Módulo 5

Delphi - Ambiente (IDE) e Criação de Programas I

INTRODUÇÃO

Esta lição visa familiarizá-lo com o Delphi 7.0 Enterprise. Durante a lição abordaremos os principais componentes do Delphi e suas principais propriedades.

CARACTERÍSTICAS DO DELPHI

* Gera um executável verdadeiro, independente de run-time.
* Utiliza a 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 componentes 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 unidade 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.
~DF	Backup de DFM	Backup de um arquivo DFM.
~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.
DCU	Unit Compilada	Arquivo PAS compilado.
DFM	Arquivo de formulário gráfico	Arquivo binário contendo as propriedades e componentes de um formulário.
DPR	Arquivo de Projeto	Escrito em Object Pascal contendo os componentes de um projeto e permite uso de código de inicialização do projeto.

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

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.

File	
New	Abre um submenu com novos itens que podem ser adicionados ao projeto.
Open	Abrir projetos, pode abrir também Units, Forms e texto no editor de código.
Reopen	Abre um submenu com os 5 últimos projetos usados e as 10 últimas units utilizadas.
Save (Ctrl+S)	Salva o arquivo aberto no editor de código.
Save Project As	Salva o projeto com outro nome ou em outro local.
Save All (Shift+Ctrl+S)	Salva as alterações realizadas em todas as units do projeto. (Uso recomendado).
Use Unit (Alt+F11)	Faz com que a Unit atual possa usar outra Unit do projeto.
View	
Project Manager (Ctrl+Alt+F11)	Mostra o gerenciador de projeto.
Object Inspector (F11)	Mostra o Object Inspector.
Toggle Form/Unit (F12)	Alterna entre o Form e a Unit.
Units (Ctrl+F12)	Mostra o código fonte de uma Unit ou do Projeto a partir de uma lista.
Forms (Shift+F12)	Seleciona um Form a partir de uma lista.
Project	
Add to Project (Shift+F12)	Adiciona uma Unit em disco ao projeto.
Remove from Project	Remove uma Unit do projeto.
View Source	Mostra o código do projeto.
Compile (Ctrl+F9)	Compila o projeto.
Options (Shift+Ctrl+F11)	Opções do projeto, como ícone do executável, nome da aplicação e opções de compilação.
Run	
Run (F9)	Compila e executa o projeto.
Program Reset (Ctrl+F2)	Pára a execução do programa.

PROGRAMAÇÃO EM WINDOWS: JANELAS E EVENTOS

Existem muitas diferenças entre a programação em DOS e a programação em Windows. Vejamos a principal: Quando programamos em DOS, nosso programa é responsável pelo fluxo de processamento. Temos que definir claramente não só que instruções, mas também em que ordem devem ser executadas. Em Windows não é bem assim. Nosso programa não controla o fluxo de processamento, ele responde e trata eventos que ocorrem no sistema. Existem muitos eventos que podem ocorrer, sendo que os principais são aqueles gerados pelo usuário através do mouse e do teclado. A coisa acontece mais ou menos assim: O usuário clica o mouse e o Windows verifica que aplicação estava debaixo do mouse no momento em que foi clicado. Em seguida ele manda uma mensagem para a aplicação informando que ocorreu um clique e as coordenadas do cursor do mouse na tela no momento do clique. A aplicação então responde à mensagem executando uma função de acordo com a posição do mouse na tela. É claro que o Delphi toma conta do serviço mais pesado e facilita muito as coisas para o programador. Detalhes como as coordenadas da tela em que ocorreu o clique, embora estejam disponíveis, dificilmente são necessários nos programas.

PROGRAMAÇÃO ORIENTADA A OBJETO (POO)

Embora não seja objetivo desta lição ensinar POO, uma breve introdução é necessária, já que o Delphi é essencialmente orientado a objeto.

De maneira prática, podemos pensar no objeto sendo uma estrutura que agrupa dados e funções para manipular estes dados. Como as funções são sempre “íntimas” dos dados, o sistema todo funciona de maneira mais segura e confiável. Além disso, a POO utiliza conceitos como encapsulamento e herança que facilitam muito a programação e a manutenção dos programas.

Neste ponto é oportuno citar que os dados de um objeto costumam ser chamados de variáveis de instância e as funções de métodos. As variáveis de instância definem as propriedades (às vezes chamadas de atributos) do objeto e os métodos definem seu comportamento.

Encapsulamento

Como as variáveis e métodos estão na mesma estrutura, pode-se pensar em variáveis e métodos privados, ou seja, dados e funções que só podem ser manipulados pelas funções que estão dentro da estrutura. Desta maneira é possível formar uma camada protetora nos dados e evitar atribuições desastradas que comprometeriam o funcionamento

do programa. Os defensores mais ortodoxos da POO dizem que todos os dados de um objeto deveriam ser privados e o número de funções públicas deve ser o menor possível, mas isso nem sempre é viável ou prático. O Delphi implementa este conceito e oferece dados/funções publicas (public) e privadas (private). Outra conseqüência do encapsulamento é que os objetos podem ser "caixas pretas". Não é necessário (teoricamente) conhecer detalhes de funcionamento de um objeto para usá-lo, basta enviar as mensagens apropriadas que ele responde com a ação desejada.

Classes

A classe representa um tipo ou categoria de objetos, o modelo a partir do qual um objeto pode ser construído. É a estrutura propriamente dita, que define os dados e métodos daquela classe de objetos. O objeto em si é uma instância da classe. Na programação estruturada podemos fazer uma analogia com os tipos e variáveis, onde a classe eqüivale ao tipo e o objeto à variável desse tipo.

Herança

É a capacidade que uma classe de objetos tem de herdar variáveis e métodos de outra classe. Esta capacidade permite que o código já escrito seja reutilizado de maneira muito mais eficiente e simples do que na programação estruturada. Um programa orientado a objeto costuma implementar verdadeiras árvores genealógicas de classes, com vários níveis de herança.

Para programar no nível do designer (veja adiante o que significa) não é necessário um conhecimento profundo de POO. Mas é preciso conhecer pelo menos a sintaxe. Todas as aplicações para Windows precisam de pelo menos uma janela, que no Delphi é chamada de Form. Cada form (assim como todos os objetos visuais) tem um objeto associado a ele e sua representação visual, que vemos na tela. Todos os componentes que incluímos no form passam a fazer parte do objeto que define o form. Exemplo: Se colocarmos um botão no form, a classe deste form será modificada para incluir este botão. Os eventos e métodos deste form, também estão na classe. Assim, supondo que o form se chame Form1 (nome default), para, por exemplo, desativar o botão que incluímos (de nome Buttom1) faríamos:

```
Form1.Buttom1.Enabled := false;
```

Note como o Buttom1 faz parte da estrutura que define o form. Mas se quisermos ativar o método RePaint (Repintar) do form faríamos:

```
Form1.Repaint;
```

Veja que Repaint, não é uma variável, tecnicamente é uma procedure, mas fazemos referência a ela como parte da estrutura Form1. Pode parecer confuso no início, mas facilita muito a programação.

