

## Para fazer protetor de tela com o Delphi

Para o pessoal que queria saber...

- A) No .Dpr ponha {\$D SCRNSAVE } depois do uses
- B) No Form principal nao ponha borda ou icone. No metodo Activate ponha left e top como 0 e o Windowstate como wsMaximize.
- C) no form.Create ponha application.OnMessage para um metodo que controle a desativacao do screen saver. Ponha tb o application.OnIdle para "rodar" o dito cujo...
- D) Tb no Form.Create teste a linha de comando para /c ou /s. Estes parametros dizem o que e' para fazer (/c configura)
- E) Compile e renomeie o .exe p/ .scr, move para o directorio do windows e...

## Linkar um OBJ ao executável

Primeiro voce deve "linkar" o OBJ ao seu executavel. No Delphi , isto eh feito com a diretriz de compilacao \$L. Fica, na sua unit principal, assim:  
{ \$L MyObject.OBJ }

Incluindo as chaves.

Logo depois, voce deve declarar a funcao contida em MyObject.OBJ da forma usual. Voce precisara conhecer os parametros usados pela mesma, bem como o tipo e a ordem em que sao passados. Voce deve incluir tambem a diretriz PASCAL ou CDECL. Sugiro tentar primeiro com PASCAL. Ficaria assim (na secao implementation:

```
function (Parametro1 : TipoDoParametro1, Parametro2 : TipoDoParametro2):  
TipoDoRetorno; pascal;
```

se nao der certo, tente:

```
function (Parametro1 : TipoDoParametro1, Parametro2 : TipoDoParametro2):  
TipoDoRetorno; cdecl;
```

caso nao seja uma funcao e sim uma procedure, tente

```
procedure (Parametro1 : TipoDoParametro1, Parametro2 : TipoDoParametro2);  
pascal;  
ou  
procedure (Parametro1 : TipoDoParametro1, Parametro2 : TipoDoParametro2);  
cdecl;
```

Se voce nao sabe quais os parametros usados pela funcao/procedure, uma solucao seria linkar o seu OBJ num programa qualquer e disassembla-lo. Ai pelo menos voce sabera a quantidade e o tipo de cada parametro. De qualquer forma, para saber para que serve cada um, tera que ser na tentativa e erro...a nao ser que voce tambem tenha paciencia para analisar o codigo disassemblado.

OBS: Se o seu OBJ não estiver num formato reconhecível pelo LINK do Delphi (um formato similar ao COFF), você pode tentar outros Linkers, e criar uma dll. Existem vários linkers gratuitos, que reconhecem vários formatos (exemplos, são lclink, djlink, walk2link e o próprio linker da microsoft...também gratuito).

### Alterar LOCAL SHARE via programação

Olha Junior no WIN 95 você pode alterar diretamente a chave do registro que seta esta opção. Fica em HKEY\_LOCAL\_MACHINE > Software > ... LOCAL SHARE "TRUE" (Pesquise com o regedit). Já no WIN 3.xxx eu também gostaria de saber. Não encontrei referências de como fazer esta alteração utilizando a API do BDE.

### Verificando se o Delphi está aberto

Proteja aquele aplicativo ou objeto que vc desenvolveu com esta rotina que identifica se o usuário está com o Delphi aberto (disponibiliza) ou fechado (trava a execução).

Bom proveito !

```
Function TForm1.JanelaExiste(Classe,Janela:String) :Boolean;
var
  PClasse,PJanela : array[0..79] of char;
begin
  if Classe = " then
    PClasse[0] := #0
  else
    StrPCopy(PClasse,Classe);
  if Janela = " then
    PJanela[0] := #0
  else
    StrPCopy(PJanela,Janela);
  if FindWindow(PClasse,PJanela) <> 0 then
    result := true
  else
    Result := false;
end;
```

```
Function TForm1.DelphiCarregado : Boolean;
begin
  Result := False;
  if JanelaExiste('TPropertyInspector','Object Inspector') then
    result := True
end;
```

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  if DelphiCarregado then
    showmessage('Delphi está ativo, bom menino!')
  else
    begin
      Showmessage('Vc não poderá utilizar esta aplicação! Mau garoto!');
```

```
application.terminate;  
end;  
end;
```

### Criando formulários

Qual a melhor maneira de criar forms em tempo de execucao:

- a) Application.CreateForm(TfmClientes, fmClientes);  
Cria o Form; o proprietário é a aplicação.
- b) fmClientes := TForm.Create(self);  
Cria o Form; o proprietário é ele mesmo.
- c) fmClientes := TForm.Create(Application);  
Cria o Form; o proprietário é a aplicação.
- d) fmClientes := TForm.Create(nil);  
Cria o Form; teoricamente sem proprietário; na prática é a aplicação.
- e) fmClientes := TfmClientes.Create(self);  
Cria o Form; o proprietário é ele mesmo.
- f) fmClientes := TfmClientes.Create(Application);  
Cria o Form; o proprietário é a aplicação.
- g) fmClientes := TfmClientes.Create(nil);  
Cria o Form; teoricamente sem proprietário; na prática é a aplicação.

Poderiam me informar a diferenca entre elas?

Quando você cria um Form dinamicamente:

1. se criar através de CreateForm, que é um método de TApplication, você passa como parâmetro a instância da classe e o nome do seu objeto (TfmClientes, fmClientes);
2. se criar através de Create - método de TForm, entre outros - você passa como parâmetro o proprietário do componente criado (no caso o Form).
  - 2.1 se o proprietário for a aplicação, o Form só será destruído quando você finalizar o aplicativo ou se você declarar Free ou Destroy no seu programa; (casos a, c, d, f, g);
  - 2.2 se o proprietário for ele mesmo (self), o form terá que ser destruído por você;
  - 2.3 se você criar, por exemplo, Form2 e passar como proprietário Form1; no momento em que Form1 for destruído, Form2 também o será.

Quanto à melhor maneira, depende de como você quer controlar a aplicação, mas leve em conta que enquanto um objeto não é destruído, ele está na memória.

### Criando alias via programação

Se for para Paradox ...

```
Session.AddStandardAlias('SeuAlias', edtPath.text, 'Paradox');  
Session.SaveConfigFile;
```

## Desabilitar acesso a windows

Ai vai um código que peguei no site da Borland que trava as teclas (Ctrl+Alt+Del),(Alt+Tab), (Ctrl+Esc)

```
var
  OldValue : LongBool;
begin
  {liga a trava}
  SystemParametersInfo(97, Word(True), @OldValue, 0);
  {desliga a trava}
  SystemParametersInfo(97, Word(False), @OldValue, 0);
end;
```

## Splash Screen

```
form2:=tform2.create(application);
form2.show;
form2.update;
.
.
.
form2.hide;
form2.free;
Application.Run;
```

Obs: apagar a primeira linha, 'Application.Initialize'.

## Para saber somente o path da aplicação

```
ExtractFilePath( Application.ExeName )
```

## Como saber se o aplicativo já foi aberto

Insira o código abaixo dentro do arquivo .DPR de sua aplicação

```
{ $R *.RES }
begin
  Application.Title := "";
  Application.HelpFile := "";
  if HPrevInst = 0 then
  begin
    F_Splash := TF_Splash.create(Application);
    F_Splash.Show;
    Application.CreateForm(TF_Menu, F_Menu);
    Application.CreateForm(TF_Error, F_Error);
    Application.CreateForm(TF_Form1, F_Form1);
    Application.CreateForm(TF_Form2, F_Form2j);
    Application.Run;
  end
  else
    messagedlg('O sistema já foi inicializado!', mtInformation, [mbOk], 0);
end.
```

### Impressão com o TPrinter ( Via gerenciador de impressão)

