

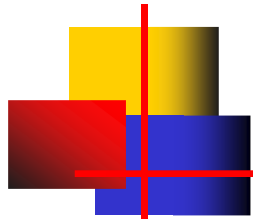


## POO - Programação Orientada a Objetos com Java

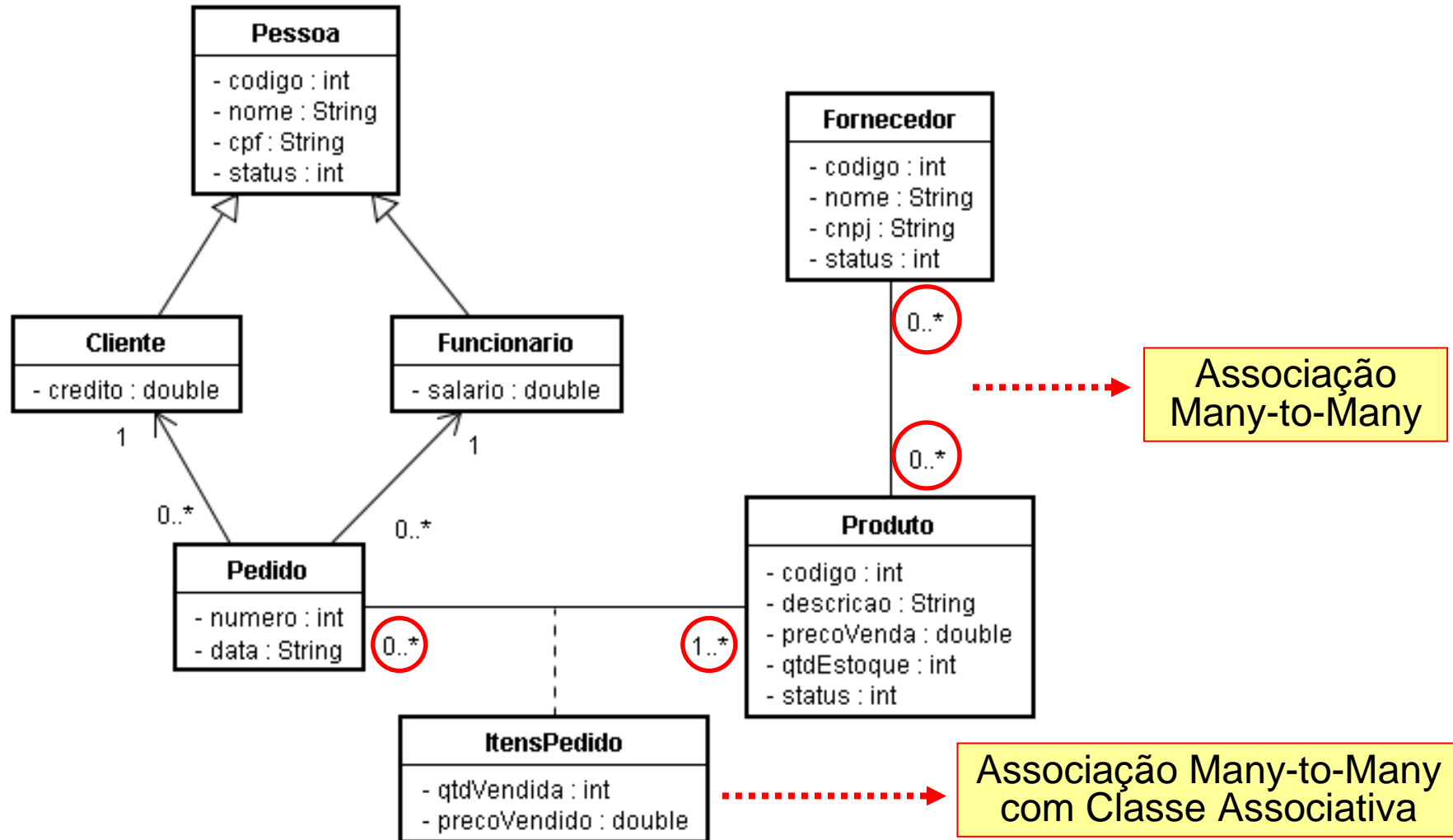
---

### Aula 05: Associação Unidirecional: Many-to-Many, Sem Classe Associativa

Prof. Sérgio Borges  
[ensinalegal@gmail.com](mailto:ensinalegal@gmail.com)  
[www.ensinalegal.net](http://www.ensinalegal.net)