O Delphi oferece dois níveis de programação distintos. Existe o nível que é chamado de *designer*, que se utiliza dos recursos de programação visual e aproveita componentes prontos, e o nível do *component writer*, que escreve os componentes para o designer utilizar nas aplicações. Podemos dizer que o *component writer* programa em um nível mais baixo (mais difícil) e o *designer* em um nível mais alto (mais fácil e compreensível). Neste curso, estamos estudando a programação no nível do *designer*.

Quando ativamos o Delphi, a tela inicial é parecida com a figura 5.1. Na janela superior, temos a Barra de Menu Principal do Delphi, à esquerda a SpeedBar (Barra de Ferramentas Rápida), com as opções mais comuns e à direita a Paleta de Componentes. Estes componentes são a base da programação visual e é onde o designer vai buscar recursos para sua aplicação. Veja com mais detalhes na figura 5.2.

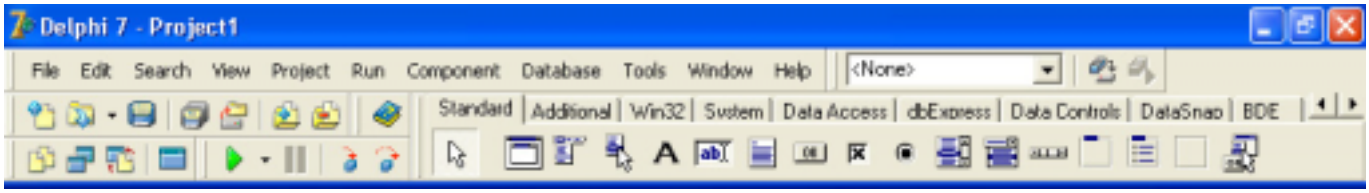


Figura 5.2 – Barra de Menu, SpeedBar e Paleta de Componentes.

O AMBIENTE DO DELPHI

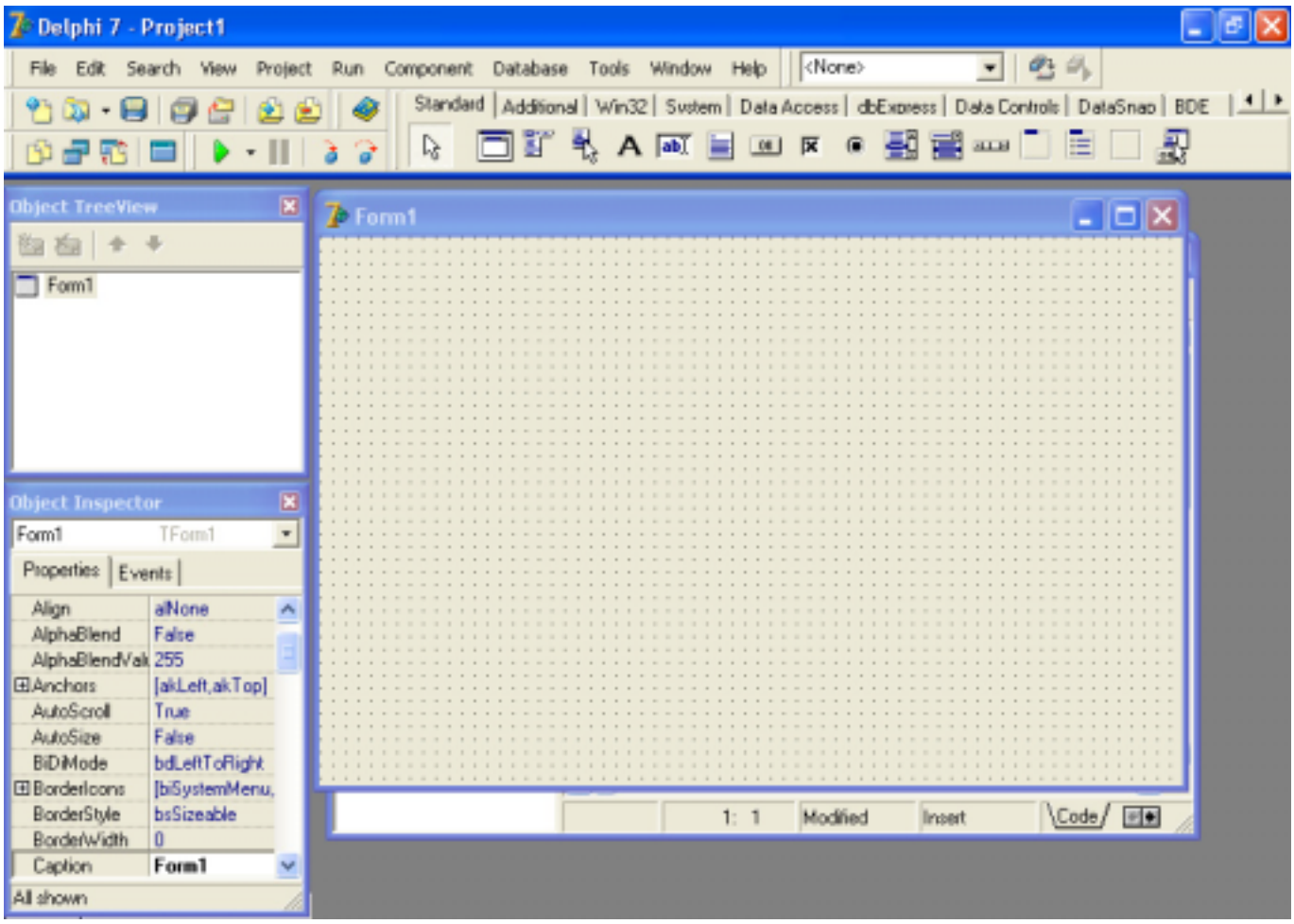


Figura 5.1 - Quando iniciar o Delphi pela primeira vez você verá a tela acima, que representa a IDE do Delphi.

Abaixo da SpeedBar, está a janela do Object TreeView, que permite visualizar e acessar todos os componentes dentro do formulário que você está utilizando no momento. Observe na figura 5.3.

Abaixo do Object Treview encontra-se o Object Inspector, que permite visualizar e modificar as propriedades e eventos de todos os componentes. É também largamente utilizado pelo designer. Veja na figura 5.4.

