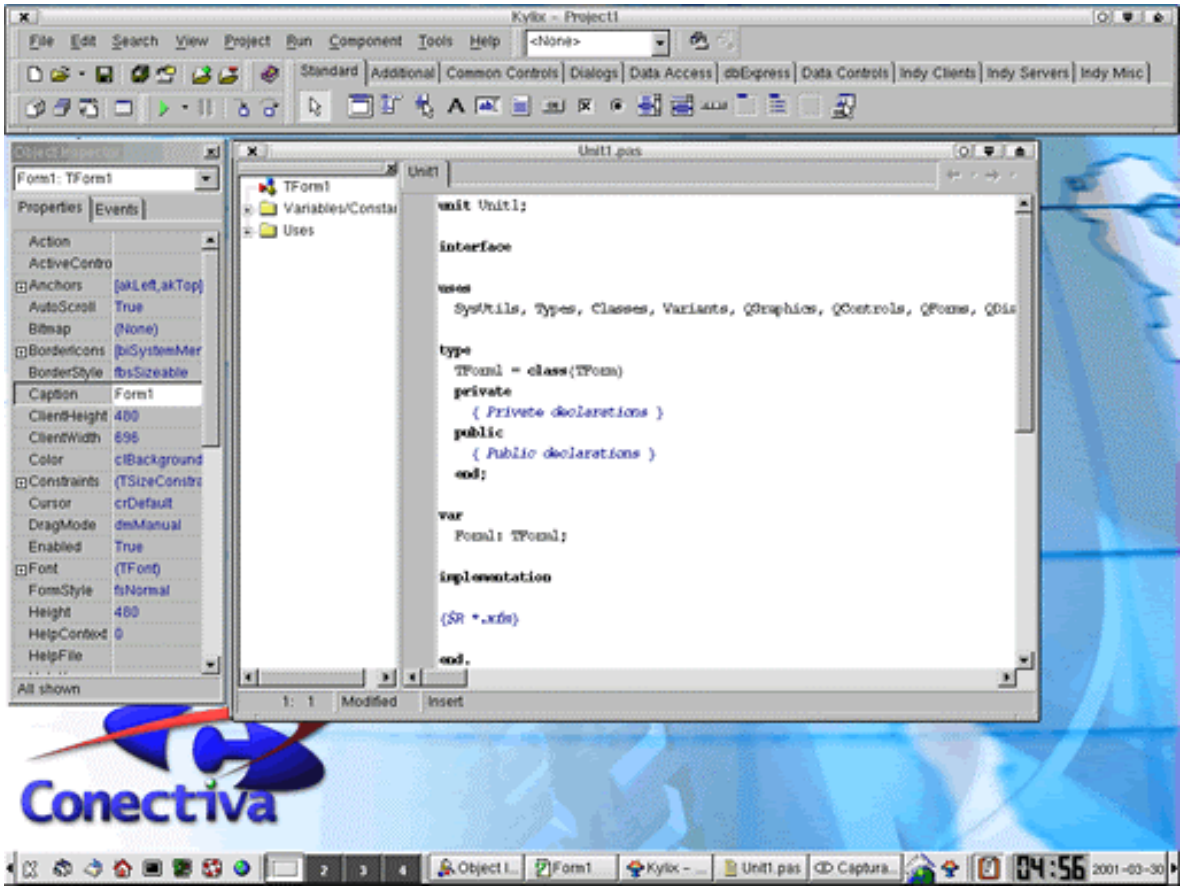
Kylix - O Delphi para Linux

Desde que foi anunciado, em 1999, o Kylix vem gerando uma grande expectativa na comunidade de desenvolvedores. O nome Kylix originalmente era o nome do projeto que portaria o Delphi e o C++ Builder para Linux e transformou-se no nome do novo produto lançado pela Borland para desenvolvimento rápido de aplicações no ambiente Linux.