# Associação Many-to-many



# Classe Produto

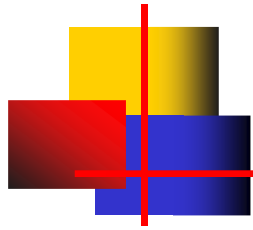
```
public class Produto {  
    private int codigo;  
    private String descricao,  
    private int qtdEstoque;  
    private double precoVenda;  
    private int status;  
    private List<Fornecedor> fornecedores;
```

Atributos da classe  
**Produto**

Coleção de Objetos  
(fornecedores) da classe  
**Fornecedor**

```
//Construtor nulo  
public Produto(){  
    fornecedores = new ArrayList<Fornecedor>();  
} // fim do construtor
```

Nos métodos construtores é necessário alocar memória para a coleção de objetos (**fornecedores**). Neste caso, objetos da classe **Fornecedor**.



# Classe Produto

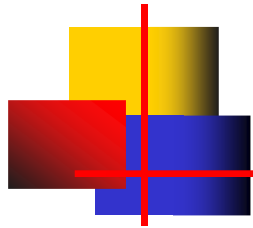


**//continuação**

```
public Produto(int codigo, String descricao,  
               int qtdEstoque, double precoVenda, int status) {  
    this.codigo = codigo;  
    this.descricao = descricao;  
    this.qtdEstoque = qtdEstoque;  
    this.precoVenda = precoVenda;  
    this.status = status;  
    fornecedores = new ArrayList<Fornecedor>();  
} // fim do construtor
```

**//continua**

Novamente, alocando memória para a coleção de objetos (**fornecedores**) no construtor com parâmetros.



# Classe Produto



// **continuação**

```
public void setCodigo(int codigo){this.codigo = codigo;}
```

```
public int getCodigo(){ return this.codigo; }
```

```
public void setDescricao(String descricao){this.descricao = descricao; }
```

```
public String getDescricao() { return this.descricao; }
```

```
public void setQtdEstoque(int qtd){ this.qtdEstoque = qtd; }
```

```
public String getQtdEstoque{ return this.qtdEstoque; }
```

```
public void setPrecoVenda(int preco) { this.precoVenda = preco; }
```

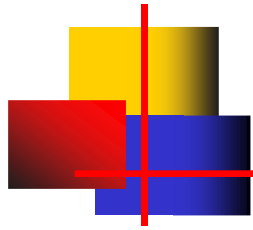
```
public double getPrecoVenda(){ return this.precoVenda; }
```

```
public void setStatus(int status) { this.status = status; }
```

```
public int getStatus(){ return this.status; }
```

// **continua**

Métodos públicos (**public**) **sets** e **gets** para manipular os atributos que são **private**, aplicando o princípio de **Encapsulamento**



# Classe Produto

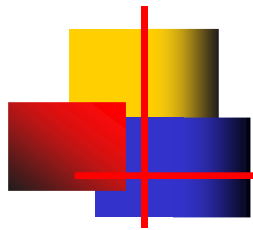


//continuação

```
public void addFornecedor(Fornecedor fornecedor){  
    this.fornecedores.add(fornecedor);  
}
```



Método que adiciona um objeto **fornecedor** na coleção (**fornecedores**) da classe Produto.



# Classe Produto



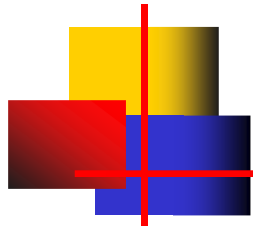
//continuação

```
public boolean removeFornecedor (Fornecedor fornecedor){  
    return this.fornecedores.remove(fornecedor);  
}
```

```
public List<Fornecedor> getFornecedores() {  
    return this.fornecedores;  
}
```

O método **remove** verifica se o fornecedor está na lista de **fornecedores** do Produto (**this**). Caso afirmativo, o objeto **fornecedor** é removido (**remove**) e é retornado true (verdadeiro)

Método **getFornecedores** que retorna os **fornecedores** de um produto (**this**), que retorna uma coleção (*List*) de **Fornecedor**.



# Classe Fornecedor



```
public class Fornecedor {
```

```
    private int codigo;
```

```
    private String nome,
```

```
    private String cnpj;
```

```
    private int status;
```

```
    private List<Produto> produtos;
```



Atributos da classe  
**Fornecedor**



Coleção de Objetos  
(**produtos**) da classe  
**Produto**

```
//Construtor nulo
```

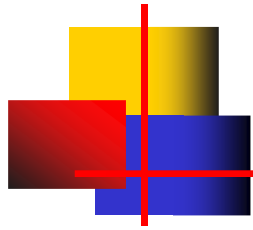
```
public Fornecedor () {
```

```
    produtos = new ArrayList<Produto>();
```

```
} // fim do construtor
```



Nos métodos construtores é necessário alocar memória para a coleção de objetos (**produtos**). Neste caso, objetos da classe **Produto**.



