



UNIVERSIDADE SALGADO DE OLIVEIRA
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO
TEORIA DA COMPUTAÇÃO E ALGORITMOS
Prof. Giuliano Prado de Moraes Giglio

Linguagem Técnica de Programação

DELPHI

Básico

2º Semestre de 2003

Programação Windows

DELPHI

- Características do Delphi
- Ambiente de Desenvolvimento Delphi
- Desenvolvendo Aplicativos
- Principais Procedimentos e Funções Pré-definidas
- Componentes - Propriedades, Eventos e Métodos

Aplicações MDI

- Use de Modelos e Experts

Os objetos

- Application
- Screen
- Printer
- Canvas

Programação Windows

Características

- Ambiente Gráfico (WYSIWYG)
- Ênfase a Interface com o usuário
- Orientação a Eventos
- Interface com o hardware definida em APIs (Interface de Programação de Aplicação)

Baseado em Janelas

Forma de Apresentação	Descrição
Não-Modal	Não interfere nas demais janelas
Modal	Desabilita a aplicação em execução até que seja fechada
System-Modal	Paralisa todo o sistema até que seja fechada

Tipo	Descrição
Principal	É a aplicação em si.
Caixa de Diálogo	Janela modal utilizada para entrada de informações
MDI Principal	Janela que pode conter outras janelas em sua área cliente
MDI Child	Janela que está contida na área cliente de outra janela
Mensagem	Janela modal que informa o usuário sobre a execução de uma tarefa

Tratamento a Eventos

- O sistema ou ambiente operacional e os usuários interagem com os aplicativos através de eventos
- Os eventos são reconhecidos pelo sistema, transformados em informações úteis aos programas (mensagens) e despachados para a janela alvo, a qual irá ou não tratá-lo
- Para Cada programa existe um loop de mensagens permitindo que este receba suas mensagens até que chegue uma ordenando a finalização do programa

Mensagens

Tipo	Descrição
Hardware	Entrada de Mouse e Teclado
Manutenção da Janela	Notificação, Solicitação de ação, Consulta
Manutenção da Interface	Menu, Ponteiro de mouse, barra de rolagem, quadros de diálogo, MDI
Terminação	Encerramento do sistema ou da aplicação
Privado	Controles de caixas de diálogo
Notificação de Recursos do Sistema	Alteração de cor, fonte, spooler, modos dos dispositivos
Compartilhamento de Dados	Área de transferência, DDE e OLE
Interna do Sistema	Mensagens não documentadas

Projeto de Interface

Padronização

- Tomar como base as aplicações existentes, principalmente as da Microsoft.
- Utilizar menus com as opções Arquivo, Editar, Janela e Ajuda.
- Usar botões de acesso rápido.
- Utilizar teclas de atalho.
- Enfatizar imagens.

Esquecer a programação DOS

- Não utilizar múltiplos níveis de menu.
- Não exagerar em cores diferentes.
- Utilizar a tecla TAB para passar de um campo para outro.
- Utilizar a tecla ENTER ou um botão para executar ações.

Simplicidade

- Procurar facilidade de uso.
- Permitir liberdade de ação.
- Permitir diferentes maneiras de se alcançar o mesmo resultado.
- Procurar as soluções mais intuitivas.
- Procurar adotar os símbolos que o usuário já esteja acostumado.

DELPHI

Características do Delphi

- Gera um executável verdadeiro, independente de *run-time*.
- Utiliza um dialeto da linguagem Object Pascal para escrever os procedimentos do programa.
- Utiliza o processo de desenvolvimento *Two-Way*, que permite tanto escrever o código em Object Pascal gerando os objetos visuais, como utilizar os métodos visuais gerando código em Object Pascal.
- Os componentes são definidos como objetos, o que permite a herança.
- Permite a criação de novos componentes na própria linguagem.
- Possui acesso facilitado a banco de dados.
- Possui ambiente de depuração integrado.
- Possui componenete para a internet

Características da Programação Delphi

- Um programa Delphi é uma estrutura de aplicativo
- Orientada ao desenho de formulários ou janelas.
- Interface com usuário feita através de componentes
- Contém um conjunto de controles pré-desenvolvidos que dão acesso às características do sistema.
- Os componentes são objetos
- Cada controle ou componente possui propriedades, métodos e pode responder a eventos.
- As propriedades podem ter seus valores definidos em tempo de desenvolvimento e alterados em tempo de execução.
- Os eventos são as mensagens que cada componente pode responder, tendo associado a eles um procedimento de evento.

Elementos da Programação Delphi :

Elemento	Descrição
Formulário (Form)	É uma janela, elemento básico onde agrupamos os componentes para formar a interface com o usuário.
Unidade (Unit)	Arquivo que contém código em object pascal. Para cada formulário existe uma unidades associada
Componente	Objetos utilizados para a construção das nossas aplicações (projeto).
Propriedade	Representam os atributos dos componentes
Método	Procedimento ou função própria do objeto.
Evento	Representam a capacidade de resposta dos componentes aos estímulos
Processador de Evento	Procedimento responsável por responder a determinado evento
Projeto (Project)	Conjunto de formulários, componentes e unidades que compõem uma aplicação

Arquivos Produzidos pelo Sistema

EXT	Tipo	Descrição
BMP, ICO	Arquivos gráficos	Arquivos de imagens nos formatos BitMaP e ICOne
DCU	Unidade Compilada Delphi	Resultam da compilação de um arquivo PAS
DFM	Arquivo de formulário gráfico	Arquivo binário contendo as propriedades e componentes de um formulário
~DF	Backup de DFM	Backup de um arquivo DFM
DPR	Arquivo de Projeto	Escrito em Object Pascal contendo os componentes de um projeto e permite uso de código de inicialização do projeto
~DP	Backup de Projeto	Backup de um arquivo DPR
DSK	Configurações de Desktop	Arquivo texto contendo as informações sobre a posição das janelas, os arquivos abertos no editor e outras configurações de Desktop
DSM	Dados do Object Browser	Armazena as informações do Object Browser
EXE	Arquivo executável linkeditado	Arquivo executável contendo as unidades, recursos e formulários compilados de um projeto
OPT	Opções do Projeto	Arquivo de teste com as configurações atuais para as opções do projeto
PAS	Código-fonte de uma unidade	Arquivo contendo o código fonte de uma unit em object pascal, o qual pode ser de um formulário ou arquivo fonte independente. Sendo de um formulário contém a sua definição de classe e código dos seus manipuladores de eventos
~PA	Backup de um PAS	Backup de um arquivo PAS
RES	Arquivo de recursos compilado	Arquivo binário associado ao projeto contendo recursos compilados, por padrão contem o ícone do projeto

Estrutura de um Projeto : **Projeto (*.DPR), Units (*.PAS) e Forms (*.DFM)**

Ambiente de Desenvolvimento Delphi

- Menu Principal
- Speed Bar
- Paleta de Componentes
- Object Inspector (Propriedades e Eventos)
- Editor de Formulário
- Editor de Código
 - Multiplas Janelas (View/New Edit Window)
 - Complemeto do Código (Exibe automaticamente as Propriedades e Métodos dos Objetos)
 - Parâmetros Automáticos (Exibe automaticamente os Parâmetros de Métodos)
 - Vizualização automática de variáveis na depuração
 - Templates de Código (Tools/Environment Options/Code Insight – Ctrl+J)
 - Busca de Texto por todo o Projeto (Search/Find in Files)
 - Marcando Blocos em colunas (Alt+Seleção)
 - Marcando Posições no Código (Ctrl+K+n = Marca, Ctrl+Q+n = Movimenta)

Desenvolvendo Aplicativos

Manipulando Componentes:

- Acrescentando
- Seleccionando
- Movendo
- Sobrepondo
- Alinhando
- Redimensionando
- Cortando, Copiando e Colando
- Criando Ordem de Acesso (Tab Order)
- Fixando

Alterando Propriedades dos Componentes em Tempo de Projeto

1. Selecione o Componente, depois mude a propriedade no Object Inspector
2. Editor de Propriedade : Simples, Lista Suspensa, Caixa de Dialogo e Propriedades Aninhadas

Escrevendo Procedimentos de Eventos

Selecione o Componente, escolha, na Object Inspector, o evento a ser respondido e Click-Duplo na coluna direita, e então escreva o procedimento

Alterando Propriedades em Tempo de execução

Componente.Propriedade := NovoValor

OBS: Quando a propriedade for aninhada use

Componente.Propriedade := Componente.Propriedade ± [ConstanteDaPropriedade]

Utilizando métodos

Componente.Método

Estrutura de uma Unidade

```
unit Unit1;
```

interface

```
uses
```

```
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,  
  Forms, Dialogs;
```

```
type
```

```
  TForm1 = class(TForm)
```

```
    private
```

```
      { Private declarations }
```

```
    public
```

```
      { Public declarations }
```

```
  end;
```

```
var
```

```
  Form1: TForm1;
```

implementation

```
{$R *.DFM}
```

```
end.
```

Principais Procedimentos e Funções Pré definidas

procedure Beep;

