

MÓDULO 2 – CONTINUAÇÃO

Visto que ao apagar seções, chaves ou valores das chaves não geram erros, temos uma situação meio preocupante. Digo preocupante, porque não saberemos quão reais são os valores das chaves armazenadas e se elas também existem é claro. Uma outra situação preocupante é o tipo do usuário, geralmente o usuário comum não sabe o que é um arquivo ini ou para que ele serve. Quando um usuário comum abre um arquivo contendo uma série de parâmetros, um frio terrível corre por sua coluna e barriga. Porque falei do usuário e da condição de desaparecimento dos valores de um arquivo ini? Simples. O primeiro problema é que devemos criar uma estrutura para não ficar re-gravando os dados nas seções e a segunda é que o próprio usuário pode apagar acidentalmente ou intencionalmente o arquivo. Um outro ponto é importante e deve ser ressaltado, vamos dizer que quando a sua aplicação é executada pela primeira vez, ela cria um arquivo ini contendo uma série de chaves que armazenam as configurações de conexão com um banco de dados, de ambiente, de usuário e etc.

Podemos responder as duas primeiras questões apresentado duas funções – SectionExists e ValueExists; ambas são do tipo **boolean** (a primeira verifica se a seção existe e a segunda verifica se o valor da chave existe). Para a terceira questão, usaremos uma função que testa a existência do arquivo ini dentro da pasta WINDOWS, veja-a na seqüência:

```
function Existe_ArquiINI(vNomeDoIni: String): Boolean;  
var  
    vCaminho : String;  
    vLocal   : String;  
begin  
    SetLength(vCaminho, MAX_PATH);  
    GetWindowsDirectory(PChar(vCaminho), MAX_PATH);  
    vLocal := Copy(vCaminho,1,StrLen(PChar(vCaminho))) + '\\' + vNomeDoIni;  
  
    if FileExists(vLocal) = True then  
        Result := True  
    Else  
        Result := False;  
End;
```

EXEMPLO 5 – EXISTE MEU ARQUIVO INI?

Usando a função apresentada acima, vamos criar o nosso exercício onde iremos testar a existência do arquivo *Configuracoes.ini*, para isso execute os passos seguintes:

1. Crie uma nova aplicação;
2. Digite a função Existe_ArquiINI;
3. Coloque um componente Label no seu form;
4. No evento **OnShow** do form, chame a função e passe como parâmetro o nome *Configuracoes.ini*.

Todos os Direitos Reservados – 2006 – Esta apostila pode ser copiada e distribuída gratuitamente. É proibida a sua venda sem autorização expressa de seu autor

Contatos: Exio – E-Mail: bitzero_000@yahoo.com.br

Exemplo:

```
if Existe_ArquiINI('Configuracoes.ini') = True then
    Label1.Caption := 'O Arquivo INI já existe!!'
else
    Label1.Caption := 'O Arquivo INI não existe no sistema!';
```

5. Compile e execute a aplicação;

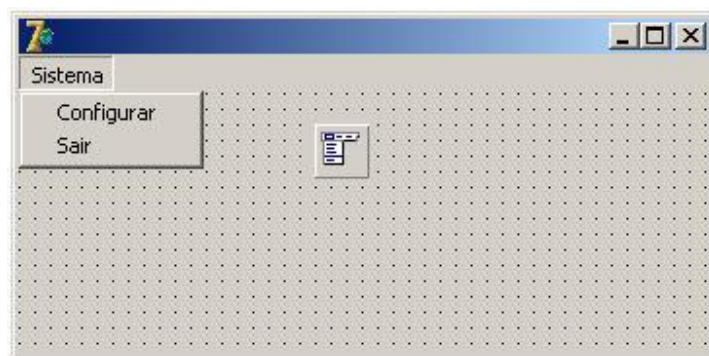
Se nenhum erro foi reportado durante a compilação do nosso pequeno programa, provavelmente o label do seu programa retornou a mensagem de que o arquivo ini **não existe** no sistema. Lembre-se que eu pedi para renomear ou excluir esse arquivo no **EXERCÍCIO 4**. Para ver a resposta de que o arquivo ini existe, você só precisa renomear ou recuperar o arquivo enviado para a lixeira. Você também pode mudar o nome do arquivo ini de **Configuracoes.ini** para por exemplo **Win.ini**, ou então, o nome de qualquer outro arquivo ini que você desejar.

