MVC

Model – View - Controller

DevTeam

PROGRAMAÇÃO ORIENTADA A OBJETO - MVC

CLASSES E OBJETOS

ABSTRAÇÃO E ENCAPSULAMENTO

DevTeam

ANALOGIA A CLASSE E OBJETO

Classe

Objeto 1

Objeto 2

Objeto 3

INFORMAÇÕES SOBRE CLASSES E OBJETOS

- A garrafa é a classe onde será colocadas os objetos assim como podemos criar uma classe e usar vários objetos. Isso não é regra mais é possível. Partido do principio da Responsabilidade Única toda classe deve ter seus objetos e suas responsabilidades.

DevTeam

VAMOS COMPREENDER 4 PILARES DO MVC

- 1 - Abstração
- 2 – Encapsulamento
- 3 – Herança
- 4 – Polimorfismo

DevTeam

ABSTRAÇÃO

- Trazer algo do mundo real para o mundo computacional... Vamos compreender melhor
- Tenho uma classe clamada Tpessoa
- Inteface
- Type
  - Tpessoa = class
    - Private
    - Public
      - Nome: string;
      - Endereco: string;
      - Data_nasc: String;
      - Salario : Double;
- End;

Classe -> TPessoa

Abstração

Estou trazendo algo do mundo real para o mundo computacional em forma de Objetos. Propriedade da Pessoa (características que todos nós temos).

DevTeam

ENCAPSULAMENTO

- O que é o **encapsulamento**? **Encapsular** os dados de uma aplicação significa evitar que estes sofram acessos indevidos. Para isso, é criada uma estrutura que contém métodos que podem ser utilizados por qualquer outra classe, sem causar inconsistências no desenvolvimento de um código.

DevTeam

EXEMPLOS DO ENCAPSULAMENTO NA PRÁTICA

```
unit U_classepessoa;

interface
type
  Tpessoa = class
    private
      Fnome:String;
      Fendereco:string;
      Fcidade:string;

    public

      property nome : string read getNome write Setnome;
      property endereco : string read Fendereco write Fendereco;
      property cidade: string read Fcidade write Fcidade;

      constructor create;
      destructor destroy;override;

  end;
```

Criamos property na seção public

Read -> Leitura
Write -> Escrita

Obs. O objeto nome

DevTeam

EXEMPLOS DO ENCAPSULAMENTO NA PRÁTICA

```
Tpessoa = class
  private
    Fnome : string;
    Fendereco:string;
    Fcidade:string;

  public

    property nome : string read getNome write Setnome;
    property endereco : string read Fendereco write Fendereco;
    property cidade: string read Fcidade write Fcidade;

end;
```

O nome foi criado as
Get e Set
do Encapsulamento

Posicione o cursor na linha do código e
Pressione Ctrl + Shift + C

DevTeam

```
type
  Tpessoa = class
   private
    Fnome:String;
    Fendereco:string;
    Fcidade:string;
    function getNome: string;
    procedure Setnome(const Value: string);

   public

    property nome : string read getNome write Setnome;
    property endereco : string read Fendereco write Fendereco;
    property cidade: string read Fcidade write Fcidade;

    constructor create;
    destructor destroy;override;

  end;
```

1

Observe que foi criado dois métodos

1 – Function

1 – Procedure

Apenas no objeto nome....

2 Observação:

que apenas o nome recebeu o Get e Set

DevTeam

```
implementation

{ Tpessoa }

constructor Tpessoa.create;
begin
    inherited create ;
end;

destructor Tpessoa.destroy;
begin

 inherited;
end;

function Tpessoa.getNome: string;
begin
    Result := Fnome; //digite essa linha de código...
end;

procedure Tpessoa.Setnome(const Value: string);
begin
    Fnome:= Value; //digite essa linha de código...
end;

end.
```

Foram criados os encapsulamentos (métodos) que tanto esperávamos.

****

Aqui iremos atribuir o valor das variáveis.

O que será isso

Porque só veio o nome???

Agora sim!