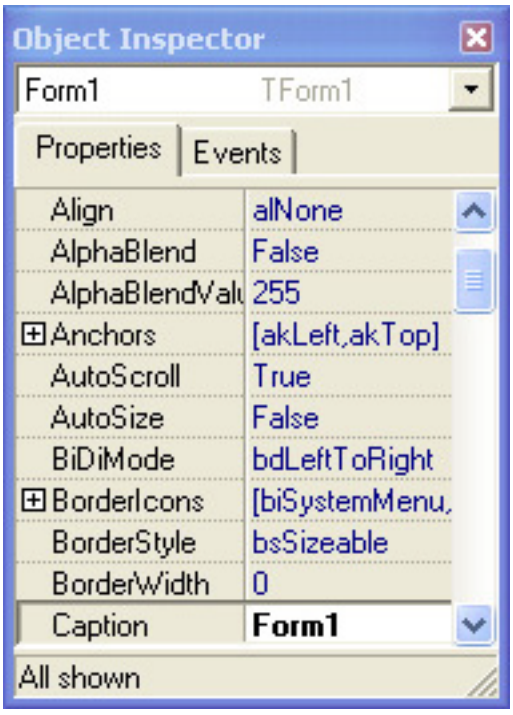


Figura 5.4 – Object Inspector

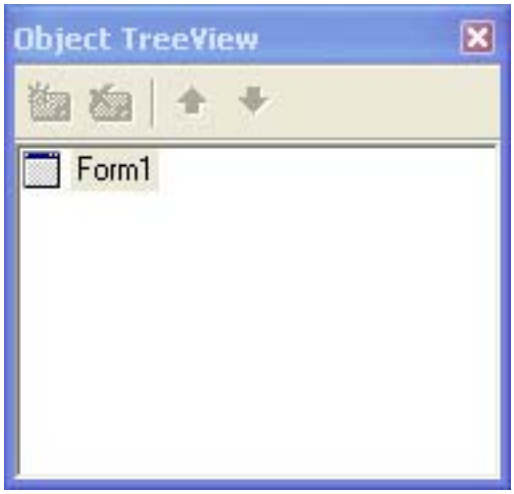


Figura 5.3 – Object TreeView

Abaixo da Paleta de Componentes ficam a janela de código-fonte e as janelas que estão sendo construídas.

As janelas que estão sendo construídas são chamadas de Forms (formulários) e cada form tem uma janela de código-fonte pertencente a ela, que são chamadas de Units.

Veja as figuras 5.5 e 5.6.

Figura 5.5 – Form (Formulário)

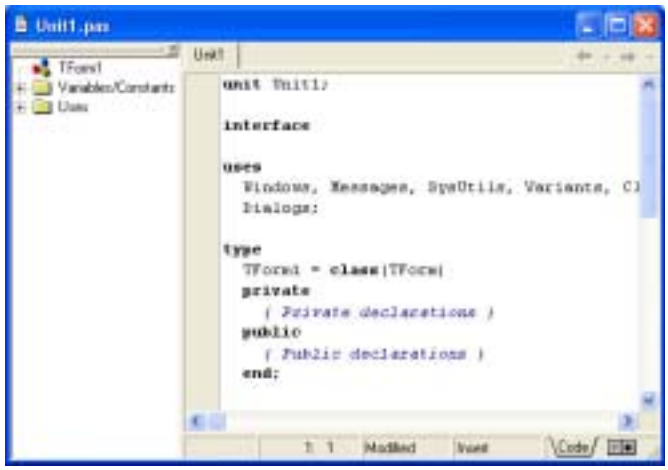


Figura 5.6 – Unit (Código-Fonte)

PALETAS DE COMPONENTES

A seguir visualizaremos as paletas de componentes que estudaremos durante este curso. O Delphi 7 possui outras paletas padrões de componentes, mas não é nosso objetivo estudá-las agora. Além disso você pode encontrar milhares de componentes na Internet.

Figura 5.7 – Paleta Standard - Uma das paletas mais utilizadas pelo programador

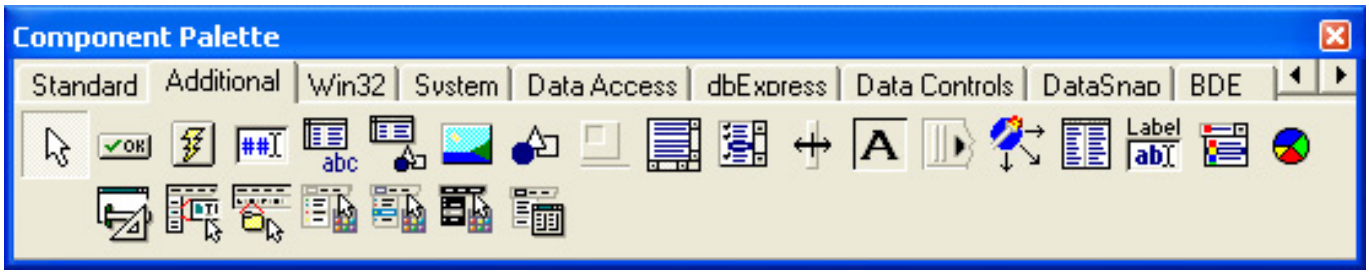


Figura 5.8 – Paleta Additional - Componentes adicionais, também muito utilizados

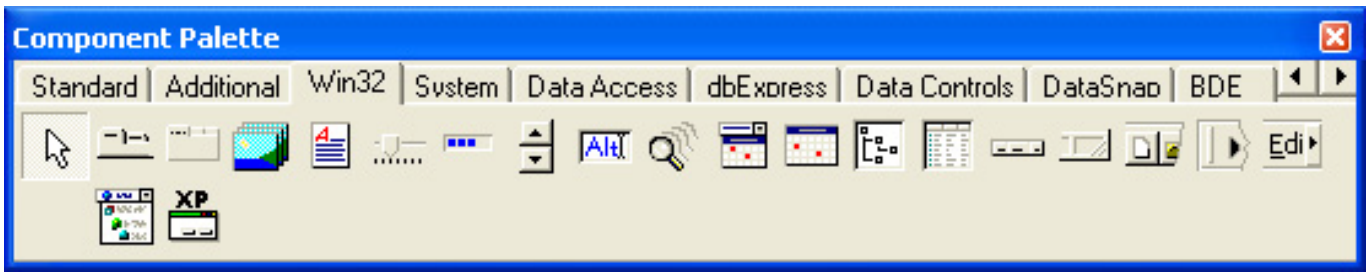


Figura 5.9 – Paleta Win32 - Componentes que utilizam recursos visuais do Windows

Figura 5.10 – Paleta System - Componentes que utilizam outros recursos do Windows

Figura 5.11 – Paleta Data Access - Componentes para acessar dados

Figura 5.12 – Paleta Data Controls - Componentes para controlar visualmente os dados dentro do formulário.

Figura 5.13 – Paleta BDE - Componentes para acessar dados via BDE.

Figura 5.14 – Paleta Dialogs - Componentes que acessam as janelas de diálogo do Windows.

Figura 5.15 – Paleta Win 3.1 - Componentes que deixam o aplicativo com o visual do Windows 3.1.

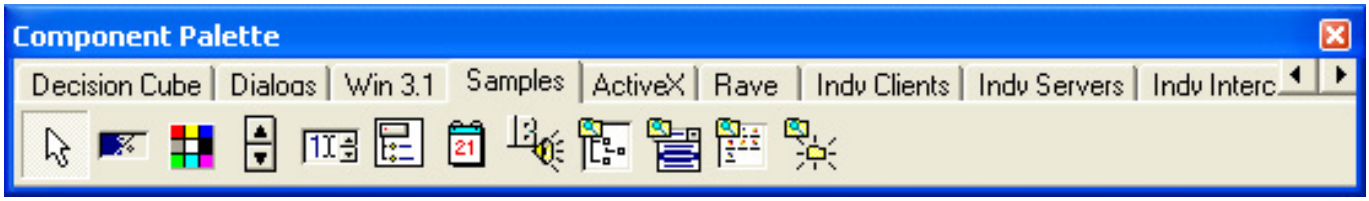


Figura 5.16 – Paleta Samples - Calendário, Paleta de Cores e outros.

