



# FAQ

frequently asked questions

ISSN 1677-6917



www.digerati.com.br

Ano 1 - Número 2 - R\$ 9,90

**AS DICAS DE**

# Delphi

que você tanto procurava

**NO CD:** programas e tutoriais

**NA REVISTA:** manual para começar a programar

**Nesta edição...**

## Open Kylix 3

Programa completo para programar no Linux

## Banco de Dados

Componentes e códigos para SQL no Delphi

## Orientação a Objetos

Os conceitos básicos da lógica que está mudando a programação

## Object Pascal

Minitutorial sobre a linguagem que é a base do Delphi

## Prática total

Monte já seu primeiro programa em Delphi, sem complicação

**+ de 50** tutoriais sobre programação, banco de dados, Kylix

**+ de 600** códigos-fonte para múltiplas aplicações

**Tudo para incrementar sua programação em Delphi**

**E muito mais...**



# FAQ #2

frequently asked questions

## Ferramentas e Utilitários

Destaque para o Kylix 3, versão aberta do Delphi para Linux

Kylix 3, Open Edition	Delphi Repository Editor v.1.0
CePeeBeDee v.2.00	Delphi Super Recall v.0.99b beta
Code Keeper v.1.20	TAG's PatternMaid v.1.0
Delphi Component Collector v.1.0	Beginner Software's Delphi Tools v.1.02
Delphi Developer Toolbar	BuyPin Header Translator v.1.1

## SQL

As melhores ferramentas para você gerenciar informações em seu banco de dados

Advanced Query Tool 4.2.3	MySQLTotal
Advanced SQL Code Collection 1.2.0	SQL Connect 3.0
CASE Studio 2.6.1	SQL Generator 2.0
Database Tour 3.2	SQL Tester 2.0
EMS MySQL Manager 1.8.5	SQLExec 3.1
MySQL 3.23	SQLScriptWriter 2.01
MySQL Backup Watcher 1.4.3	SQLyog 1.71
MySQL Server Manager 1.1	Trouble Checker 2.1

## Source Codes

Mais de 600 códigos para diversos tipos de aplicação

## Pascal

Compiladores, tutoriais, conversores, e muito mais sobre a linguagem que deu origem ao Delphi

Turbo Pascal Reference	Runtime Libraries Update
Borland Turbo Pascal	PasP4
CtoP Version 1.2b	Pascal Pro Compiler 0.01
End of Earth	Pascal Programming Tutorial 1.0
Free Pascal Compiler (Windows)	J.R. Ferguson Pascal Library
Free Pascal Compiler (Linux)	Manipulação de arquivos texto
Irie Pascal 2.0	TMT Pascal Lite 3.90

## Na Revista

Mais tutoriais e informações, incluindo a história do Delphi, guia de sites, conceitos básicos, programação na prática e dicas para usar o Delphi no seu dia-a-dia

## Componentes

Incremente seus projetos em Delphi para a Internet, crie novos menus, botões e barras, acrescente elementos de segurança, etc.

> <b>Internet</b>	AddRotator v.2.0
Active Ports component v.0.9.4	AHM E-Mail Triton v.2000
ActiveIE v.1.0	ASP File Upload Component v.1.0
> <b>Dialogs</b>	TDialog97 v.1.0
CoolDBDialogs v.1.01	TSQLConnect
MiTeC Database Connection Dialog v.1.0	1st Dialogs Library v.1.0
> <b>Barras</b>	ARMenu ToolBar v.1.0
Colorbtn v.1.0	Delphi Fundamentals v.3.00
AbAction Builder v.3.0	Dockable Toolbar v.3.0
> <b>Botões</b>	BtrDisabledHint v.1.0
Animated SpeedButton v.1.0	ButtonWithColor v.1.0
BS Morph Button v.1.00	CorelButton v.1.0
> <b>Menus</b>	BarMenu Components v.2.3.3
AdvMenus v.2.0	BtRichPopup
AlignPopupMenu v.1.1	CoolMenus v.3.07.2
> <b>Relatórios</b>	TEssHtmlReport v.1.01
HTMLReport v.3.1	ARWordReport v.1.0
Purepage HTML Conversion Engine v.3.0	OrangeReport v.1.5
> <b>Segurança</b>	Adept Shareware Defender v.1.3
1Way v.2.0	AecRijndael v.1.16
AccuLock v.1.1	AecTwoFish v.1.00
> <b>Som</b>	BTBeeperv.0.06
Beeper	Mixer Volume
Bleepint v.5.5	Sound / Nosound v.1.0

## Apostilas e Tutoriais

Mais de 50 tutoriais e apostilas incluindo: Delphi, Banco de Dados, SQL-Oracle, Programação, ASP, SMTP, Kylix e muito mais

Delphi FAQ	Criando Objetos ASP
Delphi Básico	Delphi 5
Apostilas e Tutoriais	Delphi C/S
Dicas de Delphi	SMTP e-Mail
Super Dicas	Exceções
Instruções Básicas	API Guide
Curso de Delphi Completo	Help & Menus
Banco de Dados Básico	Roteiro de Uso do InstallShield
Conceitos Gerais de Banco de Dados	Jogo da Velha em Delphi
Apostila SQL-Oracle	Mapa do Brasil
Construção de Banco de Dados	Apostila SQL Server
Breve Introdução à Programação	Arquivos de Ajuda do RX Lib
Banco de Dados Delphi	Creating WAP Applications with Borland Delphi
Criação de Componentes	Migrating from Visual Basic to Delphi

### Equipamento mínimo:

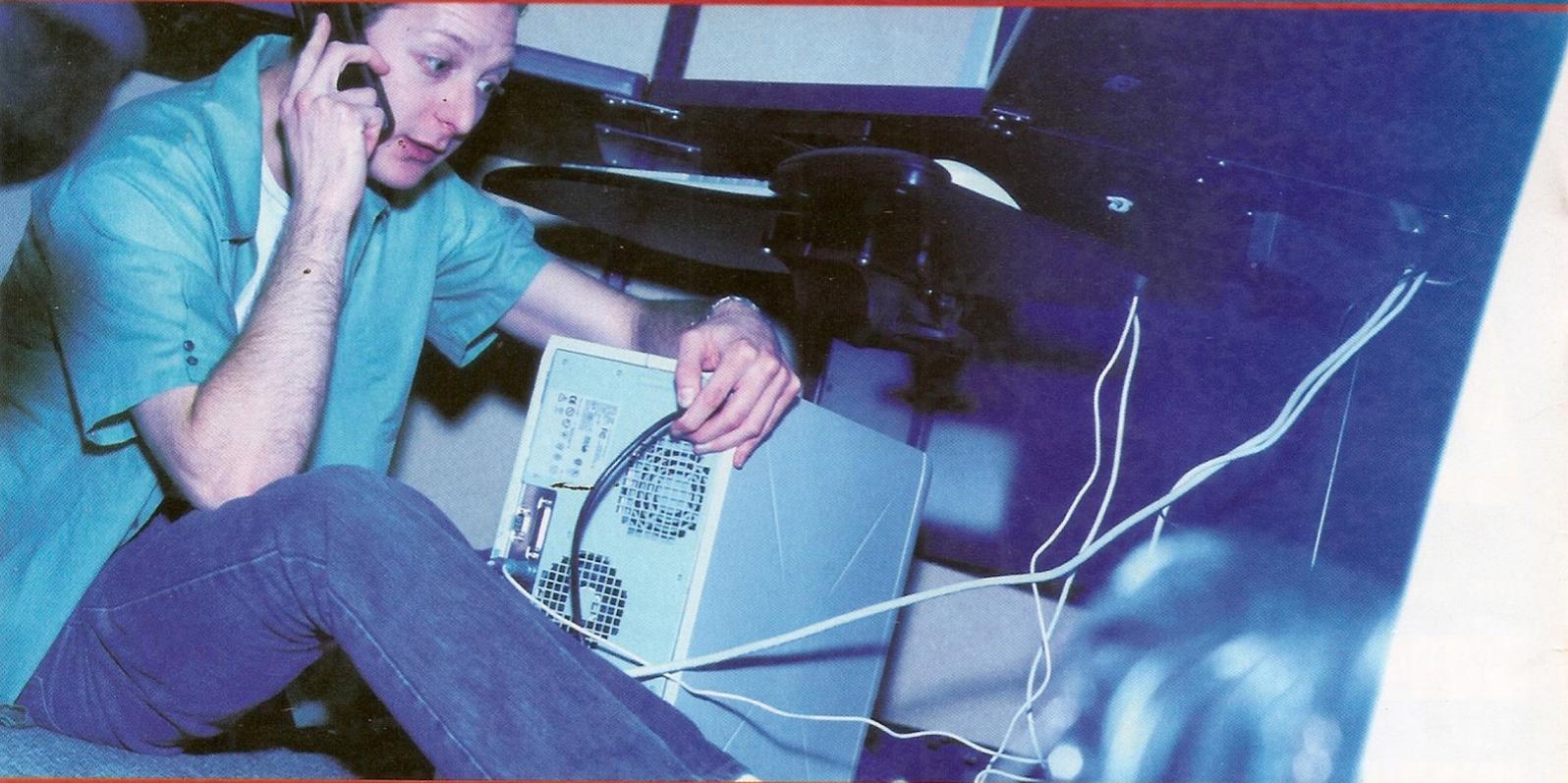
Processador Pentium II ou equivalente. 64 MB de memória RAM; 16 MB de vídeo, em resolução 800 x 600 pixels. Placa de som.

### Informações complementares:

O CD-ROM brinde é composto por uma coletânea de software freeware, shareware e demonstração.

Os requisitos de sistema podem variar de programa para programa. Alguns aplicativos, por motivos alheios a nossa vontade, podem não rodar no Windows XP.

e muito mais...



Leve sempre um disco de boot com você.

741796-9874

Revista Especial **Hardware**  
 Revista mais CD-ROM por R\$ 9,90  
 nas bancas ou no site [www.digerati.com](http://www.digerati.com)



[www.digerati.com](http://www.digerati.com)



# FAQ

frequently asked questions

**E**ntre todas as empresas de programação, a Borland é uma das mais inteligentes. Percebendo o crescimento da plataforma Linux, eles lançaram o Kylix, uma versão do Delphi, e um dos primeiros softwares de programação visual feito para este sistema operacional. Ou seja, ao contrário de ficar brigando com o novo, a Borland resolveu aproveitar suas oportunidades e ganhar dinheiro (que, aliás, é o objetivo de toda empresa, certo?).

Nesta nova edição da **FAQ**, dedicamos as nossas páginas para quem está começando a desenvolver usando o Delphi - prometemos dar uma boa olhada no Kylix em alguma edição no futuro.

Como não poderia deixar de ser, selecionamos os melhores sites para você aprender e aprimorar seus conhecimentos da ferramenta. Claro que o melhor é começar pelo site da própria Borland, para ver as mudanças no Delphi 7, lançado recentemente.

Seguimos por uma pequena história do Delphi que começa lá trás com a linguagem Pascal. Com o surgimento da lógica orientada a objetos, o Pascal avançou para o Object Pascal que continua sendo a base para o Delphi.

É por isso que colocamos uma pequena introdução ao Object Pascal, para você, que não conhece, tomar os primeiros contatos com a linguagem. Depois, damos um exemplo prático do funcionamento do Delphi com a construção de uma pequena calculadora. Com isso, dá para ter uma boa visão geral sobre os objetos e suas funções.

Mas deixamos para o final a melhor parte da revista: uma impressionante coleção das melhores dicas para quem programa em Delphi. São 60 dicas simples, mas que significarão muito na construção de seu aplicativo. Leia, treine e mantenha a **FAQ** sempre à mão.

Aliás, este é o objetivo da **FAQ**, funcionar como uma fonte de referência. Para isso, é importante que você, leitor, ajude a construir a revista, mande suas sugestões, críticas e temas para a próxima revista.

Boa leitura e boa programação.

## **06 - Guia de sites**

- Seleccionamos os melhores sites para você acompanhar as novidades, baixar tutoriais e encontrar componentes já prontos

## **10 - História**

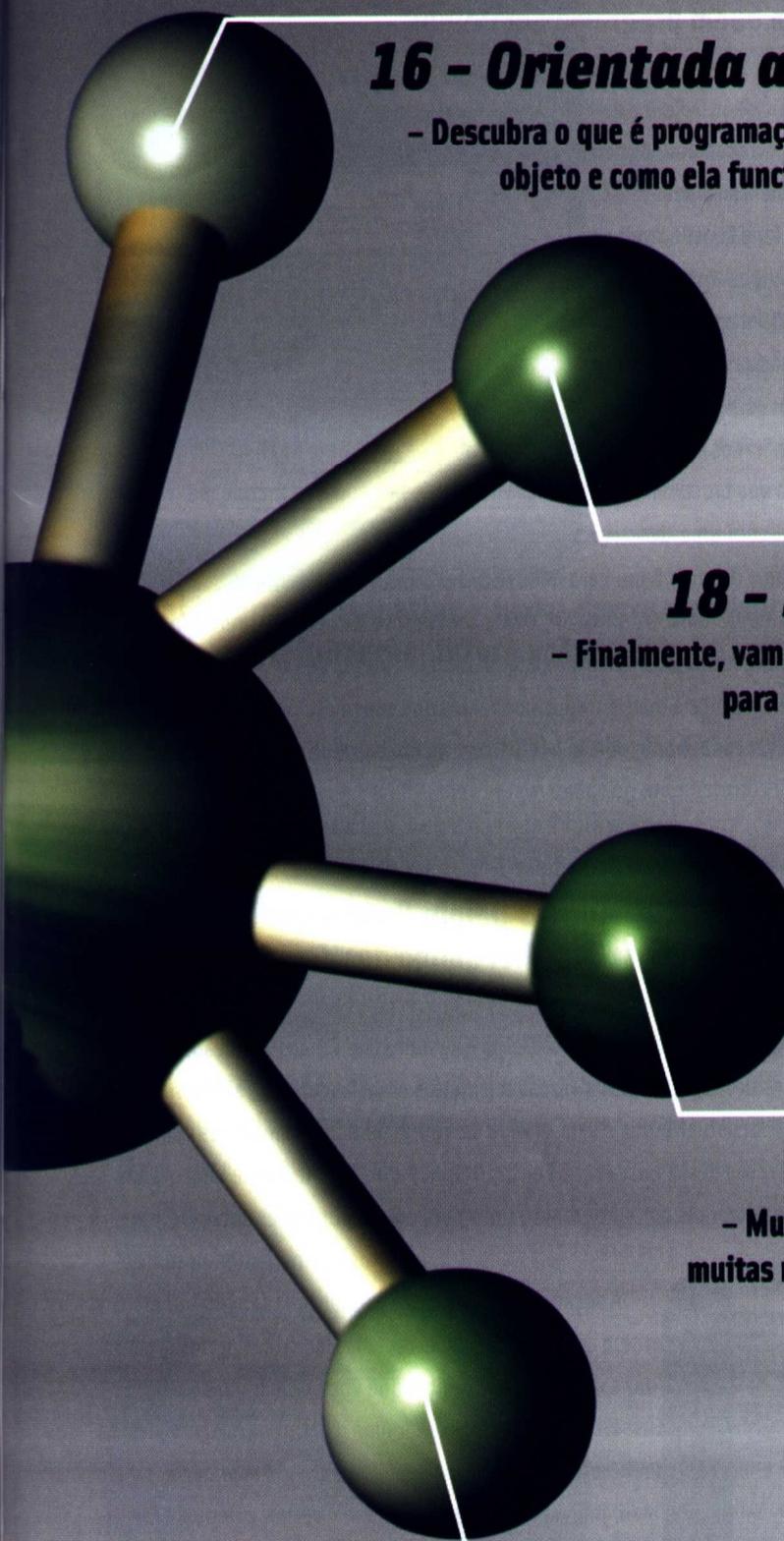
- De onde veio, para onde vai. Uma história sucinta do Delphi

## **11 - Livros**

- Escolhemos três importantes livros sobre Delphi 6, enquanto não sai nada sobre a versão 7

## **12 - Object Pascal Básico**

- O Object Pascal é a base do Delphi. Veja um pequeno tutorial sobre esta linguagem



## **16 - Orientada a Objeto**

- Descubra o que é programação orientada a objeto e como ela funciona no Delphi

## **18 - Delphi Prático**

- Finalmente, vamos construir um programinha para ver como funciona na prática

## **24 - Dicas**

- Muitas dicas, muitas dicas. Pegue muitas manhas de código que não está em nenhuma outra revista

## **42 - Guia do CD**

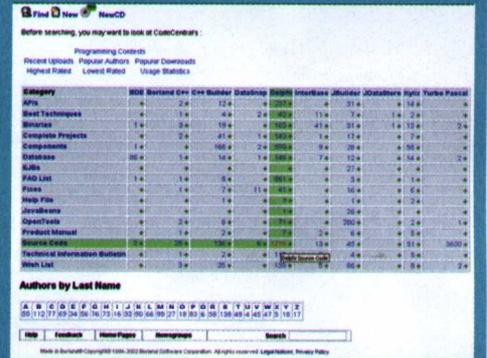
- Guiamos você por todos os cantos do CD desta edição

Por Juliano Barreto  
juliano@digerati.com.br

## Borland CodeCentral - <http://codecentral.borland.com> Parada obrigatória para quem programa em Delphi

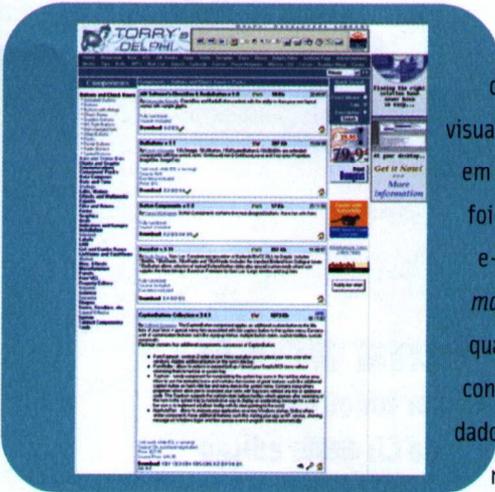
No site da desenvolvedora oficial do Delphi e do Kylix – a Borland –, existe um espaço reservado para a comunidade de programadores, é o chamado “Borland Developer Network” ou simplesmente BDN.

