Atalhos

Ctrl+E → Pesquisa incremental;
Ctrl+J → Inclusão automática, use | – barra – para posicionar o cursor ;
Ctrl+Shift+I → Endenação automática;
Ctrl+Shift+U → Retirar endenação;
Ctrl+Shift+P → Inicia uma macro;
Ctrl+Shift+R → Grava uma macro, pode-se inclusive usar Ctrl+F9;
*.dci → Pasta do Code Insigth;

OBJECT INSPECTOR

AutoMerge Propriedade do tipo Lógica que determina se os menus de diversos formulários devem ser combinados automaticamente.

ActiveControlPropriedade do tipo TwinControl, que indica o componente que possui o foco da aplicação.

AllowDelete Propriedade do tipo Lógico que determina se o usuário pode deletar o registro corrente usando a combinação de teclas Ctrl+Delete.

AllowGrayed Propriedade do tipo Lógico que determina se o componente pussuirá dois ou três estados. Se for igual a True, o componente poderá possuir três estados: cbChecked, cbUnchecked e cbGrayed. A diferença entre os estados cbGrayed e cbChecked é que no primeiro caso a marca de verificação aparece com uma cor cinza.

AllowInsert Propriedade do tipo Lógico que determina se o usuário pode inserir ou adicionar um registro usando a tecla Insert ou Ctrl+Insert.

AutoCalcFields Propriedade do tipo Lógico que determina se o evento OnCalcFields deve ser disparado quando a aplicação carrega um novo registro do Banco de Dados.

AutoDisplay Propriedade do tipo Lógico que define se o conteúdo do componente deve ser exibido automaticamente.

AutoEdit Propriedade do tipo Lógico que define se os controles conectados a um componente DataSource estarão ou não em modo de edição constante.

ClientHeight Propriedade to tipo Inteiro que define a altura, em pixel, da área-cliente de um componente.

ClientWidth Propriedade do tipo Inteiro que define a largura, em pixels, da área-cliente de um componente.

CommonAVI Propriedade do tipo TCommonAVI que define qual a anivação padrão em um componente Animate.

DefaultExt Propriedade do tipo TfileExt que especifica a extensão a ser adicionada ao nome de um arquivo quando o usuário digita o nome de um arquivo sem a sua

extensão.

Dragcursor Faz o cursor mudar sobre o objeto.

DropDownCount Propriedade do tipo Inteiro que define o número máximo de elementos a serem exibidos simultaneamente numa lista DropDown.

DropDownRows Propriedade do tipo Inteiro que define o número de colunas a serem exibidos simultaneamente numa lista DropDown.

DropDownWidth Propriedade do tipo Inteiro que define, em pixels, a largura da lista drop-down exibida pelo componente.

FixedColor Propriedade do tipo Tcolor e especifica a cor das colunas e linhas fixas em um componente de grade.

HideSelection Propriedade do tipo Lógico que define se um texto selecionado permanece selecionado quando o componente perde o foco.

ListSource Propriedade do tipo TdataSource que define o nome do componente que faz a conexão com a tabela de dados do qual será selecionado um campo cujo valor será exibido pelo componente.

PlainText Propriedade do tipo lógico que define se o texto será exibido com um único tipo de formatação ou com atributos de formatação distintos.

QuickDraw Propriedade do tipo lógico que define se a imagem a ser exibida no componente deverá ser desenhada rapidamente e com qualidade inferior ou com qualidade superior.

RowCount Propriedade do tipo Inteiro que define o número de linhas do componente.

SelectedColor Propriedade do tipo Tcolor, e define a cor da gua selecionada em um componenteem um componente TabSet.

ShortCut Cria um atalho. Ex.: Alt+A;

ShowFocus Propriedade do tipo lógico que define se um retângulo de foco deve ser desenhado no painel que exibe o registro corrente.

Spacing Propriedade do tipo Inteiro que define a distância, em pixels, entre a imagem gráfica e o texto.

StartFrame Propriedade do tipo Inteiro que define em que quadro a animação deve iniciar.

State Propriedade que define os vários estados que podem ser assumidos pelo componente.

StopFrame Propriedade do tipo inteiro que define que em quadro a animação deve parar.

TitleFont Propriedade do tipo Tfont que determina o tipo de fonte usada para exibir os títulos das colunas da grade.

ValueChecked Propriedade do tipo String cujo valor será atribuído a um campo de dados do registro corrente quando o usuário selecionar o componente.

Values Propriedade do tipo Tstring que armazena uma lista de strings em que cada item vai corresponder a um possível valor de um campo em um registro do banco de dados.

ValueUnchecked Propriedade do tipo String cujo valor será atribuído a um campo de dados do registro corrente quando o usuário retirar a seleção do componente.

WordWrap Propriedade do tipo lógico que define se o texto digitado deve passar para a linha seguinte quando atingir a margem direita do componente.

COMPONENTES

ControlBar Para ancorar barras de propriedades.

ImageList Lista com imagens para ser usadas em outros componentes, como tabcontrol.

PageScroller Para usar com edit, por exemplo, cria um scrool.

Splitter Para movimenter caixas, como barra de propriedades ou painéis.