EXEMPLO 6 – EXEMPLO DE CONFIGURAÇÃO REAL

Em todos os sistemas que desenvolvo, sempre uso muito os arquivos ini. Sou realmente um apaixonado pelos arquivos ini. Nesse exemplo, vou demonstrar o uso do arquivo ini com uma estrutura que lhe dará muitas idéias interessantes. Mãos a obra!!!

1. Crie uma nova aplicação;
2. Deixe esse Form com as seguintes configurações nas propriedades:
 - a. Caption = ''
 - b. Font = Tahoma
 - c. Name = FoPrinc
 - d. Position = poScreenCenter
3. Coloque um componente MainMenu e insira dois itens: Configurar e Sair;

Deixe-o mais ou menos parecido com a figura abaixo:



OBSERVAÇÃO: O TAMANHO DO FORM PRINCIPAL SÓ FOI REDUZIDO COM A FINALIDADE DE NÃO CONSUMIR UM ESPAÇO MUITO GRANDE DESSA APOSTILA. DEIXE-O DO TAMANHO QUE MAIS LHE AGRADAR.

Todos os Direitos Reservados – 2006 – Esta apostila pode ser copiada e distribuída gratuitamente. É proibida a sua venda sem autorização expressa de seu autor

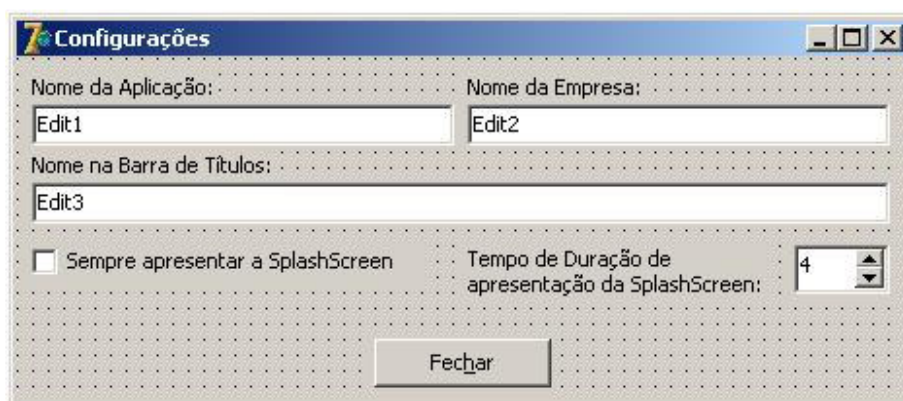
Contatos: Exio – E-Mail: bitzero_000@yahoo.com.br

4. Salve o Form com o nome de UPrinc;
5. Insira um novo Form (seria o Form2), configure as propriedades:
 - a. Caption = 'Configurações'
 - b. Font = Tahoma
 - c. Name = FoConf
 - d. Position = poScreenCenter
6. No form de Configurações, insira os seguintes componentes:
 - a. 3 Edit's (Palleta Standard)
 - b. 1 CheckBox (Palleta Standard)
 - c. 1 SpinEdit (Palleta Samples)
 - d. 1 Button com a propriedade Caption = 'Fechar'

Para o componente SpinEdit, configure as seguintes propriedades:

MinValue = 1
MaxValue = 60
Value = 4

Deixe o form de configurações mais o menos próximo à imagem abaixo:



7. Salve o form de Configurações com o nome de UConf;
8. Inserir um novo form (seria o Form3) e configure as propriedades:
 - a. Caption = ''
 - b. Font = Tahoma
 - c. Name = FoSplash
 - d. Position= poScreenCenter

OBSERVAÇÃO: SE FOR DO SEU DESEJO, COLOQUE UMA FIGURA PARA FAZER PARECER REALMENTE UMA SPLASH SCREEN. EU NÃO FAREI ISSO PARA NÃO AUMENTAR DESNECESSÁRIAMENTE O TAMANHO DO ARQUIVO.

