BLAISE PASCAL MAGAZINE 114/115

Multiplataforma /Object Pascal / Internet / JavaScript / Web Assembly / Pas2Js /
Bancos de dados / Estilos CSS / Aplicativos Web progressivos
Android / IOS / Mac / Windows & Linux

CONTEÚDO

ARTIGOS

Android / IOS / Mac / Windows & Linux

PUBLICIDADE

Niklaus Wirth
† 1 Jan. 2024

Pascal é uma linguagem de programação imperativa e procedural, projetada por Niklaus Wirth (esquerda abaixo) em 1968-69 e publicada em 1970, como uma linguagem pequena e eficiente destinada a incentivar boas práticas de práticas de programação usando programação estruturada e estruturação de dados. Um derivado conhecido como Object Pascal, projetada para programação orientada a objetos, foi desenvolvida em 1985. O nome da linguagem foi escolhido para homenagear o matemático, inventor da primeira calculadora: Blaise Pascal (veja o canto superior direito).

Editora: PRO PASCAL FOUNDATION em colaboração © Foundation Supporting Programming Language Pascal

CONTRIBUINTES

Stephen Ball
http://delphiaball.co.uk
DelphiABall

Dmitry Boyarintsev
dmitry.living @ gmail.com

Michaël Van Canneyt
,michael @ freepascal.org

Marco Cantù
www.marcocantu.com
marco.cantu @ gmail.com

David Dirkse
www.davdata.nl
mail: David @ davdata.nl

Benno Evers
b.evers @
everscustomtechnology.nl

Bruno Fierens
www.tmssoftware.com
bruno.fierens @ tmssoftware.com

Holger Flick
holger @ flixments.com

Mattias Gärtnernc-
gaertnma@netcologne.de

Max Kleiner
www.softwareschule.ch
max @ kleiner.com

John Kuiper
john_kuiper @ kpnmail.nl

Wagner R. Landgraf
wagner @ tmssoftware.com

Vsevolod Leonov
vsevolod.leonov@mail.ru

Andrea Magni
www.andreamagni.eu andrea.
magni @ gmail.com
www.andreamagni.eu/wp

Helmut Elsner
Korrektor der Deutschen
Ausgabe
helmut.elsner@live.com

Paul Nauta PLM Solution
Architect CyberNautics
paul.nauta @ cybernautics.nl

Kim Madsen
www.component4developers.com
kbmMW

Boian Mitov
mitov @ mitov.com

Jeremy North
jeremy.north @ gmail.com

Detlef Overbeek
- Editor in Chief
www.blaisepascal.eu
editor @ blaisepascal.eu

Anton Vogelaar
ajv @ vogelaar-electronics.com

Danny Wind
dwind @ delphicompany.nl

Jos Wegman
Corrector / Analyst

Siegfried Zuhr
siegfried @ zuhr.nl

Editor-chefe
Detlef D. Overbeek, Holanda Tel: Celular: +31 (0)6 21.23.62.68
Notícias e comunicados à imprensa somente por e-mail para editor@blaisepascal.eu

As assinaturas podem ser feitas on-line em **www.blaisepascalmagazine.eu** ou por ordem escrita, ou enviando um e-mail para **office @ blaisepascal .eu.** As assinaturas podem ser iniciadas em qualquer data. Todas as edições publicadas no ano civil da assinatura também serão enviadas. As assinaturas têm duração de 365 dias. As assinaturas não serão prorrogadas sem aviso prévio. O recibo de pagamento será enviado por e-mail. As assinaturas podem ser pagas enviando o pagamento para: **ABN AMRO Ban**k Account no. 44 19 60 863 ou por cartão de crédito ou **PayPal.** Nome: **Pro Pascal Foundation** (Stichting Ondersteuning Programeertaal Pascal)
IBAN: NL82 ABNA 0441960863 BICABNANL2A VAT/NL814254147B01
Departamento de Assinaturas Edelstenenbaan 21 / 3402 XA IJsselstein, Holanda + 31 (0) 6 21.23.62.68

SUBSCRIÇÕES 2024 preços
Edição eletrônica para download (8 por ano) ±60 páginas : R$ 250

Do seu editor

Olá leitores do Brasil,
Em nome do Pascal User Group of the Netherlands (editor desta revista), damos-lhe as boas-vindas. Entendemos que muitos desenvolvedores de origem brasileira estão interessados em ferramentas de desenvolvimento para Pascal e, é claro, nos últimos desenvolvimentos.

Até agora, traduzir corretamente o conteúdo da nossa Revista para o Brasil tem se mostrado um desafio, mas graças ao apoio de uma empresa internacional de software GDK, estamos tentando fazer isso da melhor maneira possível.
Especialmente pela **Bruna Elen da Silva**, que corrige a ortografia, e pelo **Ricardo Boaro** (MVP do Delphi), que analisa o significado correto em relação ao Pascal.
Bruna Elen está nos ajudando como revisora de nossa publicação e estamos muito felizes com isso.
No entanto, é claro que haverá erros de linguagem e esperamos que você nos perdoe, mas também que os repasse para que possamos aprender com eles.

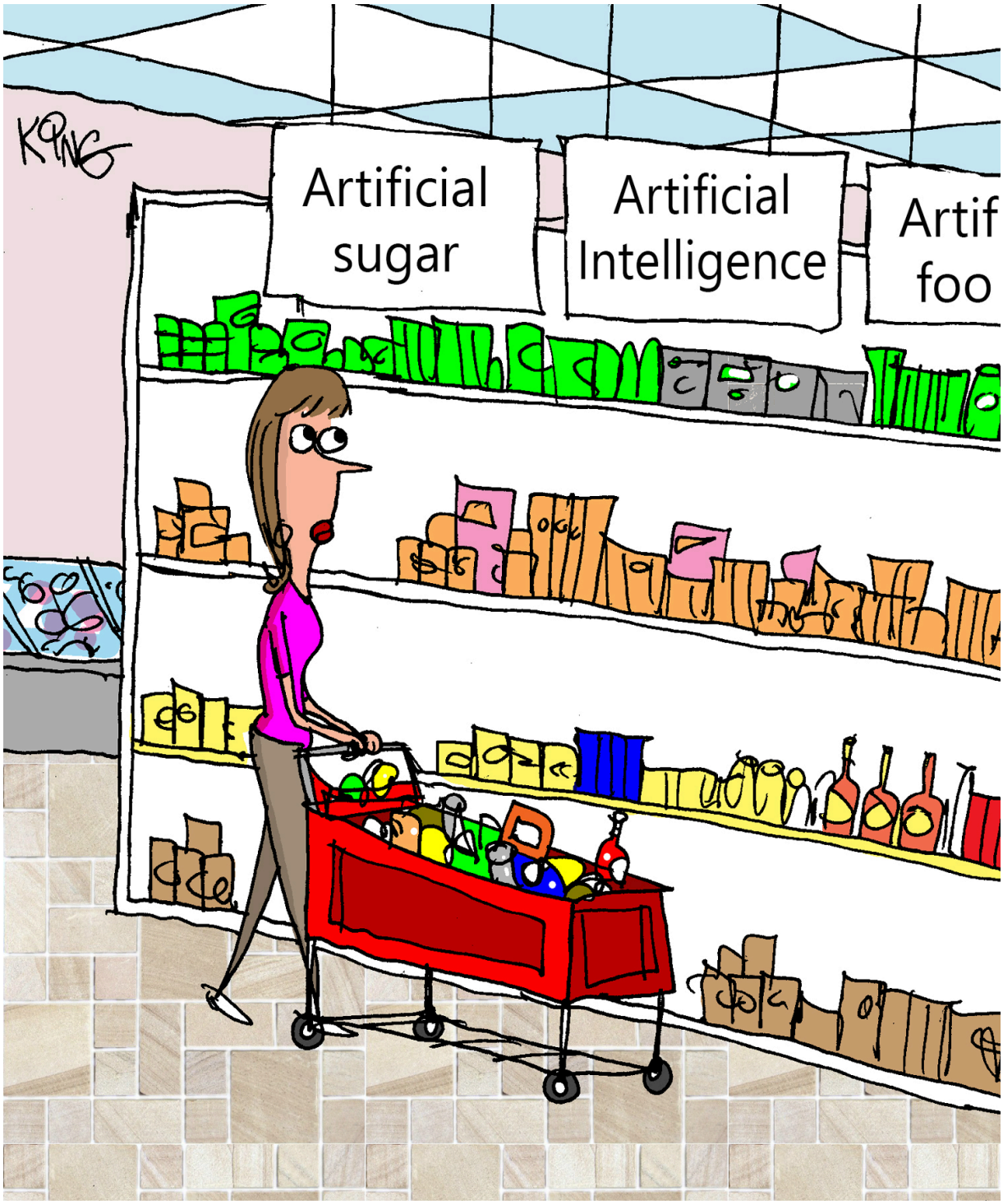
Muitos artigos contêm códigos ou projetos que são discutidos durante a implementação e você pode baixá-los gratuitamente no que diz respeito a este assunto - (114/115 da Blaise Pascal Magazine). Veja o endereço no final.
A versão brasileira desta edição é totalmente gratuita para que você possa ter uma impressão do que temos a dizer sobre a linguagem de programação Pascal do nosso grupo de usuários.
Isso significa que escrevemos sobre o Delphi como uma variante comercial e sobre a variante OpenSource FPC/Lazarus. Acompanharemos todos os desenvolvimentos de ambos os programas e até mesmo tentaremos fazer com que suas necessidades sejam conhecidas pela gerência de ambos.

POR QUE QUEREMOS FAZER ISSO?
Porque acreditamos que a linguagem de programação Pascal é a melhor linguagem para ensinar os jovens a programar. Porque queremos permitir que o maior número possível de pessoas - agora ou no futuro - possam fazer disso sua profissão.
Acreditamos que um grande número de programadores é necessário para melhorar nossa sociedade e nosso meio ambiente, desenvolver-se mais e melhor no futuro. As pessoas que se desenvolverem por meio desse idioma desenvolverão, assim, um bom nível de abstração e capacidade analítica, o que aumentará muito suas chances no mercado de trabalho, seu desenvolvimento pessoal crescerá e, por meio de uma melhor compreensão social, será de grande valor para o Brasil.
Não se trata de uma promessa, mas de uma realidade, e isso já ficou bem evidente em muitos países.
Portanto, gostaríamos de envolver as crianças nesse processo o mais cedo possível.
E isso deve ser possível, é claro, tornando os programas disponíveis sem custo. O Lazarus/FPC é de código aberto e, portanto, totalmente gratuito, e o Delphi tem um produto gratuito que você, como usuário, pode usar sob certas condições. (Consulte a página 120 desta edição)
Por enquanto, o Lazarus é um programa de desktop, mas pretendemos que, no futuro, você não precise instalá-lo, mas usá-lo via Internet - onde quer que esteja.
O grupo de usuários e a equipe do Lazarus estão trabalhando intensamente nisso.

Além das questões gerais de programação, também destacamos questões sobre tecnologia, RaspberryPI, Compilação cruzada / Desenvolvimento de software quântico / Técnicas de Internet / Bancos de dados e muitos outros tópicos, como IA (Inteligência Artificial) e todo o seu impacto social, suas ameaças, seu potencial para melhorar nosso ambiente de vida e suas muitas possibilidades de aplicação.
O futuro é seu.

Haverá muito a explicar sobre tudo isso, mas, por enquanto, desejo a você muito prazer na leitura.
E, é claro, esperamos que você apoie nossa iniciativa assinando o site.
(Consulte a página 47 desta edição ou em nosso site **`https://blaisepascalmagazine.eu`**)

De nosso consultor técnico, Jerry King



RESUMO

suporte ao **FPCUnit** no **Lazarus** recebeu uma atualização:
framework de teste de unidade **FPCUnit** agora pode se comunicar diretamente com o **Lazarus IDE** tornando ainda mais fácil a correção de erros em seu código

❶ INTRODUÇÃO

O **Free Pascal** tem uma estrutura de testes unitários há quase 20 anos: O FPCUnit está sendo usado para o teste de muitos dos pacotes incluídos no **Free Pascal.** Ele é mais ou menos compatível com o `DUnit`, a estrutura do **Delphi** de unit test framework: **Vanilla**, código de teste escrito para o **Dunit** será compilado com o **Free Pascal**.

O Lazarus IDE tem alguns assistentes para criar um caso de teste FPCUnit e iniciar um FPC unit test program. Podem ser criados dois tipos de programa de teste: um programa de teste de console e um programa de teste de **GUI**. A primeira mostra os resultados de todos os testes no console.

O segundo usa uma janela da **GUI** com uma visualização em árvore para mostrar todos os testes e o resultado dos testes. Esses dois programas têm a mesma desvantagem:
eles não interagem com o **IDE**.

Seria muito melhor se você pudesse clicar no nome do teste e ser levado à implementação desse teste no IDE. Ou, se houver um rastreamento de pilha, no local do erro e ser levado para o local onde o erro é gerado.

Bem, agora você pode: Na branch trunk do Lazarus. O pacote LazFPCUnit foi estendido com o "**Test Insight**".
A ideia para esse recurso foi tirada do plug-in Delphi com nome semelhante de **Stefan Glienke**.

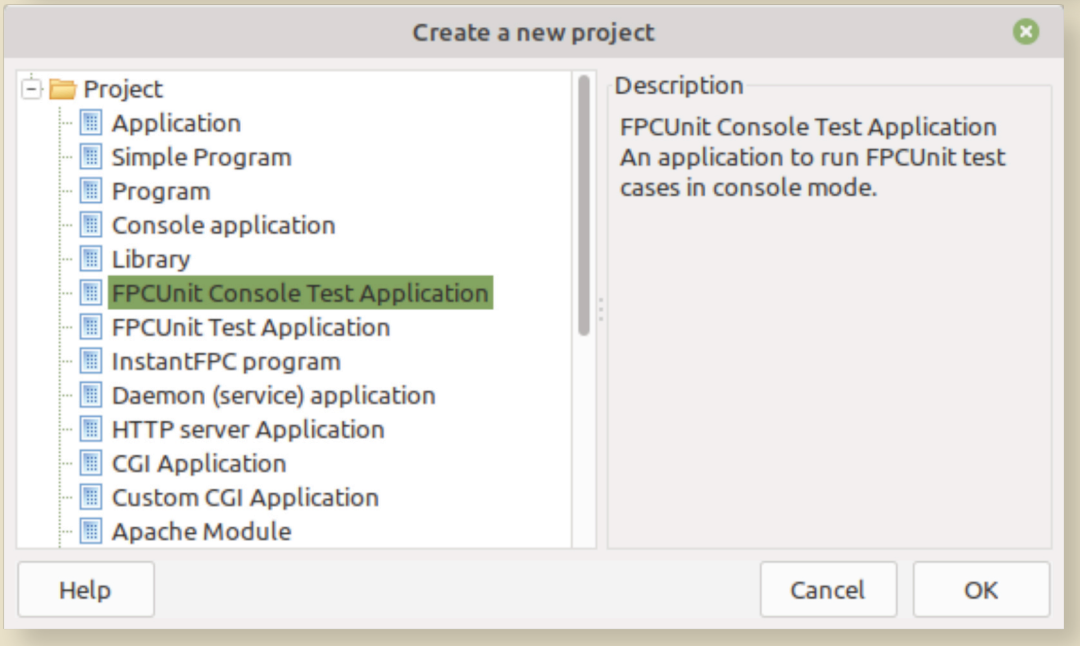


Figura 1: Criação de um programa de console do FPCUnit

2 ARQUITETURA

O sistema de **FPCUnit** unit test funciona como todos os outros sistemas de teste. Há um registro de teste e teste escutáveis. Quando os testes do registro são executados, o progresso e os resultados são relatados pelos listeners registrados.
Os Listeners do console apenas gravam no console em um dos vários formatos (você pode registrar o seu próprio), enquanto o **UI listener** adiciona node a uma Treeview.
O Listener do **testinsight** envia os resultados com solicitações HTTP para um servidor.

O local desse servidor pode ser especificado nos códigos-fonte ou em um pequeno arquivo de configuração. O suporte do TestInsight no Lazarus IDE inicia um pequeno servidor HTTP que escuta as solicitações HTTP com os resultados do teste.

O servidor http é iniciado quando você abre a janela "Test insight" no menu "View - Test insight" no do Lazarus IDE. Quando o programa de teste existe, ele o inicia com uma opção especial, para obter uma lista de testes. Quando você solicita a execução de um teste, os testes são executados e a janela exibe os resultados do teste.

Clicar duas vezes em um teste usará as ferramentas de código do **IDE** para levá-lo para o método correto no projeto de teste. Se o programa de teste não puder acessar o servidor **HTTP** do **testinsight**, ele voltará a ser executado como um programa regular do console do FPCUnit test. Você poderia fazer o mesmo com o **UI program**, se quisesse, mas essa opção não foi ativada para o programa de teste da **FPCUnit UI.**

3 USO

Para que isso funcione, você precisa ter os códigos-fonte mais recentes do IDE e, é claro, instalar o pacote lazfpcunit ele faz parte da lista padrão de pacotes do IDE. O item de menu "Novo projeto" → "Aplicativo de teste do console FPCUnit" (figura 1 na página 5) tem agora um assistente de projeto que oferece algumas opções. As seguintes opções estão disponíveis:

- **Executar todos os testes por padrão**

O programa padrão do FPCunit console mostra uma ajuda ao executá-lo sem opções de linha de comando. Se essa opção estiver marcada, o programa executará os testes quando nenhuma opção de linha de comando for especificada.

- **Formato de saída padrão**

Nessa caixa de seleção, é possível definir o formato de saída padrão para a saída do console. Você pode escolher entre XML, texto simples (com ou sem informações de tempo) ou LaTeX.

- **Usar o Testinsight para comunicar os resultados ao IDE**.

Quando essa opção é definida, os resultados do teste são enviados para o IDE com o testinsight.

- **Criar o primeiro caso de teste**

Quando marcada, o IDE inicia imediatamente o assistente "Create FPCUnit testcase" quando o projeto é criado.
A caixa de diálogo de opções é mostrada na *figura 2 na página 8.*
Depois que o projeto for criado com a opção "Use testinsight", você deverá salvá-lo e compilá-lo imediatamente. Isso permitirá que o suporte ao testinsight do IDE obtenha uma lista de testes.
Depois que o projeto for compilado, você poderá abrir a janela do Testinsight. A janela tem a mesma aparência que o gráfico FPCUnit test program - o que não é surpreendente, pois foi copiada dessa janela. Mas alguns elementos foram adicionados e ela se comporta de forma diferente da janela original.
Quando a janela for aberta, ela tentará executar o projeto atual com as opções apropriadas para obter uma lista de testes, e o resultado aparecerá com a Figura 3 na página 3 do artigo, se tudo tiver ocorrido bem.

O botão "Refresh" (Atualizar) (à esquerda) pode ser usado para atualizar a lista de testes, se você quiser. A lista de testes será atualizada em cada caso quando o programa de teste for realmente executado.
Quando o projeto ativo for alterado, a lista de testes também será atualizada automaticamente:
O caminho do executável do atual FPC unit test é sempre mostrado na barra de status na parte inferior da janela.

3 USO (CONTINUIDADE)

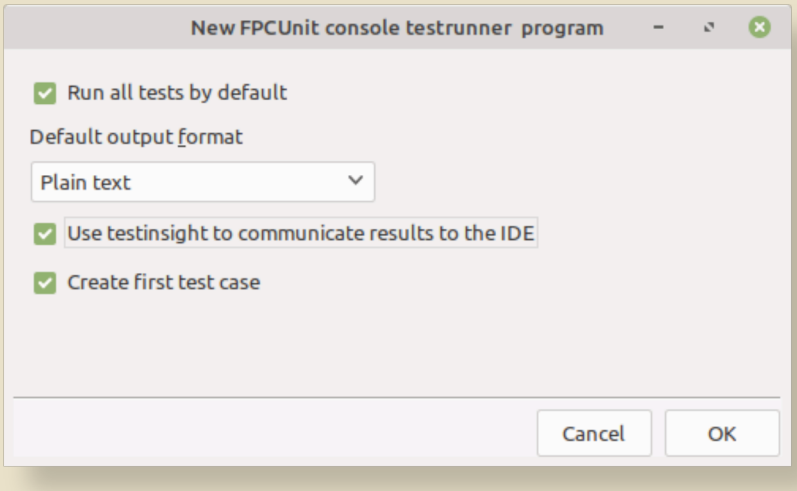


Figura 2: Opções do programa do console do FPCUnit

Quando você clicar no botão **"Executar todos os testes - Run all tests by default"** (*a seta dupla verde*), o programa de teste será executado. À medida que os resultados dos testes forem chegando, o indicador de status colorido na frente de cada teste mudará de cor de acordo com o resultado do teste. Se houver um erro, mais informações serão exibidas nos nós abaixo do nó de teste, conforme mostrado na *figura 4 na página 8*. Ao clicar duas vezes no teste ou em um nó de erro, **ele tentará localizar o código do teste e abrirá o arquivo de origem**. Na maioria dos casos, isso funcionará sem problemas. Entretanto, o mecanismo pode falhar em alguns casos:

❶ O projeto atualmente ativo no IDE não é um projeto de unit test.
❷ Os códigos-fonte do teste não fazem parte do projeto.
❸ Se você estiver usando alguns recursos mais avançados da estrutura de teste (*criando testes dinamicamente*). O **TestInsight** supõe que os nomes dos testes são nomes de métodos de uma `classe`.

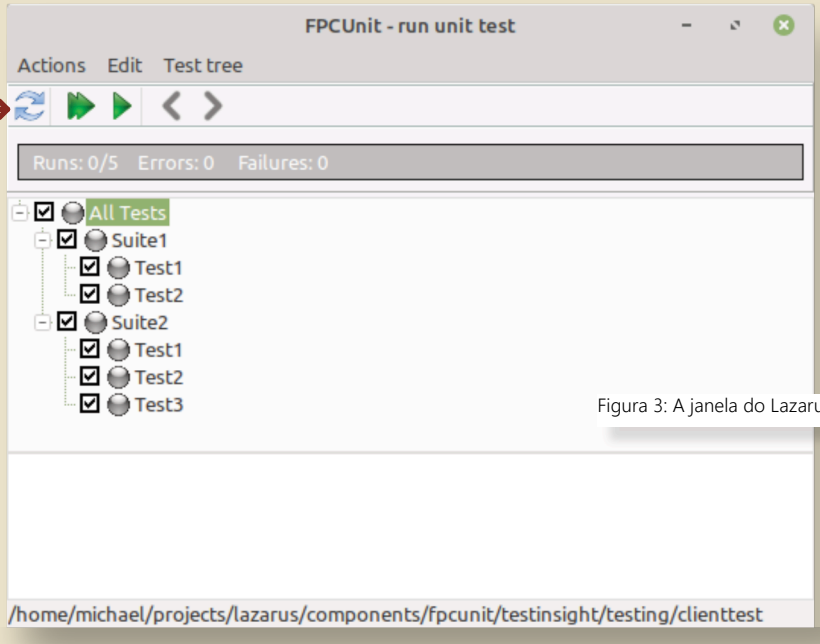


Figura 3: A janela do Lazarus Test Insight

4 CONVERSÃO DE UM PROGRAMA DE TESTE FPCUNIT EXISTENTE

Para usar o **TestInsight**, o assistente "**Novo programa**" não pode converter um programa existente para usar o **Testinsight**. Se você tiver um **programa de teste FPCUnit** existente que deseja converter para que ele use o testinsight, será necessário fazer algumas alterações no arquivo principal do projeto. O **programa típico** do **console de teste do FPCUnit** tem a seguinte fonte de arquivo de projeto principal:

```
program clienttest;
{$mode objfpc}{$H+}

uses
  Classes, jsonparser, consoletestrunner, tcTests;

type
  TMyTestRunner = class(TTestRunner)
  end;

var
Application: TMyTestRunner;

begin
  Application := TMyTestRunner.Create(nil);
  Application.Initialize;
  Application.Title := 'FPCUnit Console test runner';
  Application.Run;
  Application.Free;
end.
```

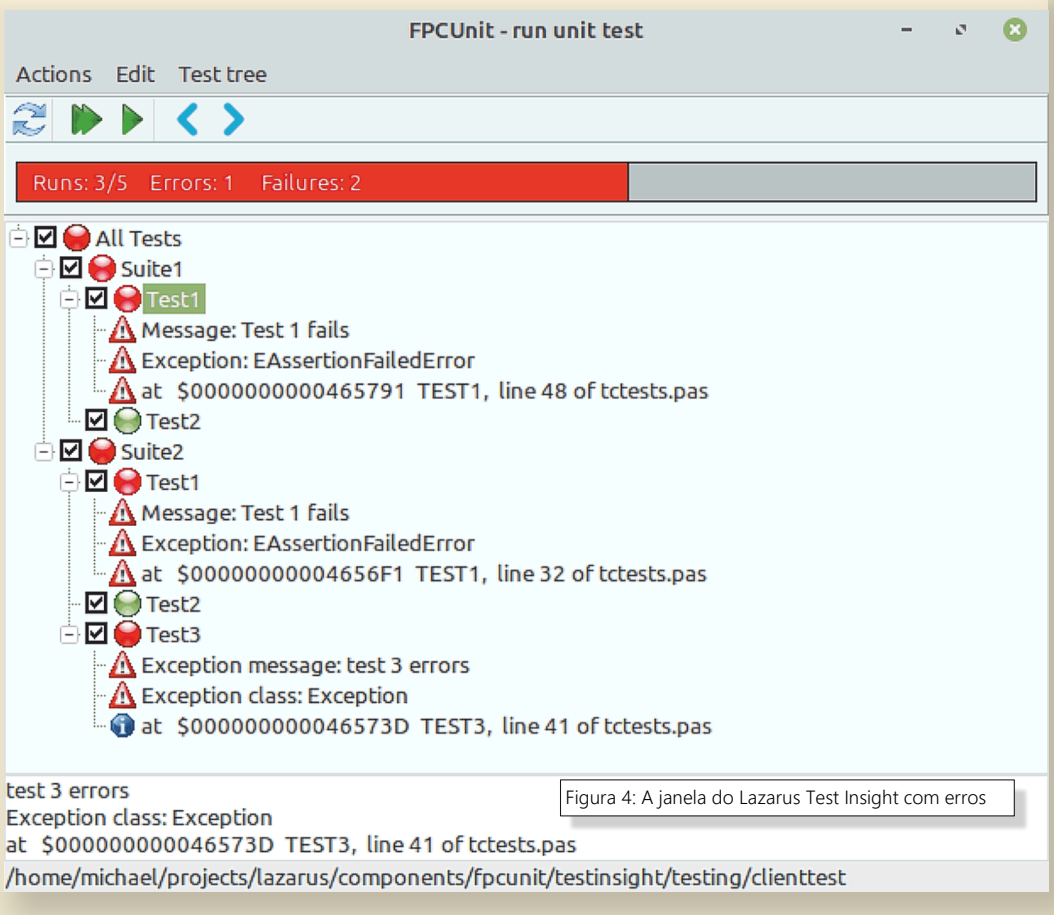


Figura 4: A janela do Lazarus Test Insight com erros

Duas coisas devem ser feitas com esse código de projeto:

❶ A unidade `fpcunittestinsight` deve ser adicionada à cláusula uses.

❷ A função `IsTestinsightListening` deve ser chamada.
Se ela retornar `True`, a rotina **`RunRegisteredTests`** deverá ser chamada. Ambos são implementados na unidade `fpcunittestinsight`.

Se IsTestinsightListening retornar false, o código original deverá ser executado.
As duas funções a serem usadas são declaradas da seguinte forma:

```
procedure RunRegisteredTests(aConfig : String = '';
baseUrl: string = DefaultUrl);

function IsTestinsightListening(aConfig : String = '';
baseUrl: string = DefaultUrl) : Boolean;
```

O argumento `config` é o nome de um arquivo `.INI` com as configurações para a execução do teste.
Por padrão, esse arquivo é `TestInsightSettings.ini`, que é o que o **IDE** usa. Os testes selecionados serão gravados nesse arquivo e a porta na qual o **IDE** está escutando (*The baseURL*)
O argumento **BaseURL** é a **URL** onde o servidor do **testinsight** está escutando.
Por padrão, é `http://localhost:8081/tests`, mas isso será definido pelo **IDE** no arquivo de configuração.
Você pode alterar esses padrões, por exemplo, para executar o programa de teste remotamente, mas ainda receberá os resultados localmente em seu PC de desenvolvimento.
Com essas alterações, o novo código do projeto terá a seguinte aparência:

```
program clienttest;

{$mode objfpc}{$H+}

uses
   Classes, jsonparser, consoletestrunner,
   tcTests, fpcunittestinsight;

type
   TMyTestRunner = class(TTestRunner)
   end;

var
   Application: TMyTestRunner;

begin
   if IsTestinsightListening() then
     RunRegisteredTests('','')
   else
   begin
     Application := TMyTestRunner.Create(nil);
     Application.Initialize;
     Application.Title := 'FPCUnit Console test runner';
     Application.Run;
     Application.Free;
   end;
end.
```

Figure 3: The Lazarus Test Insight window

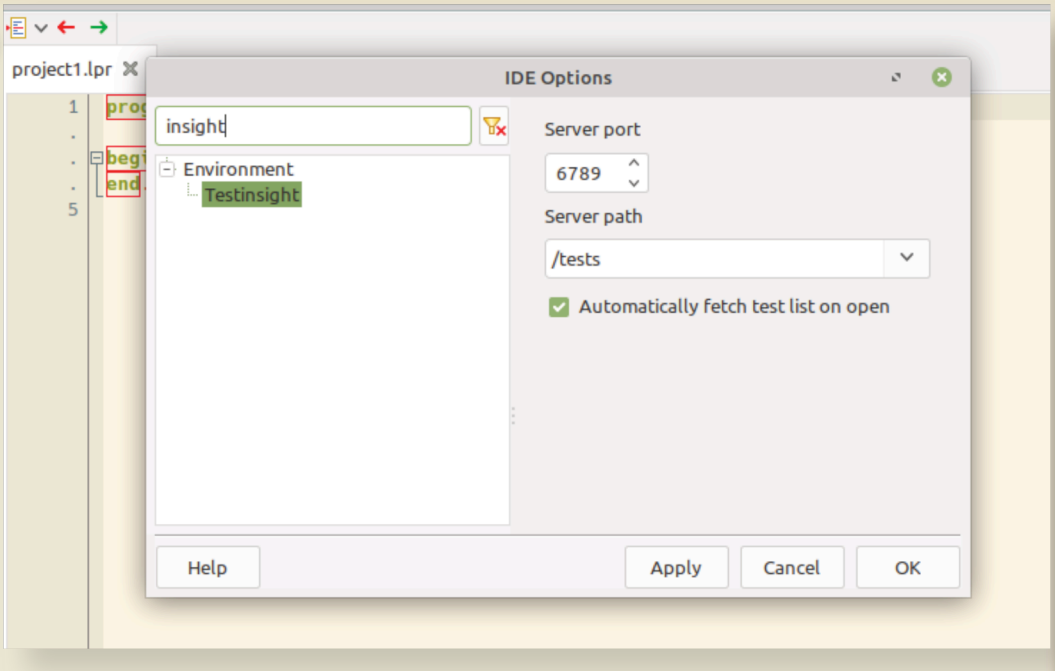


Figura 5: O painel de configuração do Lazarus Test Insight

5 CONFIGURAÇÃO

No IDE, é possível definir a porta e o local de base em que o servidor deve escutar.
No menu **"Ferramentas - Opções - Tools - Options"** do **TestInsight**, as seguintes configurações podem ser definidas:

- **Caminho base:** é o caminho no servidor HTTP para o qual as solicitações devem ser enviadas. O padrão é "/tests".
- **Porta do servidor** esse é o caminho no servidor HTTP para o qual as solicitações devem ser enviadas. O padrão é 6789
- **Busca automática** de testes Quando a janela "Test Insight" é aberta ou o projeto atual é alterado, a lista de testes é buscada automaticamente.

Quando desativada, você pode usar o botão de atualização para obter a lista de testes.

A caixa de diálogo de configurações é mostrada na *figura 5 na página 10*.

6 CONCLUSÃO

A funcionalidade "**Test Insight**" facilita fazer o **Test-Driven Development**, permitindo que você vá imediatamente para a implementação de um teste ou para um **local de erro** dentro da exibição do resultado do teste incorporada no **IDE**.

A versão atual do **"Test Insight"** é apenas **uma versão inicial:**
Algumas extensões estão planejadas, como a **filtragem de testes e a seleção automática** de testes, e o suporte **PAS2JS** do **testinsight** também está sendo desenvolvido.

EXECUTING PROGRAMS ON THE SERVER IN PAS2JS

By Michael Van Canneyt

Starter → Expert

ABSTRACT

In this article we show how to give the user of a browser-based program feedback from long-running processes on the server, using 2 components: one in PAS2JS, one in Free Pascal/Lazarus.

❶ INTRODUCTION

When using a web-based program, not everything can be done in the browser.
Often,tasks are executed through some RPC (Remote Procedure Call) mechanism on the webserver. This can be a simple task such as executing an SQL statement on a database and returning a result. Or it can be a more complicated and time-consuming task such as making a backup of a database, indexing PDF files, compiling software projects and running a test suite, or even installing software on the server. Ideally the output of these remote programs should be presented to the user.

To keep programs scalable, these tasks should be short-lived. A return time of 1 second for a HTTP request is already a long time. Starting a time-consuming task and waiting for the return using a single request is not a good idea:
the HTTP server is occupied with the request, the browser or any proxy servers between the HTTP server and the browser may time out waiting for your request.

Much better is to start the process using a HTTP request, and use a mechanism to poll the status of the executed process. In this article we present one such mechanism.

❷ ARCHITECTURE

The solution we present here consists of 2 components. One component which is used on the server, and which can be used to start a process, capture its output and poll for the status of the process. The other component takes care of the polling process on the client.

These components are ignorant of the communication mechanism between browser and server, this means that they do not implement the actual RPC calls used to start the process: There are many possible mechanisms, and some may be more suitable for your purpose than others.

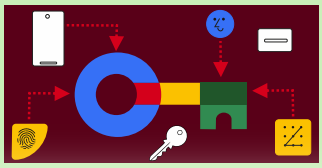
The components are called `TProcessCapture` for the server part and `TProcessCapturePoller` for the client (PAS2JS) part. The server part takes care of executing a program and redirecting the output to a file, the client part implements the polling mechanism and some callbacks to handle the actual server calls and the result. We'll demonstrate both components with a simple set of programs:

- A test program to be executed.
  It is used for demonstration purposes only.
- A HTTP server program that allows to serve HTML files and that offers an
- RPC mechanism to start the test program and handle status requests. A Simple PAS2JS program that will run in the browser and which will remotely execute the test program. It will show the output of the test program in the browser.

We'll start with the test program.

Com o Higlight o resultado na pesquisa

SEM SENHA LOGIN COM "PASSKEYS"



ABSTRATO
Neste artigo, quero explicar os aspectos da criação e do uso de passkey. Parece um pouco complicado, mas se você seguir os objetivos principais, é uma maneira muito interessante e útil de obter um serviço que proporcionará segurança e conveniência.

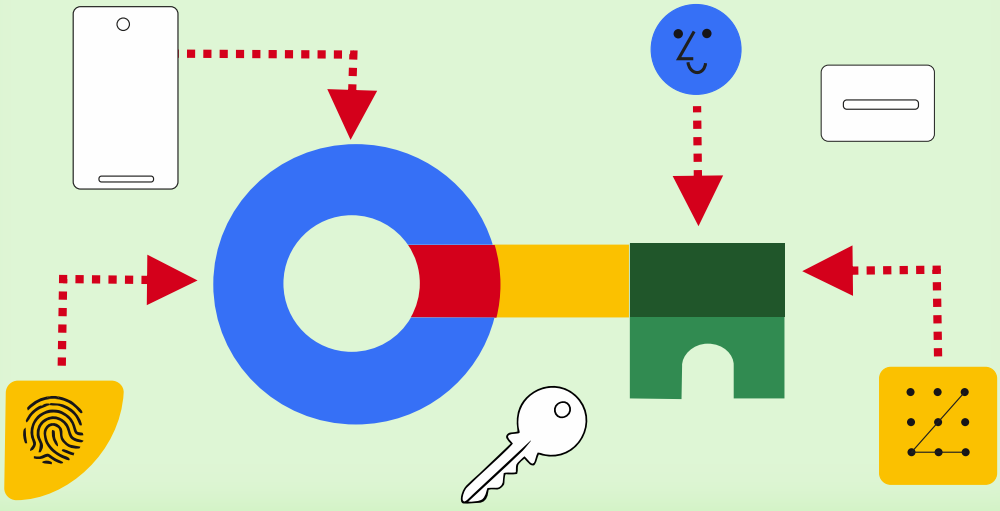
PASSKEYS (CHAVES DE ACESSO) : O QUE SÃO??
Uma **passkey** é um tipo de credencial digital que está conectada a um aplicativo ou site e a uma conta de usuário. *Uma credencial é um documento que detalha uma qualificação, competência ou autoridade emitida para um indivíduo por um terceiro com uma autoridade relevante.*
As **passkey** permitem que os usuários façam login sem exigir outros fatores de autenticação ou a inserção de um nome de usuário e senha.
As senhas e outros métodos de autenticação antiquados devem ser substituídos por essa tecnologia.

❶ VANTAGENS
As passkey permitem que você faça login em serviços da Web de forma rápida e segura:
- **sem uma senha,**
- **sem hardware especial**
- **sem o risco de phishing**.

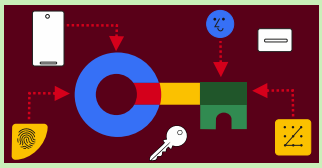
Tentarei explicar como esse sucessor de senhas funciona e onde você pode usar as passkey com seu PC, (*desktop para ser mais preciso*), smartphone e tablet.

DESVANTAGENS QUE LEVAM A VANTAGENS
Tanto os desenvolvedores quanto os usuários não gostam de senhas:
- elas proporcionam uma experiência ruim para o usuário,
- elas aumentam o atrito na conversão,
- elas criam uma responsabilidade de segurança tanto para os usuários quanto para os desenvolvedores.

O **Password Manager** (*do* **Google**) no **Android** e no **Chrome** reduz o atrito por meio do preenchimento automático; é bom para os desenvolvedores que buscam melhorias ainda maiores na conversão e na segurança, as passkey e a federação de identidade são as melhores abordagens modernas para desenvolvedores.
Uma passkey pode atender aos **requisitos de autenticação multifator** em uma única etapa, substituindo ambos: uma senha e **OTP - One Time Password** - (*por exemplo, código SMS de 6 dígitos*) para oferecer proteção robusta contra-ataques de phishing, evita a experiência do **UX* - SMS** inconvenientes (**Short Message/Messaging Service**) ou senhas de uso único baseadas em aplicativos.
***The U**ser e**X**perience **(UX)** é como o usuário se sente ao interagir com um produto, sistema ou serviço.



SEM SENHA LOGIN COM "PASSKEYS"



VANTAGENS
Continuação

Como as **passkeys** *(passkey)* são padronizadas, uma única implementação permite uma experiência sem senha em todos os dispositivos dos usuários, em diferentes navegadores e sistemas operacionais.
Bem no meio, onde esses componentes se encontram, você encontrará a chave.
Essa sobreposição é a maneira como que o seu site deve ser criado, estruturado e projetado para formar a experiência perfeita do usuário.

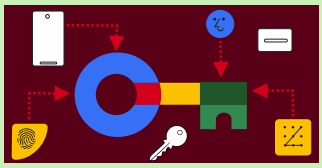
As passkeys são mais seguras

- **Os desenvolvedores salvam apenas uma chave pública no servidor** em vez de uma senha, o que significa que há muito menos valor para um agente mal-intencionado invadir os servidores e muito menos limpeza a ser feita no caso de uma violação.
- As passkey protegem os usuários contra-ataques de phishing.
  As passkey funcionam apenas nos sites e aplicativos registrados;
- **Um usuário não pode ser induzido** a se autenticar em um site enganoso porque o navegador ou o sistema operacional lidam com a verificação.
- As passkey reduzem os custos de envio de SMS, tornando-as um meio **mais seguro e econômico** para a autenticação de dois fatores.
- Muitos processos complexos são executados discretamente para garantir a segurança.
  Mas essas atividades passam despercebidas pelas pessoas.

Para você, acessar sua conta com uma passkey é tão simples quanto tocar no sensor de impressão digital do seu smartphone, por exemplo.
Lembre-se de que uma mente criativa criminosa pode facilmente fazer um carimbo de borracha.
A maior conveniência poderia convencer indivíduos que normalmente não se preocupam com a segurança a usar a nova técnica de login. com a segurança a usar a nova técnica de login.

❸ CRIMINOSOS
Todos os dias, os criminosos cibernéticos assumem o controle de milhões de contas, geralmente por meio de senhas. As análises de senhas vazadas mostram que muitos usuários tendem a escolher senhas simples. E, além disso usam-nas para vários serviços da Web.
A autenticação de dois fatores, protetora, mas incômoda, é frequentemente ignorada por conveniência. Não se pode culpar ninguém por isso. Com o tempo, as contas de sites e aplicativos se proliferam porque é preciso se registrar em quase todos os lugares atualmente.
Se você seguir consistentemente as práticas recomendadas para se proteger, terá muito trabalho a fazer. Não é um problema para pessoas com experiência em tecnologia, mas tente explicar isso para pessoas maiores de idade.
O gerenciamento eficaz de senhas pode ser um desafio devido ao fato de o conceito ser anterior à Internet, está próximo dos dinossauros e até mesmo muito antes da invenção do primeiro computador.
Seu objetivo original não incluía a intenção de deter hackers ou impedir tentativas de phishing.
Portanto, a utilização de medidas como a autenticação de dois fatores é vital para que se continue empregando o procedimento obsoleto na era contemporânea.

PHISHING
Phishing refere-se à prática fraudulenta de tentar obter informações confidenciais, como senhas ou detalhes de cartão de crédito, disfarçando-se como uma entidade confiável.
An EFF Electronic Frontier Foundation fez alguns comentários
`https://www.eff.org/deeplinks/2023/10/what-passkey`



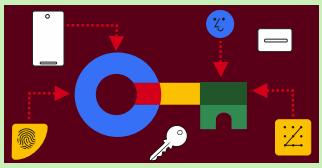
PHISHING - CONTINUAÇÃO
As passkey incluem informações sobre o nome de domínio específico para o qual foram geradas. Se uma pessoa lhe fornece uma URL para uma página de login em um nome de domínio muito parecido com o original, você poderá ser enganado. Seu navegador da Web NÃO será enganado, pois ele tem a capacidade de verificar sem esforço se há uma correspondência exata. Para garantir sua segurança, seu navegador NÃO permitirá a transmissão da passkey para o nome de domínio enganoso.
No entanto, desde que você mantenha uma senha memorizada junto com a chave de acesso, um site fraudulento poderá enganá-lo, alegando que sua passkey não está funcionando e solicitando que você digite a senha.
Nesse caso, a inserção da senha resultará na execução bem-sucedida do ataque de phishing.
O phishing continua sendo uma ameaça viável, mas as pessoas que costumam usar uma passkey para acessar um determinado site são mais propensos a ficar cautelosos quando solicitados a inserir uma senha. Isso oferece um grau de proteção, se não for absoluto.

EXEMPLO DO PAYPAL
A autenticação engloba inerentemente o domínio do site. Isso aumenta a resistência das passkey contra tentativas de phishing.
Se você acessar o **`PayPal.com`** usando um e-mail de phishing, não poderá utilizar a passkey anteriormente estabelecida para o **`PayPal.com`**.
Como alternativa, você só poderá gerar uma nova passkey para o domínio alternativo.
Isso é totalmente inútil para os phishers. Sem dúvida, ainda é imperativo que você não confie suas informações confidenciais ao site de phishing.
Veja a oferta do PayPal na próxima página.

NASCE UMA ESTRELA
Uma nova era está surgindo com a introdução das passkeys.
Sem a necessidade de senhas e sem os possíveis perigos associados a ataques de phishing
A tecnologia necessária já está presente em seu smartphone, tablet e computador.
Vários serviços da Web já estão equipados para esse novo avanço.

The Logo for Passkeys

SEM SENHA LOGIN COM "PASSKEYS"



do 30-11-2023 09:01

service@paypal.nl

Ready to improve your 2-step verification?

To Detlef Overbeek

If there are problems with how this message is displayed, click here to view it in a web browser.

Hello, [redacted]

We've got better ways to safeguard your account

Your 2-step verification can be even more effective, and still easy.

SMS codes get the job done, but what if we had other simple options that are even more secure? Well, there's good news — we do!

Download an authenticator app

Authenticator apps help you easily log in with a code just like you do with SMS codes, but they feature much stronger security benefits:

- Codes typically refresh within a minute, making it much harder to hack
- It works without mobile or internet coverage so codes can't be rerouted
- Just open the app and your code is ready (no waiting for a text!)

If you want the familiar feel of SMS codes plus all the added security, an authenticator app may be perfect.

Need one? The Google Authenticator, Microsoft Authenticator, and Authy are some popular options.

Get a physical security key

Want extra security that's extra simple? A security key is exactly that for a few key reasons:

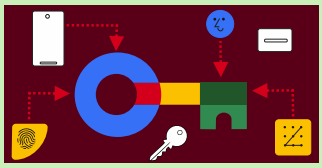
- It's a physical device so it can't be rerouted to another device
- If you lose it, the key has no identifiable info about who it belongs to
- Enter it into your USB slot or hold it near your device and that's it

Of course, you would need to buy a security key. If you need one, we support the YubiKey and other Yubico keys.

You received this email because you've set up optional 2-step verification. Learn about 2-step verification.

Improve Your 2-Step Verification

SEM SENHA LOGIN COM "PASSKEYS"



❷ COMO FUNCIONA?

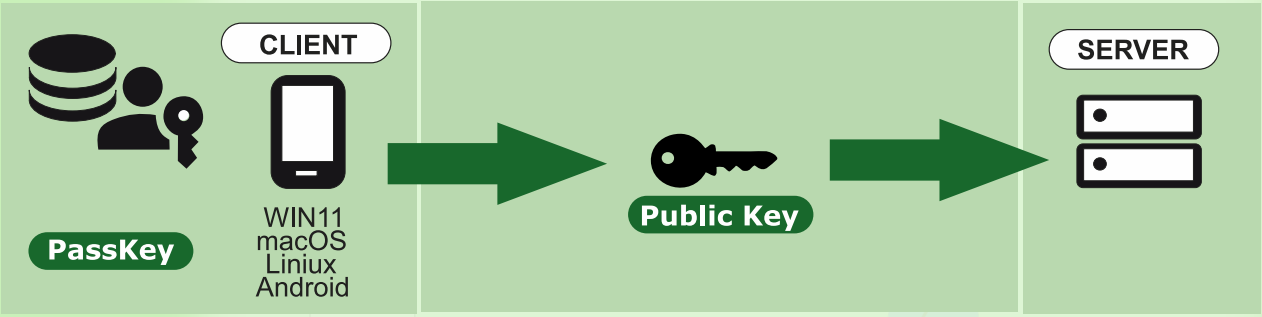
AUTENTICAÇÃO POR CHAVE DE ACESSO

Ao contrário das senhas, com a autenticação por passkey não há segredo compartilhado conhecido por você e pelo serviço que um invasor possa interceptar.
Em vez disso, você tem uma **chave privada e uma chave pública** para cada de suas contas.
Somente você possui a chave privada importante.

1* Quando você cria uma passkey para um serviço da Web, o serviço obtém apenas a chave pública e a vincula à sua conta. Com a chave pública, o serviço agora pode verificar se você tem a chave privada, sem que o serviço nunca veja essa chave privada.

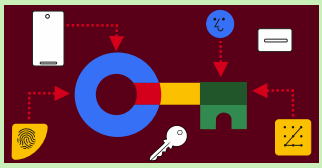
NENHUM SEGREDO COMPARTILHADO É CONHECIDO TANTO PARA VOCÊ QUANTO PARA O SERVIÇO QUE UM INVASOR PODE INTERCEPTAR

Especificamente, isso funciona da seguinte forma:
Quando você faz login por meio de uma chave de acesso, o **serviço da Web** fornece **dados aleatórios**, chamados de **desafio**, que seu dispositivo **assina com sua chave privada**.
O serviço da Web verifica a assinatura digital **e pode determinar com certeza que ela vem de sua chave privada.**
Esse método é chamado de CRIPTOGRAFIA DE CHAVE PÚBLICA.
Ele foi experimentado e testado por décadas e é usado para **HTTPS** e criptografia de e-mail, por exemplo.



A **EFF** Electronic Frontier Foundation tem alguns comentários:
Uma passkey é composta por **aproximadamente 100-1400 bytes de dados aleatórios,** gerados em seu dispositivo (*como seu telefone, laptop ou chave de segurança*) com a finalidade de fazer login em um site específico.
Depois que a passkey é gerada, o navegador a registra no site e ela é armazenada em algum **lugar seguro** (*por exemplo, seu gerenciador de senhas).* Pessoalmente, **prefiro NÃO** permitir que isso aconteça e escolher outra maneira de salvar minha chave. Lembre-se de que essa é a parte que é privada.
Veja a explicação 1 (O gerenciador de senhas não pode copiá-la para outros dispositivos).*
A partir daí, você pode usar essa **passkey para fazer login no site** sem digitar uma senha.
Ao acessar a página de login de um site, você terá a opção de "**Sign in with a passkey**". Se escolher essa opção, você receberá uma solicitação de confirmação do seu gerenciador de senhas, e será conectado após a confirmação. Para que tudo isso funcione, é necessário ter o **suporte a passkey** no site, o navegador, o gerenciador de senhas e, normalmente, **também no seu sistema operacional.**

SEM SENHA LOGIN COM "PASSKEYS"



PASSKEY INITIATION AND FLOW

1. Concordar em configurar uma chave de acesso

2. Confirme a configuração pelo dispositivo por meio de impressão digital, reconhecimento facial ou pin.

3. O aplicativo ou navegador gera uma chave privada e uma chave pública

4. ele armazena a chave privada em um ambiente seguro e a chave pública no servidor do aplicativo

5. Quando as chaves públicas e chaves privadas correspondem, a autenticação é realizada.

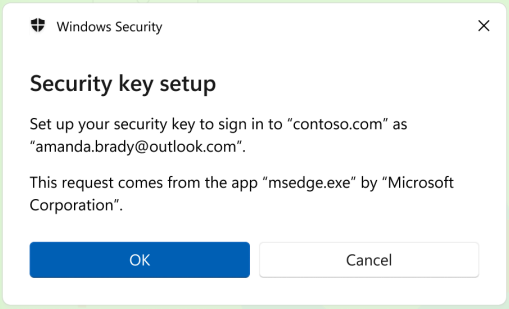
❶ Normalmente, quando os **usuários tentam fazer login** em um aplicativo que **utiliza passkey**, uma caixa de diálogo será exibida, solicitando que eles salvem uma passkey (*geralmente uma impressão digital, identificação facial ou padrão*) para fins de autenticação futura. O usuário deve verificar e continuar.

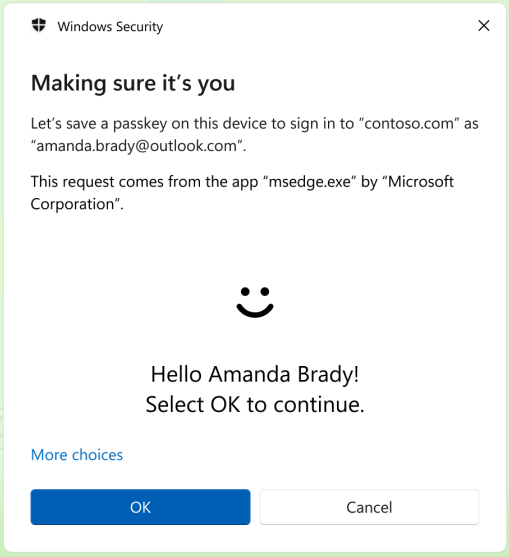
❷ Após a **concordância do usuário em utilizar a autenticação baseada em chave de acesso**, o aplicativo continuará a **gerar um par de chaves criptográficas** que consiste em uma **chave pública** e uma **chave privada**. Esse método é executado localmente no dispositivo do usuário, que **pode incluir seu computador, tablet ou smartphone**.

❸ Durante o procedimento, a **chave privada é armazenada com segurança em uma carteira**, enquanto a chave pública é **enviada ao servidor do aplicativo** para armazenamento.

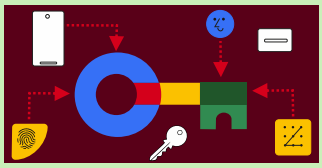
❹ Na próxima tentativa de login do usuário, o servidor de aplicativos transmitirá um **quebra-cabeça criptografado** para o dispositivo do usuário, utilizando a chave pública do dispositivo. Para verificar a identidade do usuário que está tentando fazer login, o dispositivo do usuário **decifrará** o quebra-cabeça e transmiti-lo de volta ao servidor.