É lá que você encontrará o Borland CodeCentral, que traz uma grande variedade de códigos-fonte, componentes, manuais de produtos, FAQs e aplicações prontas para desenvolvedores avançados e iniciantes em Delphi, Kylix, C++ Builder e Java. Os códigos são os grandes destaques do site e estão separados em várias categorias para que você encontre o que procura mais rapidamente. Cada um deles está com uma descrição superdetalhada que informa quem o desenvolveu, para que ele serve e em qual plataforma ele funciona melhor. Para não se perder no meio de tanta informação, o site também conta com uma ferramenta de busca, onde você procura por aplicações de acordo com a data, o autor, a categoria, etc. Apesar de o site ser em inglês, vale a pena fazer um esforço a mais e traduzir os textos que são muito bons. Para acessar o conteúdo deste site é necessário ser cadastrado, porém o registro é rápido e gratuito (não se esqueça de colocar o endereço de seu e-mail corretamente no cadastro, ele será útil no futuro).



## Torry's Delphi Page - [www.torry.net](http://www.torry.net) O melhor amigo dos desenvolvedores

Para selecionar as centenas de componentes para Delphi presentes no CD desta edição, muitos sites foram visitados, alguns deles eram apenas razoável e muitos eram fraquinhos, a maioria com componentes “shareware”, desatualizados e sem grande serventia para programadores comuns. Porém, uma das melhores surpresas dessa busca foi o Torry's Delphi Page. No site existe uma grande lista com aplicações prontas, ferramentas para programar, links para outros bons sites sobre Delphi e muitos componentes gratuitos!



A seção de componentes é atualizada semanalmente e contém as aquisições mais recentes do site. Portanto, para visualizar todos os componentes que você procura, é bom clicar em *Last Week* e depois escolher uma das categorias. A divisão foi muito bem-feita. Se você está procurando um engine para e-mail, por exemplo, é só entrar em *Internet* e depois em *e-mails*. Mas não fica só nisso, existem componentes para criar qualquer software que você puder imaginar, desde um simples contador de cliques até um avançado sistema de encriptação de dados on-line. O mascote do site é um simpático cão e depois de navegar em uma das suas generosas seções é fácil concluir o porquê. O Torry's Delphi Page realmente é o melhor amigo de quem precisa de recursos para Delphi.

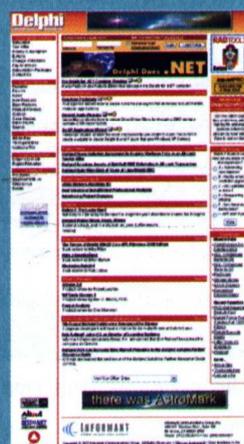
### **iMasters - [www.imasters.com.br/web/colunistas/delphi/](http://www.imasters.com.br/web/colunistas/delphi/) Os mestres da programação à sua disposição**



O que você acha de ter um professor especialista em Delphi, SQL e ASP explicando com muita paciência tudo aquilo que você perguntar, e ainda não precisar pagar nada para ter essas aulas? Os iMasters estão aí para isso. No site você conta com um time de especialistas formado por professores e profissionais da área de programação que escrevem ótimas colunas e tiram as dúvidas de programadores iniciantes e desenvolvedores avançados. A coluna de Delphi, escrita por Marcelo Varela, é uma das mais completas do site. Nela, Marcelo aborda diversos pontos da linguagem, que muitas vezes não são encontrados na Internet, nem mesmo em cursos de escolas de informática. Mas nem só de Delphi vivem os iMasters. É possível encontrar tutoriais e dicas sobre várias linguagens de programação, como Visual Basic, ASP, SQL, C#, PHP e muito mais, todas com muita qualidade. Na seção de downloads você encontrará uma boa seleção de softwares e ferramentas para ajudá-lo a programar. Para melhorar, tudo está em português e o site tem um design bonito e fácil de navegar. Os mestres estão à sua inteira disposição. Agora você não tem desculpa para não aprender.

### **Delphi Informant Magazine - [www.delphizine.com](http://www.delphizine.com) Informação, informação, informação e mais informação**

O site da revista *Delphi Informant Magazine*, o DelphiZine.com, é uma visita obrigatória para programadores Delphi. Com notícias, reviews de livros e produtos, downloads e um excelente fórum de discussão, o portal esbanja quantidade e qualidade de informação e lá é praticamente impossível não encontrar aquilo que você procura quando o assunto é Delphi. O fórum é perfeito para qualquer nível de programador, do iniciante ao profissional, e é muito bem dividido por assuntos, entre eles: Databases (Banco de Dados), Components, Kylix, e Third-party Tools (aqueles programas que ajudam você a incrementar seu projeto em Delphi de um jeito mais rápido e fácil, sem precisar programar nem uma linha sequer). Infelizmente, algumas partes do site são exclusivas para assinantes da revista, (como a seção para download de artigos), e o preço para assinatura é bem salgado, algo em torno de US\$ 49,99 dólares por ano. Se você tem dólares sobrando para comprar a revista, acessar o site é a melhor opção. Além de ser muito mais barato e rápido, você não precisa esperar a revista chegar até o Brasil para ter as notícias mais quentes do mundo dos programadores Delphi.



**FAQ**  
**07**

### **ClubeDelphi - [www.clubedelphi.com.br](http://www.clubedelphi.com.br) Entre para o Clube**



O site da revista nacional *ClubeDelphi*, traz tudo aquilo que um viciado em Delphi precisa, e o que não precisa também. Das coisas legais, podemos destacar os cursos e apostilas, o vídeo que ensina a programar uma aplicação para a Internet e as diversas seções para ajudar iniciantes, como a "Primeiros Passos" e "Dicas e Macetes", além das promoções que sorteiam livros e cursos. Já das coisas dispensáveis, encontramos produtos à venda como camisetas e bonés, além dos insistentes comerciais do programa de TV "ClubeDelphi TV". No site também é possível encontrar um lugar perto da sua casa onde você poderá comprar seu exemplar da *ClubeDelphi*. A distribuição da revista melhorou bastante e está chegando a uma série de cidades do interior do Brasil que antes não chegava. Apesar de certos exageros de adoração ao Delphi, o site é bem completo e serve como referência para programadores brasileiros com muitas dicas legais. Entrar para o Clube não custa nada e você já está convidado.

## Mundo Delphi - [www.mundodelphi.com.br/](http://www.mundodelphi.com.br/) Bem-vindo ao mundo do Delphi

A home page da antiga revista *Mundo Delphi*, que atualmente se uniu com a *ClubeDelphi*, tem um design sóbrio e bonito que a faz parecer um site gringo. Mas para nosso alívio ele é completamente em português. Realmente seria uma pena não poder ler os ótimos artigos e dicas disponíveis no site. O Mundo Delphi possui algumas seções diferenciadas como a "Personalidades", lá você pode ler minibiografias de figuras importantes na história do Delphi, como a do vice-presidente da Borland, David Intersimone, e outros desenvolvedores de renome. O site também oferece um "Treinamento on-line" que apesar de abordar apenas temas bem básicos é ótimo para aqueles que querem dar os primeiros passos no Delphi. Outro curso muito interessante que está dentro do treinamento é o que ensina a instalar componentes. A newsletter mantida pelo site é obrigatória para quem quer se manter informado sobre todas as novidades do mundo dos desenvolvedores Delphi.



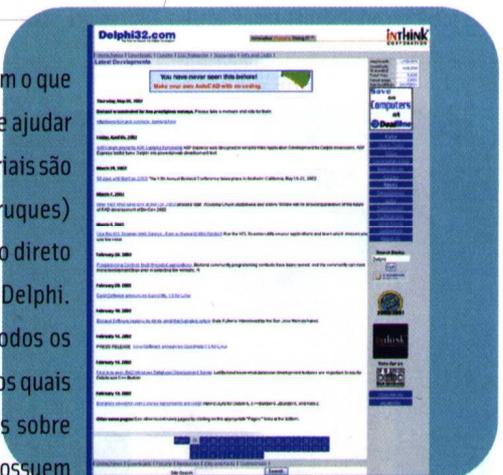
## .:Canal Delphi.: - [www.canaldelphi.com.br](http://www.canaldelphi.com.br) Um portal de Delphi, feito em Delphi!

Segundo os webmasters do "Canal Delphi", o site é "o primeiro portal sobre Delphi totalmente escrito em Delphi do mundo". Como eles criaram um site 100% em Delphi eu nem imagino, mas da qualidade do portal não podemos duvidar. Com muitas atualizações e artigos bem escritos, os assuntos abordados são recomendados para programadores intermediários ou avançados, pois requerem um bom conhecimento dos comandos e estruturas do Delphi. Um destaque positivo dos tutoriais, é que você pode postar comentários diretamente no site e questionar o autor sobre o funcionamento do código ou do comando que ele está explicando.

Por todos os códigos e comentários estarem em português, tudo fica mais fácil, inclusive conversar com os autores. O sistema de busca usado pelo site funciona bem e ajuda bastante quando você quer encontrar um artigo específico. Para se associar ao site não é preciso pagar nada, e o registro é rápido e fácil.

## Delphi32 Online Magazine - [www.delphi32.com](http://www.delphi32.com) Tutoriais, dicas e muito mais

Livros de programação custam uma fortuna, são muito pesados e praticamente escondem o que você quer saber no meio de suas milhares de páginas. Por isso, um bom site de tutoriais pode ajudar muito quando aquela dúvida chata aparece. No site Delphi32.com (que é em inglês) os tutoriais são distribuídos de maneira farta, em várias seções. Em "Tips n Tricks" (algo como Dicas e Truques) você aprende assuntos específicos de maneira super-rápida, pois os artigos desta seção vão direto ao ponto sem muitos rodeios, até por isso eles exigem algum conhecimento prévio sobre Delphi. Já na seção de "FAQs" você tem a oportunidade de ter suas perguntas respondidas com todos os detalhes e de uma forma organizada (bem diferente dos arquivos de texto sem formatação aos quais já estamos acostumados). O site ainda dispõe de uma lista com os melhores newsgroups sobre Delphi, que a própria Borland oferece aos desenvolvedores, elas são bem completas e possuem milhões de pessoas postando mensagens todos os dias. Para quem gosta de trocar idéias sobre projetos e arquivos PAS é o lugar ideal. No Delphi32.com você ainda encontra seções com listas de bugs do Delphi, links para lojas virtuais de livros, artigos e muito mais. Se você contasse quantas páginas de conhecimento existem neste site, iria ver que é bem mais completo que um livrão.



## Active Delphi - [www.activedelphi.com.br](http://www.activedelphi.com.br) Desenvolvedores ativos

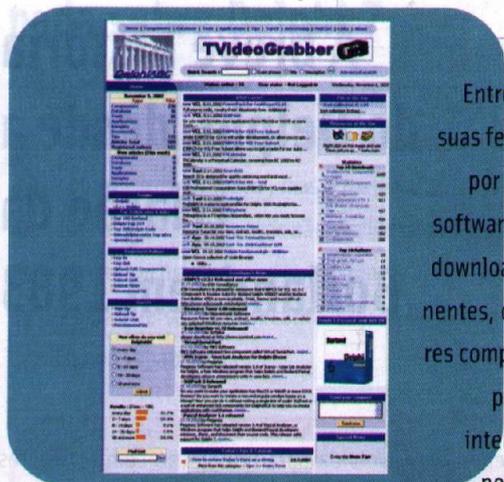
Fazer um curso ou ler alguns livros sobre programação em Delphi, não é o suficiente para se tornar um bom profissional na área. Basta checar algum site de notícias sobre programação que você vai notar que linguagens novas aparecem a cada mês e as tendências vão e vem tão rápido que os mais desatentos nem as percebem.

Por isso, é essencial estar ligado nas novidades sobre a plataforma em que você quer se especializar e sempre manter contato com o pessoal que a desenvolve. Uma das formas para fazer isso é se cadastrar gratuitamente na ActiveNews, a newsletter semanal mantida pelo site nacional, Active Delphi. A lista envia para o seu e-mail notícias, tutoriais e artigos sobre desenvolvimento de aplicações em Delphi, SQL e muito mais. Se você quiser também, é possível fazer download das mensagens antigas e conferir os assuntos que já foram abordados. O fórum do site é muito bom e receptivo, ele tem usuários com um nível técnico avançado que atendem com boa vontade às dúvidas dos menos experientes, com um número alto de posts por dia. O Active Delphi ainda oferece apostilas, componentes e dicas para download.



## DelphiABC - [www.delphiabc.com](http://www.delphiabc.com) Novidades sobre Delphi

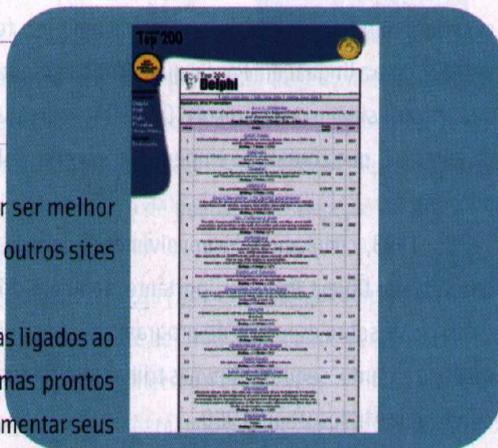
Entre os sites "não-oficiais", o "DelphiABC.com" é um dos mais completos sobre o Delphi e suas ferramentas. Com uma seção de notícias que é atualizada quase que diariamente você fica por dentro das últimas novidades e lançamentos do mercado voltado a desenvolvedores de software. Existe também uma extensa coleção de componentes, ferramentas e aplicativos para download, além de artigos e dicas sobre assuntos específicos, como Bancos de Dados, componentes, conversão de arquivos, strings e muito mais. O site mantém um "Top 10" com os melhores componentes para Delphi, e o melhor é que a maioria deles são Freeware e tem links diretos para serem baixados. Os três fóruns (sobre Delphi, C++ Builder e Kylix) também são bem interessantes. Apesar de não terem muitos usuários postando, eles são bem completos e é possível encontrar respostas para aquelas dúvidas que nenhum livro consegue dar. O site oferece uma série de vantagens para usuários cadastrados, entre elas enviar artigos, códigos-fonte, componentes e tutoriais para o site. Assim, você pode divulgar seus códigos e melhorá-los com a ajuda dos outros usuários do site.



## Jazar Top 200 - [www.top200.jazarsoft.com/delphi/](http://www.top200.jazarsoft.com/delphi/) Os 200 mais

Para finalizar este guia de sites, vamos indicar um site que vale por duzentos. Não por ser melhor que todos os outros já descritos aqui, mas por que ele é o caminho para que você encontre outros sites tão bons quanto os que acabamos de mostrar.

O ranking feito pela JazarSoft traz uma superseleção com sites que tratam de vários temas ligados ao Delphi como grandes depositários de códigos-fontes, onde você encontrará vários programas prontos para editar, componentes gratuitos para incrementar seus projetos, tutoriais on-line para aumentar seus conhecimentos em linguagem Delphi, notícias quentinhas sobre o mundo da programação e muito mais. No mesmo portal você pode acessar outros rankings separados por assuntos que interessam os programadores Delphi, como os melhores sites de Kylix e de PHP.



# Um ótimo presente de grego

**A breve história do Delphi, desde a criação da Borland, em 1983**

FAQ  
10

**A**ntes de entrar a fundo no Delphi, você, que está iniciando neste assunto, deve estar se perguntando: o que é essa linguagem e de onde ela vem?

Delphi é uma linguagem desenvolvida pela Borland, uma empresa da Califórnia fundada em 1983 por Philippe Kahn, um matemático francês. O primeiro programa de sucesso da empresa foi o Turbo Pascal, uma IDE para a linguagem Pascal que, de tão bem sucedida, conseguiu desbancar a similar da Microsoft. O sucesso marcou a área de atuação da empresa, que continuou empenhada em criar soluções para a linguagem.

Em 1993, começou o desenvolvimento de uma nova linguagem, baseada em Object Pascal e, portanto, também orientada a objetos (saiba mais sobre esse tipo de programação na página 16). A linguagem, no entanto, seguiria algumas trilhas instituídas pela Visual Basic, criada pela Microsoft em 1990.

O nome escolhido, Delphi, era, originalmente, um nome-código para a versão beta, e tinha um significado oculto, ligado ao fato de a

solução estar sendo desenvolvida para integração com servidores Oracle. Quando se quer falar com o "oráculo" (Oracle em português), para onde se vai? Delphi (referência à localização do famoso oráculo da Grécia Antiga)! O nome oficial era para ser AppBuilder, mas todos concordavam em que não era algo muito inspirado. A sorte foi que a Novell lançou, pouco tempo antes, o Visual AppBuilder, o que forçou a mudança de nome.

Delphi é, hoje, um programa para desenvolvimento e uma linguagem muito bem-sucedida, especialmente na área de criação de aplicativos para empresas em ambiente Windows. O arquiteto chefe por trás desta empreitada, assim como a do Turbo Pascal, foi Anders Hejlsberg, até a sua saída para a Microsoft, que se deu em 1996.

O Delphi também é conhecido pelo fato de ter inspirado algumas versões de sucesso, como a C++Builder, para o desenvolvimento de aplicativos em C++, e o Kylix, para ambiente Linux.

