

# Apostila de Delphi 7.0

## Conceitos Básicos

Diogo Cezar Teixeira Batista

Cornélio Procópio

9 de setembro de 2008

Apostila baseada na obra de :  
Profa. Gilene Borges Gomes  
Home page: <http://www.gomeshp.com>  
E-mail: [gilene@gomeshp.com](mailto:gilene@gomeshp.com)

## Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>7</b>
1.1	Vantagens do Delphi . . . . .	8
1.2	Desvantagens do Delphi . . . . .	8
<b>2</b>	<b>AMBIENTE DELPHI</b>	<b>8</b>
2.1	Janela: FORM . . . . .	9
2.2	Janela: OBJECT INSPECTOR . . . . .	9
2.3	Janela: CODE EDITOR . . . . .	10
2.4	Janela: OBJECT TREEVIEW . . . . .	11
2.5	Janela: MAIN . . . . .	11
2.5.1	Barra de Ferramentas . . . . .	12
2.5.2	Menu Principal . . . . .	12
2.5.3	Paleta de Componentes . . . . .	13
2.5.3.1	Paleta: STANDARD . . . . .	15
2.5.3.2	Paleta: ADDITIONAL . . . . .	15
2.5.3.3	Paleta: WIN32 . . . . .	16
2.5.3.4	Paleta SYSTEM . . . . .	17
<b>3</b>	<b>MEU PRIMEIRO PROGRAMA</b>	<b>17</b>
3.1	Desenhar as Janelas que se Deseja Usar . . . . .	17
3.2	Adaptar as Propriedades dos Objetos . . . . .	18
3.3	Escrever o Código Para os Eventos Associados . . . . .	20
3.3.1	Janela Unit . . . . .	20
3.3.2	Ajustando as Propriedades . . . . .	23
<b>4</b>	<b>AMBIENTE DE PROGRAMAÇÃO</b>	<b>25</b>
4.1	Estrutura de Projetos . . . . .	25
4.2	Project Options . . . . .	27
4.3	Gerenciamento de Projetos . . . . .	28
4.4	Ajuda . . . . .	29

---

<b>5</b>	<b>BIBLIOTECA DE CLASSES</b>	<b>29</b>
5.1	Nomenclatura . . . . .	29
5.2	Propriedades . . . . .	29
5.2.1	Propriedades Comuns . . . . .	30
5.3	Variáveis no Ambiente Delphi . . . . .	30
5.3.1	Formas de Declarar uma Variável . . . . .	31
5.4	Métodos . . . . .	31
5.4.1	Métodos Comuns . . . . .	32
5.5	Eventos . . . . .	32
5.5.1	Eventos Comuns . . . . .	32
<b>6</b>	<b>Caixas de Diálogo Predefinidas</b>	<b>34</b>
6.1	Usando o Comando MessageDLG . . . . .	34
6.2	Usando a Função Inputbox . . . . .	35
<b>7</b>	<b>Conversão de variáveis</b>	<b>36</b>
<b>8</b>	<b>Formatação de números</b>	<b>36</b>
<b>9</b>	<b>Formatação de data e hora</b>	<b>36</b>
<b>10</b>	<b>Exercícios</b>	<b>37</b>

## Lista de Figuras

1	A Janela FORM: interface . . . . .	9
2	A Janela OBJECT INSPECTOR: propriedades e eventos . . . . .	10
3	A Janela CODE EDITOR: código fonte da aplicação . . . . .	10
4	A Janela OBJECT TREEVIEW: Diagramas da aplicação . . . . .	11
5	A Janela MAIN: dividida em sete partes . . . . .	11
6	Funções da barra de ferramentas . . . . .	12
7	Como configurar os componentes das paletas. . . . .	14
8	Paleta Standard: componentes mais comuns . . . . .	15
9	Paleta Additional: mais componentes de uso comum . . . . .	16
10	Paleta Win32: aplicativos com a aparência do Windows . . . . .	16
11	Paleta System: utilizar recursos do sistema operacional . . . . .	17
12	Componentes adicionados ao form . . . . .	17
13	Propriedades do formulário alteradas: Height e Width . . . . .	18
14	Propriedades do formulário alteradas: Name e Caption . . . . .	19
15	Resultado no form . . . . .	19
16	Propriedades do TLabel e TButton alteradas . . . . .	20
17	Resultado no form . . . . .	20
18	Object inspector . . . . .	21
19	Janela Unit . . . . .	21
20	Descrição do código: Procedure . . . . .	21
21	Lista de propriedades e métodos . . . . .	22
22	Selecionando a propriedade: Digitando a primeira letra . . . . .	22
23	Resultado da aplicação . . . . .	23
24	Propriedades para alinhamento do label . . . . .	23
25	Alterando a largura do label . . . . .	24
26	Alterando a propriedade font . . . . .	24
27	Caixa de diálogo para seleção da fonte . . . . .	24
28	Exibição parcial do label . . . . .	25
29	Labels alinhados corretamente . . . . .	25

## Lista de Tabelas

1	Propriedades e Métodos de um Carro . . . . .	7
2	Descrição dos componentes da paleta standard . . . . .	15
3	Descrição dos componentes da paleta standard . . . . .	16
4	Descrição das extensões geradas pelo Delphi . . . . .	26
5	Descrição dos itens do Project Options . . . . .	27
6	Descrição das mais importantes opções de menu . . . . .	28
7	Propriedades mais comuns dos componentes no delphi . . . . .	30
8	Variáveis no ambiente delphi . . . . .	31
9	Métodos mais comuns dos componentes no delphi . . . . .	32
10	Eventos mais comuns dos componentes no delphi . . . . .	33

## **Lista de Códigos**

1	Atribuindo valor para a propriedade de um objeto . . . . .	22
2	Estrutura básica de um projeto. . . . .	26
3	Propriedades dos componentes . . . . .	29
4	Declaração de variável . . . . .	31
5	Componentes Message Dialog . . . . .	34

## 1 INTRODUÇÃO

Delphi possui um ambiente de desenvolvimento fácil de usar, com uma grande Biblioteca de Componentes Visuais (VCL - Visual Component Library). A VCL contém código de botões, campos, rótulos, gráficos, caixas de diálogo e acesso a tabelas de bancos de dados, e foi desenvolvida levando em conta as velocidades no desenvolvimento de aplicativos e na execução destes aplicativos.

O rápido desenvolvimento de aplicativos é possível graças aos vários controles disponíveis na paleta de componentes, onde o programador escolhe um destes componentes, e coloca-o diretamente no local desejado, dentro de um formulário. Formulário este que será a janela do aplicativo apresentada ao usuário.

O Delphi permite o uso de objetos, e sua criação. Ele trabalha com eventos que dão início à alguma rotina de trabalho, ou seja, o programa fica parado até que um evento ocorra. Um programa tradicional, feito para ser executado em DOS, é organizado em torno de estruturas de dados com um loop principal e uma série de sub-rotinas constituindo o aplicativo, com procedimentos e funções separados para manipular os dados.

Um programa **orientado a objetos** e eventos é organizado em torno de um conjunto de objetos. Onde cada objeto possui propriedades que o definem, e vários códigos (eventos) dando funcionalidade a este objeto. Ou seja, objetos são estruturas que combinam dados e funções em uma mesma estrutura. Um Objeto possui dados internos, que não podem ser acessados por outros objetos e dados externos, também chamados de propriedades, estas podendo ser acessadas de fora deste objeto. De maneira semelhante, um objeto possui rotinas internas que são usadas apenas internamente e rotinas externas, também chamadas de métodos, que podem ser acessadas externamente.

Um carro é um objeto que possui propriedades e métodos. A tabela abaixo lista algumas propriedades e comportamentos do objeto real carro.

Tabela 1: Propriedades e Métodos de um Carro

Propriedades	Métodos
cor	dar a partida
potência do motor	acelerar
tipo de pintura	frear

Um método é uma rotina própria do objeto que o dá funcionalidade, ou seja, torna-o vivo,

e as propriedades fazem o intercâmbio entre o objeto e o programa.