O Kylix era esperado pela comunidade Delphi pois significava a oportunidade de portar os programas Windows para o Linux com um mínimo de esforço e também pela comunidade Linux, por ser uma ferramenta que facilitaria muito o desenvolvimento em Linux.
Finalmente, em 31 de janeiro deste ano, a Borland lançou o Kylix na LinuxExpo em Nova York. Será que a ferramenta é tudo o que os desenvolvedores esperavam? Será que ela será uma nova Killer Application (aplicação que revoluciona a maneira existente de trabalhar)? Este artigo tentará trazer as respostas a estas questões.

Iniciando a execução

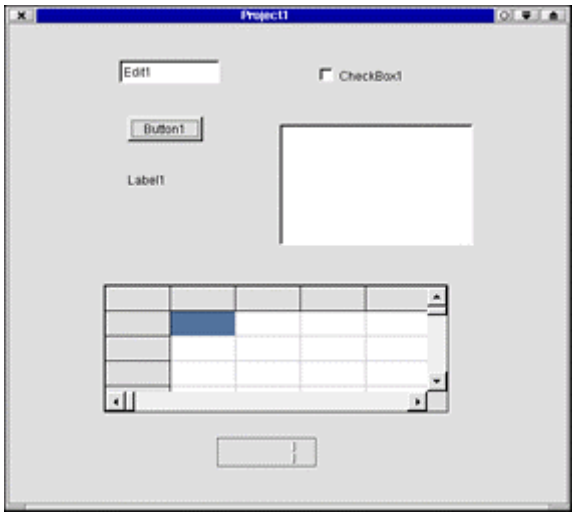
Após instalar o Linux e instalar o Kylix, ao abri-lo, vemos uma tela como o da figura 1.



Tela Kylix

Os desenvolvedores Delphi imediatamente reconhecerão esta tela e verão que ela é muito semelhante à do Delphi 5. Ponto para a Borland: nada de interfaces diferentes entre os ambientes Linux e Windows - os usuários de Delphi não terão problemas para aprender a usar o Kylix. Em seguida, colocamos alguns componentes, configuramos suas propriedades e

teclamos no botão com a seta verde, para compilar e executar o programa. Em poucos segundos, o programa está rodando, como mostra a figura 2.



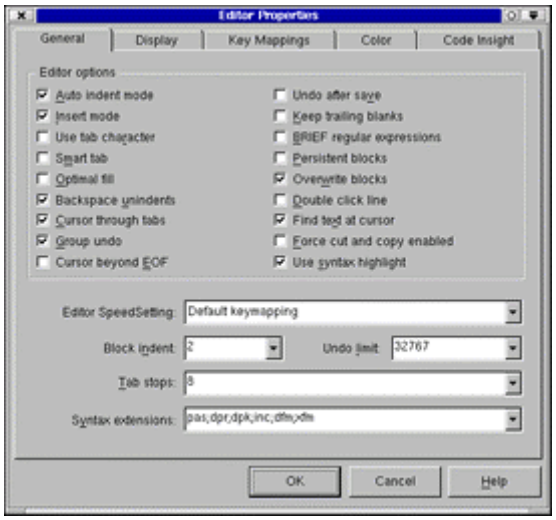
Programa Kylix em Execução

Este é um legitimo programa Linux e como tal, não necessita de nenhum runtime.
Até aqui, tudo se comporta exatamente como no Delphi. Se não tivéssemos com uma interface gráfica um pouco diferente, poderíamos dizer que estamos usando o próprio Delphi.

Passado o primeiro contato, vamos nos aprofundando um pouco no programa e examinando as semelhanças e as diferenças.

A IDE

A IDE do Kylix é muito semelhante àquela do Delphi: os menus são muito parecidos, o Object Inspector está no mesmo lugar, o editor de código e a Form estão lá. As caixas de diálogo e as configurações de personalização são semelhantes.
Como o Delphi 5, o Kylix também tem o recurso de complementação de código em seu editor. Você pode também configurar as teclas do editor usando mapas de teclas personalizados.