Bom, agora chega de história e vamos ver, na próxima página, quais são as principais fontes de consulta técnica quando o assunto é Delphi.

# A estante da FAQ

## Editoras ainda não publicaram nada sobre Delphi 7

Infelizmente, no mercado editorial brasileiro ainda não existe nada sobre o novo Delphi 7, mas o material sobre a versão anterior é extenso. E as diferenças não são tão grandes assim. Portanto, dá para aprender com os livros que listamos abaixo e migrar tranquilamente para a versão mais recente. Veja as dicas de livros sobre o tema que listamos abaixo:

### Kylix - Delphi para Linux

**Autor:** Marcos dos Santos Ferreira

**Editora:** Érica

**Preço:** R\$ 61

O Kylix é a versão para Linux do programa da Borland. Apesar dessa revista ser dedicada ao Delphi, não poderíamos deixar de citar o primo dele. A Borland, ao contrário de outras empresas, não voltou às costas ao crescente mercado do sistema operacional do pingüim. Este livro é direcionado para programadores iniciantes ou para aqueles que estão em transição entre o Windows e o Linux. O livro explora a possibilidade de migrar aplicativos e a facilidade de usar a programação visual. Também, para os que estão começando em Linux, há uma introdução ao SO.



### Delphi 5.0 e 6.0 (Básico e Intermediário)

**Autor:** Israel de Campos Avillano

**Editora:** Ciência Moderna

**Preço:** R\$ 29,70

O livro, bastante completo, inclui um CD-Rom com os códigos-fonte dos exemplos mostrados no livro. Iniciando com uma explicação sobre o que Programação Orientada a Objetos, um resumo de algoritmos e de Object Pascal. Depois desta introdução, passamos ao Delphi propriamente dito. Uma boa parte do livro é voltada para quem está realmente começando e chega até um nível intermediário. Ou seja, se você já tem um conhecimento razoável e procura aprofundamento, este livro não foi feito para você. Mas, para quem está realmente começando, é uma ótima pedida.



### Delphi 6

**Autor:** Maurício B. Longo

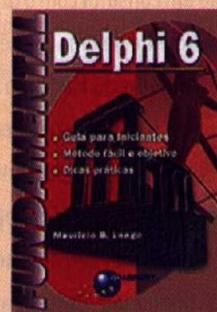
**Editora:** Brasport

**Preço:** R\$ 24

Este livro apresenta uma visão completa do Delphi, além de desenvolver toda a relação da linguagem com os bancos de dados - um dos usos mais importantes para quem está desenvolvendo para Delphi. Os seguintes temas são desenvolvidos ao longo do livro, com clareza e profundidade: Object Pascal, desenho de interfaces, acesso a banco de dados, apresentação de dados e construção de DLLs.

O livro é parte da coleção Fundamental, que inicia em um nível básico, mas segue até assuntos abordados por profissionais e pessoas que já conhecem a linguagem.

A Brasport já promete para logo o primeiro livro abordando o Delphi 7. É esperar para ver.



# Object Pascal

Um pequeno tutorial sobre a linguagem que originou e é a base do Delphi

FAQ  
12

**P**ara facilitar o uso do Delphi, é bom conhecer a linguagem Pascal. Aliás, o Delphi é uma derivação do Object Pascal que, por sua vez, é a versão orientada a objetos do Pascal. Conhecê-la é importante para os programadores. Vamos dar uma olhada em suas características principais, então.

O Object Pascal é uma extensão do Pascal que foi desenvolvida pela Apple em consulta com o próprio inventor da linguagem, Niklaus Wirth. Ela descende de uma primeira tentativa de construir uma versão orientada a objetos do Pascal, o Clascal. Existem poucas diferenças sintáticas entre o Pascal e o Object Pascal.

Neste artigo, apresentamos alguns conceitos básicos do Object Pascal, pois como ele é a base de toda a programação Delphi, isso ajudará a melhorar seu aprendizado.

A linguagem Object Pascal contém todos os conceitos da orientação a objetos em geral, como encapsulamento, herança e polimorfismo (veja a matéria sobre *Programação Orientada a Objeto* para uma melhor compreensão destes conceitos). Várias extensões foram incluídas para facilitar o uso. Podemos citar os conceitos de propriedades (particulares e públicas) e tipos de informações em modo run-time e referências de classe.

## Palavras reservadas

Existem diversas palavras que são reservadas para comandos específicos do programa e não devem ser usadas para variáveis, por exemplo, nem redefinidas.

And	Exports	Library	Set
Array	File	Mod	Shl
As	Finaly	Nil	Shr
Asm	For	Not	String
Begin	Function	Object	Then
Case	Goto	Of	To
Class	If	On	Try
Const	Implementation	Or	Type
Constructor	In	Packed	Unit
Destructor	Inherited	Procedure	Until
Div	Initialization	Program	Uses
Do	Inline	Property	Var
Downto	Interface	Raise	While
Else	Is	Record	With
End	Label	Repeat	Xor
Except			

## Números

Podemos definir variáveis e constantes de tipos Inteiro ou Real usando qualquer número decimal (de 0 a 9), mas a linguagem Object Pascal aceita as notações hexadecimal e científica.

## Constantes

A definição de constante é simples: é tudo aquilo que é fixo ou estável. Aqui, ela identifica os valores fixos do programa. A sintaxe é:

[Declaração Constante] [Identificador] (=) [constante] (;)

Veja, na tabela abaixo, algumas funções que podem ser utilizadas para a declaração de constantes:

Ab	Length	Ord	SizeOf
Chr	Lo	Pred	Succ
Hi	Low	Ptr	Swap
High	Odd	Round	Trunc

Alguns exemplos para a definição de Constantes:

```
const Min = 0;
Max = 100;
Centro = (Max - Min) div 2;
Beta = Chr(225);
NumLetras = Ord('Z') - Ord('A') + 1;
MensOla = 'Instrução inválida';
MensErro = ' Erro: ' + MensOla + ' . ';
PosErr = 80 - Length(MensErro) div 2;
Ln10 = 2.302585092994045684;
Ln10R = 1 / Ln10;
DigNumericos = ['0'..'9'];
LetrasAlpha = ['A'..'Z', 'a'..'z'];
AlphaNum = LetrasAlpha + DigNumericos;
```

## Expressões

Como em todas as linguagens, as expressões em Object Pascal são formadas por operadores e operandos, que podem ser divididas em quatro categorias básicas:

Únicos	@, Not
Multiplicativos	>, /, div, mod, and, shl, shr, as
Adicionais	+, -, or, xor
Relacionais	=, <>, <, >, <=, >=, in, is

## Identificadores

Identificadores servem, é claro, para identificar todos os tipos de componentes, constantes, tipos, variáveis, procedures, etc. Há algumas pequenas regras para se criar um identificador, vejamos as duas principais:

- ele deve ser iniciado por uma letra ou caractere underscore;
- não é permitida a utilização de espaços para a formação do nome.

Vejamos os exemplos usados no nosso tutorial da página 18: btn1, edtval e por aí vai.

## Declarações

As declarações servem para descrever as ações a serem executadas por um algoritmo.

## with... do...;

Delimita um determinado bloco de declarações para um identificador específico evitando a declaração deste identificador. A sintaxe do comando é: WITH {nome do identificador} DO {comandos};

Ex:

```
begin
  { ... comandos iniciais ... }
  with form1 do
    begin
      Caption := 'Teste';           ò Equi-
      valente a Form1.Caption
      BorderStyle := bsSizable;     ò Equi-
      valente a Form1.BorderStyle
    end;
end;
```

## array [ ... ] of ...;

Define um conjunto de variáveis ou constantes de um mesmo tipo. A sintaxe do comando é: array [{quantidade de ocorrencias}] of {Tipo};. Os arrays são controlados por três funções:

Função	Valor de Retorno
Low	Primeiro elemento
High	Aponta para o último elemento
SizeOf	Tamanho do array

Ex:

```
const
  t: array [1..50] of Char { Declara 50
  elementos para o tipo Char }
var
  s : array[1..100] of Real { Declara 100
  elementos para o tipo real }
  ind: Integer;
begin
  for Ind := Low(s) to High(s) do s[Ind] :=
  0; { Zera os elementos do array S }
  if SizeOf(t) = 'C' then exit;
  { Se o último elemento do array T for
```

```
'C' sai do bloco }
{ ... outros comandos... }
end;
```

### begin... end;

Restringe um conjunto de declarações dentro de um bloco de comandos determinado. Vejamos sua sintaxe:

```
BEGIN {comandos} END;
```

Ex:

```
begin
{ ... comandos iniciais... }
begin
{ ... bloco 1 ... }
end;
begin
{ ... bloco 2 ... }
end;
{ ... comandos finais ... }
end;
```

### if... then... else...;

Usada quando devemos escolher entre o resultado de uma condição booleana. Um caminho verdadeiro (then) ou outro falso (else). A sintaxe do comando é:

```
IF {condição} THEN {bloco de comandos}
ELSE {bloco de comandos};.
```

Ex:

```
begin
{ ... comandos iniciais ... }
if x = 3 then
{ ... Bloco verdadeiro ... }
else
{ ... Bloco falso ... };
end;
```

### goto... ;

Envia a execução do programa para o ponto que é determinado pelo Label. A sintaxe do comando é:

```
GOTO {Label};.
```

Ex:

```
label
primeiro;
begin
{ ... comandos iniciais ... }
```

```
if x = 2 then
goto primeiro;
{ ... outros comandos ... }
Primeiro:
{ ... comandos do Primeiro ... }
```

end;

### case... of... else... end;

É uma lista de declarações que satisfaz a condição de um seletor de expressões, se nenhuma parte da lista satisfaz estas condições, o subcomando else é executado. A sintaxe do comando é:

CASE (seletor) OF {Expressão 1}: {comando da expressão 1}; {Expressão 2}: {comando da expressão 2}; {Expressão n}: {comando da expressão n} ELSE {comando}; end;.

Ex:

```
begin
{ ... comandos iniciais ... }
case x of
1: { ... Bloco para x = 1 ... }
2, 3: { ... Bloco para x = 2 ou x = 3 ... }
4..6: { ... Bloco para 4 <= x <= 6 ... }
... }
else
{ ... Bloco para x < 1 ou x > 6 ... };
end;
end;
```

### repeat... until;

Repete um bloco de declarações até a satisfação da condição booleana que está no subcomando until. A sintaxe do comando é:

```
REPEAT {comandos}; until {condição};.
```

Ex:

```
begin
{ ... comandos iniciais ... }
x := 0;
repeat
x := x + 1
until (x = 2);
end;
```

### for... to (downto)... do...;

Acrescenta em 1 (um) uma variável inteira, repetindo o bloco de comandos, até que seja atingido o valor final do intervalo. O subcomando downto realiza o reverso. A sintaxe do comando é:

```
FOR {variavel} := {valor inicial} to (downto) {valor final} do {bloco de comandos};.
```

Ex:

```
begin
  { ... comandos iniciais ... }
  for i := 1 to 10 do // Executa o
[comandos A] para i = 1,2,3,4,5,6,7,8,9 e 10
    { ... Comandos A ... }
  for s := 10 downto 1 do // Executa o
[comandos B] para i = 10,9,8,7,6,5,4,3,2 e 1
    { ... Comandos B... }
end;
```

## while... do...;

O bloco de comandos será repetido enquanto determinada condição booleana for satisfeita. A sintaxe do comando é:

WHILE {condição} DO {bloco de comandos};

Ex:

```
begin
  { ... comandos iniciais ... }
  while i := 1 do // Repete o [Bloco
de comandos] enquanto i = 1
    { ... Bloco de comandos ... }
end;
```

## break; ou continue...;

O comando break interrompe um bloco de repetição for, while ou repeat saindo do bloco. A sintaxe do comando é:

BREAK; enquanto o comando continue retorna a primeira instrução do bloco de repetição for, while ou repeat. A sintaxe do comando é: CONTINUE;

Ex:

```
begin
  { ... comandos iniciais ... }
  for i := 1 to 10 do
    begin
      if i = 8 then
        break; // Salta para
os [comandos C]
      { ... comandos A... }
      if i = 5 then
        continue; / /
Retorna para o comando for pulando os [comandos
B]
      { ... comandos B ... }
    end;
  { ... comandos C ... }
end;
```

## Blocos de Procedimentos ou Funções

As procedures são declaradas na seção de tipos de declarações (abaixo do comando type) pertencendo ao objeto. Elas podem ser do tipo public (públicas - executadas por outras unidades) ou private (particulares - restritas à unidade local).

### Procedure

procedure {cabeçalho}; var {declaração das variáveis}; {bloco de comandos};

O cabeçalho da procedure é composto pelo nome do procedimento e variáveis que serão recebidas (ou modificadas) através da declaração var, ex: procedure teste(var x:string);).

```
procedure soma(a,b: integer); //
Início enviando as variáveis A e B do tipo
inteiro.
var // Declaração de
variáveis locais.
c: integer;
begin // Corpo do proce-
dimento.
  c := a + b;
end;
```

### Function

function {cabeçalho}: {resultado}; var {declaração das variáveis}; {bloco de comandos};

As funções se diferem dos procedimentos pela obrigatoriedade do retorno de um resultado, podendo este resultado ser retornado pela declaração: {nome da função} := valor ou result := valor.

```
function soma(a,b: integer) : integer; //
Início enviando as variáveis A e B do tipo
inteiro.
begin // Corpo do proce-
dimento.
  soma := a + b; // ou
result := a + b;
end;
```

### Conclusão

Este foi um pequeno artigo para apresentar conceitos básicos sobre o Object Pascal. Ele não substitui o estudo com afinco da linguagem, apenas dá uma noção e ajuda na construção da pequena calculadora que está na página 18. O que vimos aqui não é exclusivo do Object Pascal, mas é comum a várias linguagens de programação. Agora, vamos entender as características principais da Linguagem Orientada a Objetos.

# Programação orientada a objetos

## Entenda como funciona esta nova lógica

**A**ntes de prosseguirmos, será importante entender o que significa uma linguagem orientada a objetos, já que o ambiente Delphi baseia-se na linguagem Object Pascal, ou seja, o Pascal orientado a objetos.

Programar desta forma significa, nada mais que, trabalhar de forma estruturada. Mas, vejamos o que são objetos, segundo o dicionário:

1. Tudo que é apreendido pelo conhecimento, que não é o sujeito do conhecimento.
2. Tudo que é manipulável e/ou manufaturável.
3. Tudo que é perceptível por qualquer dos sentidos.

Existem outras explicações, mas essas já bastam para nós.

Na programação orientada a objetos, podemos entender a palavra "objeto" da seguinte forma: qualquer coisa que possa ser manipulada ou visualizada de alguma forma. A definição clássica é que objeto é uma instância de uma classe, ou seja, uma variável de um tipo de dado definido dentro da classe.

No ambiente Delphi, os objetos possuem duas características centrais:

**Propriedade** – refere-se a tudo aquilo que o objeto é, ou seja, ao seu conjunto de características tanto na aparência quanto na posição. Devemos considerar uma propriedade ligada diretamente a um objeto. Em termos de sintaxe, o objeto deve ser especificado primeiro, exemplo: **Objeto.Propriedade**.

**Evento** – está ligado àquilo que o objeto faz, ao seu comportamento. Esta característica é importante. Para um usuário, na maioria das vezes, não importa como um objeto funciona, mas o que ele faz. Este é o espírito por trás da programação orientada a objetos. Na sintaxe, um evento é definido por

**Objeto.Propriedade <operador> <ação>**

A comunicação entre os diversos objetos acontece através da "troca" de mensagens. Trata-se de comandos que um objeto envia a outro (no caso, o programador também é tratado como um objeto, por isso os dados de entrada podem ser chamados de mensagens, que são enviados para outros agentes, componentes do programa, ou seja, outros objetos).

Um objeto só responde a uma mensagem se conseguir entendê-la e se esse comando puder ser executado, não violando regras pré-determinadas.

Por exemplo, podemos ver uma mensagem como o comando para um carro dobrar em uma esquina. Então esse comando só poderá ser interpretado por objetos do tipo carro (que pertencem à classe Carro), ou seja, um objeto edifício não conseguirá entender essa mensagem e conseqüentemente não poderá executar esse comando.

A implementação algorítmica de uma mensagem é o que se chama de método. Quando enviamos uma mensagem a um objeto, na verdade estamos dizendo para a classe deste objeto executar o método relacionado a esta mensagem. Os métodos são locais a uma

# ntada a OBJETOS

classe e não existem fora dela (com exceção de métodos herdados, que serão vistos em herança), por isso os métodos só podem ser chamados dentro da própria classe ou através de uma instância desta classe.

## Elementos-chave

Para que uma linguagem seja orientada a objetos, é preciso que trabalhe com três elementos-chave: as classes, a herança e o polimorfismo.

**Classe** – é o coletivo de objetos. Pode ser definido como uma descrição abstrata de um grupo de objetos similares. Uma classe possui dados internos e alguns métodos, caracterizados pela forma de funções e procedimentos.

As classes são conjuntos de componentes reutilizáveis. Assim, é possível descrever as características genéricas e o comportamento de uma série de objetos, desde que sejam semelhantes.

**Herança** – é a capacidade de herdar ou aproveitar os métodos de uma classe em outra. Deste modo, é possível evitar a repetição de código inutilmente, assim como qualquer redundância que poderia inviabilizar seu funcionamento. No Delphi, podemos definir um novo tipo de classe a partir de uma classe já existente. Assim, as características são herdadas.