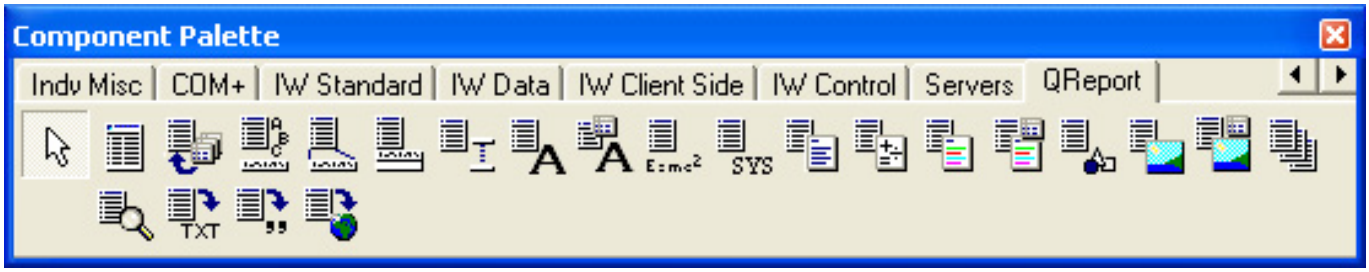


Figura 5.17 – Paleta QReport - Componentes para criar relatórios visuais.

PRIMEIRO CONTATO

Para iniciarmos poderíamos criar a versão Delphi do famoso “Alô Mundo”, mas vamos partir para alguma coisa um pouco mais interessante e aproveitar para apresentar uma propriedade que pode ser útil no futuro. Como os programas em Windows são orientados a eventos é comum desativar opções de menus e botões, até que o usuário ative as opções que o sistema precisa para inicializar. Neste primeiro programa, vamos criar uma espécie de gangorra eletrônica. Mas antes de começar, vejamos o que está acontecendo no Delphi.

Se a tela está parecida com a figura 5.1, então não há nenhum projeto selecionado e o Delphi tomou a liberdade de criar um novo projeto para você, chamando-o de Project1. Um projeto é a coleção de arquivos necessários para compilar o sistema. Como toda aplicação precisa de pelo menos um form (Form1), ele também foi criado. Finalmente, todo form tem uma Unit correspondente que é mostrada no Editor de Código (Unit1).
Dica: Para intercalar entre o Form e a Unit use a tecla F12.

Não há problema em utilizar esse projeto inicial que é oferecido pelo Delphi, mas é uma boa idéia renomear e salvar o projeto o quanto antes. Se você não escolher outro diretório, o projeto será salvo no diretório do Delphi e não é aconselhável salvar todos os seus projetos no mesmo diretório. Portanto vamos começar este programa criando um novo diretório para ele. Após isso, vamos seguir os seguintes passos:

1. Selecione a página Standard na paleta de componentes e clique no componente Button. A seguir clique no form. Um botão deve aparecer no form. Coloque mais dois botões.

2. Alinhe na horizontal os dois primeiros e coloque o terceiro logo abaixo dos dois. Se quiser um alinhamento exato, use a opção Edit/Align... do menu. Para marcar mais de um componente, deixe o Shift pressionado enquanto clica os componentes. Brinque um pouco com o tamanho dos botões e do form.

3. Clique no primeiro botão para selecioná-lo, a seguir procure no Object Inspector a propriedade Caption. Mude a string para “ON”. Para o segundo botão a string é “OFF” e para o terceiro “Close”. Para o segundo botão, mude a propriedade Enabled para False.

4. Neste ponto é uma boa idéia renomear e salvar o projeto. Escolha a opção File/Save Project As...Selecione o diretório criado para o projeto e use o nome Main.pas para a unit1 e Gangorra para o projeto.

5. Clique na área pontilhada do form e em seguida ache a propriedade Caption no Object Inspector. Mude a string para “Gangorra”. Aproveite e mude a propriedade Name para “FMain”.

6. Dê um clique duplo sobre o primeiro botão. A janela de código será ativada já com a função correspondente ao evento OnClick criada e posicionada sob o cursor. Digite o seguinte código:

```
Button1.Caption := ‘OFF’;
Button1.Enabled := False;
Button2.Caption := ‘ON’;
Button2.Enabled := True;
```

7. Selecione a janela do form e faça o mesmo com o segundo botão, mas o código fica invertido:

```
Button2.Caption := ‘OFF’;
Button2.Enabled := False;
Button1.Caption := ‘ON’;
Button1.Enabled := True;
```

8. Finalmente para o terceiro botão o código é simplesmente:

```
Close;
```

9. Já podemos executar o programa pressionando F9.

O que está acontecendo: O evento OnClick ocorre sempre que clicamos o mouse sobre um componente. O Delphi se encarrega de determinar qual é o componente que deve responder ao evento e direciona o evento para ele. Neste caso, usamos este evento para ativar/desativar botões manipulando a propriedade Enabled, que determina se o componente está ativo ou não. Quando o componente está inativo, ele não responde aos eventos. Também usamos a propriedade Caption para mudar a mensagem que aparece nos botões. Finalmente usamos o método Close do form para fechar a aplicação.

PROPRIEDADES COMUNS

Propriedade	Descrição
Align	Determina o alinhamento do componente.
Canvas	Superfície de desenho, do tipo TCanvas, onde pode se desenhar a imagem do componente.
Caption	Legenda do componente (& indica tecla de atalho para alguns componentes).
Color	Cor do componente.
ComponentCount	O número de componentes possuídos.
Components	Matriz de componentes possuídos.
Ctl3D	Define a aparência 3D do componente.
Enabled	Define se o componente está ativo, se pode ser usado.
Font	Fonte utilizada no componente.
Height	Altura.
HelpContext	Número utilizado para chamar o Help on-line.
Hint	String utilizada em dicas instantâneas.
Left	Posição esquerda.
Name	Nome do componente. O Delphi nomeia automaticamente todos os componentes que são incluídos no form (inclusive o próprio form). Usa o nome da classe do componente mais um número seqüencial. O nome atribuído pelo Delphi pode ser mantido, mas é aconselhável renomear os componentes que serão referidos no programa. Por exemplo, no programa da Gangorra, Button1 e Button2 deveriam ser renomeados, já que é feita referência a eles no código fonte, já para o Button3 não há necessidade por que não há referência a ele. Quando você renomeia um componente, o Delphi atualiza automaticamente todo o código gerado pelo Delphi, o que inclui o cabeçalho da Unit, os eventos do componente e as propriedades de outros componentes que fazem referência ao componente renomeado, mas não atualiza o código gerado por você. Exemplo: se renomearmos agora o Button1, o Delphi atualizará o cabeçalho da unit, o nome dos eventos de Button1, mas você terá que atualizar as referências que você fez ao Button1 com o novo nome. Aliás, esta é uma regra geral no Delphi: ele nunca modifica automaticamente o código gerado pelo programador, mesmo que esteja em comentário.
PopupMenu	Menu de contexto do componente.
ShowHint	Define se o Hint será mostrado.
TabOrder	A ordem de tabulação do componente, usada quando o usuário tecla TAB.
TabStop	Indica se o componente será selecionado quando o usuário teclar TAB.
Tag	Propriedade não utilizada pelo Delphi, que pode ser usada como propriedade personalizada.
Top	Posição superior.
Visible	Define se o componente está visível.
Width	Largura.

