



Exemplo - Chain of Responsibility pattern



**Certified
Developer**
The Ultimate Tech Degree

DigitalHouse >
Coding School



Exemplo - Chain of Responsibility pattern

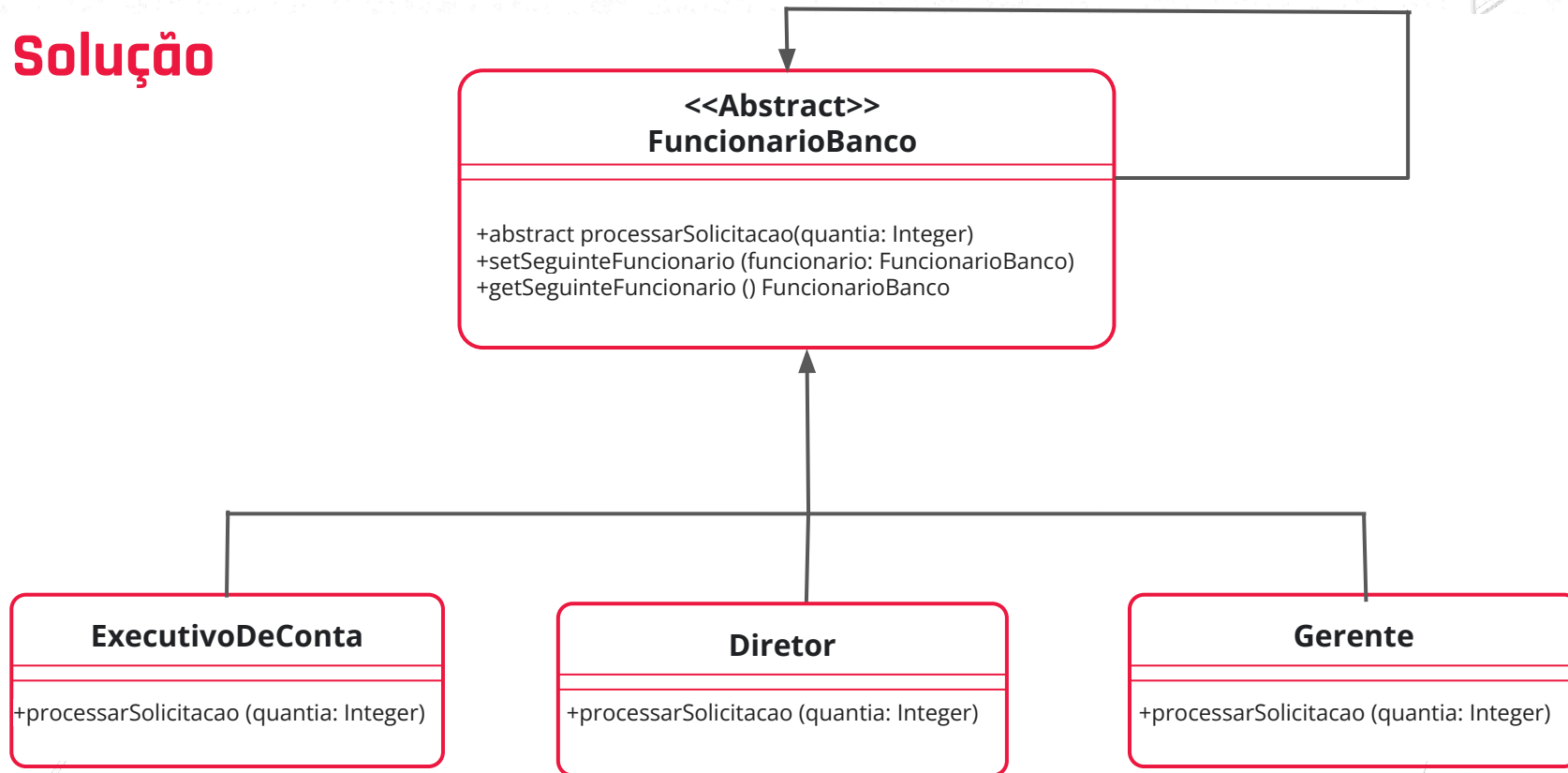
Imaginemos que estamos desenvolvendo um sistema para a área de crédito de um banco e queremos que quando um cliente solicita um crédito, um pedido seja enviado às diferentes autoridades encarregadas de autorizá-lo. O banco nos disse que:

- Se o valor não ultrapassar 60.000, o executivo de contas pode aprová-lo.
- Se o valor estiver entre 60.000 e 200.000, o gerente é quem o aprova.
- Se o valor for superior a 200.000, é aprovado pelo diretor.





Solução





Implementação das classes do diagrama UML

```
public abstract class FuncionarioBanco {  
  
    protected FuncionarioBanco seguinteFuncionario;  
    public abstract void processarSolicitacao(Integer quantia);  
  
    public FuncionarioBanco setSeguinteFuncionario(FuncionarioBanco  
    seguinteFuncionario) {  
        this.seguinteFuncionario = seguinteFuncionario;  
        return this;  
    }  
}
```

A **classe manipuladora** (handler) é responsável por iniciar a corrente com ela, neste caso, **FuncionarioBanco**.



```
public class Diretor extends FuncionarioBanco {  
    @Override  
    public void processarSolicitacao(Integer quantia) {  
        if (quantia > 200000)  
            System.out.println("Eu estou encarregado de  
administrar isso. Diretor");  
        else if (this.seguinteFuncionario != null)  
            this.seguinteFuncionario.processarSolicitacao(quantia);  
    }  
}
```

Subclasses nas quais definiremos
sob
quais critérios e como elas irão
processar a solicitação.



```
public class ExecutivoDeConta extends FuncionarioBanco {  
  
    @Override  
    public void processarSolicitacao(Integer quantia) {  
        if (quantia < 60000)  
            System.out.println("Eu estou encarregado de administrar  
isso. Executivo de Conta");  
        else if (this.seguinteFuncionario != null)  
            this.seguinteFuncionario.processarSolicitacao(quantia);  
    }  
}
```

Subclasses nas quais definiremos
sob
quais critérios e como elas irão
processar a solicitação.



```
public class Gerente extends FuncionarioBanco {  
  
    @Override  
    public void processarSolicitacao(Integer quantia) {  
        if (quantia >= 60000 && quantia <= 200000)  
            System.out.println("Eu estou encarregado de administrar  
isso. Gerente");  
        else if (this.seguinteFuncionario != null)  
            this.seguinteFuncionario.processarSolicitacao(quantia);  
    }  
}
```

Subclasses nas quais definiremos
sob
quais critérios e como elas irão
processar a solicitação.



```
public class Main {  
  
    public static void main(String[] args) {  
  
        FuncionarioBanco handlerBase = new  
        ExecutivoDeConta().setSeguinteFuncionario(new  
        Gerente().setSeguinteFuncionario(new Diretor()));  
  
        handlerBase.processarSolicitacao(78000);  
  
    }  
}
```

Test

DigitalHouse>
Coding School