Toca um beep

procedure ChDir(S: string);

Troca o diretório corrente para o diretório especificado em S.

```
begin
  {$I-}
  { Change to directory specified in Edit1 }
  ChDir(Edit1.Text);
  if IOResult <> 0 then
    MessageDlg('Cannot find directory', mtWarning, [mbOk], 0);
end;
```

function Chr(X: Byte): Char;

Retorna o caracter com o código ASCII X

```
begin
  Canvas.TextOut(10, 10, Chr(65)); { The letter 'A' }
end;
```

function Concat(s1 [, s2,..., sn]: string): string;

Concatena as strings

```
var
  S: string;
begin
  S := Concat('ABC', 'DEF'); { 'ABCDE' }
end;
```

function Copy(S: string; Index, Count: Integer): string;

Retorna uma substring de S, começando a partir de Index e tendo Count caracteres

The Copy function returns a substring of a string.

```
var S: string;
begin
  S := 'ABCDEF';
  S := Copy(S, 2, 3); { 'BCD' }
end;
```

function CreateDir(const Dir: string): Boolean;

Cria um novo diretório e retorna o sucesso da operação

function Date: TDateTime;

Retorna a Data atual

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  Label1.Caption := 'Today is ' + DateToStr(Date);
end;
```

function DateToStr(Date: TDateTime): string;

Converte Data para String

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  Label1.Caption := DateToStr(Date);
end;
```

```
function DayOfWeek(Date: TDateTime): Integer;
```

Retorna o dia da semana especificado entre 1 e 7, onde domingo é um e Sábado é 7

```
procedure TForm1.Button1Click(Sender: TObject);
var
  ADate: TDateTime;
begin
  ADate := StrToDate(Edit1.Text);
  Label1.Caption := 'Day ' + IntToStr(DayOfWeek(ADate)) + ' of the week';
end;
```

```
procedure DecodeDate(Date: TDateTime; var Year, Month, Day: Word);
```

Quebra os valores especificados no parâmetro Date em Year, Month e Day.

```
procedure TForm1.Button1Click(Sender: TObject);
var
  Present: TDateTime;
  Year, Month, Day, Hour, Min, Sec, MSec: Word;
begin
  Present := Now;
  DecodeDate(Present, Year, Month, Day);
  Label1.Caption := 'Today is Day ' + IntToStr(Day) + ' of Month '
    + IntToStr(Month) + ' of Year ' + IntToStr(Year);
  DecodeTime(Present, Hour, Min, Sec, MSec);
  Label2.Caption := 'The time is Minute ' + IntToStr(Min) + ' of Hour '
    + IntToStr(Hour);
end;
```

```
procedure DecodeTime(Time: TDateTime; var Hour, Min, Sec, MSec: Word);
```

Quebra os valores especificados em Time nos parâmetros Hour, Min, Sec e MSec.

```
procedure TForm1.Button1Click(Sender: TObject);
var
  Present: TDateTime;
  Year, Month, Day, Hour, Min, Sec, MSec: Word;
begin
  Present := Now;
  DecodeDate(Present, Year, Month, Day);
  Label1.Caption := 'Today is Day ' + IntToStr(Day) + ' of Month '
    + IntToStr(Month) + ' of Year ' + IntToStr(Year);
  DecodeTime(Present, Hour, Min, Sec, MSec);
  Label2.Caption := 'The time is Minute ' + IntToStr(Min) + ' of Hour '
    + IntToStr(Hour);
end;
```

```
procedure Delete(var S: string; Index, Count: Integer);
```

Remove a substring de Count caracteres da string S partir da posição Index

```
var
  s: string;
begin
  s := 'Honest Abe Lincoln';
  Delete(s, 8, 4);
  Canvas.TextOut(10, 10, s); { 'Honest Lincoln' }
end;
```

```
function DeleteFile(const FileName: string): Boolean;
```

Apaga o arquivo FileName do disco. Se o arquivo não puder ser apagado a função retorna False.

```
DeleteFile('DELETE.ME');
```

```
function DirectoryExists(Name: string): Boolean;
```

Verifica se Name diretório existe

```
function DiskFree(Drive: Byte): Integer;
```

Retorna o número de bytes livre no driver especificado em Drive.

Onde : 0 = Corrente, 1 = A, 2 = B,...

DiskFree retorna -1 se o driver for inválido

```
var
  S: string;
begin
  S := IntToStr(DiskFree(0) div 1024) + ' Kbytes free.';
  Canvas.TextOut(10, 10, S);
end;
```

```
function DiskSize(Drive: Byte): Integer;
```

Retorna o tamanho em bytes do driver especificado.

Onde : 0 = Corrente, 1 = A, 2 = B,...

DiskFree retorna -1 se o driver for inválido

```
var
  S: string;
begin
  S := IntToStr(DiskSize(0) div 1024) + ' Kbytes capacity.';
  Canvas.TextOut(10, 10, S);
end;
```

```
function EncodeDate(Year, Month, Day: Word): TDateTime;
```

Retorna uma Data formada por Year, Month e Day

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

```
var
  MyDate: TDateTime;
begin
  MyDate := EncodeDate(83, 12, 31);
  Label1.Caption := DateToStr(MyDate);
end;
```

```
function EncodeTime(Hour, Min, Sec, MSec: Word): TDateTime;
```

Retorna a Hora formada por Hour, Min, Sec e MSec

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

```
var
  MyTime: TDateTime;
begin
  MyTime := EncodeTime(0, 45, 45, 7);
  Label1.Caption := TimeToStr(MyTime);
end;
```

```
function ExtractFileDir(const FileName: string): string;
```

Retorna o diretório adequado para ser passado para as funções CreateDir, GetCurrentDir, RemoveDir e SetCurrentDir.

O resultado da função é uma string vazia se FileName não contiver um drive e um caminho.

```
function ExtractFileDrive(const FileName: string): string;
```

Retorna uma string contendo o drive do path de um arquivo.

```
function ExtractFileExt(const FileName: string): string;
```

Retorna a extensão do arquivo FileName

```
function ExtractFileName(const FileName: string): string;
```

Retorna o nome do arquivo

```
Form1.Caption := 'Editing ' + ExtractFileName(FileName);
```

```
function ExtractFilePath(const FileName: string): string;
```

Retorna o Path de um arquivo

```
ChDir(ExtractFilePath(FileName));
```

```
function FileAge(const FileName: string): Integer;
```

Retorna a data e a hora de um arquivo num valor que pode ser convertido para TDateTime através da função FileDateToDateTime. Retorna -1 se o arquivo não existir

```
function FileExists(const FileName: string): Boolean;
```

Retorna verdade se o arquivo existir

```
if FileExists(FileName) then
  if MsgBox('Do you really want to delete ' + ExtractFileName(FileName)
    + '?', []) = IDYes then DeleteFile(FileName);
```

```
function FileSize(var F): Integer;
```

Retorna o tamanho de um arquivo, para usar FileSize o arquivo deve estar aberto. Se o arquivo estiver vazio FileSize(F) retorna 0. F é uma variável do tipo arquivo. FileSize não pode ser usada com arquivo texto

```
var
  f: file of Byte;
  size : Longint;
  S: string;
  y: integer;
begin
  if OpenDialog1.Execute then begin
    AssignFile(f, OpenDialog1.FileName);
    Reset(f);
    size := FileSize(f);
    S := 'File size in bytes: ' + IntToStr(size);
    y := 10;
    Canvas.TextOut(5, y, S);
    y := y + Canvas.TextHeight(S) + 5;
    S := 'Seeking halfway into file...';
    Canvas.TextOut(5, y, S);
    y := y + Canvas.TextHeight(S) + 5;

    Seek(f, size div 2);
    S := 'Position is now ' + IntToStr(FilePos(f));
    Canvas.TextOut(5, y, S);
    CloseFile(f);
  end;
end;
```

```
procedure FillChar(var X; Count: Integer; value: Byte);
```

Preenche um vetor com determinado caracter. Value pode ser um byte ou char

```
var
  S: array[0..79] of char;
begin
  { Set to all spaces }
  FillChar(S, SizeOf(S), ' ');
end;
```

```
function FloatToStr(Value: Extended): string;
```

Converte um valor em ponto flutuante (real) para uma string

```
procedure ForceDirectories(Dir: string);
```

Cria múltiplos diretórios de uma só vez

```
procedure TForm1.Button1Click(Sender: TObject);
var
  Dir: string;
begin
  Dir := 'C:\APPS\SALES\LOCAL';
  ForceDirectories(Dir);
  if DirectoryExists(Dir) then
    Label1.Caption := Dir + ' was created'
end;
```

```
function FormatDateTime(const Format: string; DateTime: TDateTime): string;
```

Formata o valor DateTime usando o formato de Format.