### 1.1 Vantagens do Delphi

- Facilidade em alterações e implementações;
- Estruturação do código;
- Velocidade;
- Orientação a objetos;

### 1.2 Desvantagens do Delphi

- Pouca portabilidade;
- Integridade: quando se apaga um botão, as rotinas criadas para ele não se apagam.
- Os erros de compilação são reportados acrescentado-se o caminho completo de onde se encontra a Unit. Há dois problemas em relação a isso: consome espaço na tela; e não existe barra de rolagem para ver o final da linha.
- Existem quatro opções de salvamento diferentes. Quem não sabe exatamente como funciona cada uma delas tem bastante dificuldade para armazenar o trabalho realizado e recuperar mais tarde.
- As mensagens Yes, No, Cancel, etc., não assumem a configuração do Windows. Para traduzí-las temos que comprar o Borland Language Packet, produto adicional para a localização de software.
- O ambiente de programação do Delphi é SDI. Quando se abrem diversos formulários simultaneamente, a distribuição deles na tela fica confusa e não há como organizá-los automaticamente.

## 2 AMBIENTE DELPHI

Neste item será estudado o IDE (Integrated Developer Environment – Ambiente de Desenvolvimento Integrado) do Delphi.



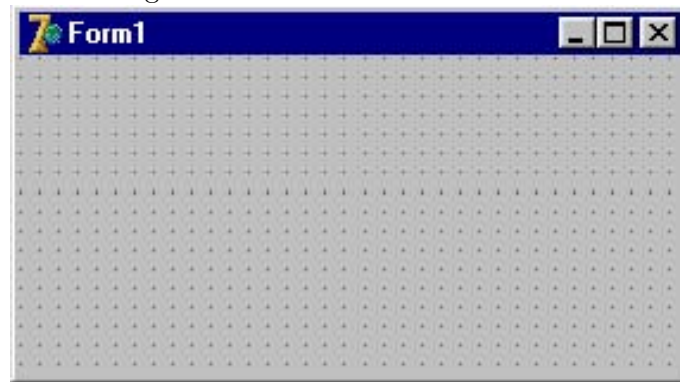
O Delphi possui um conjunto de ferramentas que permitem facilitar e agilizar a construção de programas, permitindo uma melhor interação entre o programador e o computador. Suas principais janelas são:

- Janela FORM;
- Janela OBJECT INSPECTOR;
- Janela CODE EDITOR;
- Janela OBJECT TREEVIEW;
- Janela MAIN.

## **2.1 Janela: FORM**

O FORM é a tela onde o desenvolvedor constrói sua aplicação. A partir de um FORM é que se estabelece a interação USUÁRIO-COMPUTADOR, através de botões, rótulos e outros componentes, estabelecendo-se funções, métodos ou eventos que serão ativados. Os componentes são dispostos dentro da área útil do FORM.

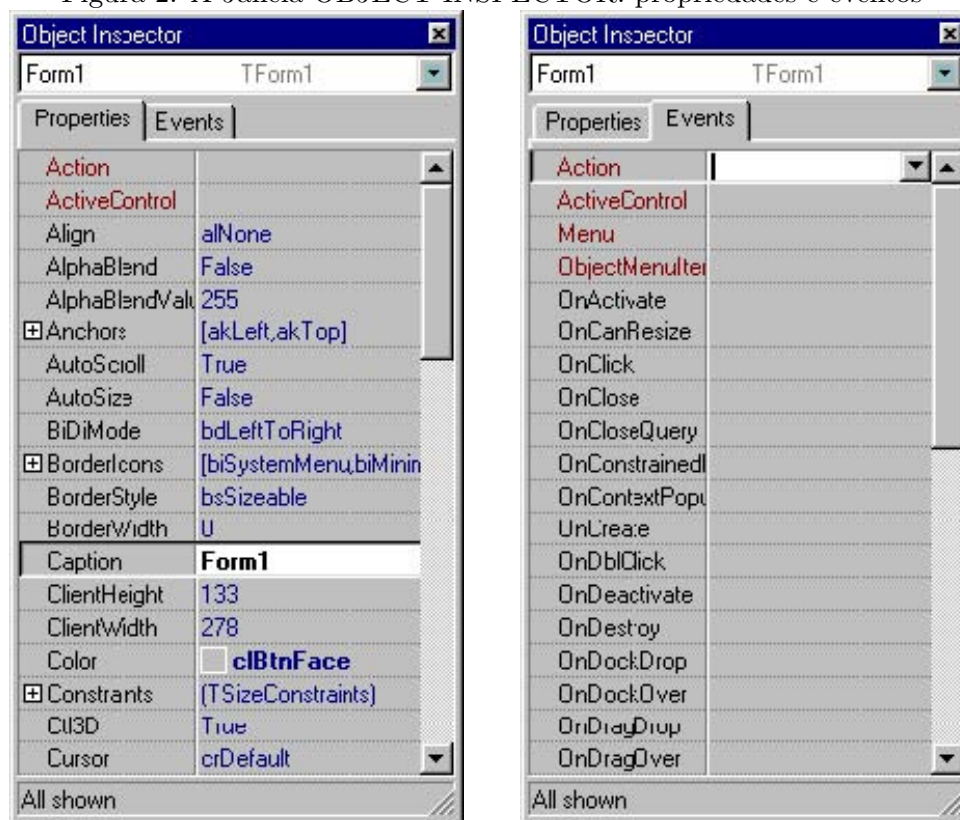
Figura 1: A Janela FORM: interface



## **2.2 Janela: OBJECT INSPECTOR**

A janela OBJECT INSPECTOR contém propriedades e eventos dos componentes inseridos em um FORM, e do próprio FORM. É na guia Properties (Propriedades), por exemplo, que se estabelecem as características de cada componente, como nome, fonte, altura, largura, etc. Já na guia Events (Eventos) estabelecem-se ações a serem tomadas pelo componente a partir de um evento associado ao mouse, teclado, sistema operacional, etc.

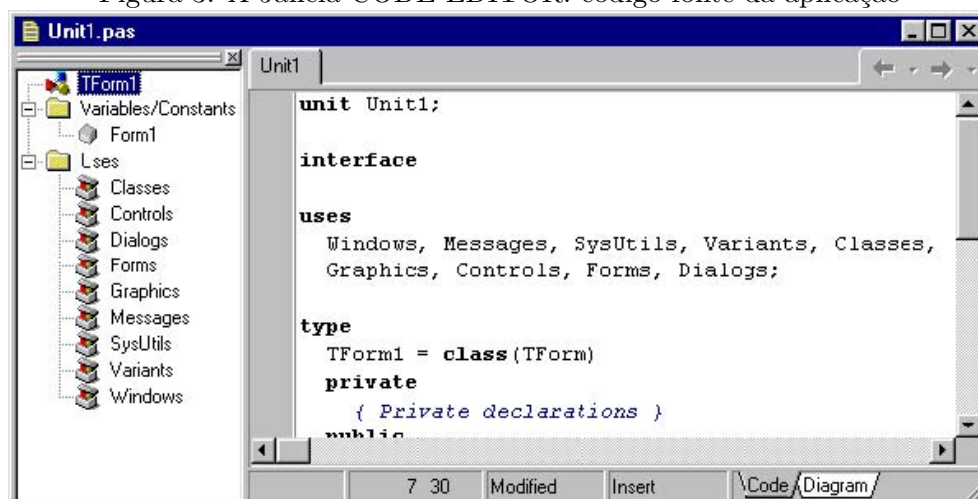
Figura 2: A Janela OBJECT INSPECTOR: propriedades e eventos



## 2.3 Janela: CODE EDITOR

A janela CODE EDITOR, ou editor de código, é onde se desenvolve o programa fonte. É neste editor que se encontra a estrutura sintática propriamente dita da Linguagem Object Pascal, que é utilizada pelo Delphi. Cabe ressaltar, no entanto, que boa parte do código escrito é gerado automaticamente.

