

Capítulo 3

Desenvolvendo uma Aplicação

Ao final deste capítulo, você deve estar apto a :

- Desenvolver uma aplicação
- Utilizar o Project Manager
- Utilizar o Intergrated Debugger

Overview

O Delphi é uma poderosa linguagem de programação orientada a objeto com um avançado ambiente de desenvolvimento visual. Estas características, quando combinadas com a arquitetura de bancos de dados Borland, permite criar rapidamente aplicações Client/Server. Neste capítulo , você aprenderá sobre o processo de construção de uma aplicação Delphi, criar uma aplicação de exemplo e explorar os conceitos de gerenciamento de projetos.

Descrição do processo

Introdução

Uma característica do Delphi é que ele permite construir aplicações rapidamente. Esta seção descreve o processo de desenvolvimento de aplicações Delphi, como segue:

- Criando um Projeto
- Adicionando um Form ao Projeto
- Adicionando Componentes ao Form
- Definindo Propriedades dos Componentes
- Adicionando Event Handlers
- Compilando, Executando e Depurando a Aplicação

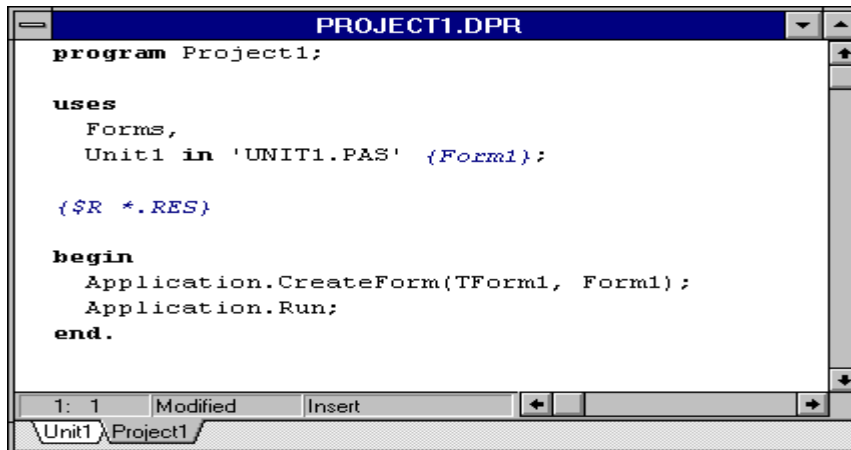
Estágio 1 - Criando um Projeto

O primeiro passo no desenvolvimento de uma aplicação no Delphi é criara um projeto. Aplicações no Delphi são gerenciadas como projetos. Criar um novo projeto gera um arquivo de projeto. O arquivo de projeto controla uma aplicação Delphi construindo os vários forms, executando a aplicação e exibindo o form principal da aplicação.

Arquivos de projeto contém código fonte Object Pascal gerado pelo Delphi que se torna parte do executável da aplicação quando for compilado e "Linkado".

Você pode começar um novo projeto inicializando o Delphi. Sempre que você inicializar o Delphi, um novo projeto é aberto. Se o Delphi já estiver aberto, você pode abrir um novo projeto através do menu **File**, Selecionando **New Project**.

O Delphi cria um arquivo de projeto default chamado PROJECT1.DPR, que o Delphi mantém durante o desenvolvimento da aplicação. Conforme o projeto for alterado, tal como adicionando novos forms, o Delphi atualiza o arquivo de projeto. O arquivo de projeto se parece com a figura a seguir:



```
program Project1;

uses
  Forms,
  Unit1 in 'UNIT1.PAS' {Form1};

{$R *.RES}

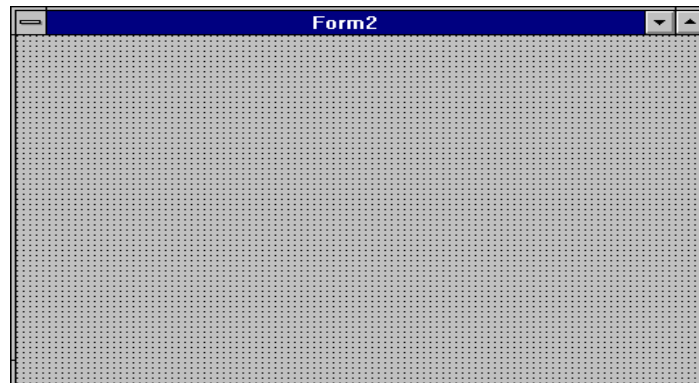
begin
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end.
```

Estágio 2 - Adicionando um Form ao Projeto

Como discutido anteriormente, os forms são a fundação das aplicações Delphi e fornece uma tela onde você pode criar a interface com o usuário de seu programa. Um projeto ou aplicação geralmente possui múltiplos forms. Adicionar um form ao projeto cria os seguintes arquivos adicionais:

- Um arquivo form com extensão .DFM contendo informações de resources para a construção do form.
- Um arquivo unit com extensão .PAS contendo código Object Pascal.

Todo form em uma aplicação possui estes dois arquivos associados a ele. Conforme for adicionando novos forms, o arquivo de projeto é atualizado automaticamente. O exemplo a seguir mostra o Form2, a ser adicionado ao projeto.