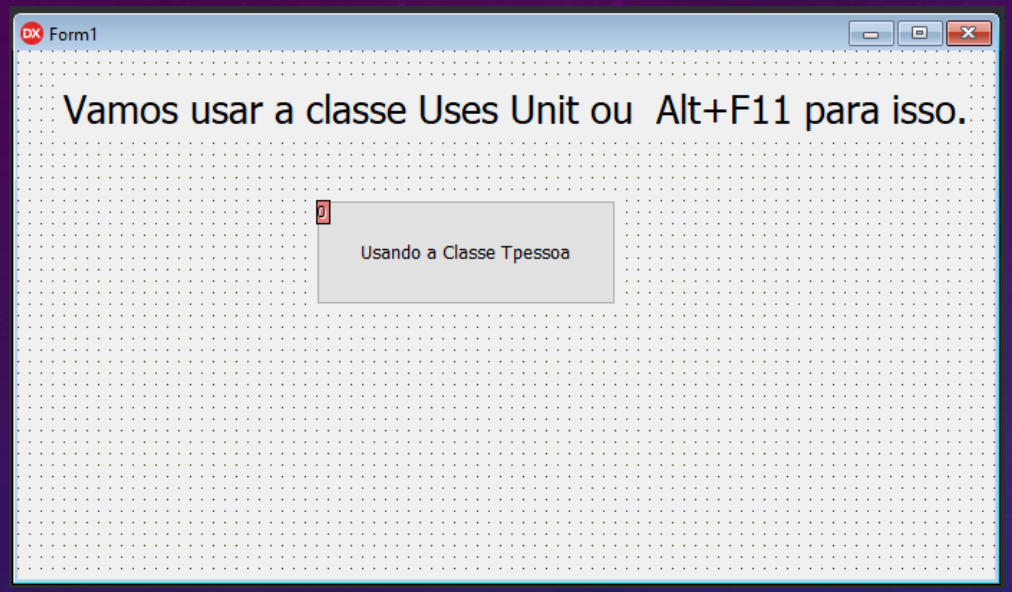
Estou
Compreendendo
Os Gets e Sets

```
function Tpessoa.getNome: string;
begin
  Result := Fnome; //todo GET -> Recebe Result (Resultado)
end;

procedure Tpessoa.Setnome(const Value: string);
begin
  Fnome:= Value; // Todo SET -> Recebe Value (valor)
end;

end.
```

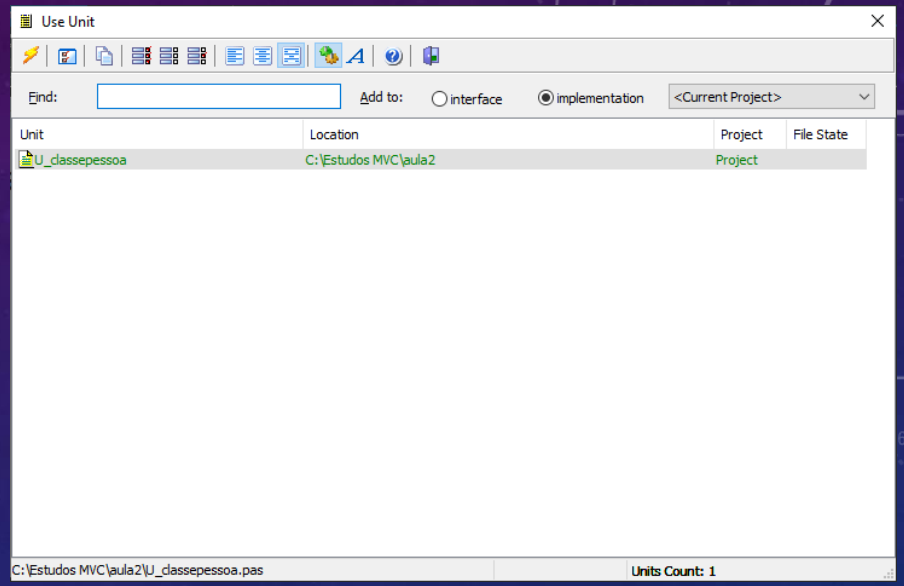
Onde vou aplicar isso na prática?