TabControl Para usar com componente notebooke imagelist.

Tools/Da...Desktop Abre um banco de dados para colocar senhas, por exemplo.

TrackBar Barra de movimento.

OBSERVAÇÕES

Teclas:
VK_LButton → 1
VK_RButton → 2
VK_Back → 8
VK_Tab → 9
VK_Shift → 10
VK_LReturn → 13
VK_Space → 20
VK_Prior → 33
VK_Next → 34
VK_End → 35

VK_Home→ 36
VK_Left → 37
VK_Up → 38
VK_Right → 39
VK_Down→ 40
VK_Print → 42
VK_Insert → 45
VK_Delete → 46
VK_NumPad0→ 96
VK_NumPad1→ 97
VK_NumPad9→ 105
VK_Multiply → 106
VK_Add → 107
VK_Separator → 108
VK_Subtract → 109
VK_Decimal → 110
VK_Divide → 111
VK_F1 → 112
VK_F2 → 113
VK_F24 → 135
VK_NumLock→ 144
VK_Scroll→ 145
VK_LShift → 160, onde L e R são as setas direcionais
VK_RShift → 161
VK_LControl → 162
VK_RControl → 163

✓ **EditMask** ! →

0 → Número obrigatório;
9 → Número opcional;
L → Letra obrigatório;
l → Letra opcional;
A → Alfanumérico obrigatório;
a → Alfanumérico opcional;
C → Qualquer obrigatório;
c → Qualquer opcional;
→ Número ou símbolo opcional;

\ → Para usar qualquer caractere, \(999\) → (000);
> → Fomata como maiúscula;
< → Formata como minúscula;

No 2º termo, quando passa para Label, por exemplo:

0 → Não salva os caracteres extras, como / ou ( ), etc. Usar quando precisar somente dos números, para operações, por exemplo;
1 → Salva todos os caracteres;

✓ **Ponteiros:**

Notação Húngara

a → Matriz
b → Booleano – int
by → Caractere se sinal – byte
c → Caractere
cb → Contagem de bits
cr → Valor de referência de cor
cx, cy → Inteiro pequeno – tamanho x e y
dw → Inteiro longo se sinal – dword: unsigned long
fn → Função
h → Manipulador – handle
i → Inteiro
m_ → Membro de dado de uma classe
n → Inteiro pequeno ou inteiro
np → Ponteiro próximo
p → Ponteiro
l → Longo
lp → Ponteiro de inteiro longo
s → String
sz → String terminada em nulo
tm → Métrica de texto
w → Inteiro sem sinal – word

Exemplo: lpszTexto → Pontoiro longo para uma string terminada em nulo chamada Texto.

✓ **Criando Ajuda:**

File → New → Contents Help → Edit... → Nome.hlp → Padrão → Nome Fantasia → OK

Add → Heading → ...

Add → Topic → Topic ID ...

File → New → Projects Help → Nome → Options → Help Title → Nome Fantasia → Files → Nome.rft

MAP → DescontoChequeInformações=1 → Custodiados=2 → ...

Windows → Padrão → ...

No Word:

Desmarcar em Formatar → Parágrafo → Manter com o próximo

Cada quebra de página é uma tópico

Cada inserir nota de rodapé é uma informação:

$ → É o título do tópico ,aparece em localizar com todas as palavras do tópico

→ É o ID

+ → É a ordem de apresentação quando >> no arquivo de ajuda

k → São as informações contidas no índice no arquivo de ajuda

Exemplo:

$#+*Desconto de Cheques* → Título, ID e índice

[k]Taxa do Contrato → Qdo clicar em 'índice, taxa d...' vem para cá
É a taxa fornecida pelo ...
[k]Iofdescontochequetópicosavançados → Duplo – abre a partir de iof em outra janela
É a taxa cobrada ...custodiadoscustodiados... → Simples – abre pequena janela
Completo
Este possui ... taxa efetiva, média ponderadadescontochequetópicosavançados, etc.

.....Quebra de página.....

[#]Entregue ao banco ou financiador para guarda.→ ID, só aparece em pequena janela qdo clicar em 'custodiados', pois não possui $

_____ Notas de rodapé
[$] Desconto de Cheque
[#] DescontoChequeInformações → Ver acima o motivo do verde
[+] DescontoCheque:01 → Depois :02, :03, ...
[k] taxa do contrato
[k] taxas
[k] iof
[#] custodiados

No Delphi, para chamar use WinHelp(Application.Handle, 'c:\Meuhelp.hlp', help_contents, sw_normal);
Para usar F1, mude no F11 o campo HelpContext para o nº em MAP e no form indique o caminho em F11 → helpfile.

Para criar um componente, abrir uma aplicação e removê-la, abrir um componente, escolher o ancestral dando um nome e Intall. Siga em frente até abrir a unit, faça as alterações desejadas – com property por exemplo – e CTRL+F9. Install Packege e compile, feche salvando a unit com o nome.pas.