Os especificadores de formato abaixo são válidos

Specifier	Displays
c	Displays the date using the format given by the ShortDateFormat global variable, followed by the time using the format given by the LongTimeFormat global variable. The time is not displayed if the fractional part of the DateTime value is zero.
d	Displays the day as a number without a leading zero (1-31).
dd	Displays the day as a number with a leading zero (01-31).
ddd	Displays the day as an abbreviation (Sun-Sat) using the strings given by the ShortDayNames global variable.
dddd	Displays the day as a full name (Sunday-Saturday) using the strings given by the LongDayNames global variable.
dddddd	Displays the date using the format given by the ShortDateFormat global variable.
dddddd	Displays the date using the format given by the LongDateFormat global variable.
m	Displays the month as a number without a leading zero (1-12). If the m specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.
mm	Displays the month as a number with a leading zero (01-12). If the mm specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.
mmm	Displays the month as an abbreviation (Jan-Dec) using the strings given by the ShortMonthNames global variable.
mmmm	Displays the month as a full name (January-December) using the strings given by the LongMonthNames global variable.
yy	Displays the year as a two-digit number (00-99).
yyyy	Displays the year as a four-digit number (0000-9999).
h	Displays the hour without a leading zero (0-23).
hh	Displays the hour with a leading zero (00-23).
n	Displays the minute without a leading zero (0-59).
nn	Displays the minute with a leading zero (00-59).
s	Displays the second without a leading zero (0-59).
ss	Displays the second with a leading zero (00-59).
t	Displays the time using the format given by the ShortTimeFormat global variable.
tt	Displays the time using the format given by the LongTimeFormat global variable.
am/pm	Uses the 12-hour clock for the preceding h or hh specifier, and displays 'am' for any hour before noon, and 'pm' for any hour after noon. The am/pm specifier can use lower, upper, or mixed case, and the result is displayed accordingly.
a/p	Uses the 12-hour clock for the preceding h or hh specifier, and displays 'a' for any hour before noon, and 'p' for any hour after noon. The a/p specifier can use lower, upper, or mixed case, and the result is displayed accordingly.
ampm	Uses the 12-hour clock for the preceding h or hh specifier, and displays the contents of the TimeAMString global variable for any hour before noon, and the contents of the TimePMString global variable for any hour after noon.
/	Displays the date separator character given by the DateSeparator global variable.
:	Displays the time separator character given by the TimeSeparator global variable.
'xx'/'xx'	Characters enclosed in single or double quotes are displayed as-is, and do not affect formatting.

Format specifiers may be written in upper case as well as in lower case letters--both produce the same result.

If the string given by the Format parameter is empty, the date and time value is formatted as if a 'c' format specifier had been given.

```
S := FormatDateTime('"The meeting is on" dddd, mmmm d, yyyy, ' +
  ' "at" hh:mm AM/PM', StrToDateTime('2/15/95 10:30am'));
```

```
function FormatFloat(const Format: string; Value: Extended): string;
```

Transforma um Float numa string usando a formatação contida em Format

Os especificadores de formato abaixo são válidos

Specifier	Represents
0	Digit place holder. If the value being formatted has a digit in the position where the '0' appears in the format string, then that digit is copied to the output string. Otherwise, a '0' is stored in that position in the output string.
#	Digit placeholder. If the value being formatted has a digit in the position where the '#' appears in the format string, then that digit is copied to the output string. Otherwise, nothing is stored in that position in the output string.
.	Decimal point. The first '.' character in the format string determines the location of the decimal separator in the formatted value; any additional '.' characters are ignored. The actual character used as a the decimal separator in the output string is determined by the DecimalSeparator global variable. The default value of DecimalSeparator is specified in the Number Format of the International section in the Windows Control Panel.
,	Thousand separator. If the format string contains one or more ',' characters, the output will have thousand separators inserted between each group of three digits to the left of the decimal point. The placement and number of ',' characters in the format string does not affect the output, except to indicate that thousand separators are wanted. The actual character used as a the thousand separator in the output is determined by the ThousandSeparator global variable. The default value of ThousandSeparator is specified in the Number Format of the International section in the Windows Control Panel.
E+	Scientific notation. If any of the strings 'E+', 'E-', 'e+', or 'e-' are contained in the format string, the number is formatted using scientific notation. A group of up to four '0' characters can immediately follow the 'E+', 'E-', 'e+', or 'e-' to determine the minimum number of digits in the exponent. The 'E+' and 'e+' formats cause a plus sign to be output for positive exponents and a minus sign to be output for negative exponents. The 'E-' and 'e-' formats output a sign character only for negative exponents.

'xx'/'xx" Characters enclosed in single or double quotes are output as-is, and do not affect formatting.

; Separates sections for positive, negative, and zero numbers in the format string.

The locations of the leftmost '0' before the decimal point in the format string and the rightmost '0' after the decimal point in the format string determine the range of digits that are always present in the output string.

The number being formatted is always rounded to as many decimal places as there are digit placeholders ('0' or '#') to the right of the decimal point. If the format string contains no decimal point, the value being formatted is rounded to the nearest whole number.

If the number being formatted has more digits to the left of the decimal separator than there are digit placeholders to the left of the '.' character in the format string, the extra digits are output before the first digit placeholder.

To allow different formats for positive, negative, and zero values, the format string can contain between one and three sections separated by semicolons.

One section: The format string applies to all values.

Two sections: The first section applies to positive values and zeros, and the second section applies to negative values.

Three sections: The first section applies to positive values, the second applies to negative values, and the third applies to zeros.

If the section for negative values or the section for zero values is empty, that is if there is nothing between the semicolons that delimit the section, the section for positive values is used instead.

If the section for positive values is empty, or if the entire format string is empty, the value is formatted using general floating-point formatting with 15 significant digits, corresponding to a call to FloatToStrF with the ffGeneral format. General floating-point formatting is also used if the value has more than 18 digits to the left of the decimal point and the format string does not specify scientific notation.

0	1234	-1234	1	0
0.00	1234.00	-1234.00	0.50	0.00
###	1234	-1234	.5	
###0.00	1,234.00	-1,234.00	0.50	0.00
###0.00;(###0.00)	1,234.00	(1,234.00)	0.50	0.00
###0.00;;Zero	1,234.00	-1,234.00	0.50	Zero
0.000E+00	1.234E+03	-1.234E+03	5.000E-01	0.000E+00
#####E-0	1.234E3	-1.234E3	5E-1	0E0

```
function Frac(X: Real): Real;
```

Retorna a parte fracional do parâmetro X

```
var
  R: Real;
begin
  R := Frac(123.456);    { 0.456 }
  R := Frac(-123.456);  { -0.456 }
end;
```

```
function GetCurrentDir: string;
```

Retorna uma string contendo o diretório corrente

```
procedure GetDir(D: Byte; var S: string);
```

Retorna o diretório corrente do driver especificado.

O onde D pode ser :

Value	Drive
0	Corrente
1	A
2	B
3	C

```
var
  s : string;
begin
  GetDir(0,s); { 0 = Current drive }
  MessageDlg('Current drive and directory: ' + s, mtInformation, [mbOk] , 0);
end;
```

```
procedure Inc(var X [ ; N: Longint ] );
```

Incrementa de uma ou N unidades o parâmetro X

```
var
  IntVar: Integer;
  LongintVar: Longint;
begin
  Inc(IntVar);           { IntVar := IntVar + 1 }
  Inc(LongintVar, 5);   { LongintVar := LongintVar + 5 }
end;
```

```
function IncMonth(const Date: TDateTime; NumberOfMonths: Integer): TDateTime;
```

Retorna Date acrescido ou decrescido de NumberOfMonths meses.

```
function InputBox(const ACaption, APrompt, ADefault: string): string;
```

Exibe uma Caixa de Entrada onde o usuário pode digitar uma string.