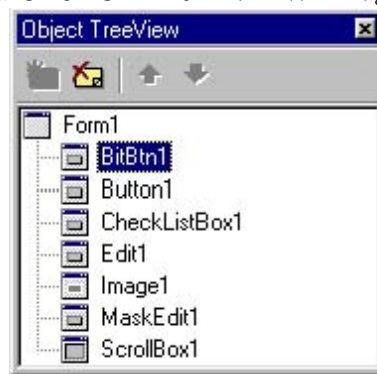
Figura 3: A Janela CODE EDITOR: código fonte da aplicação



## 2.4 Janela: OBJECT TREEVIEW

Object TreeView apresenta uma árvore do diagrama dos componentes visuais e não visuais colocados no formulário, no módulo de dados ou no frame.

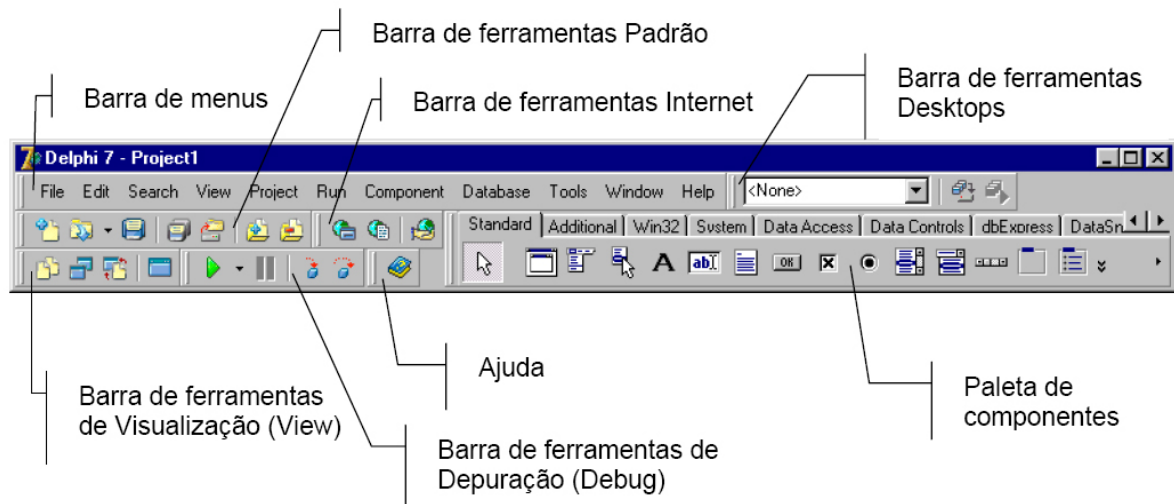
Figura 4: A Janela OBJECT TREEVIEW: Diagramas da aplicação



## 2.5 Janela: MAIN

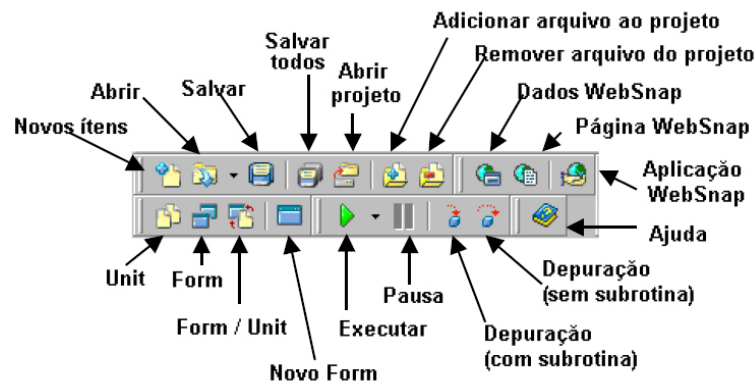
A janela MAIN, ou janela principal, controla o funcionamento do Delphi. Esta janela pode ser dividida em sete partes:

Figura 5: A Janela MAIN: dividida em sete partes



### 2.5.1 Barra de Ferramentas

Figura 6: Funções da barra de ferramentas



### 2.5.2 Menu Principal

O menu principal contém as opções de utilização do Delphi:

- **File:** permite a manipulação de arquivos do desenvolvedor (PAS, DPR, ...);
- **Edit:** apresenta opções de edição;
- **Search:** apresenta opções de pesquisa e localização;
- **View:** permite verificar detalhes do projeto;
- **Project:** permite adicionar ou remover partes em um projeto, bem como compilá-lo;
- **Run:** apresenta opções de execução e depuração do projeto;
- **Component:** permite a criação ou instalação de novos componentes no Delphi;
- **Database:** apresenta opções de uso de banco de dados;
- **Tools:** permite configurar o ambiente de trabalho, bem com acessar ferramentas externas ao Delphi;
- **Window:** permite alternar entre as principais janelas do Delphi;
- **Help:** ajuda do Delphi.

### **2.5.3 Paleta de Componentes**

A Paleta de Componentes possui todos os controles necessários para desenharmos nossa janela - formulário - como um programa de desenho livre. Para incluir um controle no formulário, existem dois métodos:

1. Click Duplo no ícone da paleta de componentes. Fará com que o controle seja inserido no centro do formulário com um tamanho padrão.
2. Selecionar o ícone na caixa de ferramentas e depois dar um clique no formulário, na posição desejada para o objeto (canto superior esquerdo deste).

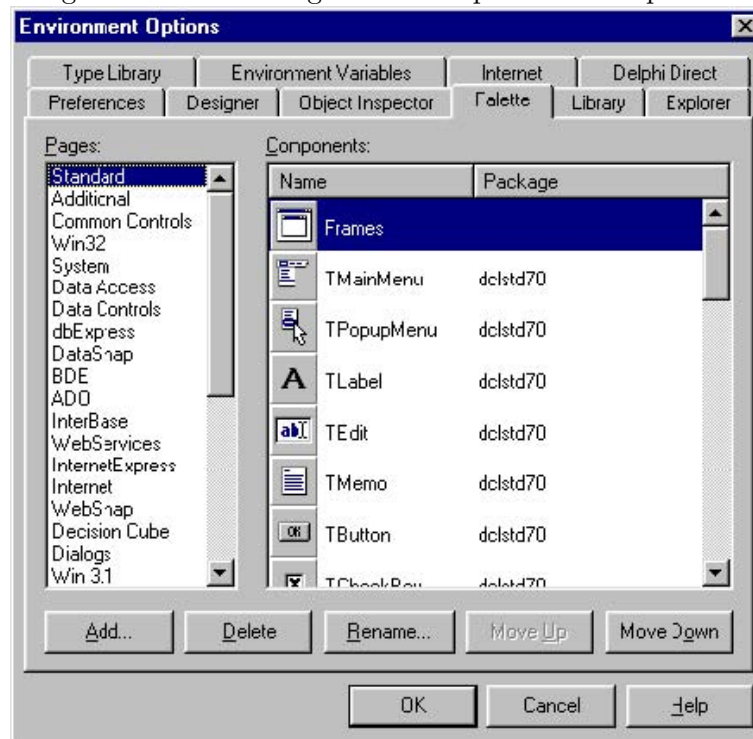
Podemos dimensionar estes controles, depois de inseridos, a qualquer momento durante o desenvolvimento. Primeiro seleciona o controle dando um clique em cima dele e depois o dimensionamos arrastando um dos oito botões dimensionadores que circundam este objeto.

A paleta de componentes é a biblioteca de classes que fornece recursos para o desenvolvimento visual em Delphi. As classes representadas na paleta de componentes estão separadas por tipos, as paletas que serão utilizadas neste curso são:

- **Standard:** componentes mais comuns e usados.
- **Additional:** componentes adicionais também de uso comum.
- **Win32:** componentes para acesso de controles comuns de interface de usuário do Windows 32-bits.
- **System:** componentes para aproveitar recursos de sistema operacional.

Os componentes podem ser incluídos ou excluídos da paleta de componentes. Basta abrir a caixa de diálogo Environment Options do menu Tools e selecionar a guia Palette.

Figura 7: Como configurar os componentes das paletas.