DevTeam

Vamos ver na prática o funcionamento



Uses
U_classepessoa




```
unit U_ViewPrincipal;

interface

uses
  Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,
  Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, U_classepessoa;
```

DevTeam

CLICK NO BOTÃO E VAMOS USAR A CLASSE TPESSOA

```
procedure TForm1.btn1Click(Sender: TObject);
var
pessoa:Tpes
begin

end;
```

type Tpessoa: class(TObject);

Partindo do princípio estamos criando uma variável do tipo Tpessoa.

Veja que o formulário já encontrou nossa classe (Tpessoa) que herda de (TObject).

Assim o mesmo está criando aquela abstração do mundo real

Para o mundo computacional como relatamos anteriormente...

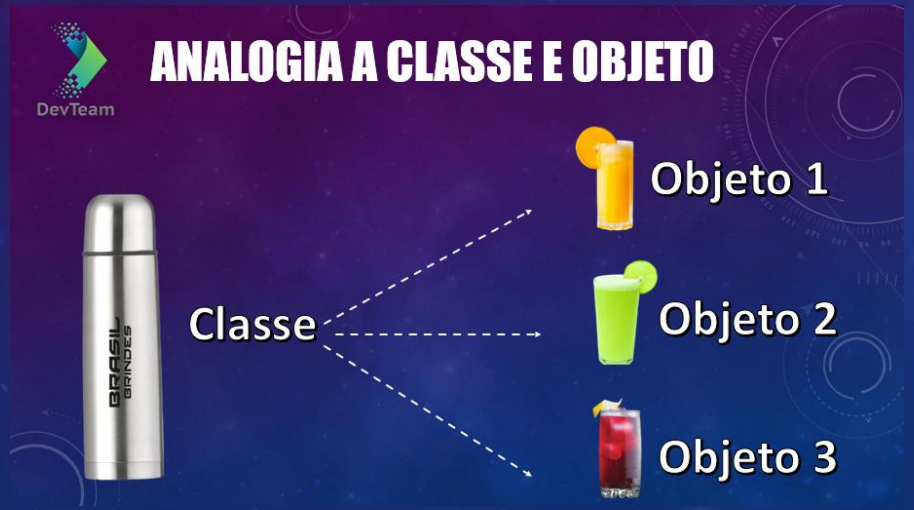
Para usarmos no formulário principal.

Para que isso?