ACaption representa o título do Input Box e APrompt é o título do edit e ADefault representa o valor inicial do Edit.

```
uses Dialogs;
procedure TForm1.Button1Click(Sender: TObject);
var
  InputString: string;
begin
  InputString:= InputBox('Input Box', 'Prompt', 'Default string');
end;
```

```
function InputQuery(const ACaption, APrompt: string; var Value: string): Boolean;
```

Semelhante ao InputBox, sendo que retorna True se o usuário fechou a O InputBox com OK e False se fechou com Cancel ou ESC. E Value armazena a string digitada.

```
procedure TForm1.Button1Click(Sender: TObject);
var
  NewString: string;
  ClickedOK: Boolean;
begin
  NewString := 'Default String';
  Label1.Caption := NewString;
  ClickedOK := InputQuery('Input Box', 'Prompt', NewString);
  if ClickedOK then { NewString contains new input string }
    Label1.Caption := 'The new string is ' + NewString + ' ';
end;
```

```
procedure Insert(Source: string; var S: string; Index: Integer);
```

Inserir uma string em outra a partir da posição Index

```
var
  S: string;
begin
  S := 'Honest Lincoln';
  Insert('Abe ', S, 8); { 'Honest Abe Lincoln' }
end;
```

```
function IntToHex(Value: Integer; Digits: Integer): string;
```

Converte o inteiro Value num Hexadecimal (Base 16). Digits indica o número mínimo de dígitos Hexa a serem retornados

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  Edit2.Text := IntToHex(StrToInt(Edit1.Text), 6);
end;
```

```
function IntToStr(Value: Integer): string;
```

Transforma um Inteiro numa String

```
procedure TForm1.Button1Click(Sender: TObject);
var
  Value: Integer;
begin
  Value := 1234;
  Edit1.Text := IntToStr(Value);
end;
```

```
function IsValidIdent(const Ident: string): Boolean;
```

Indica se um identificador é válido para o Pascal

```
function Length(S: string): Integer;
```

retorna o número de caracteres usados na string S

```
var
  S: string;
begin
  S := 'The Black Knight';
  Canvas.TextOut(10, 10, 'String Length = ' + IntToStr(Length(S)));
end;
```

```
function MaxIntValue(const Data: array of Integer): Integer;
```

Retorna o maior inteiro de um vetor

```
function MaxValue(const Data: array of Double): Double;
```

Retorna o maior valor de um vetor

```
function Mean(const Data: array of Double): Extended;
```

Retorna a média aritmética de um vetor

```
function MessageDlg(const Msg: string; AType: TMsgDlgType;
  AButtons: TMsgDlgButtons; HelpCtx: Longint): Word;
```

Exibe uma Caixa de Mensagem e obtém uma resposta do usuário.
Onde

Msg : Mensagem

AType : Tipo da caixa de mensagem

Value Meaning

mtWarning	A message box containing a yellow exclamation point symbol.
mtError	A message box containing a red stop sign.
mtInformation	A message box containing a blue "i".
mtConfirmation	A message box containing a green question mark.
mtCustom	A message box with no bitmap. The caption of the message box is the name of the application's executable file.

AButtons : Quais botões aparecerão na caixa de mensagem

Value	Meaning
mbYes	A button with the text 'Yes' on its face
mbNo	A button with the text 'No' on its face
mbOK	A button with the text 'OK' on its face
mbCancel	A button with the text 'Cancel' on its face
mbHelp	A button with the text 'Help' on its face
mbAbort	A button with the text 'Abort' on its face
mbRetry	A button with the text 'Retry' on its face
mbIgnore	A button with the text 'Ignore' on its face
mbAll	A button with the text 'All' on its face
mbYesNoCancel	Puts Yes, No, and Cancel buttons in the message box
mbOkCancel	Puts t OK and Cancel buttons in the message box
mbAbortRetryIgnore	Puts Abort, Retry, and Ignore buttons in the message box

MessageDlg returns the value of the button the user selected. These are the possible return values:

Return values

mrNone	mrAbort	mrYes
mrOk	mrRetry	mrNo
mrCancel	mrIgnore	mrAll

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  if MessageDlg('Welcome to my Object Pascal application. Exit now?',
    mtConfirmation, [mbYes, mbNo], 0) = mrYes then
  begin
    MessageDlg('Exiting the Object Pascal application.', mtInformation,
      [mbOk], 0);
    Close;
  end;
end;
```

```
function MessageDlgPos(const Msg: string; AType: TMsgDlgType;
  AButtons: TMsgDlgButtons; HelpCtx: Longint; X, Y: Integer): Word;
```

Semelhante a MessageDlg exceto por permitir indicar a posição na qual a janela será exibida

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  MessageDlgPos('Are you there?', mtConfirmation, mbYesNoCancel, 0, 200, 200);
end;
```

```
function MinIntValue(const Data: array of Integer): Integer;
Retorna o menor inteiro do vetor
```

```
function MinValue(const Data: array of Double): Double;
Retorna o menor valor de um vetor
```

```
procedure MkDir(S: string);
```

Cria um novo diretório

```
uses Dialogs;
begin
  {$I-}
  { Get directory name from TEdit control }
  MkDir(Edit1.Text);
  if IOResult <> 0 then
    MessageDlg('Cannot create directory', mtWarning, [mbOk], 0)
  else
    MessageDlg('New directory created', mtInformation, [mbOk], 0);
end;
```

```
function Now: TDateTime;
```

Retorna a data e a hora corrente

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  Label1.Caption := 'The date and time is ' + Str(Now);
end;
```

```
function Ord(X): Longint;
```

Retorna a ordem de um valor ordinal

```
uses Dialogs;
type
  Colors = (RED, BLUE, GREEN);
var
  S: string;
begin
  S := 'BLUE has an ordinal value of ' + IntToStr(Ord(BLUE)) + #13#10;
  S := 'The ASCII code for "c" is ' + IntToStr(Ord('c')) + ' decimal';
  MessageDlg(S, mtInformation, [mbOk], 0);
end;
```

```
function Pi: Extended;
```

Retorna o valor de PI

3.1415926535897932385.

```
function Pos(Substr: string; S: string): Integer;
```

Procura por uma sub-string numa string e retorna a posição da primeira ocorrência ou zero se não encontrou

```
var S: string;
begin
  S := ' 123.5';
  { Convert spaces to zeroes }
  while Pos(' ', S) > 0 do
    S[Pos(' ', S)] := '0';
end;
```

```
function Power(Base, Exponent: Extended): Extended;
```

Potência

```
function Pred(X);
```

Retorna o predecessor de um ordinal

```
uses Dialogs;
type
  Colors = (RED,BLUE,GREEN);
var
  S: string;
begin
  S := 'The predecessor of 5 is ' + IntToStr(Pred(5)) + #13#10;
  S := S + 'The successor of 10 is ' + IntToStr(Succ(10)) + #13#10;
  if Succ(RED) = BLUE then
    S := S + 'In the type Colors, RED is the predecessor of BLUE.';
  MessageDlg(S, mtInformation, [mbOk], 0);
end;
```

```
function Random [ ( Range: Integer) ];
```

Retorna um valor Randômico

$0 \leq X < \text{Range}$.

```
var
  I: Integer;
begin
  Randomize;
  for I := 1 to 50 do begin
    { Write to window at random locations }
    Canvas.TextOut(Random(Width), Random(Height), 'Boo!');
  end;
end;
```

```
procedure Randomize;
```

Inicializa o modo Randomico

```
var
  I: Integer;
begin
  Randomize;
  for I := 1 to 50 do begin
    { Write to window at random locations }
    Canvas.TextOut(Random(Width), Random(Height), 'Boo!');
  end;
end;
```

```
function RemoveDir(const Dir: string): Boolean;
```

Remove um diretório retornando True caso tenha conseguido e False caso contrário

```
procedure Rename(var F; Newname);
```

F is a variable of any file type. Newname is a string-type expression or an expression of type PChar

```
uses Dialogs;
var
  f : file;
begin
  OpenDialog1.Title := 'Choose a file... ';
  if OpenDialog1.Execute then begin
    SaveDialog1.Title := 'Rename to...';
    if SaveDialog1.Execute then begin
      AssignFile(f, OpenDialog1.FileName);
      Canvas.TextOut(5, 10, 'Renaming ' + OpenDialog1.FileName + ' to ' +
        SaveDialog1.FileName);
      Rename(f, SaveDialog1.FileName);
    end;
  end;
end;
```

```
function RenameFile(const OldName, NewName: string): Boolean;
```

Renomeia arquivos e retorna o sucesso ou insucesso

The following code renames a file:

```
if not RenameFile('OLDNAME.TXT', 'NEWNAME.TXT') then
  ErrorMsg('Error renaming file!');
```

```
procedure RmDir(S: string);
```

Remove um diretório

```
uses Dialogs;
begin
  {$I-}
  { Get directory name from TEdit control }
  RmDir(Edit1.Text);
  if IOResult <> 0 then
    MessageDlg('Cannot remove directory', mtWarning, [mbOk], 0)
  else
    MessageDlg('Directory removed', mtInformation, [mbOk], 0);
end;
```

```
function Round(X: Extended): Longint;
```

Arredonda um número real

```
function SelectDirectory(var Directory: string; Options: TSelectDirOpts; HelpCtx: Longint): Boolean;
```

Exibe um Caixa de Dialogo para seleção de Diretório. O Diretório passado para a função aparece como diretório corrente e o diretório escolhido é retonado no mesmo Diretório (Directory). O valor do diretório corrente não é alterado

These are the possible values that can be added to the Options set:

Value	Meaning
sdAllowCreate	An edit box appears to allow the user to type in the name of a directory that does not exist. This option does not create a directory, but the application can access the Directory parameter to create the directory selected if desired.
sdPerformCreate	Used only when Options contains sdAllowCreate. If the user enters a directory name that does not exist, SelectDirectory creates it.
sdPrompt	Used when Options contains sdAllowCreate. Displays a message box that informs the user when the entered directory does not exist and asks if the directory should be created. If the user chooses OK, the directory is created if Options contains sdPerformCreate. If Options does not contain sdPerformCreate, the directory is not created: the application should create it when SelectDirectory returns.