Os componentes disponíveis na VCL podem ser divididos entre:

- **COMPONENTES VISUAIS** → Podem ter sua forma e tamanho alterados no formulário (Form), além das propriedades e eventos no Object Inspector. Eles aparecem durante a execução do aplicativo exatamente como foram definidos durante o projeto.
- **COMPONENTES NÃO-VISUAIS** → Ficam apenas como a representação de um ícone no formulário (Form), mas suas propriedades e eventos podem ser alterados no Object Inspector. Eles não aparecem no formulário durante a execução do aplicativo, podendo ser ativados por comandos específicos (por exemplo, podemos citar a caixa de diálogo abrir arquivo).

**2.5.3.1 Paleta: STANDARD** A paleta Standard contém 16 componentes mais comuns para a construção de aplicações.

Figura 8: Paleta Standard: componentes mais comuns



Os componentes desta paleta, respectivamente, são:

Tabela 2: Descrição dos componentes da paleta standard

Frames	Abre uma caixa de diálogo mostrando uma lista de frames incluídos no projeto corrente.
MainMenu	Permite a construção da barra de menus e de menus suspensos.
PopupMenu	Permite a construção de menus a partir do botão direito do mouse.
Label	Permite colocar textos que não podem ser selecionados ou alterados pelo usuário.
Edit	Permite a apresentação ou a entrada de dados pelo usuário.
Memo	Permite a introdução ou exibição de uma área de texto.
Button	Permite a colocação de botões para inicialização de ações por parte do usuário.
CheckBox	Permite a colocação de caixa de verificação para a seleção de diversas opções.
RadioButton	Permite a colocação de botões de seleção de onde pode ser selecionada apenas uma opção.
ListBox	Apresenta uma lista de itens que podem ser selecionados.
ComboBox	Apresenta uma lista de itens de onde pode ser selecionado apenas um. Este componente também permite que o usuário digite sua própria opção.
ScrollBar	Permite criar as barras de rolagem verticais ou horizontais, no padrão do Windows®.
GroupBox	Permite agrupar controles como CheckBox, RadioButton, etc.
RadioGroup	Permite agrupar RadioButtons para que se faça a seleção de uma opção.
Panel	Cria painéis que contém outros componentes num formulário. São utilizados para construir barra de status, barra de ferramentas, etc.
ActionList	Cria coleções de ações que centraliza as respostas da aplicação para as ações do usuário.

**2.5.3.2 Paleta: ADDITIONAL** A paleta Additional tem 25 componentes, também de uso comum, mas com algumas funções mais especializadas.

Os primeiros componentes desta paleta, respectivamente, são:

Figura 9: Paleta Additional: mais componentes de uso comum

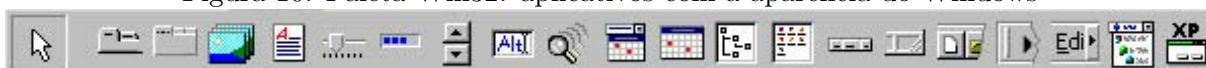


Tabela 3: Descrição dos componentes da paleta standard

BitBtn	Permite a colocação de botões com imagem bitmap.
SpeedButton	Permite a criação de barra de ferramentas e conjuntos de botões. Devem ser utilizados juntamente com o componente Panel.
MaskEdit	Permite a entrada de dados definindo-se máscaras de leitura.
StringGrid	Permite a apresentação de strings em colunas.
DrawGrid	Permite a apresentação de informações em colunas e linhas.
Image	Permite a apresentação de imagens gráficas.
Shape	Permite o desenho de figuras geométricas.
Bevel	Permite o desenho de retângulos em relevo.
ScrollBar	Cria áreas de exibição com barras de rolagem, quando necessário.
CheckListBox	Similar ao ListBox onde cada item tem um CheckBox.

**2.5.3.3 Paleta: WIN32** A paleta Win32 contém 18 componentes para criar aplicações que tenham a aparência do Windows95©.

Figura 10: Paleta Win32: aplicativos com a aparência do Windows





**2.5.3.4 Paleta SYSTEM** A paleta System contém 8 componentes que permitem utilizar em suas aplicações alguns recursos do sistema operacional.

Figura 11: Paleta System: utilizar recursos do sistema operacional



### 3 MEU PRIMEIRO PROGRAMA

Para iniciar, vamos construir um programa que quando for dado um clique no botão de comando, será mostrada uma mensagem. E posteriormente poderemos alterar a cor desta mensagem através de outros botões.

Existem três passos principais para a escrita de uma aplicação no Delphi que iremos seguir:

1. **Desenhar as janelas que se deseja usar**

Inserir no formulário os controles que serão necessários.

2. **Adaptar as propriedades dos objetos**

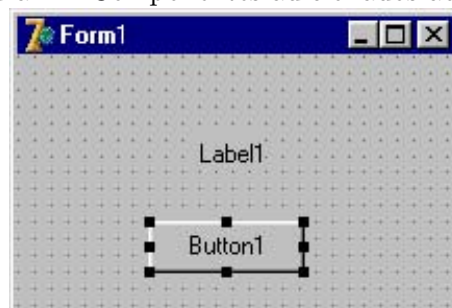
Alterar as propriedades dos controles às necessidades da aplicação.

3. **Escrever o código para os eventos associados**

Esta é a parte mais complexa do desenvolvimento, é ela que dá a funcionalidade ao programa, são as rotinas que começam a ser executadas a partir de um evento.

#### 3.1 Desenhar as Janelas que se Deseja Usar

Figura 12: Componentes adicionados ao form



1. Começamos inserindo um Label (Legenda) e um Button (Botão de Comando) no Formulário.

2. Observe que, quando o controle estiver selecionado, poderemos arrastá-lo e dimensioná-lo dentro do formulário.

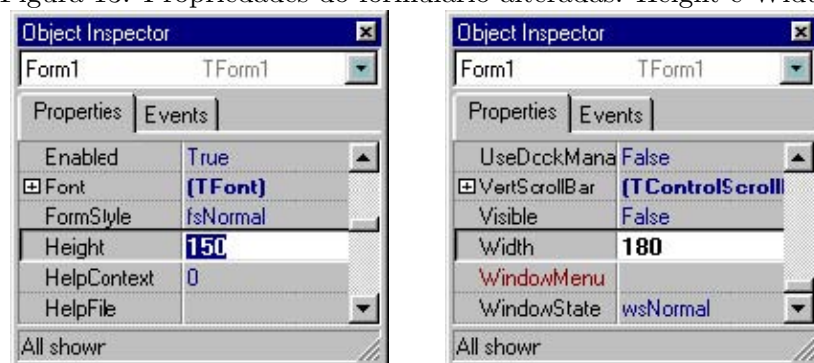
### 3.2 Adaptar as Propriedades dos Objetos

Para se alterar a propriedade de um objeto, ele tem que estar selecionado (com os oito pontos dimensionadores visíveis), depois procurar o nome da propriedade a ser alterada, na janela Object Inspector, e selecionar (no caso de valores padrão) o seu valor, ou então escrever um valor.

Dimensione o formulário da seguinte maneira:

- Selecionar a propriedade Height, e atribuir a ela o valor de 150.
- Selecionar a propriedade Width e dar o valor de 180.

Figura 13: Propriedades do formulário alteradas: Height e Width



Estes números correspondem a Pixels, que é a quantidade de pontos do monitor.

O mesmo deverá ser feito para as propriedades Name e Caption. A propriedade Name será a identificação do Objeto quando construirmos o código da aplicação. E a propriedade Caption é a palavra que aparecerá como título da janela.

Figura 14: Propriedades do formulário alteradas: Name e Caption



Após você alterar estas quatro propriedades (Caption, Height, Name e Width) do formulário, ela estará assim:

Figura 15: Resultado no form



Agora, altere as propriedades Caption e Name dos componentes **TLabel** e **TButton**.