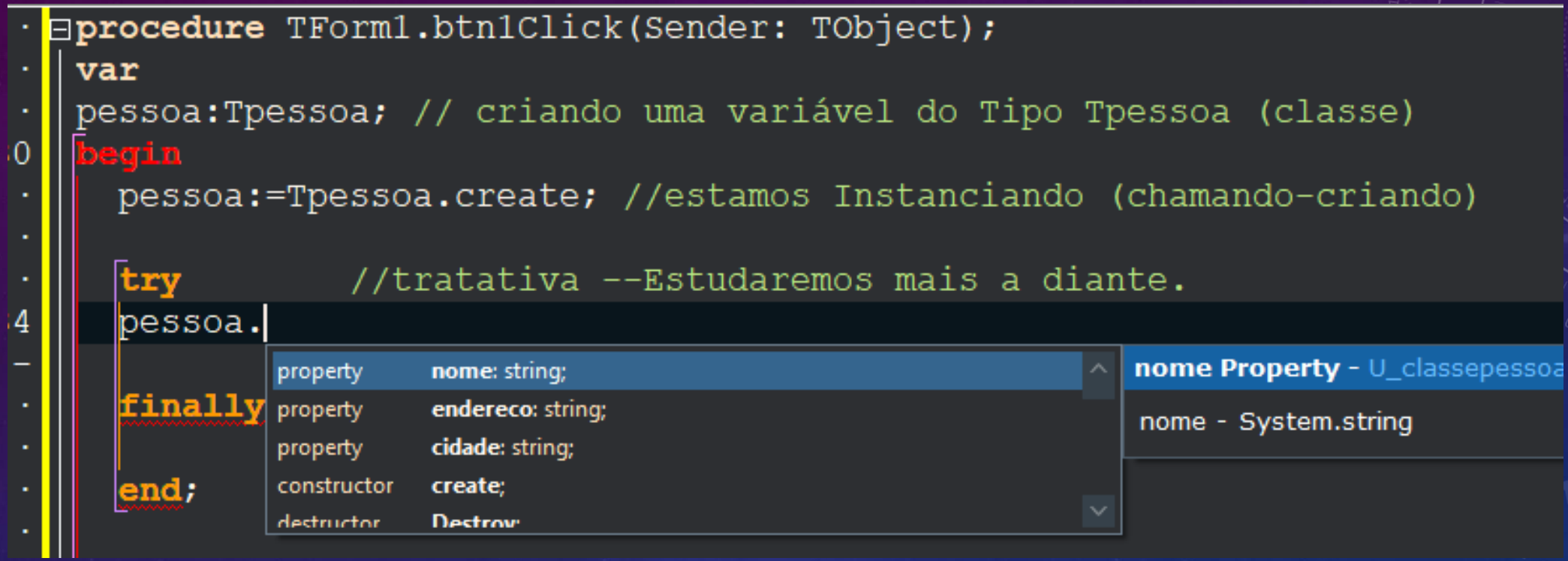
ISSO NÃO PODE DEIXAR DE ACONTECER...

```
procedure TForm1.btn1Click(Sender: TObject);
var
pessoa:Tpessoa; // criando uma variável do Tipo Tpessoa (classe)
begin
  pessoa:=Tpessoa.create; //estamos Instanciando (chamando-criando)

end;

end.
```

Instância é a concretização de uma **classe**.



VAMOS CHAMAR NOSSOS FIELDS (CAMPOS)



Observe que no momento que digitei o nome pessoa da classe Tpessoa, foi mostrado as Fields.
Nome: String
Endereco:String;
Cidade:String;
Tambem como Property assim feito na classe Tpessoa. Vamos que vamos.

DevTeam

OBSERVE O FINALLY – DA NOSSA AÇÃO

```
procedure TForm1.btn1Click(Sender: TObject);
 var
 pessoa:Tpessoa; // criando uma variável do Tipo Tpessoa (classe)
 begin
   pessoa:=Tpessoa.create; //estamos Instanciando (chamando-criando)

   try //tratativa --Estudaremos mais a diante.

       pessoa.nome:='Luis carlos';
       pessoa.endereco:='Rua São Francisco, 23';
       pessoa.cidade:='Caxias -MA';

   finally
       freea
   end;
                 procedure FreeAndNil(var Obj);


 end;
```