**Polimorfismo** – como diz o nome, polimorfismo é a capacidade que possui um objeto de assumir outras formas. Esta característica

serve para que uma variável faça referências a vários objetos de classes diferentes. O polimorfismo está diretamente ligado ao conceito de herança, pois podemos fazer mudanças em uma classe-filho que não existiam na classe-pai. Fica evidente que o conceito de polimorfismo facilita muito a vida dos programadores.

**Instância** – No mundo real, temos objetos de vários tipos. Existe o objeto genérico Carro e você possui um carro específico. Na terminologia da programação orientada a objeto, o seu carro específico é uma instância da classe Carro. Quando você cria uma instância de uma classe, você cria um objeto deste tipo e o sistema aloca memória para as variáveis de instância declaradas pela classe. Então você chama os métodos do objeto para fazer alguma coisa. Instâncias da mesma classe dividem as mesmas implementações de método (implementações do método não são duplicadas), que residem na própria classe.

## Conclusão

A programação estruturada trabalha em cima de verbos, ou seja, das ações. Já a orientada a objetos trabalha com nomes, com os dados. Estes são condicionantes das ações, fazendo com que a programação orientada a objetos seja muito mais eficiente porque possui uma estrutura para melhor simular o mundo real.

Mas, sem dúvida, é a possibilidade de reutilização de código, garantindo flexibilidade e confiabilidade, que tem levado ao crescimento e à substituição das linguagens procedurais pela orientada a objetos.

# Curso Básico

**O Delphi é difícil? Não é não. E para provar, vamos criar uma pequena calculadora só com as funções básicas, mas com a possibilidade de entender todos os conceitos básicos.**

FAQ

18

**A**presentaremos na próximas páginas um pequeno tutorial para quem está iniciando no aprendizado do Delphi. Usamos a versão Personal do Delphi7. A versão trial pode ser baixada diretamente do site da Borland ([www.borland.com](http://www.borland.com)). Após uma pequena inscrição, você recebe um email com o serial para liberação enquanto baixa o programa. Mas só faça isso se tiver uma boa conexão à Internet, afinal a versão em inglês tem 170 MB, ou pelo menos instale antes o Get Right ou algum programa assim.

Bom, iniciemos, então, este pequeno curso.

## Ambiente

Depois de tudo instalado e rodando, vamos abrir o Delphi7 e começar a fuçar um pouco no ambiente de programação, também chamada de IDE. Na figura 1, podemos ver os componentes do Delphi, como uma barra de ferramentas contendo a barra de títulos, a barra de menu, a caixa de ferramentas e a paleta de componentes.

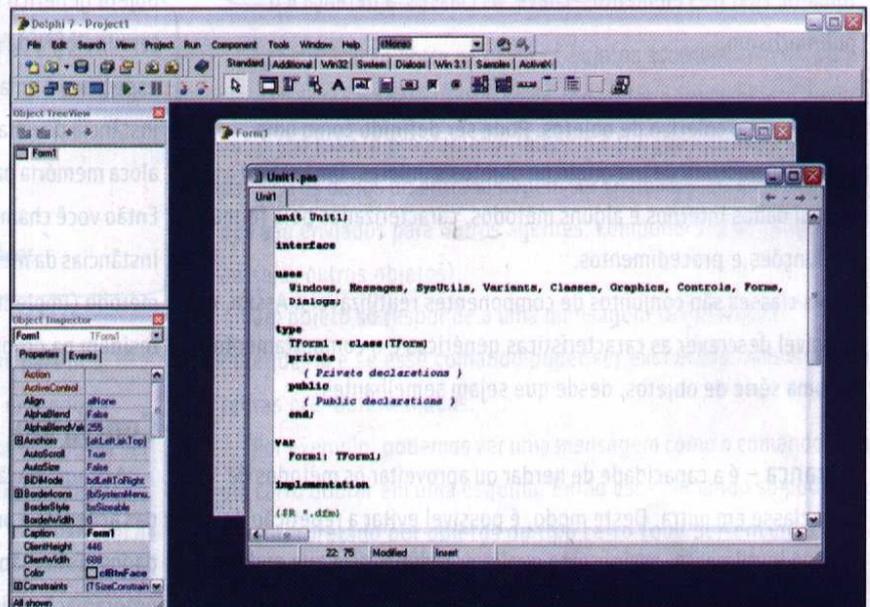


Figura 1

Há uma outra janela com o nome de Object Inspector que permite o acesso às propriedades e eventos relacionados a cada componente. Há outras janelas que iremos explicando de acordo com nosso avanço.

# de Delphi

## Construindo um projeto

Quando o Delphi é aberto, ele já está pronto para o início de um novo projeto. Você também pode clicar em File>New Application. Iremos construir uma pequena e simples calculadora.

Crie um novo diretório em C:\ para organizar seu projeto. Eu dei o nome de Projeto\_Delphi no meu. No Delphi clique com o mouse na paleta Standard (fig. 2) e selecione o componente Label. Clique em qualquer lugar do formulário para inserir o componente selecionado. Na janela Object Inspector, selecione Caption e insira o texto "Minha calculadora particular". Você pode mudar a fonte e o tamanho do texto, usando a janela do Object Inspector. Por exemplo, para mudar a fonte, clique em Font e mude tudo que quiser.

Clique agora em qualquer ponto do formulário e, no Object Inspector, insira na propriedade caption a frase "CalcDelphi". Na propriedade Name, insira "Forma Principal". Agora, é um bom momento para salvar seu projeto, certo? Em File>Save Project As, primeiro o Delphi vai pedir um nome para o módulo de código. Salve como

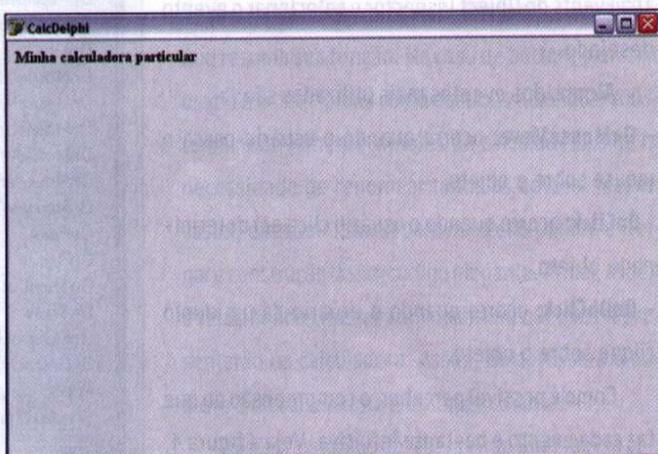


Figura 3

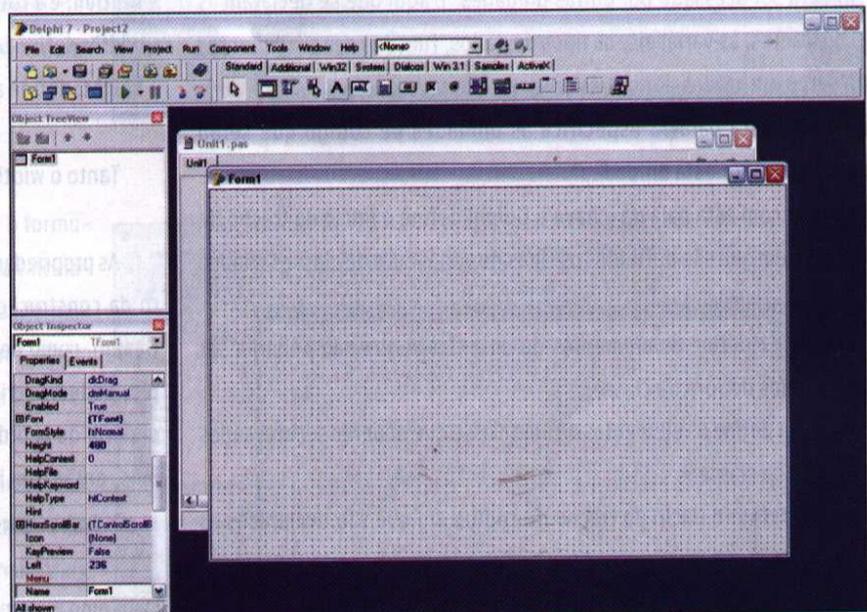


Figura 2

"projeto1.pas". Logo depois, o programa habilitará a digitação do nome do projeto. Salve como "calc".

Para fazer o programa rodar, você precisa primeiro compilá-lo, montá-lo e depois rodá-lo. Apesar de ainda não termos um programa decente, vamos ver como se faz tudo isso. Depois de salvar o projeto, clique em Project>Compile ou use o atalho Ctrl+F9. Aparentemente, nada irá acontecer mas ele estará compilando o código criado. O segundo passo é construir. Clique em Project>Build para isso. Também você não verá nenhuma mudança aparente.

É somente quando você clica no menu Run, opção Run (ou aperta F9) é que o programa é rodado. No nosso caso, ele abrirá uma janela com o texto "Minha calculadora particular" no alto e à direita (fig.3).

## Conceitos Básicos

Vamos ver alguns conceitos básicos da linguagem Object Pascal, base do Delphi. Para isso acesse o Code Editor (aquela janela que fica atrás do form), usando o atalho F12.

## Palavras-chave

Existem algumas palavras-chave ou reservadas que são usadas com significado específico e, portanto, não podem ser usadas com outra finalidade. Vejamos algumas delas que devem estar aparecendo no nosso caso, são as que estão em negrito:

- **unit**: esta palavra-chave define o nome da unidade de código.
- **interface**: define o início de um trecho de código que termina antes da palavra-chave **implementation**. Neste trecho se insere o código que poderá ser acessado por outras unidades. É aqui que se declaram as constantes, as variáveis, os tipos de dados, funções e procedimentos a serem utilizados por outras unidades de código.
- **uses**: esta palavra especifica as unidades de código que serão acessadas por esta unidade.
- **type**: com esta palavra-chave o Delphi define o início do trecho de código em que são definidos os tipos de variáveis e de classes criados pelo programa.
- **private**: define os elementos de uma classe que não poderão ser acessados de fora da classe;
- **public**: define os elementos de uma classe que poderão ser acessados de fora da classe.
- **var**: define o início do trecho de código em que são declaradas as variáveis e objetos.
- **implementation**: define o início do trecho de código em que são implementadas as funções e procedimentos declaradas no trecho iniciado pela palavra chave **interface**.
- **end**: palavra-chave usada para encerrar um bloco de código. São blocos de código:
  - um bloco de comandos iniciado pela palavra chave **begin**, caso em que **end** deve ser seguida de um ponto-e-vírgula (;).
  - a definição de uma **unit**; neste caso a palavra **end** deve ser seguida de um ponto (.) e este será o final do arquivo.

## Objetos

Já explicamos o que são objetos no artigo anterior. Em Delphi, os principais objetos são:

- **form**: como já vimos, é a interface padrão do Delphi. É a base para o desenvolvimento e onde serão colocados todos os objetos de seu programa.
- **button**: serve para colocar um botão de comando na tela.
- **label**: possibilita a apresentação de mensagens estáticas na tela.
- **image**: para a apresentação de imagens.

- **edit**: acrescenta uma caixa de texto para que o usuário possa entrar com os dados.

## Propriedades

Todo objeto possui propriedades que podem ser alteradas. Estas propriedades são as características do objeto.

- **name**: especifica o nome do objeto.
- **color**: especifica a cor do objeto.
- **caption**: é o rótulo ou mensagem que será exibida no objeto.
- **width**: define o comprimento do objeto.
- **height**: define a largura do objeto.

Tanto o width quanto o height são definidos em pixels.

As propriedades podem ser alteradas no momento da construção do programa, ou seja, durante o Design, como no momento de execução. Por exemplo, podemos criar um botão que mude a cor do formulário de acordo com o gosto do usuário. Criaríamos, então, um botão com a seguinte sintaxe:

- `objeto.propriedade = valor;`

Como exemplo:

- `form.color = clblack;`

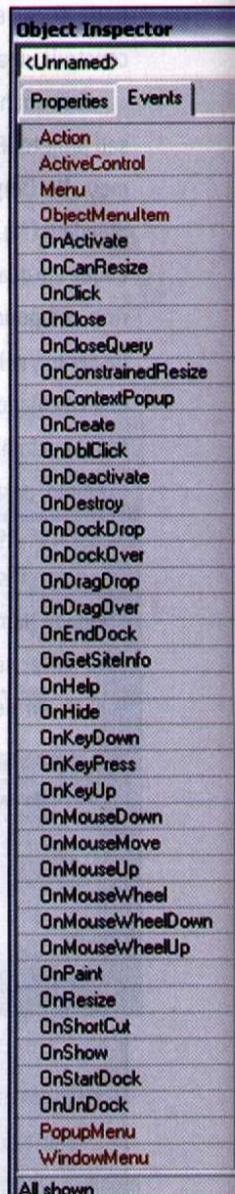
## Eventos

Toda a programação no Delphi é feita em eventos, que são as ações executadas pelos usuários do programa. Para a codificação de eventos, podemos dar um duplo clique no objeto desejado ou clicar na aba Events do Object Inspector e selecionar o evento desejado.

Alguns dos eventos mais utilizados são:

- **OnMouseMove**: ocorre quando o usuário passa o mouse sobre o objeto.
- **OnClick**: ocorre quando o usuário clica em determinado objeto.
- **OnDbClick**: ocorre quando o usuário dá um duplo clique sobre o objeto.

Como é possível perceber, a compreensão do que faz cada evento é bastante intuitiva. Veja a figura 4, com todos os eventos do Delphi.



Object Inspector	
<Unnamed>	
Properties	Events
Action	
ActiveControl	
Menu	
ObjectMenuItem	
OnActivate	
OnCanResize	
OnClick	
OnClose	
OnCloseQuery	
OnConstrainedResize	
OnContextPopup	
OnCreate	
OnDbClick	
OnDeactivate	
OnDestroy	
OnDockDrop	
OnDockOver	
OnDragDrop	
OnDragOver	
OnEndDock	
OnGetSiteInfo	
OnHelp	
OnHide	
OnKeyDown	
OnKeyPress	
OnKeyUp	
OnMouseDown	
OnMouseMove	
OnMouseUp	
OnMouseWheel	
OnMouseWheelDown	
OnMouseWheelUp	
OnPaint	
OnResize	
OnShortCut	
OnShow	
OnStartDock	
OnUnDock	
PopupMenu	
WindowMenu	
All shown	

# Construção

Passemos então à construção da nossa calculadora

## 1. Comece com a interface

Seguindo o trabalho com a interface que já havíamos criado até o momento, crie 16 botões no formulário.

Para isso, selecione o componente Button na paleta Standard e clique no formulário (figura 5). Cada Botão criado será nomeado automaticamente. Quando tiver 16 botões, inclua um componente Edit. E organize todos estes componentes com a aparência de uma calculadora. Veja a figura 6 para entender o design.

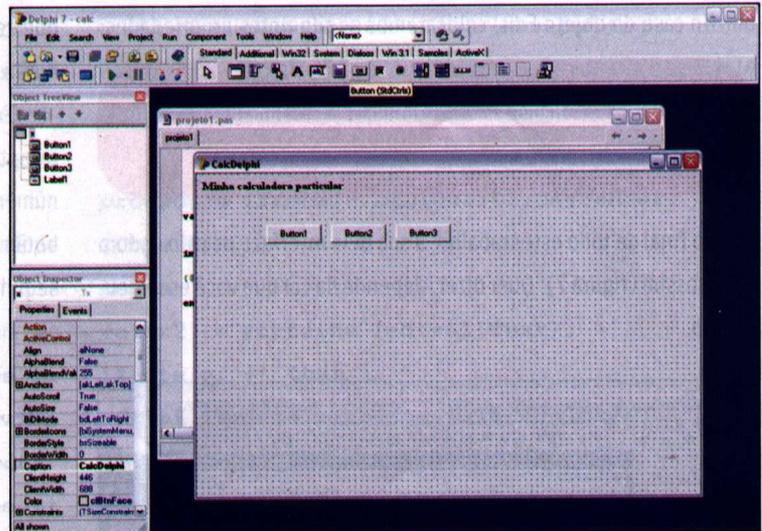


Figura 5

## 2. Configurando o Form

O segundo passo é configurar o formulário para dar melhor a idéia de uma calculadora. Antes de tudo, dê um nome ao formulário. Clique no formulário e, na janela Object Inspector, selecione Name e escreva frmcalc.

Vamos também mexer em BorderStyle (selecione bsDialog) e Color (escolha a cor que mais combine com a sua personalidade).

## 3. Trabalhando com os objetos

Agora vamos trabalhar com os vários objetos que nós criamos na primeira parte.

Sim, teremos que configurar todos aqueles botões.

Então vamos começar, para configurar cada objeto, como já falamos, é só clicar em cima do objeto.

Para melhor organizar a programação, é comum os programadores colocarem um prefixo no objeto que resuma sua função. No caso de button, o prefixo usado é o "btn", mas como estamos fazendo um sistema relativamente simples de calculadora, não há necessidade de renomear todos os botões. Mesmo assim, é recomendável para manter o padrão e treinar a construção de um código elegante. Afinal, a idéia é relacionar os nomes com o número que eles representam na calculadora. Assim, Button1 será o número 1 na calculadora e assim por diante.