Tela de Personalização do Editor

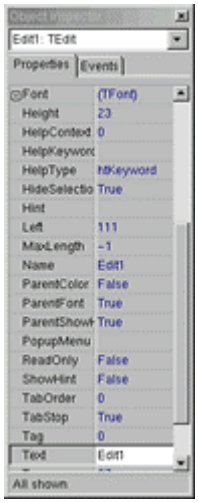
A paleta de componentes é semelhante àquela do Delphi 5. A primeira diferença que salta à vista é a falta da guia Win32 (como era de s esperar). Ela foi substituída pela Common Controls. Foi adicionada uma nova guia, dbExpress, com os novos componentes de acesso a dados. Falaremos sobre isso adiante. Uma ausência notada foi a guia QReport, com os componentes de impressão do QuickReport.

Paleta de Componentes do Kylix

Aprofundando um pouco mais

Aparentemente, a Borland quis deixar o desenvolvedor Delphi à vontade, mantendo a aparência e a maneira de trabalho semelhante. E os programas?

Ao colocar um componente qualquer na Form, vemos que as suas propriedades são muito parecidas as dos componentes Delphi. Os editores destas propriedades também são muito semelhantes. Mais um ponto para a Borland: os programas Windows que usam os componentes padrão do Delphi deverão ser portados facilmente.



Object Inspector com propriedades de um TEdit

Para chegar a isso, a Borland teve de percorrer um longo caminho: a VCL, biblioteca de componentes do Delphi é completamente baseada na API do Windows e seria muito complicado portá-la para o Linux pois, a menos que se use alguma biblioteca de emulação, como a WineLib, as funções da API do Windows não estão disponíveis. A Borland optou por um outro caminho: reescrever completamente a VCL, mantendo o aspecto básico e os componentes da VCL. Para isso, ela usou uma biblioteca de classes em C++, chamada Qt, criada por uma companhia norueguesa, chamada Troll Tech. Ele deu o nome de CLX (pronuncia-se CLIKS) à nova biblioteca de componentes.

Embora costumemos chamar de CLX a esta biblioteca visual de componentes, isto não é correto. A CLX engloba mais que apenas a biblioteca visual. Esta biblioteca de componentes visuais é chamada de VisualCLX. A CLX é o conjunto da VisualCLX, da BaseCLX, classes e rotinas básicas da CLX, da DataCLX, componentes de acesso a bancos de dados e da NetClx, componentes de acesso à Internet.

A biblioteca Qt é uma biblioteca portável, disponível para Linux, Windows, diversos tipo de Unixes, Mac e até tem uma versão disponível para Palm. Com este enfoque, a Borland ganhou

muito no aspecto portabilidade: caso tivesse usado a WineLib, o Delphi encontraria o mesmo problema de conversão se a Borland quisesse desenvolver um Delphi para outro ambiente, como o Mac ou o Solaris. Usando a biblioteca Qt, não é necessário reescrever a CLX para portar para um novo ambiente.

Por uma questão de velocidade de desenvolvimento, a IDE não foi completamente reescrita usando a CLX: a Borland optou por usar a WineLib, recompilando o código em Linux usando esta biblioteca de emulação e reescrevendo apenas algumas porções usando a CLX. Provavelmente, na próxima versão do Kylix, a IDE deverá estar completamente reescrita usando-se a CLX e, portanto, estará pronta para ser portada para outros ambientes. Neste caso, o trabalho deverá ser muito menor.

Embora sua aparência seja muito semelhante à da VCL por fora, internamente, a CLX é completamente diferente: a CLX é completamente independente da API do Windows, dependendo apenas da biblioteca Qt. Por exemplo, o componente TEdit da VCL nada mais era que o encapsulamento com componente Edit do Windows. Na CLX, o componente TEdit encapsula um componente QEdit do Qt. Isto, para o desenvolvedor de aplicações não significa nada: um TEdit continua sendo um TEdit e é usado da mesma maneira, tanto no Delphi quanto no Kylix. Para um desenvolvedor de componentes, esta alteração é significativa: internamente, o QEdit é muito diferente do Edit do Windows e a maneira de modificar o comportamento deste componente não é a mesma.