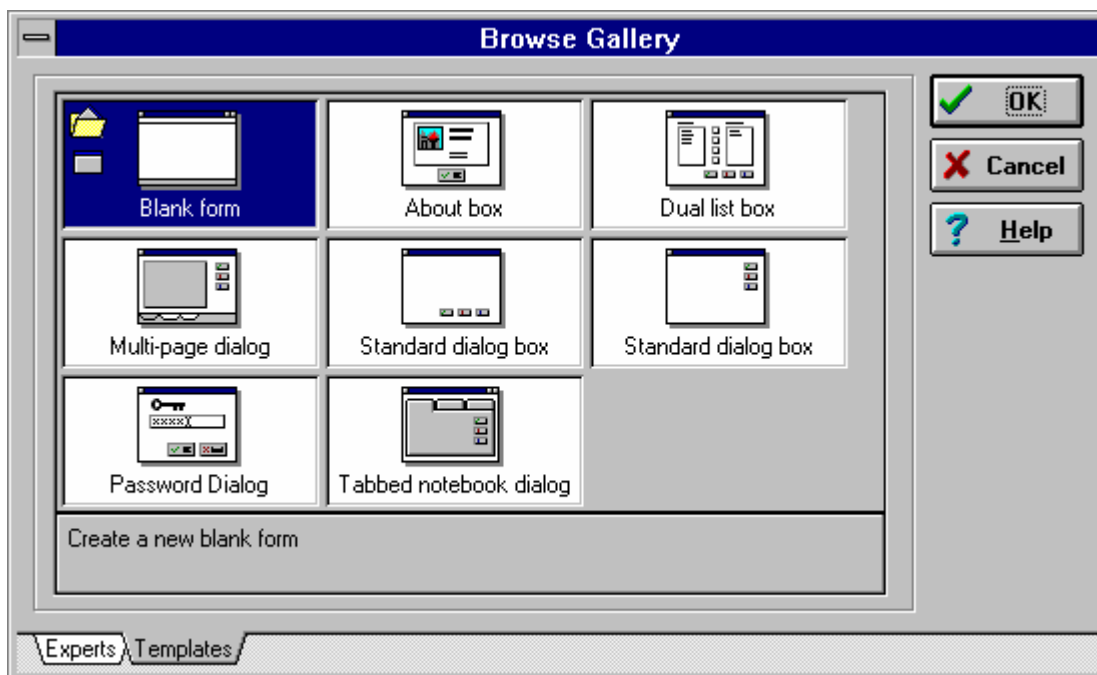


Quando você cria um projeto, um form (Form1) é criado por default. O nome de arquivo deste form é UNIT1.DFM.

Para Adicionar um Form ao Projeto

Execute os seguintes passos para adicionar um form um ao projeto:

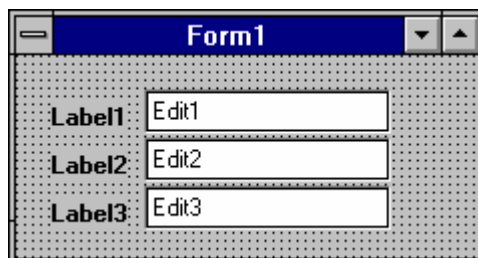
Passo	Ação
1	Para adicionar um ou mais forms ao projeto, no menu File , selecione New Form . Dependendo das configurações de Gallery na página Preferences do quadro de dialogo Environment Options, pode ocorrer: - Um form em branco aparece e é adicionado ao projeto - O quadro Browse Gallery aparece, como segue:



Passo	Ação
2	Aparecendo o quadro de dialogo Browse Gallery, selecione o tipo de form que você deseja adicionar ao projeto e de um clique em OK .

Estágio 3 - Adicionando Componentes ao Form

Como discutido anteriormente, componentes são objetos visuais de programa que você manipula durante o design. Os componentes disponíveis no momento estão na Component Palette. Após inserir um componente no form, você pode move-lo, editá-lo e reajustá-lo de acordo com suas necessidades. O exemplo a seguir mostra três componentes Label e três componentes Edit em um form:



Passos para Adicionar um Componente em um Form

Execute os seguintes passos para adicionar um componente ao form:

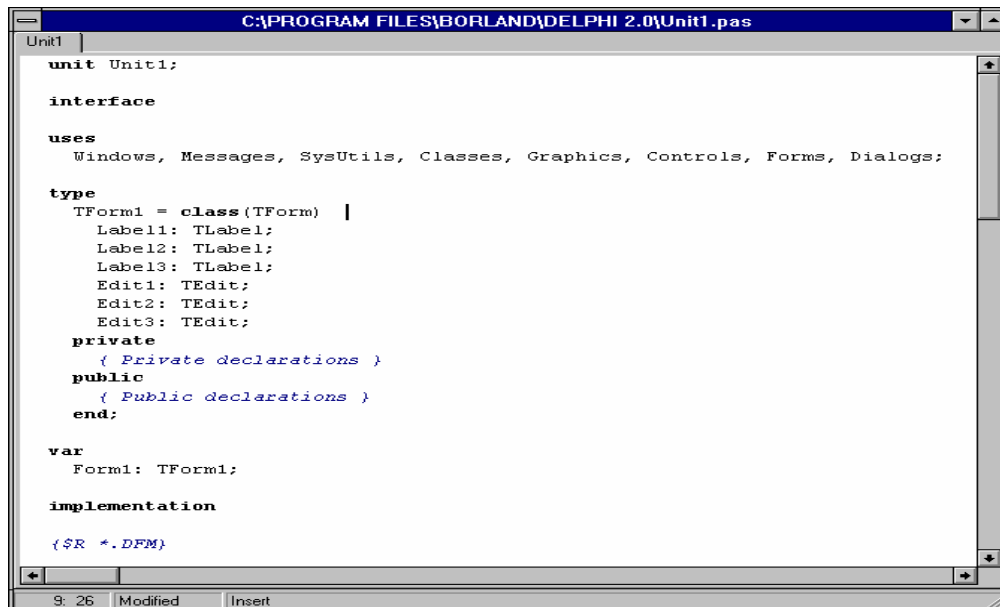
Passo	Ação
1	De um clique sobre um componente na Component Palette.
2	De um clique sobre o form onde o componente deve aparecer.
3	Reajuste o componente arrastando seus manipuladores.