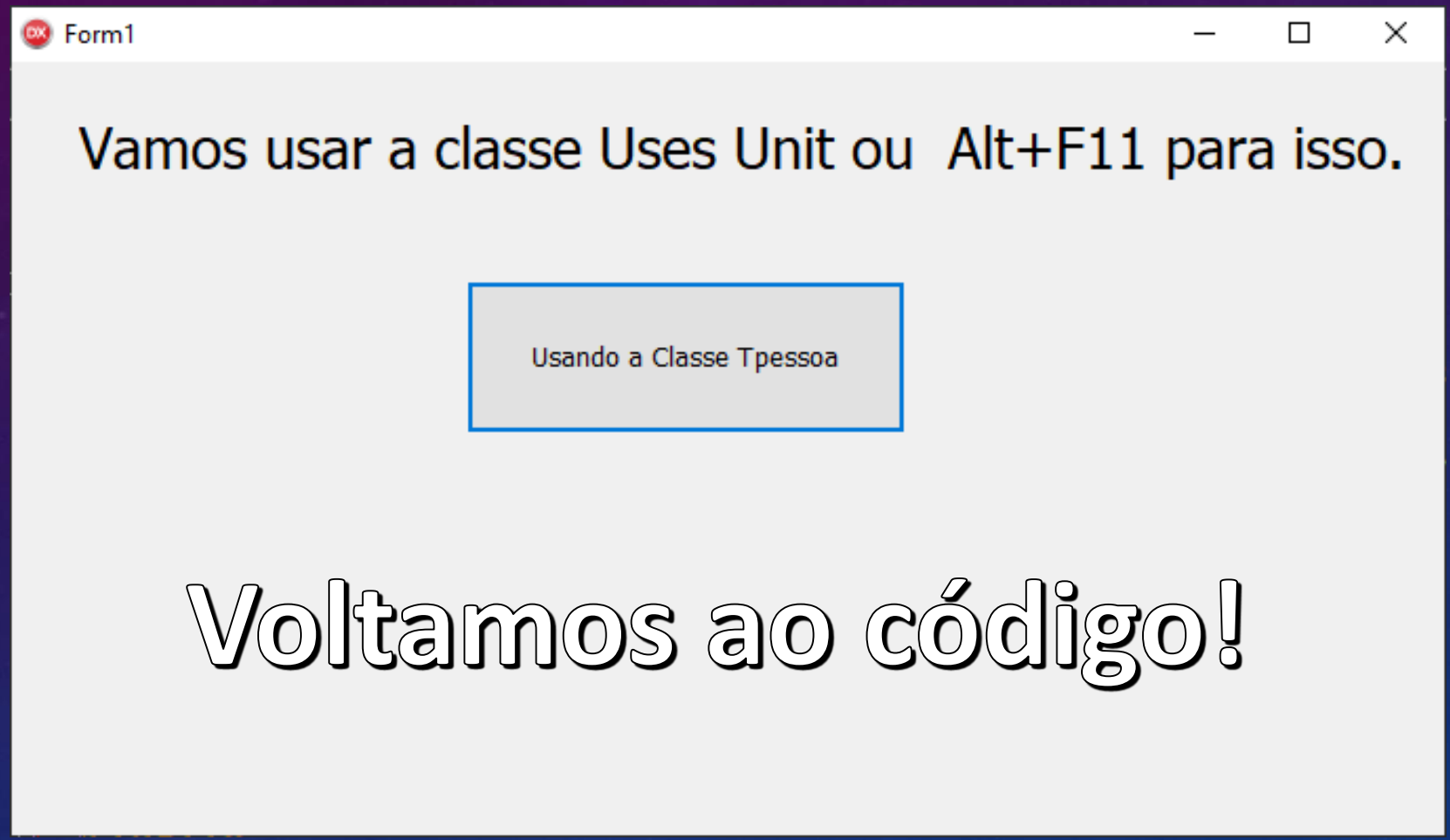
Observe que ele está pedindo a variável do Objeto Que é : pessoa

DevTeam

TUDO PRONTO! E AGORA?

```
procedure TForm1.btn1Click(Sender: TObject);
var
pessoa:Tpessoa; // criando uma variável do Tipo Tpessoa (classe)
begin
  pessoa:=Tpessoa.create; //estamos Instanciando (chamando-criando)

  try //tratativa --Estudaremos mais a diante.

     pessoa.nome:='Luis carlos';
     pessoa.endereco:='Rua São Francisco, 23';
     pessoa.cidade:='Caxias -MA';

  finally
     FreeAndNil(pessoa);
  end;
```

CLICO NO BOTÃO E NÃO ACONTECE NADA!!!
QUE COISA NÃO....