EVENTOS

Os Eventos acontecem em resposta a uma ação do usuário ou do próprio sistema, ao programar um método de evento, devemos levar em consideração que este só será executado quando o evento acontecer. Uma das tarefas mais importantes na programação baseada em eventos é determinar quais eventos serão usados e qual a ordem desses eventos, por exemplo, quando o usuário clicar em um botão, qual evento acontecerá primeiro, OnEnter, OnMouseDown ou OnClick?

Os eventos podem ser compartilhados entre componentes, dessa Forma, você pode ter um botão na barra de ferramentas que faz a mesma coisa que uma opção de menu. Para isso, basta escolher o evento na lista em vez de clicar duas vezes no Object Inspector.

Podemos também mudar os métodos de evento em código, pois os eventos também são propriedades e podem ser usados como tal. Você pode atribuir um evento de outro componente ou diretamente o nome do método, como

mostrado abaixo.

```
Button1.OnClick := Edit1.OnExit;
Button2.OnClick := Edit2Click;
```

EVENTOS COMUNS

Evento	Descrição
OnChange	O conteúdo do componente é alterado.
OnClick	O componente é acionado.
OnDblClick	Duplo-clique no componente.
OnEnter	O componente recebe o foco.
OnExit	O componente perde o foco.
OnKeyDown	Tecla pressionada.
OnKeyPress	Uma tecla é pressionada e solta.
OnKeyUp	Tecla é solta.

MÉTODOS

Os métodos realizam ações definidas pelo componente, veja os exemplos abaixo e atente para os parâmetros passados. Note que podemos chamar os métodos de evento como qualquer outro método e que os métodos de evento pertencem ao Form, não aos componentes.

```
Edit1.Clear;
Form2.Show;
Close;
ScaleBy(110, 100);
Button1.OnClick(Sender);
Button1Click(Self);
Form2.Button1Click(Sender);
```

MÉTODOS COMUNS

Método	Descrição
Create	Cria um novo Objeto de uma Classe.
Free	Destrói um Objeto e libera a memória ocupada por ele.
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.
ScrollBy	Move o componente.
ScaleBy	Gradua o componente em determina escala.
SetBounds	Muda a posição e o tamanho do componente.

JANELAS

Todo aplicativo Windows é composto por janelas, que são o elemento básico no desenvolvimento Delphi, sobre o qual um aplicativo é construído. O tipo TForm é usado no Delphi como classe base para todas as janelas, veja abaixo algumas propriedades, eventos e métodos dessa classe.

Propriedade	Descrição
Active	Indica se o Form está ativo.
ActiveControl	Determina o controle que receberá o foco por default.
AutoScroll	Adiciona barras de rolagem automaticamente, quando necessário.
BorderIcons	Define quais ícones de controle serão visíveis, quais botões vão aparecer na barra de título.
BorderStyle	Estilo da borda do Form.
FormStyle	Tipo de Form, normal, MDI pai, MDI filho ou sempre visível.
Icon	Ícone do Form.
Menu	Indica qual o menu do Form.
Position	Permite controlar a posição e tamanho do Form na exibição.
WindowMenu	Automatiza o item de menu Window (MDI).
WindowState	Estado do Form, maximizada, minimizada ou normal.
Evento	**Descrição**
OnCreate	Quando o Form é instanciado.
OnDestroy	Quando o Form é liberado da memória.
OnShow	Exatamente antes de mostrar o Form.
OnCloseQuery	É chamada para validar se o Form pode ser fechado.
OnClose	Quando o Form é fechado.
OnActivate	Quando o Form recebe o foco.
OnDeactivate	Quando o Form perde o foco.
OnResize	Quando o Form muda de tamanho.
Método	**Descrição**
Cascade	Organiza as Forms filhos em cascata (MDI).
Tile	Organiza as Forms filhos lado a lado (MDI).
ArrangeIcons	Organiza os ícones dos Forms Filhos minimizados (MDI).
ShowModal	Ativa o Form modal, que o usuário tem que fechar para poder continuar a usar a aplicação.
Show	Mostra o Form.
Close	Fecha o Form.
Previous	Ativa o Form anterior (MDI).
Next	Ativa a próximo Form (MDI).

COMPONENTES PADRÕES

TButton

Componente botão padrão do Windows, utilizado para executar ações.

Propriedade	Descrição
Cancel	Dispara o evento OnClick do botão quando a tecla ESC é pressionada em qualquer controle.
Default	Dispara o evento OnClick do botão quando a tecla ENTER é pressionada em qualquer controle.
ModalResult	Associa o botão a opção de fechamento de um Form modal.
Método	Descrição.
Click	Ativa o evento OnClick do botão.

TBitBtn

Botão especializado, com Bitmap.

Propriedade	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.
Kind	Seleciona um tipo padrão para o botão, mudando várias propriedades, como Glyph e ModalResult.

TSpeedButton
Botão com Bitmap, normalmente utilizado em barras de ferramentas.

Propriedade	Descrição
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 ficar não pressionados.
Flat	Define se a borda do botão deve aparecer apenas quando ele for apontado.

TLabel
Utilizado para exibir rótulos

Propriedade	Descrição
Alignment	Alinhamento do texto no componente.
AutoSize	Define se o tamanho do componente será automaticamente ajustado ao tamanho do Caption.
WordWrap	Retorno automático de linha.
Transparent	Define se o componente será transparente.
FocusControl	Componente que receberá o foco quando a tecla de atalho do Caption (&) for pressionada.
ShowAccelChar	Indica se o caractere & será usado para definir tecla de atalho.

TEdit
Utilizado para entrada de texto em uma única linha.

Propriedade	Descrição
Text	Texto do componente.
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	Caractere utilizado para esconder o texto digitado (Senhas).
ReadOnly	Define se será permitido alterar o texto.
Método	**Descrição**
Clear	Limpa o conteúdo do componente.
ClearSelection	Limpa o texto selecionado no componente.

TMaskEdit
Permite entrada de dados texto em uma linha, utilizando uma máscara de edição. Possui todas as propriedades do componente TEdit.

Propriedade	Descrição
EditMask	Máscara de edição.

Máscaras
Uma máscara é composta por três partes, a primeira parte é a máscara propriamente dita, a segunda parte indica se os caracteres literais serão salvos e a terceira parte indica qual o caractere utilizado para representar os espaços a serem digitados no texto.