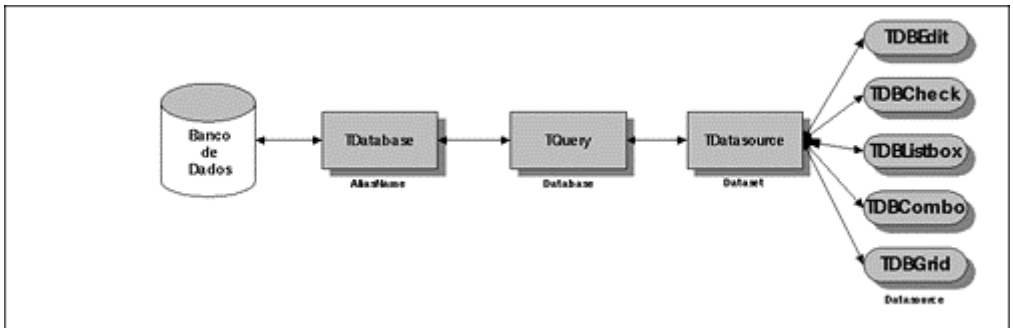
Bancos de dados

A grande diferença entre o Kylix e o Delphi está aqui. A Borland aproveitou a oportunidade do lançamento do Kylix para criar uma nova engine de banco de dados: o dbExpress. O dbExpress é uma engine muito leve, com uma API interna muito simples. Por ser uma camada muito fina, o acesso a banco de dados é muito rápido, muito semelhante ao acesso nativo.

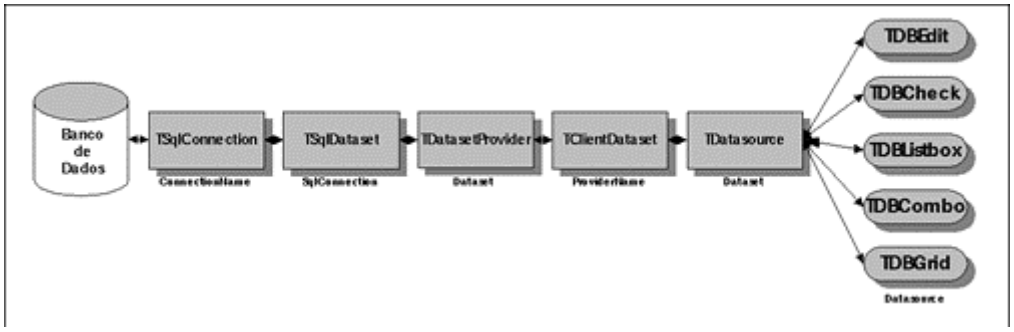
As desvantagens do dbExpress são o acesso unidirecional e o fato dele ser essencialmente voltado a bancos de dados SQL. O acesso unidirecional faz com que você só possa posicionar os registros para a frente, como se estivesse gerando um relatório. Por ser voltado a bancos de dados SQL, não há acesso a dBase ou Paradox. Isto não é de todo uma desvantagem, pois com o Interbase OpenSource, os desenvolvedores têm uma opção de excelente qualidade e baixo custo para substituir esses bancos de dados.

Já o acesso unidirecional é uma dificuldade real, pois estamos acostumados a mostrar os dados no formato de grades, que não poderiam ser usadas, neste caso. Para solucionar isto, a Borland "emprestou" do Midas o componente TClientDataset. Este é um componente muito flexível, que permite o trabalho de diversas maneiras, inclusive independente de bancos de dados.

Embora o TClientDataset introduza uma camada a mais no acesso aos dados, como mostram as figuras 6 e 7, com o TClientDataset, o acesso volta a ser bidirecional: os dados provenientes do banco de dados são lidos e guardados em memória, permitindo a movimentação em todas as direções.