Abaixo, estamos apresentando uma tabela com todas modificações necessárias nos botões:

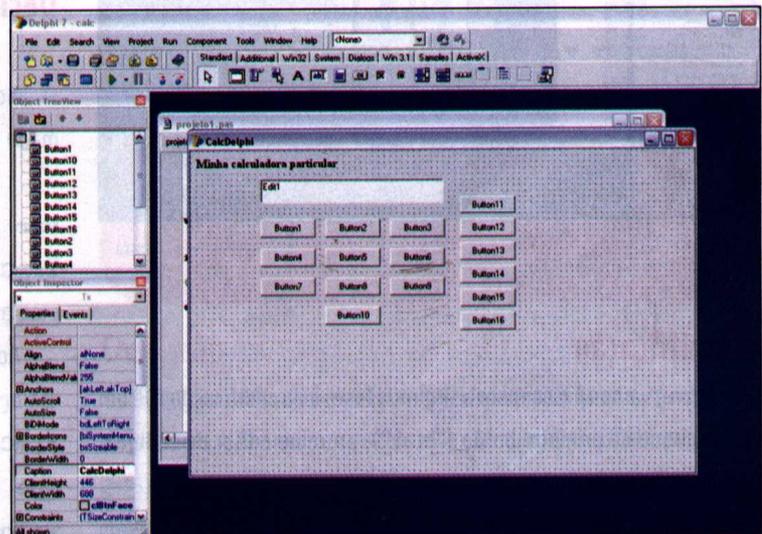


Figura 6

Objeto	Name	Caption	Tamanho da Fonte
Button1	btn1	1	14
Button2	btn2	2	14
Button3	btn3	3	14
Button4	btn4	4	14
Button5	btn5	5	14
Button6	btn6	6	14
Button7	btn7	7	14
Button8	btn8	8	14
Button9	btn9	9	14
Button10	btn10	0	14
Button11	btn11	/	14
Button12	btn12	*	14
Button13	btn13	-	14
Button14	btn14	+	14
Button15	btn15	CE	14
Button16	btn16	=	14

Figura 4

No caso do objeto Edit, colocaremos as seguintes propriedades:

Objeto	Name	Caption	Tamanho da Fonte
edittext	edtval	0	14

Ao final de todo este trabalho, a sua bela interface de calculadora ficará assim (figura 7). Quer dizer, depende da cor que você escolheu, é claro.



Figura 7

## A codificação

Agora, vamos passar para a segunda fase, a da codificação. Dê um duplo clique no primeiro botão ("btn1"), entre no editor de código e inclua os seguintes comandos.

Atenção: o texto em negrito já aparece no editor, não o digite em duplicidade.

```

procedure Tfrmcalc.Button2Click(Sender:
TObject);
begin

if edtval.Text <> '0' then
edtval.Text := edtval.Text + '1' ;

if edtval. Text = '0' then
edtval. Text := '1' ;
end;
    
```

Comentário: Na primeira parte do código, verificamos se o valor do edtval é diferente de 0 (zero). Caso afirmativo, ele apenas colocará o número 1 (um) ao lado do valor contido no edtval.text. (Lembre-se

que estamos exemplificando com o botão do número 1, mas que cada botão deve estar vinculado a um caption diferente).

O resto da programação para os outros objetos do tipo button com o caption numérico será igual ao do button 1, mas lembramos que o número a ser colocado junto ao edtval.text será o mesmo do caption do botão. Só para exemplificar, vamos colocar o código a ser colocado no segundo botão.

```

if edtval.Text <> '0' then
edtval.Text := edtval.Text + '2' ;

if edtval. Text = '0' then
edtval. Text := '2' ;
    
```

## Declarando as variáveis

No início da sua tela de códigos, você encontrará um campo chamado var, da forma que está escrito abaixo:

```

Var
frmCalculadora : TfrmCalculadora;
    
```

Adicione os seguintes comando a esse campo:

```

frmCalculadora : TfrmCalculadora;
ope : STRING;
valor, valor2 : Integer;
    
```

## Trabalhando com as operações

Vamos agora, fazer a codificação dos botões que representam as operações matemáticas. Dê um clique duplo no botão btnmais e digite (o texto em itálico já se encontra no editor, não o repita):

```

procedure Tfrmcalc.btnmenosClick(Sender: TObject);
begin
ope := '+';
valor := strtoint(edtval.Text);
edtval.Text := ' ';
btnmais.Enabled := false;
btnmenos.Enabled := false;
btndividir.Enabled := false;
btnvezes.Enabled := false;
end;
    
```

Comentário: Os blocos de comandos acima estão especificando qual operação será efetuada, junto ao travamento dos outros botões de cálculo. É importante lembrar que a função `strtoint` é responsável por transformar valores do tipo `string` (texto) para `integer` (inteiros).

Os outros botões de operações serão iguais ao `btnmais`, mas não se esqueça de mudar o valor a ser colocado junto à variável, repetindo o valor do `caption` do botão. Vamos ver o código do `btnmenos`, para não ficar nenhuma dúvida.

```
procedure Tfrmcalc.btnmenosClick(Sender: TObject);
begin
  ope := '-';
  valor := strtoint(edtval.Text);
  edtval.Text := ' ';
  btnmais.Enabled := false;
  btnmenos.Enabled := false;
  btndividir.Enabled := false;
  btnvezes.Enabled := false;
end;
```

Vamos agora tratar os últimos dois botões. O `CE` (que serve para "limpar" os números que já foram inseridos) e o `=` (igual).

Depois de clicar duas vezes sobre o botão `CE`, digite o seguinte código (não se esqueça dos códigos repetidos):

```
procedure Tfrmcalc.btnceClick(Sender: TObject);
begin
  btnmais.Enabled := true;
  btnmenos.Enabled := true;
  btndividir.Enabled := true;
  btnvezes.Enabled := true;
  edtval.Text := '\0';
  valor := strtoint('\0');
  valor2 := strtoint('\0');
end;
```

### Comentário:

O código permite que todos os botões de cálculo sejam habilitados novamente para o clique e, logo depois, zera todos os valores contidos nas variáveis (`valor` e `valor2`).

Agora, estamos chegando ao fim. Vamos partir para o último botão, o `=` (igual). Dê um duplo clique nele e insira o seguinte código:

```
procedure Tfrmcalc.btnigualClick(Sender:
TObject);
begin
  valor2 := strtoint (edtval.Text);
  if ope = '+' then
    edtval.Text := inttostr(valor+valor2);
  if ope = '-' then
    edtval.Text := inttostr(valor-valor2);
  if ope = '*' then
    edtval.Text := inttostr(valor*valor2);
  if ope = '/' then
    edtval.Text := inttostr(valor div valor2);
end;
```

### Comentário:

Nesse código, incluímos as operações matemáticas que cada botão vai preencher.

### Conclusão

E agora você tem uma pequena calculadora para fazer suas continhas mais básicas. Não é muito, mas com esse exemplo foi possível entender alguns princípios como a questão dos objetos. Agora é aprofundar seu conhecimento com as nossas dicas e mandar bala nas suas criações. A facilidade do Delphi e, agora com o Kylix, a possibilidade de portar aplicativos para o Linux, tornaram-no uma excelente ferramenta de trabalho. É por isso que, desde a sua criação, a linguagem vem ganhando adeptos em todo o mundo.

Muito estudo com os tutoriais que estão no CD e mãos à obra.

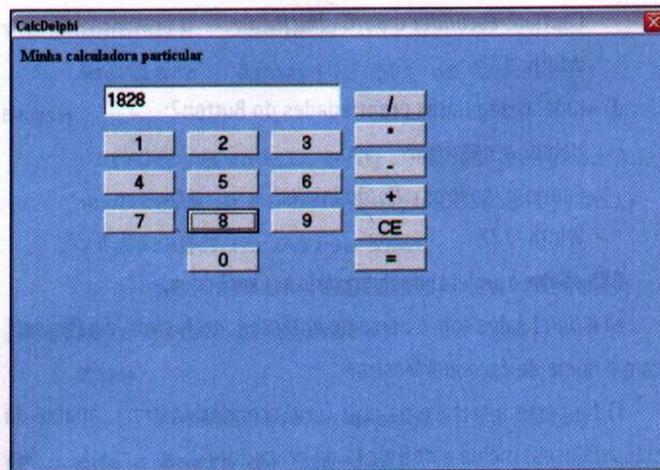


Figura 8

# Dicas de Delphi

## 1) MOSTRAR E ESCONDER O BOTÃO INICIAR

Crie um subdiretório chamado "Botão Iniciar" utilizando o Windows Explorer.

Depois abra o Delphi; feche o projeto que estiver aberto usando a opção *Close all* dentro do menu *File* e crie um novo projeto utilizando a opção *New Application* também no menu *File*.

a) Mude as seguintes propriedades do Form1:

Name: *frmEsconderMostrar*

Caption: *programa para Esconder e Mostrar o Botão Iniciar*

Position: *poScreenCenter*

BorderStyle: *bsDialog*

Height: *104*

Width: *403*

b) Insira dois Botões no formulário: na Paleta de Componentes *Standard - Button*

c) Mude as seguintes propriedades do Button1:

Name: *btnEsconder*

Caption: *esconder o Botão Iniciar*

Width: *177*

d) Mude as seguintes propriedades do Button2:

Name: *btnMostrar*

Caption: *mostrar o Botão Iniciar*

Width: *177*

**OBS.:** Salve o projeto no subdiretório que você criou:

e) A *Unit1* salve com o nome de *untEsconderMostrar*, e o *Project1* com o nome de *EsconderMostrar*

f) Na parte interface da unit (*untEsconderMostrar*), abaixo da cláusula *uses*, inclua a definição da procedure

**interface**

**uses**

*Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls;*

**Procedure**

**MostrarEsconderIniciar (Estado: Boolean);**

*{inclua esta linha}*

Na parte implementation da unit (*untEsconderMostrar*) inclua a procedure *MostrarEsconderIniciar*:

**implementation**

*{ \$R \*.DFM }*

**procedure**

**MostrarEsconderIniciar (Estado: Boolean);**

**Var taskbarhandle, buttonhandle : HWND;**

**begin**

**taskbarhandle := FindWindow('Shell\_TrayWnd', nil);**

**buttonhandle := GetWindow(taskbarhandle, GW\_CHILD);**

**If Estado = True Then**

**ShowWindow(buttonhandle, SW\_RESTORE) {mostra o botão}**

**Else**

**ShowWindow(buttonhandle, SW\_HIDE);**

*{esconde o botão}*

**end;**

# Delphi

Reunimos abaixo dicas importantes que facilitarão seu aprendizado da linguagem Delphi.

## 2) MOSTRAR E ESCONDER A BARRA DE TAREFAS

Crie um subdiretório chamado "Barra de Tarefa" utilizando o Windows Explorer.

Feche o projeto que estiver aberto usando a opção *Close all* dentro do menu *File* e crie um novo projeto utilizando a opção *New Application* também no menu *File*.

a) Mude as seguintes propriedades do Form1:

Name: *frmBarraTarefa*

Caption: *programa para esconder e mostrar a Barra de Tarefa*

Position: *poScreenCenter*

BorderStyle: *bsDialog*

Height: *104*

Width: *403*

b) Insira dois Botões no formulário: na Paleta de Componentes *Standard - Button*

c) Mude as seguintes propriedades do Button1:

Name: *btnEsconder*

Caption: *esconder a Barra de Tarefa*

Width: *177*

d) Mude as seguintes propriedades do Button2:

Name: *btnMostrar*

Caption: *mostrar a Barra de Tarefa*

Width: *177*

**OBS.:** Salve o projeto no subdiretório que você criou:

e) A *Unit1* salve com o nome de *untBarraTarefa* e o *Project1* com o nome de *BarraTarefa*.

f) Na parte interface da unit (*untBarraTarefa*) abaixo da cláusula *uses* inclua a definição da procedure

interface

uses

*Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls;*

procedure *EscondeBarraTarefa*(*EstadoBarra*: Boolean); *{inclua esta linha}*

Na parte implementation da unit (*untBarraTarefa*) inclua a procedure *EscondeBarraTarefa*:

implementation

{*\$R \*.DFM*}

procedure *EscondeBarraTarefa*(*EstadoBarra*: Boolean);

var *wndHandle* : THandle;

var *wndClass* : array[0..50] of Char;

begin

*StrPCopy*(@*wndClass*[0], 'Shell\_TrayWnd');

*wndHandle* := *FindWindow*(@*wndClass*[0], nil);

If *EstadoBarra*=True Then

*ShowWindow*(*wndHandle*, *SW\_RESTORE*)

*{Mostra a barra de tarefas}*

Else

*ShowWindow*(*wndHandle*, *SW\_HIDE*);

*{Esconde a barra de tarefas}*

end;

### 3) Pegando o nome do usuário e a empresa do Windows

Crie um subdiretório chamado "Usuário" utilizando o *Windows Explorer*.

Feche o projeto que estiver aberto usando a opção *Close all* dentro do menu *File* e crie um novo projeto utilizando a opção *New Application* também no menu *File*.

a) Mude as seguintes propriedades do *Form1*:

Name: *frmEmpresausuario*

Caption: *programa para ler do Windows nome do usuário e empresa*

Position: *poScreenCenter*

BorderStyle: *bsDialog*

Height: *123*

Width: *441*

b) Insira um Botão no formulário: na Paleta de Componentes *Standard - Button*

c) Mude as seguintes propriedades do *Button1*:

Name: *btnUsuario*

Caption: *pegar nome do usuário e empresa no Windows*

Width: *241*

d) Insira duas Caixas de Edição no formulário: na Paleta de Componentes *Standard - Edit*

e) Mude as seguintes propriedades do *Edit1*:

Name: *EdtUsuario*

Text: *vazio*

Width: *417*

f) Mude as seguintes propriedades do *Edit2*:

Name: *EdtEmpresa*

Text: *vazio*

Width: *417*

**OBS.:** Salve o projeto no subdiretório que você criou:

g) A *Unit1* salve com o nome de *untEmpresaUsuario*, e o *Project1* com o nome de *Usuario*.

h) Na parte *uses* da interface da *unit (untEmpresaUsuario)* insira a cláusula: *Registry*

interface

uses

*Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls, Registry;*

No evento **onClick** do botão *btnUsuario* inclua as seguintes linhas de código:

```
procedure TForm1.Button1Click(Sender:
TObject);
var
reg: TRegIniFile;
begin
    reg :=
TRegIniFile.create('SOFTWARE\MICROSOFT\MS
SETUP (ACME)');
    EdtUsuario.Text := reg.ReadString('USER
INFO', 'DefName', '');
    EdtEmpresa.Text := reg.ReadString('USER
INFO', 'DefCompany', '');
    reg.free;
end;
```

### 4) COMO ARRASTAR UM FORM SEM CLICAR NO CAPTION?

Quando você pressiona o botão do mouse, o Windows identifica a posição da tela onde o cursor estava no momento do clique. Se a posição é igual à área do *Caption* do Form, o Windows ativa o modo de movimentação do Form permitindo que este seja arrastado. Portanto, a maneira mais fácil de solucionar esta questão é "enganar" o Windows.

Neste exemplo, vamos considerar que o usuário poderá arrastar o Form ao clicar na área cliente deste Form:

a) Crie uma nova aplicação;

b) Adicione a seguinte declaração na seção *private* do Form:

```
procedure WMNCHitTest(var M: TWMNCHitTest); message
wm_NCHitTest;
```

c) Adicione o código deste procedimento na seção *implementation* do Form:

```
procedure TForm1.WMNCHitTest(var M:
TWMNCHitTest);
begin
```

```
inherited; { ativa a herança da mensagem }
if M.Result = htClient then { o clique foi na
área cliente? }
```

```
    M.Result := htCaption; { se sim, faz o
Windows pensar que foi no Caption. }
end;
```

Este exemplo tratou o clique na área cliente. Você pode alterar este código para suas necessidades. Eis os possíveis valores para o

**Result:**

- VALOR - Local do clique
- HTBORDER - Borda da janela que não tem a borda de tamanho
- HTBOTTOM - Borda horizontal inferior da janela
- HTBOTTOMLEFT - Canto inferior esquerdo da janela
- HTBOTTOMRIGHT - Canto inferior direito da janela
- HTCAPTION - Barra de Título(Caption)
- HTCLIENT - Área cliente
- HTERROR - igual ao HTNOWHERE, a diferença é que produz um beep indicando erro
- HTGROWBOX - Caixa de tamanho (igual ao HTSIZE)
- HTHSCROLL - Barra de rolagem horizontal
- HTLEFT - Borda esquerda da janela
- HTMENU - Em um menu
- HTNOWHERE - Plano de fundo da janela ou linha de divisão entre janelas
- HTREDUCE - Botão minimizar
- HTRIGHT - Borda direita da janela
- HTSIZE - Caixa de tamanho (igual ao HTGROWBOX)
- HTSYSTEMMENU - Botão de Sistema/Fechar da janela MDIChild
- HTTOP - Borda horizontal superior da janela
- HTTOPLEFT - Canto superior esquerdo da janela
- HTTOPRIGHT - Canto direito superior da janela
- HTTRANSPARENT - Janela em segundo plano
- HTVSCROLL - Barra de rolagem vertical
- HTZOOM - Botão maximizar

**5) BLOQUEAR A TECLA CTRL+DEL DO DBGRID**

```
procedure TForm1.DBGrid1KeyDown(Sender:
TObject; var Key: Word; Shift: TShiftState);
begin
if ((Shift = [ssCtrl]) and (key = vk_delete))
THEN
Abort;
end;
```

**6) ESCONDENDO A APLICAÇÃO DA BARRA DE TAREFAS**

Para fazer com que o ícone da aplicação em Delphi desapareça da Barra de Tarefas, execute o código a seguir:

```
var
H : HWnd;
begin
H := FindWindow( Nil, 'Project1' );
if H <> 0 then ShowWindow(H, SW_HIDE);
end;
```

**7) OS COMANDOS INC E DEC**

Você sabia que existe uma opção para a comum linha de comando: Variavel:=Variavel+1; ?