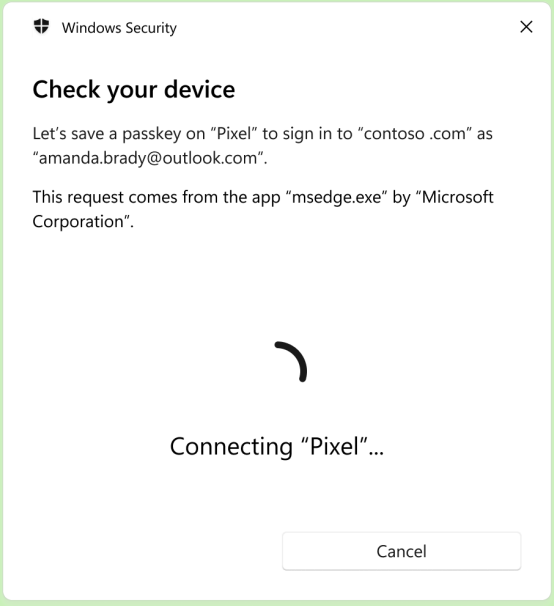
❺ Após a **conclusão do processo de verificação de correspondência das chaves pública e privada**, o usuário poderá acessar o aplicativo com êxito.

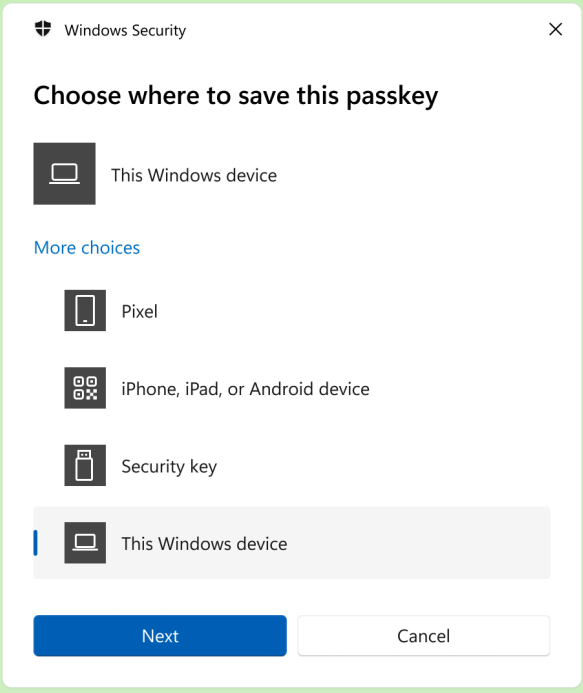


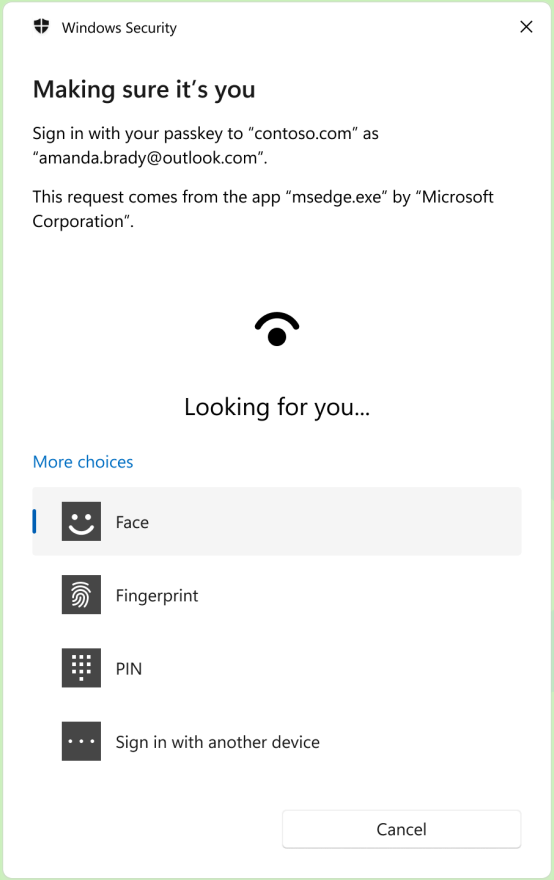


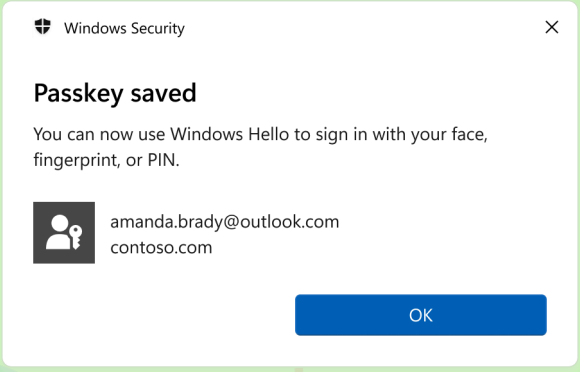
SEM SENHA LOGIN COM "PASSKEYS"

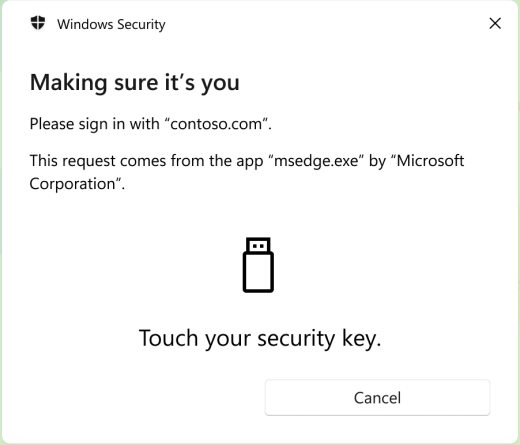




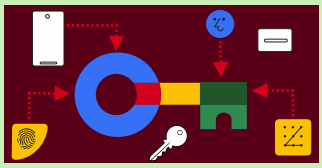








SEM SENHA LOGIN COM "PASSKEYS"

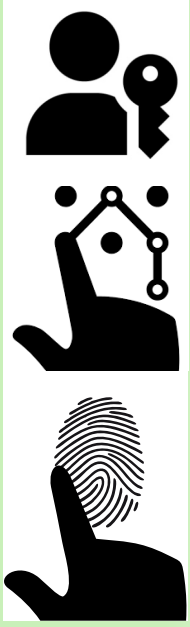


- Você pode **criar várias chaves de acesso**:

cada passkey desbloqueia uma única conta em um único site.
Para várias contas em um único site, você pode ter várias senhas para esse site. Por exemplo, se você tiver uma conta de mídia social para uso pessoal e outra para negócios, você pode ter senhas diferentes para cada conta.

- Normalmente, **você pode ter uma senha e uma passkey** em sua conta e pode fazer

login com qualquer uma delas. Geralmente, o login com uma passkey é mais rápido, pois o gerenciador de senhas se oferecerá para fazer isso em um único clique, em vez dos vários cliques que o login com uma senha geralmente requer. Além disso, o login com uma passkey geralmente permite que você ignore a autenticação tradicional de dois fatores
(SMS, aplicativo autenticador ou chave de segurança)



As Passkeys incluem um segundo fator.
Toda vez que você usar a passkey para fazer login, o navegador ou o sistema operacional poderá solicitar que você digite novamente o **PIN** de desbloqueio do dispositivo. Se você usar uma impressão digital ou reconhecimento facial para desbloquear o dispositivo, o navegador poderá solicitar que você insira novamente sua impressão digital ou mostre seu rosto, para confirmar que é realmente você que está solicitando o login.
Pessoalmente, não gosto do reconhecimento facial. Atualmente, é muito simples criar uma imagem como máscara facial

Isso proporciona dois fatores de autenticação:
O dispositivo que armazena sua passkey é algo que você tem e é acompanhado por algo que você conhece (o **PIN**) ou algo que você sabe (a senha).
Você sabe (o **PIN**) ou algo que você é (*uma impressão digital ou um rosto*).

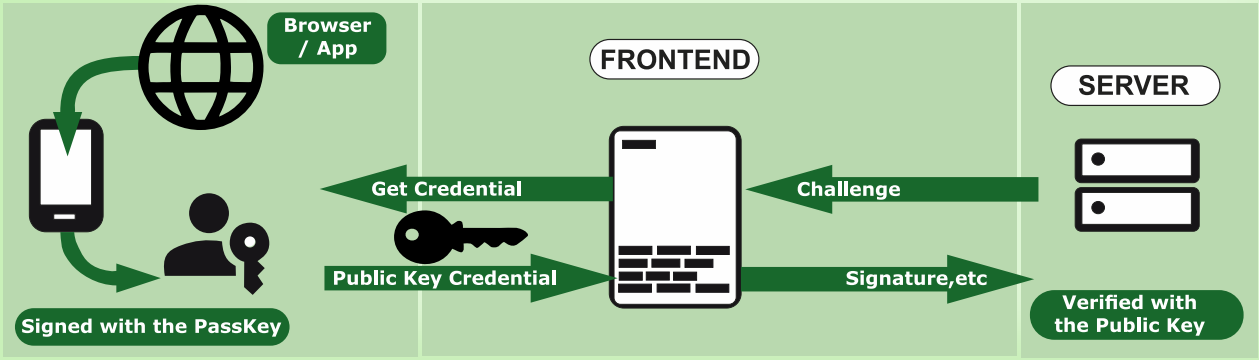
Em vez de utilizar uma senha, você receberá uma passkey exclusiva para cada conta individual.

- A **chave é salva** em seu dispositivo móvel, PC ou tablet.
- Para fazer login, basta selecionar a opção de login com senha na página apropriada.
- Verifique a utilização da passkey empregando um PIN conciso.

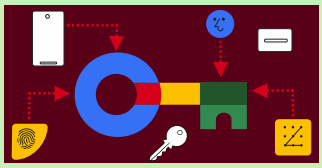
Esse procedimento é o mesmo para todas as senhas.

Pode ser potencialmente acelerado com a utilização de sua impressão digital ou digitalização facial:
Após a conclusão, seu login será feito imediatamente. *Tenha cuidado com essas características biológicas, pois elas podem ser falsificadas de forma convincente nos tempos modernos.*
O **PIN** ou os dados **biométricos** fornecidos por você são utilizados exclusivamente no local para acessar a passkey e **nunca são transmitidos para outro local**.
A passkey é automaticamente protegida por um segundo fator, sendo que o primeiro fator é a própria chave de acesso. Tudo isso sem a necessidade de autenticação de **dois fatores separada e onerosa.**

Uma passkey é **essencialmente** um par **de chaves criptográficas** que é **produzido automaticamente por um chip de segurança** em seu computador ou smartphone.
A exigência de gerar uma nova passkey para cada registro, a fim de cumprir os requisitos do site, se tornará obsoleta em um futuro próximo.



SEM SENHA LOGIN COM "PASSKEYS"



ARMAZENAMENTO E BACKUP
Uma passkey armazenada em apenas um computador ou telefone não é muito útil.
É possível fazer login em um dispositivo diferente?
E se o seu dispositivo for roubado, perdido ou se desfizer?

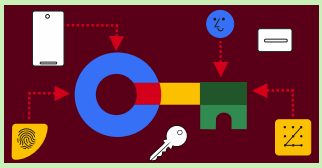
Há várias respostas aqui para armazenamento e/ou backup,
mas cada uma delas é uma história diferente
Esse é um motivo para que as chaves de acesso sejam vistas como uma opção especial para resolver as circunstâncias, mas acho que elas estão em conflito em sua estratégia...

❶ você poderia armazená-las no gerenciador **de senhas**, que as criptografa,
o que as torna vulneráveis - se você conseguir quebrar o armazenamento...
e depois faz o backup delas na nuvem.
Ele ajuda a copiá-las em todos os seus dispositivos. Não acredito nem confio na "nuvem".
Na verdade, é um nome de marketing para algo antigo:
o computador por meio de uma conexão de Internet ou cabo.
Está fora de suas mãos. Também é contra a regra não permitir que ninguém tenha a chave.

❷ há sempre a maneira antiga de guardá-lo em uma lista de itens e mantê-lo em segurança em seu escritório, mas é claro que ela pode ser encontrada e roubada.
Você terá que adicionar manualmente a chave ao dispositivo quando for solicitado

❸ Há também uma **solução física:** um **pendrive USB**
As chaves de acesso são criadas e armazenadas em uma chave de segurança física que você conecta via USB3. Para fazer login em outro dispositivo, você conecta a chave de segurança quando solicitado. Isso pode ser feito até mesmo com seu celular.
As chaves de acesso criadas dessa forma não podem ser copiadas.
Somente chaves de segurança de fabricação recente suportam isso.

❹ As chaves de acesso são criadas e armazenadas em um chip de alta segurança incorporado ao seu computador ou telefone
(*por exemplo, um TPM ou Secure Enclave, disponível na maioria dos dispositivos fabricados nos últimos anos*).
Um **TPM** (**Trusted Platform Module**) é usado para aumentar a segurança de seu PC.
Ele é usado por serviços como a criptografia de unidade BitLocker, o Windows Hello e outros, para criar e armazenar chaves criptográficas com segurança e para confirmar que o sistema operacional e o firmware em seu dispositivo são o que deveriam ser e não foram adulterados.
Um enclave seguro
Os dispositivos de computação modernos geralmente têm um chip de hardware especial dedicado a armazenar informações de importância crítica, como chaves de criptografia e hashes.
Nos PCs, esse chip é o **Trusted Platform Module** (**TPM**), enquanto nos dispositivos móveis, como **Androids** e **iPhones**, ele é chamado de Secure Enclave.



Como no ponto ❸, essas chaves de acesso não podem ser copiadas.
As respostas 3 e 4 são menos convenientes (e a resposta 3 custa algum dinheiro: comprar uma chave de segurança). No entanto elas oferecem um nível mais alto de segurança contra o roubo de seus dispositivos. Com a resposta 1, alguém que roube seu computador poderá copiar as senhas se o gerenciador de senhas estiver desbloqueado.

Além disso, as soluções 3 e 4 não resolvem realmente o problema do "dispositivo perdido".
Se estiver usando uma dessas respostas, você deve ter várias senhas armazenadas em diferentes dispositivos como backup. **Como alternativa, você pode acabar contando com a recuperação de conta baseada em e-mail.**

É ACONSELHÁVEL UTILIZAR CHAVES DE ACESSO?
Como em outras questões relacionadas à segurança e à privacidade, a resposta depende de vários fatores. Entretanto, as senhas são geralmente recomendadas para a maioria das pessoas.
Se você estiver utilizando um **gerenciador de senhas**, criando senhas longas e distintas para cada site e emprega consistentemente a funcionalidade de preenchimento automático para fazer login (em vez de copiar e colar manualmente as senhas), as senhas oferecerão um nível de proteção um pouco mais elevado com uma facilidade consideravelmente maior.

Se você não estiver usando um gerenciador de senhas no momento, a implementação de chaves de acesso aprimorará significativamente suas medidas de segurança (*e exigirá a adoção de um gerenciador de senhas*).

Ao usar a autenticação de dois fatores (**2FA**) em sites, as senhas oferecem maior conveniência e segurança potencialmente aprimorada.
As soluções de **2FA** via **SMS** ou aplicativo autenticador são suscetíveis a ataques de phishing, pois sites fraudulentos podem solicitar o código de uso único do usuário e, posteriormente, transmiti-lo, juntamente com a senha falsificada, para o site legítimo.

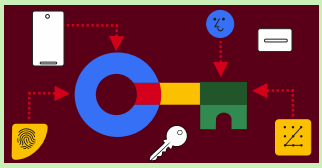
As **passkeys** oferecem segurança aprimorada em comparação com a **2FA** por **SMS** ou aplicativo autenticador devido à sua imunidade a ataques de phishing.
Ao contrário do **SMS** ou dos aplicativos autenticadores, as senhas não são suscetíveis de serem enganadas por sites fraudulentos, pois o navegador pode associar com precisão cada passkey ao site correspondente.

A **passkey 2FA** não é suscetível a ataques de phishing, portanto, a transição da chave de segurança **2FA** para uma senha oferece principalmente conveniência ao eliminar uma etapa adicional de login e a necessidade de lembrar de uma senha extra.

Ao armazenar suas chaves de acesso em uma chave de segurança protegida por um **PIN** ou **autenticação biométrica**, você pode obter resultados comparáveis ao uso da autenticação de dois fatores (**2FA**).

O armazenamento de senhas em um gerenciador de senhas reduz a segurança até certo ponto, pois pessoas não autorizadas que obtêm acesso ao gerenciador de senhas podem utilizar as senhas sem precisar posse real da chave de segurança.

SEM SENHA LOGIN COM "PASSKEYS"



Até 2023, o nível de suporte para passkeys é altamente inconsistente, especialmente quando se trata de **sincronização**.

Adam Langley (*engenheiro sênior da equipe do Google*) fornece exemplos das limitações de diferentes sistemas de gerenciamento de senhas. Ele afirma que o **Windows Hello** não sincroniza de forma alguma, o Google Password Manager só sincroniza entre dispositivos **Android**, e o **iCloud Keychain** funciona exclusivamente em dispositivos Apple.

Mesmo depois de resolver esses problemas, a sincronização de dados entre diferentes ecossistemas (como **iOS** e **Windows**) continuará a ser um desafio significativo. Organizadores de senhas de terceiros, como 1Password, Bitwarden e Dashlane, oferecem a funcionalidade de passkey e podem sincronizar dados em várias plataformas. No entanto, deve-se observar que nem todas as plataformas são atualmente compatíveis com esses serviços. Por exemplo, a partir de outubro de 2023, o 1Password não oferece suporte completo para chaves de acesso no Android.

Se quiser experimentar as passkeys em uma conta descartável,
você tem a opção de criar uma em **`passkeys.io`** ou **`webauthn.io`**.

Se você gosta de estar na vanguarda dos avanços tecnológicos, recomendo que experimente as passkeys. Durante sua jornada, você poderá encontrar obstáculos e ser forçado a recorrer ao método antigo e contestado de usar uma senha.

Os usuários podem selecionar uma conta para fazer login.
Não é necessário digitar o nome de usuário.

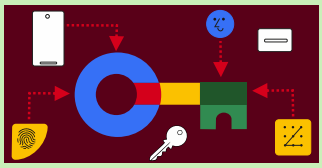
- Os usuários podem se autenticar usando o bloqueio de tela do dispositivo, como um sensor de impressão digital, reconhecimento facial ou PIN.
- Depois que uma passkey é criada e registrada, o usuário pode mudar facilmente para um novo dispositivo e usá-lo imediatamente sem precisar se registrar novamente (*ao contrário da autenticação biométrica tradicional, que exige configuração em cada dispositivo*).

Quando um usuário deseja fazer login em um serviço que usa chaves de acesso, o navegador ou o sistema operacional o ajudará a selecionar e usar a passkey correta. A experiência é semelhante à forma como as senhas salvas funcionam atualmente.

Para garantir que somente o proprietário legítimo possa usar uma chave de acesso, o sistema solicitará o desbloqueio do dispositivo.
Isso pode ser feito com um sensor biométrico, conforme explicado anteriormente, um **PIN** ou um padrão. Você pode escolher uma base não biológica ou um **PIN** ou padrão para criar uma passkey para um site ou aplicativo, o usuário deve primeiro se registrar nesse site ou aplicativo.

❶ Acesse o site/aplicativo e faça login usando o método de login existente.
❷ Clique no botão Create a passkey (Criar uma chave de acesso).
❸ Verifique as informações armazenadas com a nova chave de acesso.
❹ Use o desbloqueio de tela do dispositivo para criar a chave de acesso.

Na verdade, tudo isso acontece se você usar um telefone celular.
Para o **WINDOWS**, a abordagem é diferente...



SUPORTE A CHAVES DE ACESSO NO ANDROID E NO CHROME

As senhas podem ser **sincronizadas entre dispositivos** no mesmo ecossistema, por exemplo, as passkeys criadas no Android são armazenadas no **Google Password Manager**.

OBSERVAÇÃO: A partir do **Android 14**, os usuários podem optar por usar aplicativos de gerenciamento de credenciais de terceiros para armazenar suas chaves de acesso.
As chaves de acesso são uma tecnologia emergente e os ambientes compatíveis ainda estão evoluindo. A partir de agosto de 2023, o **Chrome** no **macOS** e no **Windows** armazena as chaves de acesso somente no **dispositivo loca**l.

GERENCIADOR DE SENHAS DO GOOGLE

O **Gerenciador de senhas** do **Google** armazena, fornece e sincroniza senhas no **Android** e no Chrome. As senhas do Gerenciador de senhas do **Google** estão disponíveis para todos os aplicativos **Android**, incluindo o **Chrome** e outros navegadores.

Quando o usuário cria uma senha em um dispositivo **Android**, ela é armazenada e sincronizada com seus outros dispositivos **Android**, e os segredos da senha são **criptografados de ponta a ponta**. Isso torna as chaves de acesso disponíveis para o usuário em todos os dispositivos Android que usam o Google Password Manager e que estejam conectados com a mesma Conta do Google.
O Gerenciador de senhas do **Google** no **Chrome** ajuda a criar e fazer login com as chaves de acesso.

Dependendo do sistema operacional do desktop (por exemplo, **Chrome OS, iOS, macOS, Windows**), os usuários podem receber um **QRCode** para usar com segurança uma passkey armazenada em seu dispositivo móvel ou uma notificação pode ser exibida solicitando que o usuário desbloqueie o telefone para usar a passkey relevante.

SUPORTE À CHAVE DE ACESSO DO CHROME EM DIFERENTES SISTEMAS OPERACIONAIS

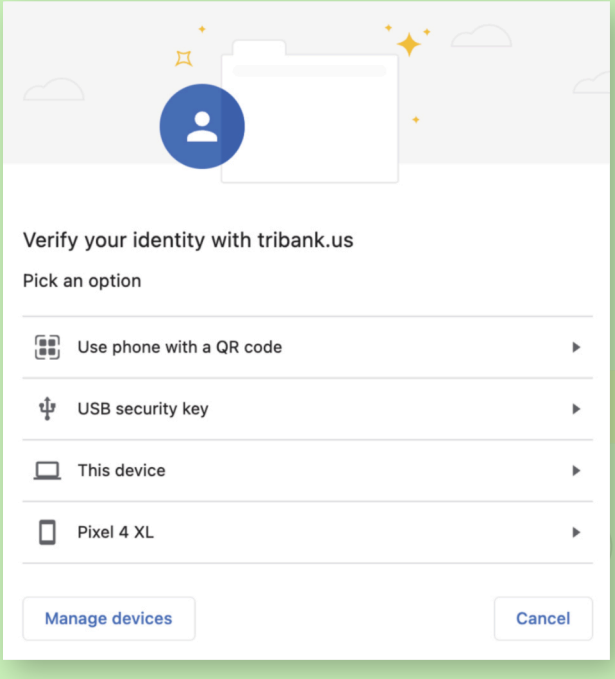
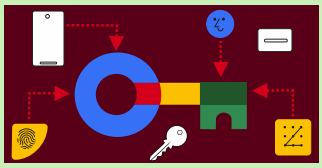


Figura 1: Seletor de autenticador

SEM SENHA LOGIN COM "PASSKEYS"



O Chrome em todas as plataformas de desktop suporta o uso de senhas de dispositivos móveis. Para usar uma passkey de seu dispositivo **Android** ou **iOS**, selecione a opção apropriada quando solicitado. Consulte *a Figura 1: na página 12 do artigo*.
Para saber mais sobre como usar um telefone para fazer login, leia Fazer login com um telefone.
https://developers.google.com/identity/passkeys/use-cases#sign-in-with-a-phone
As seções a seguir descrevem o comportamento do **Chrome** em diferentes sistemas operacionais.

ANDROID

O **Chrome** no **Android OS 9** ou posterior suporta passkeys. As senhas geradas no Chrome no Android são armazenadas no **Gerenciador de senhas do Google.** Essas **senhas** estão disponíveis em todos os outros dispositivos Android desde que o **Gerenciador de senhas do Google** esteja disponível e a mesma **Conta do Google** do usuário esteja conectada.

WINDOWS

O Chrome no **Windows** armazena as chaves de acesso no **Windows Hello**, que **não as sincroniza** com outros dispositivos a partir de **outubro de 2023**.
Quando um usuário tenta fazer login em um site pela primeira vez no **Chrome** no **Windows**, ele deve escanear um código **QR** com outro dispositivo que já tenha uma chave de acesso.
Depois disso, ele pode criar uma passkey no dispositivo Windows local para uso futuro.

MACOS

O **Chrome** no **macOS 13.5** e posterior pode usar o **iCloud Keychain** para armazenar senhas.
As chaves de acesso no **iCloud Keychain são** sincronizadas entre os dispositivos Apple do usuário e podem ser usadas por outros navegadores e aplicativos. O **Chrome** no **macOS** também pode armazenar senhas em um perfil local, o que significa que elas não são sincronizadas para outros dispositivos. O armazenamento de chaves de acesso em um perfil local está disponível em versões anteriores do **macOS**.

IOS / IPADOS

O Chrome no **iOS 16** e no **iPadOS 16** usa o **iCloud Keychain** para armazenar as chaves de acesso.
As chaves de acesso no **iCloud Keychain** são sincronizadas entre os dispositivos **Apple** do usuário e podem ser usadas por outros navegadores e aplicativos.

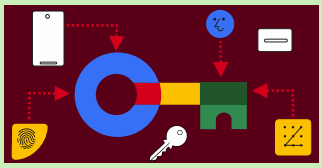
LINUX

O Chrome no Linux **não oferece suporte a senhas com um autenticador de plataforma integrado.**
Os usuários do Linux podem **usar passkeys de outro dispositivo,** como um **telefone Android** ou um iPhone, ao escanear um **QRCode.**

Operating System	Android	macOS	iOS/iPadOS	Windows	Linux
Local user verification	☑	☑	☑	☑	⊘
Passkey sync	☑	☑[1]	☑[1]	🕐[3]	⊘
Autofill	☑	☑	☑	☑[2]	⊘
Can Sign with a mobyle	☑	☑	☑	☑	☑

☑:Supported, 🕐: Planned, ⊘: No plans
[1]:Syncs with iCloud [2]: Requires Windows 11 22H2 [3]:Depends on Windows Hello

SEM SENHA LOGIN COM "PASSKEYS"



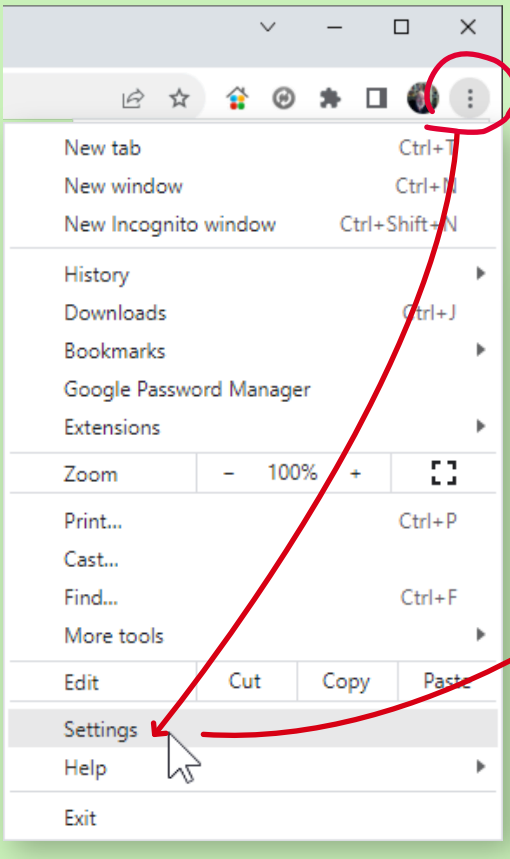


Figura 1 Botão e configurações

Para encontrar as configurações corretas, siga estas etapas:

- Abra seu navegador → no canto superior direito, você verá um botão com **três pontos**. Se você clicar nele, a janela se abrirá.
- Você verá **Configurações**.
- Clique nesse botão e a próxima página será exibida:
- Aqui está **Segurança**. Escolha **Segurança**: uma nova janela pequena é exibida. Aqui você precisa clicar em Gerenciar sua conta

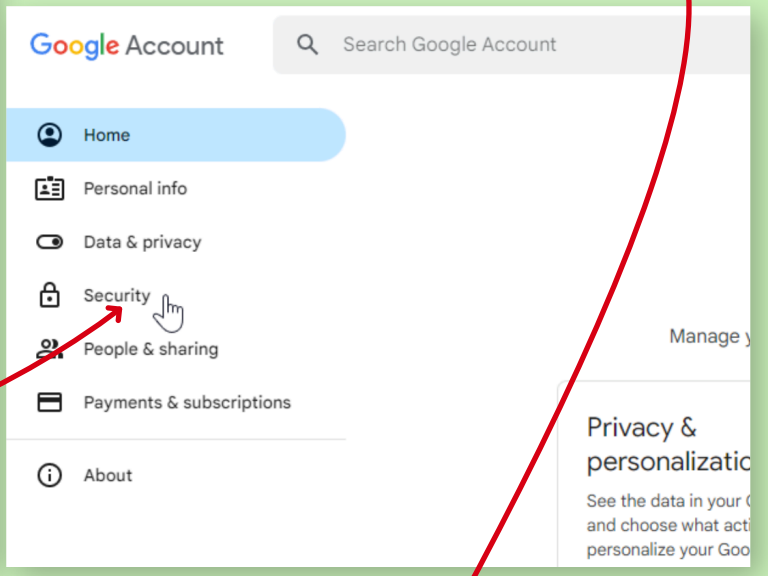


Figura 2 A segurança deve ser escolhida

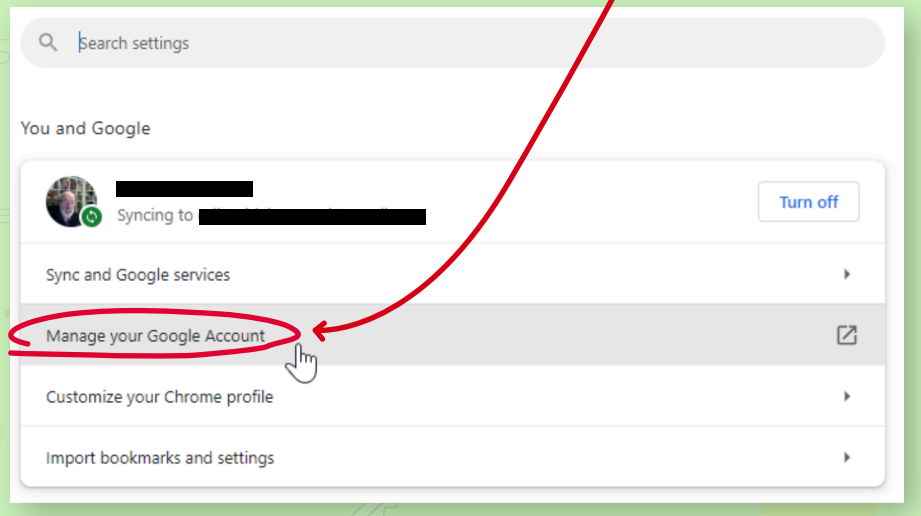
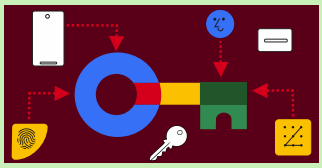


Figura 3 Você precisa gerenciar sua conta antes de fazer a configuração da chave de acesso



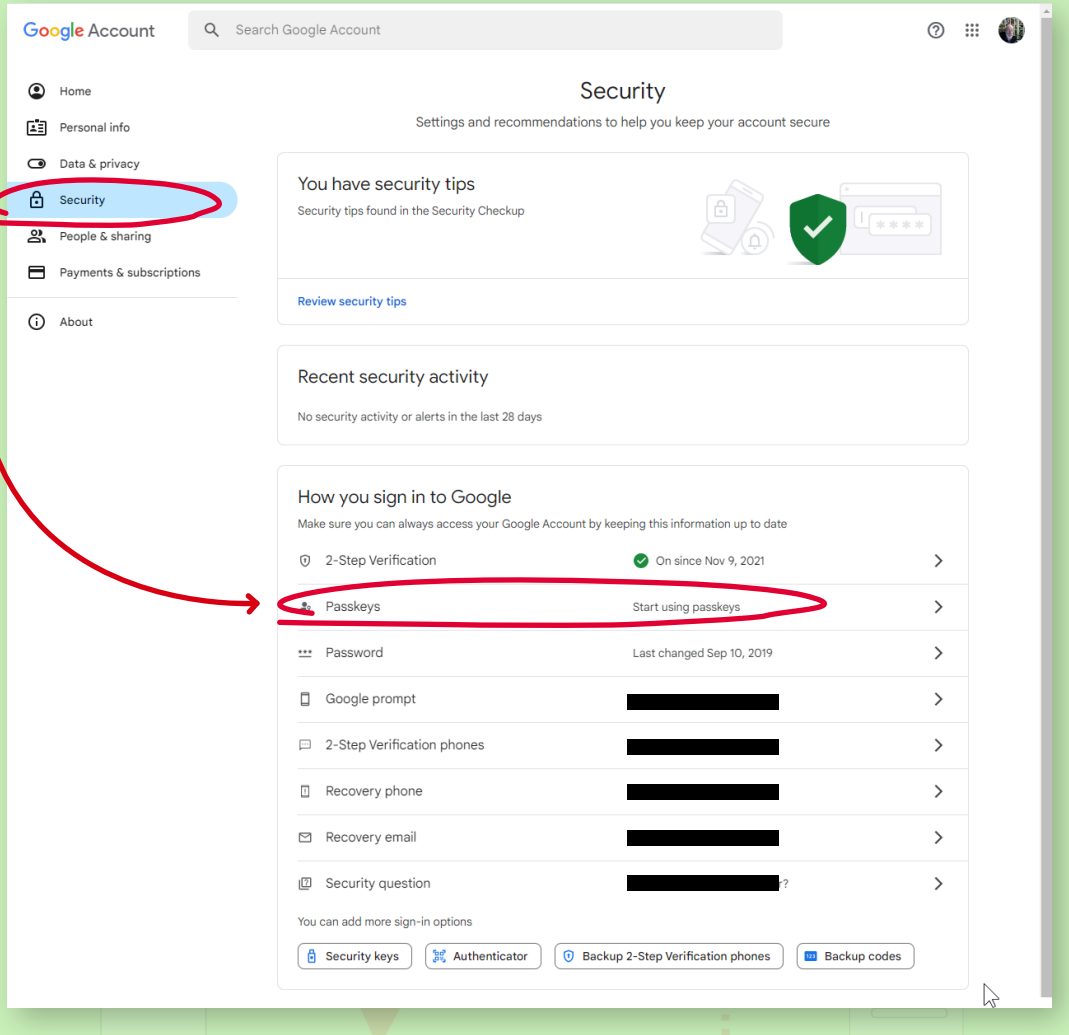
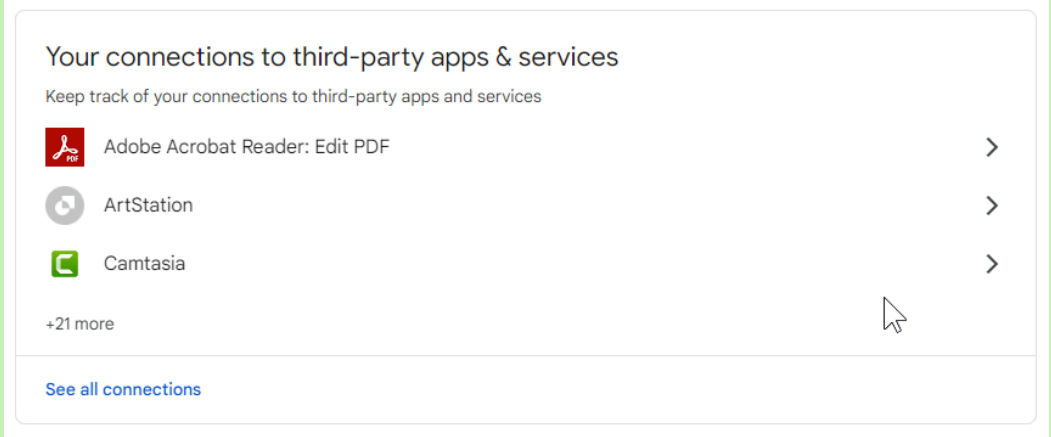
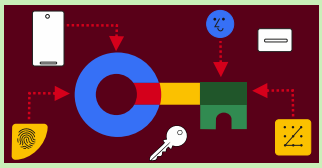


Figura 4 A seleção de segurança mostrará uma lista de itens: Escolha as chaves de acesso



SEM SENHA LOGIN COM "PASSKEYS"



Passkeys.directory

passkeys.directory

All Bookmarks

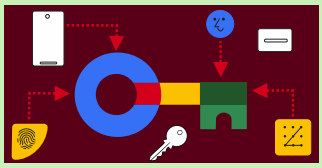
Passage Want to support passkeys in your app or website? Passage by 1Password can help. Get started

Passkeys.directory

Passkeys.directory is a community-driven index of websites, apps, and services that offer signing in with passkeys.

Provided by 1Password

DocuSign docusign.com	Sign In	MFA	Information Technology	Details
eBay ebay.com	Sign In		E-Commerce	Details
FormX.ai formx.ai	Sign In		Information Technology	Details
FusionAuth fusionauth.io	Sign In		Authentication Provider	Details
GitHub github.com	Sign In	MFA	Information Technology	Details
Google google.com	Sign In	MFA	Information Technology	Details
haeppie haeppie.com	Sign In		Information Technology	Details
Hancock hancock.ink	Sign In		Information Technology	Details
Hanko.io hanko.io	Sign In		Authentication Provider	Details
Horizon Pics horizon.pics	Sign In		Information Technology	Details
Instacart instacart.com	Sign In		eCommerce	Details
Intastellar Accounts intastellaraccounts.com	Sign In		Information Technology	Details
KAYAK kayak.com	Sign In		Travel & Leisure	Details



PERGUNTAS FREQUENTES

- As chaves de acesso funcionam em dispositivos que não têm um método de bloqueio de tela configurado?
  Isso depende da implementação do gerenciador de senhas, se um provedor de credenciais permite a criação e a autenticação de uma **passkeys** em um desafio de fator de conhecimento do usuário.
  Os provedores podem solicitar que os usuários configurem um **PIN** ou bloqueio de tela biométrico antes de criar uma senha..
- Como as chaves de acesso registradas em uma plataforma (*como o Android*) podem ser usadas para fazer login em outras plataformas (*como Web ou iOS*)?

  Uma passkey registrada no **Android**, por exemplo, pode ser usada para fazer login em outras plataformas ao conectar o telefone Android a outro dispositivo.

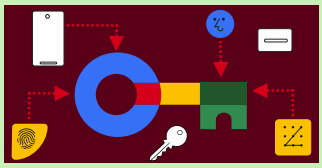
`https://developers.google.com/identity/passkeys/use-cases#sign-in-with-a-phone`

Para estabelecer uma conexão entre os dois dispositivos, os usuários precisam abrir o site que estão tentando fazer login em um dispositivo que não tenha uma passkey registrada, escanear um **QRCode** e, em seguida, confirmar o login no dispositivo em que criaram a **passkey** (*neste caso, o dispositivo* **Android**).

A CHAVE DE ACESSO NUNCA SAI DO DISPOSITIVO ANDROID

Portanto, normalmente os aplicativos sugerem a criação de uma nova passkey no outro dispositivo para facilitar o login na próxima vez.
Esse fluxo também funcionará de maneira semelhante em outras plataformas.

- Posso mover as chaves de acesso sincronizadas de um provedor de plataforma para outro?
  As chaves de acesso são salvas no provedor de credenciais definido pela plataforma.
  Algumas plataformas, como o Android, permitem que os usuários escolham o provedor de sua preferência (*um sistema ou gerenciador de senhas de terceiros*) a partir do *Android 14*, que pode ser capaz de sincronizar as senhas em diferentes plataformas.
  O suporte para mover as chaves de acesso diretamente de um provedor de plataforma para outro não está disponível no momento.
- Um usuário pode sincronizar suas **passkeys** em dispositivos **Android** que não sejam do **Google**?
  As senhas são sincronizadas somente dentro do ecossistema do dispositivo (*ou seja, de* **Android** *para* **Android** *com o Gerenciador de senhas do* **Google** *por padrão*), mas não em todo o ecossistema.
  O Android está abrindo a plataforma (*a partir do Android 14*) para permitir que os usuários selecionem qual provedor de credenciais que desejam usar (*como um gerenciador de senhas de terceiros*).
  Isso possibilitará casos de uso como a sincronização de senhas entre diferentes ecossistemas (*dependendo do grau de abertura de outras plataformas*).
- O que os desenvolvedores devem fazer com relação a **dispositivos e plataformas que não suportam chaves de acesso**?
  Recomenda-se que os desenvolvedores mantenham as opções de login existentes em seus aplicativos por enquanto para que elas continuem disponíveis para dispositivos e superfícies que não suportam passkeys.



- Uma RelyingParty pode especificar uma conta para o usuário fazer login?
  As partes confiáveis (*aplicativos de terceiros*) podem preencher o "**allowCredentials**" com uma lista de IDs de credenciais enviadas pelo backend do aplicativo, indicando quais chaves de acesso devem ser usadas para autenticar o usuário.

  OBSERVAÇÃO: a primeira etapa para ativar o suporte a chaves de acesso para seu aplicativo **Android** é associar seu aplicativo e o site.

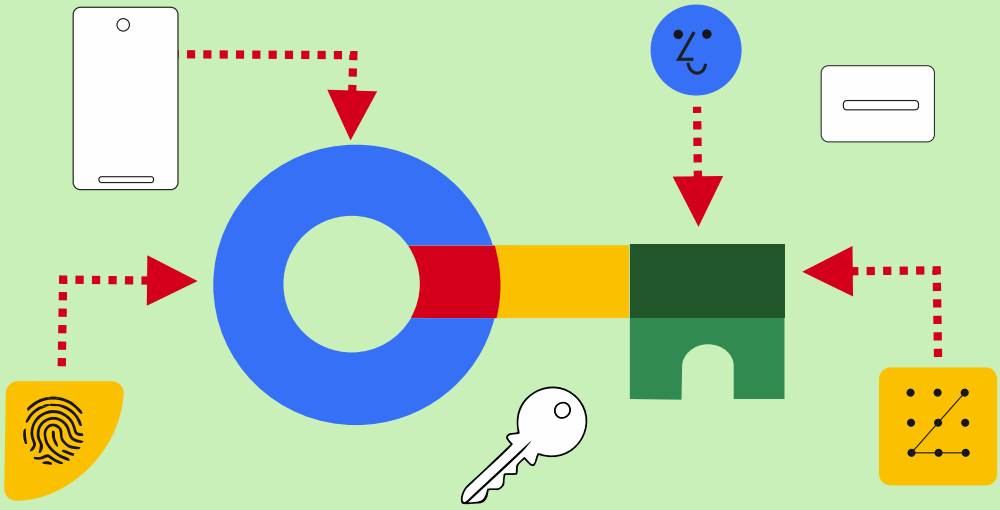
  Para isso, hospede um arquivo **JSON de Digital Asset Links** em seu site e adicione um link para o arquivo **Digital Asset Link** ao manifesto do seu aplicativo. Isso demonstra que você é o proprietário do site e do aplicativo. Para saber mais, consulte a documentação sobre **Digital Asset Links**.
- Para chaves de acesso criadas no **Chrome** em outras plataformas:

Se a senha for criada no Chrome em outras plataformas (*Mac*, *iOS*, *Windows*), então não. Confira os ambientes compatíveis para obter mais informações. Enquanto isso, **os usuários podem usar o telefone no qual criaram a passkey para fazer login**.

```
https://developer.android.com/training/sign-in/passkeys
https://developers.google.com/identity/passkeys
https://developers.google.com/identity/passkeys/use-cases
```

EXPERIMENTE O SENHOR MESMO
O senhor pode experimentar o passkeys nesta demonstração:

```
https://passkeys-demo.appspot.com/
https://medium.com/androiddevelopers/bringing-seamless-authentication-
        to-your-apps-using-credential-manager-api-b3f0d09e0093
```



MORREU O PIONEIRO DA COMPUTAÇÃO NIKLAUS WIRTH

Niklaus Wirth, uma figura proeminente no campo da ciência da computação, faleceu.
AVISO DE FALECIMENTO PROCESSAMENTO DE DADOS E TECNOLOGIA DA INFORMAÇÃO
Niklaus Wirth, uma figura proeminente no campo da ciência da computação, faleceu em 1º de janeiro de 2024, com quase 90 anos de idade. O estimado professor de ciência da computação da ETH ganhou reconhecimento mundial por seu trabalho pioneiro na criação da linguagem de programação Pascal em 1970. Em 1984, ele foi o único cientista da computação de língua alemã a ser homenageado com o Prêmio Turing, muitas vezes considerado o equivalente ao Prêmio Nobel no campo da ciência da computação.
Ele recebeu o Prêmio Turing, foi um pioneiro no campo da ciência da computação e o criador de linguagens de programação muito influentes: Niklaus Wirth fez contribuições significativas e obteve sucesso notável no campo da ciência da computação. Sua conquista mais notável foi o desenvolvimento da linguagem de programação Pascal. No entanto, sua influência vai além do Pascal. Os esforços e a dedicação de Niklaus Wirth fizeram contribuições significativas e essenciais para o avanço global da ciência da computação. Até hoje, seus esforços exerceram uma profunda influência na ciência da computação e em gerações de programadores. Os parentes de Niklaus Wirth informaram que ele faleceu serenamente em 1º de janeiro de 2024.

Niklaus Wirth foi fundamental para a criação do campo da ciência da computação na Suíça. Ele introduziu com sucesso os avanços da ciência da computação dos Estados Unidos, a nação mais importante no desenvolvimento de computadores durante aquele período, na Suíça. Isso ajudou a estabelecer a ciência da computação como um campo independente de pesquisa e profissão na Suíça. O presidente da ETH, Joël Mesot, lembra-se de Niklaus Wirth como uma figura importante que não só fez contribuições inovadoras para o desenvolvimento de linguagens de programação, mas também desempenhou um papel fundamental no estabelecimento da ciência da computação na Suíça. Niklaus Wirth ocupou o cargo de professor na ETH Zurich por um período de 31 anos, de 1968 a 1999. A ETH Zurich recebeu um Departamento de Ciência da Computação autônomo e seu curso de graduação correspondente em 1981, graças à determinação inabalável dele e de seus colegas.
Grande interesse em tecnologia desde jovem
Niklaus Wirth, nascido em 15 de fevereiro de 1934 em Winterthur, demonstrou um grande interesse pela tecnologia desde muito cedo. Durante sua infância, ele se envolveu ativamente na construção de aviões e construiu com sucesso seus primeiros rádios e amplificadores. Impulsionado por seu fervor, matriculou-se como estudante na ETH Zurich. Ele fez um curso de engenharia elétrica e obteve um diploma na mesma área. Wirth obteve seu mestrado na Universidade de Laval, no Canadá, em 1960. Seu primeiro contato com computadores, linguagens de programação e compiladores ocorreu durante seu período na Universidade da Califórnia, em Berkeley. Naquele local, ele se aventurou no setor de software e, em 1963, obteve seu PhD em Berkeley sob a supervisão de Harry Huskey. Sua pesquisa concentrou-se na expansão da linguagem de programação Algol 60.

Após seus cargos como professor assistente na Universidade de Stanford e na Universidade de Zurique, ele voltou à ETH Zurique em 1968 como professor de Ciência da Computação. Continuou a lecionar e a realizar pesquisas nessa área até 1999. Durante os períodos de 1976-1977 e 1984-1985, dedicou seu tempo à realização de pesquisas no Palo Alto Research Centre (PARC) da Xerox.
Em um período de 31 anos na ETH Zurich, Niklaus Wirth foi pioneiro na criação de várias linguagens de programação novas, incluindo Euler, PL360, Algol W, Pascal, Modula, Modula 2, Oberon e LoLa. Além disso, ele construiu os primeiros computadores pessoais (PCs) na Suíça e instruiu o primeiro grupo de cientistas da computação suíços. Além disso, é autor de várias publicações canônicas que foram traduzidas e divulgadas globalmente. Ele foi agraciado com vários prêmios, incluindo o estimado Prêmio Turing da ACM, que recebeu em 1984 como o único cientista da computação de língua alemã até o momento. Em 1988, foi homenageado com o IEEE Computer Pioneer Award. O conceito conhecido como Lei de Wirth, que afirma que a taxa de deterioração do desempenho do software é maior do que a taxa de melhoria do desempenho do hardware, foi batizado em homenagem a Niklaus Wirth.

Pascal - a busca por uma linguagem de programação influente e descomplicada
Em 1984, ocorreram eventos significativos no campo da ciência da computação e para Niklaus Wirth. A Apple lançou o Macintosh PC, a IBM apresentou o IBM Personal Computer/AT e Niklaus Wirth foi homenageado com o Turing Award, o prêmio de maior prestígio em ciência da computação, equivalente ao Prêmio Nobel de ciências naturais ou à Medalha Fields de matemática. Wirth foi homenageado por suas contribuições na criação de várias linguagens de programação, incluindo Euler, Algol W, Modula e Pascal.

O Pascal é a contribuição mais conhecida de Niklaus Wirth no campo das linguagens de programação. O principal benefício disso é sua simplicidade e refinamento. O Pascal baseia-se nos princípios explícitos da programação estrutural, conforme articulado pelo cientista da computação Edsger W. Dijkstra. Ela também foi construída sobre uma base matemática definida pelo cientista da computação Tony Hoare e incorpora a implementação de Niklaus Wirth dos conceitos do Algol W. Essa linguagem integra efetivamente princípios sólidos de programação com programação estruturada e organização de dados. Consequentemente, ela ganhou popularidade rapidamente como linguagem para educação. Pascal foi a linguagem de programação que muitas gerações de estudantes, inclusive os da ETH Zurich, usaram para sua experiência inicial de programação em universidades do mundo todo.

Niklaus Wirth sempre foi motivado a buscar novos desafios e nunca se contentou com suas realizações passadas. Embora o Pascal seja amplamente reconhecido como sua realização mais famosa, suas contribuições vão além disso. Ele também desenvolveu a linguagem sucessora Modula-2, o sistema Oberon e a estação de trabalho "Lilith", que serviu como precursora dos computadores pessoais posteriores. Wirth dedicou toda a sua vida ao avanço e aprimoramento contínuos de suas linguagens de programação. A progressão de Euler para Oberon marcou o desenvolvimento de uma linguagem que adotou a orientação a objetos e a hierarquia de tipos. O objetivo do Oberon era obter o máximo de força e simplicidade simultaneamente. Niklaus Wirth pretendia criar uma invenção que atendesse à população em geral, de acordo com o princípio de ser econômica e compreensível.

O Oberon era mais do que apenas uma linguagem. O resultado foi um sistema completo e, no final, foi publicado o livro "Project Oberon", no qual o software, a linguagem e o hardware estão descritos em cerca de 500 páginas - o orgulho e a alegria de seu trabalho: "Eu persegui o objetivo de toda uma vida de desenvolver uma linguagem que fosse tão poderosa quanto possível, mas tão simples quanto possível. "Oberon representa o último estágio dessa sequência de desenvolvimento", declarou Niklaus Wirth.

Lilith - e sua dedicação à ciência da computação na Suíça
Atualmente, a Suíça ocupa uma posição significativa no campo da ciência da computação em escala global, fazendo inúmeras contribuições essenciais tanto para os princípios teóricos quanto para a implementação prática da disciplina. A situação passou por uma transformação até a década de 1970: Embora os Estados Unidos já tivessem criado as primeiras estações de trabalho e a ciência da computação fosse amplamente estudada, a Suíça estava atrasada tanto na educação quanto na implementação prática. Um exemplo disso pode ser visto no Lilith de Wirth, que só chamou a atenção do setor depois de vários anos.
O Lilith era uma estação de trabalho de computador antiga que tinha uma tela visual de alta resolução e um mouse, o que o tornou um precursor dos computadores pessoais modernos. Niklaus Wirth a criou na ETH em 1980 como base para vários projetos de software de pesquisa. A partir de 1982, os acadêmicos da ETH se esforçaram para monetizar o sistema, mas suas tentativas não foram bem-sucedidas. O PC passou por um desenvolvimento industrial nos Estados Unidos. No entanto, o Lilith exerceu um impacto profundo em todo um grupo de cientistas da computação. Niklaus Wirth criou o sistema de computador Ceres em 1986, seguindo seu trabalho no Lilith. Esse sistema incluía o sistema operacional Oberon e a linguagem de programação Oberon. Os computadores Ceres foram utilizados para a formação de estudantes de ciência da computação na ETH Zurich até por volta de 2003.
O processo de desenvolvimento da ciência da computação na ETH e na Suíça também não foi linear: Inicialmente, Niklaus Wirth e seus colegas tiveram que superar vários obstáculos. No início da década de 1970, eles iniciaram um esforço para estabelecer a ciência da computação como uma disciplina acadêmica independente. No entanto, tanto os esforços iniciais quanto os posteriores não foram bem-sucedidos. No entanto, em resposta à evidente escassez de cientistas da computação na Suíça, a ETH Zurich estabeleceu de forma decisiva a ciência da computação como um departamento e um curso de graduação em 1981. A dedicação de Niklaus Wirth e seus colegas estabeleceu a base para o avanço da ciência da computação na Suíça.