9. No form de SplashScreen, acrescente um componente Label com as seguintes configurações para sua propriedade:
 - a. Caption = ''
 - b. AutoSize = False
 - c. Font Size = 20
 - d. Alignment = taCenter

Deixe-o mais ou menos parecido com a figura abaixo:



Veja que eu coloque um Componente Panel só para dizer que naquele espaço ali vai uma figura. Já o Label um está centralizado no painel e com a configuração de alinhamento para centralizado. É necessário que, dependendo da fonte escolhida e do tamanho definido, você deverá ter uma idéia do número máximo de caracteres que poderão ser definidos pelo usuário. Vamos continuar então!!!

10. Salve o form SplashScreen com o nome de USplash;

O que fizemos até agora foi só inserir, padronizar e salvar os form's que serão utilizados em nossa aplicação. Geralmente existirá um número bem maior de parâmetros que serão armazenados em seus arquivos de inicialização.

11. Insira uma nova **UNIT** – não é um FORM;
12. Salve-a com o nome de UGenerica.

Vamos criar essa unit com a finalidade principal de deixar todos os procedimentos e funções relativas à configuração em um único local. Isso facilita o processo de manutenção e deixa o código separado dos códigos da aplicação em si.

13. Adicione na Unit UGenerica a função Existe_ArquiINI abaixo da seção **IMPLEMENTATION**; o cabeçalho da função deve ser declarado abaixo da seção **INTERFACE**.
14. Abaixo de **INTERFACE** declare a Clausula **USES System, Windows**; o corpo inicial da sua unit deve ficar mais ou menos próximo do texto apresentado abaixo:

```
unit UGenerica;

interface

uses SysUtils, Windows;

function Existe_ArquiINI(vNomeDoIni: String): Boolean;

implementation

function Existe_ArquiINI(vNomeDoIni: String): Boolean;
var
  vCaminho : String;
```

```
vLocal : String;...
```

15. Vamos criar uma procedure que será responsável por definir os valores iniciais para os parâmetros. Vamos dizer que seria a configuração padrão da aplicação. Para criar a procedure faça:

a. Adicione na clausula **USES** a unit **Inifiles**.

b. Digite a procedure CriarValorDefault;

```
procedure CriarValorDefault(vNomeDoArquivoIni: String);
var
    vMeuIni : TIniFile;
begin
    vMeuIni := TIniFile.Create(vNomeDoArquivoIni);
    try
        vMeuIni.WriteString('APLICACAO','NOME_APLIC','Exio Systems');
        vMeuIni.WriteString('APLICACAO','NOME_EMPRE','ExiOixE');
        vMeuIni.WriteString('APLICACAO','CAPTION','Teste de Arq. Ini');
        vMeuIni.WriteBool('ENTRADA','SPLASH',TRUE);
        vMeuIni.WriteInteger('APLICACAO','TEMPO_SEG',4);
    finally
        vMeuIni.Free;
    end;
end;
```

OBSERVAÇÃO: COPIE O CABEÇALHO DA PROCEDURE E COLE-O ABAIXO DO CABEÇALHO DA FUNÇÃO EXISTE_ARQUIINI. FICANDO PARECIDO COM O TEXTO ABAIXO. TODAS AS VEZES QUE VOCÊ ACRESCENTAR UMA PROCEDURE OU FUNÇÃO, DEVERÁ EXECUTAR ESSA TAREFA.

```
function Existe_ArquiINI(vNomeDoIni: String): Boolean;
procedure CriarValorDefault(vNomeDoArquivoIni: String);
```

16. Vamos criar uma procedure que irá gravar os valores que serão definidos pelo usuário na janela de Configurações, para isso faça:

```
procedure GravarNoIni(vNomeDoArquivoIni, vSt1, vSt2, vSt3: String;
    vB1: Boolean; vInt1: Integer);
var
    vMeuIni : TIniFile;
begin
    vMeuIni := TIniFile.Create(vNomeDoArquivoIni);
    try
        vMeuIni.WriteString('APLICACAO','NOME_APLIC',vSt1);
        vMeuIni.WriteString('APLICACAO','NOME_EMPRE',vSt2);
        vMeuIni.WriteString('APLICACAO','CAPTION',vSt3);
        vMeuIni.WriteBool('ENTRADA','SPLASH',vB1);
        vMeuIni.WriteInteger('APLICACAO','TEMPO_SEG',vInt1);
    finally
        vMeuIni.Free;
    end;
end;
```