The function returns True if the user selected a directory and chose OK, and False if the user chose Cancel or closed the dialog box without selecting a directory.

This example uses a button on a form. When the user clicks the button, a Select Directory dialog box appears. The current directory displayed in the dialog box is C:\MYDIR. The user can select a directory from the directory list, or enter a new directory in the edit box. If the user enters a new directory, a message box asks the user if the directory should be created. If the user chooses Yes, the directory is created. If the user chooses No, the message box goes away without creatubg the directory. The name of the directory the user selects appears as the caption of the label:

```
uses FileCtrl;

procedure TForm1.Button1Click(Sender: TObject);
var
  Dir: string;
begin
  Dir := 'C:\MYDIR';
  if SelectDirectory(Dir, [sdAllowCreate, sdPerformCreate, sdPrompt]) then
    Label1.Caption := Dir;
end;
```

```
function SetCurrentDir(const Dir: string): Boolean;
```

Torna o Dir o diretório corrente e retorna o sucesso da operação

```
procedure SetLength(var S: string; NewLength: Integer);
```

Coloca um novo tamanho para uma string
`S[0] := NewLength.`

```
procedure ShowMessage(const Msg: string);
```

Exibe uma mensagem ao usuário

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  ShowMessage('Push this button');
end;
```

```
function Sqr(X: Extended): Extended;
```

Retorna o quadrado de X

```
function Sqrt(X: Extended): Extended;
```

Retorna a raiz quadrada de X

```
function StrComp(Str1, Str2 : PChar): Integer;
```

Compara duas strings em case sensitivity

Return value Condition

<0 if Str1 < Str2

=0 if Str1 = Str2

>0 if Str1 > Str2

```
function StringOfChar(Ch: Char; Count: Integer): string;
```

Retorna uma string contendo Count caracteres Ch

```
S := StringOfChar('A', 10);
{sets S to the string 'AAAAAAAAAA'}
```

```
function StrToDate(const S: string): TDateTime;
```

Converte uma string em data

```
procedure TForm1.Button1Click(Sender: TObject);
var
  ADate: TDateTime;
begin
  ADate := StrToDate(Edit1.Text);
  Label1.Caption := DateToStr(ADate);
end;
```

```
function StrToDateTime(const S: string): TDateTime;
```

Converte uma string para o formato DateTime

```
procedure TForm1.Button1Click(Sender: TObject);
var
  ADateAndTime: TDateTime;
begin
  ADateAndTime := StrToDateTime(Edit1.Text);
  Label1.Caption := DateTimeToStr(ADateAndTime);
end;
```

```
function StrToFloat(const S: string): Extended;
```

Converte uma string num Float

```
function StrToInt(const S: string): Integer;
```

Converte uma string num inteiro

```
procedure TForm1.Button1Click(Sender: TObject);
var
  S: string;
  I: Integer;
begin
  S := '22467';
  I := StrToInt(S);
  Inc(I);
  Edit1.Text := IntToStr(I);
end;
```

```
function StrToTime(const S: string): TDateTime;
```

Converte uma string em hora

```
procedure TForm1.Button1Click(Sender: TObject);
var
  ATime: TDateTime;
begin
  ATime := StrToTime(Edit1.Text);
  Label1.Caption := TimeToStr(ATime);
end;
```

```
function Succ(X);
```

Retorna o sucessor de um ordinal

```
uses Dialogs;
type
  Colors = (RED, BLUE, GREEN);
var
  S: string;
begin
  S := 'The predecessor of 5 is ' + IntToStr(Pred(5)) + #13#10;
  S := S + 'The successor of 10 is ' + IntToStr(Succ(10)) + #13#10;
  if Succ(RED) = BLUE then
    S := S + 'In the type Colors, RED is the predecessor of BLUE.';
  MessageDlg(S, mtInformation, [mbOk], 0);
end;
```

```
function Sum(const Data: array of Double): Extended register;
```

Calcula a soma de dos elementos de um vetor

```
function SumInt(const Data: array of Integer): Integer register;
```

Calcula a soma dos elementos de um vetor de inteiros

```
function Time: TDateTime;
```

Retorna a hora corrente

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  Label1.Caption := 'The time is ' + TimeToStr(Time);
end;
```

```
function TimeToStr(Time: TDateTime): string;
```

Converte hora para string

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  Label1.Caption := TimeToStr(Time);
end;
```

```
function Trim(const S: string): string;
```

Retira os espaços em brancos a esquerda e a direita da string S

```
function TrimLeft(const S: string): string;  
Retira os espaços em brancos a esquerda da string S
```

```
function TrimRight(const S: string): string;  
  
Retira os espaços em brancos a direita da string S
```

```
function Trunc(X: Extended): Longint;  
Retorna a parte inteira de um número
```

```
function UpCase(Ch: Char): Char;
```

Converte o caracter Ch em maiúscula

```
uses Dialogs;  
var  
  s : string;  
  i : Integer;  
begin  
  { Get string from TEdit control }  
  s := Edit1.Text;  
  for i := 1 to Length(s) do  
    s[i] := UpCase(s[i]);  
  MessageDlg('Here it is in all uppercase: ' + s, mtInformation, [mbOk], 0);  
end;
```

```
function UpperCase(const S: string): string;
```

Converte a string S em maiúsculas

