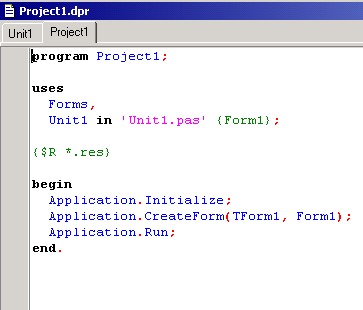
|  |  |
| --- | --- |
| **<<**[**Click to Display Table of Contents**](http://help.market.com.br/desenvolvimento/index.html?diretivas_de_compilacao.htm)**>>**  **Diretivas de compilação** | [Previous page](http://help.market.com.br/desenvolvimento/performance_com_relatorios_gra.htm)[Return to chapter overview](http://help.market.com.br/desenvolvimento/delphi_-_introducao.htm)[Next page](http://help.market.com.br/desenvolvimento/minput.htm) |
|  | |

**Diretivas de Compilação**

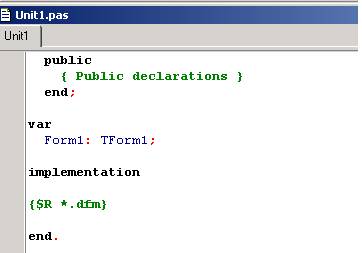
Bem, amigos o assunto é: Diretivas de Compilação. Mas antes de começar gostaria de lhes contar uma história divertida:

*Certa vez, quando programava, em meu local de trabalho, meu gerente me pediu para criar uma função que pegasse o número de versão do sistema configurada no options do projeto. Eu já tinha o algoritimo desenvolvido em algum lugar. Após algumas horas o encontrei e comecei o trabalho. Terminada a seção CTRL + C e CTRL + V lá vou eu para o F9 e ver o resultado. Ao chamar a função, para minha surpresa, tudo zero. Nada de números de Versão, Nome de Empresa, nada vezes nada. Naturalmente acionei o menu Project >> Options para ajustar as configurações do projeto. Socorro !!! Gritei. - a tela de opções estava com todos os seus controles desativados. Com os olhos lacrimejando e pálido pensei: - algo deu errado do DPR. Muito bem, corri de um lado para outro do escritório atrás de solução. Nada. Foi ai que resolvi olhar os backups do DPR e notei uma leve diferença.Um maldito "comentário" que arrancara do código dias antes: {$R \*.RES}. Opa, você também achava que era comentário? Então sente e continue a ler. Esse trecho de código não é um comentário comum e sim uma "Diretiva de Compilação", encontrada em diversos locais do Delphi. E é exatamente sobre este assunto que vou comentar. As diretivas realmente se parecem muito com comentários, exceto pelo caracter cifrão presente imediatamente após o caracter chave "{", que delimita o trecho. Mas, para que servem? Como utilizar? E por que?*

Bom, imagine que você tem uma unit completa com funções que são usadas por todos os seus sistemas, inclusive projetos para internet. Mas aquela função que exibe mensagens de aviso em determinada ação não é mostrada quando o projeto é compilado para WEB, isso porque na WEB é necessário usar uma outra forma de mostrar mensagens. O que fazer ??? Você não quer criar uma nova unit para ser usada com projetos de desenvolvimento para internet, pois em caso de manutenção no código, seria necessário alterar as funções da unit "A" e da unit "B". Então nesse caso podemos usar as diretivas e aumetar exponencialmente nosso leque de possibilidades do Delphi. Para estudarmos melhor o caso observe o código das Figuras 1 e 2:



*Figura1*

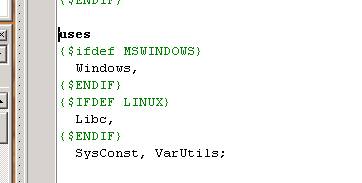


*Figura2*

Nesses exemplos encontramos a diretiva $R declarada no Project Source e no código do form.

{$R \*.RES} e {$R \*.DFM}

A Diretiva $R informa ao Delphi que devem ser criados os arquivos \*.DFM para o form e \*.RES para o projeto. Se você abrir a unit Variants do Delphi 6.0 e 7.0 e em seguida procurar por MSWINDOS, irá encontrar o trecho de código da figura3.



*Figura 3*

Aqui é decidido qual unit usar no momento da compilação, de acordo com o sistemas operacional.