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

Caractere	Descrição
!	Espaços em branco não serão considerados no texto
>	Todos os caracteres seguintes serão maiúsculos até que apareça o caractere <
<	Todos os caracteres seguintes serão minúsculos até que apareça o caractere >
\	Indica um caractere literal
l	Somente caractere alfabético
L	Obrigatoriamente um caractere alfabético
a	Somente caractere alfanumérico
A	Obrigatoriamente caractere alfanumérico
9	Somente caractere numérico
0	Obrigatoriamente caractere numérico
c	Permite um caractere
C	Obrigatoriamente um caractere
#	Permite um caractere numérico ou sinal de mais ou de menos, mas não os requer.
:	Separador de horas, minutos e segundos
/	Separador de dias, meses e anos

TMemo

Permite entrada de dados texto em múltiplas linhas. Contém propriedades e métodos do TEdit.

Propriedade	Descrição
Lines	Propriedade do tipo TStrings que armazena as linhas de texto do componente.
WantReturns	Define se a tecla ENTER será tratada como quebra de linha.
WantTabs	Define se a tecla TAB será tratada como espaço de tabulação.
ScrollBar	Define as barras de rolagem.

TStrings

Muitos componentes, como o TMemo, 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 lista.

Propriedade	Descrição
Count	Número de strings.
Text	Conteúdo do memo na Forma de uma única string.
Método	**Descrição**
Add	Adiciona uma nova string no final da lista.
Insert	Insere uma nova string numa posição especificada.
Move	Move uma string de um lugar para outro.
Delete	Apaga uma string.
Clear	Apaga toda a lista.
IndexOf	Retorna o índice do item e - 1 caso não encontre.
LoadFromFile	Carrega texto de um arquivo.
SaveToFile	Salva texto para um arquivo.

TCheckBox

Propriedade	Descrição
AllowGrayed	Determina se o checkbox terá três possibilidades de estado.
Checked	Determina se o checkbox está marcado.
State	Estado atual do checkbox .

TRadioButton

Usado em grupo, pode ser utilizado para obter inFormações lógicas mutuamente exclusivas, mas é recomendado usar o RadioGroup em vez de RadioButtons.

TRadioGroup
Componente que agrupa e controla RadioButtons automaticamente.

Propriedade	Descrição
Columns	Número de colunas de RadioButtons.
Items	Lista de strings com os itens do RadioGroup, cada item da lista representa um RadioButton.
ItemIndex	Item selecionado, iniciando em 0.

TPanel
Componente Container utilizado para agrupar componentes em um panel.

Propriedade	Descrição
BevelInner	Estilo da moldura interna do panel.
BevelOuter	Estilo da moldura externa do panel.
BevelWidth	Largura das molduras.
BorderStyle	Estilo da Borda.
BorderWidth	Largura da borda, distância entre as molduras interna e externa.

TScrollBox
Container com barras de rolagem automáticas.

TGroupBox
Componente container com um título e borda 3D.

TBevel
Moldura ou linha com aparência 3D.

Propriedade	Descrição
Shape	Tipo de moldura a ser desenhada.
Style	Define alto ou baixo relevo para a linha.

TListBox
Utilizado para exibir opções em uma lista.

Propriedade	Descrição
Columns	Número de colunas de texto 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	Lista de strings com os itens da lista.
ItemIndex	Índice do item selecionado, começando em 0.
Selected	De acordo com o índice indica se um item em particular está selecionado.
SelCount	Indica quantos itens estão selecionado.
Sorted	Define se os itens aparecerão ordenados.

TComboBox
Caixa combinada com lista suspensa.

Propriedade	Descrição
Items	Lista de strings com os itens da lista.
DropDownCount	Número de itens visíveis da lista suspensa.
Style	Estilo do ComboBox, os principais estilos são csDropDown, csDropDownList, csSimple.

TImage
Componente usado para exibir figuras.

Propriedade	Descrição
Center	Determina de a figura será centralizada no componente.
Picture	Figura a exibida, pode ser BMP, ICO, WMF ou EMF.
Stretch	Define se o tamanho da figura deve ser ajustado ao do componente

TPicture
Classe usada para guardar ícones, Bitmaps, meta arquivos do Windows ou gráficos definidos pelo usuário.

Método	Descrição
LoadFromFile	Carrega figura de um arquivo.
SaveToFile	Salva figura para um arquivo.

TPageControl
Usado para criar controles com múltiplas páginas, que podem ser manipuladas, em tempo de projeto, através do menu de contexto. Cada página criada é um objeto do tipo TTabSheet.

Propriedade	Descrição
ActivePage	Página ativa.
MultiLine	Define múltiplas linhas de guias de páginas.
TabHeigth	Altura das guias.
TabWidth	Largura das guias
Evento	**Descrição**
OnChange	Após uma mudança de página.
OnChanging	Permite a validação de uma mudança de página.
Método	**Descrição**
FindNextPage	Retorna a próxima página.
SelectNextPage	Seleciona a próxima página.

TTabSheet
Página de um PageControl.

Propriedade	Descrição
PageIndex	Ordem da página.
TabVisible	Define se a aba da página é visível.

TShape
Gráfico de uma Forma geométrica.

Propriedade	Descrição
Brush	Preenchimento da figura, objeto do tipo TBrush.
Pen	Tipo da linha, objeto do tipo TPen.
Shape	Forma geométrica.

TTimer
Permite a execução de um evento a cada intervalo de tempo.

Propriedade	Descrição
Interval	Tempo em milisegundos quando o componente irá disparar o evento OnTimer.
Evento	**Descrição**
OnTimer	Chamado a cada ciclo de tempo determinado em Interval.

TStatusBar
Utilizado para criar barras de status para exibir inFormações.

Propriedade	Descrição
SimplePanel	Indica se haverá apenas um panel.
SimpleText	Texto exibido caso SimplePanel seja True.
SizeGrip	Define se a alça de redimensionamento padrão deve ser mostrada.
Panels	Propriedade do tipo TStatusPanels, com os painéis do StatusBar.

TStatusPanels
Lista de panels de um StatusBar.

Propriedade	Descrição
Count	Número de panels.
Items	Lista de panels, cada panel é um objeto do tipo TStatusPanel.
Método	**Descrição**
Add	Adiciona um novo panel à lista.

Caixas de Diálogo
Grupo de caixas de diálogo comuns do Windows.

Método	Descrição
Execute	Mostra a caixa de diálogo e retorna True caso o usuário clique em Ok.

TOpenDialog / TSaveDialog
Caixas de diálogo para abrir e salvar arquivos.

Propriedade	Descrição
FileName	Nome do arquivo.
DefaultExt	Extensão padrão para os arquivos.
Filter	Filtro, com os tipos de arquivos que serão abertos ou salvos.
FilterIndex	Índice do filtro default.
InitialDir	Pasta inicial.
Title	Título da janela.
Options	Define características gerais do diálogo.

TFontDialog
Caixa de diálogo de escolha de fonte.

Propriedade	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 Aplicar, antes da janela fechar.

MENUS

No Delphi os menus serão desenhados no Menu Designer, que pode ser acessado no menu de contexto de qualquer componente de menu.

TMainMenu
Menu principal de um Form.

Propriedade	Descrição
Items	Itens de menu, essa propriedade guarda todas as alterações feitas no Menu Designer.

TPopUpMenu

Menu de contexto de um componente. Cada componente tem uma propriedade PopUpMenu, que indica seu menu de contexto.