Os comandos INC e DEC permitem agilizar o processamento do seu sistema. Para isso, substitua a linha acima por:

```
INC( variavel );
ou
DEC( variavel ) se você quiser diminuir ao invés de aumentar 1.
```

FAQ

27

**8) COMO FAZER UM BEEP NO COMPUTADOR**

```
messageBeep( 0 );
```

**9) COMO FAZER UMA PAUSA POR UM PERÍODO DETERMINADO**

NumSec é o tempo em segundos de espera

```
var
NumSec SmallInt;
StartTime: TDateTime;
begin
StartTime := now;
NumSec:=10;
repeat
Application.ProcessMessages;
until Now > StartTime + NumSec * (1/24/60/
60);
end;
```

## 10) DESABILITANDO O SPLASH SCREEN DO REPORT SMITH

- 1 - Localize o arquivo RS\_RUN.INI (no diretório do Windows);
- 2 - Na seção [ReportSmith] inclua a linha seguinte:  
ShowAboutBox=0
- 3 - Na seção [RS\_RunTime] inclua a linha seguinte:  
ShowAboutBox=0
- 4 - Não se esqueça de distribuir com o seu aplicativo o referido arquivo INI.

## 11) LENDO O VOLUME DO HD

```
Function  
ExtractDiskSerial (Drive:String) :String;  
Var  
Serial:DWord;  
DirLen,Flags: DWord;  
DLabel : Array[0..11] of Char;  
begin  
GetVolumeInformation (PChar (Drive+' :\'), dLabel,  
12,@Serial,DirLen,Flags,nil,0);  
Result := IntToHex (Serial,8);  
end;
```

## 12) DESCOBRINDO O NÚMERO SERIAL DO HD

```
procedure TForm1.Button1Click (Sender:  
TObject);  
var  
SerialNum : pdword;  
a, b : dword;  
Buffer : array [0..255] of char;  
begin  
if GetVolumeInformation ('c:\', Buffer,  
SizeOf (Buffer), SerialNum, a, b, nil, 0) then  
Label1.Caption :=  
IntToStr (SerialNum^);  
end;
```

## 13) PARA SABER SOMENTE O PATH DA APLICAÇÃO

```
ExtractFilePath ( Application.ExeName )
```

## 14) INTERCEPTAR AS TECLAS DE FUNÇÃO (f1, f2, f3...)

Primeiro, coloque a propriedade *KeyPreview* do formulário como *True*. Depois, insira este código no evento *OnKeyDown* do formulário:

```
procedure TForm1.FormKeyDown (Sender: TObject;  
var Key: Word; Shift: TShiftState);  
begin  
if Key = VK_F5 then  
showMessage ('I pressed the F5 key');  
end;
```

Você também pode usar as variáveis *VK\_F1* até *VK\_F12* referentes às outras teclas de função.

## 15) TRADUZINDO A MENSAGEM: "Delete Record?"

Quando clicamos sobre o botão de deleção no DBNavigator (o do sinal de menos) surge um box com a mensagem "Delete Record?" com botões Ok e Cancel. Para que apareça a mensagem em português, você deverá selecionar o componente *Table* e mudar a propriedade *ConfirmDelete* para *False* e no evento da tabela *BeforeDelete* colocar o seguinte ([flaviojr@cyber.com.br](mailto:flaviojr@cyber.com.br)):

```
procedure  
TForm1.Table1BeforeDelete (DataSet:TDataSet);  
begin  
if MessageDlg ('Eliminar o  
Registro?', mtConfirmation, [mbYes,mbNo], 0) < mrYes  
then Abort;  
end;
```

## 16) INCLUIR UM PREVIEW PERSONALIZADO NO QUICK REPORT

No relatório, criar a procedure *SHOWPREVIEW*, contendo:

```
Procedure Showpreview;  
begin  
preview.showmodal;  
end;
```

Onde *preview* é o nome do form criado para *preview*.

Não esquecer de incluir o nome da procedure na cláusula *uses*.

Após isso, deve-se incluir no evento CREATE do formulário principal ou do relatório o direcionamento do objeto Qprinter, como a seguir:

```
qprinter.onpreview:=showpreview;
```

Isso faz com que toda vez que você queira exibir um preview, o programa abra a rotina *showpreview*, abrindo o formulário criado, chamado *preview*.

## 17) EXECUTANDO PROGRAMAS EXTERNOS

Se você precisa abrir programas externos no seu aplicativo Delphi, como a calculadora do Windows, por exemplo, inclua a seguinte linha no seu programa:

```
WinExec('calc.exe', sw_show);
```

'calc.exe' é o nome do programa. Caso queira abrir um outro programa, altere este nome.

## 18) UTILIZANDO A TECLA ENTER PARA SALTAR DE CAMPO

Insira este código em um evento OnKeyPress de um controle de edição:

```
procedure TForm1.Edit1KeyPress(Sender:
TObject; var Key: Char);
begin
If Key = #13 Then
Begin
SelectNext(Sender as TWinControl, True, True
);
Key := #0;
end;
end;
```

## 19) TOCANDO UM SOM WAV SEM O MEDIA PLAYER

Acrescente MMSystem na Uses do começo na Unit

Utilize a API `SndPlaySound()`;

Para interromper o som sem ele acabar de tocar utilize a API `PlaySound(nil,0,0)`;

Ex: `SndPlaySound('c:\teste.wav', snd_Async)`;

PS: `snd_Loop` serve para repetir continuamente o som.

## 20) OBTER O DIRETÓRIO ONDE SEU PROGRAMA ESTÁ INSTALADO

Crie uma variável do tipo String e insira a seguinte linha no evento ou função desejada do formulário:

```
ExtractFilePath(Application.Exename);
```

Retornará o path atual do arquivo EXE do seu programa.

## 21) COMO BLOQUEAR UM ARQUIVO EM AMBIENTE DE REDE

É uma dica simples, mas muito importante!

Quando você programar visando uma rede e quiser bloquear um arquivo é só chamar o método *edit* da tabela que estiver usando.

EX: `Table1.edit`;

PS: Se o registro já estiver bloqueado ocorrerá um erro. Então, faça o seguinte:

```
try { para verificar o erro }
Table1.edit;
exception on TDBEngineError do { o erro..}
MessageDlg('Registro ja esta sendo
usado...!', mtInformation, [ mbOk ], 0 );
end;
```

## 22) USANDO ENTER PARA MUDAR DE CAMPO DE UM DBGRID

```
If ( Chr(Key) <> #13) Then Exit;
If ( DBGrid1.SelectedIndex + 1 <>
DBGrid1.FieldCount ) Then
DBGrid1.SelectedIndex :=
DBGrid1.SelectedIndex + 1;
```

## 23) FUNÇÃO PARA OBTER O NÚMERO DO REGISTRO ATUAL

```
Function Recno(Dataset: TDataset): Longint;
var
CursorProps: CurProps;
RecordProps: RECProps;
begin
{ Return 0 if dataset is not Paradox or dBASE
}
```

```

Result := 0;
with Dataset do
begin
if State = dsInactive then
DBError(SDataSetClosed);
Check(DbiGetCursorProps(Handle,
CursorProps));
UpdateCursorPos;
try
Check(DbiGetRecord(Handle, dbiNOLOCK, nil,
@RecordProps));
case CursorProps.iSeqNums of
0: Result := RecordProps.iPhyRecNum; { dBASE
}
1: Result := RecordProps.iSeqNum; { Paradox }
end;
except
on EDBEngineError do
Result := 0;
end;
end;
end;
end;

```

FAQ

30

## 24) ENVIANDO UM ARQUIVO PARA A LIXEIRA

```

uses ShellAPI;
Function DeleteFileWithUndo(sFileName :
string) : boolean;
var
fos : TSHFileOpStruct;
begin
FillChar(fos, SizeOf(fos), 0);
With fos do
begin
wFunc := FO_DELETE;
pFrom := PChar(sFileName);
fFlags := FOF_ALLOWUNDO
or FOF_NOCONFIRMATION
or FOF_SILENT;
end;
Result := (0 = ShFileOperation(fos));
end;

```

## 25) CARREGAR UM CURSOR ANIMADO (\*.ANI)

```

const
cnCursorID1 = 1;
begin
Screen.Cursors[ cnCursorID1 ] :=
LoadCursorFromFile('c:\win95\ursors\cavalo.ani'
);
Cursor := cnCursorID1;
end;

```

PS: O arquivo CAVALO.ANI deverá existir no diretório apontado.

## 26) TRANSFERIR O CONTEÚDO DE UM MEMO PARA O MEMOFIELD

```

var
t: TTable;
begin
t := TTable.create(self);
with t do
begin
DatabaseName := 'MyAlias'; {Nome do Alias}
TableName := 'MyTbl.db';
open;
edit;
insert;
FieldByName('TheField').assign(Mem1.lines);
post; { Requerido!!!}
close;
end;
end;

```

## 27) CAPTURANDO O CONTEÚDO DO DESKTOP

Coloque o código abaixo no evento *FormResize* do formulário.

```

procedure TForm1.FormResize(Sender: TObject);
var
R : TRect;
DC : HDC;
Canv : TCanvas;
begin
R := Rect( 0, 0, Screen.Width, Screen.Height );

```

```
DC := GetWindowDC( GetDeskTopWindow );
Canv := TCanvas.Create;
Canv.Handle := DC;
Canvas.CopyRect( R, Canv, R );
ReleaseDC( GetDeskTopWindow, DC );
end;
```

## 28) ESCRREVENDO UM TEXTO DIAGONAL USANDO O CANVAS

```
procedure TForm1.Button1Click(Sender:
TObject);
var
begin
    with Form1.Canvas do begin
        Font.Name := 'Arial';
        Font.Size := 24;
        tf := TFont.Create;
        tf.Assign(Font);
        GetObject(tf.Handle, sizeof(lf), @lf);
        lf.lfEscapement := 450;
        lf.lfOrientation := 450;
        tf.Handle := CreateFontIndirect(lf);
        Font.Assign(tf);
        tf.Free;
        TextOut(20, Height div 2, 'Texto Diagonal!');
        end;
    end;
```

## 29) EXTRAIR UM ICONE DE UM DETERMINADO APLICATIVO

Para extrair ícones de um executável, deve-se usar a função da API Extraction. Ela usa três parâmetros:

- Instance - Instância da aplicação
  - FileName - Nome do executável. Deve ser um PChar
  - NumIcon - Número do ícone a ser recuperado. Se for Word(-1), a função retorna a quantidade de ícones do executável.
- Coloque ShellAPI em uses no começo da unit.

```
procedure TForm1.Button1Click(Sender:
TObject);
var
```

```
IconIndex : word;
h : hIcon;
begin
    IconIndex := 0;
    h :=
    ExtractAssociatedIcon(hInstance, 'C:\WINDOWS\
NOTEPAD.EXE', IconIndex);
    DrawIcon(Form1.Canvas.Handle, 10, 10, h);
end;
```

## 30) ALINHANDO ITEMS DO MENU À DIREITA

Para alinhar itens do menu principal à direita, deve-se utilizar o seguinte código:

{Isto justifica todos itens à direita do selecionado}

```
procedure SetJustify(Menu: TMenu; MenuItem:
TMenuItem; Justify: Byte);
{$IFDEF WIN32}
var
    ItemInfo: TMenuItemInfo;
    Buffer: array[0..80] of Char;
{$ENDIF}
begin
    {$IFDEF VER80}
        MenuItem.Caption := Chr(8) +
MenuItem.Caption;
    {$ELSE}
        ItemInfo.cbSize := SizeOf(TMenuItemInfo);
        ItemInfo.fMask := MIIM_TYPE;
        ItemInfo.dwTypeData := Buffer;
        ItemInfo.cch := SizeOf(Buffer);
        GetMenuItemInfo(Menu.Handle,
MenuItem.Command, False, ItemInfo);
        if Justify = 1 then
            ItemInfo.fType := ItemInfo.fType or
MFT_RIGHTJUSTIFY;
            SetMenuItemInfo(Menu.Handle,
MenuItem.Command, False, ItemInfo);
        {$ENDIF}
    end;
```

### 31) ABRIR AUTOMATICAMENTE SEU NAVEGADOR PADRÃO E CARREGAR A PÁGINA DETERMINADA PELO LINK

1º Declare o procedure na seção *Public* da unit.

```
procedure JumpTo(const aAdress: String);
```

2º Coloque a cláusula *ShellAPI* na uses no início da unit.

```
procedure TForm1.JumpTo(const aAdress: String);
```

```
var  
    buffer: String;  
begin  
    buffer := 'http://' + aAdress;  
    ShellExecute(Application.Handle, nil,  
    PChar(buffer), nil, nil, SW_SHOWNORMAL);  
end;
```

```
procedure TForm1.Label1Click(Sender: TObject);
```

```
begin  
    JumpTo('www.geocities.com/  
    SiliconValley/Way/1497');  
end;
```

### 32) COPIAR REGISTROS DE UMA TABELA PARA OUTRA INCLUINDO VALORES NULL

```
procedure TtableCopiaRegistro(Origem, Destino: Ttable);
```

```
begin  
    with TabelaOrig do  
        begin  
            {Inicia um contador para os campos da TabelaOrig}  
            for i := 0 to FieldCount - 1 do  
                {Este if verifica se o campo da TabelaOrig é NULL, se for, atribui seu valor ao campo da TabelaDest}  
                if not Fields[i].IsNull then  
                    TabelaDest.Fields[i].Assign(Fields[i]);  
            end;
```

```
end; {end with}  
end;
```

Este exemplo funcionará com todos os tipos de campos se você tiver acabado de criar a TabelaDest.

Para criar um dado valor NULL: **Fields[i].Clear**

### 33) DELETAR ARQUIVOS DE UM DIRETÓRIO COM O CARACTERE CURINGA '\*'

```
procedure  
TForm1.SpeedButton1.Click(Sender: TObject);  
var  
    SearchRec: TSearchRec;  
    Result: Integer;  
begin  
    Result:=FindFirst('c:\teste\*.**',  
faAnyFile, SearchRec);  
    while result=0 do  
        begin  
            DeleteFile('c:\teste\' + SearchRec.Name);  
            Result:=FindNext(SearchRec);  
        end;  
    end;
```

### 34) CAPTURAR A LISTA DE ALIASES DISPONÍVEIS

Tudo o que você precisa é de um componente *TSession*, um componente

*TListBox* e uma *String List*.

Defina a propriedade **SessionName** do *TSession* para **'Session'**. Utilize o seguinte código:

```
procedure TForm1.Button3Click(Sender: TObject);  
var  
    MyStringList: TStringList;  
    i: integer;  
begin
```

```

MyStringList := TStringList.Create;
Session.GetAliasNames(MyStringList);
for I := 0 to MyStringList.Count
- 1 do
    ListBox1.Items.Add(MyStringList[I]);
end;

```

Utilize o Help do **TSession** e consulte seus métodos para ver, por exemplo, como capturar o diretório ou caminho de um Alias com o método `'GetAliasParams'`.

### 35) ABRIR E FECHAR A BANDEJA DO DRIVE DE CD-ROM

```

{Para Abrir:}
mciSendString('Set cdaudio door open wait',
nil, 0, handle);

{Para Fechar:}
mciSendString('Set cdaudio door closed wait',
nil, 0, handle);

```

### 36) UTILIZANDO O CODE EXPLORER

A versão 4 do Borland Delphi está recheada de novos recursos em várias áreas do produto (IDE, Internet, linguagem, aplicações multi-tier, entre outras).

Uma das primeiras coisas que notamos quando abrimos o editor de código do Delphi 4 é a presença de um painel ancorado na lateral esquerda, contendo todos os tipos, classes, propriedades, métodos, variáveis globais, rotinas globais e interfaces contidos na unit selecionada. Este painel é chamado Code Explorer e seu objetivo é tornar mais fácil a navegação entre as units do projeto e automatizar alguns processos envolvidos na criação de classes.

O Code Explorer também permite que você navegue diretamente para as declarações que são apresentadas nele, bastando dar um duplo clique com o mouse sobre a declaração desejada. Novas declarações podem ser feitas também usando o Code Explorer, facilitando o desenvolvimento do código. Por exemplo, vamos supor que você queira criar uma nova função chamada *Calculo* com dois parâmetros do tipo real e que retornará também um real. Para isso, deve-se seguir os seguintes passos:

Selecione a pasta *Variable/Constants* dentro do *Code Explorer*. Dê um clique com o botão direito do mouse e selecione *New* no menu que aparecerá, como mostra a figura a seguir.

O *Code Explorer* apresentará um novo item que permite identificar qual o tipo de declaração que está sendo feita, por meio da informação que o programador passar a ele. No nosso exemplo, digitaremos o cabeçalho da função *Calculo* como segue:

```
function Calculo(x, y : real) : real;
```

Após digitar a declaração e pressionar `<Enter>`, o *Code Explorer* criará automaticamente o cabeçalho da função na seção *Interface* e a sua implementação na seção *Implementation da unit*, como mostra a figura seguinte, evitando assim que ocorram erros de declaração no interior do código.

Um outro recurso que o *Code Explorer* fornece é conhecido como *Class Completion*. O programador pode, dentre outras facilidades, criar apenas o básico da declaração de uma propriedade e, com o simples toque de um atalho no teclado, o *Code Explorer* completará a declaração. Como exemplo, vamos declarar uma propriedade chamada *Cor* do tipo *TColor* dentro da nossa classe *TForm1*.

```

type
    TForm1=class(TForm)
private
    {declarações privadas}
public
    {declarações públicas}
property Cor:TColor;
end;

```

Agora, com o cursor posicionado sobre a declaração da propriedade, pressionamos `<Ctrl><Shift>C` e o *Code Explorer* completará toda a declaração da estrutura da classe.