Debugging avançada com o RAD Studio 11.3 Alexandria.por Max Kleiner

Como você deve saber, é possível instruir o depurador a ignorar determinados tipos de exceções. A Figura 1 desta página mostra as opções de exceção de linguagem Delphi e de exceção OS nativa do sistema operacional.
Adicione uma classe de exceção à lista em language-exceptions, e todas as exceções desse tipo e de quaisquer tipos descendentes passarão para o seu programa sem que o Delphi interfira.

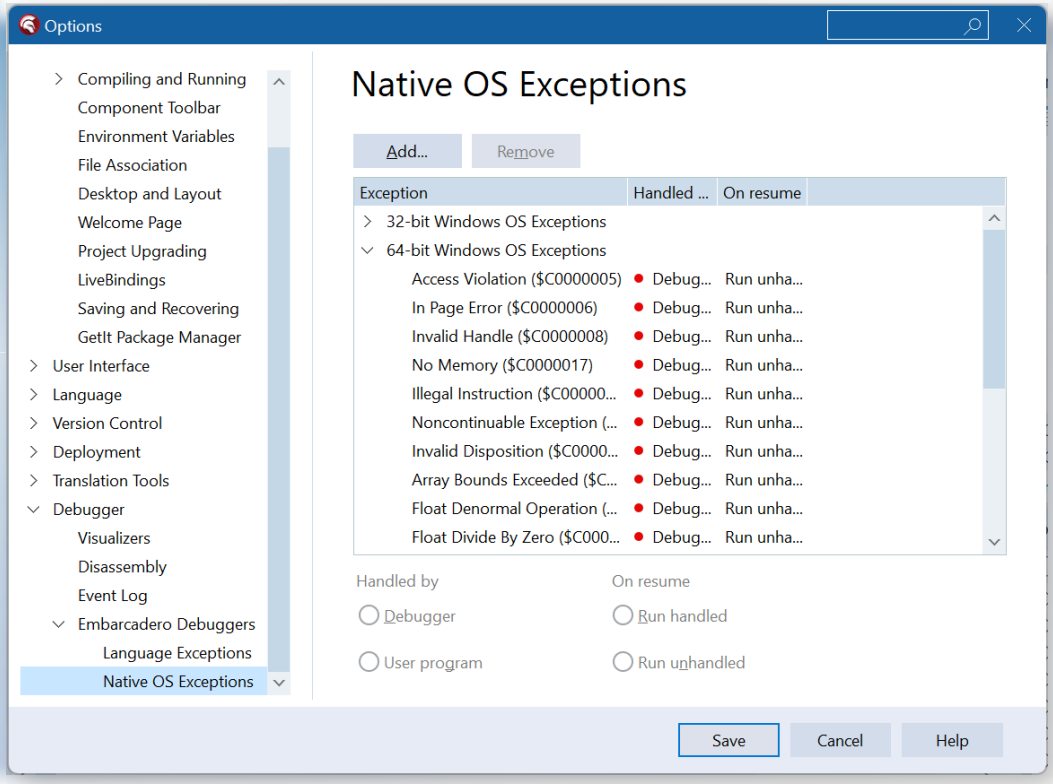


Figura 1

Em suas configurações padrão, o Delphi IDE notifica sempre que ocorre uma exceção em seu programa, como na Figura 2 desta página. O que é importante perceber é que, nesse ponto, nenhum código de tratamento de exceções do seu programa ainda não foi executado.
É tudo o próprio Delphi; seu status especial como debugger permite que ele receba a primeira notificação de qualquer exceção em seu programa, mesmo antes de seu programa saber sobre ela.
Portanto, não é tão fácil assim abrir seu aplicativo de 64 bits no IDE, adicionar e ativar as opções do debugger de 64 bits e compilar seu aplicativo como um aplicativo Windows de 64 bits com a depuração correta.

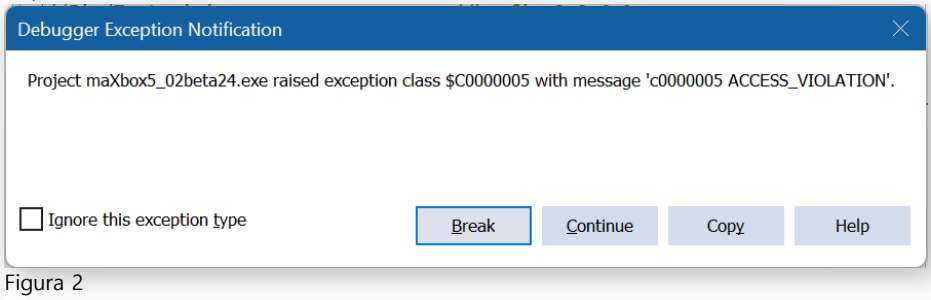


Figura 2

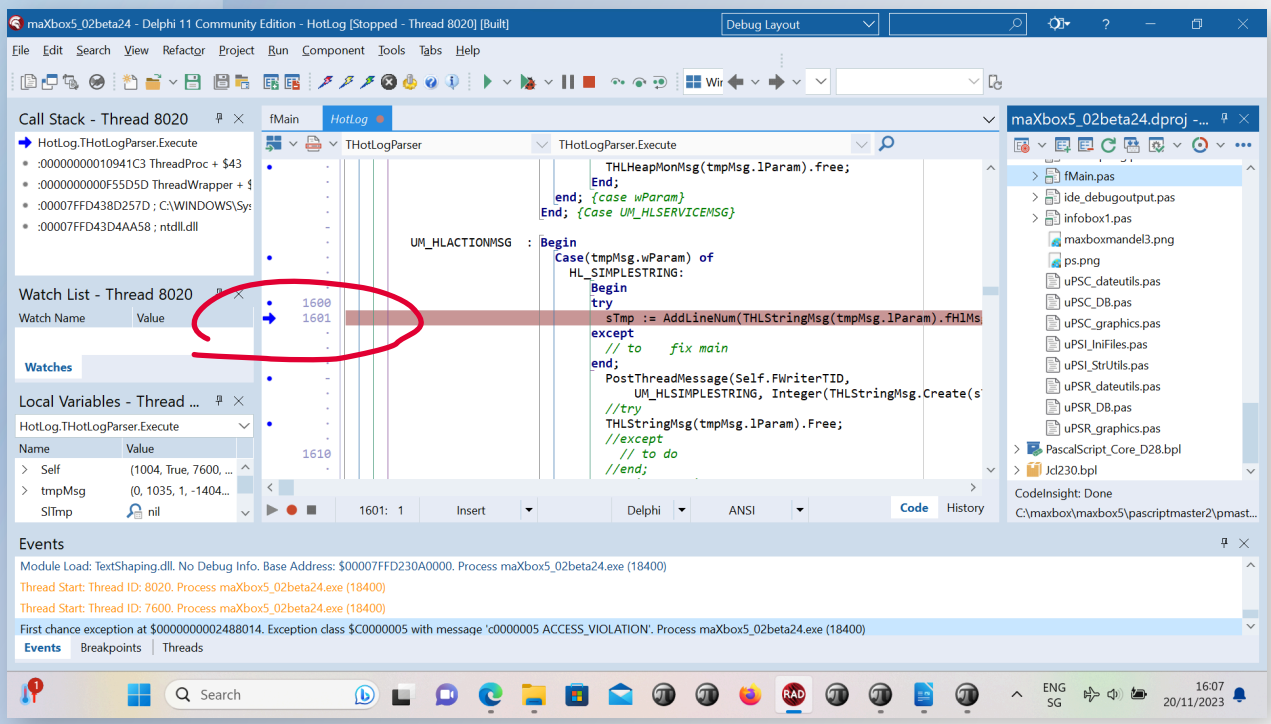
Figura 3

Você pode alterar os valores das opções em qualquer configuração, incluindo a **Base**. Você pode excluir as configurações **Debug** e **release**, mas não pode excluir a configuração **Base** ou movê-la. Ao vasculhar ou mergulhar no código-fonte do **maXbox4**, parece ser impossível migrar mais de `3354` unidades de forma decente e adequada para o **maXbox5**, também conhecido como **versão de 64 bits**.
Algumas pessoas estavam reclamando como se isso criasse problemas, mas sem fornecer um exemplo adequado. Além disso, a Embarcadero recentemente adicionou a recomendação de usar `FreeAndNil` em seu manual (*finalmente*!).
OBSERVAÇÃO: sempre compile o aplicativo no **modo Release e Debug**.
Certifique-se de que as opções do projeto estejam definidas corretamente para o modo de debug. As configurações PADRÃO para o **modo Debug NÃO estão corretas/completas** - pelo menos não no **Delphi XE7** e no **Tokyo**. Antigamente, eles definiam as opções corretas para o modo de depuração, se houvesse alguma. Portanto, habilite e estude assuntos como:

- **"Stack frames"**
- **"Map file generation (detailed)"** **"Geração de arquivo de mapa (detalhado)"**
- **"Range checking"**, **"Verificação de intervalo"**
- **"Symbol reference info"****"Informações de referência de símbolo"**
- **"Debug information"****"Informações de debug"**
- **"Overflow checking"****"Verificação de estouro"**
- **"Assertions"****"Asserções"**
- **"Debug DCUs"****"Debug DCUs"**

Na maioria das vezes, você obtém algumas **exceções de primeira chance**. Depois de quebrar a notificação de exceção do debugger na figura, você obtém exatamente a linha para corrigir o erro (*era um* **Nil Pointer**):

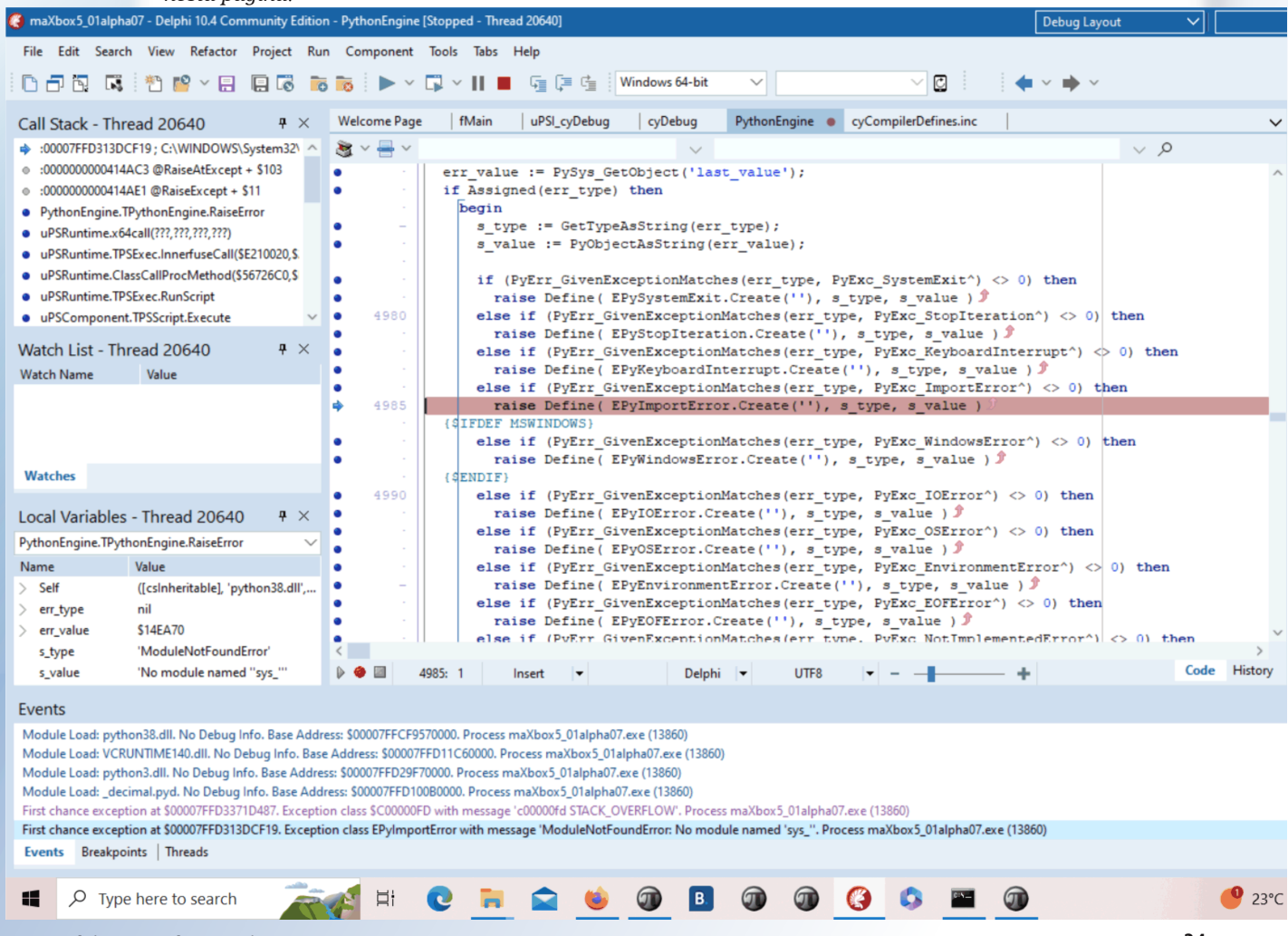
Figura 4



Você pode usar os "**pontos de interrupção avançados** advanced breakpoints " do **Delphi** para desativar o tratamento de exceções em uma região de código. Para começar, defina um **breakpoint** na linha de código em que você deseja que o **IDE** ignore as exceções. **Clique com o botão direito** do mouse no ponto de interrupção na sarjeta e abra a caixa de diálogo de propriedades do ponto de interrupção. Na seção avançada, há algumas caixas de seleção. Desmarque a caixa "**Break**" para evitar que o **debugger** interrompa seu programa nessa linha, e marque a caixa "**Ignore subsequent exceptions-Ignorar exceções subsequentes**" Depois disso, defina outro ponto de interrupção onde deseja que o depurador retome o tratamento de exceções. Altere suas propriedades para tratar exceções subsequentes.
O que são exceções subsequentes?
Elas tratam todas as exceções subsequentes **levantadas pelo processo atual** durante a **sessão de debug atual** (*o depurador parará nas exceções com base nas configurações de exceção atuais em:* **Tools → Options → Debugger Options → Embarcadero Debuggers → Language Exceptions**).

Essa opção interrompe todas as exceções.
Use-a para ativar o **comportamento normal de exceção** depois que outro ponto de interrupção **desativou o comportamento normal** usando a opção **ignorar exceções subsequentes**.
Faz sentido em um bloco de código com a opção ignorar exceções subsequentes.
Ignora todas as exceções subsequentes levantadas pelo processo atual durante a sessão de depuração atual (*o depurador não será interrompido em nenhuma exceção*). Normalmente, você sabe que **interrompe** a execução: a ação tradicional e padrão de um ponto de interrupção.
Use **Ignore subsequent exceptions** (*Ignorar exceções subsequentes*) com **Handle subsequent exceptions** (*Tratar exceções subsequentes*) como um par. Você pode cercar blocos específicos de código com o par **ignore/ Handle** para ignorar quaisquer exceções que ocorram nesse bloco de código, como na *Figura 5 a seguir nesta página.*

Figura 5



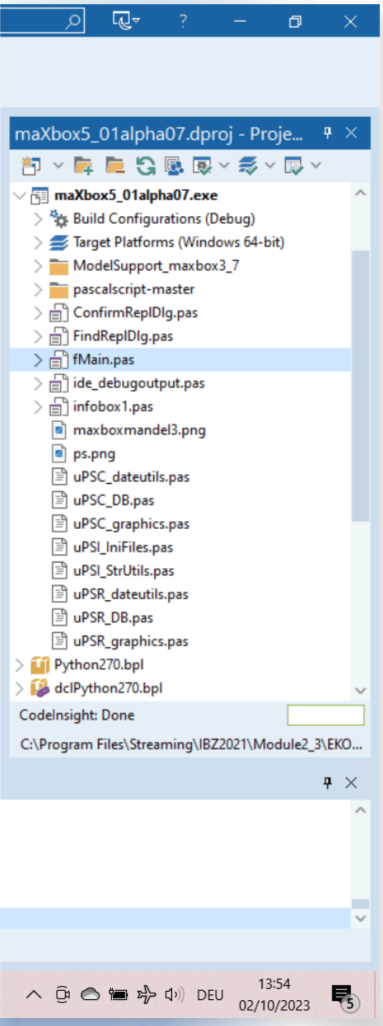
PREPARE-SE PARA DEPURAR O DEPURADOR

No **Delphi 11** - e provavelmente na maioria das outras versões - se uma **exceção** escapar do método `execute` sem ser tratada, ela será capturada pela função que chamou `Execute` e armazenada na propriedade `FatalException` do `thread`. (*Veja em* `Classes.pas`, `ThreadProc`.)
Nada mais é feito com essa exceção até que o thread seja liberado, momento em que a exceção é **também liberada**.

```
procedure TForm1.Onterminate(Sender: TObject);
var ex: TObject;
begin
  Assert(Sender is TThread);
  ex:= TThread(Sender).FatalException;
  if Assigned(ex) then begin
  // Thread terminated due to an exception
    if ex is Exception then
    Application.ShowException(Exception(ex))
```

Ao contrário da **API do Windows TerminateThread MSDN**, que força o **thread** a ser encerrado imediatamente, o método `Terminate` apenas solicita que o **thread** seja encerrado.
Isso permite que o **thread** execute qualquer limpeza e finalização antes de ser encerrado.
Para capturar as exceções que ocorrem dentro de sua função de thread, adicione um bloco try.. .except à implementação do método - `Execute`! Então, preparei o projeto e a depuração no modo Debug da seguinte forma: *Figura 6 nesta página*.

Figura 5 Continuação



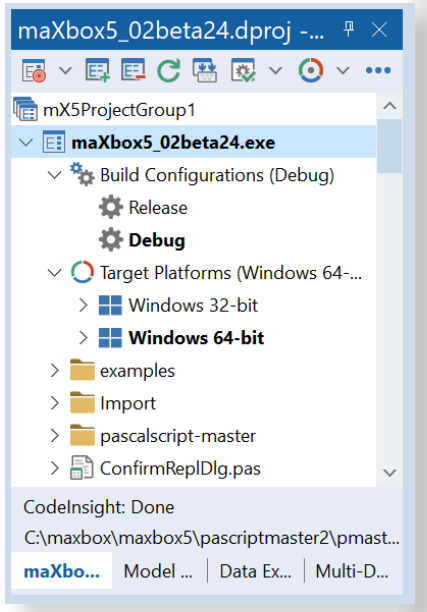
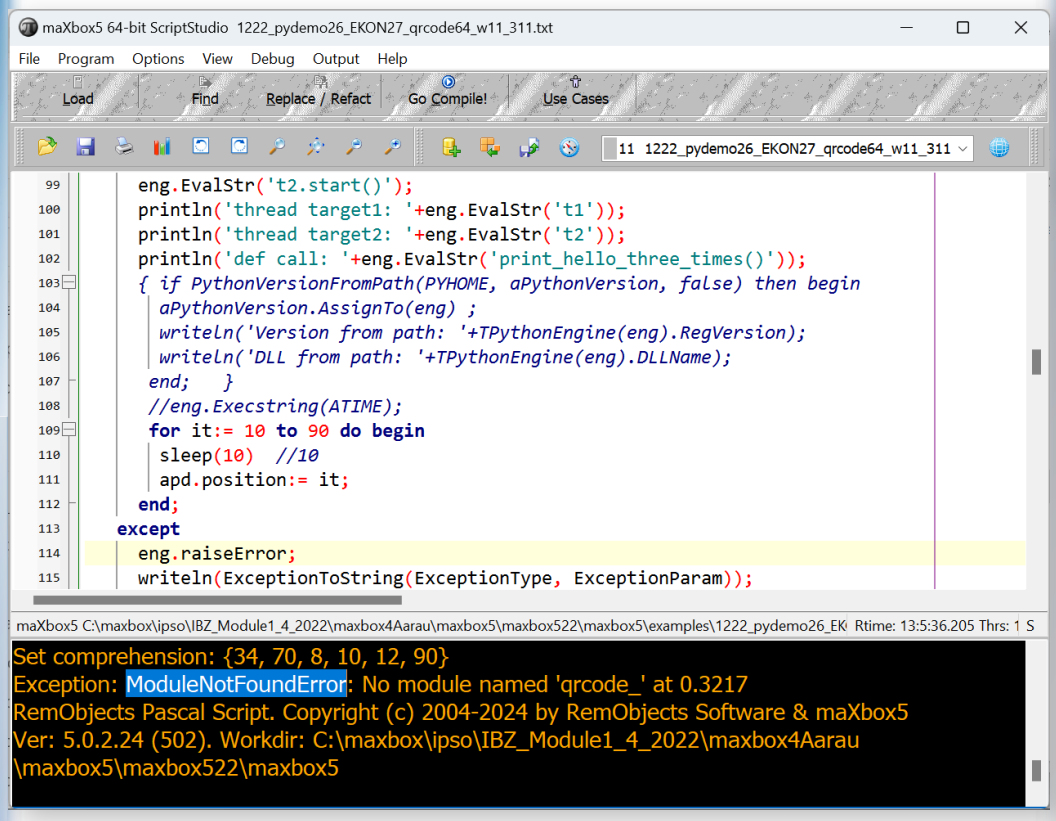


Figura 6

O estranho é que eu envolvi minha chamada **Pascal** em um `try except`, que tem manipuladores para `AccessViolationException`, `COMException` e tudo o mais, mas quando o **Delphi 10.4** ou o **Studio 11.3** intercepta a **AccessViolationException**, o depurador é interrompido na chamada do método (`doc.OCR`), e, se eu avançar, ele continua para a próxima linha em vez de entrar no bloco catch ou except.

Por isso, decidi capturar a exceção no script em uma rotina de tempo de execução para obter o máximo de no meu console a partir de uma caixa de **debug** dinâmica **maXbox**:
Veja a próxima página.



Também ativei a variável JITEnable, que controla quando o depurador **just-in-time** é chamado.

Portanto, para a reorganização dos códigos-fonte, tenho a última revisão com patches da issue #202 (*commit 86a057c*), mas não consigo compilar os arquivos no início (*Core_D27*) que fazem parte do **`PascalScript_Core_D27.dpk`** para essa plataforma para **Linux64, Win64** ou **MacOS64**.

Aqui está uma saída de código-fonte para mostrar o tratamento interno de exceções para o compilador 10.4 dccosx64 ou dcc64 (*existem resultados semelhantes para o dcclinux64*):

```
procedure TPSExec.ExceptionProc(proc, Position: Cardinal; Ex: TPSError;
                                const s: tbtString; NewObject: TObject);
var
  d, l: Longint;
  pp: TPSExceptionHandler;  //debcnt: integer
begin
  ExProc:= proc;
  ExPos:= Position;
  ExEx:= Ex;
  ExParam:= s;
  inc(debcnt);
  if maxform1.GetStatDebugCheck then
    maxform1.memo2.lines.add('debug: '+inttostr(debcnt)+'-'+s+'
        '+inttostr(proc)+' err:'+inttostr(ord(ex)));    //@fmain
  if ExObject <> nil then
   ExObject.Free;
  ExObject:= NewObject;
  //ShowMessage('We do not get this far: '+exparam);
  if Ex = eNoError then Exit;
  //maxform1.memo2.lines.add(s);
  // halt(1);
  // ShowMessage('We don't want not get this far');

  for d:= FExceptionStack.Count -1 downto 0 do begin
   pp:= FExceptionStack[d];
   if Cardinal(FStack.Count) > pp.StackSize then begin
    for l:= Longint(FStack.count) -1 downto Longint(pp.StackSize) do
     FStack.Pop;
   end;
```

Então, posso ver o **ExceptionProc** como uma exceção de primeira chance em $0000000100419E9E.

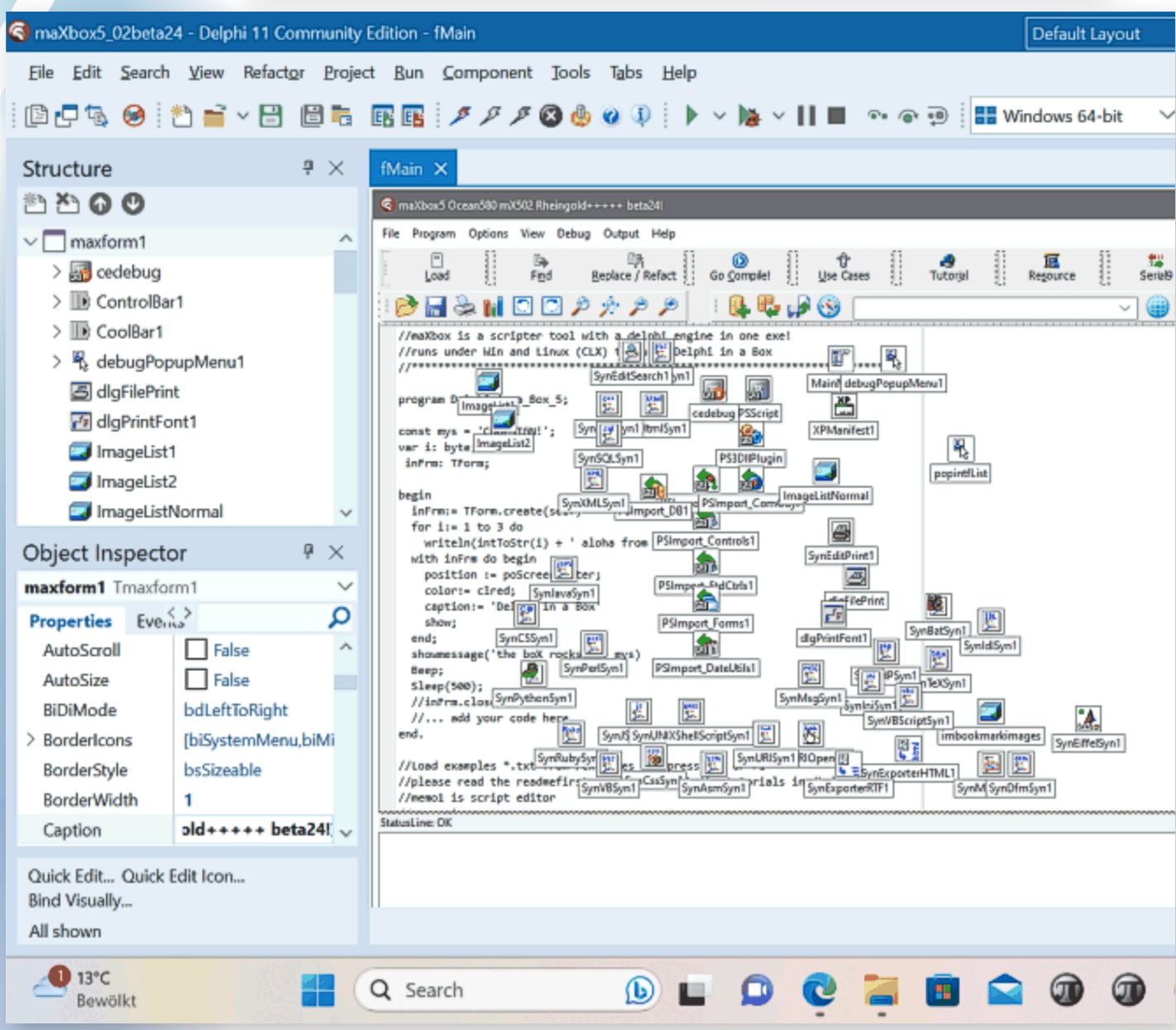
Classe de exceção `EAccessViolation` com mensagem

'Access violation at address `0000000100419E9E`,
accessing address `00000009017241F8`'.

```
Process TestApplication (5741)
Source Breakpoint at: C:\Program Files\Streaming\IBZ2021\Module2_
3\EKON26\maxbox4\pascalscript-master\pascalscript-master\Source\
uPSRuntime.pas line 2060. Process TestApplication (5741)

Upsruntime.TPSExec.Clear()(0x00000002017350d0)
Upsdebugger.TPSCustomDebugExec.Clear()(0x00000002017350d0)
Upscomponent.TPSScript.Compile()(0x0000000201734c20)
Fmain.TForm1.Compile1Click(System.TObject*)(0x00007ffeefbfe038)
Vcl.Menus.TMenuItem.Click()(0x0000000201734960)
Vcl.Menus.TMenu.DispatchCommand(unsigned short)(0x0000000201734340,2)
Vcl.Forms.TCustomForm.WMCommand(Winapi.Messages.
TWMCommand&)(0x0000000205039ff0,0x00007ffeefbfe878)
:000000010001132B System::Tobject::Dispatch(void*)
```

Outro ponto perturbador da depuração foi um redirecionamento para a debug.
Como uso o `SynPdf.pas`, preciso incluir o `SynCommons.pas` em meu projeto, mas parece que consegui mais do que queria.
Às vezes, quando depuro no **IDE** e tento me aprofundar em alguma rotina, chego à seção - **Move procedure** do `SynCommons.pas` em vez da chamada de sistema normal onde eu quero chegar!
A solução é fácil: **FastCodem**, Move e **Fillchar** estão incluídos no `SynCommons.pas` otimizados para velocidade, e o **SmartLinking** não será grande no seu exe, e o registro e todos os outros não farão parte dele.
Você pode se livrar dele, comentando as linhas correspondentes no bloco de inicialização dessa unidade. Nas novas versões da estrutura, você tem uma configuração condicional para desativá-la.

```
RedirectCode(GetAddressFromCall(@RecordCopyInvoke),@RecordCopy);
RedirectCode(GetAddressFromCall(@FillCharInvoke),@FillChar);
RedirectCode(GetAddressFromCall(@MoveInvoke),@Move);
```



Quando escolho uma segunda compilação em um **CrossVCL** no menu, sempre testo também o depurador com duas funções de código semelhantes, mas com soluções de código diferentes, por exemplo, `GetBytes`:

```
function GetBytes(value: string): TBytes;
begin
  SetLength(Result, SizeOf(value));
  Move(value, Result[0], SizeOf(value));
end;

function AnsiBytesOf(const S: string): Tbytes; //like GetBytes
begin
  Result := TEncoding.ANSI.GetBytes(S);
  //Result := GetBytes(S);
end;
```

E então, talvez por intuição, fiz uma compilação e o **AD** desapareceu e obtive minha primeira tela, compilada e o script executado:
Um dos problemas resolvidos nesse meio tempo é detectar uma violação de acesso em vez de travar o aplicativo. É como se o código no uPSRuntime.pas não conseguisse encontrar a causa da parada ou da saída, como a seguir::

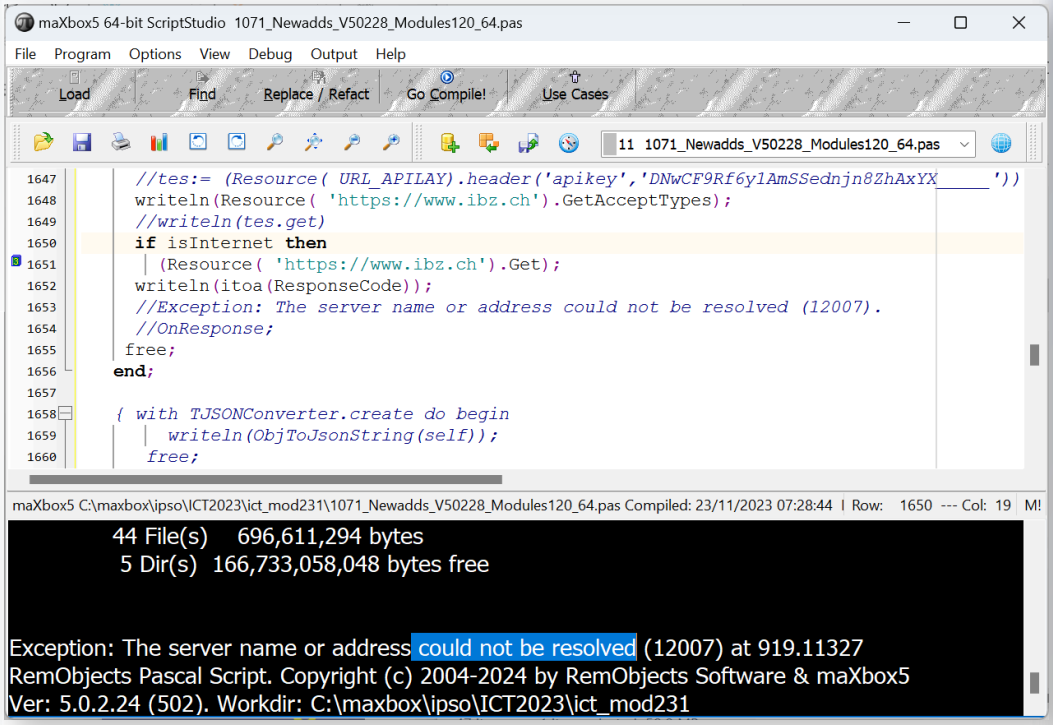
```
procedure TdynamicDll.Quit;
begin
   if not( csDesigning in ComponentState ) then begin
   {$IFDEF MSWINDOWS}
   MessageBox( GetActiveWindow, PChar(GetQuitMessage), 'Error',
                          MB_TASKMODAL or MB_ICONSTOP );
   ExitProcess( 1 );
   {$ELSE}
   WriteLn(ErrOutput, GetQuitMessage);
   Halt( 1 );
   {$ENDIF}
   end;
end;
```

Após a depuração, percebi que se trata de uma exceção de primeira chance que funciona enquanto o debbuger estiver em execução com break ou continue, mas sem o depurador o aplicativo desaparece sem encaminhar o **AV** na saída, como **AV** no endereço xyz lido do endereço 000.
Você deve saber que o manipulador `Application.OnException` só é chamado para **exceções não tratadas**. Uma exceção não tratada ocorre quando nenhum bloco try..except encontrou a exceção ou quando ela foi capturada e, em seguida, levantada novamente!

A execução continuará em qualquer bloco except externo. Se não houver nenhum, ou se os que existirem também levantarem novamente a exceção, a exceção acabará chegando ao manipulador `Application.OnException` de fato: um `Application.OnException` é a sua última chance de lidar com uma exceção não tratada. Não é a primeira oportunidade de responder a qualquer exceção.
Capturar a exceção e exibir a mensagem só é útil se você já tiver liberado o software para os usuários e não puder reproduzir o problema localmente (*e, mesmo assim, não é tão bom quanto usar uma biblioteca de exceções real, como a* **EurekaLog** *ou a* **MadExcept**). Nesse caso, já sabemos qual linha causou a exceção, qual classe de exceção foi lançada e qual foi sua mensagem. O código sugerido NÃO acrescenta informações à investigação.
Portanto, se você tiver um erro de rede ou uma exceção em sua chamada da Web de um Rest-Client, pode ser prudente verificar se a conexão não está funcionando antes de capturar a possível exceção, como visto na próxima imagem.
Dica para o `erro 12007`: Estou tentando obter a fonte HTML por trás de um site onion por meio da **WinINet API**, mas o `InternetOpenUrl()` retorna o código de `erro 12007`, o que sugere que há um problema com relação à resolução da conexão que sugere que há um problema com relação à resolução do endereço da Web.

A parte incomum desse problema é que o erro não pode ser reproduzido quando um site que não seja a **non-onion site** * é usado para o **link da Web**, o que significa claramente que não há nenhum problema na parte referente a possíveis configurações de proxy.
Além disso, a função `GetLastError` retorna um código de erro como 12007. Você também pode verificar com a ferramenta **NETMON** se foi feita uma tentativa real de conexão.

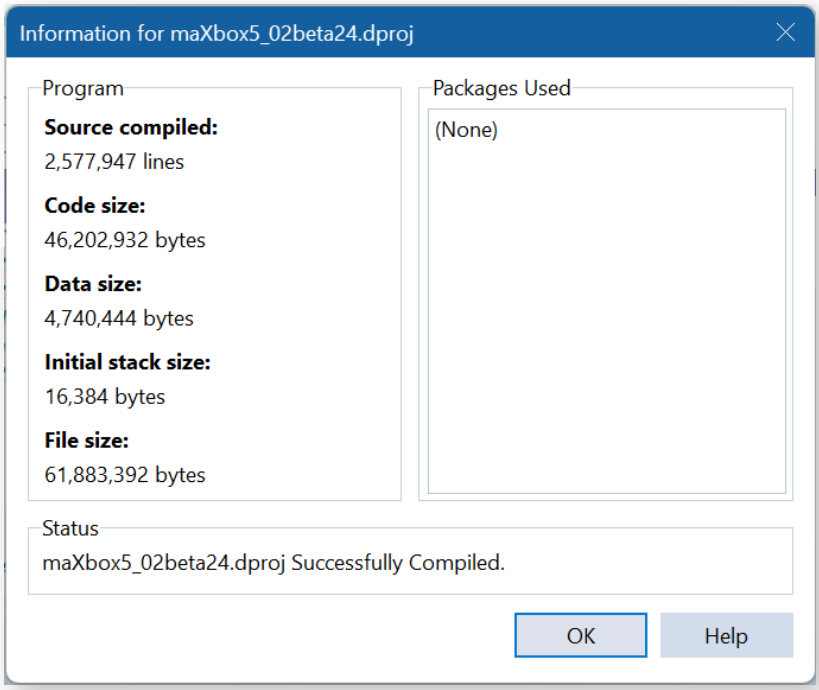
.onion é um nome de **domínio de primeiro nível** de uso especial que designa um serviço onion anônimo, que era conhecido anteriormente como "serviço oculto", acessível por meio da rede Tor. Esses endereços não são nomes DNS reais, e o **domínio de primeiro nível .onion** não está na raiz do **DNS** da Internet *Isso explica o nome: **site non onion**.



Como vimos, você pode instruir o depurador a ignorar determinados tipos de exceções.
Adicione uma classe de exceção à lista, e todas as exceções desse tipo e de quaisquer tipos descendentes passarão pelo seu programa sem que o Delphi interfira.
Você pode usar os "**pontos de interrupção avançados - advanced breakpoints**" do **Delphi** para desativar o tratamento de exceções em uma região do código.
Para começar, defina um ponto de interrupção na linha de código em que você deseja que o IDE ignore as exceções ou a chamada é uma **API** de fora, como o seguinte User-Agent.

```
'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
537.36(KHTML, like Gecko) Chrome/88.0.4324.104 Safari/537.36'}
try:
    request_result = requests.get(url, headers=headers).json()
    print(request_result)
    print('[In English]:
         ' + request_result['alternative_translations'] [0]
         ['alternative'][0]  ['word_postproc'])
    print('[Language Dectected]: ' + request_result['src'])
except:
     pass
```

CONCLUSÃO:
Os **breakpoints** pausam a execução do programa em um determinado local ou quando ocorre uma condição específica. Você pode definir **breakpoints de origem** breakpoints de carregamento de módulo no Code Editor antes e durante uma sessão de depuração. **`Application.OnException`** é a sua última chance de lidar com uma exceção não tratada. Basicamente, as exceções são lançadas primeiro no depurador e, em seguida, no programa real, onde, se não forem tratadas, ela será repetida duas vezes, oferecendo uma chance de fazer algo com ele no seu IDE, antes e depois do aplicativo em si.




```
Referências:

Projeto compilado:
https://github.com/maxkleiner/maXbox4/releases/download/
V4.2.4.80/maxbox5.zip

Documentos e ferramentas: https://maxbox4.wordpress.com

maXbox Download | heise Download
```

POR DMITRII V. KONNOV, TOBIAS GIESEN

ABSTRACT

Este artigo apresenta uma visão geral do novo recurso marcado como **PascalScript**, implementado no aplicativo de sincronização de arquivos em nuvem líder mundial chamado **Syncovery**

Palavras-chave: Syncovery 10, armazenamento em nuvem, Google Drive, Sharepoint, OneDrive, DropBox, PascalScript

INTRODUÇÃO

O **PascalScript** é uma linguagem de script que se **baseia na linguagem de programação Pascal.** Ela **permite o controle automatizado do tempo** de execução de aplicativos com scripts e software de servidor. Esse mecanismo de criação de scripts inclui um compilador e um interpretador de código de bytes, oferecendo uma solução gratuita e de código aberto.

O **PascalScript** foi projetado para ser amplamente compatível com o **Object Pascal**, tornando-o parcialmente interoperável com o **Delphi**, o **Free Pascal** e o **GNU Pascal**. Originalmente criado por **Carlo Kok** com o nome de **CajScript**, foi posteriormente renomeado como **Innerfuse Pascal Script com a versão 2.23**.

Posteriormente, a **RemObjects** assumiu o projeto, dando-lhe o nome de **RemObjects PascalScript**. Em seguida, ele foi disponibilizado como software de código aberto para integração no **Delphi Integrated Development Environment (IDE)**. A partir da versão 2.07, o **PascalScript** foi portado para o **FreePascal**. Desde 2017, o **PascalScript** foi integrado como um componente padrão no **IDE Lazarus**, expandindo ainda mais seu alcance e utilidade.

SYNCOVERY

Entre os softwares que utilizam amplamente o **PascalScript**, o **Syncovery** é um conhecido programa de backup e sincronização de arquivos na nuvem.

Devido ao número crescente de **armazenamento em nuvem** de provedores do setor **global de TI em nuvem**, como **Amazon S3, Google Drive, Microsoft Azure, OneDrive, SharePoint, DropBox, Box e Backblaze B2**, a integração do armazenamento de dados em nuvem está se tornando uma das tarefas mais populares de um usuário moderno.

Cada armazenamento em nuvem tem suas especificidades, sua interface de interação com o usuário e sua **API**. Do ponto de vista do usuário, o ideal é um software de sincronização de arquivos que assume a função de fazer backup de arquivos e, como um módulo totalmente autônomo, sincroniza o conteúdo local com a nuvem em segundo plano.

Nesse caso, a distinção entre o disco local e a nuvem é pouco nítida. O local flui suavemente para a nuvem e vice-versa. O usuário tem a oportunidade de não se preocupar com a segurança dos dados, atribuindo todas as rotinas a processos e dispositivos automatizados que lidam com eles.

O **PascalScript** é chamado no **Syncovery** para personalizar os requisitos específicos do usuário das tarefas de sincronização de arquivos.

O aplicativo fornece uma **GUI** de área de trabalho no **Windows** e no **macOS**, enquanto a edição para **Linux** inclui uma interface de usuário baseada na **Web** que é acessada por meio de um navegador.

O **Syncovery** se baseia principalmente na facilidade de compreensão do conceito de sincronização de dados. Para isso, foi proposto fornecer ao usuário uma interface gráfica intuitiva na forma de esquerda direita, conforme mostrado na *Figura. 1.*

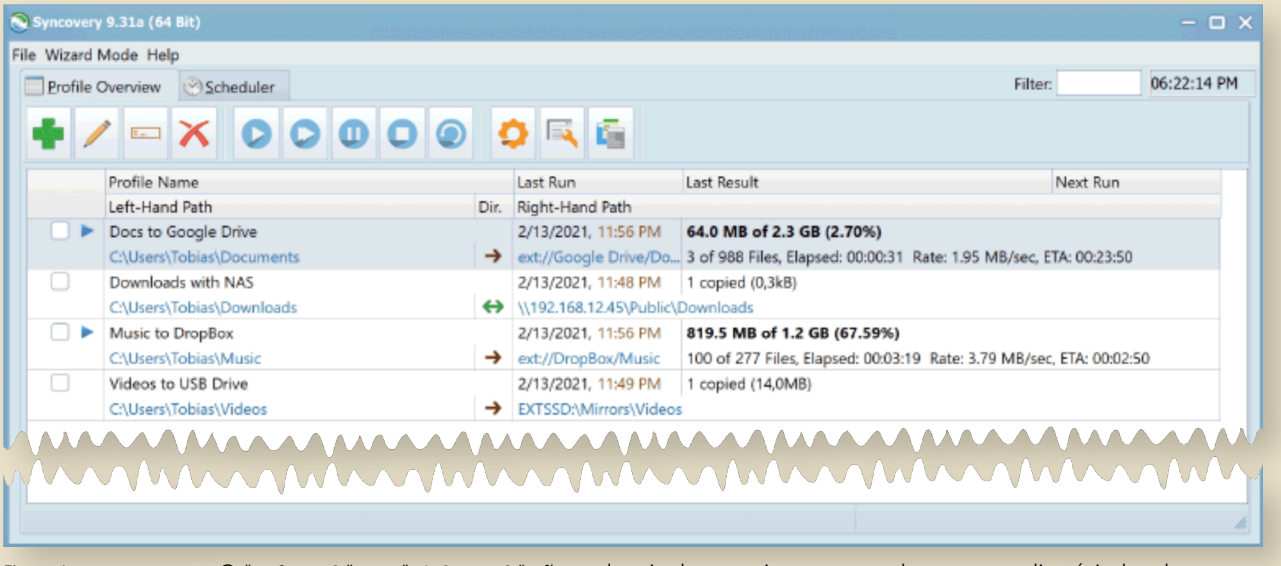


Figura 1.
Interface de usuário do Syncovery

O "**Left path**" e o "**Right path**" são os locais dos arquivos, que podem ser um diretório local ou um diretório na nuvem. O usuário configura os pontos de sincronização (pastas) de seus arquivos e o agendador integrado inicia as tarefas de sincronização.
A flexibilidade das tarefas de sincronização pode ser obtida não apenas por meio de várias configurações na **GUI** do **Syncovery GUI**, mas também com o **PascalScript**.

PASCALSCRIPT

O Syncovery inclui um mecanismo PascalScript que permite personalizar o comportamento do seu perfil de várias maneiras. O PascalScript inclui os seguintes recursos:

- **`Variables, Constants`**

Standard language constructs:

- **`Begin/End`**
- **`If/Then/Else`**
- **`For/To/Downto/Do`**
- **`Case x Of`**
- **`Repeat/Until`**
- **`While`**
- **`Uses`**
- **`Exit`**
- **`Continue`**
- **`Break`**
- **`Functions inside the script`**

Há muitas funções padrão, como `Pos`, `Copy`, `Length`, `Ord`, `GetProfileProperty`, `SetProfileProperty`, `SaveProfileSettings`, `ConcatPath`, `ExtractFileName`, `ExtractFilePath`, `ExtractURLPartAfterServer`, `ExtractFileExt`, `ChangeFileExt`, `FileExists`, `FileExistsMatching`, `EntryExists`, `FileAge`, `FileCopy`, `FileDelete`, `FileRename`,
e muitos outros com os quais você pode se familiarizar em
`https://www.syncovery.com/pascalscript/`

PascalScript was added in **Syncovery 8**, and many new hooks and functions have been added since the first release. Let us see how to start doing custom programming in PascalScript. Double-click on a profile to open the Profile Settings dialog (see Fig. 2).

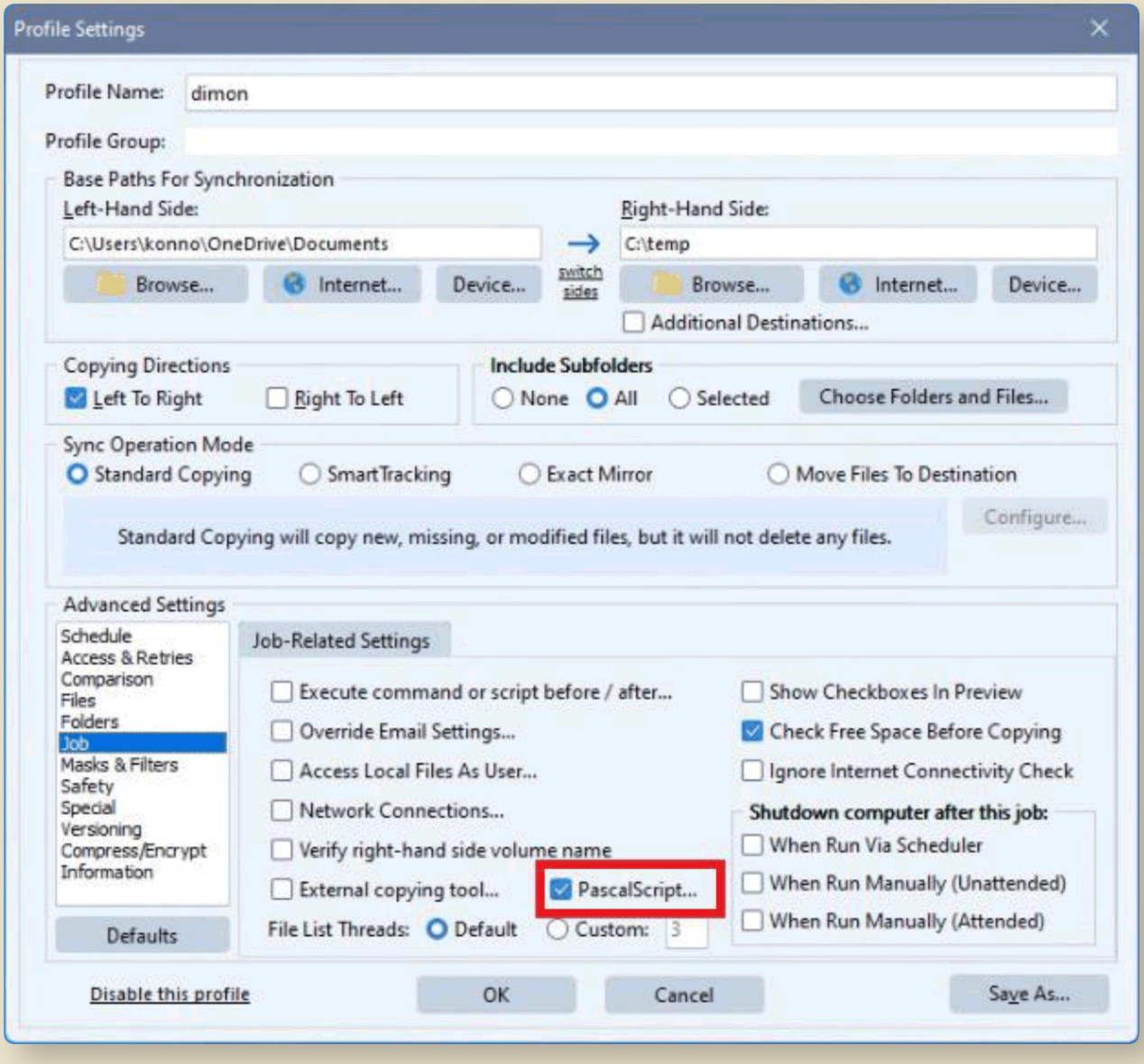


Figura 2. Profile Settings dialog

Em seguida, selecionar o item "**Job**" na lista **Advanced Settings** e clicar duas vezes na caixa de seleção "**PascalScript**" abrirá o editor **PascalScript**. Para alterar um comportamento específico, você precisa escrever uma função hook e escrever algum código.

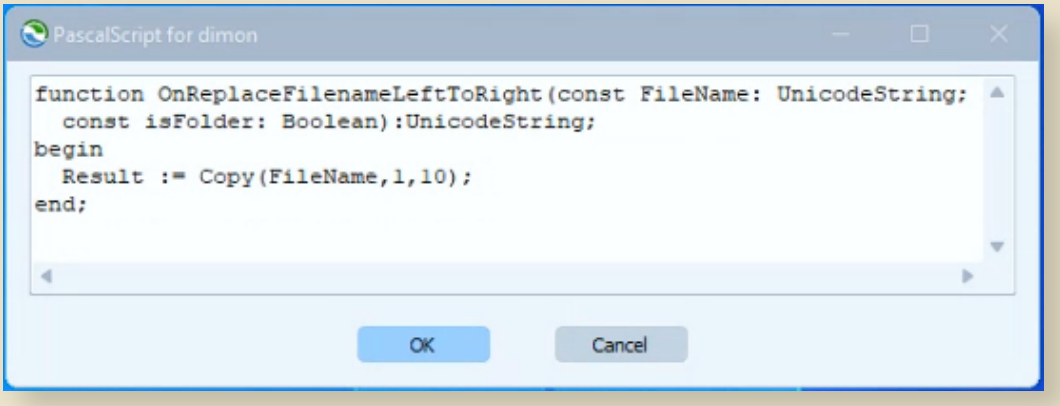


Figura 3. Editor PascalScript

Essa janela permite que você edite um script associado ao perfil do **Syncovery**. Todo manipulador de função declarado no editor será automaticamente conectado ao evento do **Syncovery**.
No script fornecido (*consulte a Figura.3, página 4 deste artigo*), podemos ver a função `OnReplaceFilenameLeftToRight` retorna um novo nome de arquivo e podemos encurtá-lo, por exemplo, se o comprimento do `FileName` for maior que 10.
Mas esse é apenas um exemplo simples das habilidades do **PascalScript** e a tarefa de cada cliente pode envolver muito mais lógica comercial.
Na maioria dos casos, nosso suporte técnico escreverá o código para você. O poder do **PascalScript** no **Syncovery** foi aprimorado regularmente à medida que novos ganchos e funções de suporte foram adicionados à linguagem.
Alguns casos de uso típicos são descritos aqui.