Você também pode adicionar um componente com um duplo-clique no componente na Component Palette. Um componente até o local desejado e reajusta-lo através dos manipuladores.

Código Fonte da Unit após Adicionar Componentes

Quando um componente é adicionado ao form, o código fonte do arquivo é modificado. Especificamente a definição type para o membro correspondente ao componente adicionado. O exemplo a seguir exibe a definição de type no arquivo unit que corresponde ao form mostrado anteriormente com três componentes Label e três componentes Edit:



```
Unit1
unit Unit1;

interface

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

type
  TForm1 = class(TForm)
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Edit1: TEdit;
  Edit2: TEdit;
  Edit3: TEdit;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation
{$R *.DFM}
```

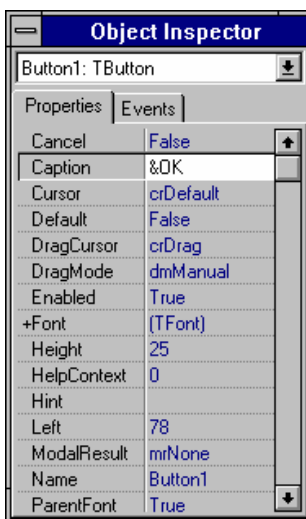
Estágio 4 - Definindo Propriedades dos Componentes

Como discutido anteriormente, cada componente tem um conjunto de atributos chamados de propriedades. Cada propriedade que possa ser alterada durante o design é exibido no Object Inspector. Você pode definir propriedade durante o design ou codificar para que uma propriedade do componente seja alterada durante a execução do programa.

Passos para Definir uma Propriedade de um Componente

Execute os passos a seguir para definir uma propriedade de um componente durante o design:

Passo	Ação
1	Dê um clique sobre o componente no form cuja propriedade você queira alterar.
2	Dê um clique sobre a aba da página Properties no Object Inspector. O exemplo a seguir mostra a página Properties para o componente Button.
3	Selecione a propriedade que você queira definir.
4	Altere o valor da propriedade na coluna Values.



Estágio 5 - Adicionando Event Handlers

Em programas dirigidos a eventos, responder a eventos do usuário ou do sistema é um ponto chave de sua aplicação. Além das propriedades, os componentes possuem uma lista de eventos que podem ser detectados. Quando você adiciona event handlers a sua aplicação, você está dizendo ao componente que execute os comandos programados sempre que um evento em específico seja detectado. Por exemplo, um botão em seu form pode detectar quando um usuário clica sobre si, o que é conhecido como evento OnClick. O evento OnClick faz com que sua aplicação execute a ação especificada no event handler.

Passos para Adicionar um Event Handler

Execute os passos a seguir para adicionar um event handler:

Passo	Ação
1	Dê um clique sobre o componente no form que precise de um event handler.
2	Dê um clique sobre a aba da página Events no Object Inspector para exibir uma lista de eventos para o componente.
3	Dê um duplo-clique à direita da coluna de eventos para fazer com que o Delphi gere um event handler vazio e exiba o handler no Code Editor.
4	Digite o código a ser executado quando ocorrer o evento.

Estágio 6 - Compilando, Executando e Depurando a Aplicação

O compilador e o depurador (debugger) são partes do ambiente Delphi. O compilador inclui um habilitador Make automático para que quando sua aplicação for alterada, somente os arquivos alterados sejam recompilados. O debugger está ativo sempre que você executar aplicações dentro do ambiente do Delphi.

Passos para Compilar e Executar a Aplicação

Execute os passos a seguir para compilar e executar a aplicação:

Passo	Ação
1	Para compilar o projeto atual sem inicializar o arquivo executável resultante, no menu Compile, selecione Compile.
2	Para compilar quaisquer alterações e executar o arquivo do projeto corrente, selecione Run no menu Run.

Tutoria: Criando uma Aplicação

Introdução

O processo a seguir é um tutorial de exemplo. Uma maneira de entender o processo de desenvolvimento de aplicações no Delphi é construir uma aplicação de exemplo. Esta seção fornece um rumo na construção de uma aplicação simples utilizando diversos componentes padrão do Delphi. A aplicação permite que o texto seja digitado em edit box e adicionado em um list box com o clique de um botão.

Estágios do Tutorial

O processo deste tutorial envolve os seguintes estágios:

Estágio	Processo
1	Criar um projeto de exemplo
2	Adicionar componentes padrão
3	Definir propriedades dos componentes
4	Adicionar um event handler
5	Compilar e executar a aplicação de exemplo

Passos para Estágio 1

Execute os passos para abrir um novo projeto e chamá-lo PSAMPLE.DPR:

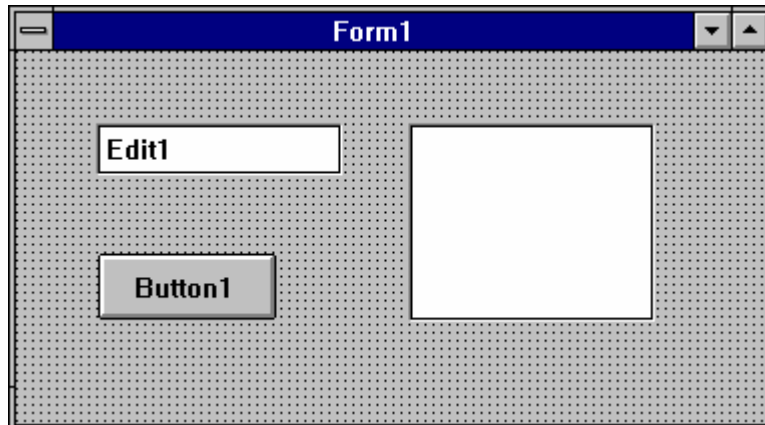
Passo	Ação
1	No menu File, selecione New Project para criar um projeto. O processo de abertura de um novo arquivo projeto adiciona automaticamente um novo form. Se o quadro de dialogo Browse Gallery aparecer, Blank Form é a opção default. Dê um clique em OK .
2	Se for solicitado a gravação das alterações do projeto atual selecione No .
3	No menu File, selecione Save Project As .
4	Quando o nome da unit for solicitado, digite <i>USAMPLE.PAS</i> .

	Este nome substitui o nome default UNIT1.PAS.
5	Quando o nome do projeto for solicitado, digite <i>PSAMPLE.DPR</i> . Este nome substitui o nome default PROJECT1.DPR.

Passos para o Estágio 2

Execute os passos a seguir para adicionar componentes da página Standard ao form no projeto PSAMPLE:

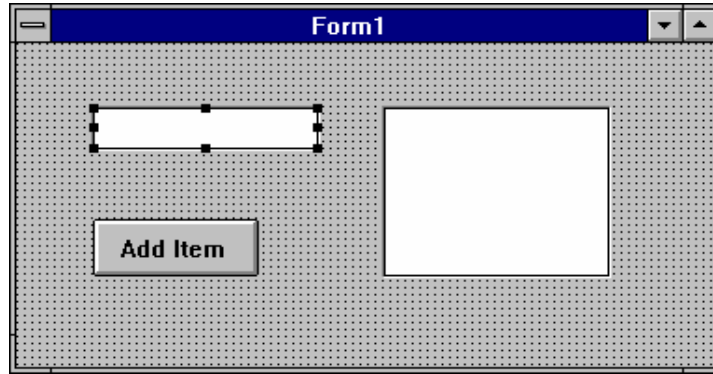
Passo	Ação
1	Dê um clique sobre a aba da página Standard da Component Palette para visualizar os componentes Standard.
2	Mova o ponteiro do mouse vagarosamente sobre cada componente para que o Help Hint seja exibido, e localize os componentes Button, Edit e ListBox.
3	Dê um duplo-clique sobre o componente Button para inseri-lo no form. Arraste o componente até a parte inferior do form.
4	Dê um duplo-clique sobre o componente Edit, e mova o componente até a parte superior do form.
5	Dê um duplo-clique sobre o componente ListBox para inseri-lo no form.
6	Arranje os componentes e reajuste o tamanho do form para que se pareça coma figura a seguir.



Passos para Estágio 3

Execute os passos a seguir para definir as propriedades dos componentes Edit e Button:

Passo	Ação
1	Dê um clique sobre o componente Edit. As propriedades do componente Edit são exibidas no Object Inspector. O nome default deste componente é Edit1, e seu tipo de objeto é TEdit.
2	No Object Inspector, dê um clique sobre a coluna Values da propriedade Text de Edit1 e a apague.
3	Defina a propriedade Caption para Add Item . Seu form deve se parecer com a figura a seguir:



Passos para o Estágio 4

Execute os passos a seguir para adicionar um event handler para o evento OnClick do componente **Add Item**:

Passo	Ação
1	Dê um clique sobre o botão Add Item de seu form para exibir as propriedades no Object Inspector.
2	Dê um clique sobre a aba Events do Object Inspector para exibir a página Events do botão.
3	Dê um duplo-clique sobre a coluna a direita do evento OnClick. O nome da procedure Button1Click aparecerá na coluna. O Delphi gera um event handler vazio e o exibe no Code Editor. Digite o código a seguir dentro das declarações begin....end; da procedure.

```

if Edit1.Text<> " then { Edit1 não esta vazio }
begin
  ListBox1.Items.Add(Edit1.text);
  Edit1.Text := "";
end;

```

Passos para o Estágio 5

Execute os passos a seguir para compilar e executar sua aplicação:

Passo	Ação
1	No menu Run, selecione Run. Esta opção compila e executa sua aplicação.
2	Digite algum valor, tal como seu nome, no componente Edit.
3	Dê um clique em Add Item para adicionar cada item a list box. Adicione nove ou dez itens a list box.
4	No menu File, selecione Save Project, e depois Close Project.

Utilizando o Project Manager

Introdução

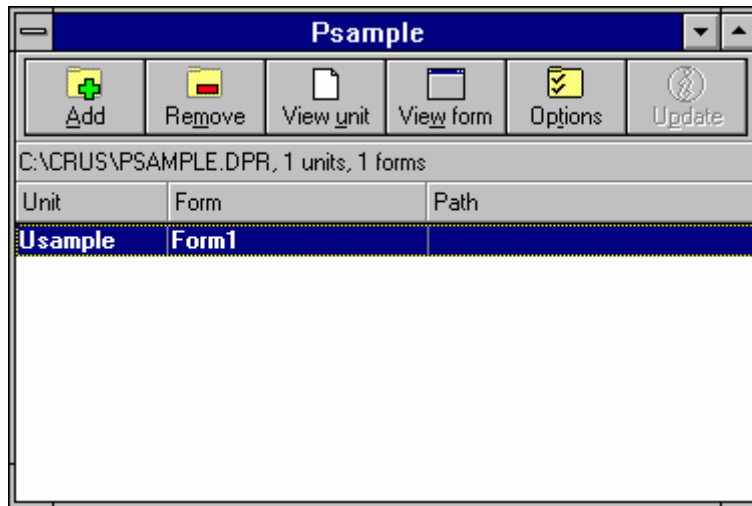
O Delphi permite que você gerencie seus projetos através do Project Manager. O Project Manager lista os arquivos que compõe seu projeto e permite que você navegue pelos arquivos. Você também pode utilizar o Project Manager para:

- Adicionar units e forms ao projeto

- Remover units e forms de um projeto
- Especificar o form principal
- Especificar a localização dos arquivos de Help e ícone

Exibindo a Janela do Project Manager







Para exibir o Project Manager, no menu **View**, selecione **Project Manager**. A janela Project Manager aparecerá, como segue:



Descrição

dos Botões do Project Manager

A tabela a seguir descreve os botões do Project Manager:

Botão	Descrição
 Add	Adiciona um arquivo de unit ao projeto atual
 Remove	Remove um arquivo de unit do projeto atual
 View unit	Exibe um arquivo de unit do projeto atual
 View form	Exibe um form no projeto atual
 Options	Exibe o quadro de diálogo Project Options para: <ul style="list-style-type: none"> • Alterar o form default, opções do compilador, aplicação, linker e diretório • Define símbolos condicionais
 Update	Grava as alterações no projeto atual

Adicionando Units e Forms ao Projeto Utilizando o Project Manager

Um projeto default inicialmente contém um form e um arquivo de unit de código fonte. Entretanto, a maioria dos projetos contém múltiplos forms e units.

Você pode dar um clique com o botão direito do mouse e selecione **New Form** no SpeedMenu para adicionar units e forms ao seu projeto. Você também pode adicionar forms e units existentes ao seu projeto utilizando o botão **Add** na SpeedBar do Project Manager e selecionar o form ou unit a ser adicionado.

Removendo Units e Forms de um Projeto Utilizando Project Manager

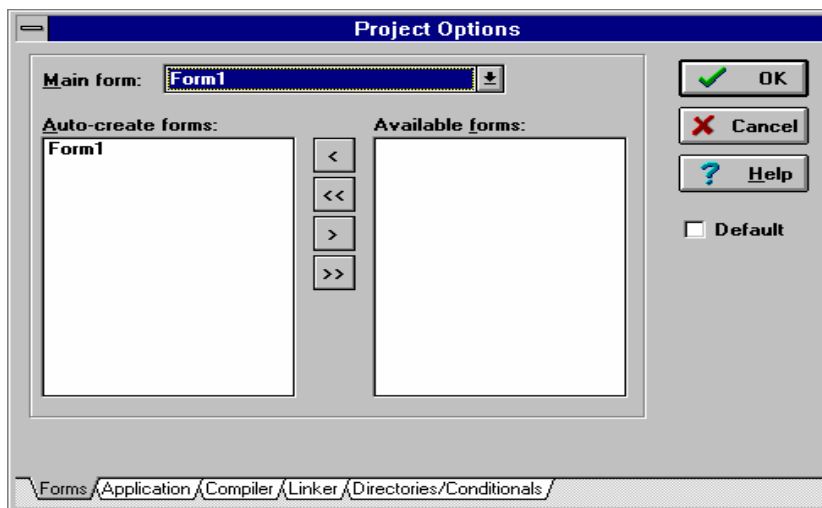
Durante o desenvolvimento de um projeto, você pode achar necessário remover forms e units de seu projeto. Você pode fazê-lo a qualquer momento durante o desenvolvimento. Entretanto, como um form está sempre associado a uma unit, você não pode remove um sem remover o outro a menos que a unit não possua nenhum form associado a ele. Você pode remover units de um projeto utilizando o botão Remove no Project Manager.



Remover arquivos utilizando qualquer outro programa gerenciador de projeto ou digitando comandos no prompt do DOS não é recomendado. Estas ações não removem as entradas da cláusula Uses dos arquivos .DPR ou da janela do Project Manager e causará erros quando você compilar o programa.

Definindo Opções de Projeto

O quadro de dialogo Project Options permite definir diversas opções que afetam seus projetos. Para acessar o quadro de dialogo Project Options, de um clique sobre o botão Options no Project Manager. O quadro Project Options aparecera, como segue:



Descrição das Páginas do Project Options

A tabela a seguir descreve cada uma das páginas no quadro de dialogo Project Options e algumas das opções mais importantes em cada página:

Página	Descrição
Forms	Especifica o form principal de sua aplicação, os forms que devem ser criados automaticamente, e a ordem destes forms.
Application	Especifica um título, arquivo de help, e um ícone para sua aplicação.
Compiler	Permite definir opções para a forma como seu programa será compilado. Estas opções correspondem a definir diretivas a seu estado positivo (+) no código de seu programa.
Linker	Permite especificar a forma como seus arquivos de programa serão linkados.
Directories / Conditionals	Especifica a localização dos arquivos que o Delphi necessita para compilar, linkar e distribuir seu programa.