# Classe Fornecedor

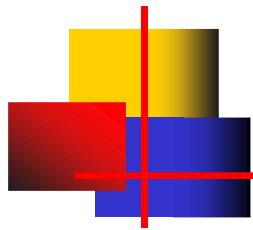


## //continuação

```
public Fornecedor(int codigo, String nome, String cnpj) {  
    this.codigo = codigo;  
    this.nome = nome;  
    this.cnpj = cnpj;  
    produtos = new ArrayList<Produto>();  
} // fim do construtor
```

## //continua

Novamente, é alocado memória para a coleção de objetos (**produtos**) no construtor com parâmetros.



# Classe Fornecedor



//continuação

```
public void setCodigo(int codigo){this.codigo = codigo;}
```

```
public int getCodigo(){ return this.codigo; }
```

```
public void setNome(String nome){this.nome = nome; }
```

```
public String getNome() { return this.nome; }
```

```
public void setCnpj(String cnpj){this.cnpj = cnpj; }
```

```
public String getCnpj() { return this.cnpj; }
```

```
public void setStatus(int status) { this.status = status; }
```

```
public int getStatus(){ return this.status; }
```

// continua

Métodos públicos  
(**public**) **sets** e **gets**  
para manipular os  
atributos que são  
**private**, aplicando o  
princípio de  
**Encapsulamento**

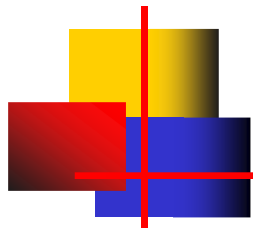
# Classe Fornecedor

//continuação

```
public void addProduto(Produto produto){  
    this.produtos.add(produto);  
}
```



Método que adiciona um objeto **produto** na coleção de produtos (**produtos**) da classe Fornecedor



# Classe Fornecedor



//continuação

```
public boolean removeProduto (Produto produto){  
    return this.produtos.remove(produto);  
}
```

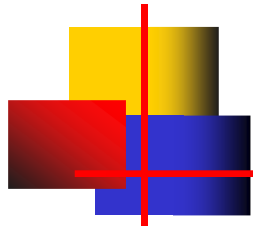


```
public List<Produtos> getProdutos() {  
    return this.produtos;  
}
```



Método **getProdutos** que retorna os **produtos** (coleção) fornecidos por um fornecedor (**this**).

O método **remove** verifica se o **produto** está na lista de **produtos** fornecidos pelo Fornecedor (**this**). Caso afirmativo, o objeto **produto** é removido (**remove**) e é retornado verdadeiro.



# Testes



```
public static void main(String[] args) {
```

```
    Fornecedor for1 = new Fornecedor(100, "Empresa A", "123");
```

```
    Fornecedor for2 = new Fornecedor(101, "Empresa B", "321");
```

```
    Produto pr1 = new Produto(50, "Lápis", 100, 1.2, 1);
```

```
    Produto pr2 = new Produto(100, "Caneta", 100, 1.5, 1);
```

```
    Produto pr3 = new Produto(150, "Caderno", 100, 5.5, 1);
```

O fornecedor **for1** fornece três produtos (**pr1**, **pr2** e **pr3**) e **for2** dois produtos (**pr1** e **pr3**)

```
    for1.addProduto(pr1); for1.addProduto(pr2); for1.addProduto(pr3);
```

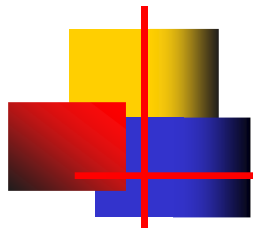
```
    for2.addProduto(pr1); for2.addProduto(pr3);
```

```
    for1.removeProduto(pr1);
```

O fornecedor **for1** deixa de fornecer o produto **pr1**

```
    System.out.println("Fornecedores: " + pr1.getFornecedores().size()); //saída: 1
```

```
    System.out.println("Produtos: " + for1.getProdutos().size()); // saída: 2
```



- Referências:

The java Tutorials. Disponível em: <http://java.sun.com/docs/books/tutorial/java/javaOO/>. Acessado em: 17-mar-09.

DEITEL, H.M.; DEITEL, P.J. *Java: como programar*. 6 ed. São Paulo: Pearson, 2005.