- **Renomeação de arquivos personalizados**:

Talvez o hook **PascalScript** usados com mais frequência no **Syncovery** seja `OnReplaceFilenameLeftToRight`. Ele permite especificar ou derivar um novo nome de arquivo, seja completamente livre ou com base no nome do arquivo original. Estão disponíveis scripts de amostra para encurtar nomes de arquivos longos ou para converter caracteres não permitidos.

- **Filtros personalizados**:

Usando o hook `OnIncludeItem`, o cliente pode usar o código para implementar filtros incomuns. Por exemplo, um script pode ser usado para impor regras de nomenclatura específicas e descartar todos os arquivos que não correspondam. Há exemplos de scripts para excluir arquivos sem extensão de nome de arquivo, bem como para processar apenas as subpastas que contêm o arquivo `READY`.to process.

- **Classificação de arquivos em subpastas**:

Outro caso de uso frequente de um **PascalScript** é a necessidade de classificar arquivos em subpastas adicionais que não existem no lado da origem. O gancho `OnBeforeFileCopy` permite alterar o caminho de destino com base no código **Pascal**. Um script de exemplo do site do **Syncovery** manipulará os caminhos de destino para cópia (*da esquerda para a direita*), adicionando a subpasta "**archive**" que não está presente no lado da origem. Outros clientes precisam que os arquivos sejam classificados em subpastas com base na data, o que também é possível.

- **Programações personalizadas**:

O hook `OnGetNextRunTime` pode ser usado para criar regras personalizadas para o agendamento de um trabalho. A melhor maneira de usá-lo é dar ao perfil uma programação simples e regular e usar o gancho para ignorar tempos de execução indesejados. Um exemplo da documentação do **PascalScript** destina-se a um perfil que está programado para ser executado "todos os dias às `XX: YY`". O gancho garante que ele seja executado somente no primeiro dia da semana em um mês.
A documentação do **PascalScript** no **Syncovery**, juntamente com vários scripts de amostra, pode ser encontrada nesta página:
`https://www.syncovery.com/pascalscript/`

CONCLUSÃO

Em breve, o **Syncovery** completará 20 anos. Com o passar dos anos, o **Syncovery** *2 se tornou uma ferramenta essencial para administradores de sistemas.

A primeira versão do aplicativo (escrita em **Delphi**) foi lançada com o nome de **Super Flexible File Synchronizer** em 2003.
O projeto é uma **German startup STEM** que acabou se transformando em um produto funcional. Em 2012, o produto foi renomeado para **Syncovery**.

Em 2021, no **Delphi Showcase Challenge**, dedicado ao 26º aniversário da linguagem de programação **Delphi** a ferramenta **Syncovery** recebeu o quarto prêmio de excelente software de sincronização de desktop que demonstra a flexibilidade do ambiente de desenvolvimento **Delphi** *3.
Embora o compilador **Delphi** ofereça suporte à compilação para Windows, Linux e Mac, ele só é usado para compilar o **Syncovery** para **Windows**.

Os binários para outras plataformas são criados usando o compilador FreePascal de código aberto.
Ao longo dos 20 anos de seu desenvolvimento, o **Syncovery** demonstrou uma fidelidade inigualável dos usuários e um desenvolvimento estável e sustentável do produto. Novos recursos estão sendo constantemente adicionados ao programa seguindo as solicitações dos usuários modernos. **Estamos ansiosos para continuar a cooperação com os usuários do Syncovery**.

REFERENCES

***1 RemObjects PascalScript**: `https://github.com/remobjects/pascalscript`
***2 Superflexible AG** `https://www.syncovery.com`
***3 Astounding Desktop Synchronization Software Displays Delphi Flexibility** // Embarcadero `https://blogs.embarcadero.com/astounding-desktop-synchronization-software-displays-delphi-flexibility/`

BLAISE PASCAL MAGAZINE 114/115

Bancos de dados / Estilos CSS / Aplicativos Web progressivos
Android / IOS / Mac / Windows & Linux

ASSINATURA ANUAL

- ## ACESSO GRATUITO À INTERNET PARA TODAS AS EDIÇÕES

  Isso significa que, se o senhor se tornar um assinante, também terá acesso a todas as edições mais antigas. Todos os assinantes recebem uma assinatura gratuita da Internet, - ***mas ela está disponível somente em inglês.***

- ## CÓDIGO DOS ARTIGOS

  Se o senhor se tornar um assinante, incluiremos retroativamente todo o código da edição de 1/1/2023 até hoje, gratuitamente.

- ## SERVIÇO

  neste primeiro ano de assinatura, o senhor receberá a versão em inglês gratuitamente.

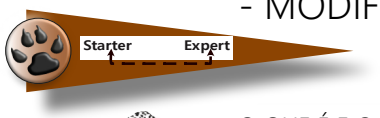
- ## PERÍODO DE ASSINATURA

  O período de uma assinatura inclui 1 ano, 8 edições com pelo menos 60 páginas de conteúdo cada. Na prática, nos esforçamos para publicar 10 edições e, na realidade, são 80 e, muitas vezes, mais de 100 páginas - como nesta edição, 154 páginas

- ## CUSTO POR ANO - PARA O BRASIL:

  R$250 ou €45 (euros). Nos outros países do mundo custa € 70 euros.

O DEBUGGER LAZARUS PARTE 5 A MUDANÇA ACONTECE - MODIFICANDO DADOS



POR MARTIN FRIEBE

O QUE É E O QUE DEVERIA SER...

Em todos os artigos anteriores, sempre que encontrávamos o bug e o corrigíamos, recompilávamos e reiniciávamos o aplicativo para ver se ele funcionava. Em nossos aplicativos de amostra, isso não foi problema.

O aplicativo executaria o código que corrigimos e poderíamos ver imediatamente se o novo código funcionava, mas e se o aplicativo levasse muito tempo para alcançar o código que corrigimos?

E se tivéssemos que interagir longamente com ele antes de sabermos se a correção era boa?

E se não tivéssemos certeza sobre a correção, mas quiséssemos apenas testar como ela funcionária se tivesse obtido um valor diferente para uma das variáveis?

Então, reiniciar e esperar que o código seja executado novamente pode ser muito incômodo.

Dependendo do problema, podemos manter o aplicativo em execução, mas testar o que aconteceria se as coisas fossem diferentes.

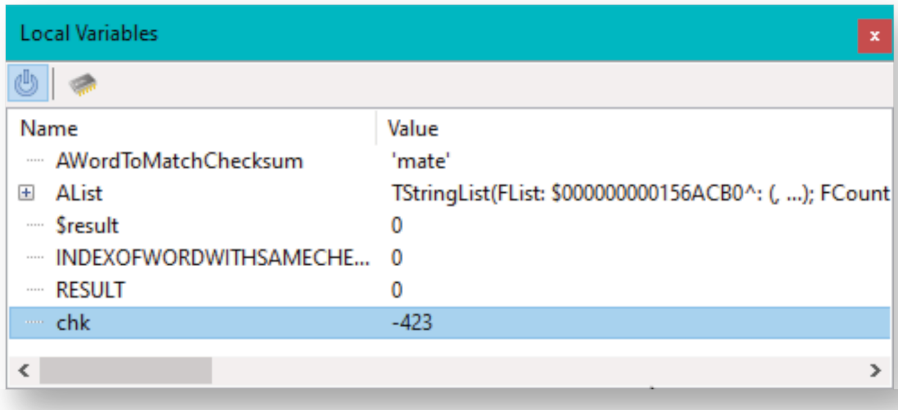
Isso não funciona para todos os tipos de alterações. Ele se limita a alterações nos dados dos aplicativos. Vamos explorar isso em um exemplo.

Nosso exemplo é bastante simples. Vamos apenas fingir que seria uma dor de cabeça executá-lo novamente. Ele procura em uma lista uma palavra com as mesmas palavras que uma determinada palavra.

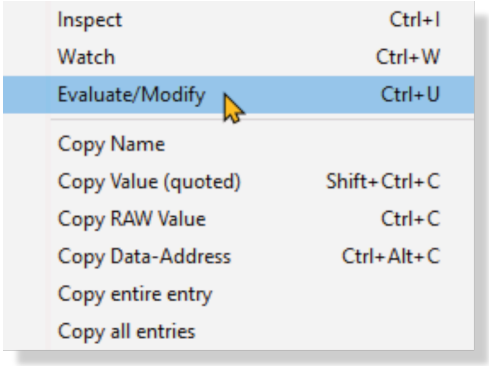
```
1. program project1;
2.
3. uses Classes;
4.
5. function Checksum(s: string): integer;
6. var
7.   i: Integer;
8. begin
9.    Result := 0;
10.    for i := 1 to Length(s) do
11.     Result := Result + ord(s[i]);
12. end;
13.
14. function IndexOfWordWithSameChecksum(AWordToMatchChecksum: String;
                                          AList: TStringList): Integer;
15. var
16.   chk: Integer;
17. begin
18.   if AList = nil then
19.     exit(-1);
20.   chk := -Checksum(AWordToMatchChecksum);
21.   Result := AList.Count - 1;
22.   while Result >= 0 do begin
23.     if Checksum(AList[Result]) = chk then
24.       exit;
25.     dec(Result);
26.   end;
27. end;
28.
29. var
30.    Words: TStringList;
31.    i: Integer;
32. begin
33.    Words := TStringList.Create;
34.    Words.Add('words');
35.    Words.Add('tame');
36.    Words.Add('town');
37.
38.    i := IndexOfWordWithSameChecksum('mate', Words);
39.    writeln(i);
40. end.
```

A MUDANÇA ACONTECE - MODIFICANDO DADOS

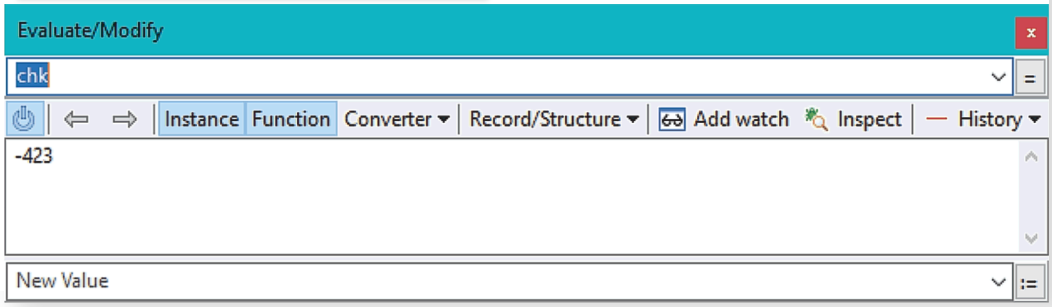
Como a soma de verificação é a soma dos valores ordinais de cada letra, as palavras com as mesmas letras em qualquer ordem são palavras com a mesma soma de verificação.
No exemplo, esperamos que "mate" seja correspondido por "**tame**".
Portanto, ele deveria imprimir **"1"**.
No entanto, ele imprime "**-1**", indicando que não encontrou nada.
Como de costume, iniciamos o debugging e interrompemos na linha 18, a primeira linha em "`IndexOfWordWithSameChecksum`". Lá, podemos usar a janela local para dar uma olhada nas variáveis envolvidas. Em seguida, examinamos a atribuição da soma de verificação a "**chk**":



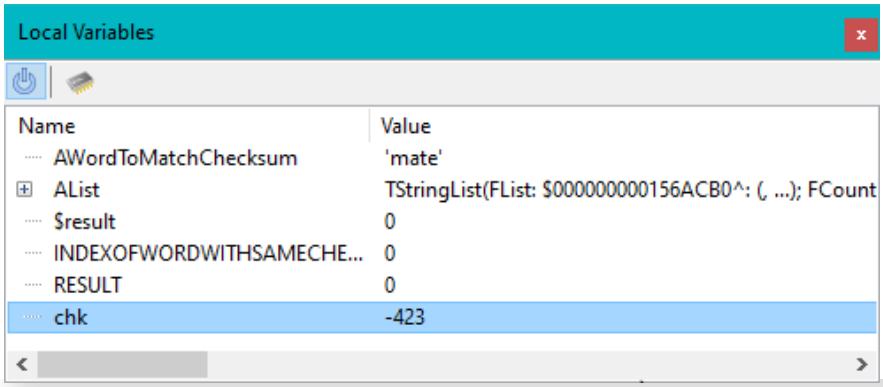
E notamos que "**chk**" tem um valor de **-423**, mas um valor negativo não é o que esperamos, pois somamos os valores ordinais de todos os caracteres em "mate" e todos eles são números inteiros positivos. Em uma inspeção mais detalhada do código-fonte, notamos que um erro de digitação foi inserido em nosso código. Há um sinal de menos unário na frente da chamada para "**Checksum**". O verdadeiro **checksum** em "**chk**" deveria ser **423.**
Agora, podemos corrigir isso alterando o código e reiniciando o aplicativo. Ou podemos alterar o valor
no aplicativo em execução e continuar a execução atual com o valor correto. No menu **pop-up** da janela de locais, escolhemos "**Evaluate/Modify**" (*Avaliar/Modificar*) na variável "**chk**":



Isso abrirá a janela correspondente.



Essa janela oferece mais uma maneira de avaliar e visualizar valores. No entanto, por enquanto, vamos nos concentrar no recurso "**Modify**". A variável que queremos alterar já está preenchida e seu valor é exibido. Na parte inferior, há uma edição para inserir um "**New Value**" e um botão **":="** para aplicá-lo.
Inserimos o valor positivo 423 na edição e pressionamos o botão. Imediatamente, podemos ver o valor sendo atualizado. Tanto a janela "**Evaluate/Modify**" quanto a janela local agora mostram **423** para a variável "**chk**". Com isso feito, continuamos nosso aplicativo até o final de "`IndexOfWordWithSameChecksum`", onde verificamos o valor de "**Result**" na janela de locais.



E aí podemos confirmar que, desta vez, o resultado tem o valor correto de "**1**", que também será impresso quando deixarmos o aplicativo ser executado até o fim. Alterar o valor sem ter de reiniciar o aplicativo nos poupou algum tempo. No entanto, depois que o erro for confirmado e soubermos como alterar o código, ainda teremos que recompilar. Caso contrário, o valor estará errado toda vez que a função for inserida, e teremos que modificar o valor várias vezes.

Poderíamos definir um ponto de interrupção e fazer isso algumas vezes, mas, eventualmente, alterar manualmente a variável perderia a vantagem de recompilar e reiniciar.
Por outro lado, se a alteração não tivesse corrigido o problema, poderíamos ter tido outra chance de investigar mais a fundo, quando a função fosse chamada novamente.
No fim das contas, cada caso é diferente e os benefícios precisam ser verificados em cada sessão de depuração.

QUANDO A MUDANÇA NÃO É UMA OPÇÃO

O recurso "**Modify**" tem algumas limitações. Atualmente, ele só funciona para tipos ordinais (`números, enum, booleano, ...`). Para estruturas e matrizes, os campos individuais podem ser alterados. Infelizmente, as cadeias de caracteres ainda não podem ser modificadas.

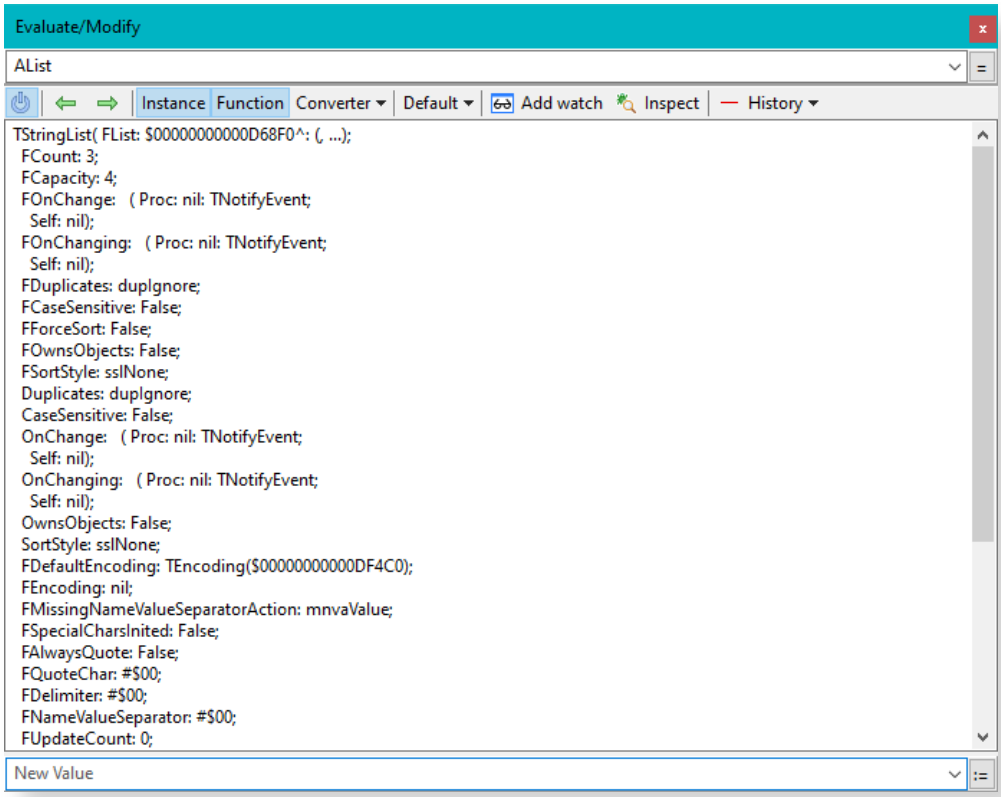
A PARTE DE "AVALIAÇÃO"

O nome completo da janela é **"Avaliar/Modificar**". Já vimos a parte "**Modify**" e agora vamos explorar a parte "**Evaluate**".

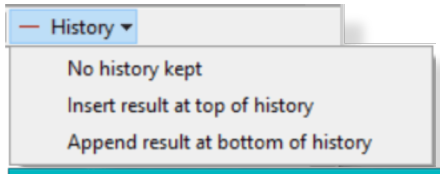
Resumindo, a janela avaliar funciona como uma janela para um único watch, mostrando o resultado da mesma forma que a janela de watches faz no "painel de detalhes".
A principal vantagem é que todas as configurações da caixa de diálogo "**Propriedades dos watches**" são diretamente acessíveis, de modo que é muito mais fácil alterá-las ou a expressão a ser avaliada.

E, usando quase toda a janela para mostrar o resultado, é muito mais fácil lê-lo do que no "**painel de detalhes**". Isso é especialmente útil para estruturas, mas também pode ajudar com despejos de memória **dumps de memória**, matrizes ou `strings` longas.

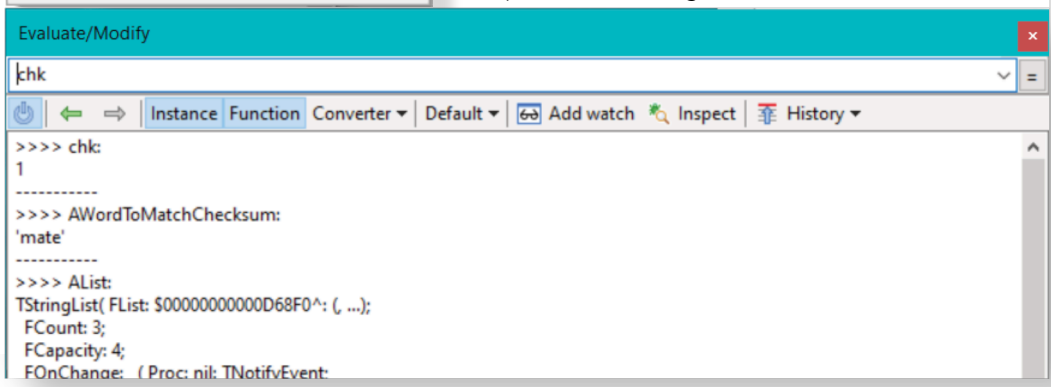
A avaliação de "AList" mostrará o seguinte.



Todos os campos são facilmente visíveis, e até mesmo os valores aninhados, como nos eventos (*por exemplo*, "`OnChange`"), são sublinhados. A barra de ferramentas oferece as mesmas propriedades da caixa de diálogo de propriedades dos watches, que foi explicada no último artigo. Embora só seja possível ver uma variável de cada vez, você pode ir e voltar entre os valores que que você avaliou usando os botões de seta verde. Você também pode selecionar valores anteriores no menu suspenso. E, se necessário, você pode manter os valores anteriores visíveis e inserir novos resultados em cima ou embaixo deles ou na parte inferior deles. O botão de ferramenta "**History**" oferece as seguintes opções:



Apresentando a seguinte visão:



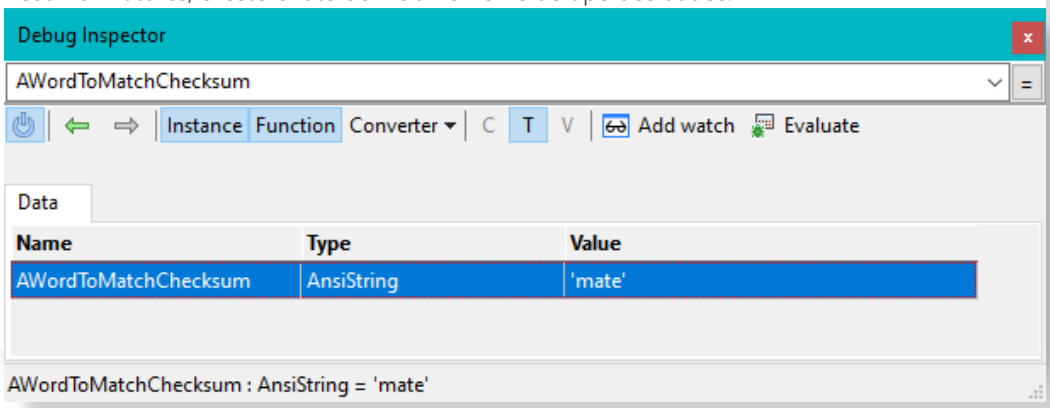
O INSPETOR DE DEBUG

Embora o tópico deste artigo seja a janela **Avaliar/Modificar,** vamos estendê-lo um pouco para examinar uma outra janela para visualizar dados do aplicativo:

O "**Inspetor de debug**".

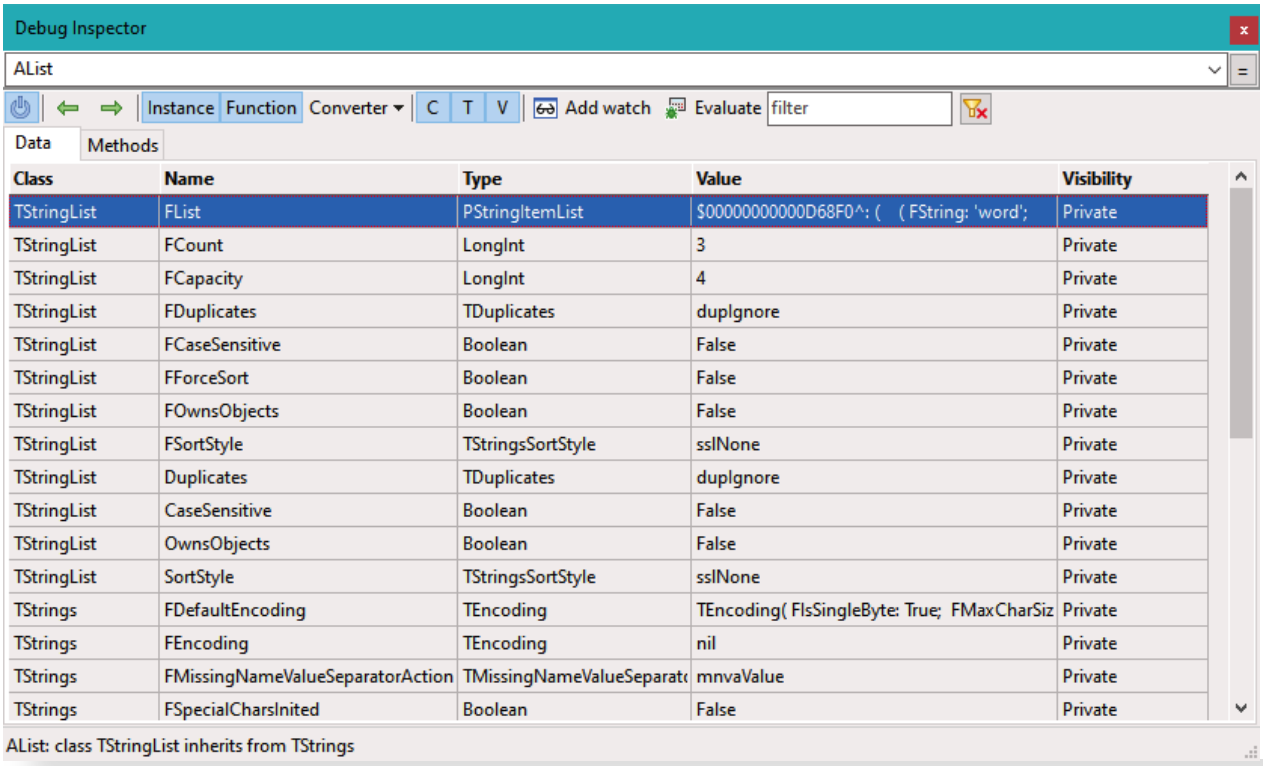
Semelhante à janela **Avaliar**, ela permite visualizar um watch de cada vez.

Para valores simples, ela não oferece muito mais do que qualquer uma das outras maneiras de visualizar **watches**, exceto o fato de incluir o nome do tipo dos dados.

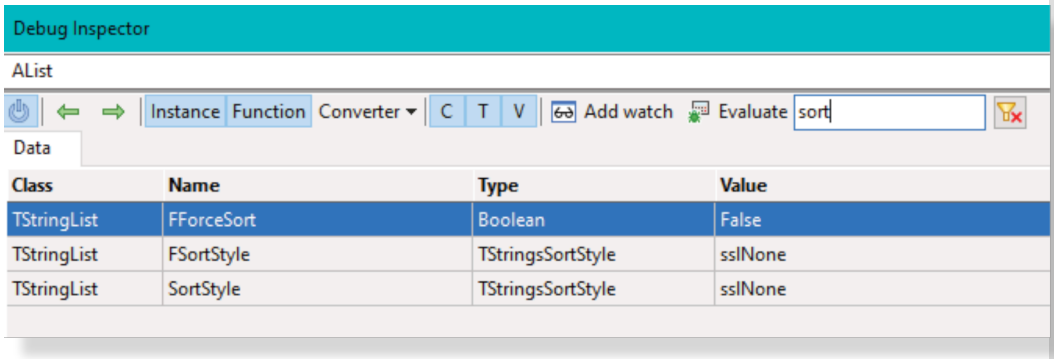


Ele tem todo o seu potencial para tipos estruturados e matrizes.

Ele lista cada campo ou entrada como uma linha separada.

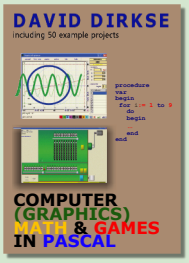


As linhas podem ser clicadas duas vezes para inspecionar o valor de um campo ou entrada para matrizes. Isso também funciona para diferenciar valores de ponteiro. Juntamente com a navegação **para frente/para trás,** os dados podem ser facilmente explorados, e qualquer valor inspecionado que seja de interesse mais permanente pode ser adicionado como **watch** com o clique de um botão. Além disso, o inspetor oferece um filtro para pesquisar os dados. Esse filtro só está disponível para valores estruturados e matrizes. Qualquer valor inserido será comparado com todas as colunas visíveis.
Assim, é possível pesquisar campos que contenham algum texto nos dados, mas também é possível pesquisar um campo pelo nome. Se em "**AList**" quisermos encontrar campos relacionados à classificação, poderemos inserir "**sort**" no filtro:



Se for o "`FSortStyle`" que estamos procurando, podemos pressionar o cursor para baixo para navegar até ele, e **Ctrl-Enter** para selecioná-la e alterar a expressão inspecionada para `Alist.FSortStyle`.
Isso permite uma navegação muito rápida, mesmo em objetos com muitos campos.
Em matrizes, isso também pode corresponder ao índice, mas somente dentro da lista de índices na página atual (*as matrizes são paginadas como na janela de* **watches**).

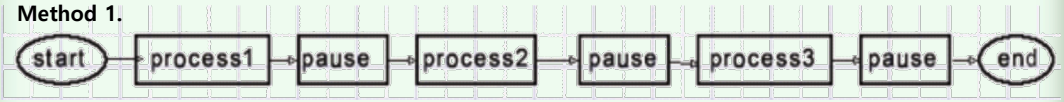
POR DAVID DIRKSE



ABSTRACT

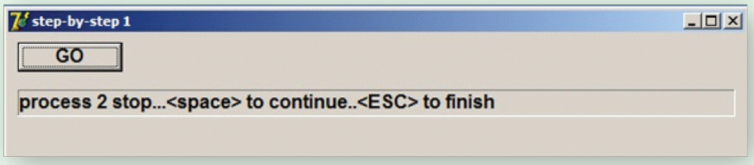
Para o software educacional, é exploratório mostrar as etapas individuais em direção a uma solução. Este artigo descreve duas maneiras de fazer isso. Em ambos os casos, um processo é composto por processo1, processo2 e processo3, que podem ser executados passo a passo.

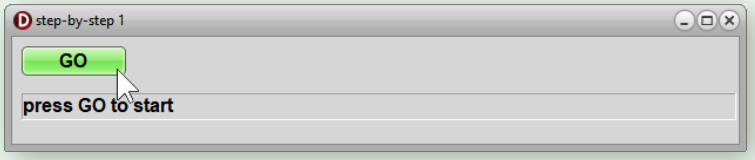
Method 1.

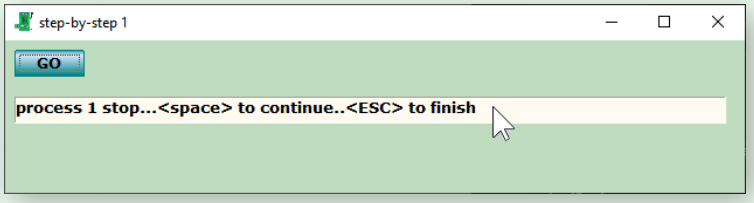


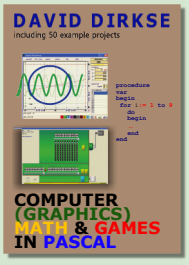
O procedimento de pausa define uma stopflag (Booleana) e chama application.processmessages até que a stopflag seja apagada. Essa limpeza é feita pressionando-se a barra de espaço. Quando a operação passo a passo não for mais desejada, pressionar a tecla escape finaliza o processo completo sem pausas. Isso é controlado pela abortflag (Booleana). O procedimento de pausa verifica o sinalizador de abortar e, se estiver definido, sai imediatamente. Pressionar a tecla escape define o sinalizador de abortar e limpa o sinalizador de parada.

```
const  msg1 = 'press GO to start';
       msg2 = '<space> to continue..<ESC> to finish';
var stopflag, abortflag : boolean;

procedure pause;
begin
  if abortFlag = false then
  begin
    stopFlag := true;
    form1.msglabel.Caption := form1.msglabel.caption + '...' + msg2;
    while stopFlag do application.processmessages;
  end;
end;

procedure process1;  //similar for process2,3
begin
  form1.msglabel.caption := 'process 1 stop';
end;

procedure TForm1.startBtnClick(Sender: TObject);
begin
  activecontrol := nil;
  abortFlag := false;
  process1;
  pause;
  process2;
  pause;
  process3;
  pause;
  msglabel.Caption := 'finished';
end;

procedure TForm1.FormKeyDown(Sender: TObject; var Key: Word;Shift: TShiftState);
begin
  case key of
    VK_SPACE   : stopflag := false;
    VK_ESCAPE  :  begin
                    abortflag := true;
                    stopflag := false;
                  end;
  end;//case
end;
```

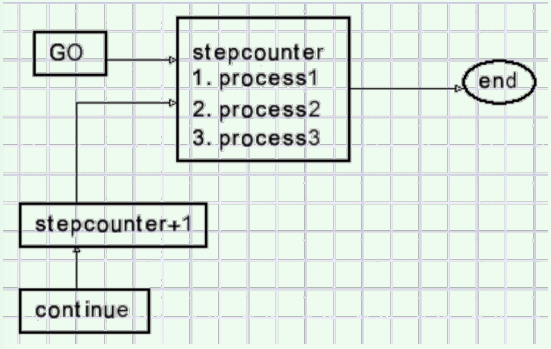




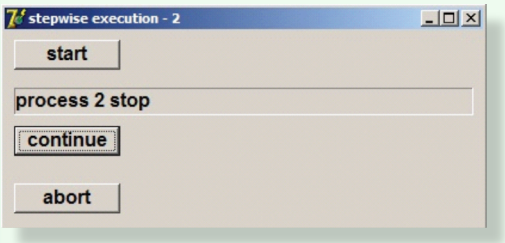




Method 2.



Esse método não usa o procedimento de pausa, mas botões para percorrer o processo. O contador de etapas (byte) seleciona os processos.
Pressionar GO coloca o contador de etapas em zero. O procedimento Nextproc aumenta o stepcounter 1 e chama o processo1.



```
var stepcount : byte;

procedure process1,2,3.....same as above;

procedure nextproc;
begin
  case stepcount of
    0 : begin
          stepcount := 1;
          nextproc;
        end;
    1 : process1;
    2 : process2;
    3 : process3;
    4 : begin
          form1.msglabel.Caption := 'finished';
          stepcount := 0;
        end;
   end;//case
end;
```

Começando o processo.....

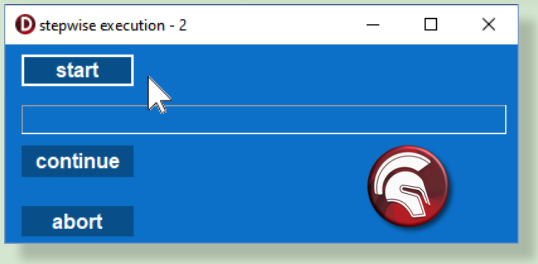
```
procedure TForm1.startBtnClick(Sender: TObject);
begin
  stepcount := 0;
  nextproc;
end;
```

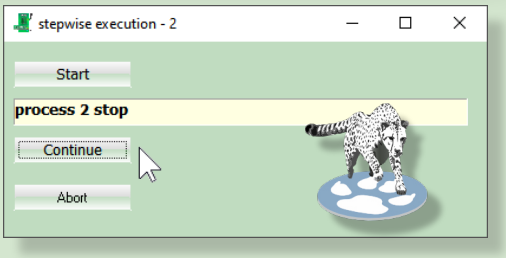
Próximo passo do processo....

```
procedure TForm1.continueBtnClick(Sender: TObject);
begin
  if stepcount > 0 then
    begin
      inc(stepcount);
      nextproc;
    end;
end;
```

finish process without pause.....

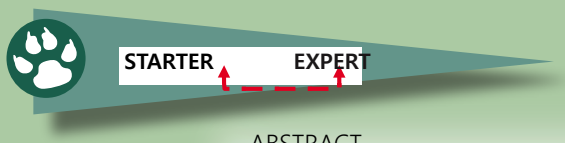
```
procedure TForm1.abortBtnClick(Sender: TObject);
begin
  while (stepcount > 0) and (stepcount < 4) do
    begin
      inc(stepcount);
      nextproc;
    end;
end;
```





Isso conclui a descrição passo a passo da execução do programa.

Lazarus 1-3



ABSTRACT

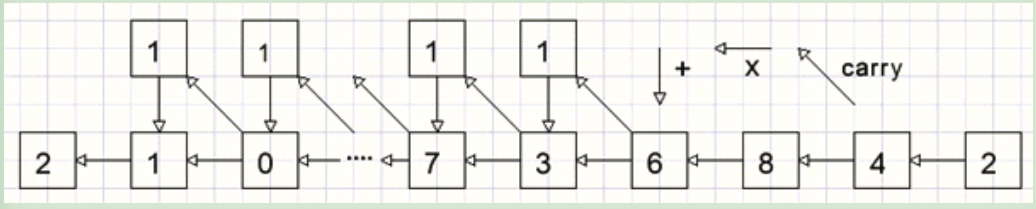
Um quebra-cabeça matemático bem conhecido é o seguinte:
Um número tem 2 como o dígito mais baixo. Se esse dígito for movido para a esquerda do número, o efeito será a multiplicação por 2.
Semelhante:
Um número tem 2 como o dígito mais à esquerda. Se esse dígito for movido para a direita do número, o efeito será a divisão por 2. **O engraçado desse quebra-cabeça** é que ele requer apenas cálculo de escola primária, no entanto, até mesmo os professores de matemática têm problemas para resolvê-lo.
Depois de algumas tentativas manuais, surge a pergunta se esse número existe de fato.
Além disso: há outros dígitos possíveis além de 2 e: há outros fatores possíveis além de 2?
Este artigo descreve um programa Delphi que procura um número com essas propriedades.

O ALGORITMO.

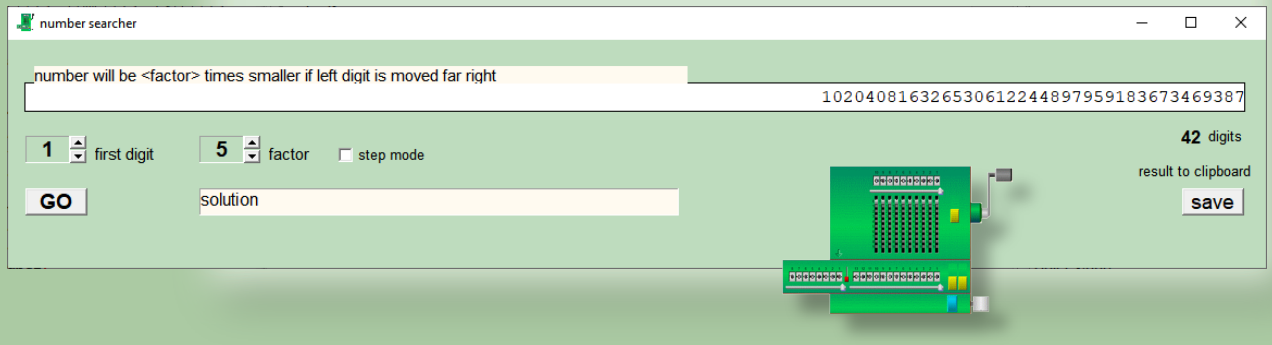
Para fins de explicação, usamos o dígito 2 como início e um fator de multiplicação de 2.
Observação: um transporte é escrito como (...)

```
To start                                    2
2 x 2 = 4                                  42
2 x 4 = 8                                 842
2 x 8 = 16                              (1)6842
2 x 6 = 12 + 1 = 13                     (1)36842
2 x 3 = 6 + 1 = 7                         736842
Finally                                105…736842
2 x 1 = 2                             2105…736842
```

Agora o primeiro e o último dígito são iguais (e não há transporte) descarte o dígito "2" mais baixo e teremos encontrado um número que divide por 2 se o primeiro dígito for movido para a direita do número.



O PROGRAMA



A imagem acima mostra o programa em funcionamento.
Os controles **UpDown**, associados aos componentes de texto estático, selecionam o primeiro dígito e o fator de multiplicação.
O botão **GO** inicia a pesquisa.
O botão **SAVE** copia o resultado para a área de transferência, o que permite a colagem em editores de texto. Se o modo de etapa estiver marcado, o programa para após cada iteração, mostrando os resultados intermediários.
O número é armazenado em `byte array A[0…120]`
O número inteiro N é o índice de `array A[ ]`.
Inicialização:

```
N := 0;
A[0] := start digit.
```

Passo 1:
`A[N]` é multiplicado por 2, o resultado é colocado em `A[N+1]` (dígito inferior) e `A[N+2]` (transporte).

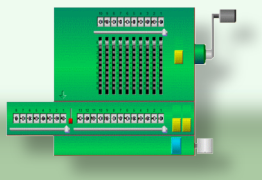
Passo 2:

```
N := N + 1;
```

As etapas 1 e 2 são repetidas até que `N = 120` (sem solução) ou `A[N] = A[0]` e `A[N+1] = 0` (sem transporte).

Este código faz o trabalho:

```
Const maxN = 120;  //maximal number of digits
.....
procedure TForm1.GObtnClick(Sender: TObject);

var carry,i,f,h,N : byte;
   hit : boolean;
begin
 activecontrol := nil;
 for i := 0 to maxN do A[i] := 0;
 displayA(0);
 A[0] := N0upDown.Position;  //starting digit
 f := factorUpdown.Position; //factor
//--
 N := 0;  //array A index
 repeat
  inc(N);
  A[N] := A[N-1]*f + A[N];
  h := 0;
  while A[N+h] >= 10  do   //handle carries
   begin
    carry := A[N+h] div 10;
    A[N+h] := A[N+h] mod 10;
    inc(h);
    A[N+h] := A[n+h] + carry;
   end;
  hit := (A[N] = A[0]) and (h=0);
until (N = maxN) or  hit;
//--
 if hit then begin
              msgText.Caption := 'solution';
             displayA(N);
           end
  else begin
      msgText.Caption := 'no solution';
      label4.Caption := '';
     end;
end;
```



Modo de etapa
Se **stepMode** for selecionado, após cada iteração, a variável stopflag será definida como verdadeira seguida por:
`while stopFlag do application.ProcessMessages;`

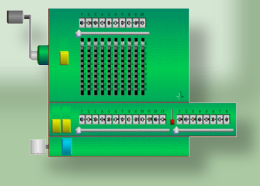
Pressionar `<barra de espaço>` redefine a **stopflag** para que o processo continue.
Isso precisa que a propriedade **keyPreview** seja definida como verdadeira no **ObjectInspector** do `form1`.
O **procedure** de pesquisa começou com
`Activecontrol := nil.`
Isso evita que o caractere `<espaço>` seja enviado para o botão **GO**, que tem o foco após ser pressionado.

Exibição da matriz A[ ]
Uso um **paintbox** para exibição.
O motivo é que isso permite a exibição de dígitos em cores diferentes.
Os carries são exibidos em vermelho.
A fonte **`Courier`** new é usada, com caracteres de largura fixa.
`A[1]` é colocado à direita, a posição do caractere x é decrementada para exibir os próximos caracteres à esquerda.
O `procedure displayA(n : byte);` faz o trabalho, n é o número de dígitos.
Qualquer caractere que diferente de zero de um índice mais alto de n de `A[n]` é pintado em vermelho.

Salvando o resultado na **Clipboard**.
A **Clipbrd Unit** é adicionada à cláusula uses.
Com o resultado salvo na `string s`, esta instrução faz o trabalho:
`clipboard.astext := s;`
Mas primeiro `A[ ]` deve ser copiado para `s`.
O dígito mais alto é transferido primeiro.
No contador `p = 3, " . "` é colocado para maior clareza, separando os dígitos triplos.

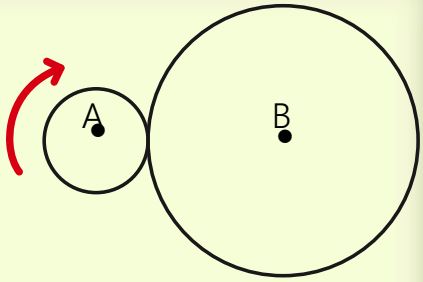
```
 var s : string;
   i,n,p : byte;
begin
 s := '';
 n := maxN;
 while (A[n] = 0) and (n > 0) do dec(n);//find first non zero digit
p := 0;
 for  i := n downto 1 do
  begin
    if p = 3 then begin
                    s := s + '.';
                    p := 0;
                  end;
   s := s + char(A[i] + ord('0'));
   inc(p);
 end;
end;
```

Consulte o código-fonte para obter mais detalhes.
Uma propriedade surpreendente da resposta (para o fator 11) é a seguinte:
Divida o número em duas metades e some-as.
O resultado mostra apenas `"9"` dígitos: `99.....99`.



STARTER EXPERT 7 D11

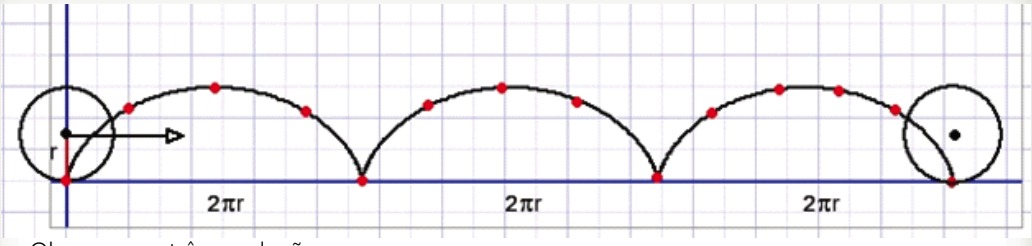
Na figura acima, o raio do círculo A é 1/3 do raio do círculo B.
A partir da posição mostrada na figura, o círculo A gira em torno do círculo B.
Ao final de quantas rotações do círculo A, o centro do círculo primeiro alcançará seu círculo de ponto inicial?



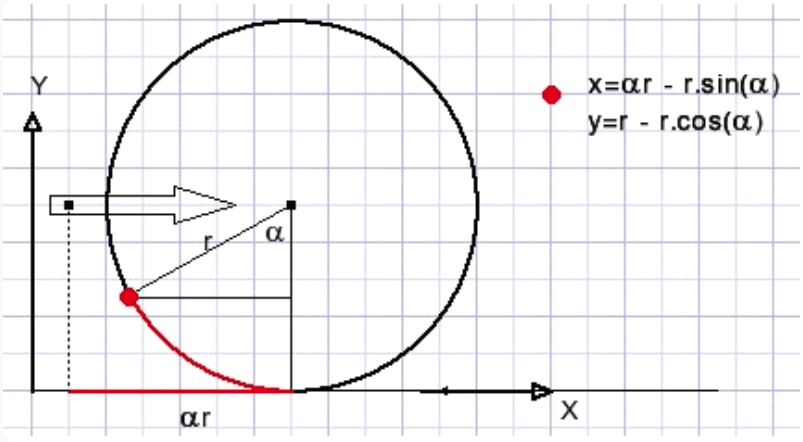
(A) $\frac{3}{2}$ (B) 3 (C) 6 (D) $\frac{9}{2}$ (E) 9

Sugiro que, antes de continuar lendo, o leitor tente responder a essa pergunta por si mesmo.

TEORIA
Podemos esticar o perímetro do círculo B e rolar o círculo **A** sobre essa linha reta.



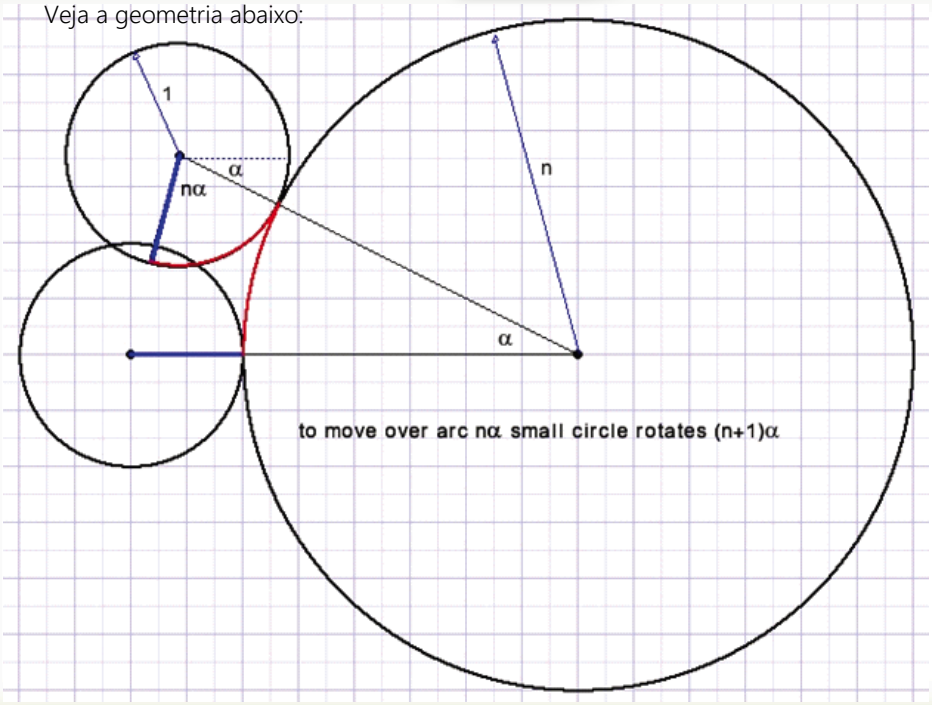
Observamos três revoluções.



For those interested in the math:

Para os interessados em matemática:
O movimento do ponto vermelho é dado por uma função paramétrica: x e y são ambos expressos como uma função do ângulo de rotação α.
No entanto, isso ainda é verdade quando o círculo A rola sobre o círculo B?
Resposta surpreendente: não é.
Veja a geometria abaixo:

Veja a geometria abaixo:



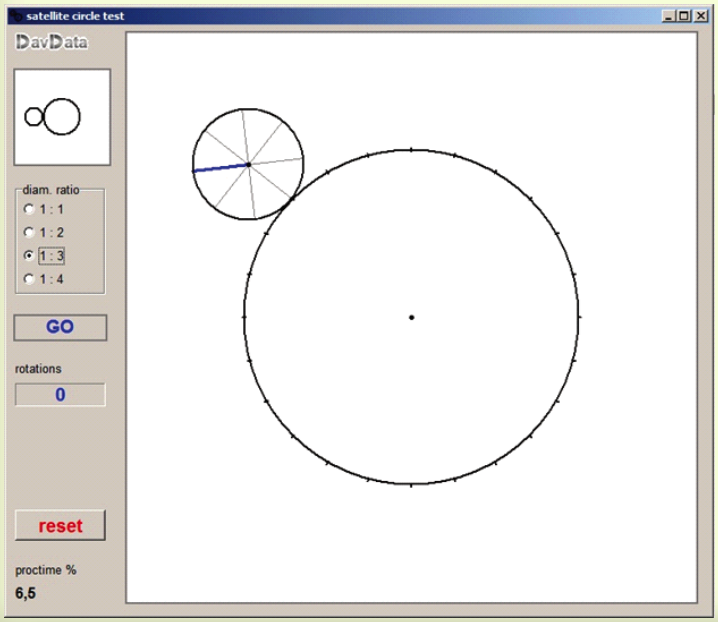
O círculo A precisa de quatro rotações para rolar completamente sobre o círculo B, e não três, como no caso de uma linha reta.

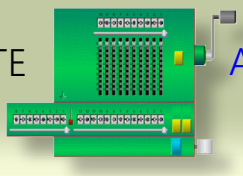
Para ilustrar esse fenômeno, escrevi um pequeno programa em Delphi.

O programa adiciona algumas opções

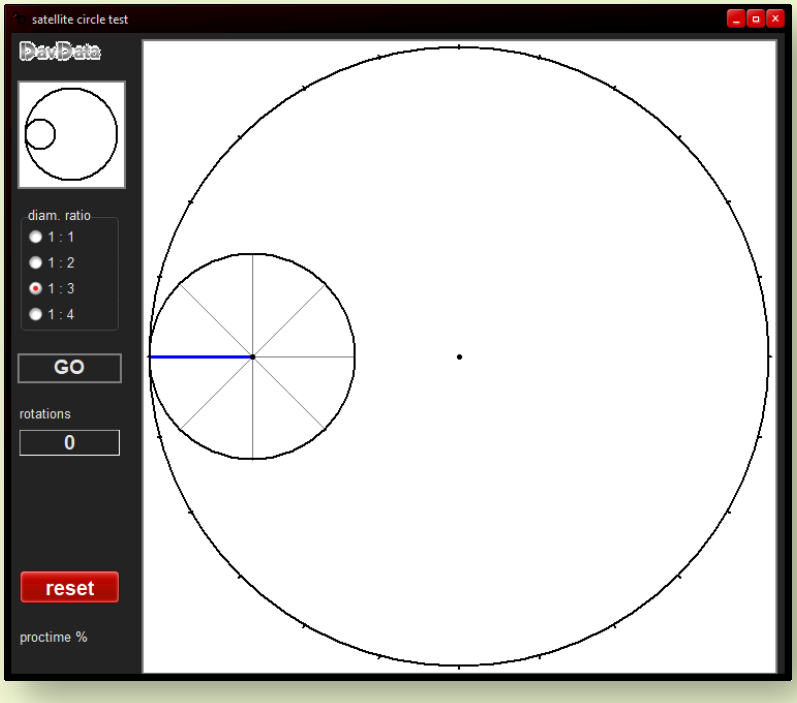
- Círculo externo ou interno A
- Razões de diâmetro 1:1 , 1:2 , 1:3 , 1 : 4

A teoria para o caso em que A é o círculo interno é deixada para o leitor

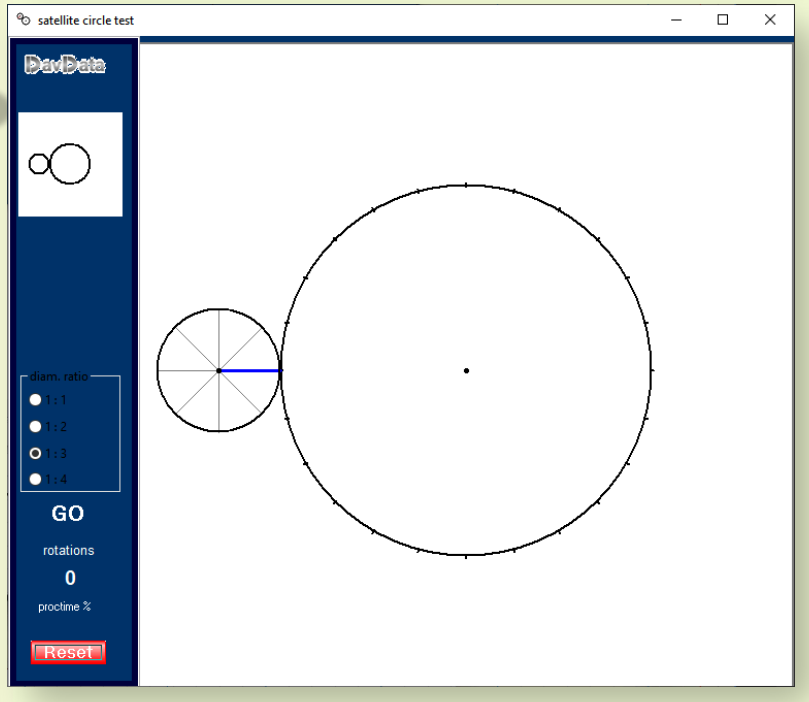


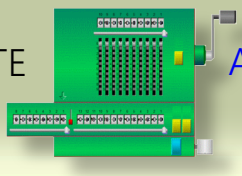


Exemplo em D11 e D12



Exemplo em Lazarus





Selecione a proporção do diâmetro.
Clique na imagem superior esquerda para selecionar A como círculo externo ou interno.
Pressione GO para rolar o círculo A sobre o círculo B.