Figura 16: Propriedades do TLabel e TButton alteradas

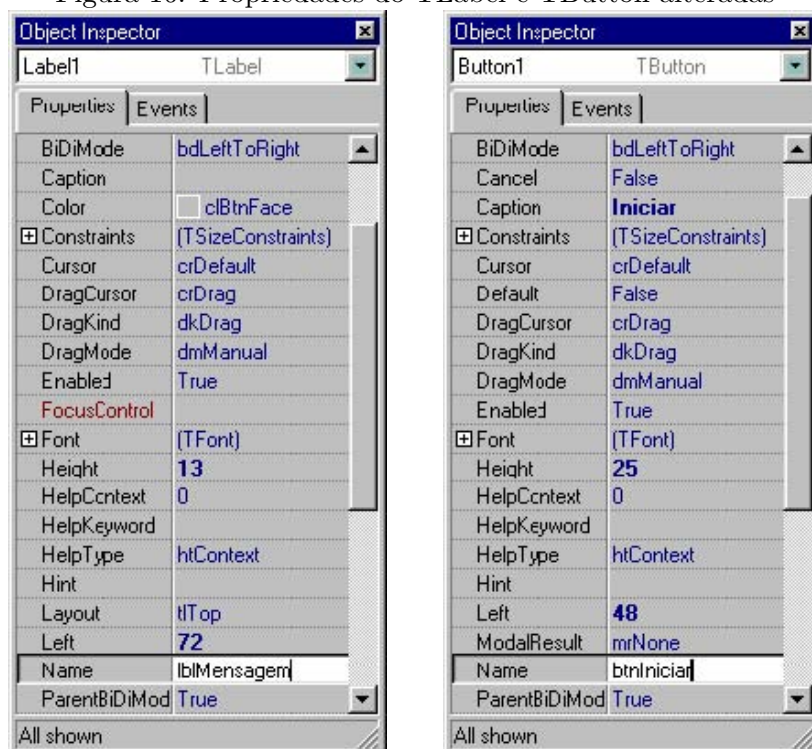


Figura 17: Resultado no form



### 3.3 Escrever o Código Para os Eventos Associados

O código é escrito na janela Unit, para acessá-la, selecione o botão Iniciar e na janela Object Inspector, selecione a guia Events e dê um duplo clique na parte direita da linha que contém o evento OnClick - a rotina escrita para este evento, será executada quando o botão Iniciar for clicado. Isto traz a janela Unit para a frente.

#### 3.3.1 Janela Unit

Nesta janela observamos o nome da procedure, identificando qual o objeto e o evento que dará início à execução do código, e onde está localizado este objeto.

Figura 18: Object inspector



Figura 19: Janela Unit

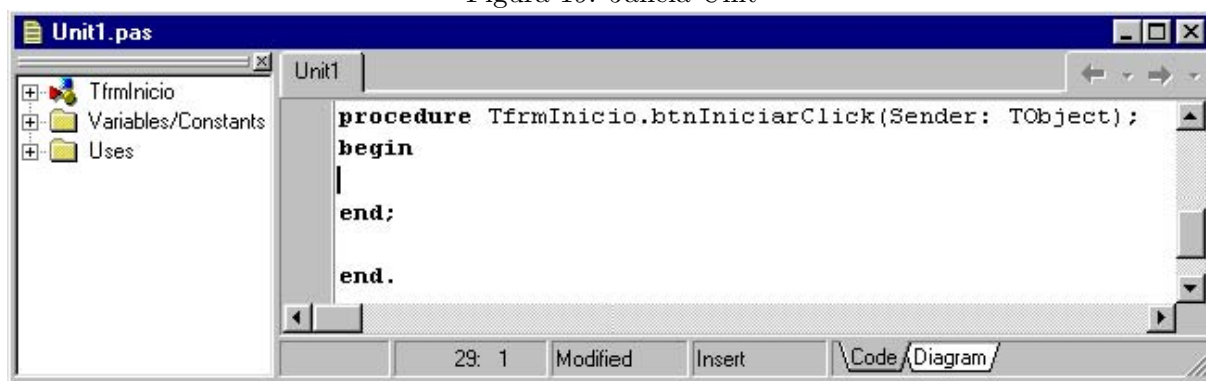
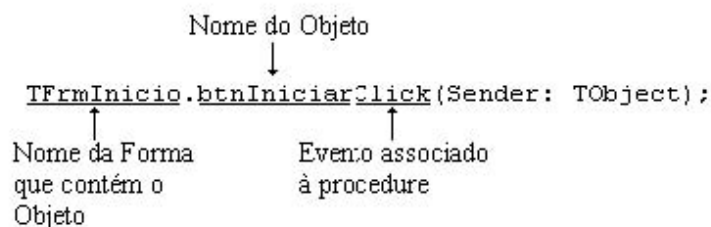


Figura 20: Descrição do código: Procedure



Todas as instruções a serem executadas por um procedimento devem estar entre as palavras reservadas `begin` e `end`.

A Janela Unit também pode ser acessada dando-se um duplo clique no objeto que se quer criar um código. Cada objeto tem um evento que é mais comumente utilizado, e é com este evento que o Delphi iniciará a Janela Unit quando acessada desta forma, isto não impede que criemos outros códigos utilizando mais de um evento ao mesmo tempo.

O nosso projeto de Início, mostrará uma mensagem no Label (objeto) com um Click (evento) no Botão "Iniciar" (objeto). Ou seja, iremos alterar a propriedade `Caption` de `lblMensagem`, esta propriedade contém o que será mostrado ao usuário.

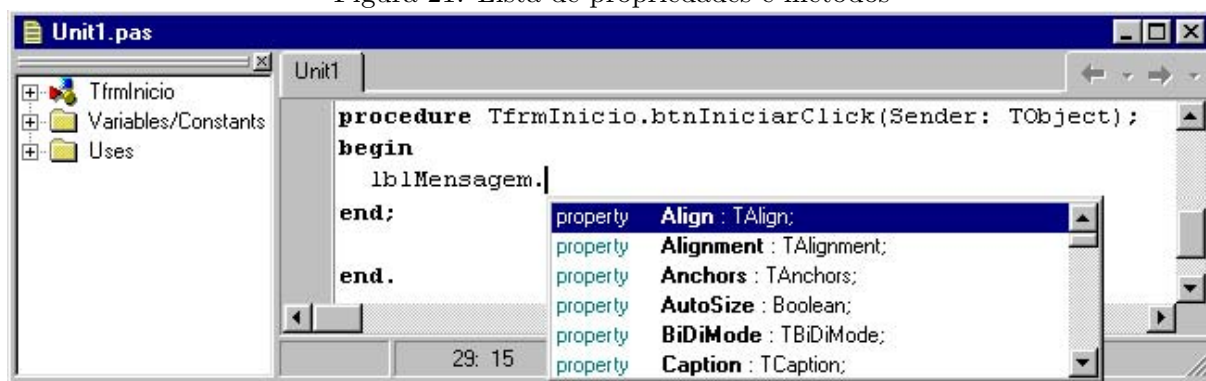
Atribuímos valores a uma propriedade de objeto seguindo o padrão:

Código 1: Atribuindo valor para a propriedade de um objeto

```
1 objeto + . + propriedade + := + valor da propriedade;
```

Abra a Janela Unit para o botão de comando e digite o código conforme a figura a seguir. Repare que ao digitar o ponto após lblMensagem, e aguardando alguns instantes, o Delphi exibirá uma lista de propriedades e métodos do controle Label.

Figura 21: Lista de propriedades e métodos



Esta ajuda do Delphi pode ser acionada para qualquer controle ou função, quando digitamos o nome de uma função, ele exibe os parâmetros necessários para a execução desta função.

Para escolher uma propriedade do Label lblMensagem, selecione-a com as setas de direção e então pressione Enter, inserindo-a na linha de comando. Ou então, digite a primeira letra da propriedade, selecionando-a.

Figura 22: Selecionando a propriedade: Digitando a primeira letra




Clique sobre o botão Run da barra de ferramentas () para que o Delphi inicie a compilação do projeto. Em seguida, dê um clique no botão Iniciar para ver o resultado.