Acesso tradicional a bancos de dados com o Delphi



Acesso a bancos de dados com o TClientDataset

Além disso, o uso do TClientDataset traz diversas vantagens:

·O acesso e modificação aos dados é feito em memória, sendo muito mais rápido. As modificações físicas no banco de dados são feitas em lote.

·TClientDataset permite desfazer alterações, usando inclusive pontos de marcação, onde todas as alterações a partir deste ponto são desfeitas

·Pode-se trabalhar inclusive em modo "briefcase" - desconecta-se do servidor de dados e trabalha-se localmente. Quando houver uma reconexão ao banco de dados, há a reconciliação das alterações

·Pode-se filtrar e ordenar os dados rapidamente na estação cliente, sem necessidade de uma nova consulta

Se você quiser, pode inclusive usar o TClientDataset sem o uso de Bancos de Dados. A isto, a Borland deu o nome de MyBase. Caso você use o MyBase para acesso aos dados, deverá distribuir ao cliente, além do executável, apenas uma dll de aproximadamente 100K. Os dados serão salvos em formato XML, podendo ser lidos por qualquer programa que acesse este tipo de dados.

O dbExpress, sendo essencialmente baseado em servidores SQL não suporta, nesta primeira versão, o Paradox e o dBase. O Kylix é distribuído com uma versão do Interbase, e pode ser também usada a versão Open Source deste banco de dados. A Borland fornece também os drivers para MySql, na versão Desktop e Db/2 e Oracle, na versão Server. Como a interface para o dbExpress é muito simples e já está aberta, com certeza rapidamente aparecerão drivers para outros bancos de dados. Já está inclusive disponível um driver dbExpress para conexão com ODBC, em http://www.easysoft.com.

Interface com o Linux

Da mesma maneira que o Delphi acessa a API do Windows, o Kylix pode acessar a API do Linux. As funções da biblioteca C padrão do Linux estão disponíveis quando se inclui a unit Libc na cláusula Uses. Desta maneira, você pode usar todos os recursos e funções do sistema operacional, permitindo executar outros programas, obter dados do usuário, entre outras coisas.

Caso a função desejada não esteja na unit Libc, você pode criar uma declaração para ela e usá-la, como num programa em Delphi. A única observação que deve ser feita é quanto à portabilidade: as funções da Libc, em geral, não serão portáveis para outros ambientes e deverão ser convertidas, mesmo quando o Delphi 6 estiver disponível. Em outras palavras, se você quiser 100% de compatibilidade com o Delphi 6, deve fugir das funções da API do Linux.

Além do acesso a funções do Linux, o Kylix também permite o acesso às classes do Qt. O Qt é uma biblioteca de classes C++, incompatível com o Kylix. Porém os desenvolvedores da

Borland fizeram um "achatamento" destas classes e colocaram as declarações na unit Qt.pas. Desta maneira, você pode usar tudo o que o Qt oferece, estendendo os componentes do Kylix. A utilização da API Qt tem a vantagem de ser portável: quando o Delphi 6 estiver disponível, estas funções também estarão lá e não precisarão de conversão.

Estilos

O uso da API do Qt trouxe um novo conceito em desenho personalizado de componentes: estilos. O Qt não usa os controles do ambiente em que está, desenhando inteiramente seus próprios componentes. Isto faz que ele possa mudar facilmente o estilo de desenho de um componente ou mesmo de toda a aplicação. O Kylix expõe esta facilidade e permite que você mude o estilo de desenho de toda a aplicação com apenas uma linha de programa: se você usar

```
Application.Style.DefaultStyle := dsMotif;
```