✓ **Ponto Flutuante:**
1.23e4 = 1.23×10^4 = 12300;
314 = 314. = 3.14e2 = +3.14e+2 = 31.4e1 = 314E+3 = 314e0

✓ **Desativando Alt+F4:**
No evento OnCloseQuery:
CancClose:=False.

✓ **Ocultando Iniciar:**

```
var
  TaskBarHandle, ButtonHandle:Hwnd;
begin
  TaskBarHandle:= FindWindow ('Shell_TrayWnd', Nil);
  ButtonHandle:= GetWindow (TaskBarHandle, GW_Child);
  ShowWindow (ButtonHandle, SW_Restore);  //mostra
```

[#] custodiados

```
  ShowWindow (ButtonHandle, SW_Hide);          //esconde
end;
```

Detectando a finalização dos Windows:

Declarar:

```
procedure WmendSession(var msg: twmendsession);
Message wm_endsession;
```

No programa:

```
procedure tform1.wmendsession(var msg: twmendsession);
begin
  If msg.endsession=true then showmessage ('O Windows está sendo finalizado
'+#13+às '+formatdatetime('c',now));
  Inherited;
end;
```

- ✓ **Retornando o diretório que a aplicação está rodando:**

```
var
  X: string;
begin
  X:=estractfilepath(application.exename);
end;
```

- ✓ **Criando uma dll, sem formulário:**

Abra uma nova dll em New → dll. Salve com o nome, por exemplo CriaDLL;

```
library CriaDLL;

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs;

  procedure TestedeDLL;          → Qualquer nome
begin
  ShowMessage ('OK');
end;

exports  Teste;

end.
```

- ✓ **Criando dll, com formulário:**

No projeto criado, altere o dpr para:

```
library Serial;

uses
  Forms,
  SerialU in 'SerialU.pas' {Form1};

procedure Livraria; export;
begin
  Application.Initialize; //Opcional
```

```
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end;

exports Livraria;

end.
```

✓ **Para acessar a dll em outro projeto use:**

```
procedure Livraria; external 'Serial.dll';

procedure TForm1.Button1Click(Sender: TObject);
begin
  (...);
  Livraria;
  (...);
end;
```

✓ **Outra dll, com formulário, sincronizada:** Fica caregada na memória.

No projeto criado, altere o dpr para:

```
library Serial;

uses
  SerialU in 'SerialU.pas' {Form1};

exports Livraria, Sincroniza;

end.
```

Na Unit do Form1, antes de **var** e depois de **type**, declare:

```
procedure Livraria; stdcall;
procedure Sincroniza(AppHandle:THandle); stdcall;
```

Retira a definação do Form1, isto é, apague:

```
var
  Form1: TForm1;
```

Inclua as procedures:

```
procedure Livraria; stdcall;
var
  Temp: TForm1;
begin
  Temp:=TForm1.Create(Application);
  Temp.ShowModal;
end;

procedure Sincroniza(AppHandle: THandle); stdcall;
begin
  Application.Handle:=AppHandle;
end;
```

```
procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  Action:=caFree;
end;
```

✓ **Para acessar a dll sincronizada em outro projeto use:**

Declarar normalmente e chamar dentro do projeto, em um botão, por exemplo:

```
procedure Livraria; stdcall; external 'Serial.dll'
procedure Sincroniza(AppHandle:THandle); stdcall; external 'Serial.dll'
```

✓ **Observação das dll**: Maiúsculas e minúsculas podem gerar erro de localização das dll.

✓ **Cor do hint:**

```
Application.HintColor:=clBlue;
```

✓ **Adquirindo o diretório do Windows:**

```
var
  Dire: array [0..255] of char;
begin
  GetWindowsDirectory (Dire, 144);
  Edit1.Text:=Dire;
end;
```

Localizando um arquivo:

```
var
  Hnd: Thandle;
  W3: Twin3FindData;
begin
  Hnd:=FindFirstFile('Teste.txt',w3);
  Edit1.Text:=IntToStr(hnd);
end;
```

✓ **Mudando os atributos de um arquivo:**

```
SetFileAttributes('C:\Teste.txt',1)   → 1 para somente leitura;
                                      → 2 para oculto;
                                      → 4 para sistema;
                                      → 6 para oculto e sistema;
                                      → 32 para arquivo.
```

✓ **Verifica resolução do sistema:**

```
if (GetSystemMetrics(SM_CXScreen)=640) and GetSystemMetrics
(SM_CYScreen)=480) then PagesDlg.WindowState:=wsMaximized;
```

✓ **Janela transparente:**

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  Brush.Style := bsClear;
```

- ✓ **Como interceptar as PF:**

```
procedure TForm1.FormKeyDown(Sender:Tobject;varKey:Word;Shift:TshiftState);
begin
  if Key=VK_F5 {por exemplo} then ShowMessage ('Tecla F5');
end;
```

- ✓ **Retornando o nome do usuário:**

Em uses acrescente Registry, depois:

```
procedure GetUserCompany;
var
  Reg: TRegIniFle;
begin
  Reg:= TregIniFile.create ('software\microsoft\mssetup (acme)\');
  Edit1.text:=reg.readstring ('user info', 'defname','');
  Edit2.text:=reg.readstring ('user info','defcompany','');
  Reg.free;
end;
```

- ✓ **StatusBar:**

StatusBbar1.TPanels[0].Ttext:='Resultado: '+IntToStr(qtdias)+' dias úteis.'

Adicionar painel em F11 → Panels.

Cria uma caixa de erro:

```
if Trim(Edit1.Text)='' then Raise Exception.Create('Erro');
```

- ✓ **Retorna o path da aplicação:**

```
ShowMessage(Application.Exename);
```

- ✓ **Minimiza quando perde o foco:** Exemplo no programa ClipTray.

Declarar:

```
procedure Minimizar(Sender: TObject);
```

No programa:

```
procedure TForm1.Minimizar(Sender: TObject);
begin
  Application.Minimize;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  Application.OnDeactivate:=Minimizar;
end;
```

- ✓ **Ao desativar a aplicação, na barra de tarefas, executa os comandos:**

```
procedure TForm1.FormActivate(Sender: TObject);
begin
  Application.OnDeactivate:=Button1Click;
end;

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

```
    begin
    comandos ...
```

✓ **Tecla de atalho:**

```
procedure TForm1.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if (ssctrl in shift) and (chr(key) in ['A', 'a']) then showmessage('Ctrl+A');
end;
```

✓ **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, 'Texto não Transparente !');
      OldBkMode := SetBkMode(Handle, TRANSPARENT);
      TextOut(10, 50, 'Texto Transparente!');
      SetBkMode(Handle, OldBkMode);
    end;
end;
```

✓ **Limpar propriedades de componentes em modo de execução:**

```
for x:=0 to ComponentCount-1 do if Components[x].ClassName='Tedit' then
Tedit(Components[x]).Clear;
```

✓ **Acessando componentes sequencialmente:**

```
While x:=1 to 10 do Form1.FindComponent('SpeedButton' + IntToStr(i));
```

ScrollBar horizontal em um TListBox:

```
procedure TForm1.FormCreate(Sender: TObject);
  begin
    SendMessage(Listbox1.Handle, LB_SetHorizontalExtent, 1000, Longint(0));
  end;
```

Tab em TMemo:

Propriedade de WantTabs de TMemo True para habilitar as abas.

```
procedure TForm1.FormCreate (Sender: TObject);
const
  TabInc: LongInt = 10;
begin
  SendMessage (Memo1.Handle, EM_SetTabStops, 1, Longint (@TabInc));
end;
```

PageUp e PageDown por tela:

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

Preenchendo um TListBox ou TMemo:

```
Listbox1.Items.SetText (' aaaaa'#13'bbbbb'#13'ccccc ');
```

Exibe o seguinte em uma janela de listbox:

aaaaa
bbbbb
ccccc

Saber a posição atual em um TMemo:

```
var
  LineNum: longint;
  CharsBeforeLine: longint;
begin
  LineNum:=SendMessage(Memo1.Handle, EM_LINEFROMCHAR,
Memo1.SelStart,0);
  CharsBeforeLine:=SendMessage(Memo1.Handle, EM_LINEINDEX, LineNum, 0);
  Label1.Caption:='Linha ' + IntToStr(LineNum +1) ;
  Label2.Caption:='Posição ' + IntToStr((Memo1.SelStart - CharsBeforeLine) + 1);
end;
```

Adquirir o pixels por polegada da impressora:

```
VertPixelsPerInch := GetDeviceCaps(Printer.Handle, LogPixelsX);
HorzPixelsPerInch := GetDeviceCaps(Printer.Handle, LogPixelsY);
```

Excluir só o fsBold:

```
Canvas.Font.Style:=Canvas.Font.Style-[fsBold];
```

Fazer um TListBox ou TComboBox perder o foco do item:

```
Listbox1.ItemIndex:=-1;
```

Desfaz última ação em um TRichEdit:

```
RichEdit.Perform(EM_UNDO, 0, 0);
```

- ✓ **Definindo tipos:**

```
type
  Chave=(1,2,3,4,...);
   ...
var
  NrChave: Chave;
  ...
```

```
begin
  ...ord(NrChave);
```

PChar:

O tipo PChar é um ponteiro par aum vetor de caracteres e o seu tamanho é conhicido através da posição onde se encontra o caracter 0 – null-terminator.

- ✓ **StrPCopy** → Converte uma string em PChar;
- ✓ **StrPas** → Converte um vetor PChar em uma string;
- ✓ **StrCopy** & **StrCat** → Copia uma PChar para outro & concatena Pchars:

```
procedure TForm1.Button1Click(Sender: TObject);
var
  Buffer: PChar;
begin
  GetMem(Buffer, Length(Label1.Caption)+Length(Edit1.Text)+1);
```

→ Aloca memória suficiente para armazenar o texto do label1 e edit1.

```
  StrCopy(Buffer, PChar(Label1.Caption));
  StrCat(Buffer, PChar(Edit1.Text));
  Label1.Caption:=Buffer;
 Edit1.Clear;
 FreeMem(Buffer);
end;
```

Pausa:

```
var
  Numsec: smallint;
  Starttime: tdatetime;
begin
  Starttime:=now;
 Nunsec:=10;
 Repeat
  Application.processmessage;
 Until now>starttime+numsec*(1/24/60/60);
end;
```

- ✓ **MessageBox**: Exemplo no programa memória.

Através desta API, não é necessário usar a cláusula Dialog em uses e possui vários formatos. Veja o Help de API(SDK).