DESCRIÇÃO DO PROGRAMA
O desenho é feito em um mapa de bitmap. O mapa é copiado para o paintbox1 no formulário principal para se tornar visível. A imagem é repintada **100 vezes por segundo.**
O rótulo inferior esquerdo mostra a porcentagem de tempo do processador necessária para uma atualização.
Veja a segunda imagem anterior.
O ângulo α começa em 0 e é incrementado em etapas de 0,005 radianos.

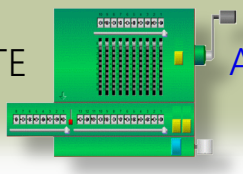
Uma revolução completa leva 2π/0,005 = 1250 etapas.
Com 100 incrementos por segundo, o tempo de revolução é de 12,5 segundos.

A pintura de uma nova imagem completa leva:
- Apagar o **bitmap**
- Pintar o círculo **B** com marcas apropriadas para coincidir com os raios do círculo **A**.
- Pintar o círculo **A** e seus raios.
- Copiar o mapa para a paintbox por `form1.paintbox1.canvas.draw(0,0,map)`

CONSTANTS AND VARIABLES

```
Type Tmode = (modeIn, modeOut);
   Tradius = record
       rad1 : word;  //radius of circle B
       rad2 : word;  //radius of circle A
       xamp : byte;  //rotation angle magnifier (=4 if ratio is 1:3)
   end;

Const  pi2 = 2*pi;
       pi04 = pi/4;
       radOut : array[0..3] of Tradius =    //outer circle mode radius
            ((rad1:100 ;rad2:100 ;xamp:2),  //1:1
            (rad1:150 ;rad2: 75 ;xamp:3 ),  //1:2
            (rad1:180 ;rad2: 60 ;xamp:4 ),  //1:3
            (rad1:200 ;rad2: 50 ;xamp:5));  //1:4
       radIn  : array[0..3] of Tradius =    //inner circle mode radius
            ((rad1:300 ;rad2:300 ;xamp:0),
            (rad1:300 ;rad2:150 ;xamp:1),
            (rad1:300 ;rad2:100 ;xamp:2),
            (rad1:300 ;rad2: 75 ;xamp:3));

var map : Tbitmap;
cx,cy : word;                 //center of circle B, measured in pixels
sx,sy : word;                 //center of circle A (satellite)
   radius1,radius2 : word;    //radius of circles B and A, preset from constants array
   angle : double;            //rotation angle of circle A center around B center
   amp           : byte;      //rotating angle magnifier, preset from constants array
   GOflag        : boolean;   //control rolling
   mode          : Tmode = modeOut;   //out- or inside circle A
   ModeEntered   : boolean;   //for mode button enter / leave control
   frotation     : double;    //circle A rotation in floating point format
   rotations     : byte;      //number of rotations of circle A
```

continuation of the code is on the next page



Continuação do código

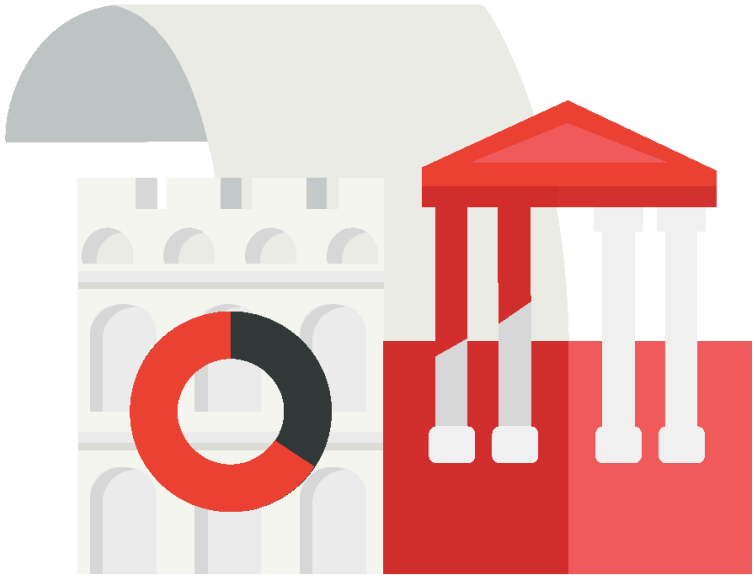
Procedure GOproc controls the rolling.

```
procedure GOproc;

var t1, t2 : Int64;

begin
   repeat
      t1 := getCPUticks;   //start CPU time
      paintmap;            //repaint image and copy to paintbox1
      t2 := getCPUticks;   //stop CPU time
      form1.proctimelabel.caption := formatfloat('0.#',proctime(t2-t1)*0.01);
                           //% of 10msec.
   repeat
      application.processmessages;   //GOflag may be cleared by GObutton release
      t2 := getCPUticks;
      until proctime(t2-t1) > 10000;  //wait for 10 milliseconds since t1
      if angle >= pi2 then angle := angle - pi2;
      angle := angle + 0.005;
      incRotation(0.005);             //rotation increment for small circle A
      GOflag := GOflag and (angle <= pi2);
      until GOflag = false;
end;
```

OBSERVAÇÃO:
o **GObutton** é, na verdade, um `TLabel`.
Ele se parece com um botão ao desenhar um retângulo ao seu redor na tela do formulário.
Os `events OnEnter, OnLeave, OnMouseDown` e `OnmouseUp` alteram as bordas do botão.
Não listarei aqui os procedimentos que pintam os círculos.
Consulte o código-fonte para obter detalhes.

CONCLUSÃO
As respostas do problema (veja a primeira figura) estão incompletas.
A resposta **(F)** -nenhuma das opções acima- ou **(F) 4** está faltando.
Essa já foi a questão 17 de um **SAT (Scholastic Aptitude Test**, *teste de aptidão escolar*) para alunos do ensino médio. Apenas alguns alunos notaram o erro.
Duvido que os professores que formularam essa questão estivessem cientes de seu erro.

Fast Reports
Os relatórios devem ser rápidos



O FastReport VCL é uma ferramenta de relatório para RAD Studio e Lazarus.

Ela ajuda seu aplicativo a buscar os dados da fonte de dados e criar documentos em qualquer formato com vários métodos de entrega.

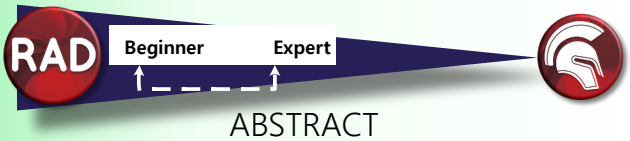
O gerador de relatórios FastReport VCL é uma solução moderna para a integração de Business Intelligence em seu software. Ele foi criado para desenvolvedores que desejam usar componentes prontos para geração de relatórios.
O FastReport VCL, com sua simplicidade de uso, conveniência e pequeno tamanho de distribuição, pode oferecer alta funcionalidade e desempenho em praticamente qualquer PC moderno.

- Inclui designer visual de relatórios
- Entrega na nuvem
- Salvamento em PDF, formatos de escritório (MS e Open), HTML e muito mais
- Edições com código-fonte
- Mais de 25 anos de experiência na criação de relatórios

Demonstrações gratuitas.
Experimente a sua em www.fast-report .com

❶ - O PRINCÍPIO DA RESPONSABILIDADE ÚNICA

POR MARCO GEUZE



ABSTRACT

Hoje, vamos nos aprofundar no S dos princípios sólidos: O princípio da responsabilidade única, mas antes de começarmos, vamos dar uma rápida recapitulada no Solid.

SOLID é um acrônimo para um conjunto de cinco princípios de desenvolvimento de software que, se seguidos, ajudam os desenvolvedores a criarem códigos flexíveis e limpos. Os cinco princípios são:

❶ O princípio da responsabilidade única
As classes devem ter uma única responsabilidade e, portanto, apenas um único motivo para serem alteradas.
❷ Princípio Aberto/Fechado
As classes e outras entidades devem ser abertas para extensão, mas fechadas para modificação.
❸ **Princípio da substituição de Liskov**
Os objetos devem ser substituíveis por seus subtipos.
❹ **Princípio da segregação da interface**
Os clientes não devem ser forçados a depender de interfaces que não utilizam.
❺ **Princípio da inversão de dependência**
Depender de abstrações em vez de concretizações.

Portanto, o princípio da responsabilidade única.
Como o nome sugere, cada classe em um programa deve ter uma única responsabilidade por apenas uma parte da funcionalidade do programa.
Mas isso parece mais fácil do que é.
O que exatamente é uma parte de um programa e como saber quando separar a funcionalidade?
É muito simples dizer que uma classe deve fazer apenas uma coisa.
Robert C. Martin expressa o princípio da seguinte forma: **"Reúna as coisas que mudam pelos mesmos motivos. Separe as coisas que mudam por motivos diferentes"** e, mais recentemente, **"Esse princípio é sobre pessoas".** Isso deve nos indicar a direção certa.

Quando você escreve um módulo de software, você quer ter certeza de que, quando forem solicitadas alterações, essas alterações só possam se originar de uma única pessoa, ou de um único grupo de pessoas fortemente acopladas representando uma única função comercial estritamente definida.
Isso significa que um módulo ou classe de software deve ter uma **única responsabilidade para esse grupo específico de pessoas.**
É mais fácil explicar isso por meio de um exemplo. Vamos dar uma olhada na classe a seguir:

```
type
  TShip = class
  private
    FPosition: TPoint;
    FHeading: Integer;
    FSpeed: Integer;
    FCargoLoad: string;
  public
    procedure SetHeading(NewHeading: Integer);
    procedure SetSpeed(NewSpeed: Integer);
    function GetCoordinate: TPoint;
    procedure PlotCourse;
    procedure LoadCargo(NewCargo: string);
    procedure PrintCargo;
    procedure ReportPosition;
    procedure CalculateProfit;
  end;
```

Continuação 1
Princípios solid

Esta é uma classe que está fazendo algumas coisas, todas relacionadas ao gerenciamento da posição e do curso de um navio, sua carga e algumas coisas de relatório. Como este é um breve exemplo para mostrar como pensar sobre o princípio da responsabilidade única, não preste muita atenção aos detalhes do código em si, o que importa é a visão geral da estrutura desse código. Trata-se de uma visão geral da estrutura dessa classe específica.

Acho que todos nós conhecemos esse tipo de classe "divina". Geralmente repletas de muitas funcionalidades e códigos, e gerenciam uma parte ou um módulo específico de seu programa. A questão é: se precisarmos fazer algumas alterações nessa classe, como podemos refatorar essa classe para garantir que reuniremos as que mudam pelo mesmo motivo e separar as coisas que mudam por motivos diferentes.

Vamos parar por um momento e pensar nas responsabilidades desse código em relação (*Os Parenteses nesse trecho não fazem sentido*) pessoas. Podemos definir algumas pessoas específicas que terão alguma responsabilidade com relação ao gerenciamento, à direção e ao rumo do navio e às ferramentas de relatório. Digamos que, nesse caso, definimos um capitão, um navegador, um encarregado de carga e um gerente financeiro. Se você pensar nessas diferentes funções, de repente fica muito fácil separar essa classe em diferentes módulos, com apenas uma responsabilidade por essa função específica. Deveríamos ter uma classe para definir o rumo e a potência (capitão), uma para gerenciar a posição e traçar o curso do navio (navegador), uma para gerenciar a carga do navio (capitão de carga) e uma para todos os nossos relatórios (financeiros) (gerente financeiro).
Nossas novas classes podem agora ter a seguinte aparência:

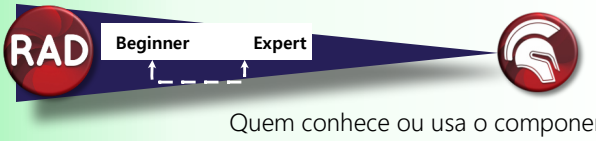
```
type
 TShipLocation = class
 private
  FPosition: TPoint;
 public
  function GetCoordinate: TPoint;
  procedure PlotCourse;
  procedure ReportPosition;
 end;

 TShipMovement = class
 private
  FHeading: Integer;
  FSpeed: Integer;
 public
  procedure SetHeading(NewHeading: Integer);
  procedure SetSpeed(NewSpeed: Integer);
 end;

 TCargo = class
 private
  FCargoLoad: string;
 public
  procedure LoadCargo(NewCargo: string);
  procedure PrintCargo;
 end;

 TShipReport = class
 public
  procedure CalculateProfit;
 end;

 TShip = class
 private
 public
  // reference to subclasses
 end;
```

Continuação 2
Princípios solid

Você teria feito o mesmo se não tivesse pensado nas pessoas por trás das responsabilidades da classe? Talvez, mas posso imaginar que as classes ShipLocation e ShipMovement poderiam ter acabado na mesma classe. Então, o que aconteceria se recebêssemos uma solicitação de recurso do capitão para adicionar um propulsor de proa para facilitar a direção do navio em canais pequenos?
Basta fazer essa alteração na classe ShipMovement, sem afetar nenhuma das outras classes.
E se quisermos implementar um novo sistema de carregamento de carga? Basta alterar a classe Cargo, novamente sem tocar em nenhuma das outras classes.
Espero que agora você tenha entendido por que o princípio da responsabilidade única é realmente sobre pessoas ou atores, e a responsabilidade da funcionalidade dos módulos ou classes de seu programa em relação a essas pessoas. E, é claro, você pode aplicar isso em diferentes níveis do seu programa, de módulos a classes a funções específicas. Se você sempre tiver esse princípio em mente ao refatorar ou projetar um módulo ou classe, tenho certeza de que seu código terá melhor manutenção e será fácil de alterar.

TRABALHANDO COM O LOCAL SQL DO FIREDAC

POR KEES DE KRAKER GDK software



Quem conhece ou usa o componente **LocalSQL** do Firedac? Ou o **BatchMove**? Para a maioria dos desenvolvedores, esses componentes são relativamente desconhecidos, embora sejam muito úteis. Em dois artigos, quero explicar o valor agregado desses componentes e mostrar como eles funcionam. Este artigo é sobre o componente **LocalSQL**.

O VALOR AGREGADO

Usando o conjunto de dados **LocalSQL**, você pode combinar dados de diferentes conjuntos de dados em uma consulta do **Firedac**.
O componente garante que os conjuntos de dados fiquem disponíveis como se fossem tabelas de banco de dados.
Os conjuntos de dados podem, portanto, ser consultados com uma consulta **SQL** de um componente **TFDQuery**.

Isso é muito útil se você tiver dados de fontes diferentes.
Por exemplo, de vários bancos de dados.
Ou parcialmente na memória e parcialmente no banco de dados. Ou de diferentes componentes de conjunto de dados, por exemplo **ADO** e **Firedac**. Com o **LocalSQL**, você pode reunir esses dados muito facilmente com uma simples consulta **SQL simples**. Ele permite mostrar os dados combinados de forma muito simples em uma grade ou criar estatísticas combinadas.

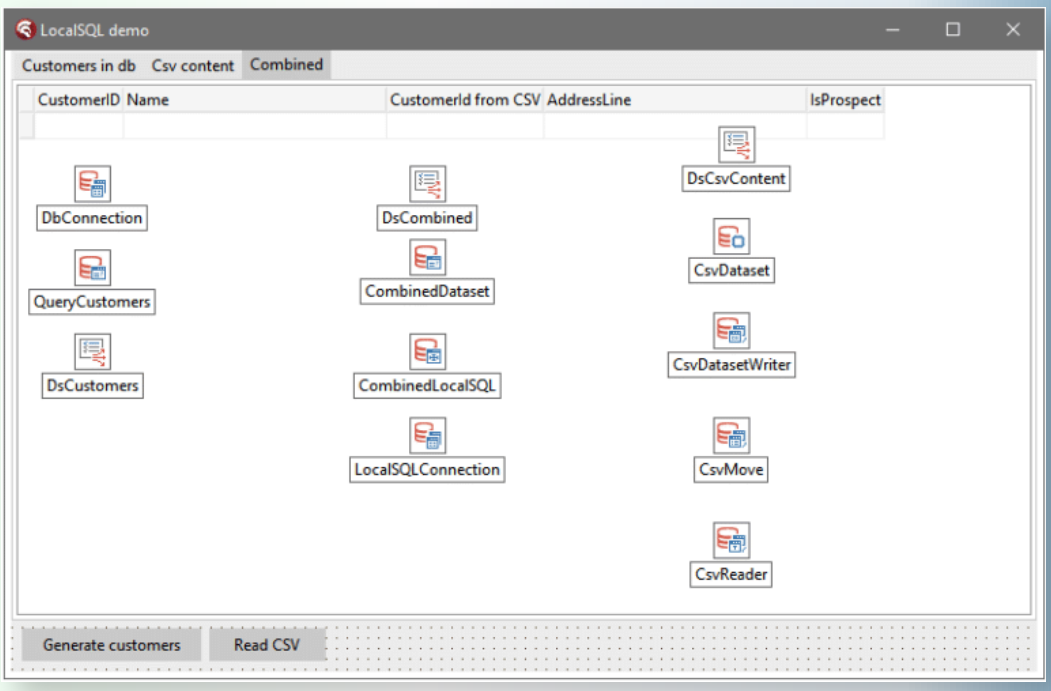
Abaixo, apresento um exemplo, combinando dados de um **CSV** com dados do banco de dados.
No banco de dados, armazenei clientes com um **ID** e um nome.
No **CSV**, apenas o id do cliente é usado.
Ao exibir o conteúdo do **CSV**, quero mostrar o nome do cliente diretamente. Isso pode ser feito com o **LocalSQL** sem muito esforço.

Continuação 1
Sql local

O PROJETO BÁSICO
Abaixo, você pode ver a configuração do meu aplicativo. À esquerda está a conexão com o banco de dados que contém a tabela **Customer**. Isso é lido por meio da **QueryCustomers**. À direita, você vê o **CsvDataset**, um **Firedac MemTable**. Nele, o **CsvMove** (`TFDBatchMove`) carrega os dados do arquivo **CSV**.

No meio, você vê o componente LocalSQL, que chamei de **CombinedLocalSQL**. Abaixo, esse componente usa **SQLite** e, portanto, requer uma conexão **SQLite**. Você não precisa configurar mais essa conexão, apenas indicar que o **driver** é **SQLite**. Nesse caso, é o **LocalSQLConnection**. O componente **CombinedDataset** é um `TFDQuery` que contém a consulta `SQL` que coleta os dados dos conjuntos de dados.

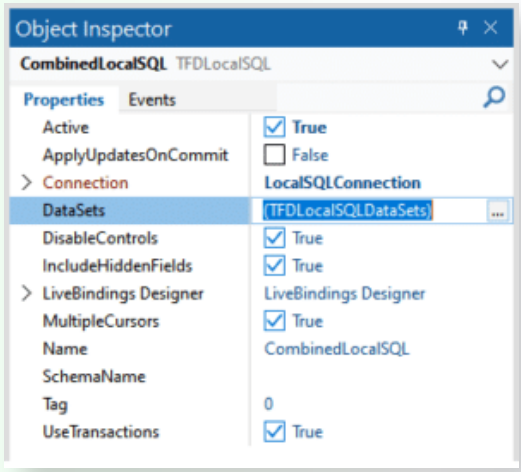
A grade mostra primeiro duas colunas com os dados do cliente do banco de dados.
As três colunas depois disso vêm do **CSV**

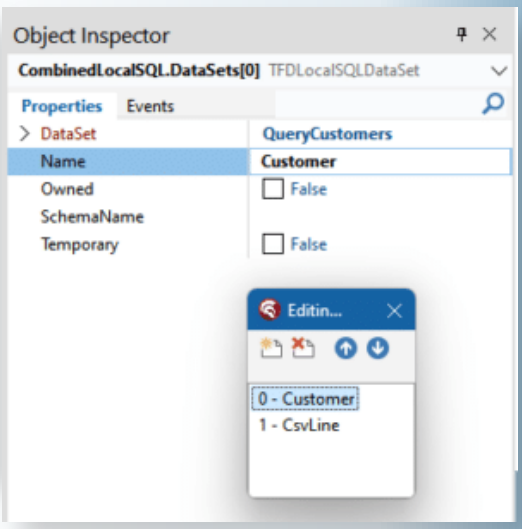


Continuação 2
Sql local

A CONFIGURAÇÃO
O componente LocalSQL, conforme mencionado anteriormente, disponibiliza os conjuntos de dados como se fossem tabelas de banco de dados. Para que isso seja possível, especificamos quais conjuntos de dados queremos usar e qual "nome de tabela" eles receberão. Nas propriedades do componente LocalSQL, você encontrará a opção "**DataSets**" para essa finalidade (veja as imagens abaixo). Adicione os conjuntos de dados e dê a eles o nome que você deseja usar como o nome da tabela na consulta.

Ao definir o componente **LocalSQL** como **Active** (*Ativo*), você poderá executar a consulta em tempo real depois disso e facilmente adicionar os campos, por exemplo. Tenha cuidado com isso, pois, depois de executar a consulta, os conjuntos de dados subjacentes também são definidos como **Active**





A consulta combinada
Usamos uma consulta **Firedac** para aplicar o LocalSQL. A conexão dessa consulta é a mesma que a conexão da conexão do **LocalSQL**. E isso é tudo. Agora podemos começar a criar a consulta como se tivéssemos uma tabela
Customer e uma tabela **CsvLine**.

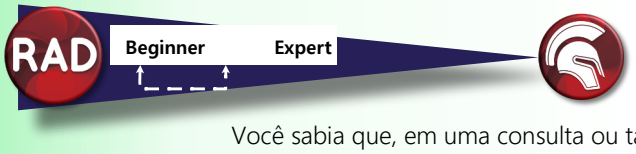
```
SELECT *
FROM Customer c
JOIN CsvLine csv ON (csv.CustomerId = c.CustomerId)
```

E, é claro, você pode estender e usar isso com todas as possibilidades do SQL (SQLite). Pense na cláusula **WHERE**, mas também nas funções de agregação, como **SUM, COUNT** etc. A consulta Firedac funciona como de costume.

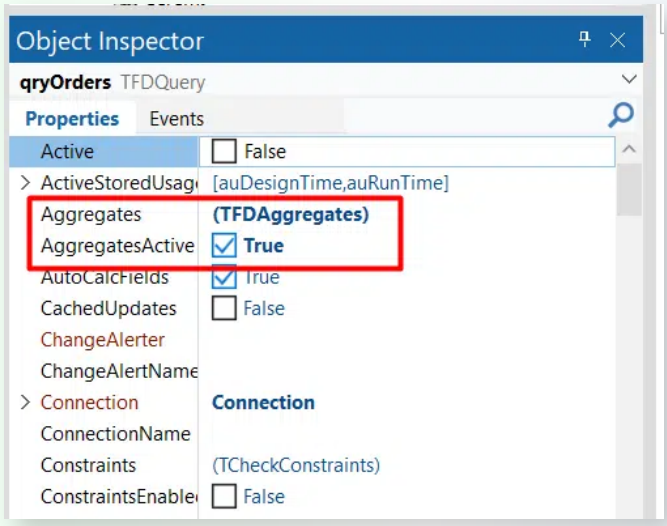
EM RESUMO
O componente **LocalSQL** abre uma diversidade de possibilidades para combinar, filtrar e agravar dados. Ele é amplamente aplicável porque suporta todas as formas de conjuntos de dados.
Ele pode tornar conjuntos de dados complicados com campos de pesquisa muito mais simples.
E funciona de forma rápida e fácil. É altamente recomendável aplicá-lo em seus projetos.

AGREGAÇÕES DE CONJUNTOS DE DADOS DO FIREDAC

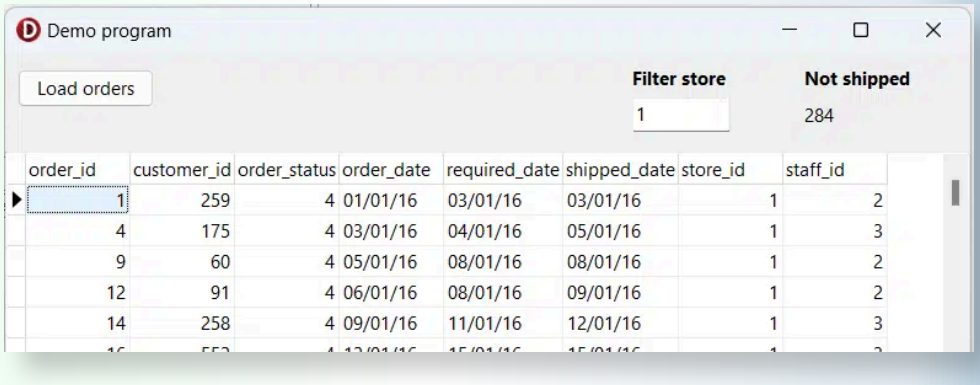
POR KEES DE KRAKER



Você sabia que, em uma consulta ou tabela do **Firedac**, é possível trabalhar com agregações em tempo de execução no nível de conjunto de dados?
A exibição de um valor total de todos os pedidos carregados ou de um total de todos os pedidos ainda a serem enviados pode ser feita com muita facilidade, sem a necessidade de uma consulta separada.
Talvez você já esteja familiarizado com um campo agregado, mas isso está limitado ao nível da linha. No entanto, com a propriedade **Aggegrates** de um `TFDDataSet`, você também pode totalizar no nível do conjunto de dados.



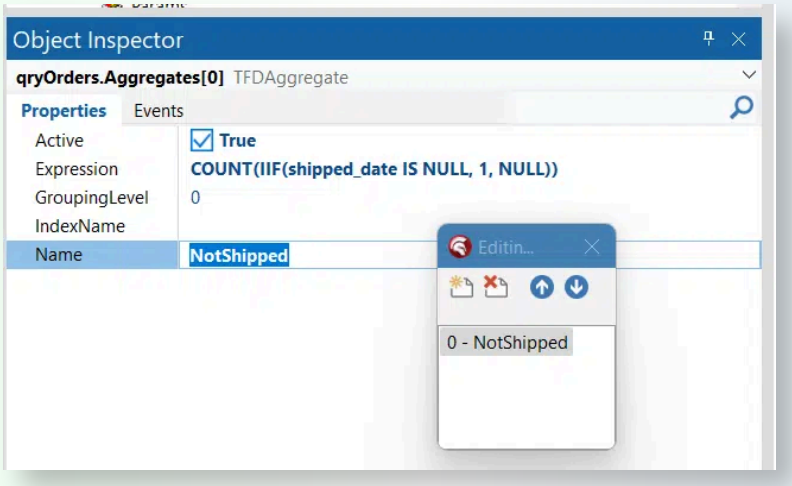
Como exemplo, tenho um programa de demonstração no qual recupero todos os pedidos de um banco de dados em uma tabela. Eu posso filtrá-los por id de loja. No canto superior direito, você pode ver o número de pedidos que ainda não foram enviados. Esse número é calculado por um campo agregado da consulta.



order_id	customer_id	order_status	order_date	required_date	shipped_date	store_id	staff_id
1	259	4	01/01/16	03/01/16	03/01/16	1	2
4	175	4	03/01/16	04/01/16	05/01/16	1	3
9	60	4	05/01/16	08/01/16	08/01/16	1	2
12	91	4	06/01/16	08/01/16	09/01/16	1	2
14	258	4	09/01/16	11/01/16	12/01/16	1	3

Para começar, defino um campo de agregação na propriedade de agregação da consulta. O campo mais importante aqui é **Expression**. É aqui que definimos a agregação que queremos executar. Possivelmente com uma condição. A sintaxe de expressão padrão do **Firedac** é usada para isso.

Continuação 1
Agregações de conjuntos de dados



No exemplo acima, eu conto o número de registros (**COUNT)** do campo **Shipped_Date** com a condição de que esse campo seja contado somente se o valor estiver vazio.
Para isso, a função **IIF** pode ser usada. Outros exemplos simples que você pode usar aqui são:

```
SUM(order_amount)

MIN(order_date)
```

Como você pode ver nas propriedades, é possível definir um campo como ativo.
Se você não fizer isso, ele obviamente não será calculado.
Um nome também pode ser útil para localizar a agregação no código.

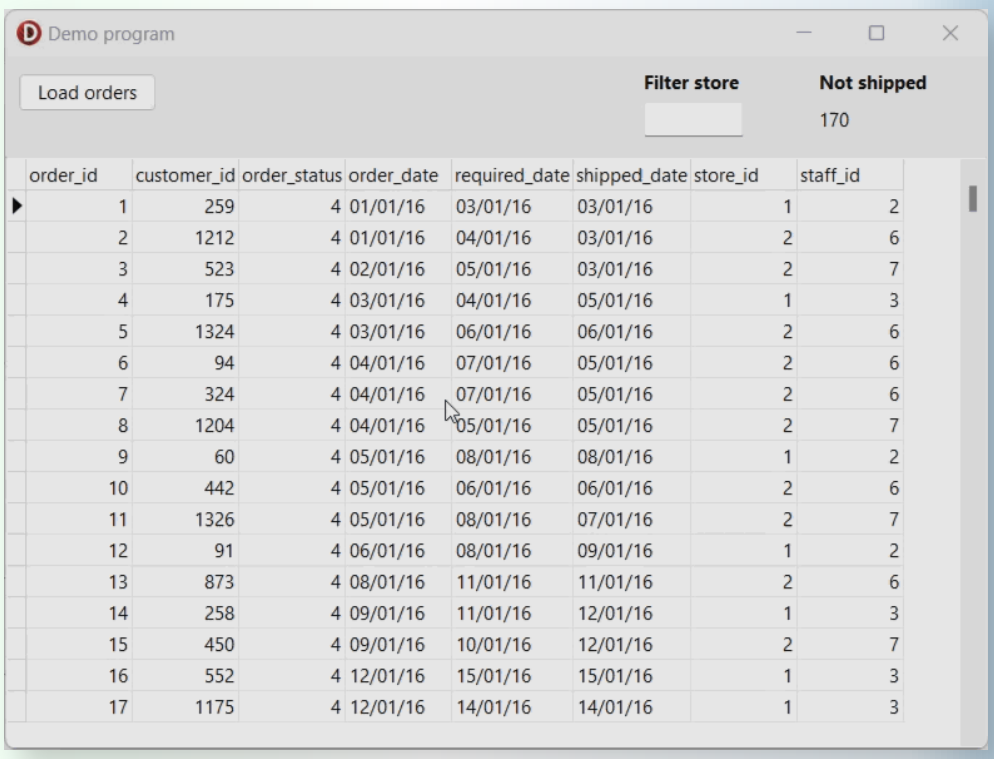
Infelizmente, não é possível vincular uma agregação como um campo de banco de dados a um controle com reconhecimento de dados.
Portanto, a renderização tem de ser tratada no código.
Criei um procedimento separado para isso, para que eu possa chamá-lo facilmente quando uma atualização for necessária.
Chamo esse procedimento no evento **AfterOpen** da consulta e à filtragem em um armazenamento.

```
procedure TfrmDemoApp.qryOrdersAfterOpen(DataSet: TDataSet);
begin
  ShowNotShippedAggregation;
end;

procedure TfrmDemoApp.ShowNotShippedAggregation;
begin
  var NotShipped := qryOrders.Aggregates.Items[0].Value;

  if NotShipped = Null then
    lblNotShipped.Caption := '0'
  else
    lblNotShipped.Caption := NotShipped;
end;
```

Continuação 2
Agregações de conjuntos de dados

Na consulta, lembre-se de definir a propriedade `AggegratesActive` como `True`, caso contrário as agregações não serão calculadas. Outro ponto a ser lembrado: as agregações operam nos dados recuperados no conjunto de dados.
Por padrão, o **Firedac** recupera 50 registros. Se você quiser um total que seja baseado em todos os registros, vá para **FetchOptions** nas propriedades da consulta e defina o **Mode** como `fmAll`.

Veja abaixo o resultado no aplicativo de demonstração.



order_id	customer_id	order_status	order_date	required_date	shipped_date	store_id	staff_id
1	259	4	01/01/16	03/01/16	03/01/16	1	2
2	1212	4	01/01/16	04/01/16	03/01/16	2	6
3	523	4	02/01/16	05/01/16	03/01/16	2	7
4	175	4	03/01/16	04/01/16	05/01/16	1	3
5	1324	4	03/01/16	06/01/16	06/01/16	2	6
6	94	4	04/01/16	07/01/16	05/01/16	2	6
7	324	4	04/01/16	07/01/16	05/01/16	2	6
8	1204	4	04/01/16	05/01/16	05/01/16	2	7
9	60	4	05/01/16	08/01/16	08/01/16	1	2
10	442	4	05/01/16	06/01/16	06/01/16	2	6
11	1326	4	05/01/16	08/01/16	07/01/16	2	7
12	91	4	06/01/16	08/01/16	09/01/16	1	2
13	873	4	08/01/16	11/01/16	11/01/16	2	6
14	258	4	09/01/16	11/01/16	12/01/16	1	3
15	450	4	09/01/16	10/01/16	12/01/16	2	7
16	552	4	12/01/16	15/01/16	15/01/16	1	3
17	1175	4	12/01/16	14/01/16	14/01/16	1	3

GDK software

embarcadero
Official Technology Partner

Nós somos a GDK.

Tem um desafio Delphi e está à procura de experiência ou capacidade? Os nossos especialistas e programadores estão prontos para concretizar as suas ambições e objectivos.

30 **Deselvovedores Delphi**
Trabalhar com o nossos Experts em Delphi

99+ **Conversões Delphi**
Atualização inteligente para o Delphi mais recente.

5 **Embarcadero MVPs**
Autoridades na comunidade Delphi

4 **Escritórios pelo mundo**
Países Baixos, Reino Unido, Brasil e EUA

GDK EM POUCAS PALAVRAS

Sobre a GDK.

Partilhamos uma paixão pelo desenvolvimento de software e gostamos de nos manter a par das últimas tecnologias. Temos uma forte equipa de especialistas com muitos conhecimentos e experiência em Delphi.

ALCANÇANDO SUAS AMBIÇÕES

Trabalhar em conjunto.

Procura um parceiro para manter e aumentar o seu software? Ou atualizar o seu software para a versão mais recente do Delphi?
Nós estamos prontos para o ajudar!

Entre em contato

www.gdksoftware.com

SCAN ME

Codolex
Simplify development

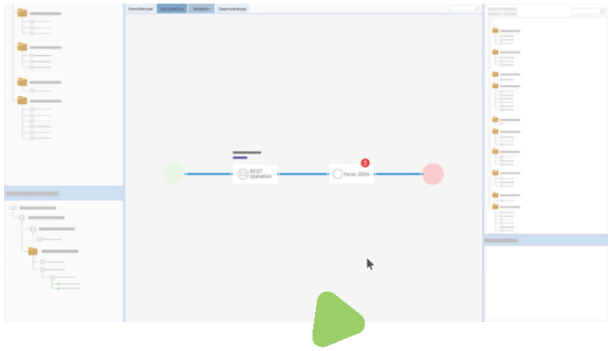
embarcadero
Official Technology Partner

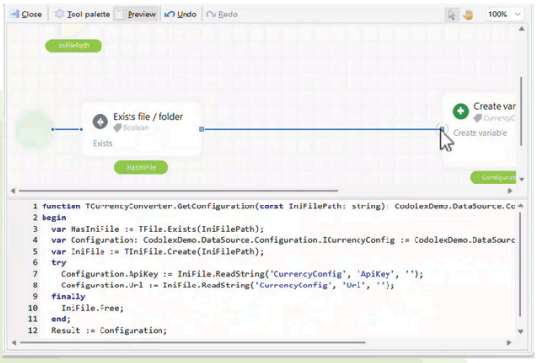
Revolucionário.

Codolex é uma solução de baixo código criada especificamente para o Delphi, que lhe permite desenvolver rapidamente, mantendo o controlo do código fonte.

Desenvolvimento visual

Desenvolva o seu código através do design visual. Compreenda a lógica mais rapidamente através da representação visual e ajuste-a facilmente modificando o fluxo. Uma imagem diz mais do que mil palavras!





Geração de código

A Codolex gera código que pode ser imediatamente utilizado nos seus projectos. Usando a função de pré-visualização no modelador, você pode ver imediatamente qual código é gerado para o fluxo. Não há dependência de fornecedor porque tudo é gerado em código.

A Codolex fornece tudo para simplificar o seu desenvolvimento

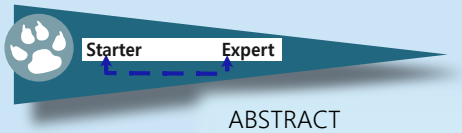
SCAN ME

www.codolex.com

Experimente

SELEÇÃO E REALCE DE TEXTO (HIGHLIGHTING) EM UM VISUALIZADOR DE PDF PAS2JS

POR MICHAEL VAN CANNEYT



ABSTRACT
Em artigos anteriores, apresentamos uma maneira de mostrar um PDF no navegador, e de pesquisar em um PDF. Neste artigo, adicionamos um recurso que estava faltando: destacar resultados de pesquisa no texto e seleção de texto.

❶ INTRODUÇÃO

Em uma série de artigos sobre o **PAS2JS**, demonstramos como mostrar um **PDF** no navegador, e como pesquisar um texto no PDF mostrado.
Isso pode ser feito com relativa facilidade usando o **PDF.js**, a biblioteca de exibição de PDFs em **Javascript da Mozilla.** Posteriormente, adicionamos o recurso de pesquisa em uma série de PDFs usando um mecanismo no lado do servidor e um indexador escrito em **Free Pascal**.
O resultado é a base para a **Blaise Pascal Magazine Library**, uma **biblioteca pesquisável de artigos publicados** na **Blaise Pascal Magazine**: um programa que funciona tanto off-line quanto on-line.
Esses programas de demonstração tinham duas desvantagens:
A primeira desvantagem era que o mecanismo de busca se **limitava a encontrar o texto e exibir a página correta no PDF.**
A segunda desvantagem era a impossibilidade de selecionar (e possivelmente copiar) o texto no PDF. Felizmente, a **API do PDF.js** contém algumas chamadas que resolvem esses dois problemas.
Neste artigo, mostraremos como usar uma dessas chamadas, **resolvendo assim os dois problemas de uma só vez**.

❷ ATUALIZANDO O PDF.JS

Antes de começar, é necessário fazer um pequeno desvio:
Entre a publicação dos vários programas que mostram o funcionamento da biblioteca PDF.js, a biblioteca foi alterada de uma forma que exige mudanças no código **Pascal** e no **HTML**.
Essas alterações não são extensas, mas são necessárias, caso contrário seu programa deixará de funcionar se já não tiver parado de funcionar, o que é provável se você tiver usado a distribuição on-line do **PDF.js** por meio de alguma **CDN (Content Delivery Network)**.
Anteriormente, a **biblioteca PDF.js** expunha uma variável global `pdfjsLib` que era definida nos programas como:

```
var
pdfjsLib : TPDFJSStatic; external name 'pdfjsLib';
```

A **biblioteca PDF.js** agora é criada como um módulo **Javascript**, semelhante a uma biblioteca carregável dinamicamente. A extensão do nome de arquivo da biblioteca também foi alterada para .mjs para refletir esse fato. Como resultado, a **variável acima não será mais definida se você carregá-la como um script normal**, porque o navegador se recusará a carregar o script.

A solução é carregar a biblioteca como um módulo:
Isso significa simplesmente que você deve adicionar um atributo de tipo com o valor module à tag de script que inclui PDF.js, por exemplo, da seguinte forma:

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/pdf.js/4.0.269/pdf.mjs"
type="module">
</script>
```

Em seguida, o script será carregado corretamente e a variável global será definida.
Se você estiver usando uma cópia privada da compilação herdada do **PDF.js**, não será necessário de alterar nada.

❸ UM MECANISMO DE SELEÇÃO E DESTAQUE

O motivo pelo qual as primeiras versões do visualizador de PDF não permitiam destaque e seleção é porque, basicamente, o **PDF é renderizado como um bitmap** em uma tela **HTML**, incluindo o texto. Deve ficar claro que o bitmap de texto desenhado não pode ser selecionado (*a menos que se use* **OCR** - *Reconhecimento Ótico de Caracteres*) e que é necessário algum outro mecanismo.
Felizmente, o **PDF.js** oferece uma solução: ele contém um mecanismo para renderizar os elementos de texto de um arquivo **PDF** como **HTML**, sobreposto na tela.
Esse **HTML** é estilizado de modo a ficar completamente transparente, e o usuário não o vê, mas é um HTML real que faz parte do **DOM** da página HTML e pode ser manipulado com as **técnicas usuais de DOM e CSS.**
Além disso, ao selecionar algo no navegador, o HTML "oculto" é levado em consideração: o **pseudoelemento** '**selected**' também funcionará e poderá ser estilizado:
Ao alterar a cor de fundo do pseudoelemento '**selected**', o texto selecionado pode se tornar visível. Isso significa que, se usarmos a API do PDF.js para renderizar o texto, teremos nosso mecanismo de seleção de texto.
Como o **HTML renderizado contém o texto real** (*mas simplesmente renderizado de forma invisível*), isso significa que quando pesquisamos um texto e é exibida uma página que contém uma correspondência do texto, **podemos percorrer o HTML criado e destacar as correspondências no texto**.
A seguir, mostraremos como fazer isso.

❹ PERMITINDO A SELEÇÃO: RENDERIZANDO O TEXTO DA PÁGINA COMO HTML.

A **API do PDF.js** tem duas chamadas que criam ou atualizam o **HTML DOM** com o texto do **PDF**. Ambas são implementadas usando uma tarefa em segundo plano e ambas retornam uma instância dessa tarefa em segundo plano:

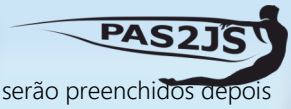
```
function renderTextLayer(params : TPDFJSRenderTextLayerParameters):TPDFTextLayerRenderTask;
function updateTextLayer(params : TPDFJSUpdateTextLayerParameters):TPDFTextLayerRenderTask;
```

A chamada que nos interessa é a tarefa RenderTextLayer.
Ela aceita o seguinte objeto de parâmetro:

```
TPDFJSRenderTextLayerParameters = class
   textContentSourceStream : TJSReadableStream; external name 'textContentSource';
   textContentSourceItems  : TTextContent; external name 'textContentSource';
   container               : TJSHTMLElement;
   viewport                : TPDFPageViewport;
   isOffscreenCanvasSupported : Boolean;
   textDivs                : TJSHTMLElementArray;
   textDivProperties       : TJSMap;
   textContentItemsStr     : TStringDynArray;
end;
```

Os primeiros 5 campos dessa classe são de entrada:

textContentSourceStream	O conteúdo de texto do PDF como um stream.
textContentSourceItemS	O conteúdo de texto do PDF, conforme retornado pela chamada chamada `getTextContent` do objeto `TPDFPageProxy`.
container	O elemento HTML no qual o texto será renderizado.
viewport	A viewport usando a qual o PDF foi renderizado
isOffscreenCanvasSupported	Defina como True se a camada do PDF puder usar uma tela fora da tela.(*necessário para determinar as larguras de texto*)



Continuação 

Os últimos 3 campos da classe são de saída: eles serão preenchidos depois que a chamada renderTextLayer tiver feito seu trabalho. Inicialmente, eles devem ser definidos como matrizes vazias.

`textDivs` - **Uma matriz com os elementos HTML gerados.**
`textDivProperties` - **Uma matriz com as propriedades necessárias para gerar os elementos HTML**.
`textContentItemsStr` - Os textos brutos dos elementos **HTML** gerados.

Demonstraremos essa chamada no **exemplo de pesquisa de PDF** que apresentamos anteriormente. Nesse exemplo, o usuário poderia inserir um **URL** ou selecionar um arquivo PDF que seria mostrado e que poderia ser pesquisado. Como lembrete, a figura 1 na página 3 do artigo mostra a aparência do aplicativo. Vamos reutilizar e expandir esse exemplo, e o código-fonte estará novamente disponível para você. Para usar a chamada `renderTextLaye`, precisamos **adicionar outros elementos** ao **HTML** do nosso visualizador de **PDF**. Anteriormente, tínhamos o seguinte **HTML** no qual o **PDF** era exibido:

```
<div class="is-flex is-justify-content-center">
  <canvas id="PDFCanvas" height="737" width="538"></canvas>
</div>
```

Agora precisamos de um elemento extra que será usado para sobrepor o texto (*com ID pdfTextLayer*), e outro para conter um parâmetro de estilo extra (com `ID pdfViewer`):

```
<div class="is-flex is-justify-content-center">
  <div id="pdfViewer" style="position: relative">
    <div class="canvaswrapper" >
      <canvas id="PDFCanvas" height="737" width="538"></canvas>
    <div>
    <div id="pdfTextlayer" class="textLayer">
    </div>
  </div>
</div>
```

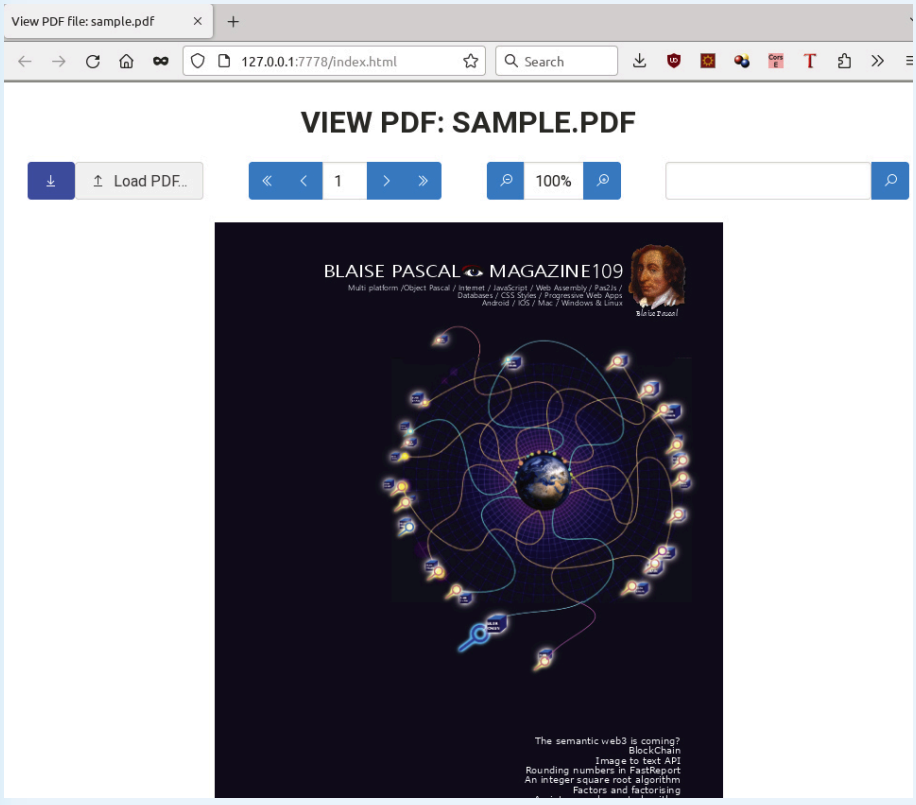


Figura 1: O visualizador de PDF em ação

Continuação 2

Os dois IDs serão vinculados a duas variáveis em nosso programa:

```
TMyApplication = class(TBrowserApplication)
  divpdfViewer,
  FTextlayer,
  lblFileLocation,
  lblZoom : TJSHTMLElement;
  // ...
end;
```

A **renderização de uma página** do documento **PDF** foi feita no método `RenderPage` do nosso programa: Nesse método, a chamada `GetPage` é usada para recuperar um objeto proxy para uma página do **PDF**, e o método render é usado para renderizar de fato a página na tela:
Em redes de computadores, um **servidor proxy** é um aplicativo de servidor que atua como intermediário entre um cliente que solicita um recurso e o servidor que fornece esse recurso.