sua aplicação fica com a aparência de uma aplicação Linux. Ao usar

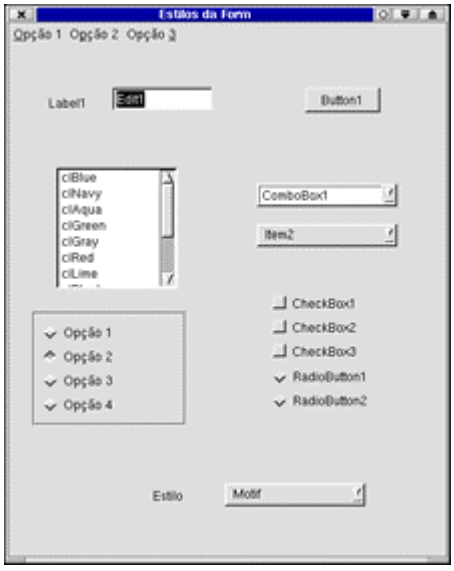
```
Application.Style.DefaultStyle := dsWindows;
```

ela fica com a aparência de uma aplicação Windows. Esta modificação de aparência não é limitada à aparência deste ou daquele sistema operacional.

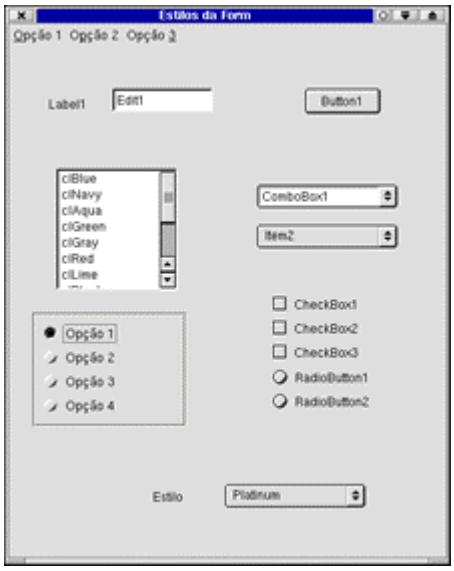
```
Application.Style.DefaultStyle := dsPlatinum;
```

dá à aplicação a aparência não encontrada em outro sistema.

Aplicação com estilo Windows



Aplicação com estilo Motif



Aplicação com estilo Platinum

Além destas modificações, você também pode criar um desenho personalizado para componentes individuais, como botões ou textos.

Botão personalizado

Impressão no Kylix

Nesta primeira versão, o Kylix não vem com um gerador de relatórios, como o QuickReport. Isto não quer dizer que não será possível imprimir com programas Kylix.

Pelo contrário, todo o suporte de impressão está implementado no Kylix, da mesma maneira que no Delphi: o Kylix tem o componente TPrinter, que permite imprimir da mesma maneira que no Delphi. O trecho de programa abaixo desenha um texto na posição 100, 100 da impressora:

```
with Printer do begin
BeginDoc;
try
Canvas.TextOut(100,100, 'Teste de impressão');
finally
EndDoc;
end;
end;
```

Além da impressão com o TPrinter, podemos ainda usar outros métodos, como o AssignPrn, ou mesmo ousar métodos exclusivos do Linux, como abrir a porta da impressora como um arquivo normal e gravar neste arquivo, ou então abrir uma comunicação (pipe) com o spool de impressão e gravar neste duto de dados.

Apesar de não ter um gerador de relatórios incluído no produto, o fato da Borland ter portado toda a interface de impressão para o Kylix é muito promissor: sem dúvida, muito em breve, os principais geradores de relatórios deverão ser portados para o Kylix, estando disponíveis para o usuário.

Portabilidade dos programas Windows

Uma preocupação muito grande a todo desenvolvedor Windows que quer portar suas aplicações para o Linux é a dificuldade que encontrará nesta tarefa. Isto depende do programa que está sendo portado.

Se você tiver programas que não utilizem bancos de dados, API do Windows e só utilizem componentes padrões do Delphi, o trabalho de portar a aplicação será muito suave: provavelmente, com a mudança das cláusulas Uses, o programa poderá ser recompilado no Kylix.
Os nomes das units que contém os componentes foram alteradas, incluindo-se um caractere Q no início do nome. Assim, Forms virou QForms, Controls tornou-se QControls e assim sucessivamente.