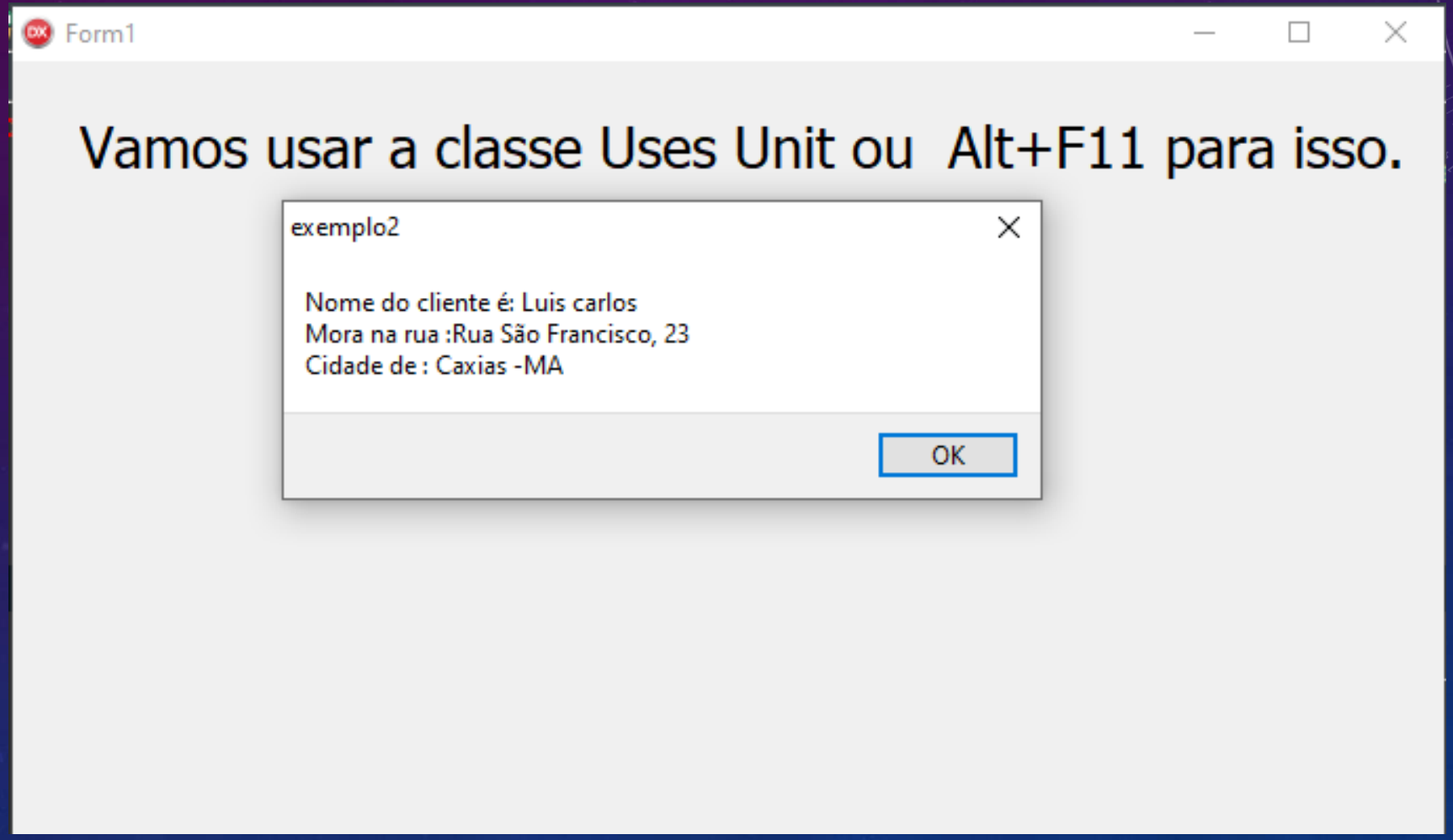


DevTeam

AGORA SIM!

```
begin
  pessoa:=Tpessoa.create; //estamos Instanciando (chamando-criando)

  try //tratativa --Estudaremos mais a diante.

     pessoa.nome:='Luis carlos';
     pessoa.endereco:='Rua São Francisco, 23';
     pessoa.cidade:='Caxias -MA';

     //dando um showmessagem para chamar a ação do botao...

     ShowMessage('Nome do cliente é: '+ pessoa.nome + #13+
                 'Mora na rua :'       + pessoa.endereco + #13+
                 'Cidade de : '        + pessoa.cidade);


  finally
     FreeAndNil(pessoa);
  end;
```

FUNCIONANDO COMO O PROPOSTO.



ATÉ A PRÓXIMA.....

DevTeam