```
procedure TForm1.BitBtn1Click(Sender: TObject);
var
  Linha:integer;
  Tamanho:integer;
  Coluna:integer;
begin
  Printer.Orientation := poLandscape;
  Printer.BeginDoc;
  Printer.Canvas.Pen.Width := 5;
  Printer.Canvas.Font.Name := 'Times New Roman';
  Printer.Canvas.Font.Size := 10;
  Linha := 20;
  Coluna:= 20;
  Tamanho := Printer.Canvas.TextWidth('a');
  Table1.First;
  while not Table1.Eof do
  begin
    if Linha = 20 then
    begin
      Coluna := 20;
      Printer.Canvas.TextOut(0,Linha,'Relação de Clientes');
      Linha := Linha - Printer.Canvas.Font.Height + 5 ;
      Printer.Canvas.TextOut(Coluna,Linha,'Cod');
      Coluna:= Coluna + (Tamanho * 5 );
      Printer.Canvas.TextOut(Coluna,Linha,'Nome');
      Coluna:= Coluna + (Tamanho * 30);
      Printer.Canvas.TextOut(Coluna,Linha,'Endereço');
      Coluna:= Coluna + (Tamanho * 30);
      Linha := Linha - Printer.Canvas.Font.Height + 5 ;
    end;
    Coluna := 20 ;
    Printer.Canvas.TextOut(Coluna,Linha,Table1.FieldName('Codigo').AsString);
    Coluna:= Coluna + (Tamanho * 5 );
    Printer.Canvas.TextOut(Coluna,Linha,Table1.FieldName('Nome').AsString);
    Coluna:= Coluna + (Tamanho * 30);
    Printer.Canvas.TextOut(Coluna,Linha,Table1.FieldName('End').AsString);
    Coluna:= Coluna + (Tamanho * 30);
    Linha := Linha - Printer.Canvas.Font.Height + 5 ;
    Table1.Next;
    if Linha > Printer.PageHeight-20 then
    Begin
      Printer.NewPage;
      Linha := 20;
    end;
  end;
  Printer.EndDoc;
end;
```

### Impressão direto para impressora

```
procedure TForm1.Button1Click(Sender: TObject);
var
```

```

F : TextFile;
i : integer;
begin
AssignFile(F,'LPT1');
Rewrite(F);
i := 0;
Writeln(F,'Teste de impressao - Linha 0');
Writeln(F,'Teste de impressao - Linha 1');
Writeln(F,#27#15+'Teste de Impressão - Linha 2');
Writeln(F,'Teste de impressao - Linha 3');
Writeln(F,#27#18+'Teste de Impressão - Linha 4');
Writeln(F,'Teste de impressao - Linha 5');
Writeln(F,#12); // Ejeta a página
CloseFile(F);
end;

```

### Definir o tamanho do papel em TPrinter

Esta procedure configura o tamanho do papel em Run-Time para ser utilizado com o objeto TPrinter; Esta procedure deve ser chamada antes de aplicar o método Printer.BeginDoc.

```

procedure TForm1.SetPrinterPage(Width, Height : LongInt);
var
  Device : array[0..255] of char;
  Driver : array[0..255] of char;
  Port : array[0..255] of char;
  hDMode : THandle;
  PDMode : PDEVMODE;
begin
  Printer.GetPrinter(Device, Driver, Port, hDMode);
  If hDMode <> 0 then
  begin
    pDMode := GlobalLock( hDMode );
    If pDMode <> nil then
    begin
      pDMode^.dmPaperSize := DMPAPER_USER;
      pDMode^.dmPaperWidth := Width;
      pDMode^.dmPaperLength := Height;
      pDMode^.dmFields := pDMode^.dmFields or DM_PAPERSIZE;
      GlobalUnlock( hDMode );
    end;
  end;
end;

```

### Como criar Forms em tempo de execução

Para você economizar memória, pode-se criar os forms de sua aplicação somente no momento da execução. Na criação do Form você define se ele é MODAL ou NÃO MODAL. Para Isso observe os seguintes códigos:

MODAL - Mostra form em modo exclusivo

```

procedure TForm1.Button1Click(Sender: TObject);
begin
Application.CreateForm(TForm2, Form2); {Carrega form na memória}

```

```
Form2.ShowModal; {Mostra form em modo exclusivo}
Form2.Free; {Libera Memória}
end;
```

NÃO MODAL - Mostra form em modo não exclusivo

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Application.CreateForm(TForm2, Form2); {Carrega form na memória}
Form2.ShowModal; {Mostra form em modo exclusivo}
end;
```

No evento OnClose do Form2 coloque o seguinte código.

```
procedure TForm2.FormClose (Sender: TObject; var Action : TCloseAction);
begin
Action:= caFree;
end;
```

Aliado a este código, deve alterar no delphi, no menu Options, opção Project. Mudando os forms a serem criados dinamicamente da coluna Auto-Create Forms para Avaliable Forms.

#### [Adaptando para resoluções de video diferentes?](#)

Sempre que procurei algo sobre esse tema, ia para no Tip da Borland #2861, que é a mesma informação do arquivo de help da Loyd's. Esse texto também aparece nos bancos de dados da Q&A. Eu suponho que essa seja a informação definitiva. Encontrei uma informação que não foi mencionada aqui. Pela lista de correios do Delphi-Talk havia mensagens de Brien King e Michael Novak que discutiam esse assunto.

\*\*\*

LOYD'S TIPS:

Resolução de Vídeo:

Quando criamos formulários, às vezes é útil escrever um código para que a tela e todos os seus objetos sejam mostrados no mesmo tamanho, não importando qual a resolução da tela. Aqui está um código que mostra como isso é feito:

Implementation

```
const
ScreenWidth: LongInt = 800; {I designed my form in 800x600 mode.}
ScreenHeight: LongInt = 600;
```

```
{SR *.DFM}
```

```
procedure TForm1.FormCreate (Sender: TObject);
```

```
begin
scaled := true;
```

```

if (screen.width <> ScreenWidth) then
begin
  height := longint(height) * longint(screen.height) DIV ScreenHeight;
  width := longint(width) * longint(screen.width) DIV ScreenWidth;
  scaleBy(screen.width, ScreenWidth);
end;
end;

```

Agora, você vai querer checar, se o tamanho dos fontes(de letra) estão OK. Antes de trocar p tamanho do fonte, você precisará ter certeza de que o objeto realmente tem a propriedade fonte pela checagem da RTTI. Isso pode ser feito assim:

```

USES tyinfo; {Add this to your USES statement.}

```

```

var

```

```

i:integer;

```

```

begin
  for i := componentCount - 1 downto 0 do
  with components[i] do
  begin
    if GetPropInfo(ClassInfo, 'font') <> nil then
      font.size := (NewFormWidth DIV OldFormWidth) * font.size;
    end;
  end;
end;

```

```

{Esta é a maneira longa de fazer a mesma coisa}

```

```

var

```

```

i:integer;

```

```

p:PPropInfo;

```

```

begin
  for i := componentCount - 1 downto 0 do
  with components [i] do
  begin
    p := GetPropInfo (ClassInfo, 'font');
    if assigned (p) then
      font.size := (NewFormWidth DIV OldFormWidth) * font.size;
    end;
  end;
end;

```

Atenção: Nem todos os objetos tem a propriedade FONT. Isso deve ser o suficiente para você começar.

Atenção: A seguir, algumas dicas para ter em mente quando representar aplicações Delphi (formulários) em diferentes resoluções de Tela:

\* Decida antecipadamente, na etapa de criação do formulário, se ele será escalável ou não. A vantagem de um não escalável é

que nada muda em tempo de execução. A desvantagem é equivalente (seu formulário pode ser muito pequeno ou grande para alguns sistemas se não for usada escala).

\* Se você não for usar formulário escalável, configure o set scaled to False.

\* Ou então, configure a propriedade scaled do formulário para True.

\* Configure a propriedade AutoScroll para False. AutoScroll = True quer dizer "não mexa no tamanho do frame do formulário em tempo de execução", o que não parece bom quando o conteúdo do formulário muda de tamanho.

\* Configure a fonte do formulário para uma True Type escalável, como a Arial MS. San Serif é uma boa alternativa, mas lembre que ainda é uma fonte bitmapped. Só a Arial dará uma fonte dentro de um pixel da altura desejada. ATENÇÃO: Se a fonte usada em uma aplicação não estiver instalada no computador, o Windows selecionará uma fonte alternativa da mesma família para utilizar. O tamanho dessa fonte pode não corresponder ao da fonte original, podendo causar problemas.

\* Configure a propriedade position do formulário para uma opção diferente de poDesigned. poDesigned deixa o formulário onde você o deixou ( no design Time), o que sempre termina fora da margem, à esquerda da minha tela 1280 x 1024 - e completamente fora da tela 640 x 480.

\* Não amontoe controles no formulário - deixe pelo menos 4 pixels entre else, para que uma mudança de um pixel nas margens (devido a apresentação em escala) não mostre controles sobrepostos.

