

{*****}

Componente da Nota Fiscal Eletrônica

Unit de Exemplo de uso do componente na Assinatura através da DLL e do Type Library

Pré Requisitos para a utilização dessas rotinas no seu projeto...

Instalar a DLL do componente no seu Projeto e poder se utilizar das rotinas...

- 1.)copiar *NFE_dll.dll*, *NFE_dll.tlb*, *Regasm.exe*, *RegistraDLL.bat* para pasta do executável
- 2.)executar o arquivo de lote *RegistraDLL.bat* e registrando todas as funções da DLL
- 3.)Importar Type Library para criar a unit referente aos métodos da DLL
 - a.)no delphi: menu Project | Options | Import Type Library
 - b.)[Add..] encontre o arquivo *NFE_dll.tlb* (dentro da pasta do executável)
 - c.)CREATE Unit (Será gerado o arquivo *NFE_dll_TLB.pas*, adicione ele ao seu projeto);

O certificado digital deve estar instalado no Windows para que seja efetuado os métodos dessa unit.

*****}

```
unit frmAssinatura;
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ComCtrls, ComObj, Buttons, XMLDoc, xmldom, XMLIntf;

type
  TFormAssinatura = class(TForm)
    GroupBox3: TGroupBox;
    GroupBox4: TGroupBox;
    rtfDocto: TRichEdit;
    rtfAssinado: TRichEdit;
    OpenDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    btCertificado: TBitBtn;
```

```

edResultado: TEdit;
GroupBox1: TGroupBox;
rtfNome: TRichEdit;
btAbrir: TBitBtn;
GroupBox2: TGroupBox;
Label2: TLabel;
tpDoc: TComboBox;
btAssinar: TBitBtn;
btGravar: TBitBtn;
btFechar: TBitBtn;

procedure btAbrirXMLClick(Sender: TObject);
procedure btGravarClick(Sender: TObject);
procedure btAssinarClick(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure FormActivate(Sender: TObject);
procedure btFecharClick(Sender: TObject);

private
    { Private declarations }

public
    function pAssinarXML(pathDocXML, tipDocXML, titCertificado: WideString;
        out aResultado: WideString): Integer;
    { Public declarations }

end;

var
    FormAssinatura: TFormAssinatura;
    xmldoc, ref, nome, xmlassinado, mensagem: Widestring;

implementation

uses NFE_dll_TLB;

{$R *.dfm}

```

[G1] Comentário: Declarada a unit gerada pelo Delphi no import type library, é nela que estão as assinaturas dos métodos a serem utilizados nessa unit

```
function TFormAssinatura.pAssinarXML(pathDocXML, tipDocXML, titCertificado: WideString; out
aResultado: WideString): Integer;
```

[G2] Comentário: Método que permite um alto de nível de abstração pra isolar toda a parte complexa da assinatura

```
var
```

```
    cAssinarXML : _AssinarXML;
```

```
    strResultado : WideString;
```

```
    intResultado : Integer;
```