Figura 23: Resultado da aplicação



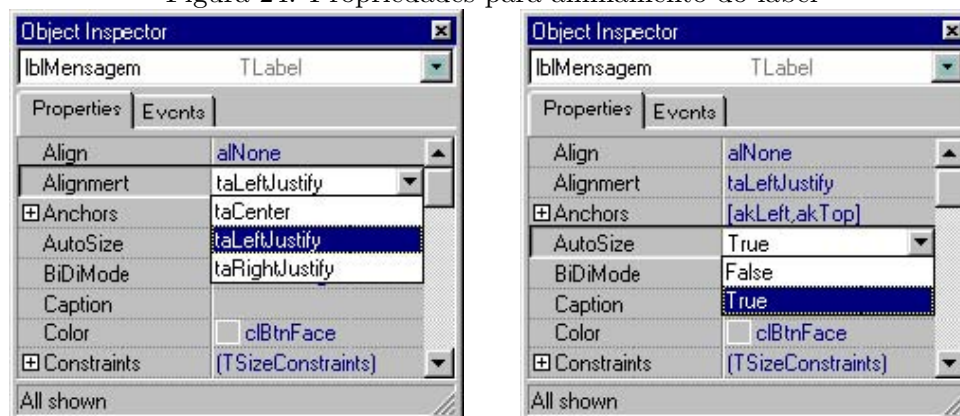
Se o seu formulário se parecer com o apresentado à esquerda, reposicione o seu componente Label. Seu formulário deve ficar como o apresentado a direita.

Finalize a execução do projeto teclando Alt+F4 ou no botão Finalizar da barra de título da janela.

### 3.3.2 Ajustando as Propriedades

Existem propriedades que possuem valores predefinidos, quando escolhemos a propriedade Alignment e damos um clique na seta da caixa de valor, aparecem os tipos de alinhamento para o texto.

Figura 24: Propriedades para alinhamento do label



Selecione o objeto lblMensagem através da Caixa de Objeto da janela Object Inspector, e altere a propriedade Alignment para taCenter, para que o texto no TLabel fique centralizado. Altere também a propriedade AutoSize para False, e no Formulário aumente a largura do TLabel.

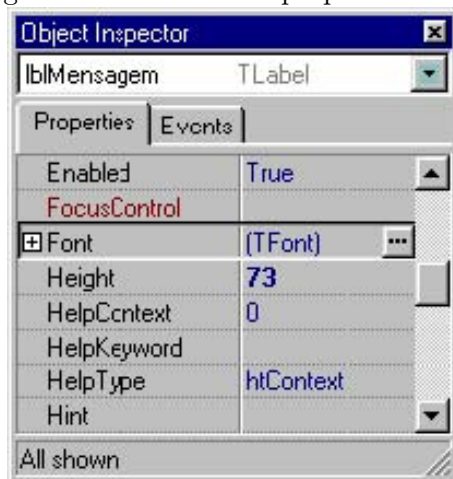
Além das propriedades descritas acima, com padrões pré-definidos, existem outras que possuem inúmeras escolhas, neste caso, ao invés de uma seta, observaremos três pontos, este é o caso da propriedade Font.



Figura 25: Alterando a largura do label

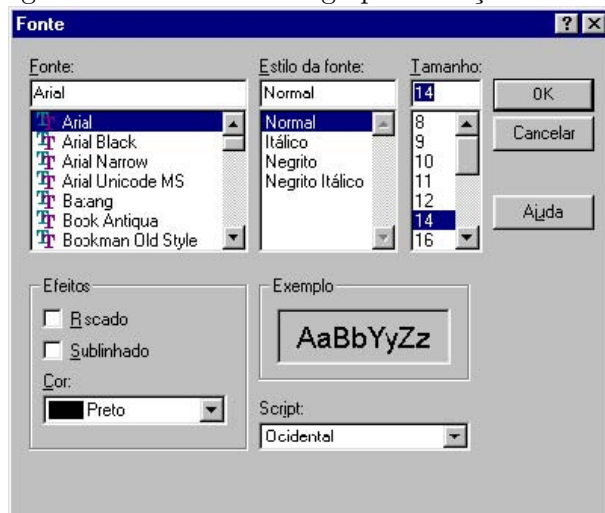


Figura 26: Alterando a propriedade font



Quando selecionamos os três pontos, aparece uma caixa de diálogo onde escolheremos o formato da fonte que será apresentada a mensagem.

Figura 27: Caixa de diálogo para seleção da fonte





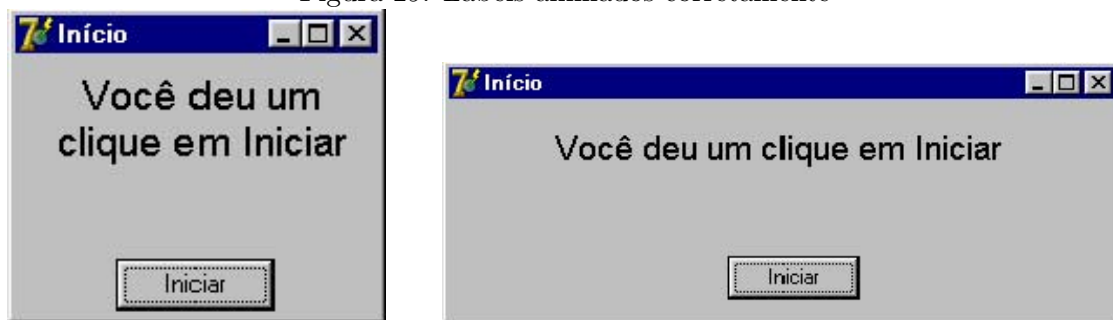
No seu projeto teste as alterações de fonte e observe as mudanças. Na figura ao lado, foi utilizada a fonte Arial com tamanho de 14 pontos. Observe que o texto não coube na área de exibição do TLabel e nem do Formulário, existem duas opções para que este texto apareça integralmente.

Figura 28: Exibição parcial do label



A primeira, é alterar para True, a propriedade WordWrap do TLabel, esta propriedade insere uma mudança de linha quando o texto atinge a margem direita do objeto. A segunda, é redimensionar os tamanhos da TLabel e do Formulário. Como mostram as figuras a seguir:

Figura 29: Labels alinhados corretamente



Salve o seu projeto, selecionando a opção Save All do menu File. Primeiramente, será solicitado o nome da Unit, dê o nome de U\_1oprograma.pas. Posteriormente, será solicitado o nome do Projeto, dê o nome de P\_1oprograma.dpr.

## 4 AMBIENTE DE PROGRAMAÇÃO

### 4.1 Estrutura de Projetos

No Delphi o "programa principal" tem extensão DPR. Um DPR é o código de programa, também denominado de projeto, a partir do qual o desenvolvedor construirá sua aplicação. É

a partir deste projeto que será gerado o código executável (EXE). Grande parte do código, no entanto, fica armazenado em UNITS com extensão PAS, onde estão o código fonte referente aos formulários.

O código fonte do projeto pode ser visualizado através da opção do menu **Project** → **View Source**. A estrutura aparece na janela Code Editor e pode-se verificar a utilização da linguagem Object Pascal©.

A implementação mínima de um projeto prevê, na grande maioria das situações, o uso (cláusula Uses), a inicialização da aplicação (comando Application.Initialize), a criação dos formulários (comando Application.CreateForm) utilizados na mesma e a execução do projeto (Application.Run).

Código 2: Estrutura básica de um projeto.

```
1 program Project1; uses
2     Forms,
3     Unit1 in 'Unit1.pas' {Form1};
4 {$R *.res} begin
5     Application.Initialize;
6     Application.CreateForm(TForm1, Form1);
7     Application.Run;
8 end.
```

Um projeto em Delphi é dividido em módulos, chamados Units, seguindo a estrutura de arquivos descrita na tabela abaixo.

Tabela 4: Descrição das extensões geradas pelo Delphi

Extensão	Descrição
DPR	Arquivo de projeto, onde são indicados as Units e o código de inicialização do programa
PAS	Código fonte de uma Unit do projeto
DFM	Definição visual de um Form. O código fonte está na Unit com o mesmo nome
DOF	Opções de configuração para o projeto
RES	Recursos do projeto, com o ícone do programa
DCU	Unit compilada
~PA, ~DF, ~DP	Arquivos temporários
DSK	Configurações de Desktop

## 4.2 Project Options

Através do item Options do menu Project, pode-se escolher diversos aspectos de um projeto.