TMenuItem

Item de menu.

Propriedade	Descrição
Checked	Indica se o item está marcado ou não.
GroupIndex	Índice do grupo do item, semelhante ao SpeedButton.
RadioGroup	Indica se o item pode ser mutuamente exclusivo com outros itens do mesmo grupo.
ShortCut	Tecla de atalho do item.

Para concluir essa lição vamos criar mais um pequeno programa. Vamos criar uma pequena calculadora, como mostra a figura 5.18.

Para isso selecione a opção File/New/ Application. Clique em cima do novo formulário e altere as seguintes propriedades:

Propriedade	Valor
Caption	Calculadora
Name	FmCalculadora
Height	200
Width	150
BorderStyle	bsDialog

Agora insira 3 Edits. Altere as seguintes propriedades:

Componente	Propriedade	Valor
Edit1	Left	11
	Top	8
Edit2	Left	11
	Top	40
Edit3	Left	11
	Top	104

Agora insira 5 Buttons e altere as seguintes propriedades:

Componente	Propriedade	Valor
Button1	Left	11
	Top	72
	Width	25
	Font	Arial – 14 – Negrito
	Name	btnSoma
	Caption	+
Button2	Left	43
	Top	72
	Width	25
	Font	Arial – 14 – Negrito
	Name	btnSubtracao
	Caption	-
Button3	Left	75
	Top	72
	Width	25
	Font	Arial – 14 – Negrito
	Name	btnMultiplica
	Caption	x (xis minúsculo)
Button4	Left	104
	Top	72
	Width	25
	Font	Arial – 14 – Negrito
	Name	btnDivide
	Caption	/
Button5	Left	11
	Top	136
	Width	121
	Font	Arial – 14 – Negrito
	Name	btnLimpa
	Caption	Limpar
	Height	33

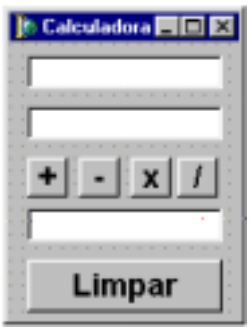


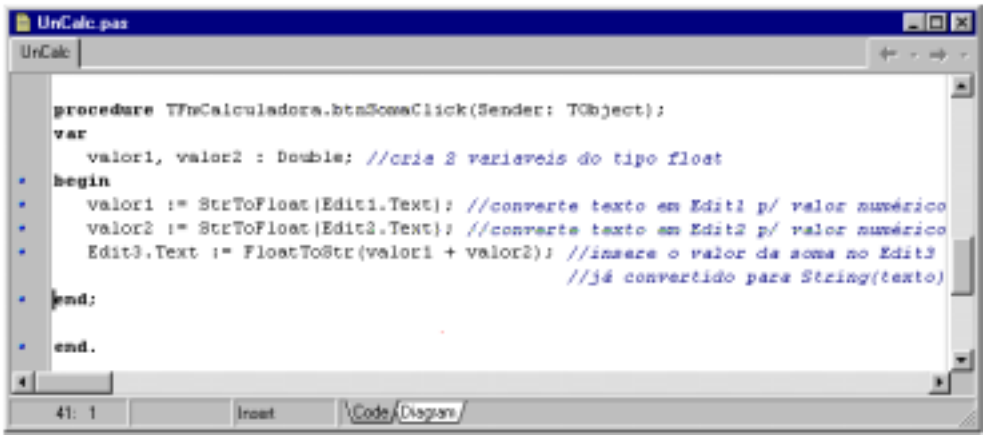
Fig. 5.18 – Proj. Calculadora

Além disso, selecione os 3 Edits e apague o conteúdo da propriedade Text.

Clique em File/Save All. Salve a Unit como UnCalc.pas e o projeto como Calculadora.dpr.

Passemos para a parte mais interessante: O código. Para que essa simples calculadora funcione precisamos utilizar algum código relacionado com o evento OnClick nos botões que inserimos. Vamos pôr em prática o que aprendemos na lição sobre Object Pascal.

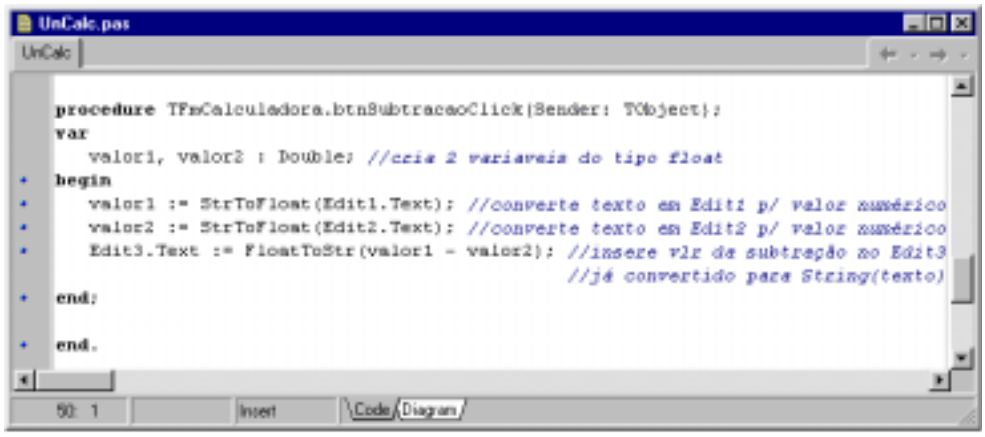
Dê um duplo clique no btnSoma para ativar o evento OnClick do mesmo. Insira o código conforme mostra a Figura 5.19.




```
Var
 Valor1, Valor2:Double;
Begin
 Valor1 := StrToFloat(Edit1.Text);
 Valor1 := StrToFloat(Edit2.Text);
 Edit3.Text := FloatToStr(Valor1 +
Valor2);
End;
```

Fig. 5.19 – Código do botão btnSoma

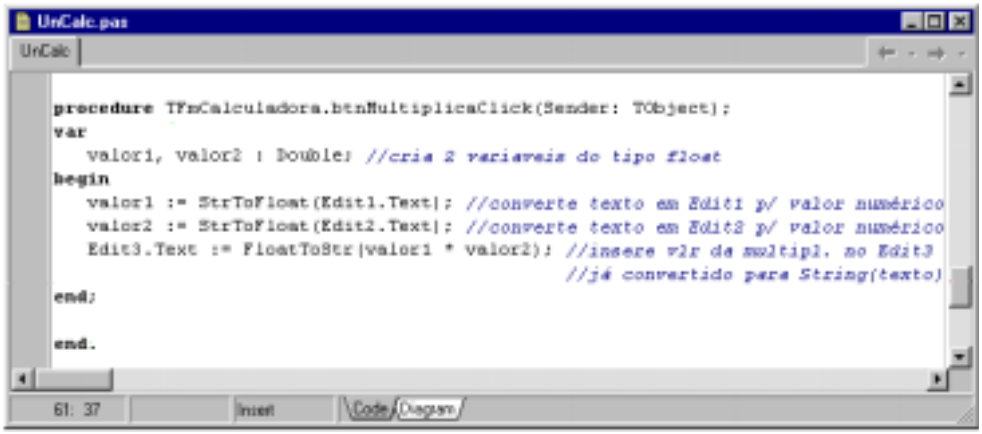
Dê um duplo clique no btnSubtracao para ativar o evento OnClick do mesmo. Insira o código conforme mostra a Figura 5.20.




```
Var
 Valor1, Valor2:Double;
Begin
 Valor1 := StrToFloat(Edit1.Text);
 Valor1 := StrToFloat(Edit2.Text);
 Edit3.Text := FloatToStr(Valor1 -
Valor2);
End;
```