Windows para MSWINDOWS e Libc para LINUX. Essa prática é muito comum em fontes de componentes Delphi.

A diretiva de compilação não pode ser confundida com comentários de código, já que o mesmo fica entre chaves {}. A diferença entre uma diretiva e um comentário está no caracter cifrão $, que é colocado imediatamente após a abertura de chaves {. Por exemplo:

{$IFDEF VER150}

   ShowMessage('Compilando em Delphi 7.0');

{$ELSE}

   ShowMessage('Não sei qual a versão do delphi');

{$ENDIF}

No caso acima, {$IFDEF} é usado para verificar se a diretiva VER150(Versão do Delphi 7.0) está definida; se verdadeiro uma showmessage irá aparecer dizendo que a compilação está sendo efetuada em Delphi 7.0, caso contrário o sistema não sabe qual o Delphi que foi usado para compilar o código. A diretiva VER está contida internamente no Delphi, veja:

VER80 - Delphi 1

VER90 - Delphi 2

VER100 - Delphi 3

VER120 - Delphi 4

VER130 - Delphi 5

VER140 - Delphi 6

VER150 - Delphi 7

VER160 - Delphi 8

VER170 - Delphi 2005

VER180 - Delphi 2006

VER180 - Delphi 2007

VER185 - Delphi 2007

WIN32 - Indicates that the operating environment is the Win32 API.

LINUX - Indicates that the operating environment is Linux

MSWINDOWS - Indicates that the operating environment is the MS Windows/li]

CONSOLE - Indicates that an application is being compiled as a console application

Podemos usar também a verificação com {$IFNDEF} que significa “Se não estiver definida”, ex:

{$IFNDEF VER150}

ShowMessage('Versão diferente de Delphi 7.0');

{$ENDIF}

Note que ao fim da verificação existe um “finalizador”. {$ENDIF} é obrigatório se estiver usando {$IFDEF}.

Poderiamos ter a seguinte estrutura:

{$IFDEF MINHA\_PRIMEIRA\_DIRETIVA}

   {$IFDEF MINHA\_SEGUNDA\_DIRETIVA}

      {$IFDEF MINHA\_TERCEIRA\_DIRETIVA}

      {$ENDIF}

   {$ENDIF}

{$ENDIF}

Mais uma vantagem disso é que as diretivas podem ser usadas em qualquer parte do código fonte. Tente fazer um IF dentro da cláusula public ou mesmo fora de uma procedure ou function. Certamente não irá conseguir. Já com diretivas, o caso é outro. Você inclusive pode criar suas próprias diretivas.

Vamos a um exemplo prático disso.

1.Abra o Delphi.

2.Mude a propriedade Name do form para fPrincipal e salve-o como Principal.pas        
Salve o projeto como Diretivas.dpr.

3.Crie dois novos forms e mude a propriedade Name para fLoja1 e fLoja2 respectivamente, e salve-os como Loja1.pas e Loja2.pas.

4.Para destinguir um do outro coloque um label em cada e mude o caption para Loja1 e Loja2. Se preferir mude também outras características dos forms como por exemplo: cor e tamanho. Figura 4

5.Entre então em Project >> Options >> Forms e mude os dois forms fLoja1 e fLoja2 para que não sejam criados automaticamente pelo Delphi. Retire-os da lista Auto-Create Forms.

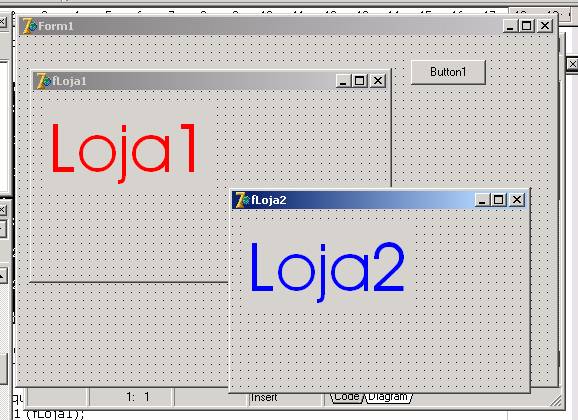
6.Vamos aproveitar e configurar outra característica de nosso projeto.

7.Clique na guia Directories/Conditionals

8.Em Conditional Defines digite WEB;(essa será nossa diretiva de compilação)

9.Agora imagine que quando o sistema estiver sendo compilado para Internet ele deve abrir o form fLoja1 e quando o aplicativo for Cliente  
Servidor o form fLoja2 será aberto.