17. Vamos criar a **função** que fará a leitura de todos os dados armazenados nas seções e chaves do nosso arquivo ini. Muitas vezes há a necessidade de recuperarmos valores de uma chave somente, em outras ocasiões você deverá recuperar valores de seções inteiras.

Vamos primeiro nos preocupar em recuperar somente o valor da chave desejada, então criamos a função dessa forma:

```
function LeValorIndividual(vNomeDoArquivoIni: String;
  vNumero: SHORT): Variant;
var
  vMeuIni : TiniFile;
begin
  vMeuIni := TiniFile.Create(vNomeDoArquivoIni);
try
Case vNumero of
  0:Result := vMeuIni.ReadString('APLICACAO','NOME_APLIC','Exio Systems');
  1:Result := vMeuIni.ReadString('APLICACAO','NOME_EMPRE','ExioIxE');
  2:Result := vMeuIni.ReadString('APLICACAO','CAPTION','Teste de Arq.Ini');
  3:Result := vMeuIni.ReadBool('ENTRADA','SPLASH',TRUE);
  4:Result := vMeuIni.ReadInteger('APLICACAO','TEMPO_SEG',4);
end;
finally
  vMeuIni.Free;
end;
end;
```

OBSERVAÇÃO: NÃO ESQUEÇA QUE NESSA FUNÇÃO SE O VALOR NÃO FOR LOCALIZADO OU A CHAVE AINDA NÃO EXISTIR, O RESULTADO DA FUNÇÃO SERÁ O VALOR COLOCADO NO ÚLTIMO PARÂMETRO.

18. Para finalizarmos o nosso trabalho na unit **UGenerica**, vamos declarar uma **Constante**; a declaração dessa constante deve ficar entre a clausula **Implementation** e **Interface** (depois dos nomes das funções). Ficará mais ou menos assim:

```
interface

uses SysUtils, Windows, IniFiles;

function Existe_ArquiINI(vNomeDoIni: String): Boolean;
procedure CriarValorDefault(vNomeDoArquivoIni: String);
procedure GravarNoIni(vNomeDoArquivoIni, vSt1, vSt2, vSt3: String;
  vB11: Boolean; vInt1: Integer);
function LeValorIndividual(vNomeDoArquivoIni: String;
  vNumero: SHORT): Variant;

Const aplic = 'Aplicacao.ini';

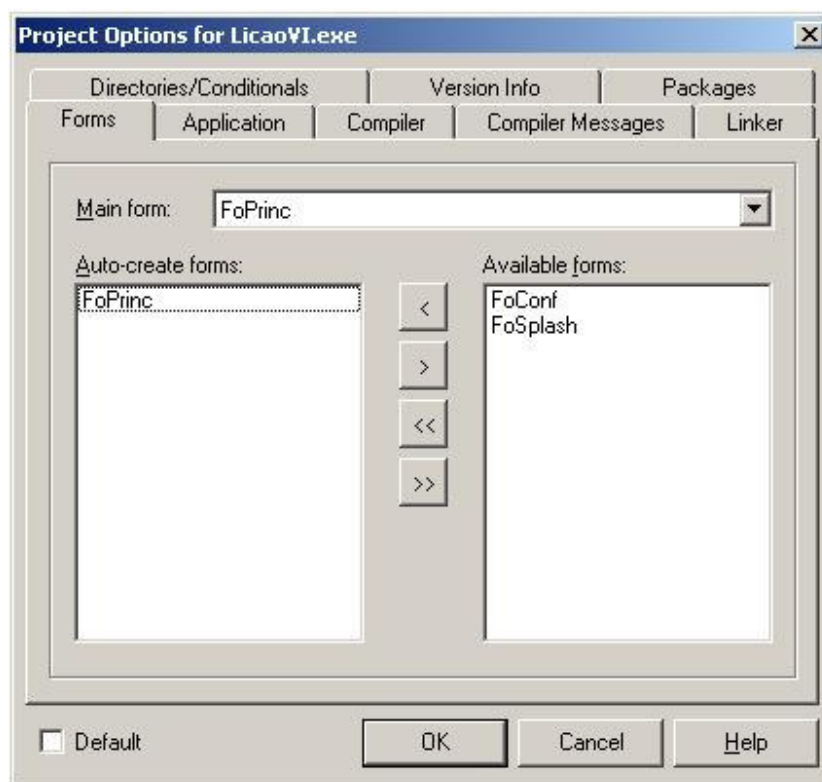
implementation

function Existe_ArquiINI(vNomeDoIni: String): Boolean;
```