Utilizando Integrated Debugger

Introdução

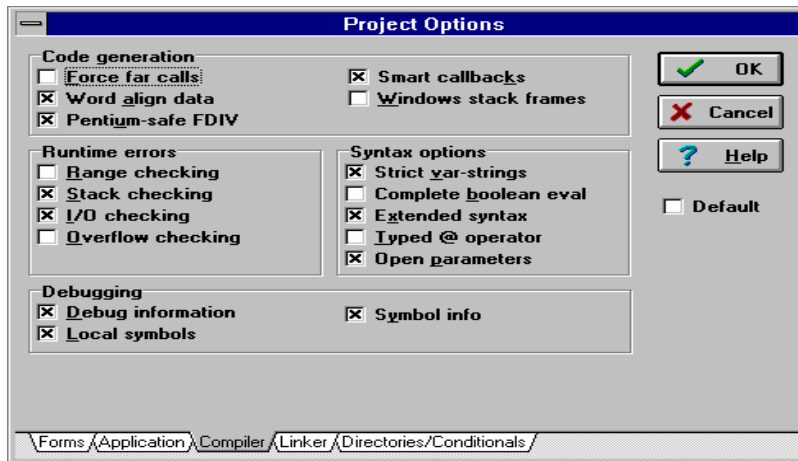
O Delphi possui um depurador totalmente integrado que permite depurar uma aplicação sem deixar o ambiente visual de desenvolvimento. Ele oferece a capacidade de :

- Gerar informações a depuração de dentro de seu executável
- Habilitar e desabilitar a depuração integrada de dentro do IDE
- Definir breakpoints
- Visualizar o conteúdo das variáveis do programa
- Modificar valores de dados durante a execução do programa
- Visualizar o call stack

Gerando Informação de Depuração

O Delphi gera informação simbólica de depuração quando você compila seu programa com a ação de depuração habilitada. As opções de depuração encontram-se na página compiler do quadro de dialogo Project Options.

A página Compiler e opções de depuração para uma aplicação típica aparece como segue:



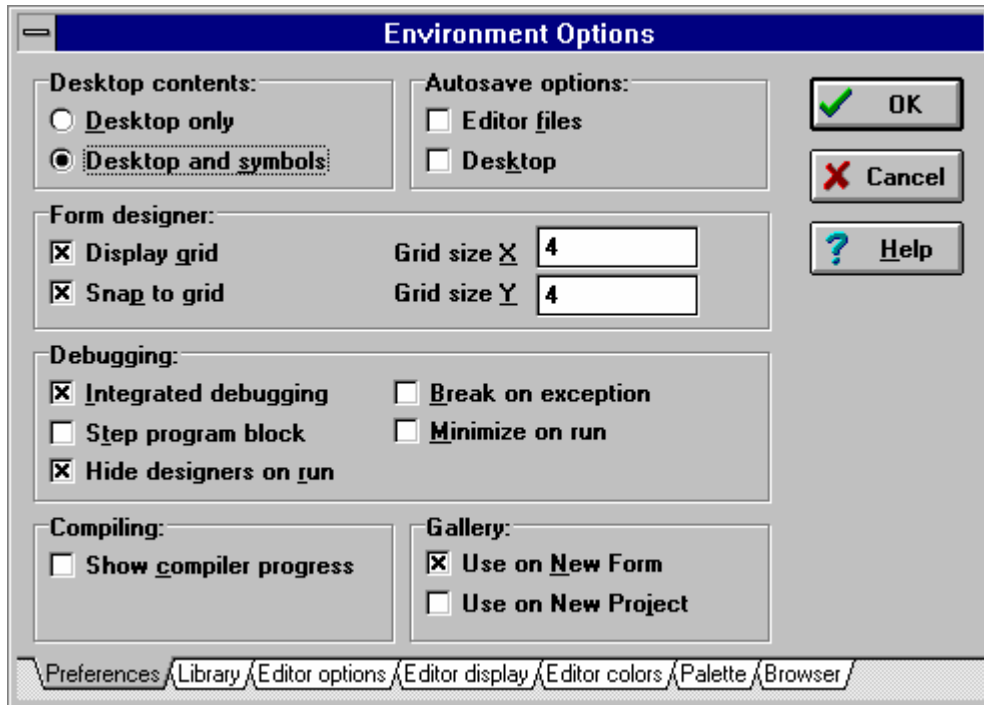
Descrição das Opções do Depurador na Página Compiler

A tabela a seguir descreve as opções de depuração na página Compiler do quadro de diálogo Project Options:

Opção	Descrição
Debug Information	Inserir informação de depuração no arquivo UNIT.DCU. Informação de depuração consiste de uma tabela com linhas numeradas para cada procedure que mapeia endereços de código de objetos nos números.
Local Symbols	Gera local symbols information. Local symbols information consiste de : <ul style="list-style-type: none"> Os identificadores na implementation part (e não na interface part) do módulo Os identificadores dentro das procedures e funções do módulo. Local symbols information não inclui variáveis globais ou nomes declarados na interface section de uma unit.
Symbol Info	Gera symbol information. Symbol reference information consiste de tabelas que fornecem os números de linhas de todas as declarações e referências aos símbolos em módulo.

Habilitando o Intergrated Debugger

A depuração integrada está habilitada após a instalação do Delphi. É uma opção de ambiente da página Preferences que você pode definir. A página Preferences do quadro de diálogo Environment Options aparece como segue:



Descrição das Opções de Depuração na Página Preferences

A tabela descreve as opções de depuração na página Preferences do quadro de diálogo Environment Options:

Opção	Descrição
Intergrated Debugging	Quando habilitada, a depuração integrada está em efeito.
Step Program Block	Quando habilitada, o depurador passa pelos blocos de programa.
Hide Designer on run	Quando habilitada, a interface de design é oculta enquanto a aplicação estiver sendo executada.
Break on exception	Quando habilitada, a aplicação para quando ocorrer uma exception e a linha de código causadora é exibida no Code Editor.
Minimize on run	Quando habilitada, o Delphi é minimizado sempre que você executar uma aplicação dentro do IDE.