```
procedure TForm1.Button1Click(Sender: TObject);  
var  
  I: Integer;  
begin  
  for I := 0 to ListBox1.Items.Count -1 do  
    ListBox1.Items[I] := UpperCase(ListBox1.Items[I]);  
end;
```

Componentes - Propriedades, Eventos e Métodos

Propriedades Comuns

Nome	Descrição
Align	Determina como o componente será alinhado no seu container [alNone, alTop, alBottom, alLeft, alRight, alClient]
Caption	Legenda do componente (& indica a tecla de atalho)
Cursor	Desenho que representa o cursor da mouse [crDefault, crNone, crArrow, crCross, crIBeam, crSize, crSizeNESW, crSizeNS, crSizeNWSE, crSizeWE, crUpArrow, crHourGlass, crDrag, crNoDrop, crHSplit, crVSplit, crMultiDrag, crSQLWait, crNo, crAppStart, crHelp, crHandPoint]
Name	Nome da instância do componente
Left	Distância em Pixel da borda esquerda do componente até a borda esquerda do FORM
Top	Distância em pixel da borda superior do componente até a borda superior do FORM
Height	Altura em pixel do componente
Width	Largura em pixel do componente
ComponentCount	O número de componentes possuídos por um componente container
Components	Uma matriz de componentes possuídos por um componente container
Color	Indica a cor de fundo do componente
Font	Fonte utilizada no componente
Ctl3D	Define a aparência 3D ou 2D de um componente
Enabled	Define se o componente esta ou não ativo
Visible	Define se o componente esta ou não visível
Hint	String utilizada na exibição de dicas instantâneas
ShowHint	Define se o hint será mostrado automaticamente
PopupMenu	Menu que será acionado com o botão direito do mouse
TabOrder	A ordem de tabulação do componente
TabStop	Indica se o componente será ponto de parada para a tecla TAB
HelpContext	Número utilizado para chamar o help on-line sensível ao contexto

Eventos Comuns

Nome	Descrição
OnChange	O conteúdo do componente é alterado
OnClick	O componente é acionado (Mouse ou Enter)
OnDblClick	Duplo-clique sobre o componente
OnEnter	O componente recebe o foco
OnExit	O componente perde o foco
OnKeyDown	Tecla(s) são pressionada(s) inclusive teclas de controle Parametros : Key = Código da tecla Shift = conjunto que indica a(s) tecla(s) de controle pressionadas (ssShift, ssAlt, ssCtrl, ssLeft, ssRight, ssMiddle, ssDouble)
OnKeyPress	Uma tecla é pressionada, onde Key contém o caracter pressionado
OnKeyUp	Uma tecla é solta

Métodos Comuns

Nome	Descrição
Create	Cria uma nova instância
Destroy	Destrói a instância
Show	Torna o componente visível
Hide	Torna o componente invisível
SetFocus	Coloca o foco no componente
Focused	Determina se o componente tem o foco
BringToFront	Coloca o componente na frente dos outros
SendToBack	Coloca o componente atrás dos outros
ScaleBy	Gradua o componente em determina escala. Ex: Button1.ScaleBy(90,100) altera o tamanho do botão para 90% do tamanho original
SetBounds	Muda a posição e o tamanho do componente (ALeft,ATop,AWidth,AHeigh)

Componentes

Form - Elemento básico no desenvolvimento Delphi formando o alicerce sobre, o qual um aplicativo é construído.

Propriedades	Descrição
Active	Indica quando o formulário esta ativo.
ActiveControl	Determina o controle que receberá o foco quando o formulário é ativado
AutoScroll	Adiciona barras de rolagens automaticamente quando um formulário é redimensionado de forma a cobrir componentes
HorzScrollBar	Adiciona Barra de rolagem Horizontais quando necessário
VertScrollBar	Adiciona Barra de rolagem Verticais quando necessário
BorderIcons	Define quais ícones de controle serão visíveis [biSystemMenu, biMinimize, biMaximize, biHelp]
BorderStyle	Estilo da borda da janela [bsDialog, bsSingle, bsNone, bsSizeable, bsToolWindow, bsSizeToolWin]
FormStyle	Tipo da janela [fsNormalfs, MDIChild, fsMDIForm, fsStayOnTop]
Icon	Ícone da janela
Menu	Indica qual o componente menu do formulário será apresentado
Position	Permite controlar a posição e tamanho dos formulários na execução [poDesigned, poDefault, poDefaultPosOnly, poDefaultSizeOnly, poScreenCenter]
WindowState	Estado da janela (normal, minimizado ou maximizado)

Eventos	Descrição
OnCreate	O formulário é criado
OnShow	Antes de mostrar a janela
OnCloseQuery	É chamada para validar se a janela pode ser fechada
OnClose	Ocorre quando a janela é fechada
OnActivate	Ocorre quando a janela torna-se ativa
OnDeactivate	Ocorre quando a janela perde o foco
OnResize	Ocorre quando a janela muda de tamanho

Métodos	Descrição
Show	Mostra uma janela não-modal
ShowModal	Ativa uma janela modal
Close	Fecha a janela
Refresh	Redesenha a Janela

Button - Componente utilizado para representar ações

Propriedades	Descrição
Cancel	Dispara o evento OnClick do botão quando a tecla ESC é pressionada
Default	Dispara o evento OnClick do botão quando a tecla ENTER é pressionada
ModalResult	Associa o botão a opção de fechamento de um Form modal

BitBtn - Botão contendo uma Legenda e um BitMap, possui as propriedades e métodos do SpeedButton

Propriedades	Descrição
Kind	Seleciona um BitMap padrão para o botão
Style	Indica a aparência do botão (win3.11, win95, winxx)

SpeedButton - Botão contendo um BitMap, normalmente utilizado na construção de barra de ferramentas

Propriedades	Descrição
Glyph	BitMap exibido pelo botão
LayOut	Posição do BitMap no Botão
Margin	Indica o espaço entre a borda do botão e o BitMap
Spacing	Indica o espaço entre o BitMap e o Texto do botão
Down	Estado do botão (Pressionado ou não)
GroupIndex	Indica quais botões pertencerão ao mesmo grupo
AllowAllUp	Permite que todos os botões de um grupo possam estar não pressionados

Métodos	Descrição
Click	Ativa o evento OnClick do botão

Label - Utilizado para exibir rótulos

Propriedades	Descrição
Alignment	Alinhamento do texto no componente
AutoSize	Define se o tamanho do componente será automaticamente ajustado ao tamanho da legenda
WordWrap	Retorno automático
Transparent	Define se o componente será transparente
FocusControl	Nome do componente que receberá o foco
ShowAccelChar	Indica se & será fará ou não parte da legenda

Edit - Utilizado para entrada de dados texto em uma única linha.

Propriedades	Descrição
Text	Armazena a entrada de dados
AutoSelect	Indica se o texto será ou não selecionado quando o componente receber o foco
MaxLength	Número máximo de caracteres permitidos
CharCase	Define se as letras aparecerão em maiúsculo, minúsculo ou normal
PasswordChar	Caracter utilizado para esconder os dados digitados (Senhas)
ReadOnly	Define se será permitido alterar o texto
SelLength	Comprimento da seleção
SelStart	Início da seleção
SelText	Texto selecionado

MaskEdit - Permite entrada de dados texto em uma linha, utilizando uma máscara de edição. Possui todas as propriedades do componente Edit

Propriedades	Descrição
EditMask	Máscara de edição

A propriedade EditMask consiste em uma máscara de edição permitindo definir quais os possíveis caracteres e a formatação para a propriedade Text.

Essa máscara consiste de três partes separadas por ;. A primeira parte é a máscara propriamente dita, a segunda parte indica se os caracteres literais serão armazenados na propriedade text (0 - não armazena; 1 - armazena). A terceira parte indica qual o caracter utilizado representar os espaços a serem digitados no texto

Estes são os caracteres especiais que podem compor a máscara de edição

Caracter	Descrição
!	Espaços em branco não aparecerão
>	Todos os caracteres seguintes serão maiúsculos até que apareça o caracter <
<	Todos os caracteres seguintes serão minúsculos até que apareça o caracter >
\	Indica um caracter literal
l	Somente caracter alfabético
L	Obrigatoriamente um caracter alfabético (A-Z, a-z)
a	Somente caracter alfanumérico
A	Obrigatoriamente caractere alfanumérico (A-Z, a-z, 0-9)
9	Somente caracter numérico
0	Obrigatoriamente caracter numérico
c	permite um caracter
C	Obrigatoriamente um caracter
#	Permite um caracter numérico ou sinal de mais ou de menos, mas não os requer.
:	Separador de horas, minutos e segundos
/	Separador de dias, meses e anos

Memo - Permite entrada de dados texto em múltiplas linhas.

Propriedades	Descrição
Alignment	Indica como será o alinhamento do texto
Lines	Armazena as linhas de texto
WantReturns	Define se a tecla ENTER será tratada pelo Formulário ou pelo Memo
WantTab	Define se a tecla TAB será tratada pelo Formulário ou pelo Memo
WordWrap	Indica se a linha digitada será quebrada, automaticamente, de acordo com o tamanho do componente
ScrollBar	Indica se Memo terá barras de rolagem

Métodos	Descrição
Clear	Limpa o memo
ClearSelection	Limpa o texto selecionado no memo

Obs: Muitos componentes possuem propriedades do Tipo TStrings, essa classe permite armazenar e manipular uma lista de Strings. Toda propriedade do tipo TStrings permite acesso indexado aos itens da listas (Propriedade[indíce])

TString

Propriedades	Descrição
Count	Número de linhas

Métodos	Descrição
Add	Adiciona uma nova linha no final da lista
Insert	Insere uma nova linha numa posição especificada
Delete	Apaga uma linha
Clear	Apaga toda a lista
IndexOf	Retorna o índice do item e -1 caso não encontre
LoadFromFile	Carrega de um arquivo texto
SaveToFile	Salva para um arquivo texto