19. Salve tudo o que foi feito até agora e vamos partir para outros pontos de nosso sistema.
20. Vamos agora pedir que o form Principal de nossa aplicação use a unit UGenerica. Para isso clicamos no FoPrinc dando-lhe o foco. Vamos com o mouse no **Menu do Delphi** e clicamos no Item **File**; no menu

que se expande clique sobre o item **Use Unit...** ou pressione a combinação de teclas **ALT + F11**. Na janela Use Unit clique sobre o nome **UGenerica** e pressione o botão **OK**.

21. Vamos agora fazer uma pequena configuração. No **Menu do Delphi** clique no Item **Project**. No menu que se expande, clique sobre o último Item do menu, o Item **Options** ou pressione a combinação de teclas **SHIFT + CTRL + F11**. A janela Options se abre, clique na aba **Forms**. Nessa aba você verá os três forms que já criamos até agora, são eles: FoPrinc, FoConf e FoSplash. Clique com o mouse sobre o form FoConf e pressione o botão **>**; isso faz com que o nome do form mude para a janela **Available forms**. Execute o mesmo processo com o form Splash, deixando somente o FoPrinc no lado esquerdo da janela. Pressione o botão **OK** para confirmar as alterações e fechar a janela options. Veja a figura abaixo:



22. Vamos voltar para o nosso form Principal, no FoPrinc clique no evento **onShow** e vamos usar as funções que criamos em UGenerica. Digite os seguintes comandos:

```
if (Existe_ArquiINI(aplic) = False) then
begin
  CriarValorDefault(aplic); //Vai primeiro criar o arquivo
  FoPrinc.Caption := LeValorIndividual(aplic,2);
  Application.Title := LeValorIndividual(aplic,0);
end
```

Todos os Direitos Reservados - 2006 - Esta apostila pode ser copiada e distribuída gratuitamente. É proibida a sua venda sem autorização expressa de seu autor

Contatos: Exio - E-Mail: bitzero_000@yahoo.com.br

```

else
begin
  FoPrinc.Caption := LeValorIndividual(aplic,2);
  Application.Title := LeValorIndividual(aplic,0);
end;

```

23. No form FoPrinc, vamos digitar o comando para o evento **onClick** quando o usuário clicar sobre o item de menu **Sair**. Para isso, clique com o mouse sobre o item Sair e digite o seguinte comando:

```

ShowMessage('A aplicação será encerrada agora!');
Close;

```

24. Compile e execute a aplicação. Vamos ver dois efeitos interessantes. O primeiro é que deixamos o Caption com um padrão inicial e o segundo é que alteramos a propriedade **Title** da aplicação. Essa propriedade é usada em algumas caixas de mensagens e alguns outros eventos das aplicações. Veja o botão com o nome na Barra de Tarefas do Windows. Clique sobre o item de menu **Sair** de nossa aplicação e veja o título da caixa de mensagens – não é o nome do executável e sim o nome que definimos.

25. Agora vamos colocar a janela SplashScreen para trabalhar. Novamente no nosso form FoPrinc, clique sobre ele para que tenha o foco e vamos pedir para que ele use a Unit USplash. Pressione o item **File** do **Menu do Delphi** e no menu que se expande clique sobre o item **Use Unit...**, ou então pressione a combinação de teclas **ALT + F11**. A janela Use Unit se abre. Clique no nome da unit USplash e pressione o botão **OK**.