\* Para labels de uma linha alinhadas à esquerda ou à direita, configure o AutoSize para True. Para outras formas de alinhamento configure o AutoSize para False.

\* Tenha certeza de que há espaço em branco suficiente num componente de labels para alterações no tamanho da fonte - um espaço de 25% do comprimento da linha de caracteres mostrada é um pouco a mais do que se precisa, mas é mais seguro. (Você vai precisar de um espaço equivalente a 30% de expansão para string labels se você pretende traduzir sua aplicação para outra linguagem). Se o Autosize estiver em False, tenha certeza de que realmente configurou o tamanho do label corretamente. Se o Autosize estiver em True, esteja certo de que há espaço suficiente para que o label se amplie.

\* Em labels de múltiplas linhas ou de termos ocultos, deixe pelo menos uma linha em branco na base. Isso vai ser necessário para incluir o que estiver sobrando quando o texto for oculto de maneira diferente, pela mudança do tamanho da fonte com a escala. Não assuma isso porque está usando fontes grandes. Você não tem que deixar sobra de texto - as fontes (grandes) de outros usuários podem ser maiores que as suas!

\* Tenha cuidado quando abrir um projeto em IDEs com resoluções diferentes. Assim que o formulário for aberto, sua propriedade Pixel per Inch será modificada, e gravada para o DFM se você salvar o projeto. É melhor testar a aplicação rodando sozinho, e

editar o formulário em apenas uma resolução. Editar em várias resoluções e tamanhos de fonte provoca problemas de fluxo e tamanho dos componentes.

\*Falando em fluxo de componentes, não represente o formulário em escala muitas vezes, quando estiver sendo criado ou quando tiver sendo executado. Cada escala introduz erros de roundoff que se acumulam muito rapidamente, uma vez que as coordenadas são rigorosamente interias. Quando valores fracionários forem retirados das origens e tamanhos do controle com cada sucessiva representação em escala, os controles parecerão deslizar para noroeste e ficar menores. Se você quer deixar seus usuários representarem o formulários em escala quantas vezes quiserem, comece com um formulário recentemente criado para que erros de escala não se acumulem.

\* Definitivamente, não mexa na propriedade Pixel per Inch do formulário.

\* Em geral, não é necessário criar formulários em uma resolução específica, mas é essencial que você os revise em 640 x 480 com fontes pequenas e/ou grandes, e em alta resolução com fontes pequenas e/ou grandes antes de liberar suas aplicações. Isso deverser parte de sua lista de conferência para testar a compatibilidade do sistema regularmente.

\* Preste bastante atenção em todos os componentes que são basicamente TMemos de uma linha - com o TDBLookupCombo. O controle de edição (multi-linhas) do Windows sempre mostra apenas linhas inteiras de texto. Se o controle for muito curto para sua fonte, um TMemos não mostrará coisa alguma, e um TEdit mostrará um pedaço do texto. É melhor fazer esses componentes um pouco maiores do que deixá-los um pixel menores e não aparecer nada do texto.

\* Tenha em mente que toda representação em escala é proporcional à diferença da altura da fonte entre o modo de execução e o modo de desenho, NÃO à resolução ou ao tamanho do monitor. Lembre também que as origens dos seus controles serão alteradas quando o formulário for representado em escala. Você não pode aumentar componentes muito bem sem também movê-los um pouco, novamente.

Obtendo e modificando a posição do cursor em um TMemos

Modificando a posição:

```
ActiveControl:=Memo1;  
MemoCursorTo(Memo1,2,3);
```

Obtendo a Posição:

```
GetMemoLineCol(Memo1,Linha,Coluna);
```

[Executar um programa do DOS e fechá-lo em seguida](#)

```
WinExec("command.com /c programa.exe",sw_ShowNormal);
```

## Como posso rolar um form usando pgUP and pgDn.

Versão: Todas

Plataforma: Windows/Win32

Q. Como posso fazer funções de rolagem num componente TForm usando comandos de teclado? Por exemplo, rolar pra cima e pra baixo quando pressionar PgUp ou PgDown. Existe algum método simples de fazer isso???

R. O rolamento do form é completo fazendo-se uma modificação na posição das propriedades VertScrollbar ou HorzScrollbar do form. Como mostrado no código a seguir:

```
procedure TForm1.FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
const
  PageDelta = 10;
begin
  With VertScrollbar do
    if Key = VK_NEXT then
      Position := Position + PageDelta
    else if Key = VK_PRIOR then
      Position := Position - PageDelta;
end;
```

## Tocando Sons WAV

Para reproduzir sons no formato WAV em um programa em Delphi é simples, o usuário deverá colocar na cláusula

Uses o MMSystem. E no corpo do programa o comando:

```
SndPlaySound('C:\Windows\Media\Som.wav',SND_ASYNC);
```

## Colocar Funções em uma DLL

Edite diretamente no DPR, e depois salve como Funcoes.dpr:

```
Library Funcoes;
```

```
Uses SysUtils,WinTypes,WinProcs;
```

```
{ Uma função que tira os espaços no início e no final de uma string }
```

```
Function Trim(J:String):String; Export;
```

```
Begin
```

```
While J[Length(J)]=#32 do Dec(J[0]);
```

```
If Length(J)>1 then
```

```
While (J[1]=' ') do
```

```
Begin
```

```
Delete(J,1,1);
```

```
If Length(J)<=1 then J:='';
```

```
end;
```

```
Result:=J;
```

```
end;
```

```
Exports { Torna visível para os programas }
```

```
Trim;
```

```
Begin
```

End.

Para usar num programa:

```
Unit Unit1;
Interface
uses
SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
Forms, Dialogs, StdCtrls, Buttons;
type
TForm1 = class(TForm)
procedure FormCreate(Sender: TObject);
procedure FormClick(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;
Var
Form1: TForm1;
Implementation
{ Declara a funcao }
Function Trim(J:String):String; External 'funcoes.dll';
{$R *.DFM}
Procedure TForm1.FormClick(Sender: TObject);
begin
Caption:=Trim(' Visite sempre o Delphi Club '); { Note os espacos }
end;
```

As vantagens de colocar as funções em DLL são:

1. O programa exigirá menos memória
2. Você poderá reaproveitar as funções
3. Em alguns casos pode-se atualizar apenas as dll para um upgrade

### [Compactando tabelas](#)

Para compactar (remover fisicamente todos registros apagados) de uma tabela Paradox deve-se utilizar o seguinte código

```
procedure ParadoxPack(Table : TTable);
var
  TBDesc : CRTblDesc;
  hDb: hDbiDb;
  TablePath: array[0..dbiMaxPathLen] of char;
begin
  FillChar(TBDesc,Sizeof(TBDesc),0);
  with TBDesc do begin
    StrPCopy(szTblName,Table.TableName);
    StrPCopy(szTblType,szParadox);
    bPack := True;
  end;
  hDb := nil;
  Check(DbiGetDirectory(Table.DBHandle, True, TablePath));
  Table.Close;
  Check(DbiOpenDatabase(nil, 'STANDARD', dbiReadWrite,
```

```

        dbiOpenExcl,nil,0, nil, nil, hDb));
    Check(DbiSetDirectory(hDb, TablePath));
    Check(DBIDoRestructure(hDb,1,@TBDesc,nil,nil,nil,False));
    Table.Open;
end;