CheckBox - Utilizado para obter informações “lógicas” independentes

Propriedades	Descrição
AllowGrayed	Determina se o checkbox terá duas ou três possibilidades
Checked	Determina se o checkbox está selecionado
State	Estado atual do checkbox (cbUnchecked, cbChecked, cbGrayed)

RadioButton - Utilizado para obter informações lógicas mutuamente exclusivas

Propriedades	Descrição
Checked	Determina se o RadioButton esta selecionado

GroupBox - Utilizados para agrupar componentes, sobre determinado título.

RadioGroup - Componente que agrupa e controla RadioButtons

Propriedades	Descrição
Columns	Número de colunas de botões de rádio
Items	Valores dos botões de rádio. Items[n] acessa o n-ésimo componente
ItemIndex	Item selecionado. (-1 nenhum, começa em 0)

Panel - Utilizados para agrupar componentes e criar barras de ferramentas.

Propriedades	Descrição
BevelInner	Estilo interno da superfície do Panel
BevelOuter	Estilo externo da superfície do Panel
BevelWidth	Distância entre as superfícies externas e internas
BorderStyle	Estilo da Borda
BorderWidth	Largura da borda

ScrollBar - Semelhante ao componente GrupoBox, sendo que, possui barras de rolagem

Propriedades	Descrição
HorzScrollBar	Barra Horizontal (Increment, Tracking e Visible)
VertScrollBar	Barra Vertical (Increment, Tracking e Visible)

Bevel - Define linhas, retângulos e molduras nas janelas.

Propriedades	Descrição
Shape	Tipo de figura a ser desenhada
Style	Define alto e baixo relevo

ListBox - Utilizado para escolher em uma lista grande de opções.

Propriedades	Descrição
Columns	Número de colunas da lista
MultiSelect	Define se será permitida a seleção de múltiplos itens
ExtendedSelect	Define se a seleção poderá ser estendida pelo uso das teclas Shift e Ctrl
IntegralHeight	Define se os itens poderão aparecer parcialmente ou somente por completo
Items	Valores dos itens da lista
ItemIndex	Item selecionado. (-1 não existe item selecionado e o 1º é 0)
Selected	De acordo com o índice indica se um item em particular esta selecionado.
SelCount	Indica quantos itens estão selecionado
Sorted	Define se os itens aparecerão ordenados

ComboBox - Reúne as características de Edit e ListBox.

Propriedades	Descrição
Items	Valores a serem exibidos na lista
DropDownCount	Número de itens visíveis da lista
Text	Conteúdo texto digitado na ComboBox
Style	csDropDown - permite edição e exibe os itens mediante solicitação csDropDownList - não permite edição e mostra itens no edit ao pressionar a 1º letra do item. CsSimple - permite edição e exibe a lista
Sorted	Define se os itens aparecerão ordenados

CheckListBox – Possui toda a funcionalidade do ListBox exibindo uma CheckBox p/ cada item

Propriedades	Descrição
Checked[n]	Retorna true se o item n estiver selecionado
State[n]	Retorna o estado do item n : [cvUnchecked, cbChecked, cbGrayed]

Eventos	Descrição
OnClickChecked	Quando um item é marcado ou desmarcado

ScrollBar - Permite a escolha de um valor numérico dentro de uma faixa.

Propriedades	Descrição
Min	Valor mínimo possível
Max	Valor máximo possível
Position	Posição Atual
LargeChange	Incremento da posição quando o click é na barra
SmallChange	Incremento da posição quando o click é na seta
Kind	Se a barra é vertical ou horizontal

Eventos	Descrição
OnEditMask	Permite atribuir uma máscara a célula. A linha e a coluna da célula editada são recebidas pelo evento e mascarar é representada pelo parâmetro Value

StringGrid e DrawGrid - Utilizado para entrada ou exibição de dados no formato de tabela.

Propriedades	Descrição
ColCount	Número de colunas
RowCount	Número de linhas
Col	Coluna corrente
Row	Linha Corrente
DefaultColWidth	Largura das colunas
DefaultRowHeight	Altura das linhas
DefaultDrawing	Define se o desenho das células será automático ou manual
FixedCol	Quantidade de Linhas Fixas
FixedRow	Quantidade de colunas fixas
GridLineWidth	Espessura das Linhas divisórias
Options	Define características do Grid. goEditing e goTabs
Cells[col,lin]	Permite acessar as células da Grid, por coluna e linha, sendo que a primeira linha e coluna têm valores 0.

Image - Exibe uma imagem.

Propriedades	Descrição
Picture	Arquivo Bitmap, Ícone ou Windows Metafile exibido
Center	Centraliza a figura no componente
Stretch	Define se o tamanho da figura deve ser ajustada ao do componente
Transparente	Torna o fundo visível ou opaco

Shape - Exibe uma figura geométrica

Propriedades	Descrição
Brush	Cor e preenchimento da figura
Pen	Cor e preenchimento da borda
Shape	Figura geométrica

Splitter – Permite o redimensionamento de componente em tempo de execução

Propriedades	Descrição
Beveled	Exibe o componente com aparência 3D
MinSize	Tamanho mínimo para os componentes

TabControl - Contem guias, as quais podem ser utilizadas para alterar outros componentes

Propriedades	Descrição
MultiLine	Permite múltiplas linhas para as guias
Tabs	Guias
TabPosition	Posição das “orelhas” em cima ou embaixo
HotTrack	Destaca a “orelha” quando o curso do mouse esta em cima da mesma
TabIndex	Guia ativa
ScrollOpposite	Transfere as “orelhas” das linhas anteriores à selecionada para a outra extremidade do Componente

Eventos	Descrição
OnChange	Quando uma guia é selecionada

PageControl - Contem páginas, as quais, podem possuir diversos componentes. Para inserir uma nova página dê um clique com o botão direito do mouse e escolha NewPage. Cada página criada é um objeto do tipo TTabSheet e possui as seguintes propriedades : Caption, PageIndex e TabVisible, onde PageIndex indica a ordem de apresentação de cada página e TabVisible permite exibir ou ocultar as páginas.

Propriedades	Descrição
MultiLine	Permite múltiplas linhas para as páginas
ActivePage	Página ativa
PageIndex	Índice da página ativa

Eventos	Descrição
OnChange	Quando uma página é selecionada

ImageList – Representa uma coleção de imagens do mesmo tamanho, sua principal função é armazenar imagens para outros componentes, onde cada imagem pode ser referenciada através de seu índice no vetor de imagens.

Propriedades	Descrição
Count	Número de imagens na lista

RichEdit – Componente para editar texto no formato Rich Text, esse componente permite diferentes formatações para o seu texto, diferenciando-o do componente Memo. Possui todas as propriedades do memo.

Propriedades	Descrição
HideScrollBar	Somente exibe as barras de rolagem quando necessário
HideSelection	Indica se o texto continuara com a indicação de selecionado mesmo que o componente perca o foco
SelAttributes	Formatação do Texto selecionado.
Color	- Cor dos caracteres
Name	- Nome da fonte usada
Size	- Tamanho da Fonte
Style	- Estilo da Fonte [fsBold,fsItalic, fsUnderline,fsStrikeout]

TrackBar - Componente utilizado para seleção de valores inteiros dentro de uma faixa

Propriedades	Descrição
Orientation	Orientação vertical ou horizontal
Min	valor mínimo
Max	valor máximo
Position	Posição corrente
TickStyle	Estilo de exibição das marcações
TickMarks	Aparência do indicador
PageSize	Determina o incremento que deve ser dado quando as teclas PgUp e PgDn forem pressionadas
LineSize	Determina o incremento que deve ser dado quando as setas forem pressionadas
SelStart	Posição de início do preenchimento
SelEnd	Posição de término do preenchimento

ProgressBar - Componente para indicar visualmente o andamento da execução de tarefas

Propriedades	Descrição
Min	valor mínimo
Max	valor máximo
Step	incremento que deve ser dado a propriedade position em cada mudança
Position	Posição corrente

Métodos	Descrição
StepIt	Incrementa Position de Step unidades
StepBy	Incrementa Position de n unidades

UpDown - Utilizado normalmente associado a outro componente para realizar incremento em dados numéricos

Propriedades	Descrição
Associate	Indica o componente associado
AlignButton	Indica o alinhamento do UpDown em relação ao componente associado (udLeft, udRight)
Min	Valor mínimo
Max	Valor máximo
Orientation	Orientação do componente (UdHorizontal, udVertical)
Wrap	Salto do valor mínimo para o máximo e vice-versa
Increment	Incremento dado ao componente associado
ArrowKeys	Indica que o componente recebe os incrementos das teclas de SETAS
Positon	Valor corrente
Thousands	Indica se irá aparecer o separador de milhar
Wrap	Salto altomático de Max para Min e vice-versa

HotKey - Obtém em tempo de execução teclas de atalho e podem ser utilizadas para definir as teclas de atalho para outros componentes quem tenham a propriedade ShortCut

Propriedades	Descrição
HotKey	Combinação de teclas para a HotKey
InvalidKeys	Especifica as teclas inválidas para modificadores [hcNone , hcShift, hcCtrl, hcAlt, hcShiftCtrl, hcShiftAlt, hcCtrlAlt, hcShiftCtrlAlt]
Modifiers	Teclas modificadoras [hkShift, hkCtrl, hkAlt, hkExt]

Animate – Componente capaz de exibir um AVI, o qual representa um formato de arquivo multimídia com imagens e sons, mas este componente apenas exibe as imagens.

Propriedades	Descrição
Active	Indica se a animação esta sendo exibida ou não
AutoSize	Ajusta automaticamente o tamanho do componente ao tamanho da imagem
Center	Centraliza a animação
FileName	Nome do arquivo AVI
FrameCount	Número de Frames da animação