26. Vamos finalizar os comandos digitados no evento **OnShow** do form FoPrinc. Vamos alterar o conteúdo anterior para (digite somente o conteúdo que está em sublinhado e negrito):

```

if (Existe_ArquiINI(aplic) = False) then
begin
  CriarValorDefault(aplic); //Vai primeiro criar o arquivo
  FoPrinc.Caption := LeValorIndividual(aplic,2);
  Application.Title := LeValorIndividual(aplic,0);

  if (LeValorIndividual(Aplic,3) = True) then
  begin
    FoSplash := TFoSplash.Create(Self);
    FoSplash.Label1.Caption := LeValorIndividual(Aplic,1);
    FoSplash.Show;
    FoSplash.Refresh;
    Sleep(1000 * LeValorIndividual(aplic,4));
    FoSplash.Close;
    FoSplash.Free;
  end;

end
else
begin
  FoPrinc.Caption := LeValorIndividual(aplic,2);
  Application.Title := LeValorIndividual(aplic,0);

```

```

if (LeValorIndividual(Aplic,3) = True) then

```

Todos os Direitos Reservados – 2006 – Esta apostila pode ser copiada e distribuída gratuitamente. É proibida a sua venda sem autorização expressa de seu autor

Contatos: Exio – E-Mail: bitzero_000@yahoo.com.br


```

begin
  FoSplash := TFoSplash.Create(Self);
  FoSplash.Label1.Caption := LeValorIndividual(Aplic,1);
  FoSplash.Show;
  FoSplash.Refresh;
  Sleep(1000 * LeValorIndividual(aplic,4));
  FoSplash.Close;
  FoSplash.Free;
end;
end;

```

É claro que podemos melhorar a lógica desse código, porém o meu objetivo é deixar as coisas mais claras o possível para facilitar a vida de quem ainda não entende muito de programação e facilitando para aqueles que já tem uma certa “manha”. Já trabalhamos com os Forms Principal e Splash Screen, vamos finalizar o nosso trabalho com o form Configurações. Adelante hombres e mujeres!!!

27. Abra o form FoConf (Configurações). No evento **OnShow** do form vamos acrescentar o seguinte código:

```

Edit1.Text := LeValorIndividual(aplic,0);
Edit2.Text := LeValorIndividual(aplic,1);
Edit3.Text := LeValorIndividual(aplic,2);
CheckBox1.Checked := LeValorIndividual(aplic,3);
SpinEdit1.Value := LeValorIndividual(aplic,4);

```

Você pode estar querendo se perguntar porque não testamos se o arquivo ini já existe aqui na parte das configurações. É bem simples de responder. Porque o teste já foi feito na inicialização do Sistema, assim sendo, não há a necessidade de refazer o teste.

28. No evento **onClose** do Form FoConf vamos acrescentar o seguinte código:

```

GravarNoIni(aplic,Edit1.Text,
            Edit2.Text,
            Edit3.Text,
            CheckBox1.Checked,
            SpinEdit1.Value);

FoConf.Release; //Libera o form da memória;

```

29. Vamos acrescentar o comando **Close**; no Botão **Fechar** do form FoConf;

Salve todo o seu projeto. Compile-o para verificar se não existem erros de digitação ou sintaxe.

30. Vamos voltar para o form FoPrinc e pedir que ele use o form FoConf. Assim, quando o usuário clicar sobre o item de menu

Configurar o form será chamado. Bem você já sabe como fazer isso porque já demonstrei duas vezes, então... (Dica: **ALT + F11**).

31. No evento **OnClick** do item de menu **Cadastrar**, digite o seguinte código:

```
FoConf := TFoConf.Create(Self);  
FoConf.ShowModal;
```

32. Compile e rode a sua aplicação. Teste o comando de menu **Configurar** e altere os dados no form de acordo com seus desejos.

Veja que trabalhar com arquivos INI não exige um alto grau de conhecimento. Na verdade, a única coisa que exige é cuidado dependendo do que será ou é feito.