Controlando a Execução do Programa

Uma característica importante do depurador é que ele permite selecionar como o programa será executado, uma declaração por vez. Uma linha de código por vez, uma função inteira por vez, e assim por diante. O depurador permite controlar a execução de seu programa das opções no menu **Run**. O menu **Run**, como segue:

Run	
Run	F9
Parameters...	
Step Over	F8
Trace Into	F7
Run to Cursor	F4
Show Execution Point	
Program Pause	
Program Reset	Ctrl+F2
Add Watch...	Ctrl+F5
Add Breakpoint...	
Evaluate/Modify...	Ctrl+F7

Descrição do Menu Run

A tabela a seguir fornece uma declaração das opções disponíveis no menu Run.

Opção	Descrição
Run	Permite compilar e executar suas aplicações.
Parameters	Permite especificar os parâmetros de inicialização para sua aplicação
Step Over	Permite executar seu programa uma linha por vez, pulando procedures, executando-as como uma única unit.
Trace Into	Permite executar seu programa uma linha por vez, rastreando a procedure e seguindo a execução de cada linha.
Run to Cursor	Permite executar o programa até a localização do cursor no Code Editor
Program Pause	Permite interromper temporariamente a execução de um programa
Program Reset	Permite terminar o programa sendo executado e remove-lo da memória
Add Watch	Abre o quadro de dialogo Watch Properties, permitindo criar e modificar observações
Add Breakpoint	Abre o quadro de dialogo Edit Breakpoint, permitindo criar e modificar pontos de interrupção
Evaluate/Modify	Abre o quadro de dialogo Evaluate/Modify, permitindo avaliar ou alterar o valor de uma expressão existente

Definindo Pontos de Interrupção

Você pode definir pontos de interrupção onde quiser que a execução seja interrompida. Pontos de interrupção são particularmente úteis quando utilizados em conjunto com Watches. Seu programa é executado na velocidade normal até atingir o ponto de interrupção. Atingindo o ponto, o depurador exibe o Code Editor com a linha contendo o ponto de interrupção, permitindo modificar o código ou exibir o valor de variáveis utilizando a janela de observação.

Para definir um ponto de interrupção, você pode:

- Dar um duplo-clique a esquerda da linha do código fonte onde quiser definir um ponto de interrupção.
- Selecionar Toggle Breakpoint utilizando o Speedmenu Code Editor.

Pontos de Interrupção no Code Editor

Quando definir um ponto de interrupção, a linha de código correspondente ao ponto é realçada e um ícone de sinal de parada aparece na margem esquerda como mostrado na figura a seguir:



Você também pode utilizar a página Editor Colors do quadro de diálogo Environment Options para definir uma cor diferente, indicando pontos de execução, linhas de pontos de interrupção inválidos, e a propriedade habilitado/desabilitado do ponto de interrupção.

```
USAMPLE.PAS
procedure TForm1.Button1Click(Sender: TObject);
begin
  If Edit1.Text = '' Then
    Edit1.Text := 'Aprendendo Delphi';
  end;
end.
```

31: 1 Modified Insert

Usample

Visualizando o Conteúdo de Variáveis

A janela Watches monitora a alteração do valor das variáveis ou expressões durante a execução de seu programa. Conforme seu programa seja executado, quer esteja pulando sobre o código, o conteúdo da janela de observação é alterado conforme os valores das variáveis contidas na expressão de observação.

Adicionando Expressões de Observação

Para iniciar uma expressão de observação, selecione Add Watch no menu Run. O quadro de diálogo Watch Properties aparece, como segue:



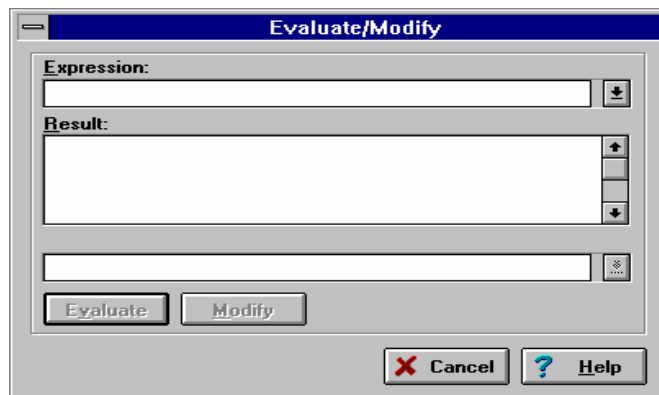
Exibindo

Expressões de Observação

A janela Watches exibe os valores atuais de expressões de observação e permite adicionar, editar, deletar, habilitar e desabilitar observações. Para exibir a janela Watch List, selecione Watches no menu View. A janela Watch List aparece.

Quadro de Diálogo Evaluate/Modify

No Delphi, você pode alterar os valores de variáveis e itens nas estruturas de dados durante a depuração. Para avaliar ou alterar o valor de uma expressão existente, utilize o quadro de diálogo Evaluate/Modify. Alterações feitas no quadro de diálogo Evaluate/Modify não afetam o código fonte ou o programa compilado. Para tornar as alterações permanentes, modifique seu código fonte e recompile seu programa. O quadro de diálogo Evaluate/Modify aparece como segue:



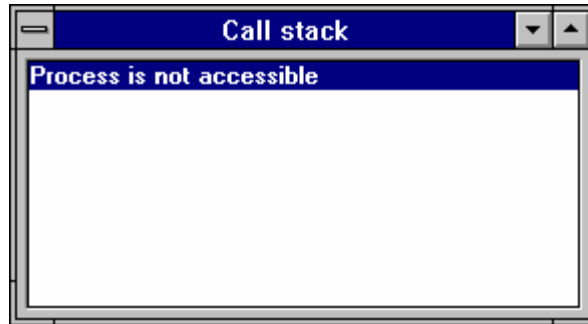
Visualizando o Call Stack

Quando estiver depurando, você pode achar útil a ordem das chamadas de funções ativas. A janela Call exibe o fluxo atual das chamadas de funções.