Tabela 5: Descrição dos itens do Project Options

Forms	Controla quais são os formulários criados automaticamente
Application	Especifica o título, o nome do arquivo de ajuda e o nome do ícone associado com a aplicação
Compiler	Especifica as opções gerais para o compilador que determina como o código é compilado
Compiler Messages	Controla se as dicas e advertências do compilador estão ativadas, e permite selecionar quais avisos serão apresentados
Linker	Gerencia como os arquivos de programas são linkados
Directories/Conditionals	Especifica a localização dos arquivos necessária para compilar e linkar seu programa
Version Info	Especifica as informações de identificação do produto: número da versão, língua
Packages	Especifica os pacotes de tempo de execução e de projeto para instalar seu projeto

### 4.3 Gerenciamento de Projetos

Segue uma descrição das mais importantes opções de menu para o gerenciamento de projetos, algumas dessas opções têm um botão correspondente na barra de ferramentas.

Tabela 6: Descrição das mais importantes opções de menu

File	
New	Abre um diálogo com novos itens que podem ser adicionados ao projeto
New Form	Adiciona um novo formulário ao projeto
Open	Abrir projetos pode abrir também Units, Forms e texto no editor de código
Save	Salva o arquivo aberto no editor de código
Save Project As	Salva o projeto com outro nome ou local
Save All	Salva o projeto e todas as Units deste projeto
Use Unit	Faz com que a Unit atual possa usar outra Unit do projeto
Add file to Project	Adiciona uma Unit em disco ao projeto
Remove file from Project	Remove uma Unit do projeto
View	
Project Manager	Mostra o gerenciador de projeto
Project Source	Mostra o código do projeto
Object Inspector	Mostra o Object Inspector
Toggle Form/Unit	Alterna entre o Form e a Unit
View Units	Mostra o código fonte de uma Unit ou do Projeto a partir de uma lista
View Forms	Seleciona um Form a partir de uma lista
Project	
Compile	Compila o projeto
Options	Opções do projeto, como ícone do executável, nome da aplicação e opções de compilação
Run	
Run	Compila e executa o projeto

## 4.4 Ajuda

O sistema de ajuda do Delphi é a referência mais completa, seguida pelos manuais do usuário cedidos com o sistema. Se quiser ajuda sobre um componente, selecione-o e aperte F1, o mesmo pode ser feito com propriedades e eventos, no Object Inspector e comandos, no editor de código.

# 5 BIBLIOTECA DE CLASSES

## 5.1 Nomenclatura

Para nomear os componentes podemos usar uma convenção muito utilizada, onde as primeiras letras, minúsculas, identificam o tipo do componente e o restante identifica a função deste, assim, btnSair, seria o nome do botão de sair.

Se a função do componente for um nome composto esse nome deve ser escrito com os primeiros nomes abreviados e com letras de caso variável, como em btnRelVendas, que seria o botão do relatório de vendas ou btnRelVenProduto, que seria o botão do relatório de vendas por produto.

## 5.2 Propriedades

As propriedades são características dos componentes, como foi mostrado anteriormente. Para alterar propriedades em código use a sintaxe de ponto, como mostrado abaixo.

Código 3: Propriedades dos componentes

```
1  \\Tipo String
2
3  Button1.Caption := 'Fechar';
4  Label1.Caption := Edit1.Text + ', ' + Edit2.Text;
5
6  \\Tipo Numérico
7
8  Button2.Height := Button2.Height * 2;
9  Button3.Width := Button1.Width + Button2.Width + 12;
10
11 \\Tipo Enumerado
12
13 BorderStyle := bsDialog;
14 Panel1.Color := clWindow;
```

```
15
16  \\Propriedades Aninhadas de Classe
17
18  Memo1.Lines.Text := 'E agora , José?';
19  Label1.Font.Color := clBlue;
```

### 5.2.1 Propriedades Comuns

As propriedades mais comuns, presentes nos componentes do Delphi são:

Tabela 7: Propriedades mais comuns dos componentes no delphi

Nome	Descrição
Align	Determina o alinhamento de controle
Caption	Legenda do componente (com o &, indica atalho)
Name	Nome da instância do componente
Left	Posição esquerda
Top	Posição superior
Height	Altura do componente
Width	Largura do componente
Color	Indica a cor do componente
Font	Indica a fonte (letra) a ser usada no componente
Enabled	Indica se o componente está ativado ou não
Visible	Indica se o componente está visível ou não
Hint	String utilizada para dicas instantâneas
ShowHint	Mostra ou não as dicas instantâneas
PopupMenu	Menu que será mostrado com o click do botão direito do mouse
TabOrder	Define a ordem de tabulação do componente (tecla TAB)
TabStop	Indica se o componente será ponto de parada para a tecla TAB

### 5.3 Variáveis no Ambiente Delphi

Variável é um local reservado da memória, onde são guardados dados que podem ser mudados em tempo de execução. O nome de uma variável pode ter até 255 caracteres, tem que começar com uma letra, não pode conter caracteres e deve ser única. O nome pode conter números e sublinhados e não pode ser uma palavra reservada. Não existe diferença entre letras maiúsculas

e minúsculas. Existem vários tipos de variáveis, dependendo do tipo de dados que queremos que ela armazene.

Tabela 8: Variáveis no ambiente delphi

Tipo	Nome	Bytes	Faixa
Lógicos	Boolean	1	1 byte booleano
Inteiros	Integer	2	-32768 a 32767
Reais	Real	6	2,9.10-39 a 1,7.1038
Reais	Double	8	5.10-324 a 1,7.10308
Caracteres	String	255	1 byte cada caracter

### 5.3.1 Formas de Declarar uma Variável

As variáveis são declaradas usando-se a palavra reservada `Var`, o nome da variável, dois pontos e o tipo:

Código 4: Declaração de variavel

```
1 Var Valor1 : Real;
```

Elas podem ser declaradas em três locais diferentes, conforme a sua abrangência:

- **Variável Local:** ela será utilizada somente pelo procedimento onde está declarada; terminado o procedimento ela desaparecerá da memória. É declarada logo após o cabeçalho do procedimento.
- **Variável de Unidade (Unit):** a variável será utilizada por todos os procedimentos e funções da unidade. É declarada logo após a palavra reservada `implementation`.
- **Variável de projeto:** é a variável que poderá ser utilizada por toda a aplicação, ou seja, poderá ser utilizada por outras unidades. É declarada na seção `interface` da unidade.

## 5.4 Métodos

Os métodos são as ações realizadas pela classe, ou seja, são as funções às quais estão associados os comportamentos da classe.

### 5.4.1 Métodos Comuns

Os métodos mais comuns das classes do Delphi são:

Tabela 9: Métodos mais comuns dos componentes no delphi

Nome	Descrição
Create	Cria uma nova instância do objeto
Destroy	Destrói a instância do objeto
Show	Torna o componente visível
Hide	Torna o componente invisível
SetFocus	Coloca o foco no componente
Focused	Determina se o componente está com o foco
BringToFront	Coloca o componente na frente dos demais
SendToBack	Coloca o componente atrás dos demais

## 5.5 Eventos

Os eventos são blocos de comandos que associam a ocorrência de alguma atividade a uma ação a ser tomada. O Delphi possui mais de 130 eventos pré-definidos, que estão associados à ocorrência de ações, nas mais diversas classes.

Uma linguagem de programação orientada a eventos não tem necessariamente um início ou final lógico, isto é, de acordo com uma sequência de comandos, pois as ações só são realizadas a partir do instante que ocorre um evento.

Um evento pode estar associado:

- ao clique ou movimento do mouse;
- ao acionamento do teclado;
- as operações sobre janelas (abrir, fechar, criar, redimensionar, mover, ...);
- aos erros de execução; etc.