```

### Verifica validade de CGC e CPF

```
unit CPFeCGC;
```

```
interface
function cpf(num: string): boolean;
function cgc(num: string): boolean;
```

```
implementation
```

```
uses SysUtils;
```

```
function cpf(num: string): boolean;
var
n1,n2,n3,n4,n5,n6,n7,n8,n9: integer;
d1,d2: integer;
digitado, calculado: string;
begin
n1:=StrToInt(num[1]);
n2:=StrToInt(num[2]);
n3:=StrToInt(num[3]);
n4:=StrToInt(num[4]);
n5:=StrToInt(num[5]);
n6:=StrToInt(num[6]);
n7:=StrToInt(num[7]);
n8:=StrToInt(num[8]);
n9:=StrToInt(num[9]);
d1:=n9*2+n8*3+n7*4+n6*5+n5*6+n4*7+n3*8+n2*9+n1*10;
d1:=11-(d1 mod 11);
if d1>=10 then d1:=0;
d2:=d1*2+n9*3+n8*4+n7*5+n6*6+n5*7+n4*8+n3*9+n2*10+n1*11;
d2:=11-(d2 mod 11);
if d2>=10 then d2:=0;
calculado:=inttostr(d1)+inttostr(d2);
digitado:=num[10]+num[11];
if calculado=digitado then
    cpf:=true
else
    cpf:=false;
end;
```

```
function cgc(num: string): boolean;
var
n1,n2,n3,n4,n5,n6,n7,n8,n9,n10,n11,n12: integer;
d1,d2: integer;
digitado, calculado: string;
begin
n1:=StrToInt(num[1]);
n2:=StrToInt(num[2]);
n3:=StrToInt(num[3]);
```

```

n4:=StrToInt(num[4]);
n5:=StrToInt(num[5]);
n6:=StrToInt(num[6]);
n7:=StrToInt(num[7]);
n8:=StrToInt(num[8]);
n9:=StrToInt(num[9]);
n10:=StrToInt(num[10]);
n11:=StrToInt(num[11]);
n12:=StrToInt(num[12]);
d1:=n12*2+n11*3+n10*4+n9*5+n8*6+n7*7+n6*8+n5*9+n4*2+n3*3+n2*4+n1*5;
d1:=11-(d1 mod 11);
if d1>=10 then d1:=0;
d2:=d1*2+n12*3+n11*4+n10*5+n9*6+n8*7+n7*8+n6*9+n5*2+n4*3+n3*4+n2*5+n1*6;
d2:=11-(d2 mod 11);
if d2>=10 then d2:=0;
calculado:=inttostr(d1)+inttostr(d2);
digitado:=num[13]+num[14];
if calculado=digitado then
  cgc:=true
else
  cgc:=false;
end;

end.

```

### Gera número por extenso

```

unit Ext;

interface
function extenso (valor: real): string;

implementation

uses
  SysUtils, Dialogs;

function extenso (valor: real): string;
var
  Centavos, Centena, Milhar, Milhao, Texto, msg: string;
const
  Unidades: array[1..9] of string = ('Um', 'Dois', 'Tres', 'Quatro', 'Cinco',
    'Seis', 'Sete', 'Oito', 'Nove');
  Dez: array[1..9] of string = ('Onze', 'Doze', 'Treze', 'Quatorze', 'Quinze',
    'Dezesseis', 'Dezessete', 'Dezoito', 'Dezenove');
  Dezenas: array[1..9] of string = ('Dez', 'Vinte', 'Trinta', 'Quarenta',
    'Cinquenta', 'Sessenta', 'Setenta',
    'Oitenta', 'Noventa');
  Centenas: array[1..9] of string = ('Cento', 'Duzentos', 'Trezentos',
    'Quatrocentos', 'Quinhentos', 'Seiscentos',
    'Setecentos', 'Oitocentos', 'Novecentos');

function ifs(Expressao: Boolean; CasoVerdadeiro, CasoFalso: String): String;
begin
  if Expressao

```

```

then Result:=CasoVerdadeiro
else Result:=CasoFalso;
end;

```

```

function MiniExtenso (trio: string): string;
var
Unidade, Dezena, Centena: string;
begin
Unidade:="";
Dezena:="";
Centena:="";
if (trio[2]='1') and (trio[3]<>'0') then
begin
Unidade:=Dez[stroioint(trio[3])];
Dezena:="";
end
else
begin
if trio[2]<>'0' then Dezena:=Dezenas[stroioint(trio[2])];
if trio[3]<>'0' then Unidade:=Unidades[stroioint(trio[3])];
end;
if (trio[1]='1') and (Unidade=") and (Dezena=")
then Centena:='cem'
else
if trio[1]<>'0'
then Centena:=Centenas[stroioint(trio[1])]
else Centena:="";
Result:= Centena + ifs((Centena<>" and ((Dezena<>" or (Unidade<>")), ' e ', ")
+ Dezena + ifs((Dezena<>" and (Unidade<>"), ' e ', ") + Unidade;
end;

```

```

begin
if (valor>999999.99) or (valor<0) then
begin
msg:='O valor está fora do intervalo permitido.';
msg:=msg+'O número deve ser maior ou igual a zero e menor que 999.999,99.';
msg:=msg+' Se não for corrigido o número não será escrito por extenso.';
showmessage(msg);
Result:="";
exit;
end;
if valor=0 then
begin
Result:="";
Exit;
end;
Texto:=formatfloat('000000.00',valor);
Milhar:=MiniExtenso(Copy(Texto,1,3));
Centena:=MiniExtenso(Copy(Texto,4,3));
Centavos:=MiniExtenso('0'+Copy(Texto,8,2));
Result:=Milhar;
if Milhar<>" then
if copy(texto,4,3)='000' then
Result:=Result+' Mil Reais'
else
Result:=Result+' Mil, ';

```

```

if (((copy(texto,4,2)='00') and (Milhar<>"
    and (copy(texto,6,1)<>'0')) or (centavos="))
    and (Centena<>" then Result:=Result+' e ';
if (Milhar+Centena <>" then Result:=Result+Centena;
if (Milhar=") and (copy(texto,4,3)='001') then
    Result:=Result+' Real'
else
    if (copy(texto,4,3)<>'000') then Result:=Result+' Reais';
if Centavos=" then
begin
    Result:=Result+'.';
    Exit;
end
else
begin
    if Milhar+Centena=" then
        Result:=Centavos
    else
        Result:=Result+' e '+Centavos;
if (copy(texto,8,2)='01') and (Centavos<>" then
    Result:=Result+' Centavo.'
else
    Result:=Result+' Centavos.';
end;
end;

end.

```

[Preenche com quantidade determinada de zeros o lado esquerdo de uma string](#)

```

unit Zero;

interface
function RetZero(ZEROS:string;QUANT:integer):String;

implementation

function RetZero(ZEROS:string;QUANT:integer):String;
var
    I,Tamanho:integer;
    aux: string;
begin
    aux:=zeros;
    Tamanho:=length(ZEROS);
    ZEROS:=";
    for I:=1 to quant-tamanho do
        ZEROS:=ZEROS+'0';
    aux:=zeros+aux;
    RetZero:=aux;
end;
end.

```

## Ponto Decimal

```
if Key in [',','.' ] then Key := DecimalSeparator;
```

Coloque no evento OnKeyPress dos seus TEdits numéricos

## FindNearest numa Query

```
Query.Locate('campo onde ira porcurar',Texto a buscar,[loPartialKey])
```

## Relatórios em HTML

Em vez de Quickreport1.Print faça :

```
QuickRep1.ExportToFilter(TQRHtmlExportFilter.Create('teste.html'));
```

## Desligando Windows via programação

```
function ExitWindowsEx(uFlags : integer;           // shutdown operation
                      dwReserved : word) : boolean; // reserved
external 'user32.dll' name 'ExitWindowsEx';
```

```
procedure Tchou;
```

```
const
```

```
EWX_LOGOFF   = 0; // Dá "logoff" no usuário atual
EWX_SHUTDOWN = 1; // "Shutdown" padrão do sistema
EWX_REBOOT   = 2; // Dá "reboot" no equipamento
EWX_FORCE    = 4; // Força o término dos processos
EWX_POWEROFF = 8; // Desliga o equipamento
```

```
begin
```

```
ExitWindowsEx(EWX_FORCE, 0);
```

```
end;
```

## Como saber se o CD está no drive

```
Function MidiaPresente(MediaPlayer: TMediaPlayer): Boolean;
```

```
var
```

```
Params: MCI_STATUS_PARMS;
```

```
S: array [0..255] of char;
```

```
r: Integer;
```

```
begin
```

```
//verifica se existe um cd inserido
```

```
Params.dwItem:= MCI_STATUS_MEDIA_PRESENT;
```

```
r:= MCISendCommand(MediaPlayer.DeviceID, MCI_STATUS,
```

```
MCI_STATUS_ITEM, Integer(Addr(Params)));
```

```
if r <> 0 then
```

```
begin
```

```
MCIGetErrorString(r, S, SizeOf(S));
```

```
ShowMessage('Erro: ' + StrPas(S));
```

```
end
```

```
else
```

```
Result:= Params.dwReturn = 1;
```

```
end;
```

## Tradução de Mensagens

Depois de algum tempo pesquisando uma forma de fazer aparecer as mensagens em português, consegui uma solução muito fácil de implementar no ambiente de programação do Delphi 3.