```
procedure TMyApplication.renderPage(aNum: Integer);

function renderOK(aValue : JSValue) : JSValue;

Var
  N : Integer;
begin
  FPageRendering:=false;
  if (FPageNumPending <> -1) then
    begin
      N:=FPageNumPending;
      FpageNumPending:=-1;
      renderPage(N);
    end;
  Result:=True;
end;

function havePage (aValue : JSValue) : JSValue;

var
  page : TPDFPageProxy absolute aValue;
  viewport : TPDFPageViewport;
  renderContext: TPDFRenderParams;
  renderTask : TPDFRenderTask;
  viewportParams : TViewportParameters;

begin
  viewportParams:=TViewportParameters.new;
  viewportParams.scale:=FScale;
  viewport:=page.getViewport(viewportParams);
  Fcanvas.height := viewport.height;
  Fcanvas.width := viewport.width;
  renderContext:=TPDFRenderParams.New;
  renderContext.canvasContext:=Fctx;
  renderContext.viewport:=viewport;
  renderTask:=page.render(renderContext);
  renderTask.promise.&then(@renderOK);
  Result:=True;
end;

begin
  FpageRendering:=True;
  pdfDoc.getPage(aNum).&then(@HavePage);
  edtPageNo.Value:=IntToStr(anum);
end;
```

Como você pode ver no código da rotina `havePage`, os parâmetros para a tarefa de renderização da página **PDF** são os mesmos que os da chamada `RenderTextLayer`. Portanto, faz sentido colocá-los em um registro que é declarado em nossa classe de aplicativo: *(Consulte a próxima página)*

Continuação 3

```
TCurrentPageInfo = record
page : TPDFPageProxy;
viewport : TPDFPageViewport;
renderContext: TPDFRenderParams;
renderTask : TPDFRenderTask;
viewportParams : TViewportParameters;
end;
TMyApplication = class(TBrowserApplication)
divpdfViewer,
FTextlayer,
lblFileLocation,
lblZoom : TJSHTMLElement;
FCurrentPageInfo : TCurrentPageInfo;
// ...
end;
```

Agora podemos reescrever o procedimento `havePage` para que ele use o registro recém-introduzido para armazenar as informações necessárias para renderizar a página:

```
function havePage (aValue : JSValue) : JSValue;
var
page : TPDFPageProxy absolute aValue;
begin
FCurrentPageInfo.Page:=page;
FCurrentPageInfo.viewportParams:=TViewportParameters.new;
FCurrentPageInfo.viewportParams.scale:=FScale;
FCurrentPageInfo.viewport:=page.getViewport(FCurrentPageInfo.
viewportParams);
Fcanvas.height := FCurrentPageInfo.viewport.height;
Fcanvas.width := FCurrentPageInfo.viewport.width;
FCurrentPageInfo.renderContext:=TPDFRenderParams.New;
FCurrentPageInfo.renderContext.canvasContext:=Fctx;
FCurrentPageInfo.renderContext.viewport:=FCurrentPageInfo.viewport;
 FCurrentPageInfo.renderTask:=page.render(FCurrentPageInfo.
renderContext);
FCurrentPageInfo.renderTask.promise.&then(@renderOK);
Result:=True;
end;
```

Quando a renderização é concluída, o procedimento RenderOK é chamado, e agora podemos adicionar o código necessário para renderizar o texto. Começamos obtendo o texto real usando a chamada getTextContent. O leitor atento notará que a chamada getTextContent é a mesma chamada que foi usada para recuperar o texto do PDF para pesquisar no PDF.
A chamada renderOK se torna então:

```
function renderOK(aValue : JSValue) : JSValue;
Var
N : Integer;
begin
FPageRendering:=false;
if (FPageNumPending <> -1) then
begin
N:=FPageNumPending;
FpageNumPending:=-1;
renderPage(N);
end;
Result:=True;
FCurrentPageInfo.page.getTextContent().&Then(@RenderText);
divpdfViewer.style.setProperty('--scale-factor',FloatToStr(FScale))
end;
```

Continuação

O elemento **pdfViewer** obtém uma propriedade de estilo que define uma variável **CSS** de fator de escala: Essa variável é necessária para o CSS que é gerado pelo **PDF.js**.
O valor da variável é definido como a escala atual usada para desenhar o **PDF**.
Esquecer de defini-la (*ou defini-la com um valor errado*) resultará em um **HTML** que não está posicionado corretamente sobre a imagem do **PDF**.
Quando o texto é recuperado, a chamada de retorno `RenderText` que fornecemos ao objeto Javascript Promise retornado por getTextContent, será chamada.
A callback simplesmente prepara os argumentos para a chamada da **API RenderTextLayer,** usando o `FCurrentPageInfo` e o resultado da promessa:

```
Function TMyApplication.RenderText(aValue: JSValue) : JSValue;
Var
  aContent : TTextContent absolute aValue;
  aTextRender : TPDFJSRenderTextLayerParameters;
begin
  FTextlayer.InnerHTML  :='';
  aTextRender           :=TPDFJSRenderTextLayerParameters.New;
  aTextRender.container :=FTextlayer;
  aTextRender.isOffscreenCanvasSupported :=true;
  aTextRender.textContentSourceItems    :=aContent;
  aTextRender.viewPort  :=FCurrentPageInfo.viewport;
  pdfjsLib.renderTextLayer(aTextRender).promise.&then(@TextDone);
end;
```

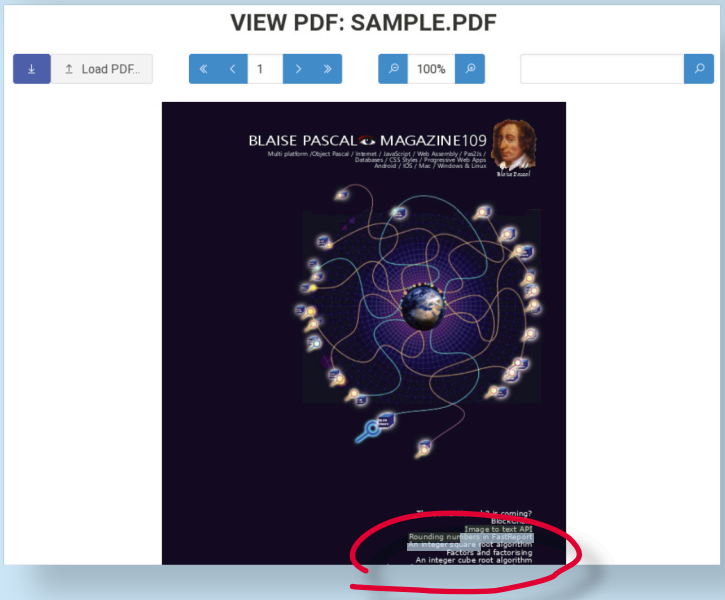


Figura 2: Seleção no visualizador de PDF em ação

Com o código acima em vigor, o recurso de selecionar e copiar texto está quase pronto: o **HTML necessário é gerado e posicionado no local correto** da página HTML. O que está faltando é um **CSS** que é exigido pelo HTML gerado. O leitor interessado pode encontrá-lo no arquivo viewer.css, que incluímos na página **HTML** com um elemento **HTML** de link simples:

```
<link rel="stylesheet" href="viewer.css">
```

O resultado de tudo isso pode ser visto na figura 2 da página 8: algum texto (em várias linhas) é selecionado na visão geral dos artigos.
Se a **escala do PDF for alterada**, ele será **renderizado novamente** e o texto também será **renderizado novamente:** Como a variável CSS de escala é definida antes da renderização, as duas estarão sempre sincronizadas.

❺ DESTAQUE DAS CORRESPONDÊNCIAS DE PESQUISA

O que ainda está faltando é o **destaque dos resultados da pesquisa** depois que **uma página é renderizada** como resultado de uma operação de pesquisa. Adicionar essa funcionalidade não é tão difícil:
No retorno de chamada `TextDone` para a chamada `renderTextLayer`, temos a oportunidade de fazer isso. Observando o HTML gerado pelo PDF.js, podemos ver que ele é simplesmente uma série plana de elementos span HTML com algum posicionamento aplicado: os elementos `span` contêm apenas texto.
Para destacar o texto, podemos **simplesmente fazer um loop sobre os elementos** de extensão e verificar se a extensão contém o texto. Basicamente, esse é o mesmo loop feito no algoritmo de pesquisa, a diferença é que agora fazemos um loop
sobre o HTML gerado em vez das estruturas retornadas pela chamada getTextContent.
O código a seguir verifica se uma pesquisa foi realizada. Caso contrário, ele é encerrado imediatamente. Se uma pesquisa foi realizada, **ele prepara a expressão regular usada para a pesquisa**, e percorre todas as tags de span, chamando `Highlight` para cada elemento de span:

```
Function TMyApplication.TextDone(aValue: JSValue) : JSValue;
var
  El : TJSELement;
  aRegex : TJSRegExp;
  aReg,aTerm : String;
begin
  aTerm:=edtSearch.Value;
  if aTerm='' then exit;
  aReg:=Format('%s',[aTerm]);
  aRegEx:=TJSRegExp.New(aReg,'gi');
  El:=FTextlayer.firstElementChild;
  While Assigned(El) do
    begin
      if (el is TJSHTMLElement) and SameText(El.tagName,'span') then
        HighLight(TJSHTMLElement(El),aRegex);
      El:=El.nextElementSibling;
    end;
end;
```

Na rotina de destaque (`highlite`), modificamos o HTML interno do elemento span. Se uma correspondência (ou múltiplas correspondências) for encontrada, cercaremos a correspondência com um novo intervalo
elemento ao qual damos uma **classe CSS de 'highlight':**
Por exemplo, se o texto de pesquisa for inteira, o seguinte elemento span:

```
<span>Um algoritmo de raiz quadrada inteira</span>
```

torna-se

```
<span>Um algoritmo de raiz quadrada <span class="highlight">inteira</var></span>
```

A classe '`Highlight`' é escolhida pelo **CSS para colorir o fundo do texto de forma transparente**, e o texto na tela abaixo aparecerá destacado.
Este mecanismo pode ser codificado da seguinte forma:
Na verdade, é **um loop simples usando a expressão regular**: contanto que a expressão regular encontre uma correspondência, o texto entre a correspondência e o final do a correspondência anterior é adicionada ao elemento span como está e, em seguida, o texto correspondente é adicionado incorporado em um novo elemento span. Se não houver mais correspondências, a rotina anexa o texto restante.

```
Procedure TMyApplication.HighLight(El : TJSHTMLElement; aRegex : TJSRegexp);
Var
   S,aText, aLeft : String;
   Matches : TStringDynArray;
   aLast : Integer;
   aSpan : TJSHTMLElement;

begin
   aText:=El.innerText;
   Matches:=aRegex.exec(aText);
   aLast:=1;
   // We exit at once if there is nothing to do.
   if not Assigned(Matches) then
     exit;
   // Clear the HTML
   EL.InnerHTML:='';

   While Assigned(Matches) do
     begin
       // Add preceding text
       S:=Copy(aText,aLast,aRegex.lastIndex-Length(Matches[0]));
       // Create span.
       El.AppendChild(document.createTextNode(S));
       aSpan:=TJSHTMLElement(document.createElement('span'));
       aSpan.InnerText:=Matches[0];
       aSpan.className:='highlight';
       El.AppendChild(aSpan);
       // Search again.
       aLast:=aRegex.LastIndex+1;
       Matches:=aRegex.exec(El.innerText);
     end;
     // Append last text.
     if aLast<length(aText) then
       begin
         S:=Copy(aText,aLast,Length(aText)-aRegex.lastIndex);
         El.AppendChild(document.createTextNode(S));
       end;
end;
```

Com isso, o destaque está completo. O resultado pode ser visto na figura 3 da página 11.

❻ CONCLUSÃO

A adição de uma camada de texto ao visualizador **PDF.js** permite implementar operações de seleção, copiar e colar fora da caixa.
Conforme mostrado neste artigo, ela também pode ser usada para implementar, de maneira bastante simples, **o destaque de termos** de pesquisa em uma página.
O mecanismo não é perfeito, pois o texto do PDF às vezes é dividido em diferentes elementos do PDF (*por exemplo, quando a formatação é alterada*):
Consequentemente, o **HTML** será dividido em **tags HTML**. Para situações comuns, o mecanismo é perfeitamente adequado.

Exemplo: veja a próxima página

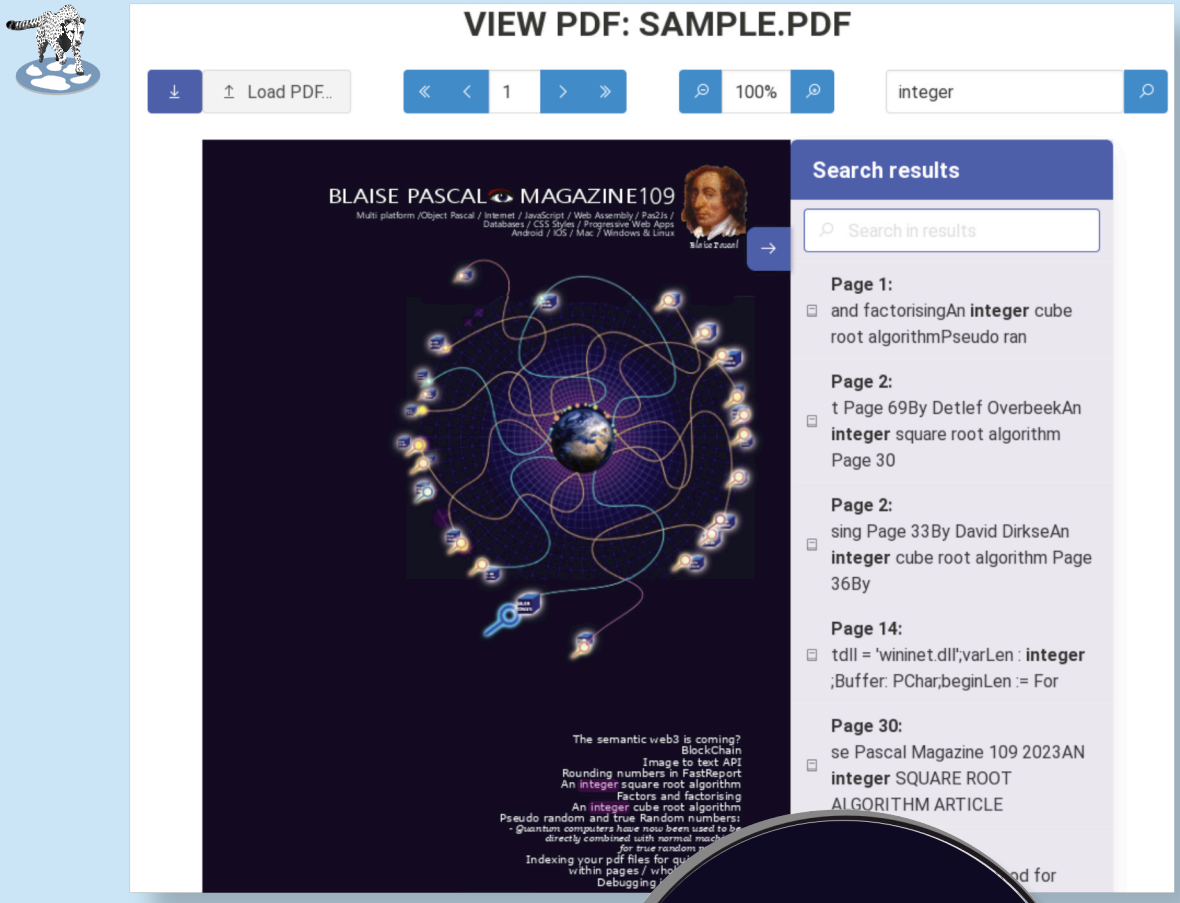


Figura 3: Realce do resultado no visualizador de PDF em ação Figura

Acesse http://www.components4developers.com
Para obter mais informações sobre a kbmMW

Components4Developers decidiu permitir que o mundo tenha a oportunidade de brincar com as centenas de milhares de linhas de código, contendo milhares de funções, procedimentos, métodos, classes e tipos, cobrindo o vasto número de recursos que fazem do **kbmMW** o líder dos ambientes de desenvolvimento n-tier.

Lançamos agora nossa terceira Community Edition, que contém quase todos os recursos da **kbmMW Enterprise Edition**, mas limitada em algumas áreas, em parte devido a limitações técnicas, em parte devido a algumas limitações artificiais. Além disso, ela contém o **kbmMemTable Community Edition**, que é uma correspondência direta do **kbmMemTable Standard Edition**.

O uso do **Community Edition é GRATUITO**, desde que os aplicativos em que ele é usado forneçam uma receita direta ou indireta que não chegue a US$ 5.000 e que a empresa que distribui/produz os aplicativos tenha um faturamento anual inferior a US$ 50.000.

Leia o arquivo license.txt para obter todos os detalhes.

A Community Edition só pode ser usada para a produção de aplicativos Windows de 32 bits, mas o usuário tem acesso a SmartServices, REST, WIB, kbmMemTable, SQL, ORM, SmartEvent, SmartBinding, Scheduler, Logging, Configuração, Cifras, transportes, XML, JSON, JSON5, YAML, BSON, MessagePack, i18n e inúmeros outros recursos sobre os quais o senhor já leu aqui no blog!

A Community Edition também pode ser usada como uma avaliação de 60 dias do kbmMW Enterprise Edition.

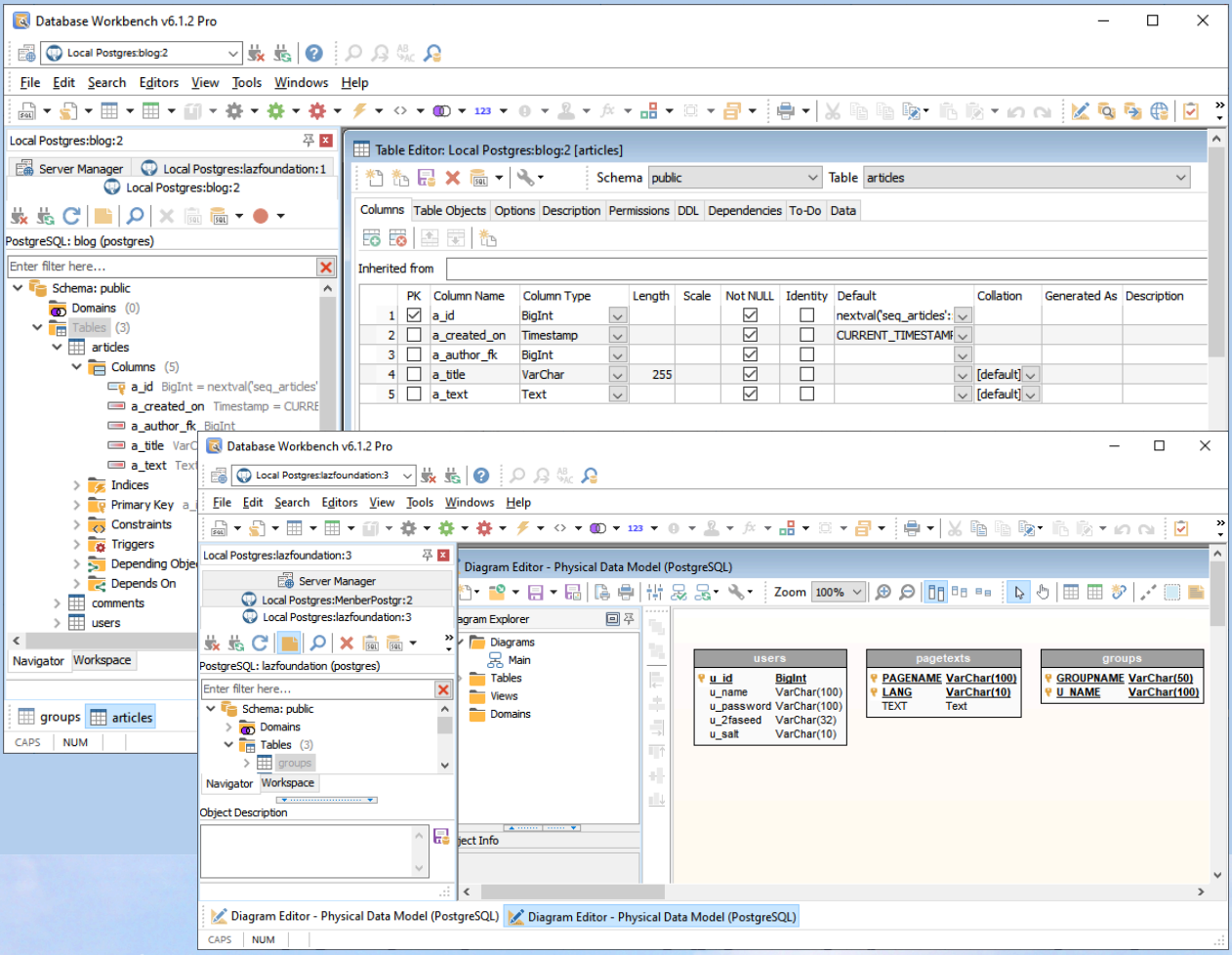
Se quiser ir além, adquira o **kbmMW Enterprise Edition**, que realmente tem a melhor relação custo-benefício e, embora seja o framework n-tier mais antigo existente, em sua maior parte compatível com versões anteriores, para **Delphi**, é também o único que o leva à vanguarda da tecnologia atual.

A **Community Edition** não contém código-fonte e só funcionará com a versão do **Delphi** para a qual foi lançada.
A versão atual corresponde à **versão mais recente do Delphi Sydney 12.0.0, incluindo a Community Edition.**

Na próxima edição do Blaise, apresentaremos uma série de artigos de demonstração curtos que mostrarão como trabalhar com a community edition

NOVO!!! kbmMW Community Edition V.5.23.00 lançado para o Delphi 12.0.0 Athens!

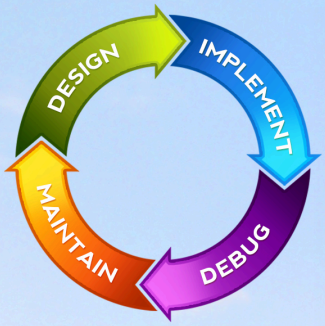
COMPONENTS 4 DEVELOPERS



Introducing

Database Workbench 6

database development environment



Esta versão oferece suporte completo ao SQLite

Interface de usuário consistente, editores de código modernos, habilitados para Unicode, compatível com HighDPI, designer de ER, engenharia reversa, navegação de metadados, editores de objetos visuais, migração de metadados, comparação de metadados, depuração de rotina armazenada, visualizador de plano SQL, gerador de dados de teste, impressão de metadados, importação e exportação de dados, bombeamento de dados, Grant Manager, tarefas de DBA, snippets de código, SQL Insight, VCS integrado, editor de relatórios, pesquisa de metadados de banco de dados, várias ferramentas de produtividade e muito mais...

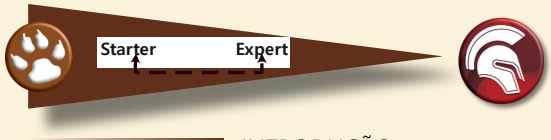
Para SQL Server, Oracle, MySQL, MariaDB, Firebird, InterBase, NexusDB and PostgreSQL

Upscene *Database tools for developers*

www.upscene.com

INSTALAÇÃO DAS VERSÕES MAIS RECENTES DO FAST REPORTS NO LINUX - LAZARUS.

POR ALKEXANDER REDKOW



INTRODUÇÃO

Na edição 110-111 da "Blaise Pascal Magazine", havia um anúncio sobre uma atualização do software **Fast Repor**t e um artigo **"FASTREPORT FOR LAZARUS - LINUX"** de Sergey Plastun.
Sou um usuário ávido desse software desde a versão "0", passei por todas as versões e mantenho quase mil formulários de relatório. Recentemente, tive uma experiência interessante com a última iteração do **FR**, o **FastreportVCL** Pro 2023.2.

O anúncio dizia: "Um sistema de instalação com autorização on-line - instale e atualize todos os seus produtos de uma só vez".

INSTALAÇÃO

Bem, isso parece bom.
Porém no trabalho não tenho conexão com a Internet (*e acho que não sou o único*).
Portanto, uma instalação on-line em si não é para mim. Mesmo assim, fiz uma tentativa em casa.
Fiz o download do arquivo "**setup.exe**" e o iniciei. Após a **"autorização on-line"**, vi um formulário vazio e não sabia o que fazer em seguida. Então, entrei em contato com o suporte técnico (que, a propósito, é ótimo e é sempre um prazer lidar com ele - rápido, exato e muito humano) e recebi uma resposta.

Para prosseguir, preciso ter o **Delphi ou o Lazarus instalado**, mas eu tenho o Lazarus - como instalação portátil pelo utilitário **fpcupdeluxe**. Para mim, isso é imprescindível, pois estamos migrando do Windows para o Linux e os executáveis de todos os aplicativos que estou mantendo têm de ser de três tipos:
Win32, Win64 e Linux. Então, novamente para o suporte técnico.
Seguindo o conselho deles, instalei o Lazarus a partir de uma distribuição (oportunidade útil para dar uma olhada no novo 3.0RC1) e a configuração foi adiante (levou quase 2 horas, talvez minha conexão com a Internet estivesse baixa).
No final, eu tinha um monte de pastas na pasta **Components do Lazarus:**

```
2023.3
    Sources
        LibLazarus 3.0RC1 (elazarus)
            FPC
              Win64
            Sources
              FastCore
              FastGraphics
              Localization
              FastReport
              FastScript
```

O restante foi muito fácil, como aconselhado no artigo mencionado acima.
A pasta "Win64" contém todos os arquivos `.lpk` de que precisamos.
Copiei essas pastas para a instalação do **fpcupdeluxe**, compilei e instalei os pacotes - e funcionou!

Mas isso funcionou no **Windows**.
E quanto ao **Linux**?
Perguntei ao suporte técnico sobre isso e eles disseram que enviariam um pacote deb (e rpm) para o **Linux**. E eles foram fiéis à sua palavra (*versão 2023.3*).

Agora ele é enviado junto com o instalador on-line.
Talvez, como nas versões anteriores (*sem instalação on-line*), ele realmente não seja necessário se você já tiver os pacotes de instalação de uma instalação do **Windows**.
Basta copiá-los para o **Linux** com a instalação do **Lazarus** e executar a rotina de compilação/instalação, mas agora eu tenho esse pacote, então fiz o download e experimentei.
Tudo correu bem e rápido. Instalei o pacote
`fast_report-professional-2023.3.0.deb` (*usamos o Linux do tipo debian*) com o apt e obtive estas pastas:

```
/usr/share/FastReport-Professional
    Core
    Demos
    FastReport
    FastScript
    Graphics
    Localization
    Lpks
        Libs
        Res
        fr_lazarus.lpk
        frxchartlazarus.lpk
        frxe_lazarus.lpk
        frxlazdbf.lpk
        frxlazsqlite.lpk
        frxPDFlazarus.lpk
        frxrichlazarus.lpk
        fs_ibx.lpk
        fs_lazarus.lpk
```

Como vemos, todos os arquivos `.lpk` que precisamos compilar e instalar estão na pasta **`/usr/share/FastReportProfessional/Lpks`**.

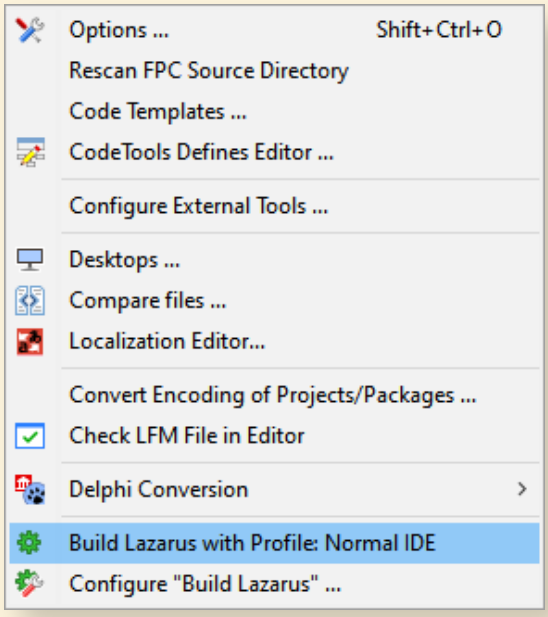
O **Lazarus** em minha conta sem privilégios se recusou a compilar pacotes devido à falta de privilégios de gravação na pasta de instalação.
Eu poderia copiar a pasta de instalação para a pasta **Lazarus/Components** como sempre fiz e depois corrigir a propriedade e os privilégios.

Mas eu tenho algumas instalações do Lazarus para fins de teste.
Então, em vez disso, iniciei o **Lazarus** com sudo, depois de compilar e instalar os pacotes **FR**, iniciei o **Lazarus** com minha conta sem privilégios e tudo correu bem.

Quanto ao artigo, tenho alguns comentários.

- A edição Professional tem componentes cliente/servidor" está sendo sugerida, isso não é verdade, apenas a Enterprise e a Ultimate têm componentes cliente/servidor.
- Clicar em "Usar" - não é suficiente, você precisa clicar em "Instalar" também.
  A instalação do `frxrichlazarus.lpk` requer a instalação prévia do pacote richmemo.
  Depois de instalar cada pacote, o **Lazarus** será reinicializado - o **Lazarus** não precisa apenas reinicializar, ele reinicia depois de se reconstruir e isso leva um bom tempo.
  Portanto, depois de instalar cada pacote (na verdade, depois de marcar cada pacote para instalação) O **Lazarus** exige: reconstruir **agora ou não**?

  Se você responder "Not", a reconstrução só poderá ser feita uma vez depois que o último pacote for concluído



E uma última observação sobre o "**Online Installer**".
Acho que seria útil ter uma opção para adicionar manualmente uma instalação do **Lazarus** não registrada no sistema para casos como **fpcupdeluxe** ou **Code Typhon Studio.**

E agora vamos nos divertir um pouco.

Você o compra, você o instala e ele funciona

Certo dia, assisti à apresentação de uma nova versão do **FR**.
No final, houve uma espécie de leilão.
Os participantes deveriam elogiar o **FR**, o diretor do **FR** julgava a expressão e dava (*ou não*) um prêmio.
As recompensas incluíam canetas, pen drives e camisetas, todas com o logotipo da **FR**.
As camisetas eram consideradas de alto preço e eram dadas apenas de vez em quando. Quando houve uma espécie de calmaria, eu disse preguiçosamente (*sem me preocupar em ficar de pé*):
"**Não sei por que alguém deveria elogiar a FastReport. É uma coisa tão sem graça...**".

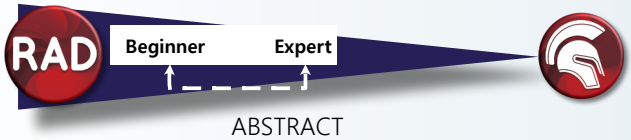
O diretor ficou quase apoplético e disse: **"Mas por quê?!"**.
Eu expliquei: "Se você pegar quase todos os outros softwares e tentar fazê-los funcionar - é uma luta, um trabalho árduo.
E quando, depois de noites sem dormir, litros de café,
maços de cigarros e conversas intermináveis, você faz um software funcionar
- é uma vitória, uma felicidade!

E agora o **FR.**
Você o compra, instala e ele funciona. **Muito chato**".
O Diretor quase ficou apoplético novamente, mas encontrou forças e disse:
"**Uma camiseta para o usuário!**".
Agora, de alguma forma, eu cresci demais para essa camiseta, mas ainda posso mostrá-la.

O QUE HÁ DE NOVO NO RAD STUDIO?

DELPHI 12 /. O RAD STUDIO 12 FOI LANÇADO

POR DETLEF OVERBEEK



ABSTRACT

Este artigo tem o objetivo de dar a você uma visão dos mais novos recursos do Delphi. Mostrarei os principais itens e algumas palavras sobre a integração com o Skia

Versões de plataforma compatíveis com o Rad studio 12.

Ele oferece suporte oficial para **iOS 17** (*somente para Delphi*), **Android 14** e **macOS Sonoma**, e também oferece suporte ao **Ubuntu 22 LTS** e ao **Windows Server** 2022.

Literais de strings de várias linhas

Para código-fonte Delphi. Os literais de string multilinha permitem a incorporação mais fácil de SQL, HTML, JSON, XML texto de várias linhas em um código-fonte de aplicativo.

O Object Pascal do Delphi permite literais de string e strings estáticas incorporadas no código. Historicamente, elas eram limitadas a "strings curtas" com um limite de 255 caracteres. Com o Delphi 12, isso muda. Literais de strings longas: O Delphi 12 agora suporta 4K caracteres por linha. Strings de várias linhas: O Delphi 12 apresenta strings de várias linhas, delimitadas por uma aspa tripla ("'). Essa sintaxe

facilita muito o uso de strings de várias linhas, sem a necessidade de usar o sinal '+'.

```
Begin
  Var MultilineString :=
  '''
     Long String Literals: Delphi 12 now supports 4K characters per line.
     Multiline Strings: Delphi 12 introduces multiline strings, enclosed by a triple quote (''').
     This syntax makes it really easy to use multiline strings, without having to use the '+' sign.
  '''
end.
```

Suporte SKIA para design de interface do usuário no **FireMonkey**
Melhora o desempenho e a qualidade da renderização de gráficos e controles de **IU** em todas as plataformas de destino.
O **FireMonkey** sempre usou estilos para a renderização da interface do usuário.
Esses estilos determinam a aparência e a funcionalidade dos elementos da interface do usuário em várias plataformas, incluindo **DirectX** e Metal.

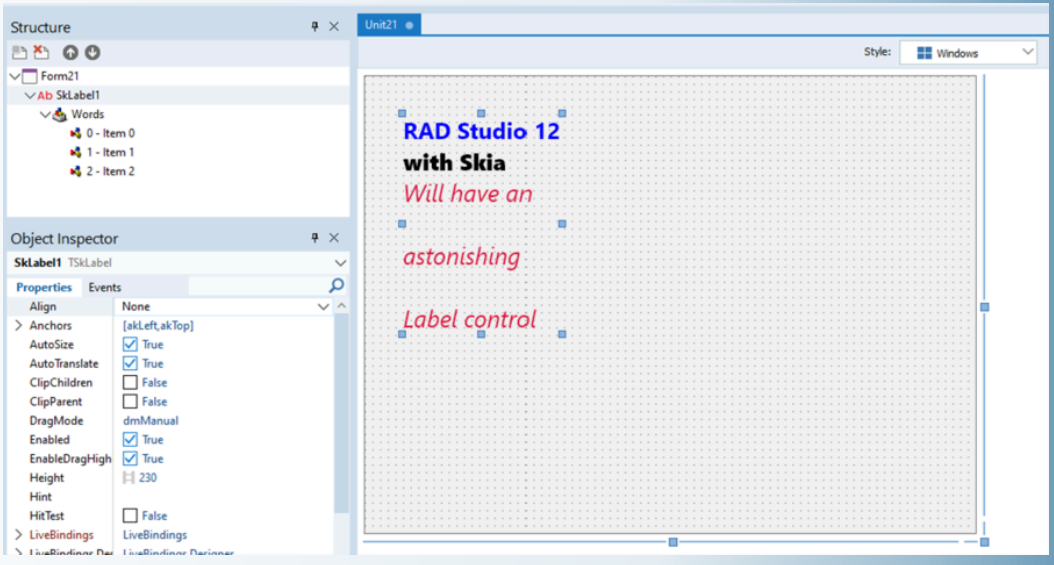
A própria **Skia** é reconhecida por sua capacidade em aplicativos gráficos 2D.
A biblioteca, desenvolvida pelo **Google**, melhora muito o desempenho.
Os usuários da biblioteca Skia4Delphi podem estar familiarizados com seus recursos.

O **RAD Studio** faz interface com o **Skia** aproveitando o projeto de código aberto Skia4Delphi, mas inclui recursos adicionais não encontrados nos projetos de código aberto, como o driver **Vulkan**.
Ele fornece uma **API 2D** abrangente para renderizar imagens em modelos móveis, de servidor e de desktop.
É compatível com todas as estruturas (**Console**, **FMX** e **VCL**) e plataformas do **RAD Studio.**

Fornece **APIs** 2D comuns, abstraindo as complexidades na implementação de bibliotecas de baixo nível usadas como **OpenGL**, **Vulkan*** (*veja a próxima página*), **DirectX** ou **Metal**, implementando otimizações e novos recursos. Consulte a documentação do **Skia** para conhecer todos os principais recursos e saber como ativar a integração do **Skia**.

Controles de **IU** baseados no **Skia**
A integração da biblioteca **Skia** também oferece alguns novos controles e componentes nativos específicos. Esses controles só estarão disponíveis se o **Skia** estiver ativado e implantado na plataforma de destino.
`TSkAnimatedImage`: o controle que carrega e renderiza imagens animadas, inclusive animações vetoriais. **TSkLabel**: implementa novos recursos que não eram compatíveis com o `TLabel` ou eram difíceis de implementar, como:

- Famílias de fontes; (lista de **fallback** de fontes como no **CSS**)
- Peso e inclinação da fonte;
- Suporte a vários estilos no texto;
- Suporte para **BiDi** (*da direita para a esquerda*);
- Suporte para justificar o alinhamento horizontal;
- Suporte à cor de fundo em partes do texto;
- Opção de tamanho automático;
- Decorações avançadas (*sublinhado ondulado, sobrelinhado, linha tracejada e outras*).



*** Backend Vulkan**

O **Skia** com **RAD Studio** oferece suporte ao backend Vulkan para Android e, opcionalmente, para Windows. O **Skia Canvas** com **RAD Studio** utiliza automaticamente o **Vulkan no Android**, se suportado, resultando em desempenho gráfico aprimorado e eficiência energética em comparação com o **OpenGLES**. Para ativar o **Vulkan** no **Windows** quando suportado,
defina o booleano `FMX.Types.GlobalUseVulkan` como `True` e o booleano `FMX.Skia.GlobalUseSkiaRasterWhenAvailable` como `False` na seção de inicialização.

```
uses
 System.StartUpCopy,
 FMX.Forms,
 FMX.Types,
 FMX.Skia,
 Unit1 in 'Unit1.pas' {Form1};

{$R *.res}

begin
 GlobalUseSkia := True;
 GlobalUseSkiaRasterWhenAvailable := False;
 GlobalUseVulkan := True;
 Application.Initialize;
 ...
```

OBSERVAÇÃO:
a preferência pelo backend Vulkan no Android pode ser desativada usando o mesmo booleano **GlobalUseVulkan** empregado para ativá-lo no Windows, mas deve ser definido como False para desativar.

`TSkPaintBox:`
Controla para pintar com as **APIs** do **Skia** diretamente na tela com o evento `OnDraw`.
`TSaAnimatedPaintBox:`
Permite definir a duração de uma animação e desenhar o progresso dessa animação usando **APIs do Skia** por meio do evento `OnAnimationDraw`.
`TSkSVG:` controle para exibir **SVG**.

```
procedure TForm1.SkPaintBox1Draw(ASender: TObject; const ACanvas: ISkCanvas;
                                 const ADest: TRectF; const AOpacity: Single);
begin
  var LPaint: ISkPaint := TSkPaint.Create;
  LPaint.Shader := TSkShader.MakeGradientSweep(ADest.CenterPoint,
                                [$FFFCE68D, $FFF7CAA5, $FF2EBBC1, $FFFCE68D]);
  ACanvas.DrawPaint(LPaint);
end;
```

DEMONSTRAÇÕES DO SKIA
As seguintes demonstrações do Skia estão incluídas na instalação.

Skia Demo FMX - RAD Studio
`https://docwiki.embarcadero.com/RADStudio/Athens/en/Skia_Demo_FMX`

Localização
Você pode encontrar o projeto de amostra do Skia Demo FMX em:

Iniciar > Programas > Embarcadero RAD Studio 12 > Amostras e navegue até:
- `Object Pascal\Multi-Device Samples\Skia4Delphi`
- `PP\Multi-Device Samples\Skia4Delphi`

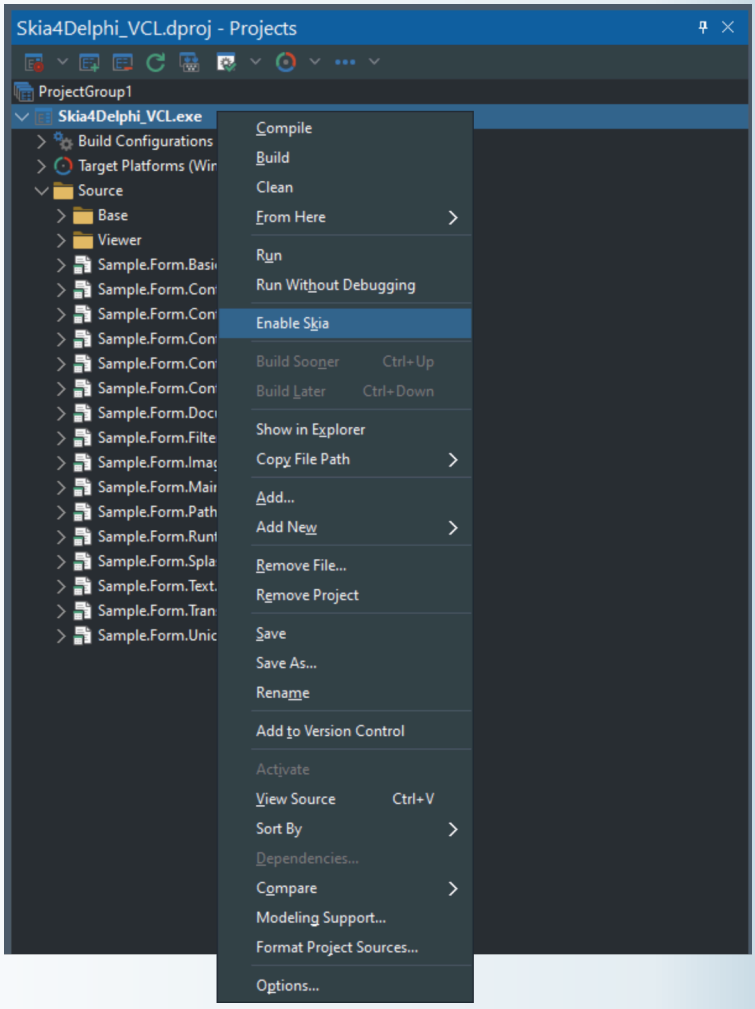
Skia4Delphi - RAD Studio

→ Go Up to Third Party Software Add-Ins: (See the standard path below)

`C:\Users\Public\Documents\Embarcadero\Studio\23.0\Samples\Object Pascal\VCL\Skia4Delphi`

Although the library is compatible with all frameworks, some features are unique:

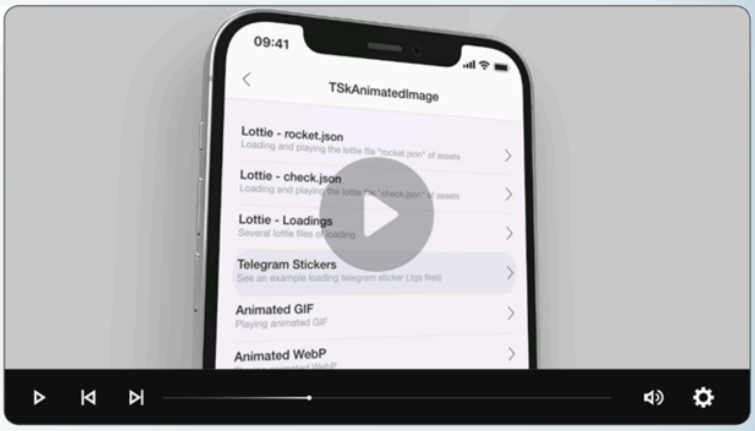
Feature set	Console	VCL	FMX	Description
Skia API	✓	✓	✓	Acesso à biblioteca **Skia** pura, por meio de uma única unidade: `System.Skia.pas` (*ou System.Skia.hpp*)
Controls		✓	✓	`TSkAnimatedImage`, `TSkLabel`, `TSkPaintBox` and `TSkSvg`
App render			✓	Substituição do mecanismo gráfico FMX pelo Skia



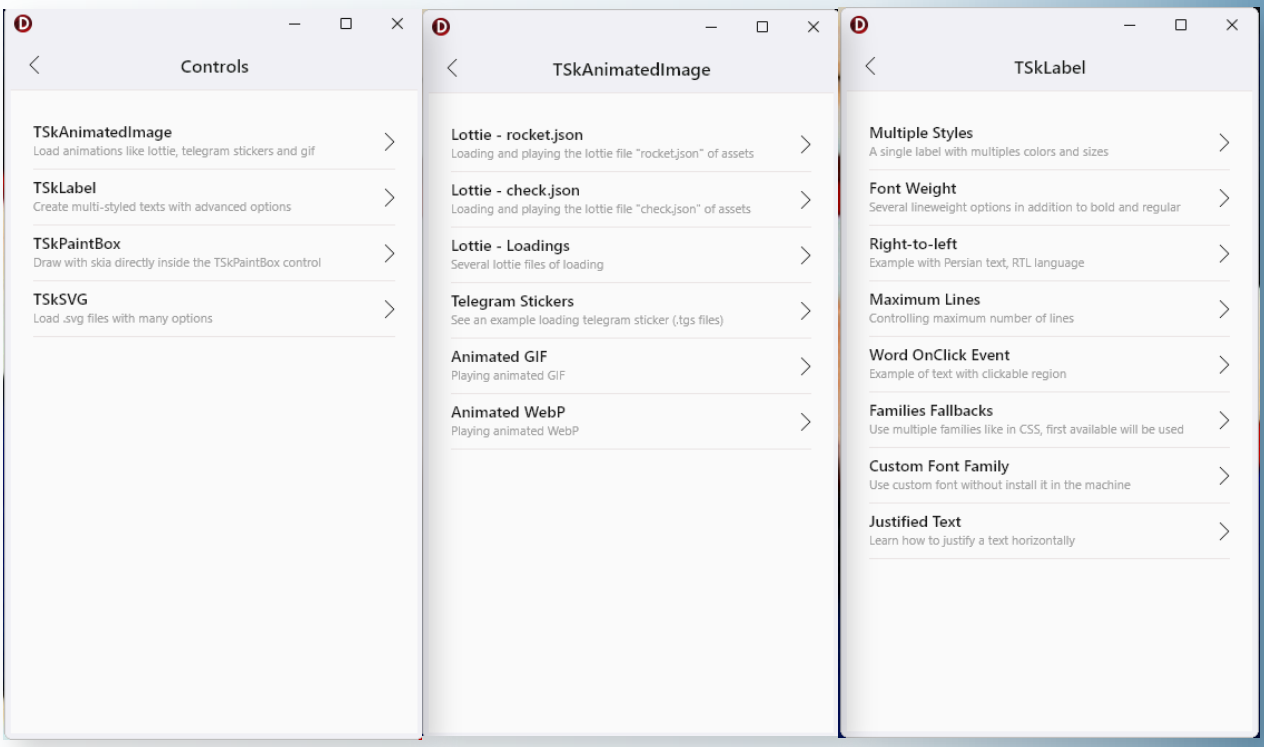
Habilitação do Skia

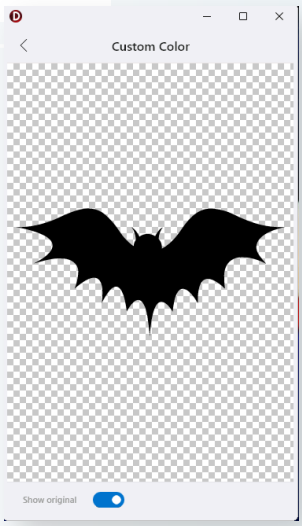
Antes de usar qualquer recurso ou unidade do **Skia**, **você deve habilitar seu projeto para usar o Skia.** Para fazer isso, abra seu projeto no seu projeto no **RAD Studio**, clique com o botão direito do mouse no projeto e clique em **Enable Skia:**

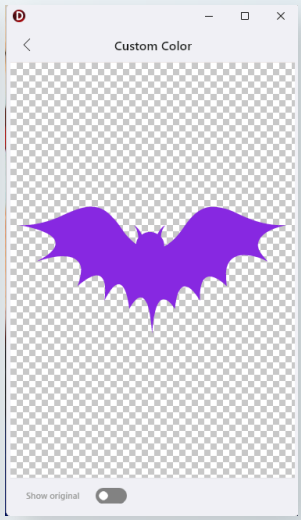
Essa ativação é feita apenas para aplicativos. Os pacotes que usam o **Skia** não precisam fazer essa ativação. Essa etapa configura automaticamente os links e as implantações dos binários da biblioteca no projeto. Se você não habilitar o **Skia** e não usar suas unidades, seu aplicativo terá problemas na inicialização.

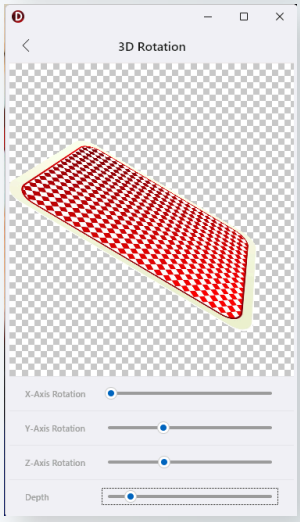


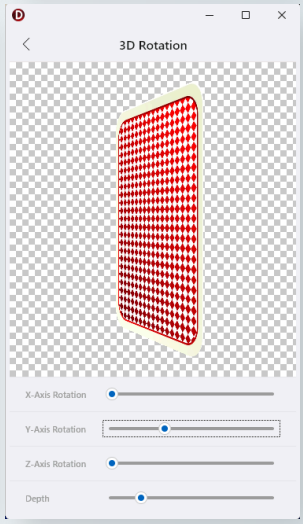
O aplicativo padrão para mostrar os exemplos

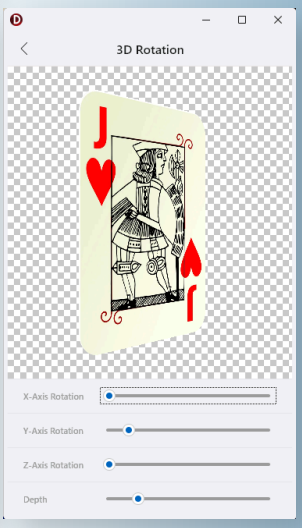


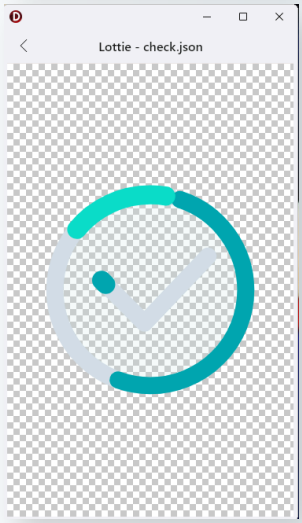








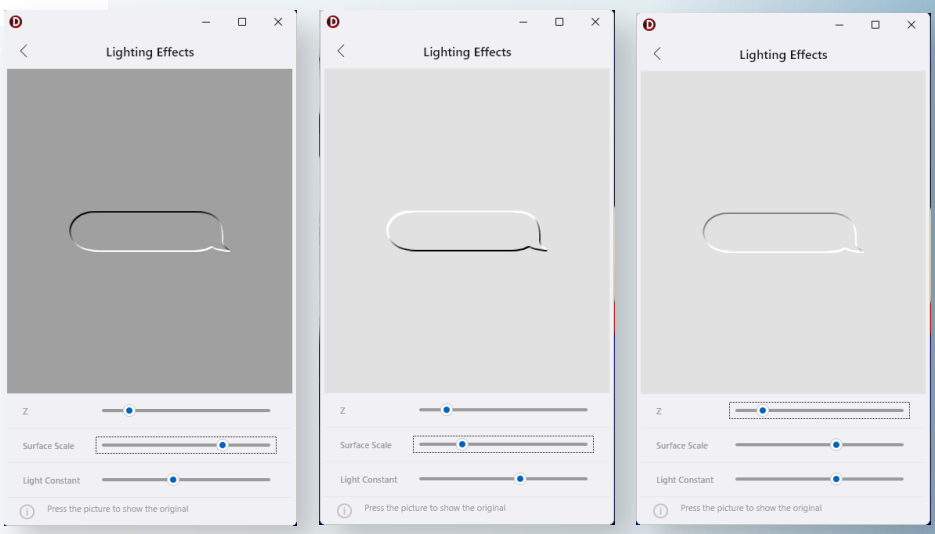


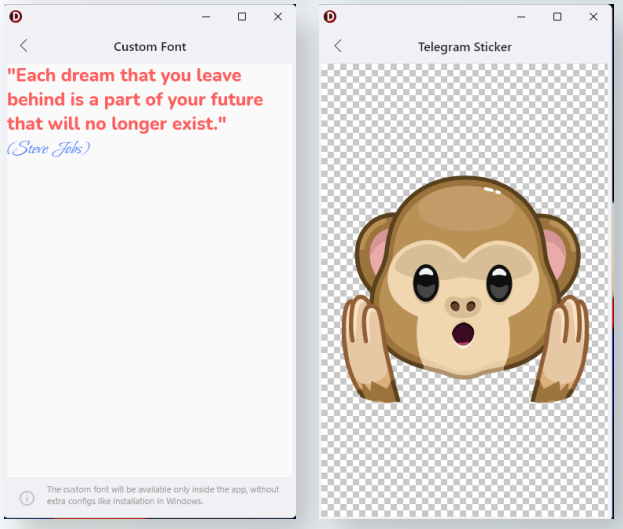


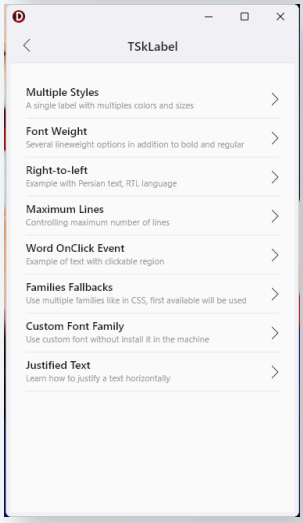
Você pode imaginar qualquer aprimoramento adicional da interface do usuário para seu aplicativo.
Na próxima edição, criarei alguns exemplos extras para que você tenha algum exemplo de código livre para usar.

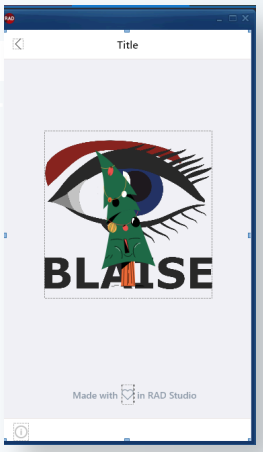

RAD Studio 12
embarcadero


SKIA

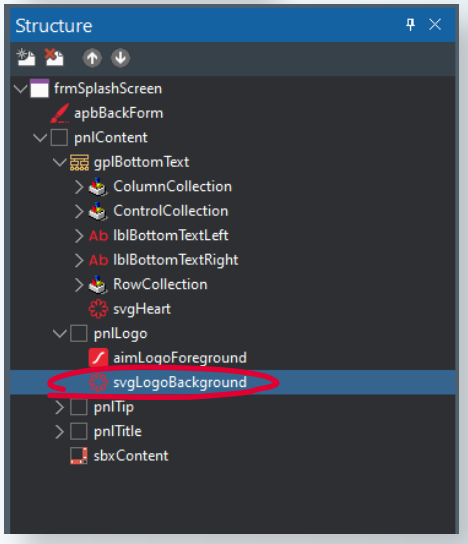

Lighting Effects
Z
Surface Scale
Light Constant
Press the picture to show the original
Lighting Effects
Z
Surface Scale
Light Constant
Press the picture to show the original
Lighting Effects
Z
Surface Scale
Light Constant
Press the picture to show the original

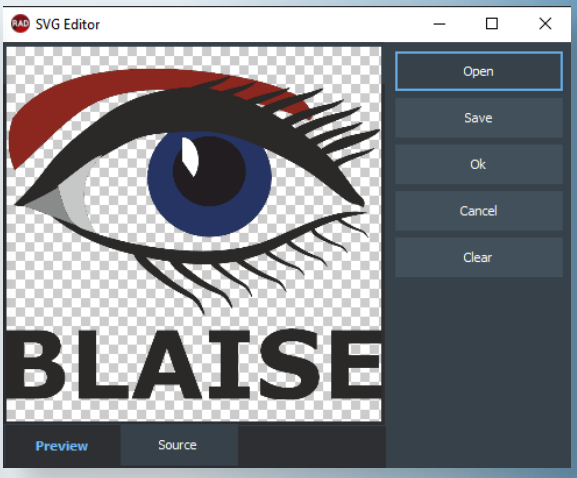

Custom Font
"Each dream that you leave behind is a part of your future that will no longer exist."
(Steve Jobs)
The custom font will be available only inside the app, without extra configs like installation in Windows.
Telegram Sticker


TSkLabel
Multiple Styles
A single label with multiples colors and sizes
Font Weight
Several lineweight options in addition to bold and regular
Right-to-left
Example with Persian text, RTL language
Maximum Lines
Controlling maximum number of lines
Word OnClick Event
Example of text with clickable region
Families Fallbacks
Use multiple families like in CSS, first available will be used
Custom Font Family
Use custom font without install it in the machine
Justified Text
Learn how to justify a text horizontally

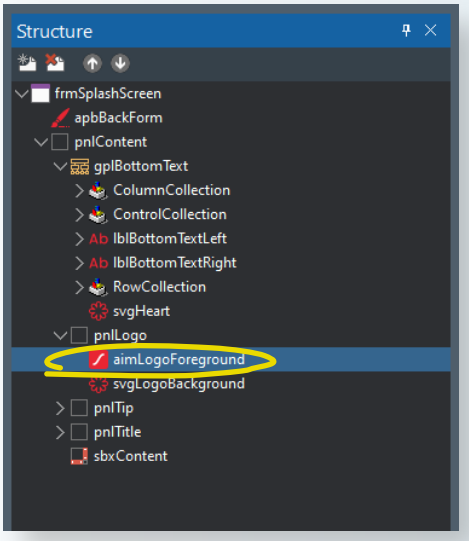


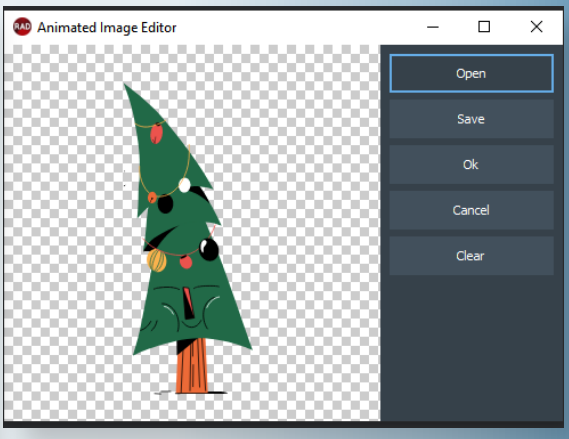
Para que você tenha uma visão rápida do que está acontecendo dentro do projeto, pode ser interessante fazer algumas alterações e usá-las. Sem escrever um único trecho de código, criei minha própria tela inicial e criei uma saudação de Natal.
Na verdade, essa tela inicial consiste em dois itens principais: um plano de fundo e um primeiro plano em movimento.
Para criar o `.svg`, tive que fazer alguns truques:
Criei um bitmap a partir do desenho vetorial do olho e, em seguida transformei-o em um `.svg` novamente. Caso contrário, o **SVG** não seria transformado com todas as camadas e cores. Também alterei o movimento de primeiro plano com um exemplo de **lottie** existente.
Coloquei o exemplo do código nas próximas duas páginas para que você tenha uma impressão da dificuldade de lidar com isso.





Background SVG





You can download your example from your personal downloads address.

SKIA