Fig. 5.20 – Código do botão btnSubtração

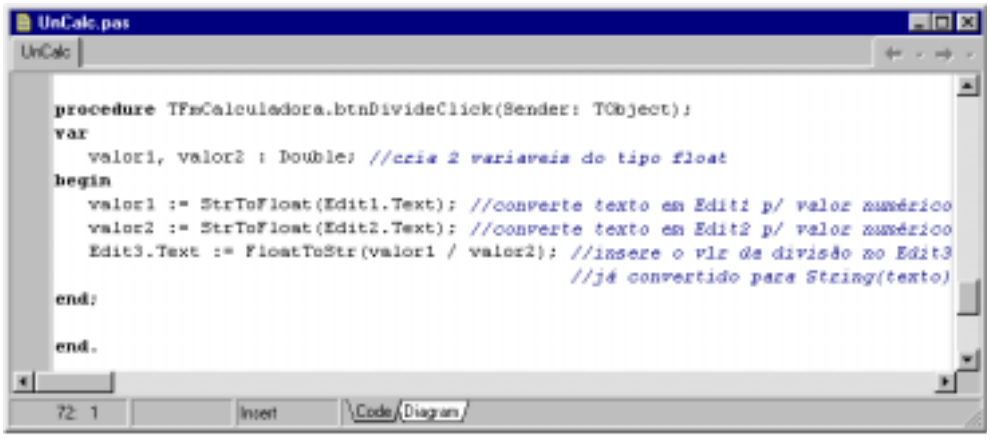
Dê um duplo clique no btnMultiplica para ativar o evento OnClick do mesmo. Insira o código conforme mostra a Figura 5.21.




```
Var
 Valor1, Valor2:Double;
Begin
 Valor1 := StrToFloat(Edit1.Text);
 Valor1 := StrToFloat(Edit2.Text);
 Edit3.Text := FloatToStr(Valor1 *
Valor2);
End;
```

Fig. 5.21 – Código do botão btnMultiplica

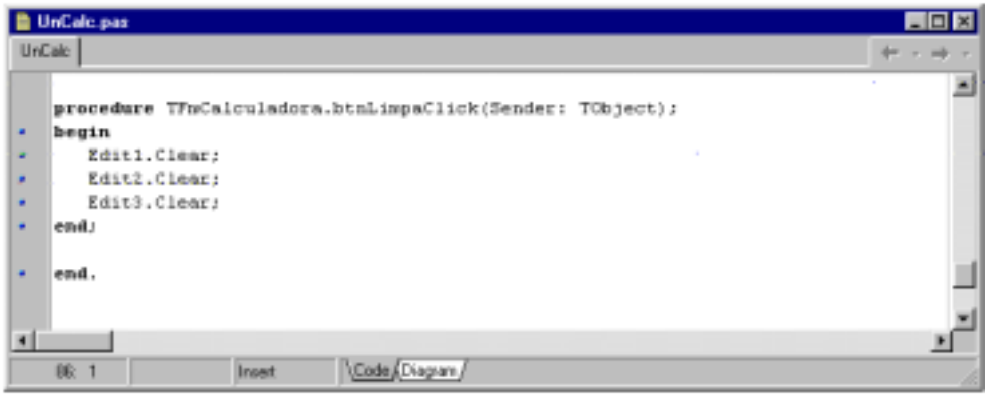
Dê um duplo clique no btnDivide para ativar o evento OnClick do mesmo. Insira o código conforme mostra a Figura 5.22.




```
Var
 Valor1, Valor2:Double;
Begin
 Valor1 := StrToFloat(Edit1.Text);
 Valor1 := StrToFloat(Edit2.Text);
 Edit3.Text := FloatToStr(Valor1 /
Valor2);
End;
```

Fig. 5.22 – Código do botão btnDivide

Dê um duplo clique no btnLimpa para ativar o evento OnClick do mesmo. Insira o código conforme mostra a Figura 5.23.




```
Edit1.Clear;
Edit2.Clear;
Edit3.Clear;
```

Fig. 5.23 – Código do botão btnLimpa

É evidente que esse programa é bem simples e contém diversos “furos”. Por exemplo, insira uma letra no Edit1 e um número no Edit2, após isso clique no botão de soma. Aparecerá um erro, afinal você não pode transformar uma letra em um número. Tente também inserir 10 no Edit1 e 0 no Edit2. Clique no botão de divisão. Outro erro, pois não se pode dividir por zero. Você teria que prever todas essas possibilidades e criar um código compatível para a solução de cada problema. No entanto, o exemplo proposto alcançou o objetivo.

EXERCÍCIOS (Resolva todos os exercícios e concorra a um brinde no final do curso)

01) Crie um Form que mostre a hora atualizada a cada segundo.

02) Crie um programa igual à calculadora padrão do Windows.

ADQUIRA JÁ A SÉRIE DE CDS COMPLETE

Essencial para qualquer técnico ou usuário de computador.

100 documentos (apostilas, livros, tutoriais, manuais...)
Programas para:
- Desempenho
- Manutenção
- Recuperação
- Segurança

Clientes VPN
Compressores de Executáveis
Emuladores
Programas Essenciais
Pacote Escritório
Ferramentas para BOOT
Ferramentas para HD
Máquina Virtual
Diversos utilitários.

Quer Passar no Vestibular? Este CD vai te ajudar.

Diversos documentos de atualidades, biologia, matematica, português, geografia, história, química, idiomas, física.

Diversas provas de vestibular.
Simulados.

423 E-Books para você ler e passar no vestibular.
Livros dos mais consagrados autores.

Os e-books estão separados por autores e por períodos literários: barroco, modernismo, humanismo etc.

Veja detalhes no site: www.alberteije.com.

Ação	**56**	**Derrote diversos inimigos em jogos de 1ª e 3ª pessoa.**
Corrida	**24**	**Vários jogos de corrida.**
Esportes	**58**	**Hockey, Baseball, Bicicross, Boliche, Golf, Futebol e muito mais.**
Cassino	**81**	**Diversos jogos de cartas e cassino.**
Naves	**24**	**Jogos de nave em 1ª e 3ª pessoa.**
Diversos	**120**	**Jogos de estratégia, raciocínio, aventura e muitos outros.**
Essenciais	**13**	**Acrobat, Divx, Emule, FireFox, Thunderbird, Winamp, WinRAR, Instalador de Codecs, FlashPlayer e outros.**

Acesse o site www.alberteije.com e veja mais detalhes sobre cada CD e como Adquirí-los.