Veja como fica na figura abaixo:



*Figura 4*

Para saber que tipo de sistema estamos compilando, vamos usar a diretiva WEB que foi definida por nós.

No form principal coloque um botão e em seu evento onClick digite o código a seguir:

{$IFDEF WEB}

   fLoja1 := TfLoja1.Create(Self);

   Try

      fLoja1.ShowModal;

   Finally

      fLoja1.Free;

   end;

{$ELSE}

   fLoja2 := TfLoja2.Create(Self);

   Try

      fLoja2.ShowModal;

   Finally

      fLoja2.Free;

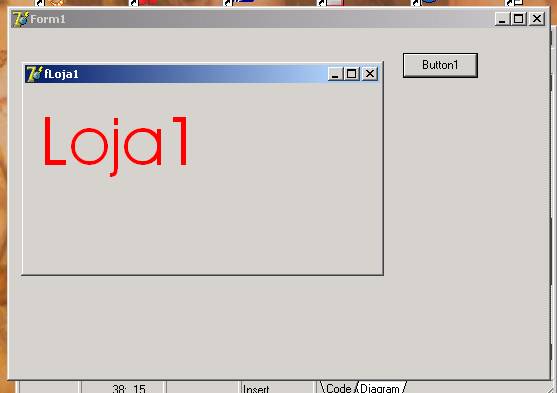
   end;

{$ENDIF}

Não esqueça de adicionar os dois forms a cláusula USES do form principal;

Clique em Build All e execute o projeto.

Observe que ao clicar no botão que foi adionado no form Principal, o form1 (fLoja1) é aberto; Isso acontece porque a diretiva WEB está presente no sistema.



*Figura 5*

Feche o sistema e faça um novo teste. Entre em Project >> Options guia Directories/Conditionals. Retire de Conditional Defines a diretiva WEB e recompile o código usando Build All. Execute o projeto e clique novamente no botão.

Desta vez o fLoja2 é aberto.



*Figura 6*

No nosso exemplo usamos o tratamento {$IFDEF WEB} para verificar se a diretiva WEB está definida para esse projeto, então abrimos o form1. Caso negativo, o form2 é aberto. Isso poderia ser usado para desenvolvedores que reutilizam fonts para sistemas diferentes. Ex:

{$IFDEF CADASTROS}

   ShowMessage('Sistema de Cadastros');

{$ENDIF}

Você ainda pode usar as diretivas $DEFINE E $UNDEF antes da compilação.

Ex.

{$IFDEF VER150}

   {$DEFINE MINHA\_DIRETIVA}

{$ENDIF}

Nesse caso estou definindo uma diretiva com o nome MINHA\_DIRETIVA para o projeto.

Há ainda a possibilidade de usar arquivos do tipo INC, onde as diretivas são definidas. Procure por arquivos com esta extensão nos diretórios do Delphi, encontrará uma porção deles; nem todos possuem diretivas, esses arquivos também podem conter variáveis, constantes e etc. Para usar esses arquivos é necessário fazer uso da diretiva $I ou $INCLUDE.

Ex.

{$I arquivo.inc} ou {$INCLUDE arquivo.inc};

Se não especificado o diretório, o Delphi procura o arquivo no mesmo diretório onde o .pas do arquivo que possui a diretiva está gravado. Para essas diretivas também é possível usar outros tipos de arquivos, como por exemplo \*.pas.

Para finalizar, entre na unit Qdialogs do Delphi 7.0 e procure por SelectDirectory e veja a declaração da function:

{$IFDEF LINUX}

   function SelectDirectory(**const** Caption, Root: WideString;

      var Directory: WideString; ShowHidden: Boolean = False): Boolean;

{$ENDIF}

{$IFDEF MSWINDOWS}

   function SelectDirectory(**const** Caption: string; **const** Root: WideString;

      out Directory: WideString): Boolean;

{$ENDIF}

A declaração da função muda de acordo com o sistema operacional.

Para ir um pouco mais além, tente mudar o parãmetro de uma função ou procedure usando diretivas.

Function Minha\_Function({$IFDEF VER140} Parametro : String{$ELSE} Parametro : integer) : Boolean;

Fonte

<http://www.clubedainformatica.com.br/mostraartigo.php?artigo=99>