Janela Call Stack

A janela Call Stack exibe chamadas de funções ativas. Cada função aparece com uma lista de argumentos utilizadas na chamada. Você também pode utilizar a janela Call Stack para visualizar ou editar o código-fonte

associado com uma determinada chamada de função. Para visualizar a janela Call Stack, você pode selecionar View na barra de menu, e depois Call Stack. A janela Call Stack aparece como segue:



Lab: Modificando a Aplicação de Exemplo

Objetivos

Este Lab reforça sua habilidade em:

- Adicionar um componente ao form.
- Definir propriedades de um componente.
- Adicionar um event handler.
- Compilar um programa Delphi.

Cenário

Neste capítulo, você aprendeu a criar uma aplicação em um projeto chamado PSAMPLE. Em PSAMPLE, você adicionou os componentes ListBox, Button e Edit ao Form. O Evento OnClick do form adiciona itens digitados no componente Edit a list box. Neste lab, você vai melhorar esta aplicação, adicionando um botão Deletar e um Sair para esta aplicação.

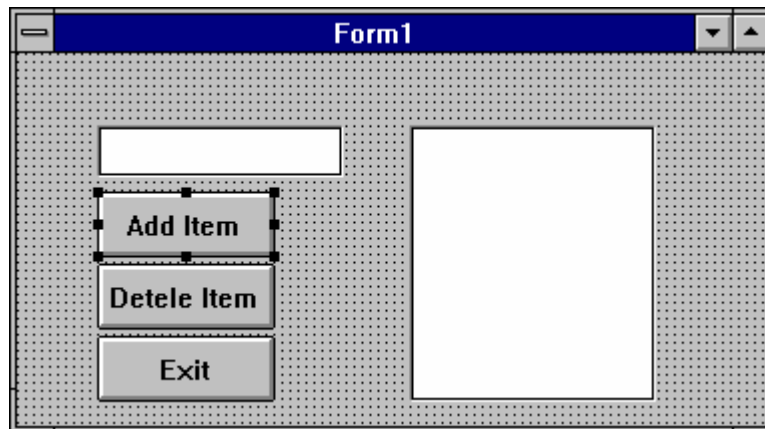
Processo

Utilize as seguintes diretivas para modificar a aplicação PSAMPLE:

Estágio	Processo
1	Abra a aplicação PSAMPLE criada neste capítulo e adicione dois botões ao form.
2	Defina as propriedades para estes botões para que um botão (Button2) tenha o Caption <i>Delete Item</i> e o outro botão (Button3) tenha o Caption <i>Exit</i> e o primeiro com Add Item .
3	Adicione um event handler para o evento OnClick do botão com o Caption Exit . Utilize o método Close do TForm para sair da aplicação. A declaração para este evento segue abaixo: <i>Close;</i>
4	Adicione um event handler para o OnClick do botão com o Caption Delete Item . O event handler completo segue abaixo: <i>Procedure TForm.Button2Click(Sender: TObject); begin</i>

	<pre> with ListBox1 do begin if ItemIndex <> -1 then Items.Delete(ItemIndex); end; end; </pre>
5	Inclua também no event OnClick do botão com Caption Add Item o código abaixo: <pre> ListBox1.Items.Add(Edit1.Text); </pre>
6	Grave e execute a aplicação
7	Adicione quatro ou cinco itens à lista box
8	Delete itens da lista

Seu form deve estar similar a figura a seguir:



Processos Opcionais

O event handler para o evento OnClick do botão **Deletar Item** utiliza uma declaração de programação chamada de declaração **with**.

Utilize o sistema de help para determinar o que esta declaração faz. Revise o event handler Button2Click, habilitando o método para que você possa deletar a declaração with e as palavras-chave **begin...end** associadas.

Resumo do Capítulo

Pontos chave

Após completar este capítulo, você aprendeu:

- Que arquivos de projeto do Delphi controlam as aplicações construindo vários forms, executando as aplicações e exibindo o form principal da aplicação.
- Que adicionar um form cria dois arquivos:
 - Um arquivo de form com extensão .DFM contendo informações de resource do Windows e código Object Pascal do Delphi para a construção do form.
 - Um arquivo de unit com extensão .PAS contendo código Object Pascal.

- O Project Manager do Delphi lista os arquivos que compõem sua aplicação e permite navegar pelos arquivos.
- Que o Delphi possui um depurador totalmente integrado que permite depurar aplicações sem ter que deixar o ambiente de desenvolvimento visual.

Termos e definições

A tabela a seguir é uma referência rápida para os termos explicados neste capítulo.

Termo	Descrição
Breakpoint	Uma marcação em seu programa que causa uma pausa na execução durante o processo de depuração.
Event handler	Uma procedure que diz ao componente para que execute determinadas declarações do programa quando um evento em especial é detectado.
Project File	Um arquivo que controla uma aplicação Delphi construindo vários forms, executando a aplicação e exibindo o form principal da aplicação. O nome default é PROJECT1.DPR.
Watches	Expressões que permite monitorar o valor das variáveis ou expressões enquanto seu programa estiver sendo executado.

