public class Principal {  
    public static void main(String args[]) {  
        Aluno aluno01 = new Aluno();  
        aluno01.  
    }  
}
```

- equals(Object obj) : boolean - Object
- getClass() : Class<?> - Object
- hashCode() : int - Object
- notify() : void - Object
- notifyAll() : void - Object
- toString() : String - Object
- wait() : void - Object
- wait(long timeout) : void - Object
- wait(long timeout, int nanos) : void - Object

# Public/Private - Classe

```
public class Aluno {  
  
    private int matricula;  
    private Date dataNascimento;  
    private String nome;  
  
    public static void main(String args[]){  
  
        Aluno a1 = new Aluno();  
        a1.  
    }  
}
```

blems @ Javado  
ated> Principal (12)

- ▣ dataNascimento : Date - Aluno
- ▣ matricula : int - Aluno
- ▣ nome : String - Aluno
- ◆ clone() : Object - Object
- equals(Object obj) : boolean - Object
- ◆ finalize() : void - Object
- getClass() : Class<?> - Object
- hashCode() : int - Object
- notify() : void - Object
- notifyAll() : void - Object
- toString() : String - Object
- wait() : void - Object
- wait(long timeout) : void - Object
- wait(long timeout, int nanos) : void - Object
- <sup>S</sup>main(String[] args) : void - Aluno



# Exercício

## ■ Exercício 01

- Crie uma classe Empregado que inclui 3 campos como variáveis de instâncias - nome (tipo string), sobrenome (tipo String) e um salário mensal (tipo double). Forneça um método "set" e um "get" para cada campo.
- Crie um classe de teste. Instancie um objeto da classe Empregado e atribua os valores "Pedro", "Silva" e 1000 para seus atributos. Imprima o nome do empregado e seu salário atual
- Em seguida
  - Crie um método aumentarSalario(). Esse método deve ler o salário atual do empregado, dobrar esse valor e atualizar no salário do empregado. Depois, imprima o novo salário

## Solução - Classe Empregado

```
3 public class Empregado {  
4  
5     private String nome;  
6     private String sobrenome;  
7     private double salario;  
8  
9     public String getNome() {  
10         return nome;  
11     }  
12  
13     public void setNome(String nome) {  
14         this.nome = nome;  
15     }  
16  
17     public String getSobrenome() {  
18         return sobrenome;  
19     }  
20  
21     public void setSobrenome(String sobrenome) {  
22         this.sobrenome = sobrenome;  
23     }  
24  
25     public double getSalario() {  
26         return salario;  
27     }  
28  
29     public void setSalario(double salario) {  
30         this.salario = salario;  
31     }  
32 }
```

## Solução - Classe de Execução

```
3 public class Principal {  
4  
5     public static void main(String args[]){  
6  
7         Empregado p1 = new Empregado();  
8  
9         p1.setNome("Pedro");  
10        p1.setSobrenome("Silva");  
11        p1.setSalario(1000);  
12  
13        System.out.println("Nome Empregado: " + p1.getNome() + " " + p1.getSobrenome() );  
14        System.out.println("Salário: R$" + p1.getSalario());  
15    }
```

## Solução - Novo Método

```
3 public class Empregado {  
4  
5     private String nome;  
6     private String sobrenome;  
7     private double salario;  
8  
9     public String getNome() {  
10         return nome;  
11     }  
12  
28  
29     public void setSalario(double salario) {  
30         this.salario = salario;  
31     }  
32  
33     public void aumentarSalario(){  
34         double salarioAtual = this.getSalario();  
35         double novoSalario = salarioAtual * 2;  
36         this.setSalario(novoSalario);  
37     }  
38 }
```

## Solução - Novo Método

```
public void aumentarSalario(){  
    double salarioAtual = salario;  
    double novoSalario = salarioAtual * 2;  
    salario = novoSalario;  
}
```

```
public void aumentarSalario(){  
    setSalario(salario * 2);  
}
```

```
public void aumentarSalario(){  
    setSalario(getSalario() * 2);  
}
```

```
public void aumentarSalario(){  
    salario = salario * 2;  
}
```

## Solução - Classe de Execução

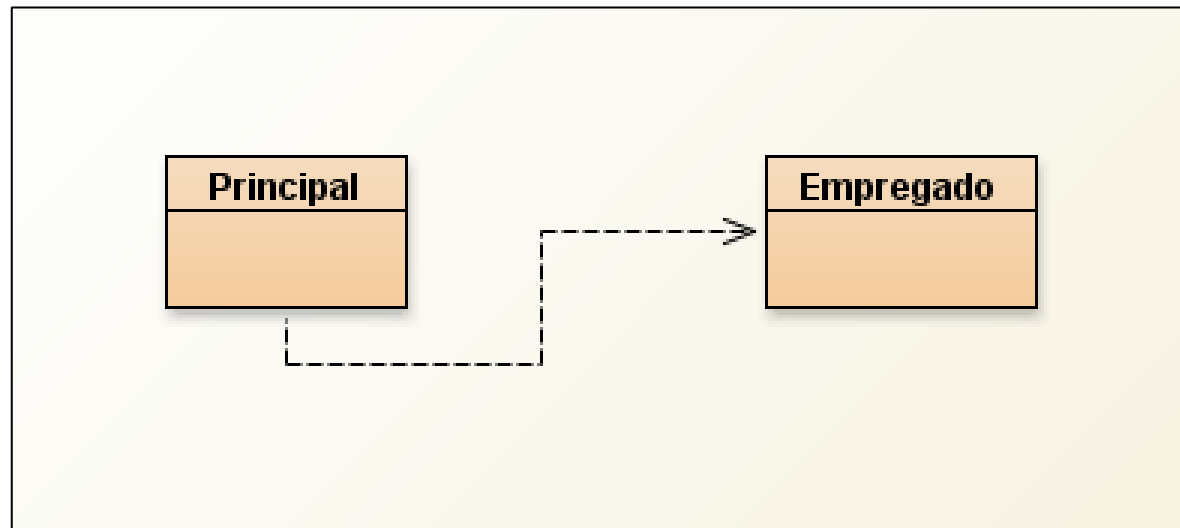
```
3 public class Principal {
4
5     public static void main(String args[]){
6
7         Empregado p1 = new Empregado();
8
9         p1.setNome("Pedro");
10        p1.setSobrenome("Silva");
11        p1.setSalario(1000);
12
13        System.out.println("Nome Empregado: " + p1.getNome() + " " + p1.getSobrenome() );
14        System.out.println("Salário: R$" + p1.getSalario());
15
16        p1.aumentarSalario();
17
18        System.out.println("Novo NSalário: R$" + p1.getSalario());
19    }
20
21 }
```

Problems Javadoc Declaration Console Progress

<terminated> Principal (13) [Java Application] C:\Program Files\Java\jdk1.6.0\_21\jre\bin\javaw.exe (17/08/2011 14:13:42)

Nome Empregado: Pedro Silva  
Salário: R\$1000.0  
Novo NSalário: R\$2000.0

# Representação em Modelo



# Classes e Objetos - Implementação

## Orientação a Objetos - DCC025

Prof. Edmar Welington Oliveira  
[edmar.oliveira@ufjf.edu.br](mailto:edmar.oliveira@ufjf.edu.br)

Universidade Federal de Juiz de Fora - UFJF  
Departamento de Ciência da Computação - DCC