### 37) COPIANDO ARQUIVOS VIA DELPHI

```

Function
CopiaArquivo (scrname, destname : string) : byte;
var
    source, destination : file;
    buffer : array [1..1024] of byte;

```

```

readcnt, writecnt: word;
pname, dname, fname, ename: String;
{USO:
R:=COPIAARQUIVO('C:\diretorio\FILE.EXT', 'C:\
diretorio\FILE.EXT');
Devolve 0=Ok, 1=Erro no Origem, 2=Erro no
Destino, 3=Disco Cheio}
begin
    AssignFile(source, scrname);
    Try
        Reset(source, 1);
    Except
        CopiaArquivo:=1;
        Exit;
    end;
    If destname[length(destname)]='\' then
    begin
        pname:=scrname;
        destname:=destname+separa(scrname, '\', Ocorre
        (scrname, '\')+1);
        end;
        AssignFile(destination, destname);
        Try
            Rewrite(destination, 1);
        Except
            CopiaArquivo:=2;
            Exit;
        end;
        Repeat
            BlockRead(source, buffer, sizeof(buffer), readcnt);
            Try
                BlockWrite(destination, buffer, readcnt, writecnt);
            Except
                CopiaArquivo:=3; {Disco
                Cheio?}
                Exit;
            end;
        until (readcnt=0) or

```

```

(writecnt<>readcnt);
    CloseFile(destination);
    CloseFile(source);
    CopiaArquivo:=0;
end;

```

### 38) ABRIR UM TCOMBOBOX SEM CLICÁ-LO

```

ComboBox1.DroppedDown := True;

```

### 39) MUDAR A COR DA CÉLULA ATIVA DO DBGRID

A rotina abaixo deverá ser colocada no evento OnDrawDataCell, do DBGrid.

```

procedure TForm1.DBGrid1DrawDataCell(Sender:
TObject; const Rect: TRect; Field: TField;
State: TGridDrawState);
begin
    if gdFocused in State then
        with (Sender as TDBGrid).Canvas do
            begin
                Brush.Color:=clRed;
                FillRect(Rect);
                TextOut(Rect.Left, Rect.Top,
                Field.AsString);
            end;
        end;
end;

```

### 40) COMO INCREMENTAR UM MÊS NUMA DATA

```

IncMonth(Data, 1);

```

No exemplo, a variável Data é do tipo TDateTime.

### 41) VERIFICAR SE EXISTE DISQUETE NO DRIVE

```

unit UTestaDrive;
interface

```

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

```
type
  TForm1 = class(TForm)
    Button1: TButton;
    procedure Button1Click(Sender:
  TObject);
    function TemDiscoNoDrive(const
  drive : char): boolean;

  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

```
var
  Form1: TForm1;
```

**implementation**

```
{ $R *.DFM }
```

```
function TForm1.TemDiscoNoDrive(const drive :
char): boolean;
var
  DriveNumero : byte;
  EMode : word;
begin
  result := false;
  DriveNumero := ord(Drive);
  if DriveNumero >= ord('a') then
    dec(DriveNumero, $20);
  EMode :=
  SetErrorMode(SEM_FAILCRITICALERRORS);
  try
    if DiskSize(DriveNumero-$40) = -1
  then
    Result := true
  else
    messagebeep(0);
```

```
finally
  SetErrorMode(EMode);
end;
end;

procedure TForm1.Button1Click(Sender:
TObject);
begin
  if TemDiscoNoDrive('a') then
    ShowMessage('Tem disco No drive
A:');
  else
    ShowMessage('Não tem Disco no
Drive A:');
end;
end.
```

**42) ACESSAR O AMBIENTE DOS**

Para acessar as variáveis do ambiente DOS, deve-se usar a função da API **GetDosEnvironment**. Ela retorna um PChar que pode ser avaliado.

**43) EXECUTAR UM PROGRAMA (DOS) E FECHAR SUA JANELA EM SEGUIDA**

Quando você executa um programa DOS no Windows 95, sua janela permanece aberta até ser fechada pelo usuário. Para executar um programa DOS que fecha sua janela após a execução, deve ser especificado "command.com /c programa" na linha de comando. Usando a função da API WinExec para executar um programa chamado proddos.exe, a chamada deve ser:

```
WinExec('c:\command.com /c progdos.exe',
sw_ShowNormal);
```

Se o programa deve ser executado sem que seja visualizado pelo usuário, o segundo parâmetro deve ser sw\_Hide. Deve ser especificada a extensão .com, senão o programa não será executado.

#### 44) INSTANCIAR UMA ÚNICA VEZ UM EXECUTÁVEL CORRESPONDENTE A UMA DETERMINADA APLICAÇÃO

No Microsoft® Windows®, existe uma tabela do sistema chamada 'atom table'. Esta tabela armazena strings com seus correspondentes identificadores. Existem várias funções da API do Windows, chamadas de 'atom functions', que permitem que uma aplicação insira, apague, procure por uma determinada 'atom string', etc. O código abaixo garante que somente uma instância do executável de sua aplicação estará carregado em memória. O form1 seria o form principal da aplicação:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
    {Procura na tabela para verificar se o
    programa já está rodando}

    {Substitua a string 'MINHA STRING' por
    uma de sua conveniência}
    if GlobalFindAtom('MINHA STRING') = 0
    then
        {zero significa não encontrar}
        atom := GlobalAddAtom('MINHA
        STRING')
    else
        begin
            {Se o programa já estiver rodando,
            então mostrar a mensagem e parar}
            MessageDlg('A aplicação já encontra-se em execução!!', mtWarning, [mbOK], 0);
            Halt;
        end;
    end;

procedure TForm1.FormDestroy(Sender:
TObject);
begin
    {Remove o item da tabela, de modo que a
    aplicação possa ser executada novamente}
    GlobalDeleteAtom(atom);
end;
```

#### 45) MODIFICAR VÁRIAS PROPRIEDADES DE UM OBJETO AO MESMO TEMPO

Utilize o comando with, desse modo:

```
With Edit1 do
begin
    Text := '@Delphi';
    Width := 30;
    Enabled := True;
end;
```

#### 46) PARA EMITIR UM SOM QUANDO O ENTER FOR PRESSIONADO

No Evento OnKeyPress de um Edit:

```
if Key = Chr(VK_RETURN) then
    Perform(WM_NEXTDLGCTL, 0, 0);
```

#### 47) ENCOLHENDO O EXECUTÁVEL

Em Delphi 1.0, marcando a checkbox *Optimize for size and load time*, em *Options/Project/Linker*, não funciona (aparece uma mensagem de erro de disco cheio, mesmo com muito espaço). O Delphi 1.0 vem com um programa DOS, W8LOSS, que faz o mesmo. Para usá-lo, deve-se digitar o seguinte:

W8LOSS programa.exe

Isto encolherá o executável em aproximadamente 20%, diminuindo o tempo de carga.

#### 49) DESABILITAR AS TECLAS DE ACESSO (CTRL+ALT+DEL), (ALT+TAB), (CTRL+ESC)

```
var
    OldValue : LongBool;
begin
    {liga a trava}
```

```
SystemParametersInfo(97, Word(True),
@OldValue, 0);
  {desliga a trava}
  SystemParametersInfo(97, Word(False),
@OldValue, 0);
end;
```

### 50) FAZER UM SPLASH SCREEN

```
form2:=tform2.create(application);
form2.show;
form2.update;

form2.hide;
form2.free;
Application.Run;
```

Obs.: apagar a primeira linha, 'Application.Initialize'.

### 51) FUNÇÃO PARA ENCRYPTAR UMA STRING

```
function encrypt( dummy: Pchar):Pchar;
var
  x: Integer;
  w: Word;
  s: String;
  c: Char;
begin
  s:=StrPas(dummy);
  w:=StrLen(dummy);
  for x:=1 to w do
  begin
    c:=s[x];
    c:=char ( ord ( c) xor 159);
    s[x]:=c;
  end;
  StrPCopy(dummy,s);
  encrypt:=dummy;
end;
```

### 52) FUNÇÃO REVERTER UMA STRING

```
function TForm1.StrReverse(MyString :
string) : String;
var
  i: integer;
  HelpString: string;
begin
  HelpString := '';
  for i := 1 to Length(MyString) do
  HelpString := MyString[i]+HelpString;
  Result := HelpString;
end;
```

### 53) FUNÇÃO DE CONVERTER UM NÚMERO INTEIRO PARA BINÁRIO

```
{Integer to Binary}
function IntToBin(Value: LongInt;Size:
Integer): String;
var
  i: Integer;
begin
  Result:='';
  for i:=Size downto 0 do
  begin
    if Value and (1 shl i)<>0 then
    begin
      Result:=Result+'1';
    end
    else
    begin
      Result:=Result+'0';
    end;
  end;
end;
```

### 54) FUNÇÃO DE CONVERTER UM NÚMERO BINÁRIO PARA INTEIRO

```
{Binary to Integer}
function BinToInt(Value: String): LongInt;
```

```

var
  i, Size: Integer;
begin
  Result:=0;
  Size:=Length(Value);
  for i:=Size downto 0 do
  begin
    if Copy(Value,i,1)='1' then
    begin
      Result:=Result+(1 shl i);
    end;
  end;
end;
end;

```

## 55) FUNÇÃO PARA ENCRYPTAR E DESENCRIPTAR UMA STRING

```

const
  StartKey = 981; {Start default key}
  MultKey = 12674; {Mult default key}
  AddKey = 35891; {Add default key}
{Encryptar}
function Encrypt(const InString: string;
  StartKey, MultKey, AddKey: Integer): string;
var
  I: Byte;
begin
  Result := '';
  for I := 1 to Length(InString) do
  begin
    Result := Result +
CHAR(Byte(InString[I]) xor (StartKey shr 8));
    StartKey := (Byte(Result[I]) +
StartKey) * MultKey + AddKey;
  end;
end;
{Decryptar}
function Decrypt(const InString: string;
  StartKey, MultKey, AddKey: Integer): string;
var
  I: Byte;
begin

```

```

Result := '';
for I := 1 to Length(InString) do
begin
  Result := Result +
CHAR(Byte(InString[I]) xor (StartKey shr 8));
  StartKey := (Byte(InString[I]) +
StartKey) * MultKey + AddKey;
end;
end;

```

## 56) UMA ROTINA PARA VERIFICAR ERROS EM TODA A APLICAÇÃO

Para tratar erros de forma genérica, em todo o seu programa, insira no método ON CREATE do formulário principal a linha:

```
Application.OnException:=RotinaGeral;
```

RotinaGeral é uma procedure na qual deverá constar o código para verificar e enviar as mensagens de erro do seu sistema.

## 57) ENVIANDO INFORMAÇÕES DIRETO PARA A IMPRESSORA

Muitas vezes torna-se necessário, ou até mesmo, imprescindível que você envie informações diretamente para a impressora, uma vez que a utilização da impressão típica do Windows é um pouco demorada e o uso do driver Genérico/Somente Texto não é muito confiável.

Uma boa solução para enviar informações diretamente para a impressora é usar o seguinte código:

```

Procedure TForm1.Button1Click(Sender:
Object);
var
  Imp: TextFile;
begin
  AssignFile(Imp, 'LPT1');
  Rewrite(Imp);
  Write(Imp, 'Isto vai sair na impressora');
  CloseFile(Imp);
end;

```

Desta forma será possível, inclusive, utilizar os códigos de configuração da impressora. Para a impressora padrão Epson, por exemplo, você poderia utilizar algo assim:

```
Write(Imp, #27#69 + 'Teste' + #27#70); {
  impressão em negrito }
Write(Imp, #15 + 'Teste' + #18); { impressão
  no modo condensado }
Write(Imp, #12); { salto de página }
```

## 58) IMPRIMIR DIRETAMENTE PARA A IMPRESSORA SEM PASSAR PELO GERENCIADOR DE IMPRESSÃO

```
procedure TForm1.Button1Click(Sender:
  TObject);
var
  F : TextFile;
  i : integer;
begin
  AssignFile(F, '&rsquo;LPT1');
  Rewrite(F);
  i := 0;
  Writeln(F, '&rsquo;Teste de impressao - Linha 0');
  Writeln(F, '&rsquo;Teste de impressao - Linha
  1');
  Writeln(F, #27#15+&rsquo;Teste de Impressão -
  Linha 2');
  Writeln(F, '&rsquo;Teste de impressao - Linha 3');
  Writeln(F, #27#18+&rsquo;Teste de Impressão -
  Linha 4');
  Writeln(F, '&rsquo;Teste de impressao - Linha 5');
  Writeln(F, #12); // Ejeta a página
  CloseFile(F);
end;
```

## 59) CRIAR BARRA DE STATUS COM SUPORTE ÀS CAIXAS DE EDIÇÃO DO WINDOWS 95

No evento OnCreate do Form:

```
Application.OnHint := DisplayHint;
```

Crie (e defina) uma Procedure DisplayHint (no Form Principal):

```
StatusBar1.Panels[0].Text :=
  Application.Hint;
```

## 60) CAPTURANDO UMA TELA DO WINDOWS

```
procedure TForm1.Button1Click(Sender:
  TObject);
var
  DeskTopDC: HDC;
  DeskTopCanvas: TCanvas;
  DeskTopRect: TRect;
begin
  DeskTopDC :=
  GetWindowDC(GetDeskTopWindow);
  DeskTopCanvas := TCanvas.Create;
  DeskTopCanvas.Handle := DeskTopDC;
  DeskTopRect :=
  Rect(0,0,Screen.Width,Screen.Height);
  Form1.Canvas.CopyRect(DeskTopRect,DeskTopCanvas,
  DeskTopRect);
  ReleaseDC(GetDeskTopWindow,DeskTopDC);
end;
```

## 61) CONFIGURAÇÕES INTERNACIONAIS

Normalmente o Delphi busca os formatos de data/hora, moeda e formato numérico da Configuração Internacional do Painel de Controle. Isto pode levar a erros quando avaliando datas, números ou listas.

Para evitar estes erros, você pode mudar as constantes definidas no Delphi, como DecimalSeparator, ShortFormatDate e outros desta maneira:

```
DecimalSeparator := '.';
ShortFormatDate := 'mm/dd/yy';
```

Isto terá precedência sobre a configuração padrão. Para uma lista completa das variáveis, procure em Currency Formatting Variables na ajuda do Delphi.

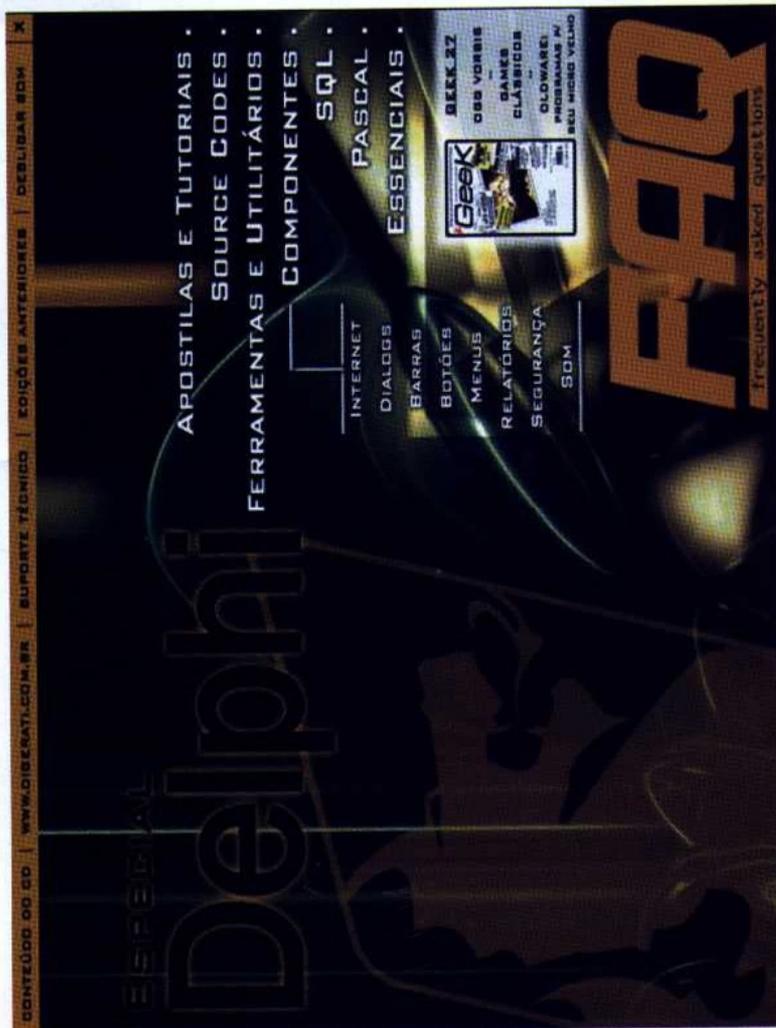
# GUIA DO CD

por Juliano Barreto  
juliano@tigerati.com.br

## Delphi sem Limites

O Delphi é uma das linguagens de programação mais usadas no atual mercado de trabalho, e isso não é por acaso. Com a linguagem e seus componentes especiais é possível desenvolver todo tipo de software que você imagina. Por isso, reunimos as ferramentas mais úteis para que seus projetos saiam da mesmice e você atinja o patamar de um programador profissional. Com o uso do SQL para interação do seu programa com banco de dados ou com os componentes para Internet, você criará aplicações prontas para serem distribuídas e até vendidas (inclusive com a possibilidade de criar versões shareware e demo). Já para aqueles que ainda não conhecem ou não dominam totalmente os comandos do Delphi, selecionamos vários tutoriais que ensinam o básico, o intermediário e o avançado e seguem a grade de programação dos melhores cursos. Também para aprender a linguagem mais a fundo você terá a oportunidade de conhecer o Pascal, que é tido como o "pai" do Delphi.

Para completar, no CD você ainda encontra ferramentas para otimização de código, criação de instaladores e centenas de códigos-fonte para adaptar e aprender a criar aplicações cada vez melhores.