*CHEGA DE YES/NO !!!*

```
messengerdlg('Confirma ? mtConfirmation, [mbYes, mbNo], 0);
```

Aí vai:

- 1 - No diretório DELPHI3\LIB, copie o arquivo consts.dcu para consts.old;
- 2 - Inicie o Delphi e crie um nova Unit;
- 3 - Insira nesta, o arquivo consts.int do diretório DELPHI3\DOC E faça as devidas alterações nas mensagens que desejares alterar e nas partes duplicadas da Unit como "implement" e etc, também deixe o cabeçalho como Unit Consts.
- 4 - Salve esta nova Unit no diretório DELPHI\LIB e pronto todas as mensagens alteradas por você estarão aplicadas nos seus próximos programas sem uma linha de programa e da forma que você quiser.

## Função que devolve tempo decorrido em uma string

```
Function NumDiasExtenso(NumDias:integer):string;  
var  
Anos, Meses, Dias : integer;  
sAnos, sMeses, sDias : string;  
begin  
  { --- Calcula o número de anos --- }  
  Anos := 0;  
  while NumDias >= 365 do  
  begin  
    Anos := Anos + 1;  
    NumDias := NumDias - 365;  
  end;  
  if Anos > 1 then  
    sAnos := ' anos,'  
  else  
    sAnos := ' ano,';  
  
  { --- Calcula o número de meses --- }  
  Meses := 0;  
  while NumDias >= 30 do  
  begin  
    Meses := Meses + 1;  
    NumDias := NumDias - 30;  
  end;  
  if Meses > 1 then  
    sMeses := ' meses e '  
  else  
    sAnos := ' mês e ';  
  
  { --- O Número de dias é a sobra --- }
```

```

Dias := NumDias;
if sDias > 1 then
sDias := 'dias'
else
sDias := 'dia';

Return := Inttostr(Anos)+sAnos+inttostr(Meses)+sMeses+inttostr(Dias)+sDias;
end;

```

Criando uma rotina para pegar todos os erros do programa.

```

Procedure MostraErro;
Begin
ShowMessage('Ocorreu algum erro!');
end;

TForm1.Create;
Begin
Application.OnException:=MostraErro;
end;

```

Capturando conteúdo do desktop

```

procedure TForm1.FormResize(Sender: TObject);
var
R : TRect;
DC : HDC;
Canv : TCanvas;
begin
R := Rect( 0, 0, Screen.Width, Screen.Height );
DC := GetWindowDC( GetDesktopWindow );
Canv := TCanvas.Create;
Canv.Handle := DC;
Canvas.CopyRect( R, Canv, R );
ReleaseDC( GetDesktopWindow, DC );
end;

```

Obtendo número do registro atual

```

Function Recno(Dataset: TDataset): Longint;

var
CursorProps: CurProps;
RecordProps: RECProps;

begin
{ Return 0 if dataset is not Paradox or dBASE }
Result := 0;
with Dataset do
begin
if State = dsInactive then DBError(SDataSetClosed);
Check(DbGetCursorProps(Handle, CursorProps));
UpdateCursorPos;
try
Check(DbGetRecord(Handle, dbiNOLOCK, nil, @RecordProps));
case CursorProps.iSeqNums of

```

```

0: Result := RecordProps.iPhyRecNum; { dBASE }
1: Result := RecordProps.iSeqNum; { Paradox }
end;
except
on EDBEngineError do
Result := 0;
end;
end;
end;

```

#### Enviando um arquivo para a lixeira

```

uses ShellAPI;

Function DeleteFileWithUndo(sFileName : string ) : boolean;

var
fos : TSHFileOpStruct;

Begin
FillChar( fos, SizeOf( fos ), 0 );
With fos do
begin
wFunc := FO_DELETE;
pFrom := PChar( sFileName );
fFlags := FOF_ALLOWUNDO
or FOF_NOCONFIRMATION
or FOF_SILENT;
end;
Result := ( 0 = ShFileOperation( fos ) );
end;

```

#### Desabilitar o CTRL+ALT+DEL e ALT+TAB

```

Var
numero: integer;
begin
SystemParametersInfo(97,Word(true),@numero,0);
end;

{ Para habilitar é só chamar a mesma função com Word(false) }

```

#### Carregar um cursor animado (\*.ani)

```

const
cnCursorID1 = 1;
begin
Screen.Cursors[ cnCursorID1 ] :=
LoadCursorFromFile('c:\win95\cursors\cavalo.ani' );
Cursor := cnCursorID1;
end;

```

#### Saindo do Windows

```

{ Reinicia o Windows }
ExitWindowsEx(EWX_REBOOT, 0);

{ Desliga o Windows }
ExitWindowsEx(EWX_SHUTDOWN, 0);

{ Força todos os programa a desligarem-se }
ExitWindowsEx(EWX_FORCE, 0);

```

### Modificando a posição do cursor em um Memo

Modificando a posição:

```

ActiveControl:=Memo1;
MemoCursorTo(Memo1,2,3);

```

Obtendo a Posição:

```

GetMemoLineCol(Memo1,Linha,Coluna);

```

### Traduzindo a mensagem "Delete Record ?"

Quando clicamos sobre o botão de deleção no DBNavigator (o do sinal de menos) surge uma box com a mensagem "Delete Record?" com botões **Ok** e **Cancel**.

Para fazer aparecer a mensagem em português deverá selecionar o componente Table e mudar a propriedade **ConfirmDelete** para **False** e no evento da tabela **BeforeDelete** colocar o seguinte:

```

procedure TForm1.Table1BeforeDelete(DataSet:TDataSet);
begin
if MessageDlg('Eliminar o Registro?',mtConfirmation,[mbYes,mbNo],0)<>mrYes then Abort;
end;

```

### Pegando o Nome do usuário e a Empresa do Windows

```

Uses Registry;
Procedure GetUserCompany;
var
reg: TRegIniFile;
begin
reg := TRegIniFile.create('SOFTWARE\MICROSOFT\MS SETUP (ACME)');
Edit1.Text := reg.ReadString('USER INFO','DefName,');
Edit2.Text := reg.ReadString('USER INFO','DefCompany,');
reg.free;
end;

```

### Escrevendo um Texto na Diagonal usando o Canvas

```

procedure TForm1.Button1Click(Sender: TObject);
var
lf: TLogFont;
tf: TFont;
begin
with Form1.Canvas do begin
Font.Name := 'Arial';
Font.Size := 24;
tf := TFont.Create;

```

```

tf.Assign(Font);
GetObject(tf.Handle, sizeof(If), @If);
If.IfEscapement := 450;
If.IfOrientation := 450;
tf.Handle := CreateFontIndirect(If);
Font.Assign(tf);
tf.Free;
TextOut(20, Height div 2, 'Texto Diagonal!');
end;
end;

```

#### Fundo do texto transparente

```

procedure TForm1.Button1Click(Sender: TObject);
var
OldBkMode : integer;
begin
with Form1.Canvas do begin
Brush.Color := clRed;
FillRect(Rect(0, 0, 100, 100));
Brush.Color := clBlue;
TextOut(10, 20, 'Não é Transparente!');
OldBkMode := SetBkMode(Handle, TRANSPARENT);
TextOut(10, 50, 'É Transparente!');
SetBkMode(Handle, OldBkMode);
end;
end;

```

#### Formatação de Casas Decimais

```

procedure TForm1.Button1Click(Sender: TObject);
var num : integer;
begin
num:=12450;
Edit1.text:=formatfloat('###,###,##0.00', num)
end;

```

#### Escondendo/Mostrando o botão Iniciar

```

procedure EscondeIniciar(Visible:Boolean);
Var taskbarhandle,
buttonhandle : HWND;
begin
taskbarhandle := FindWindow('Shell_TrayWnd', nil);
buttonhandle := GetWindow(taskbarhandle, GW_CHILD);
If Visible=True Then Begin
ShowWindow(buttonhandle, SW_RESTORE); {mostra o botão}
End Else Begin
ShowWindow(buttonhandle, SW_HIDE); {esconde o botão}
End;
end;

```

#### Esconde/Mostra a Barra de Tarefas

```

procedure EscondeTaskBar(Visible: Boolean);
var wndHandle : THandle;
    wndClass : array[0..50] of Char;
begin
  StrPCopy(@wndClass[0], 'Shell_TrayWnd');
  wndHandle := FindWindow(@wndClass[0], nil);
  If Visible=True Then Begin
  ShowWindow(wndHandle, SW_RESTORE); {Mostra a barra de tarefas}
  End Else Begin
  ShowWindow(wndHandle, SW_HIDE); {Esconde a barra de tarefas}
End;
end;