```
{******************************************************************************}
{                                                                              }
{                       Skia4Delphi                                            }
{                                                                              }
{ Copyright (c) 2021-2023 Skia4Delphi Project.                                 }
{                                                                              }
{ Use of this source code is governed by the MIT license that can be found in  }
{ the LICENSE file.                                                            }
{                                                                              }
{******************************************************************************}
unit Sample.Form.SplashScreen;

interface

{$SCOPEDENUMS ON}

uses
  { Delphi }
  Winapi.Windows, Winapi.Messages, System.Classes, System.Types, Vcl.Forms,
  Vcl.Graphics, Vcl.Controls, Vcl.ExtCtrls,

  { Skia }
  System.Skia, Vcl.Skia,

  { Sample }
  Sample.Form.Base;

type
  TfrmSplashScreen = class(TfrmBase)
    gplBottomText: TGridPanel;
    lblBottomTextLeft: TSkLabel;
    lblBottomTextRight: TSkLabel;
    svgHeart: TSkSvg;
    svgLogoBackground: TSkSvg;
    aimLogoForeground: TSkAnimatedImage;
    pnlLogo: TPanel;
    apbBackForm: TSkAnimatedPaintBox;
    procedure aimLogoForegroundAnimationFinish(Sender: TObject);
    procedure apbBackFormAnimationDraw(ASender: TObject; const ACanvas: ISkCanvas; const ADest: TRectF;
      const AProgress: Double; const AOpacity: Single);
    procedure pnlContentAlignPosition(Sender: TWinControl; Control: TControl;
      var NewLeft, NewTop, NewWidth, NewHeight: Integer; var AlignRect: TRect; AlignInfo: TAlignInfo);
  private
    FBackFormImage: ISkImage;
    FFrontFormImage: ISkImage;
  protected
    class function FormBackgroundColor: TColor; override;
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

procedure TfrmSplashScreen.aimLogoForegroundAnimationFinish(Sender: TObject);

  procedure PaintControl(const AControl: TWinControl; const ACanvas: TCanvas; const AOffset: TPoint);

  begin
    SaveDC(ACanvas.Handle);
    try
      SetWindowOrgEx(ACanvas.Handle, -(AControl.Left + AOffset.X), -(AControl.Top + AOffset.Y), nil);
      AControl.Perform(WM_PRINT, ACanvas.Handle, PRF_CHILDREN or PRF_CLIENT or PRF_ERASEBKGND);
    finally
      RestoreDC(ACanvas.Handle, -1);
    end;
  end;
```

SKIA

```
  function MakeScreenshot(const AForm: TfrmBase): ISkImage;
  var
    LBitmap: TBitmap;
  begin
    LBitmap := TBitmap.Create;
    try
      LBitmap.SetSize(AForm.pnlContent.Width, AForm.pnlContent.Height);
      LBitmap.Canvas.Lock;
      try
        PaintControl(AForm.pnlContent, LBitmap.Canvas, Point(0, 0));
      finally
        LBitmap.Canvas.Unlock;
      end;
      Result := LBitmap.ToSkImage;
    finally
      LBitmap.Free;
    end;
  end;

begin
  FFrontFormImage := MakeScreenshot(Self);
  FBackFormImage := MakeScreenshot(Application.MainForm as TfrmBase);
  BeginUpdate;
  try
    apbBackForm.BoundsRect := Rect(0, 0, pnlContent.Width, pnlContent.Height);
    apbBackForm.Align := alClient;
    apbBackForm.Parent := pnlContent.Parent;
    pnlContent.Visible := False;
    apbBackForm.Visible := True;
  finally
    EndUpdate;
  end;
end;

procedure TfrmSplashScreen.apbBackFormAnimationDraw(ASender: TObject;
  const ACanvas: ISkCanvas; const ADest: TRectF; const AProgress: Double;
  const AOpacity: Single);
begin
  ACanvas.Save;
  try
    ACanvas.Scale(1 / TSkAnimatedPaintBox(ASender).ScaleFactor, 1 / TSkAnimatedPaintBox(ASender).
                  ScaleFactor);
    ACanvas.DrawImage(FBackFormImage, 0, 0);
    ACanvas.SaveLayerAlpha(Round(255 * (1 - AProgress)));
    try
      ACanvas.DrawImage(FFrontFormImage, 0, 0);
    finally
      ACanvas.Restore;
    end;
  finally
    ACanvas.Restore;
  end;
end;

class function TfrmSplashScreen.FormBackgroundColor: TColor;
begin
  Result := FormBorderColor;
end;

procedure TfrmSplashScreen.pnlContentAlignPosition(Sender: TWinControl;
  Control: TControl; var NewLeft, NewTop, NewWidth, NewHeight: Integer;
  var AlignRect: TRect; AlignInfo: TAlignInfo);
begin
  inherited;
  if Control = pnlLogo then
  begin
    NewLeft := AlignRect.Left + ((AlignRect.Width - Control.Width) div 2);
    NewTop := AlignRect.Top + ((AlignRect.Height - Control.Height) div 2);
  end;
end;

end.
```

RAD Studio 12

Como usamos o Skia?

"O Skia4Delphi é uma API gráfica 2D multiplataforma para Delphi e C++ Builder baseada na Skia Graphics Library do Google. Fornece APIs 2D comuns, abstraindo a complexidade da implementação das bibliotecas de baixo nível
bibliotecas de baixo nível subjacentes, como OpenGL, Vulkan, DirectX, Metal e outras.

RAD Studio 12

Principais características do Skia (1/2)

Funções	Detalhes
Formas de desenho 2D, SVG	caminhos e textos Renderização e criação de SVG
Decodificadores de imagem	**BMP, GIF, ICO, JPG, PNG, WBMP, WEBP** e imagens brutas
Codificadores de imagem	PNG, JPG e WEBP
Player de animação	**Lottie, Telegram Sticker, GIF**s animados e **WEBP** animado
Anti-aliasing	Alta qualidade de caracteres, sem bordas irregulares
Fonte	Peso da fonte, famílias, tailbacks e fontes personalizadas (*sem instalação*), ligadura

RAD Studio 12

Principais características do Skia (2/2)

Funções	Detalhes
Texto	Vários estilos, número máximo de linhas, espaçamento entre linhas, justificação, contorno do texto, gradiente de cor e ornamentos
Renderização do	idioma da direita para a esquerda de textos em persa, árabe, hebraico etc.
PDF	Geração de PDF vetorizado
Unicode	Analisador de grafemas Filtros Filtros de cor, máscara e imagem
Clippings	Suporte para muitas operações avançadas de clipping como caminhos e shaders
Gradientes	(*gradientes de cor*) Gradientes de cor lineares, radiais, curvos e cônicos
Shader	Criação de shaders para a execução de determinados desenhos diretamente no processador gráfico, por meio de uma única linguagem de shader (**SkSL**)

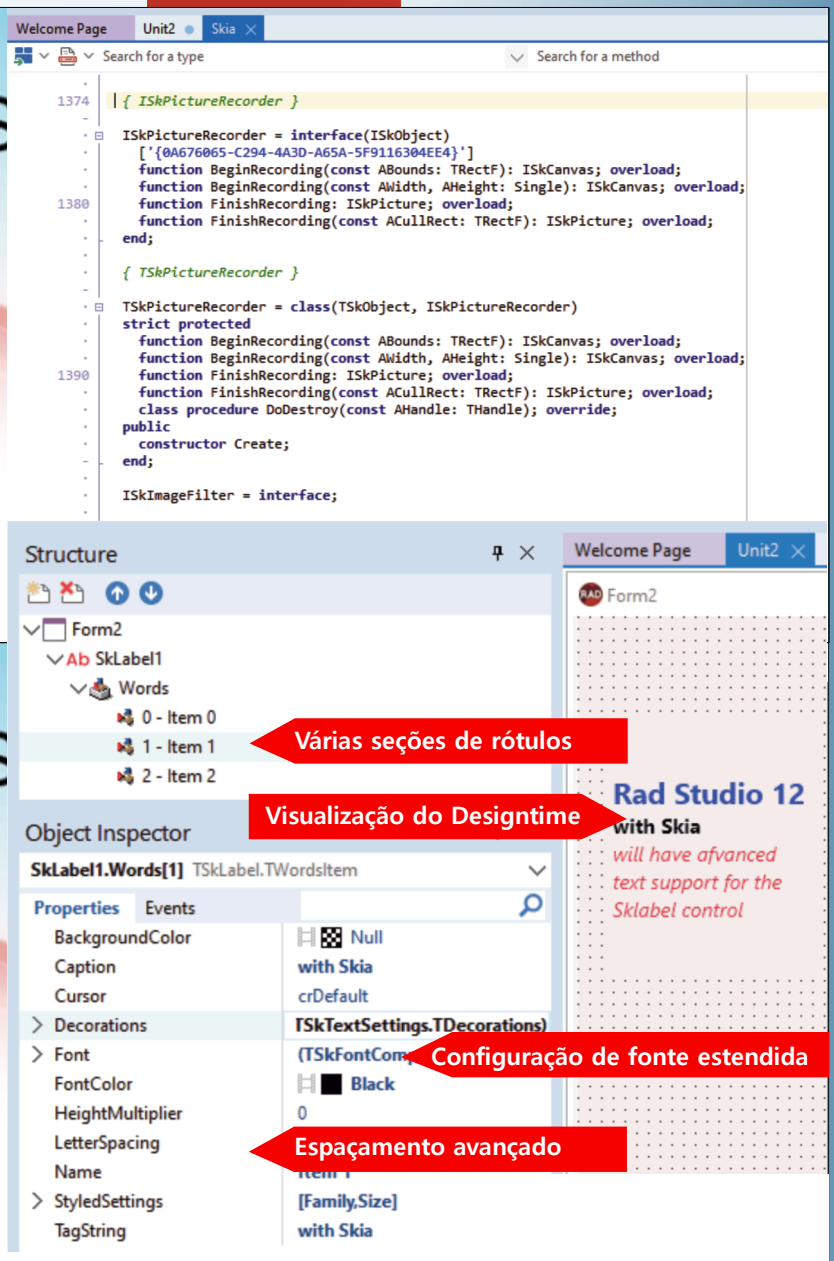
RAD RAD S

Skia no RAD Studio

Diferentes
Níveis de utilização diferentes
para Delphi e C++Builder

1. skiaAPI

Acesso à biblioteca pura do
biblioteca Skia, por meio de uma
única unidade única:
Skia.pas ou Skia.hpp



RAD RAD S

Skia no RAD Studio

Diferentes níveis de uso
para Delphi e C++Builder

2. Controles da IU

(para FMX e VCL)

- TSkAnimatedImage
- TSklabel
- TSkPaintBox
- TSkSvg

Skia no RAD Studio

Diferentes níveis de uso para Delphi e C++Builder

3. Renderização de aplicativos e estilos

Substituição do mecanismo gráfico FMX pelo
Skia. Melhor desempenho, anti-aliasing aprimorado
`FMX.Skia.GlobalUseSkia := True`
Desenho aprimorado com antialiasing no Skia (*à direita*)

Verbessertes Zeichnen mit Antialiasing

RAD Studio 12

Skia no RAD Studio

Diferentes níveis de uso para Delphi e C++Builder

4. Codecs para controle de imagem

Os novos codecs de imagem compatíveis com o Skia são registrados nas estruturas. Os controladores de imagem FMX ou VCL podem carregar e salvar diretamente novos formatos de imagem, como o WebP.

RAD Studio 12

Novos controles de interface do usuário (FMX/VCL) baseados no Skia

- **`TSkAnimatedImage`**
  - Suporta arquivo Lottie, adesivo do Telegram, GIF animado, WebP animado
- **`TSkLabel`**
  - Tamanho da fonte, inclinação da fonte, vários estilos no texto, BiDi (da direita para a esquerda)
  - Alinhar o alinhamento horizontal e muito mais
- **`TSkPaintBox & TSkAnimatedPaintBox`**
  - Pintura com Skia APIs diretamente na tela com o evento OnDraw
- **`TSkSVG`**
  - Basta exibir o SVG

RAD Studio 12

Skia e FireMonkey
Por que o Skia?

- Skia é uma biblioteca gráfica multiplataforma 20
- O Skia oferece renderização de alto desempenho e alta qualidade
- O Skia suporta operações gráficas avançadas em todas as plataformas
- O Skia oferece suporte a vários formatos de imagem e vídeo
- O Skia se torna a nova base para a renderização do FireMonkey

SKIA

RAD Studio 12

O que há de novo no FireMonkey? Muitas, na verdade!

- Suporte da biblioteca **Skia Graphic Library**
- Para todas as plataformas de destino, tanto **Delphi** quanto C++Builder
- Conclusão da reformulação do **`TEdit/TMemo`** com estilo
- Suporte à **API 33 do Android** (*veja mais adiante*)
- Novo designer de ícones e de tela inicial (*abordado anteriormente*)
- Exibições divididas para plataformas móveis
- Mais recursos e qualidade menores

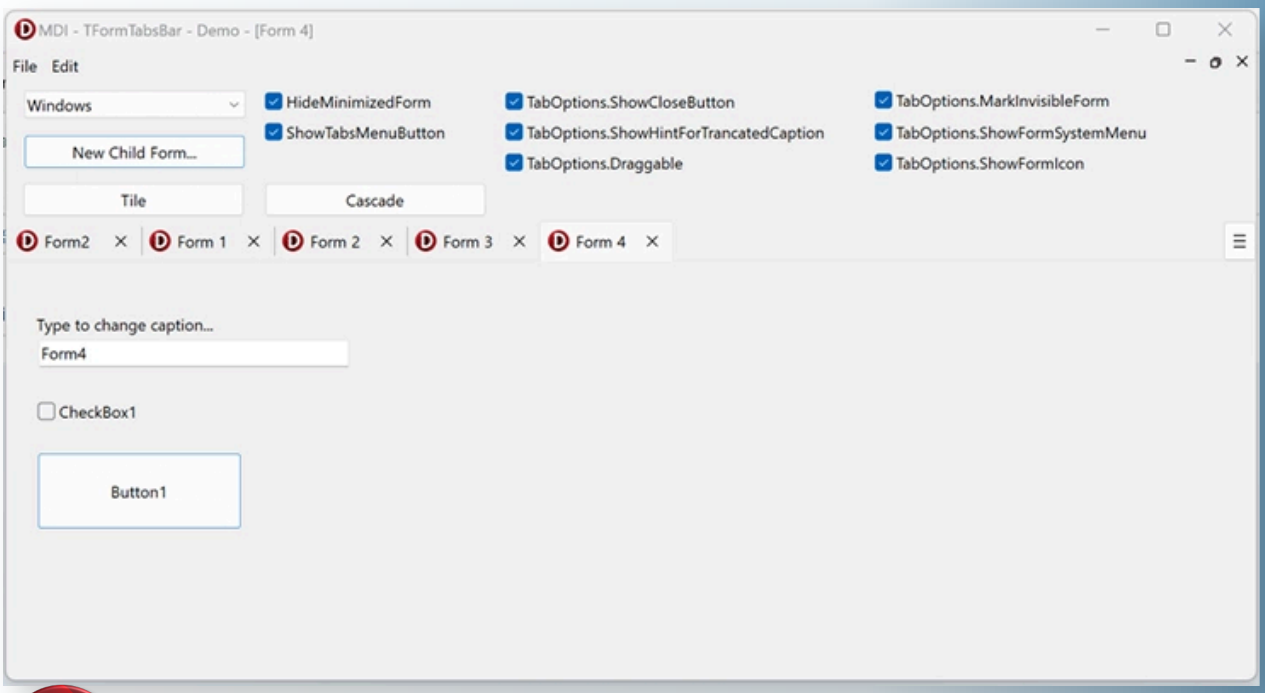
RAD Studio 12

Suporte ao FireMonkey Skia para todas as plataformas

- Para C++Builder e também para **Delphi**, **FireMonkey**, mas também **VCL**
- Uso do projeto de código aberto **Skia4Delphi**
- Inclusão de funções adicionais que não estão incluídas no projeto de código aberto:
- Suporte a Vulkan
  - Melhor desempenho gráfico e eficiência energética no Android em comparação com o **OpenGLES**
- **Skia Shading Language (SKSL)** para efeitos e filtros
- Codificador **WebP**
- Impressora nativa para Windows e impressão em PDF para todas as plataformas
- Também usado para o novo ícone e o assistente de splash

Design móvel aprimorado com melhorias no **FireMonkey**
Aprimoramentos no suporte à plataforma Android, painéis de tela dividida para **iOS** e **Android**, ícone de conjunto completo e assistente de tela inicial, suporte para o nível 33 da API do **Android**

VCL modernizada com **MDI** reformulada e interface de usuário com guias para **VCL**
Modernização aprimorada de aplicativos com suporte para **HighDPI** e novos designers de **VCL** originados dos controles **VCL** de assinatura da **Konopka**

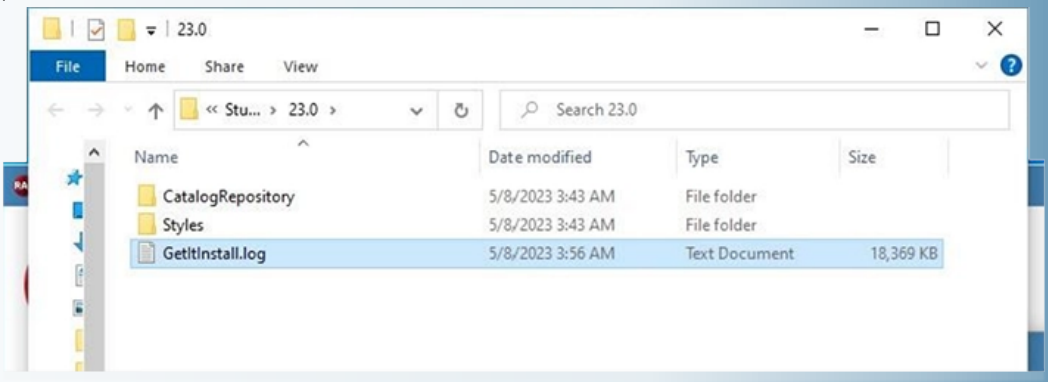


Mais APIs do **Windows** prontas para uso em **Object Pascal**
Conjunto abrangente de todos os cabeçalhos de API do Windows convertidos para **Object Pascal** para facilitar aos desenvolvedores **Delphi** a chamada de qualquer API da plataforma Windows


RAD Studio 12
File Edit Search View Refactor Project Run Component Tools Tabs Help
Welcome Page
Structure Object Inspector
Welcome to RAD Studio
Create Delphi or C++
Windows VCL Application - Delphi
A UI app using Windows native controls, using the Windows-native VCL framework
Multi-Device Application - Delphi
A UI app that works on any OS or device, using the FireMonkey framework
Package - Delphi
A smart dynamically loaded library, carrying extra metadata
Console Application - Delphi
A command-line app
Dynamic Library - Delphi
A library that can be loaded and unloaded
DUnitX Project - Delphi
Create New DUnitX Test Project
VCL Form - Delphi
VCL Frame - Delphi
A set of UI controls in their own container, which can be reused by placing them on another form or frame
Data Module - Delphi
A container for database interaction components on any platform
FireMonkey Frame - Delphi
A set of UI controls in their own container, which can be reused by placing them on another form or frame
Multi-Device Form - Delphi
A UI window that works on any OS or device
Unit - Delphi
A Delphi source code file
Open Recent
Skia4Delphi_VCL.dproj
D:\SPP\Blaise\Blaise_UK_114_115_2023\Authors\Detlef\Delphi\Skia\Project5Splash\
Project1.dproj
D:\SPP\Blaise\Blaise_UK_114_115_2023\Authors\David Dirkse\RollingCircles\circlesproject\
Project1.dproj
D:\SPP\Blaise\Blaise_UK_114_115_2023\Authors\David Dirkse\stepbystep-1\
ShaderButton_VCL.dproj
D:\SPP\Blaise\Blaise_UK_114_115_2023\Authors\Detlef\Delphi\Skia\
Detlef.dproj
D:\D12Athens\
New in GetIt
IPWorks 2020 Delphi Edition
IPWorks is a comprehensive framework of easy-to-use components for Internet development, including FTP, HTTP, SMTP, POP, IMAP, LDAP, DNS, SMS, SOAP, etc.
IPWorks 2020 C++ Builder Edition
IPWorks is a comprehensive framework of easy-to-use components for Internet development, including FTP, HTTP, SMTP, POP, IMAP, LDAP, DNS, SMS, SOAP, etc.
Login Screens
This FireMonkey UI template includes three different designs for implementing a login screen in a multi-device application.
Trial - IPWorks IPC 2022 Delphi Edition
Components for adding interprocess communication (IPC) over Named Pipes and seamless external process execution capabilities to your applications through simple, easy-to-use, and intuitive APIs.
Trial - IPWorks IPC 2022 C++ Builder Edition
Components for adding interprocess communication (IPC) over Named Pipes and seamless external process execution capabilities to your applications through simple, easy-to-use, and intuitive APIs.
Trial - IPWorks ZIP 2022 Delphi Edition
Components offering fast and secure advanced file and streaming compression, compliant with compression standards such as Zip, Jar, Tar, and GZip through simple, easy-to-use, and intuitive APIs.
Trial - IPWorks ZIP 2022 C++ Builder Edition
Components offering fast and secure advanced file and streaming compression, compliant with compression standards such as Zip, Jar, Tar, and GZip through simple, easy-to-use, and intuitive APIs.
Trial - IPWorks SSH 2022 Delphi Edition
Components offering fast and robust implementations of Secure Shell (SSH) secure communication protocols, enabling strong encryption and advanced cryptography through simple, easy-to-use, and intuitive APIs.
Trial - IPWorks SSH 2022 C++ Builder Edition
Components offering fast and robust implementations of Secure Shell (SSH) secure communication protocols, enabling strong encryption and advanced cryptography through simple, easy-to-use, and intuitive APIs.
Trial - IPWorks SNMP 2022 Delphi Edition
Components for building secure SNMP Network Management applications, including full support for advanced SNMPv3 security and efficient MIB management, through simple, easy-to-use, and intuitive APIs.
Promoted in GetIt
Bottom Navigation Template
The Bottom Navigation template is designed to embrace Material Design user interface and user experience philosophy. It looks great on both iOS and Android, and on Windows for testing.
VCL Style - Windows10ClearDay
VCL Windows Style - Windows10 Clear Day. Modern VCL with Windows10 UI. Updated for HighDPI.
Edit Layout

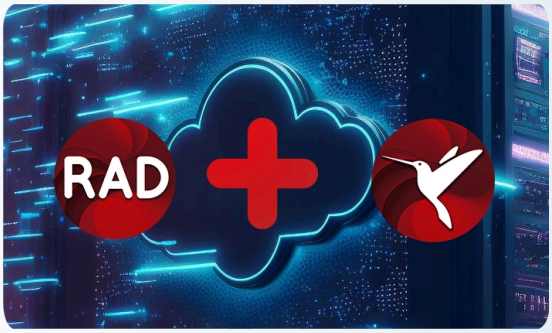

Startup Layout
Extend the IDE
Manage Features
GetIt Package Manager
Skia4Delphi_VCL.dproj
C:\Users\Public\Documents\Embarcadero\Studio\23.0\Samples\Object Pascal\VCL\Skia4Delphi\
Project1.dproj
D:\SPP\Blaise\Blaise_UK_114_115_2023\Authors\David Dirkse\stepbystep-2\
PGGLedenAdmin.dproj
F:\Nieuwe Ledenadmin\
Project1.dproj
D:\D12Athens\
Learn
How to create a real iOS app even if you do not have a Mac...
Have you ever wanted to create an app that works on iOS devices such as an iPhone or iPad? Follow along in this comprehensive, step by step guide as Embarcadero Developer Advocate Ian Barker shows you how to create...
How to create a real Android app step by step guide | Ian B...
Learn more about Embarcadero Technologies products at https://embarcadero.com
What can you do with RAD Studio 12 | Ian Barker
RAD Studio 12 is here and it's AWESOME. Join Ian Barker as he run through some of the features with some cool...
What's New in RAD Studio 12 Athens
Join us for this special live presentation where we're going to unveil the details of exactly what we have coming in the next release of RAD Studio....
Enterprise Software Development Article Challenge - Results
There were a lot of great entries for our Enterprise Software Development Article Challenge showing the use of Delphi, C++Builder, InterBase and RAD Server in many different enterprise use cases. Join us as we review the entries and...
See What's New in RAD Studio 11.3 Alexandria - Webinar R...
Embarcadero has just released RAD Studio 11 Alexandria Release 3 (RAD Studio 11.3, Delphi 11.3, and C++Builder 11.3)....
Looking Forward with Modern Delphi - Delphicon 2023
Delphi fundamentally changed software development with its release 28 years ago, but it didn't stop. As software development and the platforms we use evolve, so does Delphi. Today's modern Delphi stands apart from other...
User Interface Design with Actions - Ray Konopka - Delphic...
Actions have been a core feature of Delphi for quite some time, and have recently been added to the FMX framework. Unfortunately, this extremely powerful feature of Delphi is still widely underused and in many cases misused by...
Secrets of Visual Design on Windows 11
Good apps work properly, are easy to use, and fulfill their user's needs. Great apps do all this and look amazing too. Never underestimate the wow factor in customer...
Upgrading and Maintaining Delphi Legacy Projects
Get Bill's book on Delphi Legacy Projects https://wmeyer.tech/books/
Legacy projects represent code that continues to provide...
Markdown_HTML Rendering in RAD Studio 11.2 Alexandria
The RAD Studio 11.2 IDE now supports Markdown (.md) files and both Markdown and HTML richly formatted previews. Opening a Markdown file will show a rich rendered preview of the file....
Active Code Rendering - New in 11.2 Alexandria
Starting RAD Studio 11.2, the code editor will now render code between inactive IFDEF-s, i.e., code that is conditionally compiled out / not compiled, differently from code that will be...
FireDAC SQL Highlighting new in RAD Studio 11.2 Alexandria
The FireDAC SQL Editor now has SQL Highlighting to improve your productivity when editing SQL.
Upgrade or download the...
iOS Simulator in Delphi 11.2 Alexandria
The iOS Simulator support enables developers to test and debug their Delphi applications on different Apple devices and on multiple form factors using the iOS Simulator, without the need to buy the specific hardware. It also...
See What's New in Delphi, C++Builder, and RAD Studio 11....
Find out what is new in the 11.2 Alexandria release of your favorite developer tool: RAD Studio, Delphi, and C++Builder. The new 11.2 release will be binary compatible with 11.0 and 11.1 Alexandria, adding new features and...
Navigator
Your code at your fingertips. Ever wanted to jump to the uses clause, to a class's constructor, to a property definition? Navigator lets you move between any section of code quickly, easily, and without your fingers leaving the...
Close Welcome screen when opening a new project
Palette Projects Model View Data Explorer Multi-Device Preview

Mais APIs do Windows prontas para uso em Object Pascal. Conjunto abrangente de todos os cabeçalhos de API do Windows convertidos para Object Pascal, para facilitar para os desenvolvedores Delphi a chamada de qualquer API da plataforma Windows



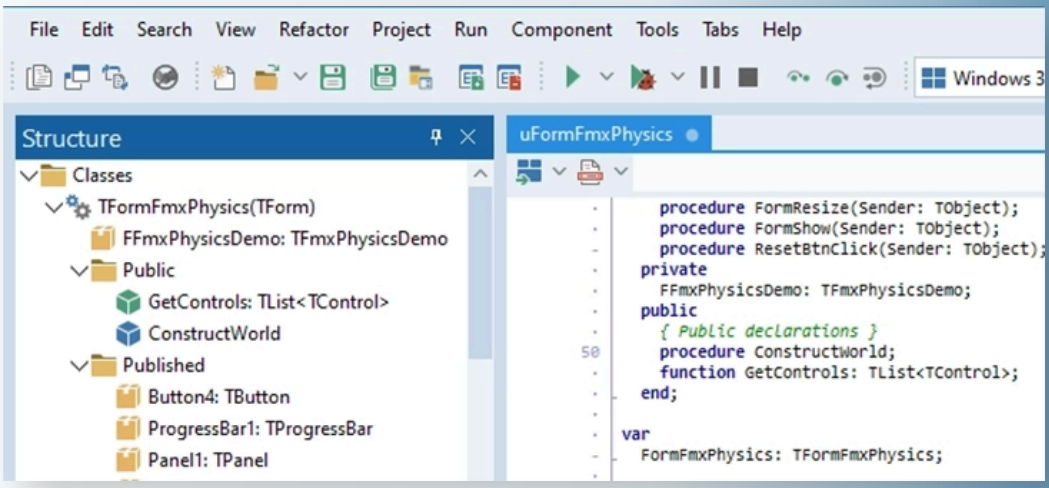
Suporte a QBE no FireDAC,
Novo assistente JSON para Delphi. Query-by-Example disponível no FireDAC. Assistente de mapeamento de dados JSON para gerar classes correspondentes à estrutura de dados JSON, mapear dados para objetos como XML e transmitir para um novo arquivo

Use a autenticação biométrica!
O **RAD Studio 12** oferece um novo componente de **autenticação biométrica móvel** para aplicativos móveis FireMonkey aplicativos

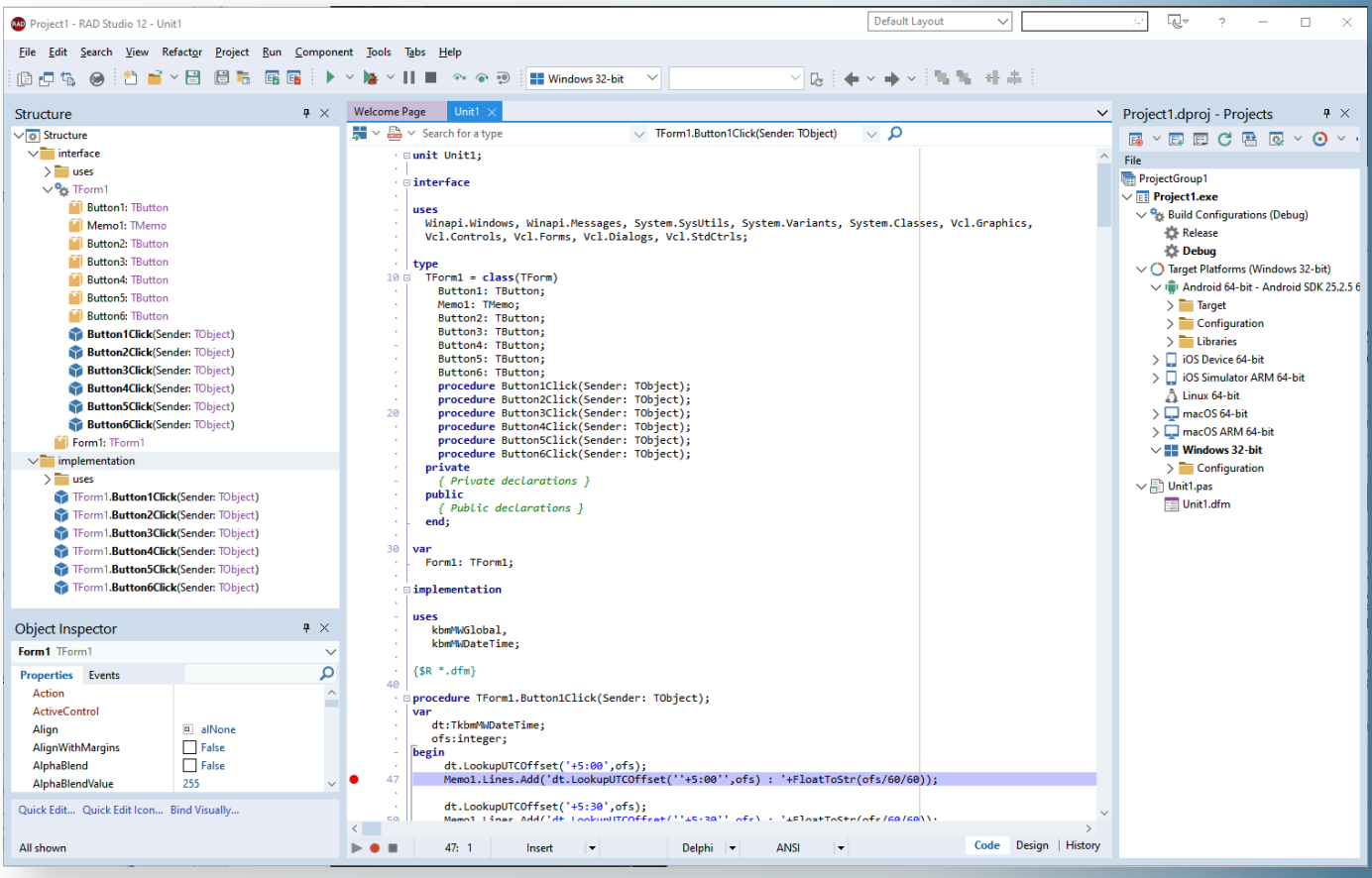


Implemente o Embedded InterBase Dev Edition!
O RAD Studio 12 vem com o recém-lançado InterBase 2020 Update 5 Developer edition e a Edição IBLite/ToGo

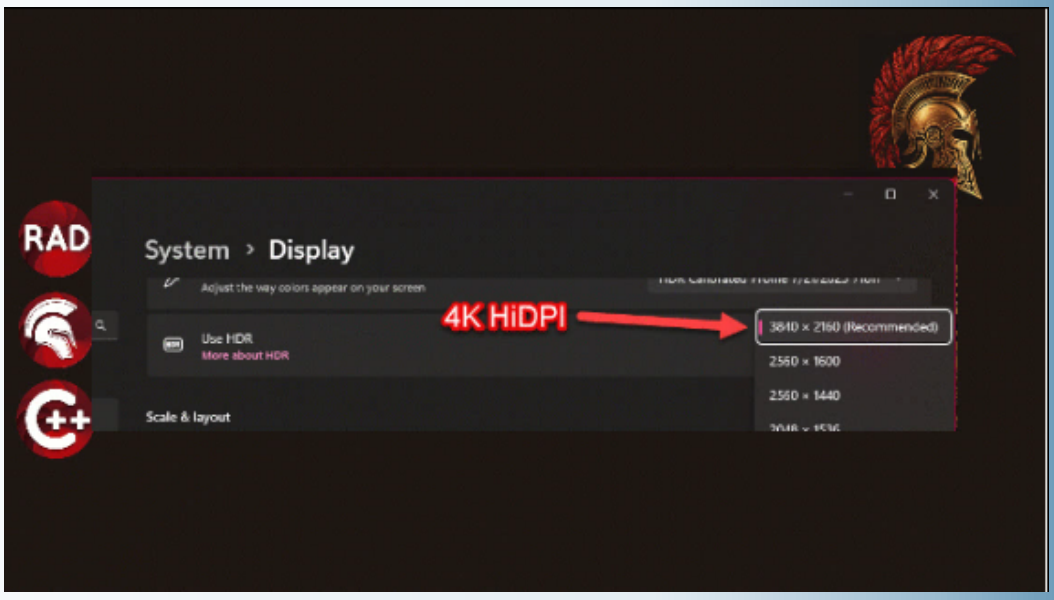
Segurança aprimorada de aplicativos por meio de restrições de SQL
Segurança mais profunda do aplicativo por meio de restrições aos comandos SQL, bloqueios de vários comandos e alterações de SQL

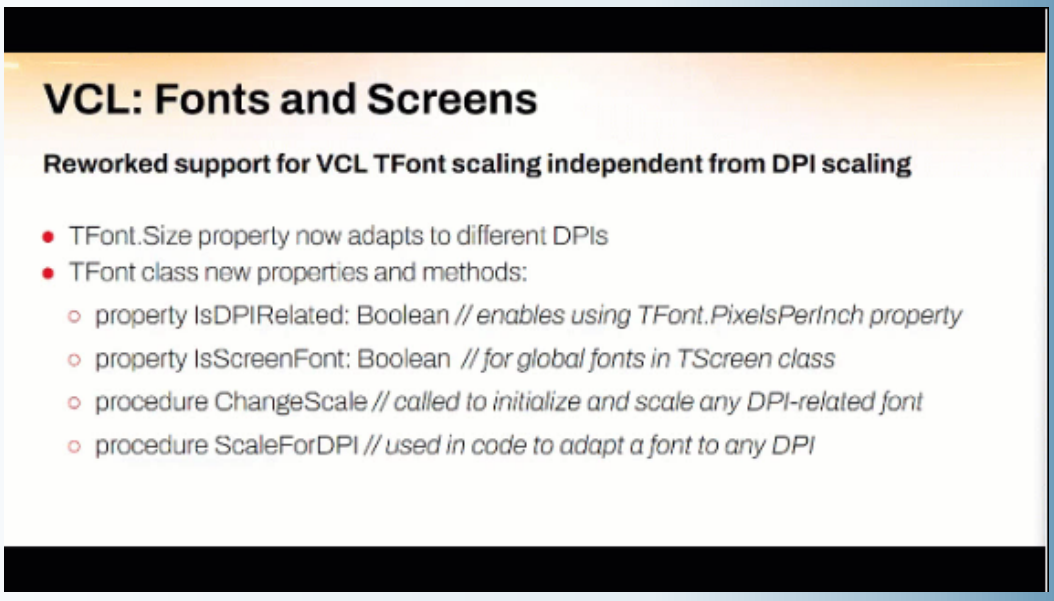


Suporte para IDs inteligentes no RAD Server
APIs REST hospedadas mais avançadas e flexíveis com novos **IDs inteligentes (Sqids).** Melhor desempenho, aprimoramentos na paginação de dados, melhor autenticação de sessão.



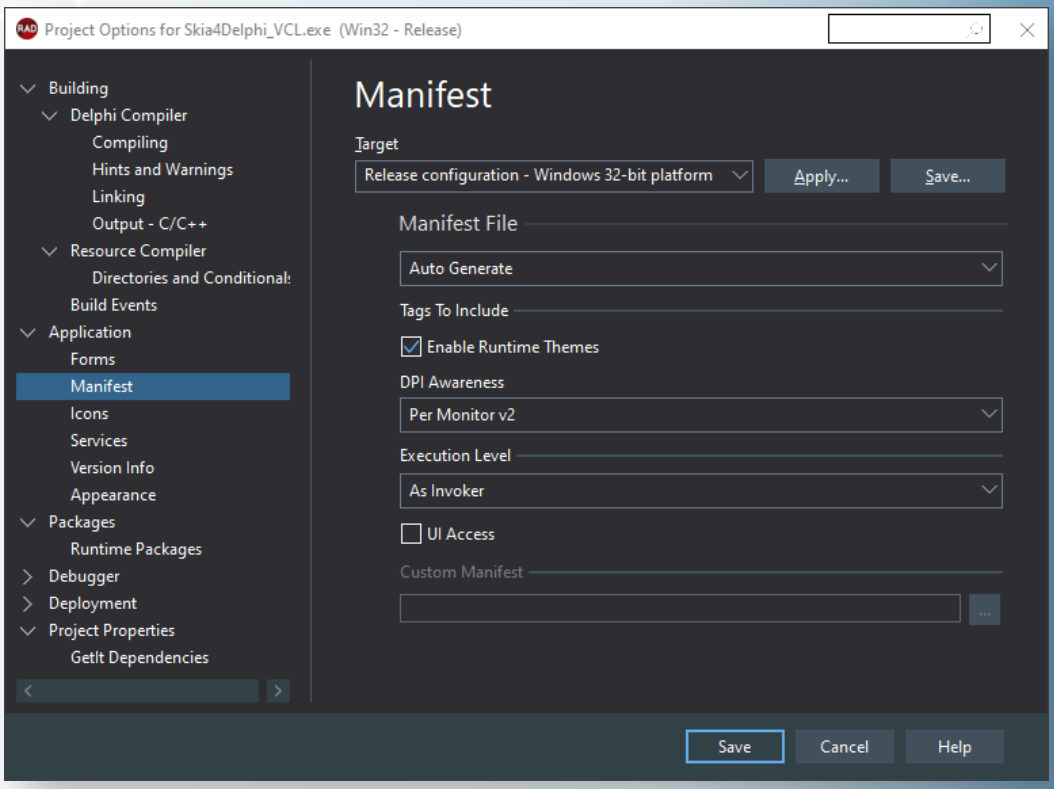
Use o RAD Studio em telas 4k+!
O RAD Studio 12 oferece suporte a alto DPI para o IDE permitindo que os desenvolvedores trabalhem em telas maiores e de alta resolução. O suporte total para os mais recentes monitores de alta resolução 4k+ melhora as atividades diárias do atividades diárias do desenvolvedor com fontes e ícones mais limpos e nítidos, além de suporte a alta resolução em todas as nas janelas do IDE, inclusive nos designers de formulários VCL e FMX e no editor de código



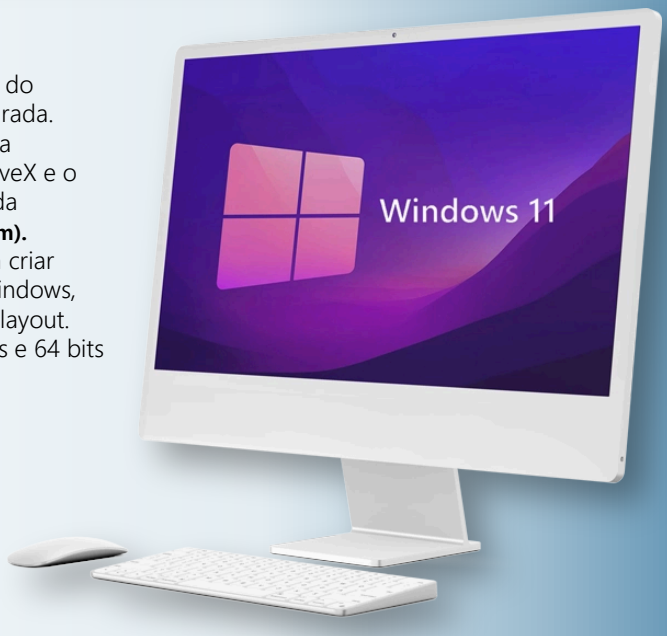



RAD Studio 12
Embarcadero

Options
IDE
Default Folders
Compiling and Running
Build Log
Component Toolbar
Environment Variables
File Association
Desktop and Layout
Welcome Page
Project Upgrading
LiveBindings
Saving and Recovering
GetIt Package Manager
User Interface
Object Inspector
Palette
Colors
Difference Viewer
Merge Viewer
Reopen Menu
IDE Style
Structure
Form Designer
FireUI Live Preview
Device Manager
High DPI
Editor
Line Endings
Search
Tabs
Status Bar
Language
Color
Structural Highlighting
Display
Key Mappings
Language
Toxicity Metrics
Delphi
C++
HTML
Formatter
Version Control
Git
Mercurial
Subversion
Deployment
Connection Profile Manager
Provisioning
SDK Manager
Modeling
Debugger
Visualizers
Disassembly
Event Log
Embarcadero Debuggers
Language Exceptions
Native OS Exceptions
High DPI
VCL Designer High DPI mode *
Automatic (Screen PPI)
Custom Design PPI
96
Scale grid size / snap tolerance to design PPI *
VCL Designer High DPI mode *
Automatic (Screen PPI)
Automatic (Screen PPI)
Low DPI (96 PPI)
User Editable
Scale grid size / snap tolerance to design PPI *
(*) Feature not supported by FireMonkey
Save
Cancel
Help



Alvo Windows 11
Suporte oficial para provisionamento do **Windows 11** com geração **MSIX** integrada. Navegador da Web componente para Windows, com suporte para o IE ActiveX e o novo controle **Microsoft WebView 2** da **Microsoft (Edge baseado no Chromium).** Aprimorado VCL Form Designer para criar visualmente aplicativos nativos do Windows, com dicas de snap-to e diretrizes de layout. **Delphi e RTL** para-**Windows** de 32 bits e 64 bits **Windows**.



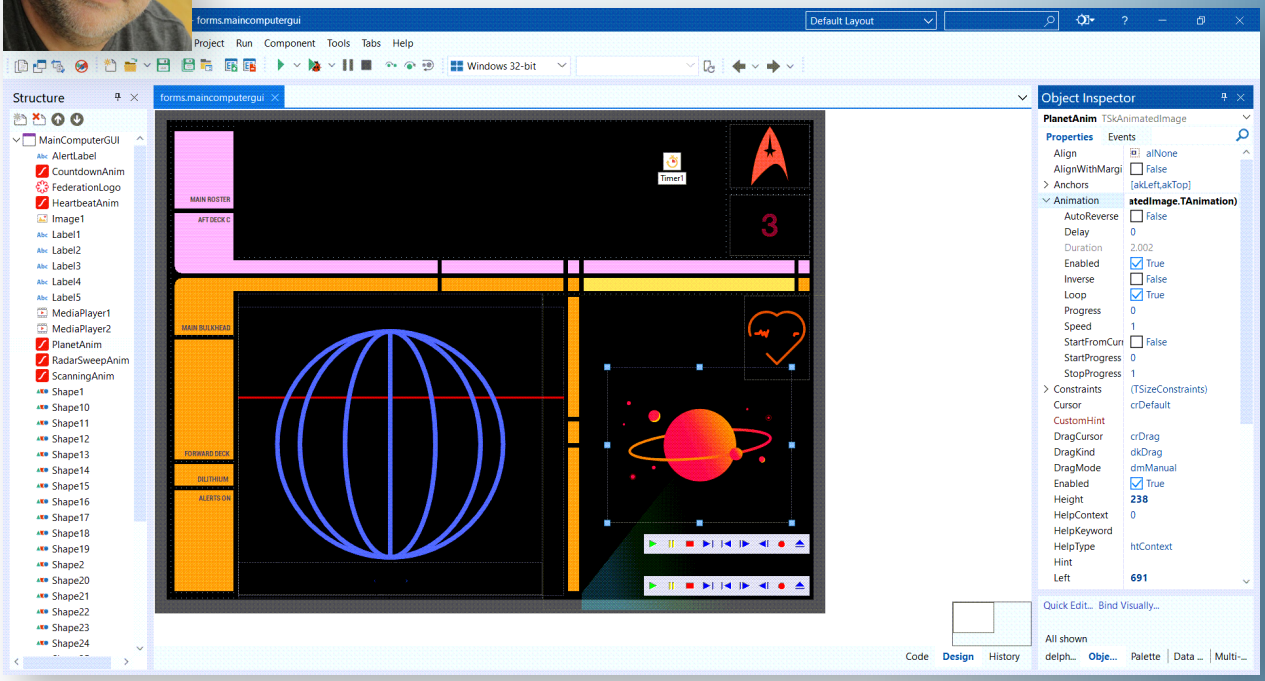
Use os estilos da VCL no momento do design!
O VCL Styles agora oferece suporte em tempo de design:
Faça o protótipo de UIs com estilo ainda mais rápido, vendo imediatamente em tempo de design como seus formulários e controles estilizados serão exibidos durante a execução. A visualização em tempo de design como os estilos afetarão a IU em tempo de execução melhorando o processo de design e teste das interfaces de UIs modernas. Criar interfaces de usuário melhores com mais rapidez é especialmente útil quando se trabalha com estilos por controle.

Implementação no Apple Silicon série M!
Compile para macOS (Apple Silicon série M) e use o novo pacote universal para envio para a AppStore. Agora você pode compilar para os processadores Intel existentes para os novos do macOS (Apple Silicon). A compilação para as versões mais recentes do processador permite o desempenho mais rápido em todas as plataformas e suporta o empacotamento universal para a loja de aplicativos do macOS app store.

Colabore remotamente!
Suporte aprimorado à área de trabalho remota para VCL e IDE ajudando os desenvolvedores que trabalham remotamente do escritório. Depuração aprimorada para aplicativos remotos e aplicativos Windows de 64 bits remotos e locais e aplicativos macOS Aplicativos de 64 bits (Intel e **ARM**). Suporte aprimorado à suporte à área de trabalho remota aumenta a eficiência da sua equipe eficiência de sua equipe e melhora seus resultados.

UM "CONSOLE DE COMPUTADOR DE NAVE ESTELAR DO FUTURO - EXEMPLO DE USO DO SKIA4DELPHI

BY IAN BARKER



O código para esse projeto pode ser encontrado em
https://github.com/checkdigits/spacecomputer