### 5.5.1 Eventos Comuns

Alguns dos principais eventos do Delphi são:



Tabela 10: Eventos mais comuns dos componentes no delphi

Nome	Descrição
OnActivate	Ocorre quando o programa ativa o objeto pela primeira vez, ou quando se retorna de um outro aplicativo.
OnChange	Ocorre quando muda o conteúdo de um objeto.
OnClick	Ocorre quando o usuário dá um clique no botão esquerdo do mouse.
OnClose	Ocorre quando o objeto é fechado.
OnCreate	Ocorre quando o objeto é criado.
OnDblClick	Ocorre quando é feito um duplo clique com o botão esquerdo do mouse.
OnDeactivate	Ocorre quando se sai do objeto.
OnDestroy	Ocorre quando se elimina um objeto.
OnDragDrop	Ocorre quando um objeto é arrastado para outro objeto e solto.
OnDragOver	Ocorre quando um objeto é arrastado para cima de outro objeto.
OnDropDown	Ocorre quando se abre um objeto ComboBox ou ListBox.
OnEnter	Ocorre quando o objeto recebe o foco.
OnException	Ocorre quando ocorre um erro de execução na aplicação.
OnExit	Ocorre quando o objeto perde o foco.
OnHelp	Ocorre quando é solicitada a abertura de um arquivo de ajuda.
OnHide	Ocorre quando o objeto passa a ser oculto.
OnKeyDown	Ocorre quando o usuário pressiona uma tecla, incluindo Shift, Alt e Insert.
OnKeyPress	Ocorre quando o usuário pressiona uma tecla ASCII.
OnKeyUp	Ocorre quando o usuário solta uma tecla.
OnMinimize	Ocorre quando se minimiza uma janela.
OnMouseDown	Ocorre quando o usuário clica em um botão do mouse e o cursor é posicionado sobre a área clicada.
OnMouseMove	Ocorre quando o usuário move o cursor dentro da área selecionada.
OnMouseUp	Ocorre quando o usuário solta um botão do mouse.
OnPopup	Ocorre quando se ativa um menu popup com o botão direito do mouse.
OnResize	Ocorre quando se muda o tamanho do objeto.
OnRestore	Ocorre quando se restaura uma janela que foi minimizada.
OnRun	Ocorre quando uma aplicação inicia sua execução.
OnShow	Ocorre antes que o objeto se torne visível.
OnTimer	Ocorre em intervalos periódicos de tempo.

## 6 Caixas de Diálogo Predefinidas

O Delphi oferece vários comandos para a exibição de caixas de diálogo comuns, como caixas de mensagem (Message Boxes) e caixas de entrada (Input Boxes). Esses comandos permitem que aplicativos com recursos simples de entrada e saída sejam criados rapidamente.

### 6.1 Usando o Comando MessageDlg

O comando MessageDlg mostra uma caixa de mensagem que pode conter símbolos especiais, botões adicionais e outros elementos. O comando tem vários parâmetros que devem ser especificados. Veja a sintaxe do comando MessageDlg:

```
MessageDlg(<Mensagem>,<Tipo da Caixa>,<Botões>,<Número de ajuda>);
```

Os diálogos MessageDLG podem ser adicionados com os seguintes comandos:

---

#### Código 5: Componentes Message Dialog

---

```
1 //Caixa de confirmação
2
3 ShowMessage('Este comando não pode ser usado no momento.');
```

---

```
4
5 //Information
6
7 MessageDlg ('Mensagem de informação;', mtInformation, mbOKCancel, 0);
8
9 //Warning
10
11 MessageDlg ('Mensagem de aviso', mtWarning, mbOKCancel, 0);
12
13 //Error
14
15 MessageDlg ('Mensagem de erro', mtError, mbOKCancel, 0);
16
17 //Cofirm
18
19 MessageDlg ('Mensagem de confirmação', mtConfirmation, mbOKCancel, 0);
20
21 //Pesonalizado
22
```

---

```
23 MessageDlg ('Outra mensagem', mtCustom, mbOKCancel, 0);
```

---

<Botões> é usado para definir o conjunto de botões que será exibido na parte de baixo da caixa de mensagem. Há alguns conjuntos de botões predefinidos. Um deles foi usado nos exemplos anteriores: `mbOKCancel`, que mostra os botões OK e Cancel. Os conjuntos predefinidos são resumidos a seguir:

`mbYesNoCancel` → Mostra os botões Yes, No e Cancel.

`mbAbortRetryIgnore` → Mostra os botões Abort, Retry e Ignore.

`mbAbortIgnore` → Mostra os botões Abort e Ignore.

`mbOKCancel` → Mostra os botões OK e Cancel.

`MessageDlg` retorna um valor do botão selecionado pelo usuário. Estes são os possíveis valores de retorno:

`mrOk mrRetry mrYes mrNone mrNoToAll mrAll mrCancel mrIgnore mrNo mrAbort mrYesToAll`

## 6.2 Usando a Função Inputbox

Essa função mostra uma caixa simples com um campo para a entrada (um componente `Edit`) de dados e os botões OK e Cancel. `InputBox` retorna uma `String` com o texto digitado pelo usuário. A função `InputBox` recebe três parâmetros:

`InputBox (<Título da Caixa>, <Texto do prompt>, <Texto padrão>);`

<Título da Caixa> → define o texto que é exibido na barra de título da caixa.

<Texto do prompt> → é o texto exibido na parte interna da caixa.

<Texto Padrão> → é o texto padrão exibido dentro do campo de entrada.

Este texto aparece inicialmente selecionado. Para não mostrar um texto padrão use uma string vazia (`''`). Veja um exemplo a seguir:

```
InputBox ('Escolha de país', 'Digite o nome do país:', 'Brasil');
```

## 7 Conversão de variáveis

No Delphi é possível converter variáveis de um tipo em outro, isso só não é possível quando um tipo maior tenta ser convertido para um tipo menos.

Para converter String para um outro tipo:

```
SrtTo[Int, Float, Time, Date].
```

Para converter outros tipos para string:

```
IntToStr, FloatToStr, TimeToStr, DateToStr.
```

## 8 Formatação de números

função `FloatToStr`, transforma um número em texto, mas não padroniza a sua apresentação. Caso seja necessário formatar um dado a ser exibido, deve-se usar a função:

```
FormatFloat ( Formato , Expressão)
```

**Formato** → a maneira como deverá ser mostrada a expressão.

**Expressão** → expressão numérica ou string a ser formatada.

Em "formato" o número 0 será mostrado ou trocado pelo caractere em sua posição, já o símbolo (#) não será mostrado. Pode-se inserir símbolos na função `Format`, como no exemplo: \$, % ou E.

Formato	5 positivo	5 negativo	5 decimal
0	5	-5	1
0,00	5,00	-5,00	0,50
###0	5	-5	1
###0,0	5,0	-5,0	0,5
\$###0;(\$###0)	\$5	(\$5)	\$1
\$###0,00;(\$###0,00)	\$5,00	(\$5,00)	\$0,50
0%	500%	-500%	50%
0,00E+00	5,00E+00	-5,00E+00	5,00E-1

## 9 Formatação de data e hora

Para formatar data e hora usamos a função:

```
FormatDateTime (formato , data);
```

Formato	Exibido
d/m/yy	10/7/96
dd-mm-yyyy	01-Jun-1996
dd-ddd	02-dom
hh:mm AM/PM	08:50 AM
h:mm:ss a/p	8:50:20 a
d/m/yy h:mm	03/12/95 9:30

## 10 Exercícios

1. Crie um programa que tenha 2 campos para o usuário digitar nome e sobre nome de uma pessoa. O programa deverá concatenar e exibir o nome completo em uma label.
2. Crie um programa com um botão que ao ser clicado mostre uma mensagem "Hello World!";
3. Crie um programa qualquer que ao ser fechado, faça a confirmação, se o usuário não confirmar a janela o programa não deverá ser fechado.
4. Crie um programa que simule um relógio.
5. Crie um programa que simule uma calculadora com as 4 operações básicas, as operações deverão ser escolhidas por um radio group.
6. Crie um programa que formate em reais um valor informado por um inputText.