## Configurações recomendadas

A interface do CD é melhor visualizada em monitores com resolução 1.024 por 768 e micros com pelo menos 32MB de memória RAM. Porém, alguns programas exigem muito mais da máquina para ser instalados e mais ainda para funcionar corretamente. Recomendamos que ao colocar o CD em seu drive, feche os aplicativos mais pesados que estiverem rodando na sua máquina, e no caso da instalação de algum software é sempre bom fechar todas as janelas para evitar que o seu PC trave. Os programas necessários para abrir os códigos-fonte, as apostilas e o restante do conteúdo do CD estão na categoria "Essenciais". Para usuários de sistemas baseados no Linux disponibilizamos um arquivo em HTML na raiz do diretório do CD. Nele, você encontrará todo o conteúdo do CD.

### **Categoria: SQL** **Seu projeto acessando Bancos de Dados**

A qualidade e rapidez do acesso a informações de um Banco de Dados pode ser o diferencial entre um projeto de sucesso e um "projetinho qualquer". Por isso, o mercado de trabalho hoje em dia praticamente exige um conhecimento básico de SQL para quem programa em Delphi. Nesta categoria você encontrará ferramentas para gerenciar dados e melhorar a relação entre seu projeto e a base de dados que você usa. E ainda vai conhecer a "evolução" do SQL, o MySQL, que está sendo cada vez mais usado para aplicações da Internet. E para buscar ainda mais conhecimento sobre o assunto, a categoria "Tutoriais e Apostilas" oferece vários tutoriais a respeito de construção e estrutura de Bancos de Dados, SQL e Access.

### **Categoria: Componentes** **Busque o algo mais**

Habilidade para programar e um senso apurado de lógica são importantíssimos para desenvolver um software útil e competitivo. Porém, usando os componentes você pode tirar mais vantagens do Delphi e criar softwares com uma aparência profissional. Os engines, menus, botões e barras que já vêm otimizados e prontos para colocar em seu programa podem mudar o conceito do seu projeto para melhor. Alguns foram feitos sob medida para seu projeto; confira as principais categorias e seus respectivos destaques:

## >> SUPORTE TÉCNICO

Em caso de dúvidas referentes ao uso ou a instalação da interface do CD o nosso departamento de suporte técnico terá prazer em ajudá-lo. Você pode entrar em contato conosco em horário comercial de segunda a sexta, pelo telefone (11) 3217-2626 ou pelo seguinte e-mail: [atendimento@digerati.com.br](mailto:atendimento@digerati.com.br)

### **Categoria: Pascal** **As raízes do Delphi**

Conhecer os comandos e a estrutura das linhas de código do Pascal é um passo vital para programar melhor em Delphi. As duas linguagens têm a mesma origem, com a diferença da segunda ser mais moderna e flexível e hoje em dia ter muito mais mercado. Porém, quem acha que o Pascal é uma linguagem "morta" está completamente enganado. Em vários cursos de renomadas instituições o aprendizado dos melhores profissionais desenvolvedores de Delphi tem o Pascal como seu ponto de partida. Por isso selecionamos as ferramentas básicas para programar em Pascal e na categoria "Source Codes" mais de 200 códigos-fonte de programas escritos na linguagem que originou o Delphi. Assim temos um pouco de nostalgia para os mais velhos e um bom aprendizado para os iniciantes.



## Categoria: **Source Codes**

### Programas em Delphi como você nunca viu

Sem dúvida uma das grandes vantagens do Delphi é o seu grande poder de se adaptar às necessidades dos desenvolvedores. E você poderá conferir isso nesta sessão. Os códigos-fonte estão compactados, porém, na maioria deles existe um arquivo de texto dizendo como ele deve ser compilado ou então que dentro do source (no arquivo .PAS) você encontrará algumas linhas de comentário dizendo mais sobre o programa. Os programas têm muitas utilizações que servirão de base para que você crie seus próprios projetos, mesclando linhas de comando de um com as rotinas de outros. Entre os mais de 600 arquivos, você encontrará softwares divididos nas seguintes subcategorias:

**Windows API Call:** São códigos que executam tarefas do Windows, como abrir programas, apagar arquivos e realizar outras tarefas básicas como salvar, recortar, colar, etc...

**Complete Applications:** Programas que já estão completos e prontos para o uso, com seus códigos-fonte abertos para estudo e alteração

**Games:** Uma coleção com jogos simples que fazem uso de string e variáveis, muito interessante para quem gosta de lógica de programação

**Gráficos:** São softwares e comandos para manipulação e tratamento de imagens

**Internet e Redes:** Clientes de e-mail, programas para acessar sites FTP, incrementos para interação entre seu projeto e o browser do usuário, detector de conexão ativa e muito mais para colocar seu software "on-line"

**Pascal:** Mais de 200 programas completos para aprender a aplicação e os fundamentos da linguagem que originou o Delphi

**Matemáticos:** Códigos que usam a linguagem Delphi para realizar cálculos matemáticos, dos mais simples aos mais avançados

**Sound:** Players, Mixers, softwares para controlar o volume dos auto-falantes do micro do usuário e muito mais para adicionar som aos seus projetos Delphi

**Standard:** Código simples de programas simples, ótimo para iniciantes que desejam aprender a programar na prática

## Subcategoria: **Som**

Adicionar controles de som aos seus projetos pode ser um importante diferencial.

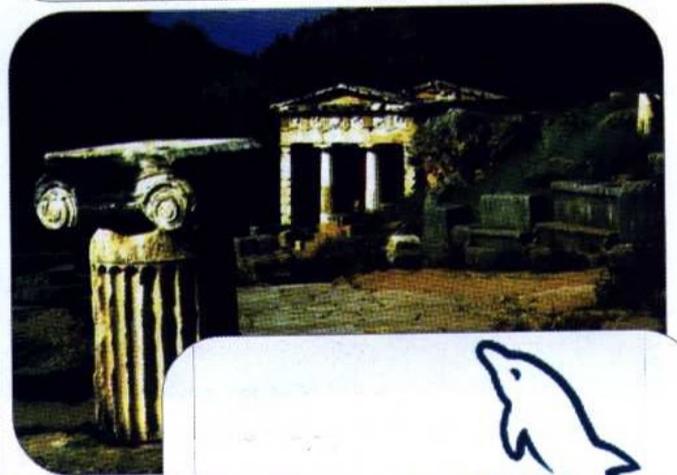
Um botão de volume impressiona bastante, e a capacidade de tocar arquivos de som podem dar um toque de originalidade ao seu projeto.

## Categoria: **Ferramentas e Utilitários** Faça a festa no Delphi!

Nesta categoria fizemos uma seleção com os melhores softwares "Third-party".

Esse apelido, que significa mais ou menos "programas de terceiros", denomina os softwares desenvolvidos por outras companhias para complementar as aplicações feitas em Delphi.

Com eles, desenvolver o software dos seus sonhos fica muito mais fácil, das primeiras linhas programadas (com mais organização) até a criação do pacote de instalação (menor e em um só arquivo). E para completar, o *Open Kylix 3.0*, que para quem não conhece é a versão open source do Delphi para sistemas GNU/Linux. Ele é totalmente funcional e gratuito e serve como um novo horizonte para quem está cansado de programar para o instável Windows.



MySQL™

## Subcategoria: **Internet**

Essa é a subcategoria que possui mais componentes do CD. São mais de 50 componentes para criação de projetos com suporte a envio e recebimento de e-mail, como o *AHM E-Mail Triton v.2000*, o *ISGMapi v.2.5*, o *SakEmail v.3.0.1* e outros para acesso a sites FTP como o *Monster FTP Client v.0.4.7*, o *FirewallFTP*, o *FTP Client Engine* e o *Ftp4w.dll v.3.0*. Além de engines para programar um software que entra nos canais de bate-papo do IRC, como o *Galaxy IRC Component v.2.6*, o *TmlRCCControl v.1.2* e outros.

Na sessão de Componentes Especiais para Internet você também irá encontrar ferramentas para detectar a conexão do computador do usuário e para interagir com o browser instalado na máquina que seu programa está rodando.

## **Categoria: Apostilas e Tutoriais** **Quem sabe, sabe**

Na categoria "Cursos e Apostilas" do CD você irá encontrar mais de 50 documentos que aumentarão seu nível de conhecimento, seja você um programador experiente ou apenas um curioso que está começando. Todos os tutoriais são muito úteis para desenvolver habilidades que você deixa de lado por falta de tempo ou por pura necessidade.

Por exemplo, se você trabalha com banco de dados, provavelmente não deve dominar a programação de aplicativos para segurança ou para Internet, como firewalls, e com isso perde a chance de se tornar um profissional melhor que se aproveita de outras vantagens que o Delphi oferece. Porém, se você ainda nem é um profissional, nada melhor do que começar bem. Com as apostilas de nível universitário presentes no CD desta edição, você vai poder fazer um curso de programação com qualidade sem sair de casa e com o seu ritmo.

Para quem já sabe um pouco ou para quem não sabe nada, aprender é o caminho.



## **Subcategoria: Barras**

Rotinas de programação bem escritas, estabilidade e rapidez são requisitos essenciais de um software bem sucedido. Porém, de nada adianta se a aparência do software assustar o usuário e deixar os menus comandos escondidos. Uma boa saída para isso são os componentes da categoria "Barras". Eles organizam os comandos criando abas com as quais você deixa tudo mais organizado e fácil para quem vai usar seu software.

## **Subcategoria: Menus**

Um programa complexo com muitas ações e muitos botões não pode depender apenas das poucas opções que o Delphi dispõe. Por isso os *Menus personalizados* são uma ferramenta essencial para que seu software possa ser mais organizado e fácil de usar. Uma interface mais bonita deixa seus programas mais fáceis de usar e seus amigos ainda vão perguntar:

"Como você fez isso?"

## **Subcategoria: Relatórios**

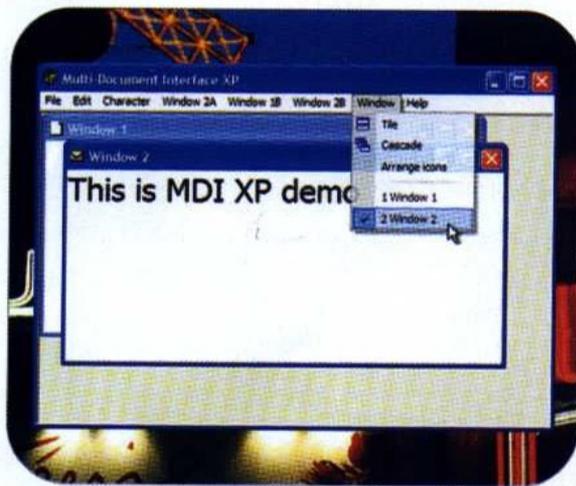
Se você acha que as opções para geração de relatórios oferecidas pelo QuickReport não são o suficiente, experimente incrementar ele com um dos componentes desta categoria. Além de exportar seus relatórios para formatos mais populares, como o universal HTML, você poderá visualizá-los e imprimi-los de uma forma mais rápida e fácil, e isso conta muito quando o tempo está contra você.

## **Subcategoria: Segurança**

Um dos mercados que mais crescem na área de softwares é o da segurança para usuários domésticos. As desenvolvedoras de antivírus e softwares para proteção em geral (firewalls, anti-spam, antipop-up etc..) ganham cada vez mais prestígio e dólares. O que você acha de entrar para este clube? As ferramentas estão aí, são componentes para encriptação de dados, proteção e criação de senhas e muito mais.

## **Subcategoria: Botões**

Seguindo a mesma filosofia das barras, os botões também podem melhorar muito as suas aplicações. Botões inteligentes e configuráveis podem ser fortes aliados para obter informações sobre os usuários do seu software além de melhorar sua aparência e dar mais credibilidade ao seu produto.



## **Subcategoria: Dialogs**

As janelas de diálogo podem tornar o uso do seu programa muito mais intuitivo e rápido. Dominando as janelas padrão do Windows e sabendo alterá-las, tudo fica mais bonito e simples. Com um bom uso das janelas de diálogo é possível até criar assistentes que ensinam os usuários a usar seu programa.

# FAQ

frequently asked questions

Complete sua coleção: [www.digerati.com](http://www.digerati.com)  
Entrega grátis para todo o Brasil

## ATENDIMENTO AO LEITOR

Fone: (11) 3217-2626 (9h às 21h)  
[www.digerati.com.br](http://www.digerati.com.br)  
[suporte@digerati.com.br](mailto:suporte@digerati.com.br)  
Marcos Raul de Oliveira,  
Eduardo Rodrigues e Rodrigo França

## ATENDIMENTO DE VENDAS

Fone: (11) 3217-2600  
Simone Araújo

## Revista FAQ

### Diretor Editorial

Alessio F. Melozo ([alessio@digerati.com.br](mailto:alessio@digerati.com.br))  
MTB 026412

### Editor

Marcelo C. Barbão ([mbarbao@digerati.com.br](mailto:mbarbao@digerati.com.br))

### Editor Assistente

Maurício Martins ([mauricio@digerati.com.br](mailto:mauricio@digerati.com.br))

### Redatores

Bruno Cesar, João Marinho e Fernando Wiek

### Arte

Marina Fiorese, Heiber Bimbo e Fábio Augusto

### Colaboraram nesta edição:

Marcelo Jaloto Machado

### Departamento Multimídia

Design e programação: Rodrigo Rudiger  
Seleção de programas: Juliano Barreto

### Revisão

Priscila Cassettari, Cíntia Yamashiro

Os artigos assinados não refletem necessariamente  
a opinião da revista, e sim de seus autores.



Essa revista é mais uma publicação da

**DIGERATI**  
editorial

### Digerati Comunicação e Tecnologia Ltda

Rua Haddock Lobo, 347 – 12º. Andar  
CEP 01414-001 São Paulo SP  
Fone: (11) 3217-2600 Fax: (11) 3217-2617  
[www.digerati.com.br](http://www.digerati.com.br)

### Diretores

Alessandro Gerardi – ([gerardi@digerati.com.br](mailto:gerardi@digerati.com.br))  
Luís Afonso G Neira – ([afonso@digerati.com.br](mailto:afonso@digerati.com.br))  
Alessio Fon Melozo – ([alessio@digerati.com.br](mailto:alessio@digerati.com.br))

### Diretor Comercial

René Luiz Cassettari – ([rene@digerati.com.br](mailto:rene@digerati.com.br))

### Marketing

Érica V. Cunha, Simone Siman, Carlos Ignatti, José Antonio Martins

### Recursos Humanos

Viviane Cardoso – ([viviane@digerati.com.br](mailto:viviane@digerati.com.br))

### Logística de Produção

Pierre Abreu – ([pierre@digerati.com.br](mailto:pierre@digerati.com.br))

### Tecnologia da Informação

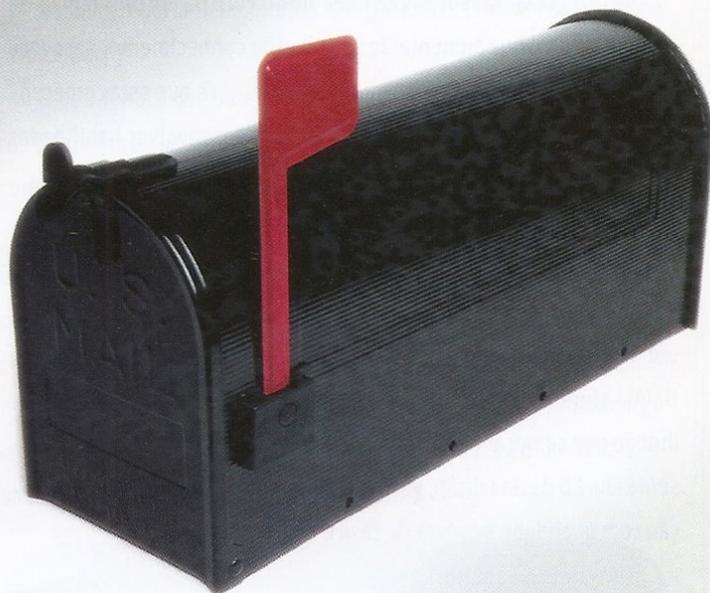
Flávio Tâmega – ([flavio@digerati.com.br](mailto:flavio@digerati.com.br))

### Impressão e Acabamento

Oceano Indústria Gráfica Ltda.  
Fone: (11) 4446-6544

### Distribuidor Exclusivo para bancas de todo o Brasil

Fernando Chinaglia Distribuidora SA  
Fone: (21) 3879-7766



Complete a sua coleção sem sair de casa.

É só digitar [www.digerati.com](http://www.digerati.com) e escolher a revista.



Visite o novo site da Digerati.

Nós entregamos a sua revista.

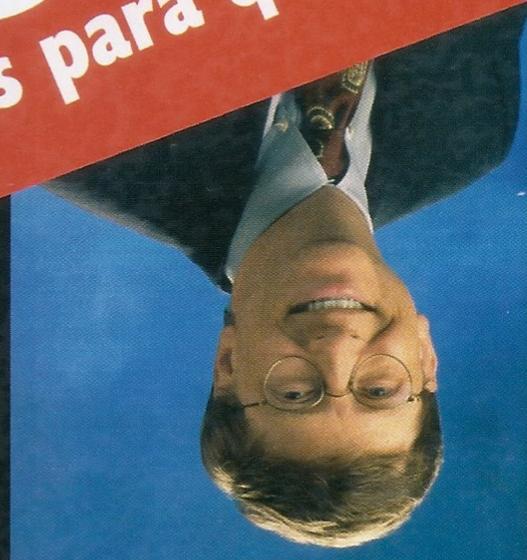


**DIGERATI**  
editorial

[www.digerati.com](http://www.digerati.com)

" 640 KB  
 são suficientes para qualquer um. "

Bill Gates, 1981



www.geek.com.br

O site da revista Geek



www.digerati.com