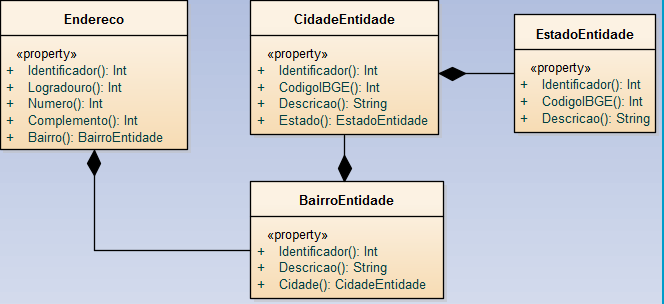
Thiago Garcez Penachia

**Princípio da Responsabilidade Única**



**Princípio do Aberto-Fechado**

public abstract class Arquivo

{

    public abstract void Gerar();

}

public class ArquivoWord : Arquivo

{

    public override void Gerar()

    {

        // codigo para geracao do arquivo

    }

}

public class ArquivoPdf : Arquivo

{

    public override void Gerar()

    {

        // codigo para geracao do arquivo

    }

}

public class ArquivoTxt : Arquivo

{

    public override void Gerar()

    {

        // codigo para geracao do arquivo

    }

}

public class GeradorDeArquivos

{

    public void GerarArquivos(IList<Arquivo> arquivos)

    {

        foreach(var arquivo in arquivos)

            arquivo.Gerar();

    }

}

**Princípio de Substituição de Liskov**

public class GeradorDeArquivos

{

    public void GerarArquivos(IList<Arquivo> arquivos)

    {

        foreach(var arquivo in arquivos)

        {

            if (arquivo is ArquivoWord)

                ((ArquivoWord)arquivo).GerarDocX();

            else if (arquivo is ArquivoPdf)

                ((ArquivoPdf)arquivo).GerarPdf();

        }

    }

}

**Princípio da Inversão de Dependência**

IDispositivo = interface

procedure Ligar;

procedure Desligar;

end;

TLampada = class(TInterfatedObject, IDispositivo)

public

procedure Ligar();

procedure Desligar();

end;

TBotao = class

private

FDispositivo: IDispositivo;

public

constructor Create(Dispositivo: IDispositivo);

procedure Acionar();

end;

implementation

constructor TBotao.Create(Dispositivo: IDispositivo);

begin

FDispositivo := Dispositivo;

end;

procedure TBotao.Acionar();

var

DispositivoEstaLigado: Boolean;

begin

if DispositivoEstaLigado then

FDispositivo.Desligar()

else

FDispositivo.Ligar();

end;

**Princípio de Segregação de Interface**

    float getSalario();

}

interface Comissionavel(){

    float getComissao();

}

abstract class Funcionario implements Assalariado {

    private float salarioBase;

    private String cargo;

    private String nome;

*@override*

    public float getSalario(){

        return this.salarioBase;

    }

}

class Vendedor extends Funcionario implements Comissionavel  {

    private float totalVendas;

*@overrride*

    public float getSalario(){

        return this.salarioBase + this.getComissao();

    }

*@override*

    public float getComissao(){

        return this.totalVendas \* 0.1;

    }

}

class Representante implements Comissionavel {

    private float totalVendas;

*@override*

    public float getComissao(){

        return this.totalVendas \* 0.3;

    }

}