```

### Desabilitando o Alt+Tab

```

procedure TurnSysKeysOff;
var OldVal : LongInt;
begin
  SystemParametersInfo (97, Word (True), @OldVal, 0)
end;
procedure TurnSysKeysOn;
var OldVal : LongInt;
begin
  SystemParametersInfo (97, Word (False), @OldVal, 0)
end;
Por: Adenilton Rodrigues - arinfo@estaminas.com.br

```

### Detectando o Numero Serial do HD

```

Function SerialNum(FDrive:String) :String;
Var Serial:DWord;
    DirLen,Flags: DWord;
    DLabel : Array[0..11] of Char;
begin
  Try
  GetVolumeInformation(PChar(FDrive+':\'),dLabel,12,@Serial,DirLen,Flags,nil,0);
  Result := IntToHex(Serial,8);
  Except Result := "";
end;
end;

```

### Como Limpar Todos os Edit's de um Form de uma só vez?

```

Procedure LimpaEdit;
var i : Integer;
begin
  for i := 0 to ComponentCount -1 do
  if Components[i] is TEdit then
  begin
  TEdit(Components[i]).Text := "";
  end;
end;

```

## Marcando um pedaço do código

As vezes quando vc tem uma unidade com muitas linhas de código (umas 1000 por exemplo), fica difícil achar o bloco de código que você quer; e para facilitar isso o Delphi tem um tipo de "bookmark" de código. Para colocar o bookmark, posicione no lugar onde você quer marcar e pressione CTRL+SHIFT+ o número do bookmark que você vai criar de (0..9), por exemplo CTRL+SHIFT+0. Para retornar ao bloco marcado você deve pressionar CTRL+ o número do bookmark. Por exemplo CTRL+1. Ps: A opção Editor FindTextAtCursor deve estar marcada, ou estas teclas não irão funcionar.

## Um programinha para alterar o papel de parede do Windows

```
program wallpapr;
uses Registry, WinProcs;
procedure SetWallpaper(sWallpaperBMPPath : String; bTile : boolean );
var
  reg : TRegIniFile;
begin
  // Mudando o Registro HKEY_CURRENT_USER
  // Control Panel\Desktop
  // TileWallpaper (REG_SZ)
  // Wallpaper (REG_SZ)
  reg := TRegIniFile.Create('Control Panel\Desktop' );
  with reg do begin
    WriteString( " 'Wallpaper',sWallpaperBMPPath );
    if( bTile )then begin
      WriteString( " 'TileWallpaper', '1' );
    end else begin
      WriteString( " 'TileWallpaper', '0' );
    end;
  end;
  reg.Free;

  // Mostrar que o parametro do sistema foi alterado
  SystemParametersInfo( SPI_SETDESKWALLPAPER,0, Nil, SPIF_SENDWININICHANGE );
end;
begin
  SetWallpaper( 'c:\winnt\winnt.bmp', False );
end.
```

## Alterando cor de linha de um DBGrid

Coloque a propriedade defaultdrawdata do dbgrid em FALSE

No evento onDrawColumnCell do seu grid coloque o seguinte:

```
procedure TForm1.DBGrid1DrawColumnCell(Sender: TObject;
const Rect: TRect; DataCol: Integer; Column: TColumn;
State: TGridDrawState);
begin
  If table1PRAZO.Value > DATE then // condição
  Dbgrid1.Canvas.Font.Color:= clFuchsia; // coloque aqui a cor desejada
  Dbgrid1.DefaultDrawDataCell(Rect, dbgrid1.columns[datacol].field, State);
end;
```

## Diretório de instalação do windows

```
function PegaSysDir: string;
var
MeuBuffer: Array [1..128] of Char;
retorno: Integer;
Begin
retorno:=GetSystemDirectory(@MeuBuffer,128);
if (retorno>128) OR (retorno=0) then
PegaSysDir:=""
else
PegaSysDir:=StrPas(@MeuBuffer);
End; {prc}
```

## Exclusividade para o programa

Gostaria de saber como fazer para que, ao iniciar minha aplicacao Delphi, eu " desabilite " o shell do Windows (Explorer). Ou seja, o que eu preciso e' de uma forma de fazer com que apos a minha aplicacao seja iniciada, o usuario nao tenha como alternar entre programas, acessar outros icones, etc

No System.ini você tem uma configuração como esta :

```
Shell=Explorer.exe
```

Basta trocar por

```
Shell=Myprog.exe
```

Ou usando delphi

```
procedure TForm1.ChangeShell(String programa);
var ArquivoIni : Tinifile;
begin
try
ArquivoIni := Tinifile.Create('System.ini');
ArquivIni.WriteSection('Config','Shell','Myprog.exe');
fynally
ArquivoIni.Destroy;
end;

end;
```

## Substituindo TAB pelo ENTER

```
procedure TF_Padrao.FormKeyPress(Sender: TObject; var Key: Char);
begin
if Key = #13 then
if not (ActiveControl is TDBGrid) then
begin
Key := #0;
Perform(WM_NEXTDLGCTL, 0, 0);
end
```

```
else if (ActiveControl is TDBGrid) then
with TDBGrid(ActiveControl) do
if selectedIndex < (fieldcount -1) then
selectedIndex := selectedIndex +1
else
selectedIndex := 0;
end;
```

Ou então, pode-se tentar o seguinte método:

Utilize o evento onkeydown do componente e insira o seguinte comando:

```
if Key = VK_RETURN then Perform(Wm_NextDlgCtl,0,0);
```

este comando testa a tecla pressionada, se ela for um enter, manda o foco para o componente posterior.

### Copiando arquivos

```
CopyFile(Pchar(Origem),Pchar(Destino),false);
```

Onde Origem e' a variavel de que contem o nome do arquivo de origem  
Destino e' a variavel que contem o nome do arquivo destino  
False : Instrui para sobrescrever o arquivo destino (caso encontre)

### Criando tabela em tempo de execução

Use os metodos FieldDefs e CreateTable para isso. Veja como criar uma estrutura temporaria:

```
with TTable.Create(Application) do begin
Active := False;
DatabaseName := 'C:\TEMP';
TableName := 'TESTE.TMP';
TableType := ttDefault;

FieldDefs.Add('CODCLI', ftString, 5, False);
FieldDefs.Add('NOMCLI', ftString, 40, False);
FieldDefs.Add('DATCAD', ftDate, 0, False);
CreateTable;
Free;
end;
```

### Executar comandos do Dos

```
WinExec(PChar('command.com /c format a: /v ' +Edit1.Text),SW_SHOWNORMAL);
```

### Armazendo BMP's em arquivos RES

1. Criem um arquivo texto, por exemplo: RECURSOS.RC com um conteudo igual a este:  
BITMAP\_1 BITMAP "C:\Imagens\Grafico.bmp"  
para todos os bitmap's que vc deseja;
2. Compilem este arquivo usando o BRCC32.EXE que esta no diretorio BIN do Delphi sera gerado o arquivo RECURSOS.RES; e
3. Coloquem dentro do fonte do projeto:  
{*\$R RECURSOS.RES*}

Para usar o bitmap faça o seguinte:

```
VarTipoTBitmap:= LoadBitmap(HInstance,'BITMAP_1');
```

### QR armazenado num Blop

Os campos do Tipo TBlobField, tem metodos que permitem que sejam armazenados dados contidos em arquivos, ou em um Stream...

No primeiro caso (dos arquivos), o codigo seria algo como:

```
TBlobField(SuaTabela.FieldByName('SeuCampo')).LoadFromFile('NomedoArquivo');
```

No segundo caso, poderia ser feito um exemplo com o TRichEdit:

```
var  
Stream : TMemoryStream;  
begin  
Stream := TMemoryStream.Create;  
try  
RichEdit1.Lines.SaveToStream(Stream);  
Stream.Seek(0,soFromBeginning);  
TBlobField(SuaTabela.FieldByName('SeuCampo')).LoadFromStream(Stream);  
finally  
Stream.Free;  
end;  
end;
```

Ambos os exemplos, assumem que a tabela ja' estaria em modo de Edicao ou de Insercao.