```
if Application.MessageBox('Deseja fechar?', 'Finalização', mb_IconQuestion +
mb_YesNo + mb_DefButton2)=IDYes then Close;
```

- ✓ **Memória**: Exemplo no programa memória.

```
procedure TForm1.BitBtn1Click(Sender: TObejct);
const cBytesPorMb=1024*1024;
var
  M: TMemoryStatus;
begin
  M.dwLength:=SizeOf(M);
  GlobalMemoryStatus(M);
  Memo1.Clear;
  with Memo1.Lines do
```

```
        begin
          Add(Format('Memória em uso: %d%%', [M.dwMemoryLoad]));
          Add(Format('Total de física: %f MB', [M.dwTotalPhys/cBytesPorMB]));
          Add(Format('Total máx. paginação: %f MB', [M.dwTotalPageFile/
cBytesPorMB]));
          Add(Format('Paginação disponíbel: %f MB', [M.dwAvailPageFile/
cBytesPorMB]));
          Add(Format('Total vitual: %fMB', [M.dwTotalVirtual/cBytesPorMB]));
          Add(Format('Virtual disponível: %fMB', [M.dwAvailVirtual/cBytesPorMB]));
        end;
      end;
```

✓ **Liberando memória(±0,6Mb)**: Exemplo nos programas ClipTray e Memória.
No evento **OnCreate** do form antes de qualquer outra expressão:

```
FreeLibrary(GetModuleHandle('OLEAUT32'));
FreeLibrary(GetModuleHandle('OLE32'));
```

E no evento **OnDestroy:**

```
LoadLibrary('OLEAUT32');
LoadLibrary('OLE32');
```

Verificar no System Information do Norton as DLL's utilizadas pelos programas clicando neles. Caso a DLL seja realmente necessária mesmo após a liberação, esta retorna a ser aberta – exclui a linha de liberação desta para evitar bugs.

✓ **Escondedo o aplicativo do CTRL+ALT+DEL**: Exemplo no programa memória.

1. Inicie o projeto com apenas 01 form;
2. Declare a função antes de implementation → **function** RegisterServiceProcess (dwProcessID, dwType: Integer): Integer; stdcall; **external** 'KERNEL32.DLL';
3. Adicione 02 botões;
4. No OnClick do Button1 → RegisterServiceProcess (GetCurrentProcessID, 1);
5. No OnClick do Button2 → RegisterServiceProcess (GetCurrentProcessID, 0);

✓ **Desligando o monitor:**

```
SendMessage(Application.Handle, wm_SysCommand, sc_MonitorPower, 0 ou -1);
```

✓ **Verificar se aplicativo está aberto:**

```
if IsWindow(FindWindow(nil, 'Delphi 5')) then...
```

→ Use **not** para obter o inverso.

✓ **Não mostrar a aplicação na TalkBar:** Exemplo no programa memória.
Acione Project – View Source e altere para:

```
uses Windows → Adicionar ao(s) o(s) existente(s)
{$R *.RES}
var
  ExtendedStyle: Integer;
begin
  Application.Initialize;
  ExtendedStyle:=GetWindowLong(Application.Handle, gwl_ExStyle);
  SetWindowLong(Application.Handle, gwl_ExStyle, ExtendedStyle or
```

```
ws_Ex_ToolWindow and not ws_Ex_AppWindow);
          Application.CreateForm(TForm1, Form1);
          Application.Run;
        end;
```

Provocando um clique automatico:

No evento OnDblClick button1.perform(wm_buttondown.0,0); em um componente que não seja Button1.

- ✓ **Acionando uma procedure em outro objeto:**

```
procedure TForm1.ListBox1KeyPress(Sender: TObject; var Key: Char);
begin
  if Key=#13 then
    begin
      Key:=#0;
      ListBox1DblClick(Sender);  //Se teclar ENTER, efetue a procedure abaixo
    end;
end;

procedure TForm1.ListBox1DblClick(Sender: TObject);
begin
  ShowMessage('DoubleClick acionado');
end;
```

- ✓ **Abrir e fechar drive de CD:** Exemplo no programa memória.

Declarar MMSystem e tentar com MediaPlayer1.Visible com True e False, com CD no drive. AutoOpen=True eDeviceType=dtCDAudio.

```
begin
  with MediaPlayer1 do
    if (MediaPlayer1.Mode=mpOpen) then mciSendCommand(MediaPlayer1.DeviceId,
MCI_SET, MCI_SET_DOOR_CLOSED, 0)
    else mciSendCommand(MediaPlayer1.DeviceId, MCI_SET,
MCI_SET_DOOR_OPEN, 0);
end;
```

- ✓ **Mover formulário sem título:** Exemplo no programa memória.

Declarar:

```
procedure WMNCHitTest(var Msg:TWMNCHitTest); message WM_NCHITTEST;
```

No programa:

```
procedure TForm1.WMNCHitTest(var Msg: TWMNCHitTest);
begin
  DefaultHandler(Msg);
  if Msg.Result=HTCLIENT then Msg.Result:=HTCAPTION;
end;
```

- ✓ **Tocar som sem MediaPlayer:** Exemplo no programa memória.