Sem dúvida, você terá alguns problemas com características diferentes entre o Linux e o Windows, como a inexistência do conceito de drives ou a mudança de separadores de caminho, por exemplo.

Os programas que utilizam bancos de dados sofrerão alterações mais profundas:

· Paradox e dBase não são mais suportados - o ideal é converter a base de dados para Interbase e usar suas características. Caso a base de dados seja muito pequena, o MyBase é uma alternativa viável.

· Como o dbExpress é baseado em Sql, é interessante converter o uso de Tables para Queries, minimizando o tráfego na rede

· Os componentes TTable, TQuery e TStoredProc devem ser convertidos para TSqlTable, TSqlQuery e TSqlStoredProc. Todo o acesso que não seja unidirecional deve contar ainda com um componente TDatasetProvider e um TClientDataset. Os componentes data aware devem estar ligados ao TClientDataset

· Não existe mais o componente TUpdateSql, nem a figura de CachedUpdates, que devem ser substituídos pelo TClientDataset
Quando seu programa usar componentes de terceiros, se você tiver o fonte, deverá tentar recompilá-lo e convertê-lo para o Kylix. Caso isto não seja possível, deve tentar obter um substituto ou mesmo uma versão compatível com o Kylix, junto ao fabricante do componente.

Se o programa usar características específicas do Windows, como o uso de API, ou componentes ActiveX, automação Ole, podem ocorrer quatro hipóteses:

· Conversão usando a API do Linux - esta é menos recomendável, pois introduzirá incompatiblidades quando você quiser transformar o programa em multiplataforma

· Conversão usando a API do Qt - esta opção é a ideal, pois será completamente portável quando o Delphi 6 estiver disponível

· Substituição das características específicas por outras disponíveis no Linux, como a mudança de comunicação entre programas via COM ou mensagens por TCP/IP ou named pipes

· Eliminação destas características no programa Linux

Uma vez convertido, o programa estará pronto para ser novamente portado para o Delphi 6: a menos que use características específicas do Linux, um programa Kylix poderá ser recompilado sem modificações no Delphi 6. Caso você queira introduzir características específicas, poderá utilizar as variáveis de compilação LINUX e MSWINDOWS, como no exemplo a seguir:

```
{$IFDEF LINUX}
// coloque aqui o código específico Linux
{$ENDIF}
{$IFDEF MSWINDOWS}
// coloque aqui o código específico Windows
{$ENDIF}
```

Conclusões

Sem dúvida, a Borland fez um excelente trabalho ao portar o Delphi para o Linux. Embora os dois sistemas operacionais sejam muito diferentes, ela trouxe a mesma funcionalidade e maneira de trabalho para o Linux, de maneira a não intimidar o desenvolvedor Windows.
Apesar das diferenças entre os dois sistemas operacionais, estimo que o trabalho de portar grande parte das aplicações será relativamente pequeno, além da curva de aprendizado da nova ferramenta ser praticamente zero, para usuários do Delphi.

Para os desenvolvedores Linux, o Kylix, sem dúvida, traz a possibilidade de desenvolver aplicações robustas, em menos tempo e com menor custo (aqui não estamos falando no custo da ferramenta: obviamente, não se pode competir com o custo do compilador gcc, mas sim do número de homens/hora necessários para o desenvolvimento da aplicação).

Acredito que o Kylix é a ferramenta que faltava ao Linux: embora o Linux tenha inúmeras ferramentas, e um número muito grande de desenvolvedores, não era um ambiente muito propício para o desenvolvedor médio de aplicações comerciais. Com o Kylix, este desenvolvedor passa a ter uma excelente ferramenta para portar suas aplicações e criar novas, específicas para o Linux, impulsionando seu uso.