[G3] Comentário: Criando uma variável do tipo _AssinarXML interface implementada da variável

```
    /// AssinarXML : Assinatura Digital XML no padrão do Projeto NF-e
```

```
    ///
```

```
    ///
```

```
    /// Entradas:
```

```
    /// XMLString: string XML a ser assinada
```

```
    /// RefUri : Referência da URI a ser assinada (Ex. infNFe
```

```
    /// X509Cert : certificado digital a ser utilizado na assinatura digital
```

```
    ///
```

```
    /// Retornos:
```

```
    ///
```

```
    /// vResultado: código do resultado
```

```
    ///
```

```
    /// 0 - Assinatura realizada com sucesso
```

```
    /// 1 - Erro: Problema ao acessar o certificado digital - %exceção%
```

```
    /// 2 - Certificado digital inexistente para %nome%
```

```
    /// 3 - XML mal formado + exceção
```

```
    /// 4 - A tag de assinatura %RefUri% inexiste
```

```
    /// 5 - A tag de assinatura %RefUri% não é unica
```

```
    /// 6 - Erro Ao assinar o documento - ID deve ser string %RefUri(Atributo)%
```

```
    /// 7 - Erro: Ao assinar o documento - %exceção%
```

```
    ///
```

```
    ///
```

```
    /// vResultadoString : Literal da mensagem resultado
```

```
begin
```

```
    try
```

```
        cAssinarXML := CoAssinarXML.Create;
```

```
        cAssinarXML.Assinar(pathDocXML, tipDocXML, titCertificado, intResultado, strResultado);
```

```
aResultado := strResultado;

Result := intResultado;

finally
  cAssinarXML := nil;
end;
end;

procedure TFormAssinatura.btAbrilXMLClick(Sender: TObject);
var
  Fentrada: File;
  NumRead, i: integer;
  buf: char;
  S: string;

begin
  case tpDoc.ItemIndex of
    0: OpenFileDialog1.Filter := 'NF-e|*-nfe.xml|Arquivo XML|*.*|Todos Arquivos|*.*';
    1: OpenFileDialog1.Filter := 'Pedido de Cancelamento NF-e|*-ped-can.xml|Arquivo XML|*.*|Todos Arquivos|*.*';
    2: OpenFileDialog1.Filter := 'Pedido de Inutilização de número de NF-e|*-inu-inu.xml|Arquivo XML|*.*|Todos Arquivos|*.*';
  end;

  if OpenFileDialog1.Execute then
  begin
    AssignFile(Fentrada, OpenFileDialog1.FileName);
    Reset(Fentrada, 1);
    S := '';

    for i := 1 to FileSize(Fentrada) do
    begin
      Blockread(Fentrada, buf, 1, NumRead);
      S := S + buf;
    end;

    CloseFile(Fentrada);
```

```
    rtfDocto.Text := S;

    xmlDoc := S;

    edResultado.Text := '';

    xmlAssinado := '';

    rtfAssinado.Text := xmlDoc;

    GroupBox3.Visible := true;

    GroupBox4.Visible := false;

    btAssinar.enabled := true;

    btAssinar.setfocus;

end;

end;

procedure TFormAssinatura.btGravarClick(Sender: TObject);
var
    Fsaida: TextFile;
begin
    if rtfAssinado.Text <> '' then
    begin
        SaveDialog1.Title := 'Salvar a mensagem assinada';

        case tpDoc.ItemIndex of

            0: SaveDialog1.Filter := 'NF-e|*-nfe.xml|Arquivo XML|*.*|Todos Arquivos|*.*';

            1: SaveDialog1.Filter := 'Pedido de Cancelamento NF-e|*-ped-can.xml|Arquivo XML|*.*|Todos Arquivos|*.*';

            2: SaveDialog1.Filter := 'Pedido de Inutilização de número de NF-e|*-inu-inu.xml|Arquivo XML|*.*|Todos Arquivos|*.*';

        end;

        if SaveDialog1.Execute then
        begin
            AssignFile(Fsaida, SaveDialog1.FileName);

            Rewrite(Fsaida);

            Write(Fsaida, rtfAssinado.Text);

            CloseFile(Fsaida);

        end;

    end;

else
```

```

    MessageDlg( 'Não existe Documento assinado para gravar...', mtInformation, [mbOk], 0);
end;

procedure TFormAssinatura.btAssinarClick(Sender: TObject);
var
    i: integer;
    assXML: TextFile;
    CaminhoXML: String;

begin
    if xmlDoc <> '' then
    begin
        if rtfNome.Text <> '' then
        begin
            case tpDoc.ItemIndex of
                0: ref := 'infNFe';
                1: ref := 'infCanc';
                2: ref := 'infInut';
            end;

            nome := rtfNome.Text;

            CaminhoXML := 'C:\Assinando.xml';
            AssignFile(assXML, CaminhoXML);
            Rewrite(assXML);
            Write(assXML, rtfDocto.Text);
            CloseFile(assXML);

            i := pAssinarXML(CaminhoXML, ref, nome, mensagem);

            if i <> 0 then
                MessageDlg( 'Processo de assinatura falhou...', mtInformation, [mbOk], 0);

            GroupBox4.Visible := true;
            edResultado.Text := inttostr(i) + ' - ' + mensagem;

```

[G4] Comentário: Utilização da função pAssinar passando apenas os parâmetros:

CaminhoXML: Local físico do Arquivo a ser assinado e alterado;

Ref: Tipo de arquivo a ser assinado de acordo com a tag superior

Nome: String do titular do certificado utilizado na assinatura

Mensagem: Variável string que vai receber todo o retorno texto dessa função

```
    rtfAssinado.Text := xmlAssinado;

    btGravar.enabled := true;

    btGravar.setfocus;

    rtfAssinado.Lines.LoadFromFile(CaminhoXML);

    DeleteFile(CaminhoXML);

end

else

    MessageDlg( 'Nome do titular do Certificado não informado...', mtInformation, [mbOk],
0);

end

else

    MessageDlg( 'Documento XML para assinatura não informado...', mtInformation, [mbOk], 0);

end;

procedure TFormAssinatura.Button1Click(Sender: TObject);

var

    cBuscarCeritificado : _BuscarCertificado;

    smensagem: wideString;

begin

    cBuscarCeritificado := CoBuscarCertificado.Create;

    cBuscarCeritificado.BuscaNome(smensagem);

    rtfNome.Text := smensagem;

    btAbrir.Enabled := True;

    btAbrir.SetFocus;

    GroupBox2.Visible := True;

    tpDoc.Visible := True;

end;

procedure TFormAssinatura.FormActivate(Sender: TObject);

begin

    rtfNome.Text := '';

    rtfAssinado.Text := '';

    rtfDocto.Text := '';
```

```
GroupBox2.Visible := false;
GroupBox3.Visible := false;
GroupBox4.Visible := false;
btAssinar.enabled := false;
btAbrir.enabled := false;
btGravar.enabled := false;
end;

procedure TFormAssinatura.btFecharClick(Sender: TObject);
begin
    Close;
end;

end.
```