```
SndPlaySound('C:\WINDOWS\MEDIA\Notify.wav', SND_ASYNC);
```

- ✓ **Desativar ScreenSave:** Exemplo no programa memória.

Declarar:
procedure AppMessage(**var** Msg: TMsg; var Handled: Boolean);

No evento OnCreate:
Application.OnMessage:=AppMessage;

No programa:
procedure TForm1.AppMessage(**var** Msg: TMsg; var Handled: Boolean);
begin
if (Msg.Message=WM_SysCommand) **and** (Msg.wParam=SC_ScreenSave) **and** CheckBox1.Checked **then** Handled:=True;
end;

- ✓ **Desenhando relógio na barra de título:**

```
var
  x: array [0..79] of Char;
  y: String;
...
procedure TForm1.Timer1Timer(Sender: TObject);
var
  z: hDC;
begin
  z:=GetWindowDC(Handle);
  Rectangle(z, 4, 4, 70, 23);
  y:=TimeToStr(Time);
  StrPCopy(x, y); //Transforma y em Char
  TextOut(z, 10, 6, x, StrLen(x));
  ReleaseDC(Handle, z);
end;
```

- ✓ **Informações do HD:**

```
var
  Serial, Tamanho, Flags: DWord;
  Nome : Array[0..11] of Char;
  FAT: Array[0..5] of Char;
begin
  GetVolumeInformation('C:\', Nome, 12, @Serial, Tamanho, Flags, FAT, 6);
```

- ✓ **Hexa para inteiro:**
  Hexa:=StrtoInt('0x'+'FF'); → retorna 255, por exemplo.

- ✓ **Formulário redondo:** Exemplo no programa Desativa Proteção.

```
var
Circulo : THandle;
begin
  Circulo:=CreateEllipticRgn(1,1,32,32);
  SetWindowRgn(Handle,Circulo,True);
```

- ✓ **Criando um cursor:**
  Defina após **implementation** uma constante:

```
const
  NovoCursor=1;
```
Defina após *{$R *.DFM}* o arquivo que contém o cursor:

{$R Arquivo.res} → Necessário criar

No programa:

```
procedure TForm1.FormShow(Sender: TObject);
begin
  Screen.Cursors[NovoCursor]:=LoadCursor(hInstance, 'Sigma');
  Cursor:=NovoCursor;
end;
```

✓ **Trocar papel de parede:** Exemplo no programa Executor.

```
var
  Papel: String;
  Trocar: array [ 0..255] of char;
begin
  Randomize;
  Papel:='C:\Windows\Wallpaper\Wallpaper'+IntToStr(Random(10))+'.bmp';
  if FileExists(Papel) then
    begin
      Registro:=TRegIniFile.Create;
      Registro.RootKey:=HKEY_CURRENT_USER;
      Registro.OpenKey('Control Panel\Desktop',False);
      Registro.WriteString('Wallpaper',Papel);
      StrPCopy(Trocar,Papel);
      SystemParametersInfo(SPI_SETDESKWALLPAPER,0,@Trocar,0);
      Registro.Free;
    end;
```

✓ **Não mostrar formulário principal:** Exemplo no programa Executor.

No dpr inclua Application.ShowMainForm:=False;

✓ **Abrir correio eletrônico:** Exemplo no programa InfoListas.

Declarar em uses após implematation: ShellAPI;

```
var
  St: Array [0..255] of Char;
begin
  ShellExecute(Handle, 'Open', StrPCopy(St, 'MailTo:'+Table1e_mail.AsString), nil,
nil, SW_Show);
end;
```

SQL:

→ Select * from Table1 left join Table2 on Table1.KeyField = Table2.ExternalKey;

→ Select * from Table1 left outer Table2 on Table1.KeyField = Table2.KeyField;

→ Select * from Table1, Table2 where Table1.KeyField = Table2.KeyField;

✓ **Para verificar se operação foi concretizada:**

```
var
  Temp: Boolean;
begin
```

```
Temp:=StrToDate('12/02/2001');
if Temp then Showmessage('OK') else Showmessage('Erro!');
```

- ✓ **Nome do computador na rede:**
  Registro.OpenKey('System\CurrentControlSet\Services\VXD\VNETSUP',false);

- ✓ **Fecha um formulário ativo se este não for visível:**
  if not(Screen.ActiveForm.Visible) then Screen.ActiveForm.Close;

Converte Ascii em String:

```
var
  btI: Byte;
  stIn, stOut: string;
begin
  stIn:='ABCDE';
  stOut:= '';
  for btI :=1 to Length(stIn) do stOut:=stOut+IntToStr(Ord(stIn[btI]));
  ShowMessage(stOut);
```

- ✓ **Verificar teclado em todos os formulários:**
  Declarar:
  procedure Teclado(**var** Msg: TWMKey; **var** Resposta: Boolean);