Repetitions	Número de repetições quando a animação for executada. O valor zero indica repetições indefinidas

DateTimePicket – Componente que permite a seleção visual de uma datas ou horas

Propriedades	Descrição
Time	Hora selecionada
Date	Data Selecionada
DateMode	A forma como a data poderá ser selecionada [dmComboBox, dmUpDown]
DateFormat	Formato da Data [dfShort, dfLong]
Kind	Seleciona o componente para Data ou Hora [dtkDate, dtkTime]
ShowCheckbox	Exibe um CheckBox
Check	Indica se o CheckBox esta selecionado

TreeView - Permite exibição de dados em forma hierárquica. Este componente possui métodos e eventos que permitem controle e modificação da sua estrutura em tempo de execução.

Propriedades	Descrição
Items	Define os itens da hierarquia.
Ident	Recoo dos sus-itens
ShowLines	Determina se haverá uma linha ligando os sub-itens
ShowRoot	Determina se haverá uma linha ligando os itens raízes
ShowButtons	Indica se o botão a esquerda do item será visível
HideSelect	Indica se quando o componente perder o foco a seleção continuará ativa
SortedType	nsNone não é ordenado nsData os iten são ordenados quando os dados são alterados nsText os itens são ordenados quando o Caption é alterado. nsBoth a ordenação e feita em ambos os casos
Selected	Item selecionado. Podemos acessar o conteúdo selecionado através de Select.Text;

ListView - Componente que permite exibir de várias maneiras uma lista de itens.

Propriedades	Descrição
ViewStyle	Determina se os itens devem ser apresentados em colunas com cabeçalhos e sub-itens, verticalmente ou horizontalmente, com ícones grandes ou pequenos
LargeImages	Lista de Imagens (TImagesList) contendo a bitmap's a serem exibidos e somente é usada quando a propriedade ViewStyle é vsIcon
SmallImages	Lista de Imagens (TImagesList) contendo a bitmap's a serem exibidos e somente é usada quando a propriedade ViewStyle é vsSmallIcon
Items	Items a serem exibidos
Columns	Cabeçalhos das colunas da Lista
ShowColumnHeaders	Exibe os cabeçalhos das colunas
ColumnClick	Indica se os cabeçalhos das colunas terão a aparência de botões
IconOptions	Opções de exibição dos ícones quando ViewStyle for vsIcon ou vsSmallIcons Arrangement alinhado no topo ou esquerda do ListView AutoArrange os ícones são alinhados automaticamente WrapText a propriedade caption será quebrada
SortedType	nsNone não é ordenado nsData os iten são ordenados dados são alterados nsText os itens são ordenados quando o Caption é alterada. nsBoth a ordenação e feita em ambos os casos
Selected	Item selecionado. Podemos acessar o conteúdo selecionado através de Select.Caption;

HeaderControl - Utilizado para exibir barra de títulos ajustáveis e cabeçalhos. Podemos abrir o editor de seções através do botão direito do mouse.

Propriedades	Descrição
Sections[n]	Cabeçalhos do componente, possuindo as seguintes propriedades : Text, Width, Min, Max, Style, Alignment, AllowClick e Index. Onde uma seção em particular pode ser acessada através do seu índice
HotTrack	Diferencia o cabeçalho quando o mouse esta sobre este
Eventos	Descrição
OnSectionClick	Clique sobre uma seção
OnSectionResize	A seção tem seu tamanho alterado

StatusBar - Utilizado para criar barra de Status para exibir informações sobre a aplicação. Podemos abrir o editor de paines através do botão direito do mose

Propriedades	Descrição
SimplePanel	Indica se o StatusBar possuirá um ou vários panels
SimpleText	Texto exibido caso SimplePanel igual a TRUE
SizeGrip	Indicador de tamanho padrão do Windows
Panels	Painéis do StatusBar, cada panel permite a exibição de informação. Os painéis são acessados como elementos de um vetor (Panels[n]) e possuem as seguintes propriedades : Text, Width, Style, Bevel, Alignment

ToolBar – Componente utilizado para criar barras de ferramentas. Para adicionar botões e sepradores usamos o botão direito do mouse. Este é um componente container e podemos colocar outros componentes no seu interior, os quais serão automaticamente alinhados. Cada botão adicionado possui as seguinte propriedades : ImageIndex, Style[tbsButton, tbsCheck, tbsDivide, tbsDropDown, tbsSeparator] e Indeterminate

Propriedades	Descrição
Flat	Os botões ficam terão a aparência do Office 97
Images	Componente ImageList que conterà as figuras dos botões
Buttons[n]	Acessa os botões através do índice n

CoolBar – Componente utilizado para criar barras de ferramentas que podem ser arrastadas e reposicionadas. Podemos editar as faixas através do botão direito do mouse.

Propriedades	Descrição
Bands	Armazenas as Faixas (Cool bands)
BandBorderStyle	Indica o estilo da borda entre os CollBands
BitMap	BitMap exibindo no fundo do componente
FixedOrder	Indica se o usuário poderá modificar a ordem dos CollBands
FixedSize	Indica se o usuário poderá modificar o tamanho dos CollBands
ShowText	Indica se o valor da propriedade Text de cada TollBand será exibido ao lado de sua faixa
Vertical	Indica se os CollBand serão verticais
Images	Coleção de desenhos exibidos ao lado de cada CoolBand

Diálogos Comuns

Grupo de caixas de diálogo comuns a muitos programas.

Parte integrante do Windows, são atualizadas com a atualização do Windows.

Necessitam pouca ou nenhuma programação para serem utilizadas. Facilitam a padronização em tarefas comuns. Depois de serem executados os Diálogos Comuns armazenam em suas propriedades as escolhas do usuário.

Método	Descrição
Execute	Ativa a caixa de diálogo e retorna True caso o dialogo comum seja encerrado com o botão OK.

OpenDialog, SaveDialog, OpenPictureDialog e SavePictureDialog - Caixas de diálogo para abrir e salvar arquivo

Propriedades	Descrição
FileName	Nome do arquivo
DefaultExt	Extensão padrão para os arquivos
Filter	Define os tipos de arquivos que podem ser abertos ou salvos
FilterIndex	Número do filtro default
InitialDir	Diretório inicial
Title	Título da janela
Options	Define características da janela de abrir ou salvar

FontDialog - Caixa de diálogo de escolha de fonte.

Propriedades	Descrição
Device	Define se deve utilizar fontes para tela, impressora ou ambos
MinFontSize	Tamanho mínimo da fonte
MaxFontSize	Tamanho máximo da fonte
Options	Define características das fontes

Evento	Descrição
OnApply	Ocorre após o usuário pressionar o botão apply e antes da janela fechar

ColorDialog - Caixa de diálogo para escolha de cor.

Propriedades	Descrição
Options	Define características da caixa de diálogo

PrintDialog - Caixa de diálogo de impressão.

Propriedades	Descrição
Copies	Número de cópias inicial
PrintToFile	Define se a impressão será direcionada para arquivo
PrintRange	Faixa de páginas a ser impressa (todas ou somente o selecionado)
FromPage	Página inicial
ToPage	Página final
MinPage	Menor número de página que o usuário pode escolher
MaxPage	Maior número de página que o usuário pode escolher
Options	Define características da caixa de diálogo

PrintSetupDialog - Dá acesso à caixa de diálogo de configuração de impressora.

FindDialog e ReplaceDialog - Caixas de diálogo de pesquisa e substituição de texto.

Propriedades	Descrição
FindText	Texto a pesquisar
ReplaceText	Texto a substituir (somente em ReplaceDialog)
Options	Define características da caixa de diálogo

Evento	Descrição
OnFind	Ocorre após o usuário pressionar o botão Find Next
OnReplace	Ocorre quando o usuário pressiona o botão Replace (somente em ReplaceDialog)

Método	Descrição
CloseDialog	Fecha a caixa de diálogo

MainMenu e PopupMenu - Usado para criação de barras de menus, onde o MainMenu representa o Menu suspenso presente nos aplicativos e o PopupMenu representa o menu instantâneo (botão direito do mouse) associado aos componentes através da propriedade PopupMenu.

Propriedades	Descrição
Items	Itens do menu, utiliza o MenuEditor

Eventos	Descrição
OnPopup	Ocorre quando o menu popup é ativado

MenuEditor - Utilizado para definição dos itens e sub-itens do menu

Operações : Inserir, Deletar e Criar SubItems

MenuItem - Item do menu.

Propriedades	Descrição
Caption	Texto do item
Checked	Se o item está marcado ou não
Visible	Se o item está visível ou não
Enabled	Se o item está ativado ou desativado
ShortCut	Tecla de atalho
Break	Indica quebra de coluna
GroupIndex	Indica que o item do menu pertence a um grupo
RadioItem	Indica que o item de menu funcionará como um Radio, ou seja, dentro do mesmo grupo apenas um item será selecionado.

Eventos	Descrição
OnClick	Quando o item de menu é selecionado, usado para executar a função do item

Criando Aplicações MDI

Definindo uma janela pai

- FormStyle → fsMDIForm
- WindowMenu → Item do menu onde aparecerá a lista de janelas

Definindo uma janela filha

- Deve-se eliminar a criação automática da janela no projeto.
- FormStyle → fsMDIChild
- No evento OnClose, deve-se utilizar um código semelhante a:

```
Action := caFree;  
JanelaFilha := nil
```

Abrindo uma janela filha

- No Tratamento de Abertura da janela filha, deve-se utilizar um código semelhante a:

```
if JanelaFilha = nil then  
  Application.CreateForm(ClasseFilha, JanelaFilha)  
else  
begin  
  JanelaFilha.BringToFront;  
  JanelaFilha.WindowState := wsNormal  
end
```