### Deletando um arquivo

```
if FileExists('C:\MEUDIR\MEUARQ.DAT') then DeleteFile('C:\MEUDIR\MEUARQ.DAT');
```

### Diretório Windows e System

```
Function ExtractWindowsDir : String;  
Var Buffer : Array[0..144] of Char;  
Begin  
GetWindowsDirectory(Buffer,144);  
Result := FormatPath(StrPas(Buffer));  
End;
```

```
Function ExtractSystemDir : String;  
Var Buffer : Array[0..144] of Char;  
Begin  
GetSystemDirectory(Buffer,144);  
Result := FormatPath(StrPas(Buffer));  
End;
```

```
Function ExtractTempDir : String;  
Var Buffer : Array[0..144] of Char;  
Begin  
GetTempPath(144,Buffer);  
Result := FormatPath(StrPas(Buffer));  
End;
```

### Alterar papel de parede

```
procedure ChangeWallpaper(bitmap: string);
var pBitmap : pchar;
begin
bitmap:=bitmap+#0; {bitmap contém um arquivo *.bmp}
pBitmap:=@bitmap[1];
SystemParametersInfo(SPI_SETDESKWALLPAPER, 0, pBitmap, SPIF_UPDATEINIFILE);
end;
```

### Como fazer um “Hot Link”

Adicione um componente com o URL. Digite o seguinte código no seu evento OnClick:

```
procedure TForm1.URLLabelClick(Sender: TObject);
var TempString : array[0..79] of char;
begin
  StrPCopy(TempString,URLLabel.Caption);
  OpenObject(TempString);
end;
```

Insira a seguinte procedure logo após implementation:

```
procedure TTKOAboutBox.OpenObject(sObjectPath : PChar);
begin
  ShellExecute(0, Nil, sObjectPath, Nil, Nil, SW_NORMAL);
end;
```

Adicione “ShellAPI” no uses.

### Como saber se o disquete está no drive.

```
function DiskInDrive(const Drive: char): Boolean;
var DrvNum: byte;
EMode: Word;
begin
result := false;
DrvNum := ord(Drive);
if DrvNum >= ord('a') then dec(DrvNum,$20);
EMode := SetErrorMode(SEM_FAILCRITICALERRORS);
try
  if DiskSize(DrvNum-$40) <> -1 then result := true else messagebeep(0);
  finally SetErrorMode(EMode);
end;
end;
```

### Formatar disquete.

```
{implementation section}
....
const SHFMT_ID_DEFAULT = $FFFF;
// Formatting options
SHFMT_OPT_QUICKFORMAT = $0000;
SHFMT_OPT_FULL = $0001;
SHFMT_OPT_SYSONLY = $0002;
// Error codes
```

```
SHFMT_ERROR = $FFFFFFFF;  
SHFMT_CANCEL = $FFFFFFFE;  
SHFMT_NOFORMAT = $FFFFFFFD;
```

```
function SHFormatDrive(Handle: HWND; Drive, ID, Options: Word): LongInt; stdcall; external 'shell32.dll'  
name 'SHFormatDrive'
```

```
procedure TForm1.btnFormatDiskClick(Sender: TObject);  
var  
  retCode: LongInt;  
begin  
  retCode:= SHFormatDrive(Handle, 0, SHFMT_ID_DEFAULT, SHFMT_OPT_QUICKFORMAT);  
  if retCode < 0 then ShowMessage('Could not format drive');  
end;
```

[Como detectar as teclas de “seta”.](#)

Use os eventos KeyDown ou KeyUp e teste se Key = VK\_LEFT ou VK\_RIGHT, etc.

[Caps Lock e Num Lock](#)

```
procedure TMyForm.Button1Click(Sender: TObject);  
Var KeyState : TKeyboardState;  
begin  
  GetKeyboardState(KeyState);  
  if (KeyState[VK_NUMLOCK] = 0) then KeyState[VK_NUMLOCK] := 1  
  else KeyState[VK_NUMLOCK] := 0;  
  SetKeyboardState(KeyState);  
End;
```

Para a tecla Caps Lock basta trocar VK\_NUMLOCK por VK\_CAPITAL.

[BDE em 1 disquete](#)

Depois que apanhei bastante do BDE, recorri a lista e ninguém conseguiu me ajudar ... consegui resolver o problema. E como acredito que outras pessoas tenham o mesmo problema, resolvi colocar essa dica na lista. Por favor, se alguém tiver algo a acrescentar ou mesmo corrigir, sinta-se a vontade para compartilhar conosco.

Arquivos Exenciais para o BDE:

EUROPE.BLL

USA.BLL

IDR20009.DLL

IDAPI32.DLL

BLW32.DLL

IDAPI32.CFG <--- esse arquivo pode ter qualquer outro nome, desde que seja configurado no registro.

Drivers de Banco de Dados:

IDPDX32.DLL <--- Driver Paradox

IDASCI32.DLL <--- Driver ASCII

IDDBAS32.DLL <--- Driver DBase

IDODBC32.DLL <--- Driver ODBC

O BDE precisa de pelo menos um Driver de Banco de Dados para funcionar. Esses acima são apenas alguns, existem vários outros.

O BDE 4.51 + Driver Paradox compactados com o Algoritmo ZIP, ocupam aproximadamente 650 Kb.

Entradas no Registro do Win95:

HKEY\_LOCAL\_MACHINE

SOFTWARE\Borland\Database Engine

DLLPATH -> localização do BDE (Unidade+Caminho Completo)

CONFIGFILE01 -> localização do arquivo de configuração (Unidade+Caminho Completo+Nome do Arquivo)

SOFTWARE\Borland\BLW32

BLAPIPATH -> localização do BDE (Unidade+Caminho Completo)

LOCALE\_LIB1 -> localização do arquivo USA.BLL (Unidade+Caminho Completo+USA.BLL)

LOCALE\_LIB2 -> localização do arquivo EUROPE.BLL (Unidade+Caminho Completo+EUROPE.BLL)

Segue um pequeno exemplo de como registrar o BDE no Registro do Win95:

```
begin
Registry.RootKey := HKEY_LOCAL_MACHINE;
Registry.CreateKey('SOFTWARE\Borland\Database Engine');
Registry.OpenKey('SOFTWARE\Borland\Database Engine', False);
Registry.WriteString('DLLPATH', 'C:\ARQUIVOS DE PROGRAMAS\BDE\');
Registry.WriteString('CONFIGFILE1', 'C:\ARQUIVOS DE
PROGRAMAS\BDE\IDAPI32.CFG');
Registry.OpenKey('\', False);
Registry.CreateKey('SOFTWARE\Borland\BLW32');
Registry.OpenKey('SOFTWARE\Borland\BLW32', False);
Registry.WriteString('BLAPIPATH', 'C:\ARQUIVOS DE PROGRAMAS\BDE\');
Registry.WriteString('LOCALE_LIB1', 'C:\ARQUIVOS DE
PROGRAMAS\BDE\USA.BLL');
Registry.WriteString('LOCALE_LIB2', 'C:\ARQUIVOS DE
PROGRAMAS\BDE\EUROPE.BLL');
end;
```

Para compilar esse código, será necessário declarar a Unit Registry.

Como eu disse, esse é um exemplo bem simples. Ele nem mesmo verifica se o BDE já está registrado ou não.

Para criar o Alias através do seu instalador, você pode usar a função da API do BDE chamada DbAddAlias.

[Cor de fundo do hint](#)

Veja as propriedades de TApplication...

Application.HintColor := clAqua;

Application.HintPause := ...

Application.HintShortPause := ...

[Margem para RichText](#)

Se for um richedit e margens laterais(direita e esquerda) tenta

RichEdit1.Paragraph.FirstIndent -> Paragrafo  
RichEdit1.Paragraph.LeftIndent -> margem esquerda  
RichEdit1.Paragraph.RightIndent -> margem direita

### Mostrando progresso de uma SQL

Algumas pessoas estavam interessadas em saber como apresentar o progresso de um TQuery enquanto ele esta sendo aberto (ou executada, no caso de um INSERT / UPDATE / DELETE).

A tecnica que vou demonstrar nao apenas serve para o proposito procurado, mas tambem serve para mostrar o progresso de diversas outras atividades que o BDE executa, como:

- \* Criacao de tabelas
- \* Criacao de indices para tabelas
- \* Reestruturacao de tabelas
- \* Execucao de queries (ja comentado)
- \* alguma outra coisa que no momento nao me ocorre... :))

Importante:

1) No meu exemplo, estou usando o Delphi 3.02. Caso seu Delphi seja de uma versao menor, vc devera ter um trabalho extra para repor a classe TBDECallback. Acredito que seja possivel fazer uma rotina que funcione em Delphi 1, mas que com certeza dara um certo trabalhinho, ah, isso dara... :-/

2) Ate agora so usei esse codigo com tabelas Paradox, mas realmente acredito que ele venha a funcionar com base de dados Interbase, Oracle, etc...