```
procedure TPagesDlg.Teclado(var Msg: TWMKey; var Resposta: Boolean);
begin
  Operações...
end;
```

OnCreate do programa:
Application.OnShortCur:=Teclado;

- ✓ **Fecha um programa:**
  if IsWindow(FindWindow(**nil**,'Calculadora')) **then**
PostMessage(FindWindow(**nil**,'Calculadora'), WM_QUIT,0,0);
  ou usar no FindWindow('OpusApp',**nil**) para o Word, por exemplo. Verificar os nomes no WinSigth32.

Enter como Tab:

```
procedure TForm1.DoEnterAsTab(var: Msg: TMsg; var Handled: Boolean);
begin
  if Msg.Message=VK_KEYDOWN then if Msg.wParam=VK_RETURN then
Keybd_Event(VK_TAB,0,0,0);
end;
```

- ✓ **Programa sem form:** Apenas 16Kb.
  Digitar direto no dpr, não usar unit:

```
program Project1;

uses
```

```
  Windows;

procedure Som;
begin
  MessageBeep(16);
end;

begin
  Som;
end.
```

- ✓ **Verificar pressionamento do mouse:**

```
if (GetKeyState(VK_LBUTTON)=-127) or (GetKeyState(VK_LBUTTON)=-128)
then ShowMessage('botão esquerdo');
if (GetKeyState(VK_RBUTTON)=-127) or (GetKeyState(VK_RBUTTON)=-128)
then ShowMessage('Botão direito');
```

- ✓ **Processar mensagens pendentes:**

```
Application.ProcessMessages;
```

Exponenciação

$X^y = \exp(\ln(x)*y)$

- ✓ **Atributos:**

```
SetFileAttributes(Arquivo,X);
```

X → 0 – Nada
X → 1 – Somente leitura
X → 2 – Oculto
X → 3 – Somente leitura e oculto
X → 4 – Sistema
X → 5 – Somente leitura e sistema
X → 6 – Oculto e sistema
X → 7 – Somente leitura, oculto e sistema

- ✓ **Desativando Alt+F4:**

```
procedure Form1.OnCloseQuery...
begin
  CanClose:=False;
end;
```

- ✓ **Retornando o tamanho de um arquivo:**

```
function TForm1.Tamanho(Arquivo: String): Integer;
begin
  with TFileStream.Create(Arquivo, fmOpenRead or fmShareExclusive) do try
Result:=Size finally Free end;
end;
```

Cálculo do Valor de Custo:

Entrada de	1 a 4 campos	5 a 20 campos	+20 campos

Dados			
1 tabela	3	3	4
2 tabelas	3	4	6
+2 tabelas	4	4	6

Saída de Dados	**1 a 5 campos**	**6 a 20 campos**	**+20 campos**
1 tabela	4	4	5
2 tabelas	4	5	7
+2 tabelas	5	7	7

Arquivos Internos	**1 a 4 campos**	**5 a 20 campos**	**+20 campos**
Por tabela	5	7	10

Atribuir peso de 0 a 5 para:
Comunicação de dados
Processamento
Performance
Utilização do equipamento
Volume de transações
Entrada de dados on-line
Eficiência do usuário final
Atualização on-line
Processamento complexo
Reutilização do código
Facilidade de implementação
Facilidade operacional
Múltiplos locais
Facilidade de mudanças

Complexidade do Banco de Dados	
Simples	0,65
Médio	0,80
Complexo	1,00

Complexidade de Alteração	
Baixa	0,20
Regular	0,40
Média	0,50
Alta	0,75
Elevada	1,00
Complexa	1,25

- ✓ **SQL:**

  Criar Database
  CREATE DATABASE ‘C:\CLIENTES.GDB’
  USER ‘SYSDBA’
  PASSWORD’masterkey’;

Criar tabela

```
CREATE TABLE CLIENTES
(
 CODIGO          INTEGER,
 NOME            VARCHAR(45)
);
```

Alterar nome do campo

```
ALTER TABLE CLIENTES
ALTER NOME TO NOME_CLIENTE;
```

Alterar tipo do campo

```
ALTER TABLE CLIENTES
ALTER NOME TYPE CHAR(50);
```

CHAVE PRIMÁRIA

```
CREATE TABLE CLIENTES
(
CODIGO          INTEGER NOT NULL,
NOME            VARCHAR(45),
CONSTRAINT PKCODIGO PRIMARY KEY (CODIGO)
);
```

CHAVE ESTRANGEIRA

```
CREATE TABLE DEPENDENTES
(
CODIGO          INTEGER NOT NULL,
CODIGO_CLIENTE INTEGER
NOME            VARCHAR(45),
CONSTRAINT PKCODIGO PRIMARY KEY (CODIGO)
CONSTRAINT FKCODIGO FOREIGN KEY (CODIGO_CLIENTE)
REFERENCES CLIENTES (CODIGO)
);
```

APAGAR CHAVE

```
ALTER TABLE CLIENTES
DROP CONSTRAINT PKCODIGO;
```

COUNT

```
SELECT COUNT(*) FROM CLIENTES
```

UPDATE

```
UPDATE CLIENTES SET TELEFONE='22-98120656' WHERE NOME='SÁVIO'
```

IN ou OR

```
SELECT * FROM CLIENTES WHERE NOME IN ('SÁVIO','CLER')
SELECT * FROM CLIENTES WHERE NOME ='SÁVIO'OR NOME='CLER'
```

DISTINTIC

SELECT DISTINTIC CEP FROM CLIENTES – Retorna todos os CEP distintos, sem repetição

JOIN
SELECT CLIENTES.NOME, NFSAIDA.PRODUTO FROM CLIENTES JOIN NFSAIDA ON CLIENTES.CHAVE=NFSAIDA.CHAVE_CLIENTE
Seleciona todos os produtos vendidos, por cliente, onde PK=FK.

SELECT * FROM FUNCIONARIOS

JOIN (((PESSOAIS JOIN PROFISIONAIS ON

PESSOAIS.MATRICULA_FUNCIONARIO =

PROFISSIONAIS.MATRICULA_FUNCIONARIOS)

JOIN FALTAS ON PESSOAIS.MATRICULA_FUNCIONARIOS =

FALTAS.MATRICULA_FUNCIONARIOS)

JOIN FERIAS ON PESSOAIS.MATRICULA_FUNCIONARIOS =

FÉRIAS.MATRICULA_FUNCIONARIOS)

ON PROFISSIONAIS.MATRICULA_FUNCIONARIOS =

FUNCIONARIOS.MATRICULA WHERE ...

Faz um JOIN entre PESSOAIS e PROFISSIONAIS, como o resultado deste,
Faz um JOIN entre ESTE e FALTAS, com o resultado deste,
Faz um JOIN entre ESTE e FERIAS, com o resultado deste e por último
Faz um JOIN entre ESTE e FUNCIONARIOS, na linha em negrito.

LEFT JOIN
Apresenta todos os clientes, mesmo sem produtos

RIGHT JOIN
Apresenta todos os produtos, mesmo sem clientes

INNER JOIN
Apresenta todos os clientes e produtos. Igual a LEFT+RIGHT

SELECT
SELECT * FROM CLIENTES.NOME AS ‘Nome do Cliente’ WHERE NOME=’JOÃO’ – Apresenta ‘Nome do Cliente’ como título do DBGrid

SELECT CIDADE, COUNT (CIDADE) FROM PESSOAIS GROUP BY CIDADE

SELECT CARGO, COUNT(CARGO), SECRETARIA FROM PROFISSIONAIS GROUP BY CARGO, SECRETARIA

Quantos cargos em cada secretaria

SELECT CARGO, SUM(SALÁRIO) AS SOMA, SECRETARIA FROM PROFISSIONAIS GROUP BY CARGO, SECRETARIA

Soma dos salários por cargo e secretaria

SELECT CARGO, SUM(SALÁRIO*2) AS SOMA, SECRETARIA FROM PROFISSIONAIS WHERE SECRETARIA = 'OBRAS' GROUP BY CARGO, SECRETARIA HAVING SALÁRIO>0 AND SALÁRIO<1000

SELECT, CARGO, SEXO FROM PROFISSIONAIS JOIN FUNCIONÁRIOS ON PROFISSIONAIS.MATRICULA_FUNCIONARIO = FUNCIONÁRIO.MATRICULA GROUP BY CRGO, SEXO

Cargos por sexo

SELECT CARGO, SALÁRIO, SEXO FROM PROFISSIONAIS JOIN FUNCIONÁRIOS ON PROFISSIONAIS.MATRICULA_FUNCIONARIO = FUNCIONÁRIO.MATRICULA GROUP BY CARGO, SALÁRIO, SEXO ORDER BY SALÁRIO DESC

Ordem decrescente

SELECT * FROM NFSAIDA WHERE VALOR > (SELECT VALOR_VENDA FROM NFSAIDA WHERE PRODUTO ='CANETA BIC')

Todas as notas onde o valor total seja igual ao valor de venda da CANETA BIC

SELECT * FROM CLIENTES WHERE EXISTS (SELECT * FROM NFSAIDA WHERE CLIENTES.CHAVE = NFSAIDA.CHAVE_CLIENTE) ORDER BY NOME

Todos os clienest que possuem nota de venda

SELECT 'Nome: '||NOME||'Título: '||TITULO||'Zona: '||ZONA FROM FUNCIONÁRIOS WHERE NOME LIKE '%M%A%'

Seleciona dados e concatena de funcionários que nome tenha em qualquer parte em ordem M e A

DELETE FROM NFSAIDA WHERE CHAVE_CLIENTE = (SELECT CHAVE FROM CLIENTES WHERE NOME = 'SAVIO'

DELETE FROM NFSAIDA WHERE NOT EXISTS(SELECT * FROM CLIENTES)

Apaga todas as notas de ex-clientes

SELECT NOME, PRODUTO FROM CLIENTES JOIN NFSAIDA ON CLIENTES.CHAVE = NFSAIDA.CHAVE_CLIENTE AND PRODUTO LIKE '%CD%'