3) Nao sei se com o uso do Opus, Apollo ou qualquer outro substituto do BDE a tecnica ira funcionar, uma vez que nao se estaria trabalhando com o BDE original. Talvez alguem da lista possa dar essa informacao.

Teoria

=====

Segundo o help do Delphi, "o TBDECallback eh um wrapper para uma funcao de callback do BDE. Com ele eh possivel instruir o BDE para que o mesmo execute algumas tarefas em resposta a eventos que ocorram durante uma chamada de uma funcao do BDE. " - Fim do plagio do arquivo de help.

O tipo de callback depende de um parametro CBType que eh fornecido no momento da criacao do TBDECallback. E, entre os diversos valores que o CBType pode apresentar, existe um que muito nos interessa; o cbGENPROGRESS. :))

Assim, vc deveria criar uma funcao de callback do tipo cbGENPROGRESS chamada AtualizaGauge e indicar que a mesma eh que devera ser executada

"entre cada respiracao" do BDE. Na rotina AtualizaGauge, o BDE iria te informar o percentual de progresso da tarefa .  
O que voce faria nessa rotina ? Simples... atualizar o Gauge / ProgressBar.

Tudo muito bonito, tudo muito comovente, mas agora vamos para o lado pratico...

### Pratica

Para que o BDE possa informar o progresso da tarefa, ele precisa obter essa informacao da base de dados que esta sendo utilizada. Acontece que, por razoes diferentes, nem sempre ele eh capaz de saber o PERCENTUAL da tarefa. Numa copia de registros de uma tabela para outra, ele pode saber que ja foram copiados 270 registros, mas nao saber que esse esforco representa 36 % de todos os registros que serao copiados.

Assim sendo, na funcao de callback que sera criada, receberemos um parametro do tipo pCBPROGRESSDesc, que por sua vez eh um ponteiro para uma estrutura que contem duas informacoes:

iPercentDone => percentual do servico realizado  
szMsg => texto descrevendo o progresso do servico.

Como usar esses parametros ? Simples: sempre que o iPercentDone for negativo, voce devera considerar o texto descrito no campo szMsg. Se for igual ou maior que zero, entao vc devera considerar o valor do proprio iPercentDone.

Uma boa noticia para quem se preocupa com as mensagens que aparecem em ingles, quando se quer na verdade mostra-las em portugues: a mensagem fornecida por szMsg devera sempre aparecer no formato <mensagem><:><valor>

.....

Exemplo:

Records copied: 170

Assim, voce pode procurar pelos dois pontos ":" e pegar o valor que vem a seguir para montar sua propria informacao em portugues.

Pessoalmente, ate agora nunca obtive um iPercentDone positivo. Li no newsgroup da Borland que poucas bases de dados eram capazes de informar o real percentual para o BDE. Se nao me engano, o Sybase era um deles... NAO ESTOU CERTO DISSO.

Vamos para um exemplo pratico ? Crie um projeto novo, e coloque um: TQuery, TButton, TProgressBar e TLabel.

Sua query deve ser montada para abrir uma tabela razoavelmente grande, de modo que a operacao de abertura demore um pouco.

Agora vamos aos codigos:

1) Acrescente a unit BDE no seu USES da unit.

2) Acrescente algumas declaracoes na declaracao do seu Form:

```
=====
type
TForm1 = class(TForm)
... (bla bla bla)
private
{ Private declarations }
FCBPROGRESSDesc: pCBPROGRESSDesc;
FProgressCallback: TBDECallback;
function GetDataCallback(CBInfo: Pointer): CBRTYPE;
public
{ Public declarations }
end;
=====
```

No evento OnCreate do seu Form:

```
=====
procedure TForm1.FormCreate(Sender: TObject);
begin
FCBPROGRESSDesc := AllocMem(SizeOf(CBPROGRESSDesc));
FProgressCallback := TBDECallback.Create(Self, Query1.Handle,
cbGENPROGRESS, FCBPROGRESSDesc, SizeOf(CBPROGRESSDesc),
GetDataCallback, True);
end;
=====
```

Percebam que no segundo parametro do Create do callback, eu coloquei Query1.Handle.

Caso voce queira usar isso numa TTable, coloque Table1.Handle.

E se quiser que essa funcao de callback seja chamada para todos os "progressos" de qualquer componente DataSet, voce deixa esse parametro como NIL.

No evento OnDestroy do Form:

```
=====
procedure TForm1.FormDestroy(Sender: TObject);
begin
FProgressCallback.Free;
FreeMem(FCBPROGRESSDesc, SizeOf(CBPROGRESSDesc));
end;
=====
```

E agora, a tao falada funcao de callback:

```
=====
function TForm1.GetDataCallback(CBInfo: Pointer): CBRTYPE;
begin
Result := cbrCONTINUE;
end;
=====
```

```

with pCBPROGRESSDesc(CBInfo)^ do
begin
if iPercentDone < 0 then
begin
Label1.Caption := szMsg;
Label1.Refresh;
ProgressBar1.StepIt; {Apenas para ficar rodando o gauge}
end
else
ProgressBar1.Position := iPercentDone;
end;
end;
=====

```

Agora eh so executar a query no clicar do botao e curtir o visual... :))

**IMPORTANTE !!!!!**

Caso voce receba uma mensagem de erro informando que nao foi possivel inicializar o BDE (o que provavelmente aconteceu, pois voce esta criando uma funcao de callback do BDE, quando ate entao nenhuma tabela havia sido aberta), va no DPR do seu projeto (Menu View -> Project Source) e faca o seguinte:

- 1) Acrescente a unit BDE no uses do projeto.
- 2) Acrescente a instrucao

```
DbiInit(nil);
```

```
apos a instrucao Application.Initialize;
```

Isso deve resolver o problema.

Bom, nao vou me alongar mais, porque senao essa mensagem vai ficar maior do que ja esta...

Espero que tenha contribuido para a solucao desse problema de mostrar progresso de uma query. Qualquer duvida mandem mensagem.

[Mudar de cor a linha do dbGrid](#)

```

procedure TForm1.DBGrid1DrawDataCell(Sender: TObject; const Rect: TRect;
Field: TField; State: TGridDrawState);
begin
if Table1.FieldName('Pagou').Value = True then
DBGrid1.Canvas.Brush.Color := clGreen
else
DBGrid1.Canvas.Brush.Color := clRed;
DBGrid1.Canvas.FillRect(Rect);
DBGrid1.DefaultDrawDataCell(Rect,Field,State);
end;

```

[Código usados pelas impressoras HP](#)

Veja abaixo alguns códigos usados pelas impressoras HP:

```
RESET = 027/069
BOLD1 = 027/040/115/051/066
BOLD0 = 027/040/115/048/066
ITALIC1 = 027/040/115/049/083
ITALIC0 = 027/040/115/048/083
UNDERLINE1 = 027/038/100/049/068
UNDERLINE0 = 027/038/100/064
LPI6 = 027/038/108/054/068
LPI8 = 027/038/108/056/068
CPI5 = 027/040/115/053/072
CPI6 = 027/040/115/054/072
CPI8 = 027/040/115/056/072
CPI10 = 027/040/115/049/048/072
CPI12 = 027/040/115/049/050/072
CPI17 = 027/040/115/049/054/046/054/055/072
CPI20 = 027/040/115/050/048/072
```

#### Verificando atributo do arquivo

Crie uma var do tipo word, por ex., Attributes. Depois, atribua a esta var o valor retornado por FileGetAttr. Ex.:

```
var
Attributes: Word;
begin
Attributes := FileGetAttr( 'nomedoarquivo' );

// Supondo 4 CheckBoxe's, 1 para cada atributo, Ok?
CheckBox1.Checked := (Attributes and faReadOnly) = faReadOnly;
CheckBox2.Checked := (Attributes and faArchive) = faArchive;
CheckBox3.Checked := (Attributes and faSysFile) = faSysFile;
CheckBox4.Checked := (Attributes and faHidden) = faHidden;
end;
```