```
function TMainComputerGUI.RandomKlingonString(const DesiredLength: integer): string;
var  Klingon:   char;
begin
 Result := '';
 for var count := 1 to DesiredLength do
 begin
  if Random(5) mod 5 = 0 then
   Result := Concat(Result, ' ')
  else
   begin
    Klingon := Char($F8D0 + Random(25));
    Result  := Result + Klingon;
   end;
 end;
end;

function TMainComputerGUI.RandomLineOfNumbers: string;
var
 iWidth: integer;
 iSpaces: integer;
 iIndex: integer;

const
 cSpaces: array[0..5] of integer = (4, 6, 4, 8, 4, 4);

begin
 Result := '';
 for var iLoop := 0 to 10 do
 begin
  iSpaces := cSpaces[iLoop mod 6];
  Result  := Result + Format('%.4f%s', [Random, StringOfChar(' ', iSpaces)]);
 end;
end;
```

12 vezes 12 novos recursos no Delphi 12

O RAD Studio 12 inclui alguns grandes aprimoramentos para o C++Builder e o webinar de lançamento e outros conteúdos on-line destacam isso. No entanto, também é uma versão fantástica para os desenvolvedores **Delphi**. Eu compilei 12 listas com 12 melhorias cada para o **Delphi 12**. Portanto, esta não é uma lista de 12 melhorias para o Delphi 12. É uma lista de 12x12=144 melhorias, mais meia dúzia de melhorias para Windows nativo, total de 150 - excluindo todos os aprimoramentos existentes para o C++Builder, já que aqui eu quero enfatizar os aprimoramentos para o Delphi 12, mas não para o C++Builder. Quero enfatizar o lado do Delphi (mas a maioria dos recursos abaixo é, de fato, para ambas as linguagens).

https://blogs.embarcadero.com/3-x-12-vcl-enhancements-in-delphi-12/.

A primeira postagem do blog continha 3 x 12 VCL Enhancements in Delphi 12, como você pode ler em

https://blogs.embarcadero.com/3-x-12-firemonkey-and-android-enhancements-in-delphi-12/

A segunda postagem do blog se concentrou no FireMonkey e no suporte à plataforma Android, como você pode ver em

A terceira parte tinha três listas focadas em três áreas relacionadas, Delphi Runtime Library (RTL), acesso a banco de dados e acesso à Internet, como você pode ver em

https://blogs.embarcadero.com/3-x-12-rtl-data-and-internet-enhancements-in-delphi-12/

Esta última postagem do blog da série tem 12 entradas para o IDE, o instalador e a linguagem Delphi e a linguagem Delphi. Além de um bônus de 6 sobre a integração da API do Windows, elevando o total geral para 150.

Em VCL: MDI e gerenciamento de formulários

MDI revisado para suporte a estilos **HighDPI** e **VCL**

Os formulários filhos agora podem ter uma nova borda moderna e plana (*a nova propriedade é* `TStyleManager.ChangeChildFormSystemBorder`)

Os símbolos de quadro desativados do **MDI Child** não são desenhados

A propriedade pai funciona para o aninhamento de qualquer formulário em outro formulário, com gerenciamento completo de quadros

Limpeza e aprimoramentos significativos do **MDI**

O novíssimo controle `TFormsBar`

Ocultação automática de subjanelas minimizadas

A interface **`IFormVisualManage`**r

A propriedade **VisualManager** da classe `TCustomForm`

O assistente **MDI** atualizado

Uma nova propriedade `ShowInTaskbar` para `TForm`

Um novo construtor `CreateScaledNew` na classe `TCustomForm`

Controles VCL

Suporte à visualização em mosaico para `TListView` (incluindo as novas propriedades `TileOptions` e `TileColumns`)

`TGroupCollection` agora tem duas propriedades Items

Novo estilo de **ToolButton**: `tbsWholeDropDown`

O controle **TNumberbox** tem um modo adicional "nbmInt64" que aceita números de 64 bits como entrada

O **ActivityIndicator** tem suporte para cores personalizadas (`propriedade IndicatorCustomColor`), novos ícones predefinidos **RotatingLines** e **Refresh** e outros aprimoramentos

O **`TControlList`** tem as novas propriedades `SelectedItemsCount` e `SmoothMouseWheelScrolling`

O **`TControlList`** oferece suporte a tipos de elementos adicionais, como `TControlListCheckBox` e `TControlListRadioButton, TWICImage` e `TImageCollection` agora têm um método `Dormant`() para reduzir o uso de **GDI (Graphics Device Interface)**

O novo componente **TSkLabel** baseado em **Skia** para **VCL**

O novo componente **TSkPaintBox** baseado em **Skia** para a **VCL**

O novo componente **TSkAnimatedPaintBox** baseado em **Skia** para a **VCL**

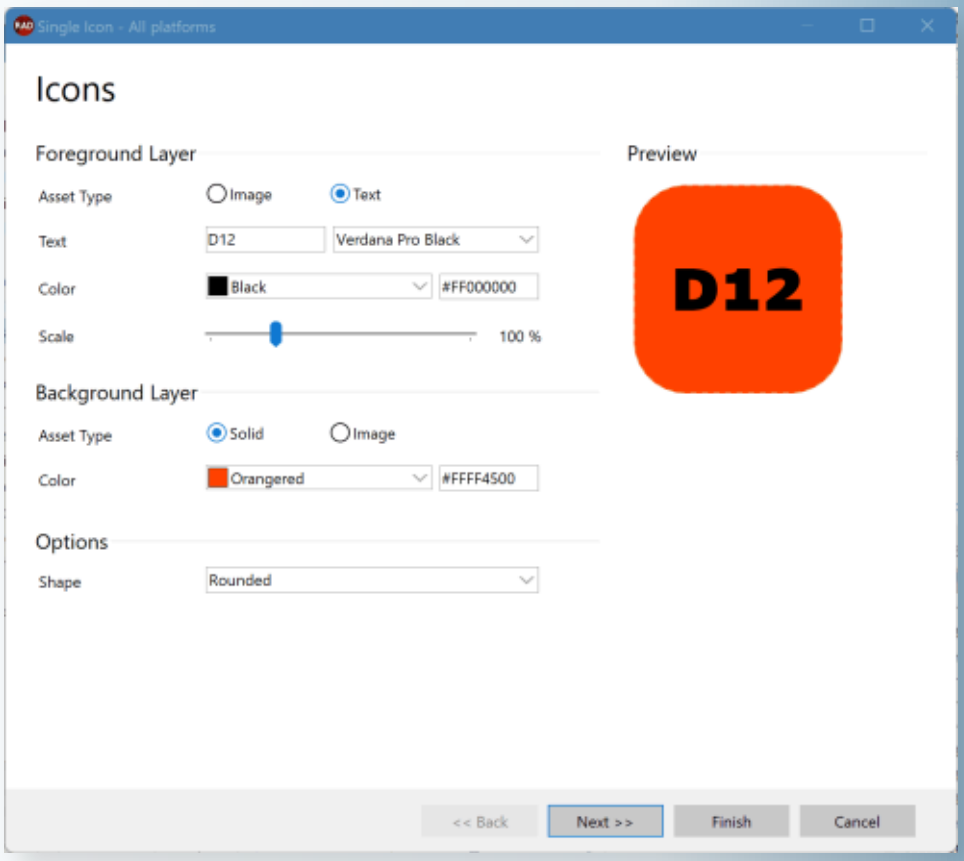
O novo componente `TSkSvg` baseado em **Skia** para **VCL**

Bitmaps (GDI do Windows)
Um bitmap é um objeto gráfico usado para criar, manipular (dimensionar, rolar, girar e pintar) e armazenar imagens como arquivos em um disco. Esta visão geral descreve as classes de bitmap e as operações de bitmap.

No IDE

❶ A caixa de diálogo Localizar em arquivos agora tem uma nova opção "Máscara de exclusão de subdiretório"
❷ O destaque de sintaxe foi adicionado às dicas do **Error Insight,** à barra de ferramentas de navegação e à pilha de chamadas
❸ A sintaxe da visualização **Structure** destaca métodos e tipos e adiciona destaque de sintaxe às mensagens do **Error Insight**
❹ A página **Options - IDE - Saving and Recovering** (**Opções - IDE - Salvando e recuperando**) tem uma nova caixa de seleção para salvar o estado do editor
❺ A janela **Markdown** agora muda de cor quando o tema do **IDE** é alterado
❻ A página de boas-vindas agora suporta a rolagem suave da roda do mouse
❼ As contagens de bitmap **GDI** são menores por meio do **IDE**, pois as imagens são feitas inativas
❽ Os modelos de código e as palavras-chave de linguagem agora podem ser exibidos no Autocompletar código baseado em **Delphi LSP**
❾ O recurso autocompletar código agora adiciona chaves de **matriz [ ]** para tipos de matriz
❿ Todas as plataformas Assistente de ícone único
⓫ As mensagens do **PAServer**, incluindo dicas, serão mostradas no painel Mensagens do **IDE**
⓬ Nova interface **ToolsAPI**, **IOTARawEditReader**

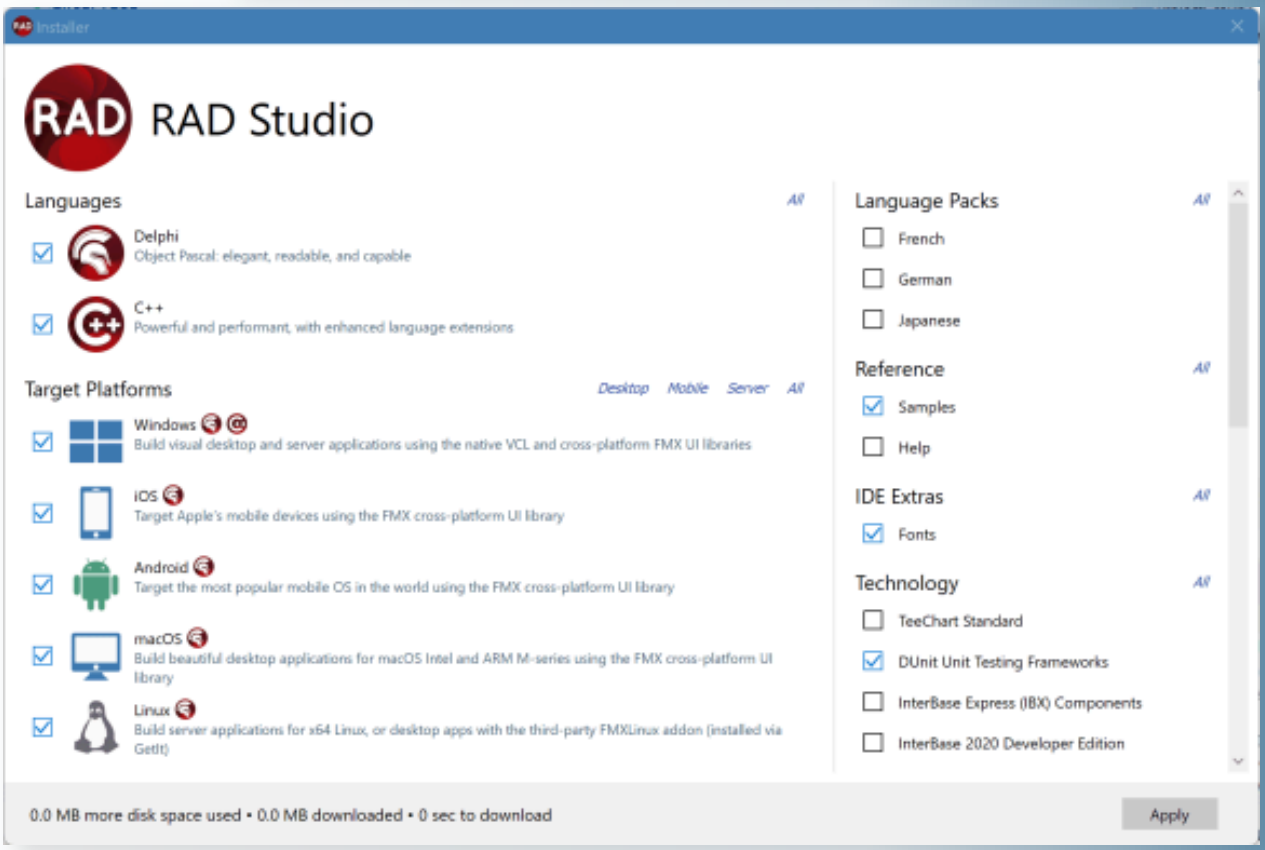
Abaixo: O assistente de ícone único para todas as plataformas



No GetIt e no Instalador

- O **Platform Manager** foi renomeado para **Feature Manager**
- O **Feature Manager** tem uma interface de usuário completamente nova, construída com controles e estilos **VCL** regulares
- O Feature Manager oferece todas as opções em uma única página
- O Feature Manager separa os idiomas e as plataformas que o usuário deseja instalar
- O Feature Manager inclui configurações predefinidas para cenários comuns, como desktop ou celular
- O Feature Manager tem um novo botão "Errors" (Erros) que mostra diretamente os erros de instalação e tem um link direto para o arquivo de registro de erros
- A ferramenta de linha de comando GetItCmd agora está registrando no arquivo GetItInstall.log
- O gerenciador de pacotes **GetIt** tem a opção de carregar vários pacotes **GetIt** locais com uma única operação
- A versão integrada do **DUnitX** foi atualizada
- A versão integrada do Indy foi atualizada
- As ferramentas de tradução VCL, há muito obsoletas, foram removidas da instalação do produto principal
- O suporte bastante antigo ao Modeling tornou-se um recurso opcional no Feature Manager

Abaixo: A nova janela do Feature Manager



E, finalmente, na linguagem Delphi

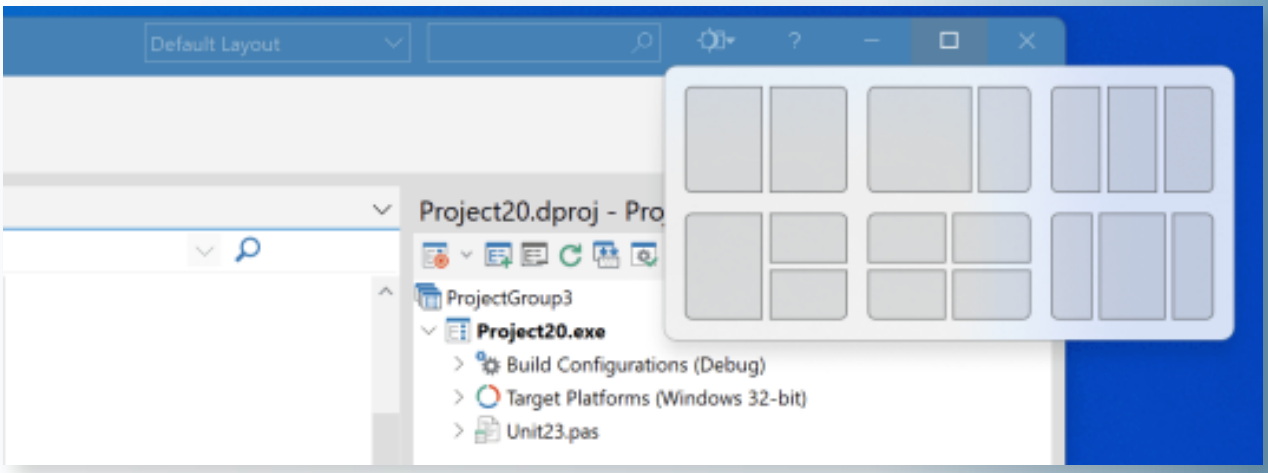
1. Cadeia de caracteres literal longa Com mais de 255 caracteres
2. Literais de string de várias linhas
3. Diretiva **TEXTBLOCK** para especificar o formato de quebras de linha de strings com várias linhas
4. **NativeInt**, um tipo mapeado para Integer ou Int64, dependendo da plataforma, agora é um alias de tipo fraco
5. Avisos aprimorados em classes genéricas
6. Novo símbolo **LLVM** definido em todos os compiladores Delphi baseados em **LLVM**
7. Opção para exportar o gráfico de usos de unidades em um arquivo **GraphViz** (`-graphviz`)
8. Capacidade de excluir famílias de unidades do arquivo do **GraphViz** (`-graphviz-exclude`)
9. Suporte para comparações de **NaN** (**não é um número**), conforme exigido pelo **IEEE**
10. Código gerado otimizado para operações div quando o divisor é uma constante
11. Duas novas funções para a unidade `System`, `GetCompilerVersion` e `GetRTLVersion`
12. As **exceções de ponto flutuante** agora estão desativadas por padrão em todas as plataformas

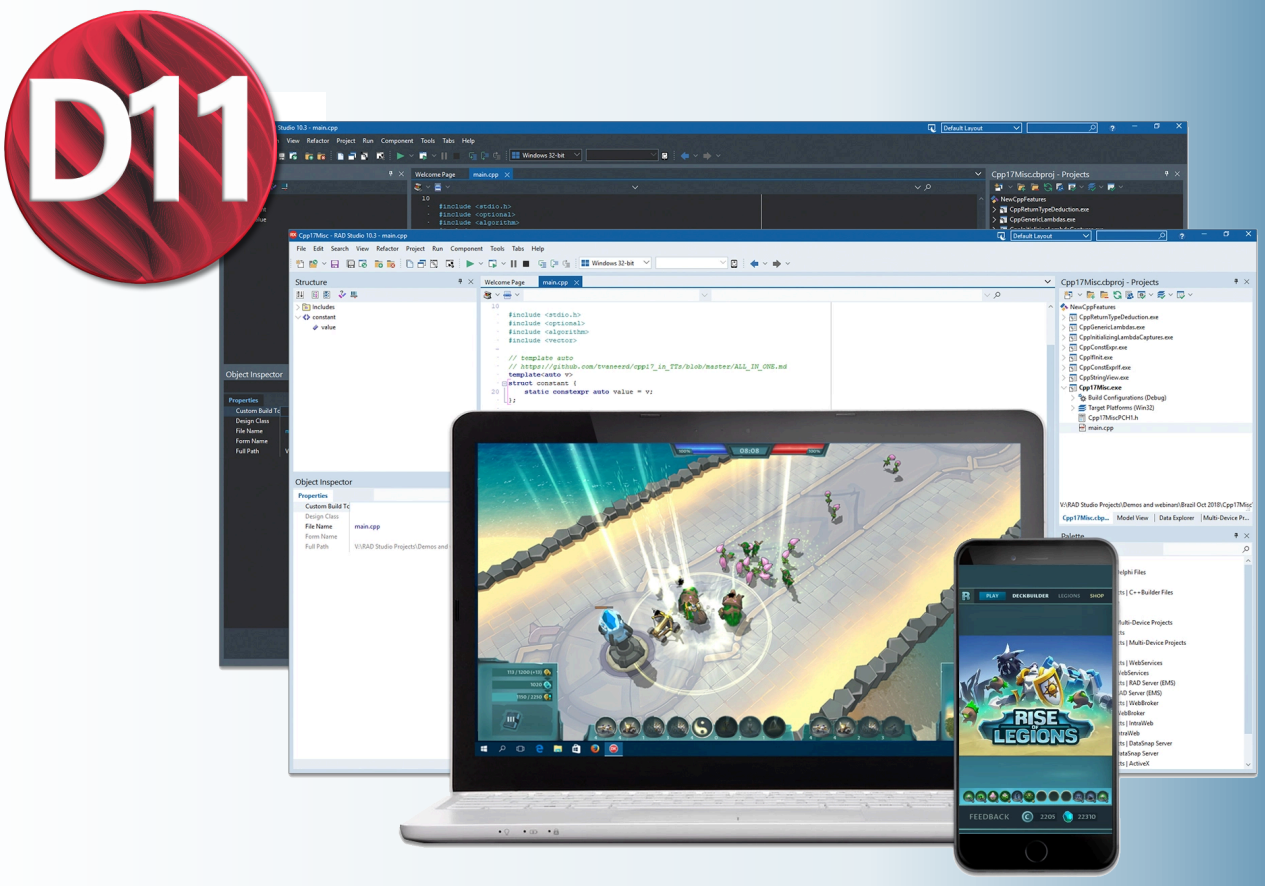
O exemplo de literais de cadeia de caracteres Delphi multilinha no IDE é visto na página 1 deste artigo.

Seção de bônus: Na plataforma Windows

- Nova definição de **WinAPI** baseada nos metadados **WinMD** da **Microsoft** (*311 arquivos de cabeçalho com 41 MB de código* **Delphi**)
- **APIs WinRT e API** de controle **WebView** 2 atualizadas para a versão mais recente
- **Navegador Edge:** UserAgent está disponível em `ICoreWebView2Settings`; **ICoreWebView2Profile2** contém os métodos `ClearBrowsingData`, `ClearBrowsingDataAll`, e `ClearBrowsingDataInTimeRange`; evento `OnDownloadStarting` do `TEdgeBrowser`, método `NavigateWithWebResourceRequest`, métodos `Print` e `ShowPrintUI`
- Novos estilos do **Windows 11**
- Método **`EnableImmersiveDarkMode`** e propriedade `RoundedCorners` da classe `TForm` para `Windows 11`
- Suporte à barra de título para layouts de snap do Windows 11 (veja a imagem abaixo)

O Snap Layout para a barra de título também funciona para o IDE





FREE COMMUNITY EDITION

Obtenha a Community Edition gratuitamente

Delphi IDE gratuito com todos os recursos para criar aplicativos nativos multiplataforma

Posso obter o Delphi CE?

Se o VOCÊ for uma pessoa física, poderá usar o Delphi CE para criar aplicativos para seu próprio uso e aplicativos que poderá vender até que sua receita atinja US$ 5.000 por ano.

Se o VOCÊ for uma pequena empresa ou organização com receita de até US$ 5.000 por ano, também poderá usar o Delphi CE. Quando a receita total da sua empresa atingir US$ 5.000 ou a sua equipe aumentar para mais de cinco desenvolvedores, o senhor poderá passar para uma licença comercial irrestrita com a edição Professional.

O Delphi CE também é perfeito para **startups** em estágio inicial que estão iniciando sua visão de produto antes de garantir capital! Desenvolva seu aplicativo profissional com a Community Edition sabendo que o senhor pode pular a curva de aprendizado que seus concorrentes enfrentam ao criar aplicativos para várias plataformas.

Consulte as perguntas frequentes sobre a Community Edition para obter mais detalhes.

`https://www.embarcadero.com/products/delphi/faq`

ALTERAÇÕES NAS INTERFACES LCL

COCOA

- **IME** (Editor de método de entrada, que permite a entrada de caracteres não presentes fisicamente em um teclado do computador) com suporte total, como **Chinês/Japonês/Coreano, DeadKeys, Emoji e símbolos**.
- Suporte total a vários monitores.
- Suporte total a docking, incluindo o IDE (*ambiente de desenvolvimento integrado*).
- Cursor totalmente refatorado e significativamente aprimorado, compatível com o **macOS Ventura**.
- `TPageControl` significativamente aprimorado.
- Muitos controles aprimorados (`TComboxBox/TListBox/TDateTimePicker/TStaticText/TSpeedButton/TPanel` etc.)
- Muitos vazamentos de memória foram corrigidos.

Qt

- Implementação de `TCheckBox.Alignment` e `TRadioButton.Alignment`.
- `TCustomComboBox.AdjustDropDown` e `TCustomComboBox.ItemWidth`.

Qt5

- O **Qt5** usa loop de eventos nativo em todas as plataformas.
  As ligações **C** são atualizadas. A versão mínima das ligações **C** para o **lazarus 3.0 é 1.2.15**.
- Observe que a maioria das distribuições **Linux** não terá uma biblioteca **libqt5pas** apropriada até o próximo lançamento após o lançamento formal do **Lazarus 3.0**.
  Construa a sua própria a partir da árvore de código-fonte do Lazarus ou faça o download em **`https://github.com/davidbannon/libqt5pas/releases/latest`**
- Implementado `TCheckBox.Alignment` e `TRadioButton.Alignment`.
- Implementados `TCustomComboBox`.AdjustDropDown e TCustomComboBox.ItemWidth.

Qt6

- Widgetset do **Qt6** implementado. As ligações **C** são baseadas no **Qt6 6.2.0 LTS**.
  A versão mínima do **C** bindings para o **lazarus 3.0 é 6.2.7.**
- Observe que a maioria das distribuições **Linux** não terá uma biblioteca libqt6pas apropriada até o próximo lançamento após o lançamento formal do **Lazarus 3.0.**
  Construa a sua própria a partir da árvore de código-fonte do **Lazarus** ou faça o download em **`https://github.com/davidbannon/libqt6pas/releases/latest`**
- `Implementado TCheckBox.Alignment` e `TRadioButton.Alignment.`
- Implementou `TCustomComboBox.AdjustDropDown` e `TCustomComboBox.ItemWidth.`

Gtk3

- Os vínculos pascais do **Gtk3** foram completamente reformuladas.
- Diversos aprimoramentos de estabilidade.
- Agora requer **GTK >= 3.24.24** e **Glib2.0 >= 2.66**

MUDANÇAS NA LCL

TCUSTOMIMAGELIST

TCustomImageList se tornou mais extensível:

1. Foi adicionado o método `MarkAsChanged` protegido, que define `FChanged` como `true` (*isso permite criar acionadores personalizados para o evento* `OnChange`).
2. Os métodos virtuais protegidos `DoAfterUpdateStarted` e `DoBeforeUpdateEnded` são adicionados. Eles são chamados no primeiro BeginUpdate e no último EndUpdate, respectivamente.

TTaskDialog

- Comportamento antigo do **Win32**:
  Um ícone de espaço reservado era usado para
  `FooterIcon = tdiNone e MainIcon = tdiNone`.
- Novo comportamento Win32:
  Nenhum ícone é usado para `FooterIcon = tdiNone e MainIcon = tdiNone`.
- Motivo: Remoção da falha de desenho.
  O texto foi movido para permitir mais conteúdo e melhor alinhamento.
  Consulte o problema `#39172`

TSpeedButton

- Comportamento antigo: As legendas de várias linhas só podiam ser inseridas por código.
  E as legendas com várias linhas eram sempre alinhadas à esquerda.
- Novo comportamento: As legendas de várias linhas também podem ser inseridas no **ObjectInspector** objetos. Nova propriedade Alinhamento para especificar se a legenda deve ser alinhada à esquerda/direita ou centralizada. Padrão: centralizada, como no **Delphi**.
- Motivo: melhor usabilidade.

Alterações em TLabel.Transparent,. Color e .ParentColor

- Comportamento antigo: A propriedade **Transparent** era vinculada a `Color=clNone`
- Novo comportamento: Transparent é uma propriedade autônoma
- Motivo: Compatibilidade com o **Delphi** e para corrigir problemas de `ParentColor`.
- Solução: se você estiver definindo a propriedade `Color`, `Transparent` não será automaticamente alterado de `True` para `False` agora, você precisa fazer isso sozinho. Isso está em conformidade com o Delphi e também resolve problemas com alterações de `Color/ParentColor`.

TPanel.VerticalAlignment

- Comportamento antigo: A legenda do painel era sempre centralizada verticalmente.
- Novo comportamento: A nova propriedade `VerticalAlignment` (`taAlignTop`, `taAlignBottom`, `taVerticalCenter`) permite colocar a legenda também na parte superior ou na parte inferior do interior do painel.
- Motivo: Compatibilidade com o **Delphi** e melhor usabilidade

TCalendar

- somente se `MaxDate > MinDate`. Infelizmente, os **widgetsets do GTK2/3** não são compatíveis com isso, portanto, selecionar uma data fora do intervalo `MinDate/MaxDate` ainda será possível.

TCheckbox, TRadioButton

- Cálculo diferente do tamanho da `checkbox/radiobutton` para atender corretamente ao recurso de "**facilidade de acesso**" do **Win-10**. Consulte a edição `nº 39398`
- Consequência: os arquivos **`lfm`** conterão tamanhos diferentes desses controles (*se dimensionados automaticamente*) em comparação com as versões anteriores.

MUDANÇAS NA LCL CONTINUAÇÃO

Grids

- Agora você pode definir as propriedades `ParentColor` e `ParentFont` dos editores de células incluindo `goEditorParentColor` resp. `goEditorParentFont` na propriedade Options2 da grid.

TShellTreeView

- Nova propriedade `ExpandCollapseMode` que define opções para saber se um nó em colapso deve limpar seus nós filhos.
- `FileSortType` como `fstCustom` e fornecendo uma função de comparação personalizada no evento `OnSortCompare`.

TShellListView

- O `TShellListView` agora subclasses `TListItem`, para que possa armazenar informações de arquivo nele. Se você usar `OnCreateItemClass` para criar seu próprio descendente de `TListItem`, talvez seja aconselhável basear sua própria classe em `TShellListItem` em vez de em `TListItem`. Dessa forma, você também terá acesso à propriedade `FileInfo` do `TShellListItem`.

TTreeView

- Adiciona `ShowSeparators` como uma propriedade publicada, assim como `ShowLines` e `ShowRoot`. `TTrayIcon` Somente **gtk2**

MUDANÇAS DE IDE

Mapa de caracteres

- Caracteres redimensionáveis para melhorar a legibilidade.
- O mapa de caracteres foi separado do **IDE** e transferido para pacotes separados. O pacote do designtime é instalado por padrão para que não haja diferença no **IDE**. Agora os usuários podem acessá-lo em seus próprios aplicativos depois de adicionar o pacote de tempo de execução "`charactermappkg.lpk`" aos requisitos do projeto.

DEBUGGER

Opções do Projeto

- **"Run(F9)"** (**Executar**) pode ser **"Debug" (Depurar)** ou **"Run without debug" (***Executar sem depuração*). Nos modos de **Debug/Release**, o modo de liberação não invocará mais o depurador por padrão. Os modos de **Debug/Release** existentes devem ser editados para permitir isso.
- Configurações de " **Debugger backend** " por projeto. Além de escolher um **backend** específico das configurações globais do **IDE**, um **backend** pode ser configurado apenas para o projeto (*ou seja, configurações especiais do servidor* **gdb**)

Dialogs do IDE

- Janela **Watches** aprimorada
- Expandir/desdobrar para classes, registros, etc.
- Expandir/desdobrar com navegador paginado para arrays
- Arrastar e soltar para reordenar os watches
- Arrastar e soltar para criar watches de "nível superior" a partir de entradas aninhadas (`em listas expandidas/desdobradas`)
- Coluna de endereço para tipos com ponteiro interno (`classes, cadeia longa, Dyn Array, ponteiro (real)`)
- Power Button

MUDANÇAS DE IDE CONTINUAÇÃO

Janela Inspecionar aprimorada

- Corrigido: Atualização do valor quando o contexto é alterado
- Opções adicionadas para chamada de função / e "Conversor" (*consulte* `FpDebug: SysVarToLstr`)
- Adicionado filtro para pesquisar texto no nome ou no valor.
- Adicionado **ctrl-Up/Down/Page-Up/Page-Down** para navegar na grade.
- **Alt-Esquerda/Direita** para o histórico. E **ctrl-Enter** para selecionar.

Janela Avaliar/Modificar aprimorada

- Novo layout
- Adicionado DisplayFormat
- Opções adicionadas para chamada de função / e "Conversor" (consulte FpDebug: SysVarToLstr)

Janela Assembler aprimorada

- Adicionada navegação no histórico (para frente/para trás)
- Para FpDebug: adicionadas anotações para pular/chamar alvos e permitir clicar com o botão ctrl pressionado para desmontar o endereço do alvo.

FpDebug / LazDebuggerFp

- Aprimoramento da "**chamada de função**" na avaliação do watch.
  Consulte **FpDebug-Watches-FunctionEval**
- **%RAX** Acesso a registros da **CPU** na expressão de observação (*somente registros completos, ainda não AH ou AL ou EAX em 64 bits*)
- Funções intrínsecas: FpDebug-Watches-Intrinsic-Functions
- Operadores intrínsecos/extendidos: Fatia de `MyArray[1..3]` array com mapeamento de operador.
- **FpDebug-Watches**-Intrinsic-Functions#Intrinsic_Operators
- Opção para detectar "**variant**" e chamar **"SysVarToLStr"** no aplicativo de destino.
- Suspender/retomar threads individuais (*deve ser feito enquanto o aplicativo estiver pausado, e será aplicado na etapa/execução subsequente)*
- Aprimoramentos parciais da depuração na DLL:
  `https://wiki.freepascal.org/Debugger_Status#Other`
- Disassembler agora anota linhas para `call/jmp/jne/...` com informações sobre o endereço de destino (*nome da função, arquivo, linha*)
- **F7/F8 Step-Into/Over** agora pode ser usado para iniciar o depurador e executar a primeira linha do programa principal primeira linha do `begin/end` do programa principal.

LazDebuggerFpLldb (padrão no MacOS)

- Adicionado Mem-Limits (e `String`, `Pchar`, `Array`) à configuração do depurador.
  Consulte **`https://wiki.freepascal.org/LazDebuggerFp`**
  (`MaxMemReadSize`, `MaxStringLen`, `MaxArrayLen`, `MaxNullStringSearchLen`)
  Limitar os resultados padrão para `watches/locals/stack-params,...`
  pode evitar uma avaliação lenta.
  Os `Arrays` podem então ser pesquisadas na janela watches, usando o novo
  "paged browser for arrays" (*expandir por meio de* `[+]`)

Propriedades de ponto flutuante no ObjectInspector

- O **ObjectInspector** agora não permite explicitamente definir o valor de uma propriedade de ponto flutuante para +/-Inf ou **NaN (Not a Number)**.
- Motivo: embora `+/-Inf e NaN` sejam valores válidos para uma propriedade de ponto flutuante, eles não podem ser transmitidos (*portanto, o formulário não pôde ser carregado*) e a configuração para **NaN** causou estragos no **IDE**.
- Solução: definir o valor em tempo de execução (*no código*)

MUDANÇAS DE IDE CONTINUAÇÃO

Leitura de nomes de unidades em lfm

- O gravador de componentes do FPC 3.3.1 suporta opcionalmente a gravação de tipos com seus nomes de unidade como nome da unidade/tipo. O Lazarus agora pode ler isso.

Tipos de classe ambíguos

- Agora é possível registrar duas classes de componentes com o mesmo nome, por exemplo, fresnel.`TButton` e `StdCtrls.TButton`.
  Você pode até mesmo colocar ambas no mesmo formulário.

Editor

- Destaque para o **PasDoc**

Opções do IDE

Em Ferramentas -> Opções -> Ambiente -> Página da janela, essas três configurações agora estão ativadas por padrão:

- O título do IDE começa com o nome do projeto
- O título do IDE mostra o diretório do projeto
- O título do IDE mostra o modo de compilação selecionado

Isso foi solicitado por muitos usuários.
Se a barra de título do IDE não tiver informações sobre o projeto ativo, o usuário deverá abrir o **ProjectInspector** ou Opções do projeto para vê-las, o que é inconveniente.

Exemplos do Lazarus

- Uma nova abordagem para Exemplos que copia um Exemplo para uma nova área de trabalho (*evitando o problema de somente leitura do* **Linux**) e, possivelmente, um modelo de pesquisa mais fácil de usar.

ALTERAÇÕES NA INTERFACE IDE

COMPONENTES

TAChart

- O `TLegendClickTools` agora é capaz de detectar cliques em itens de legenda de série e relata a série clicada no novo evento `OnSeriesClick`.
- Nova `TDatapointMarksClickTool` que se torna ativa quando o usuário clica nas marcas de uma série.
- Nova propriedade `TickWidth` para os eixos do gráfico.
- Nova propriedade `FullWidth` para que o título e o rodapé do gráfico executem seu plano de fundo em toda a largura do gráfico.
- Nova propriedade `RandomColors` para `TRandomChartSource`.
- Novas propriedades `YIndexWhiskerMin`, `YIndexBarMin`, `YIndexCenter`, `YIndexBarMax`, `YIndexWhiskerMax e YDataLayout` em `TBoxAndWhiskerSeries` para uma atribuição mais flexível atribuição mais flexível de valores de y às partes da forma de caixa/pescador.
- Nova opção `aipInteger` no conjunto `TAxisIntervalParamOptions`, que define os rótulos dos eixos somente em valores inteiros e, assim, suprime os rótulos dos eixos em valores inteiros, portanto, suprime os rótulos intermediários indesejados em gráficos de barras e ajuda a impor rótulos em gráficos logarítmicos nas potências da base logarítmica (*geralmente 10*).
- Novo evento `OnAddStyleToLegend` para `TChartStyles`. Ele tem um parâmetro booleano `AddToLegend` com o qual você pode determinar se o nível da série que usa esse estilo é exibido na legenda.

TDateTimePicker

- Novas propriedades `MonthDisplay` e `CustomMonthNames`.
  Elas foram criadas para substituir propriedade `MonthNames`, que foi descontinuada.
- Nova propriedade `DecimalSeparator`. Permite que um valor especificado pelo usuário seja usado em vez de um caractere Colon codificado.

IDE INTERFACE CHANGES CONTINUATION

`TDBDateTimePicker`

- Adiciona Options como uma propriedade publicada.
- Publica a propriedade **DecimalSeparator**.
- Publica a propriedade `Alignment` ausente. Consistente com o `TDateTimePicker`,

`T(Float)SpinEditEx`

- Nova propriedade **Orientation** que permite organizar os botões giratórios horizontalmente.

`TCheckListBox`

- Adicionada as propriedades `HeaderColor` e `HeaderBackgroundColor`. Usadas em itens de lista em que a propriedade `Header` está ativada. Implementado para o conjunto de **widgets Win32.**

- O **lazbuild** agora pode compilar projetos PAS2JS passando a variável de ambiente PAS2JS com o caminho do executável do pas2js.
- Grupos de projetos com projetos pas2js agora podem compilar sem serem abertos.
- **Novo tipo de projeto APLICAÇÃO WEB PROGRESSIVA**
- **Novo tipo de projeto APLICAÇÃO WEB ELETRÔNICA**
- O **pas2jsdsgn** agora usa o pacote **SimpleWebServerGUI**, substituindo seu próprio controlador de servidor **http**.
- **F9**, Run agora constrói e inicia um servidor **HTTP** e um navegador
  Coleção de ícones do **Lazarus**
- Não é um componente, mas a instalação do **Lazarus** agora contém uma pasta com ícones de uso geral para uso em barras de ferramentas, menus, botões etc. de qualquer aplicativo GUI (*pasta* **`images/general_purpose`**).
  As imagens vêm em vários tamanhos e, portanto, são compatíveis com a lista de imagens em escala do **Lazarus v2.0+**. Autor: **Roland Hahn** (**`https://www.rhsoft.de/`**).
  Licença: Creative Commons CC0 (sem restrições de uso).

gir2pascal

- As fontes do `gir2pascal` (*uma ferramenta para converter* **GObject Introspection**em *descrições* **Pascal** *em arquivos* **Pascal**) agora estão incluídos na árvore de código-fonte do **Lazarus** (`diretório tools/gir2pascal`) e mantida lá.

lazdelphi

- Um complemento do IDE que adiciona um analisador de erros e dicas do compilador Delphi. Você pode executar o `dcc32.exe` como ferramenta externa ou executar o comando `before` nas opções do compilador e usar o analisador do **Delphi Compiler** para a saída, de modo que os erros/dicas na janela **Messages** possam abrir o código-fonte. Consulte Ferramenta do compilador Lazarus Delphi.

Formato de código Jedi

- Alguns aprimoramentos na formatação de código.
- A ferramenta de linha de comando `jcf` agora é um aplicativo de modo de texto e não requer mais o **XWindow** no **LINUX** para ser executada.

DockedFormEditor

- Caso ainda esteja usando o editor de formulário acoplado **sparta**, use o pacote **dockedformeditor** em seu lugar.

ALTERAÇÕES QUE AFETAM A COMPATIBILIDADE

CONFIGURAÇÕES DO IDE

RUN: Executar > Parâmetros de execução

O **`Working-Dir`** e o `Launch-/Host-App` agora podem ser especificados em relação ao **`Project-Dir`**. Como resultado disso, e devido a inconsistências entre **"Run in debugger"** e **"Run without debug"** alguns detalhes de como esses campos são resolvidos foram alterados. Portanto, em alguns casos, talvez seja necessário ajustar suas configurações.

O diretório de trabalho agora é determinado da seguinte forma

- BuildMode.RunParams "working directory" (diretório de trabalho) definido pelo usuário (*novo, agora pode ser relativo a diretório do projeto / para obter o "diretório de saída" que contém o exe*: `"$Path($(OutputFile))"`)
- Diretório do projeto, se não for virtual
  Diretório do aplicativo host (RunParams), ou se (e somente se) o aplicativo host estiver vazio do `Project.exe` (*Se o aplicativo host estiver em %PATH, então não há "diretório de trabalho"*).
  As duas primeiras etapas são as mesmas de antes da alteração. A terceira etapa era usada anteriormente apenas para "**debugging**", mas "**Run without debug**" usava: `"Launch-App", Host-App", Project.exe`.

Alteração

O caminho do **"Launch App"** não é mais considerado

O local do **Launch-/Host-App** agora é determinado da seguinte forma

❶ Um aplicativo com caminho absoluto é usado como dado

❷ Um caminho relativo (incluindo nenhum caminho) é resolvido como relativo ao Project-dir.

❸ Um aplicativo sem nenhum caminho (*se não for encontrado na etapa 2*) é pesquisado no ambiente **`%PATH.`**

Alteração

A pesquisa em **`%PATH`** era feita apenas por **"run without debug",** mas não por debug.
Agora ela é feita por ambos.

Alteração

Foi adicionada a verificação de um exe relativo ao diretório do projeto.

Incompatibilidade de LCL

`TLabel:` auto dimensionado e alinhado à direita

Comportamento antigo:

O rótulo auto dimensionado com `Alignment=taRightJustify` mas `Anchors=[akLeft,...]` crescia para a esquerda.

Novo comportamento: A etiqueta cresce até o momento.

Motivo:

Não foi possível implementar o comportamento também para rótulos ocultos sem extensões significativas na extensões significativas na **LCL**. A **LCL** tem um recurso diferente e mais genérico de ancoragem baseada em controle que produz o mesmo efeito (*veja Remedy down*), portanto, não é necessário nem desejado duplicar esse recurso e tornar o código da **LCL** mais complexo e propenso a bugs.

Remédio

: Use a ancoragem **LCL** em um controle secundário.

Ancore o lado direito da etiqueta em outro controle. Assim, o rótulo de tamanho automático crescerá para a esquerda, mas não se moverá para a direita quando o controle pai for redimensionado, como acontece com uma simples `akRight` sem um controle de referência.

MUDANÇAS QUE AFETAM COMPATIBILIDADE CONTINUAÇÃO

`TDateEdit/TTimeEdit`

O valor de `NullDate` foi alterado.
Motivo:
Era impossível selecionar de fato a data correspondente a **NullDate** (30 dez 1899 por padrão) no controle.

Solução (1):
Se o seu código dependia do fato de **NullDate** ser 0,0, será necessário ajustar o código.

Solução (2):
Se seu código usava **NullDate** para um **TTimeEdit**, altere-o para a nova constante `NullTime`.
OBSERVAÇÃO: **NullDate** é, na verdade, uma constante gravável. Isso foi mantido por motivos de compatibilidade. No entanto, é uma má ideia alterar seu valor para qualquer coisa que seja uma data real que esteja dentro do intervalo do controle.

Cocoa

Algumas variáveis de configuração global foram movidas de `CocoaInt` para `CocoaConfig`, como `CocoaBasePPI`, `CocoaIconUse`, `CocoaToggleBezel`, `CocoaToggleType` etc.

GTK3

O GTK3 não é mais suportado em Linux anteriores, como o Ubuntu 20.04. Ele requer GTK >= 3.24.24 e Glib2.0 >= 2.66

LAZUTILS

`Mask unit`

A unidade de máscaras foi completamente reescrita.

Motivos:
velocidade: o antigo método `Matches`() tinha características O(n^2) ou mesmo O(n^3).
controle aprimorado sobre como a máscara é interpretada.
Foram adicionados novos tipos (*para parâmetros*) e uma classe `TMaskWindows` dedicada.
`TMask.MatchesWindowsMask` e o antigo tipo `TMaskOptions` foram descontinuados e serão removidos na próxima versão.

Range e Sets

A antiga implementação de máscaras suportava sets, mas não ranges.
A nova implementação é compatível com **sets ([abc])** e **ranges ([a-c])**.
Como consequência, um `'-'` dentro de uma construção desse tipo agora é interpretado como parte da definição de intervalo e não como um `'-'` literal.
Motivo: é bom ter intervalos por padrão (a *implementação antiga simplesmente não tinha isso*). Decidimos que é um pequeno preço a pagar.
Solução:
escape o `'-'` com `EscapeChar` (*que tem como padrão* `'\'`) ou exclua `mocRange` do parâmetro `TMaskOpcodes`.

Os construtores não falham mais em uma máscara inválida

- Ao fornecer uma máscara inválida para os antigos construtores `T(Windows)Mask(List)` era gerada uma exceção.
- Os novos construtores não geram uma exceção nesse caso.
  Em vez disso, é gerada uma exceção quando `Matches`() é chamado.
- **Motivo**: não é muito agradável que um construtor falhe.

MUDANÇAS QUE AFETAM COMPATIBILIDADE CONTINUAÇÃO 2

Unidade de traduções
Adicionada a função `GetLanguageID`.
Ela retorna um registro com o código do idioma (em ISO 639-1 ou ISO 639-2) e o código do país (em ISO 3166) para a localidade atual do sistema.
Adição da função `GetLanguageIDFromLocaleName`.
Ela analisa o nome da localidade Unix e retorna um registro com o código do idioma e o código do país.
É útil para analisar identificadores de idioma passados, por exemplo, por meio de parâmetros de linha de comando.
A implementação é baseada no procedimento `GetLanguageIDs` da unidade `GetText`, mas foi reescrita para ter as seguintes propriedades:

- Os códigos de idioma e país são sempre retornados em formatos ISO no Windows.
- O identificador de localidade Unix é analisado corretamente e os códigos de idioma/país são extraídos corretamente.
- Não presuma que o código do idioma seja sempre de duas letras (ISO 639-1), ele pode ter um comprimento maior (por exemplo, três letras, como na ISO 639-2).
- O ID da localidade é retornado em um tipo de registro.

Isso permitirá retornar campos adicionais de forma compatível com as versões anteriores no futuro. Atualmente, ele contém o código do idioma, o código do país e a ID do idioma (combinação de código do idioma e código do país).

Essas funções são usadas agora em toda a base de código do Lazarus. Isso melhora muito a detecção automática de idioma e o carregamento de traduções corretas pelo Lazarus:

- Os identificadores de idioma de três letras (ISO 639-2) não são mais truncados em duas letras. Assim, as traduções para esses idiomas serão carregadas corretamente quando disponíveis.
- Anteriormente, no Windows, alguns códigos de idioma e país eram obtidos em formato não ISO, o que impedia o carregamento correto de algumas traduções, por exemplo, chinês (zh_CN).
- No Unix, as traduções com códigos de país, como português do Brasil (pt_BR) ou chinês (zh_CN) são carregadas corretamente agora.
- O macOS agora é tratado como qualquer outro Unix.

Isso remove a dependência da lista de idiomas no pacote do Lazarus (que tinha de ser mantida manualmente) e, portanto, corrige o carregamento de traduções em tcheco, húngaro, português brasileiro e ucraniano.
Unidade LazUTF8

- LazGetLanguageIDs obsoleto (retorna a combinação de códigos de idioma e país) e LazGetShortLanguageID (retorna apenas o código do idioma).

Motivo: essa funcionalidade pertence à unidade Translations (as chamadas desses procedimentos são quase sempre seguidas por chamadas a procedimentos da unidade Translations). e esses procedimentos são agora invólucros finos da função GetLanguageID da unidade Translations.
Solução: use a função GetLanguageID da unidade Translations.

MUDANÇAS QUE AFETAM COMPATIBILIDADE CONTINUAÇÃO 3

Unidades LazUTF8Classes e LazUTF8SysUtils
Tudo nessas unidades estava obsoleto há muito tempo e agora elas foram removidas.
`LazUTF8SysUtils` foi renomeado anteriormente para `LazSysUtils` e essa versão obsoleta foi deixada por um período de trânsito.

A `classe TStringListUTF8` pode ser substituída por `TStringList`.
A `classe TMemoryStreamUTF8` pode ser substituída por `TMemoryStream`.
O `procedure global LoadStringsFromFileUTF8` pode ser substituído por `TStrings.LoadFromFile`.
O `procedure global SaveStringsToFileUTF8` pode ser substituído por `TStrings.SaveToFile`.
As funções `NowUTC` e `GetTickCount64` podem ser encontradas em **LazSysUtils**.

INCOMPATIBILIDADE DE COMPONENTES

LazControls
Classes derivadas do `TSpinEditExBase`
Todas as classes derivadas do `TSpinEditExBase` devem implementar um método `SameValue`.
Esse método é definido como um método abstrato no `TSpinEditExBase`.
Motivo: Todas as classes derivadas usavam Math.SameValue.
Isso é incorreto para comparar tipos inteiros
(*mesmo que seja seguro ao comparar valores relativamente pequenos*).
Solução: infelizmente, você terá que ajustar seu código.

TFloatSpinEditEx
A propriedade **`NumbersOnly`** não é mais publicada.
Motivo: a propriedade não faz sentido para esse controle e apenas confunde os usuários.
Solução: se você realmente precisar que `NumbersOnly` seja `True`, deverá defini-la no código.

FpVectorial
O elemento Size no registro `FPVectorial TvFont` agora é um valor de ponto flutuante (`type double`).
Motivo: Evita erros de arredondamento porque as coordenadas do desenho já são duplas.
Solução: É raro que essa alteração tenha efeito no código do usuário.
Somente quando o tamanho da fonte for armazenado em uma variável, ele deverá ser declarado como `double` em vez de `integer`.

TurboPower_ipro
As declarações de tipo e os nós html foram movidos da **`unit IpHtml`** para unidades separadas, `IpHtmlTypes, IpHtmlClasses` e `IpHtmlNodes`.
Isso pode interromper a compilação de projetos existentes.
Motivo: melhorar a capacidade de manutenção da unidade IpHtml, que é extremamente longa
Solução: Adicionar `IpHtmlTypes, IpHtmlClasses` e/ou `IpHtmlNodes` à cláusula uses da(s) `unit(s)` do projeto quando um erro de **"identifier not found"** referente a esse pacote for relatado pelo compilador.

DRAWING TURTLE FIGURES IN LAZARUS

PROF DR HANS ZANTEMA
TECHNISCHE UNIVERSITEIT EINDHOVEN / RADBOUD UNIVERSITEIT NIJMEGEN

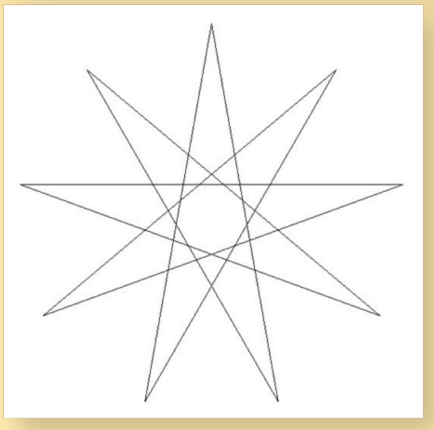
RESUMO
Neste artigo, apresentamos exemplos de figuras incríveis de tartarugas criadas pelo Lazarus, usando apenas algumas linhas de código.

DEFINIÇÃO DE FIGURA DE TARTARUGA
Para uma sequência (finita ou infinita) de zeros e uns, e dois ângulos h0 e h1,
a figura da tartaruga correspondente é definida como o caminho seguido por um robô que executa os seguintes comandos, um a um, para todos os elementos da sequência:

- Se o símbolo for 0, a direção de caminhada do robô gira o ângulo h0
- Se o símbolo for 1, a direção de caminhada do robô gira o ângulo h1
- Em ambos os casos, o robô dá um passo à frente, de alguma unidade de comprimento

Tradicionalmente, um robô com esses comandos básicos de girar e andar para frente, é chamado de tartaruga. Isso remonta à programação LOGO na década de 1960.

UM PRIMEIRO EXEMPLO
Como primeiro exemplo, considere a sequência infinita 00000... composta apenas de zeros, e escolha h0 = 160º .
Como a sequência não contém uns, o valor de h1 não tem importância.
A figura da tartaruga consiste em um número infinito de segmentos de comprimento unitário, cada um obtido obtidos após girar 160º , fazendo sempre um ângulo agudo de 20o .
Depois de fazer isso 9 vezes, no total a tartaruga girou 9 x 160º, o que é um múltiplo de 360º, e a tartaruga está de volta ao seu ponto de partida original.
Em todas as etapas seguintes, apenas os segmentos que foram desenhados anteriormente são sobrescritos. Portanto, para essa sequência infinita, a figura da tartaruga resultante é finita e tem a seguinte aparência:



DESENHO DE FIGURAS DE TARTARUGAS NO LAZARUS
Para desenhar essa figura de tartaruga no Lazarus, para cada etapa, calculamos o novo local (x,y) da tartaruga e usamos o comando lineto para desenhar a linha até esse novo local.
Além das variáveis x e y, precisamos de uma variável h para representar a direção real da tartaruga, uma variável u para representar o comprimento da unidade e uma variável n para representar o número de etapas a serem executadas. As variáveis x e y precisam ser reais e devem ser arredondadas para números inteiros para que o comando lineto possa ser aplicado.

Para o exemplo que acabamos de dar, após a inicialização de todas essas variáveis em um ambiente de tela, isso pode ser lido da seguinte forma:

```
 moveto(round(x),round(y));
for i := 1 to n do
       begin
       h := h + 160;
       x := x + u * cos(h);
       y := y + u * sin(h);
       lineto(round(x),round(y));
       end;
```

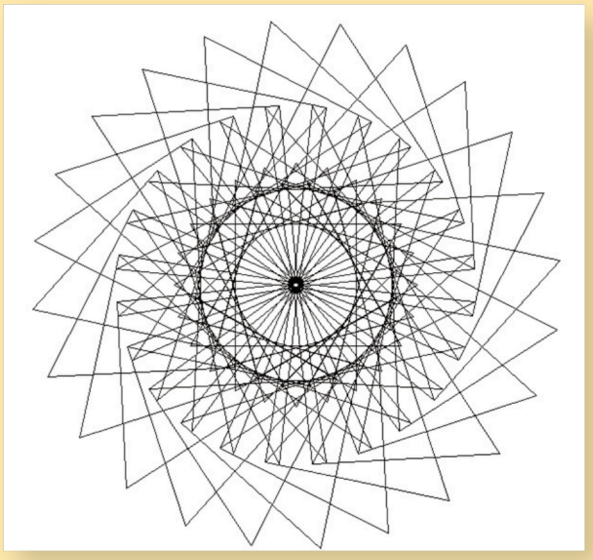
Para este exemplo, n deve ser escolhido como qualquer número com pelo menos 9.
Para sequências que contêm 0 e 1, a parte inicial da sequência de comprimento n é armazenada em uma matriz a e, no loop for acima, o ajuste da variável h é substituído por

```
if a[i] = 0 then h := h + h0 else h := h + h1;
```

SEQUÊNCIAS PERIÓDICAS
Uma sequência é chamada de periódica se consistir em um número infinito de cópias da mesma sequência finita.
Por exemplo, uma parte inicial de uma sequência periódica é obtida por

```
    for i := 1 to n do
        if (i mod 9) in [0,2,3,4] then a[i] := 0 else a[i] := 1;
```

e escolhendo h0 = 120 e h1 = -147, obtém-se a seguinte figura de tartaruga:



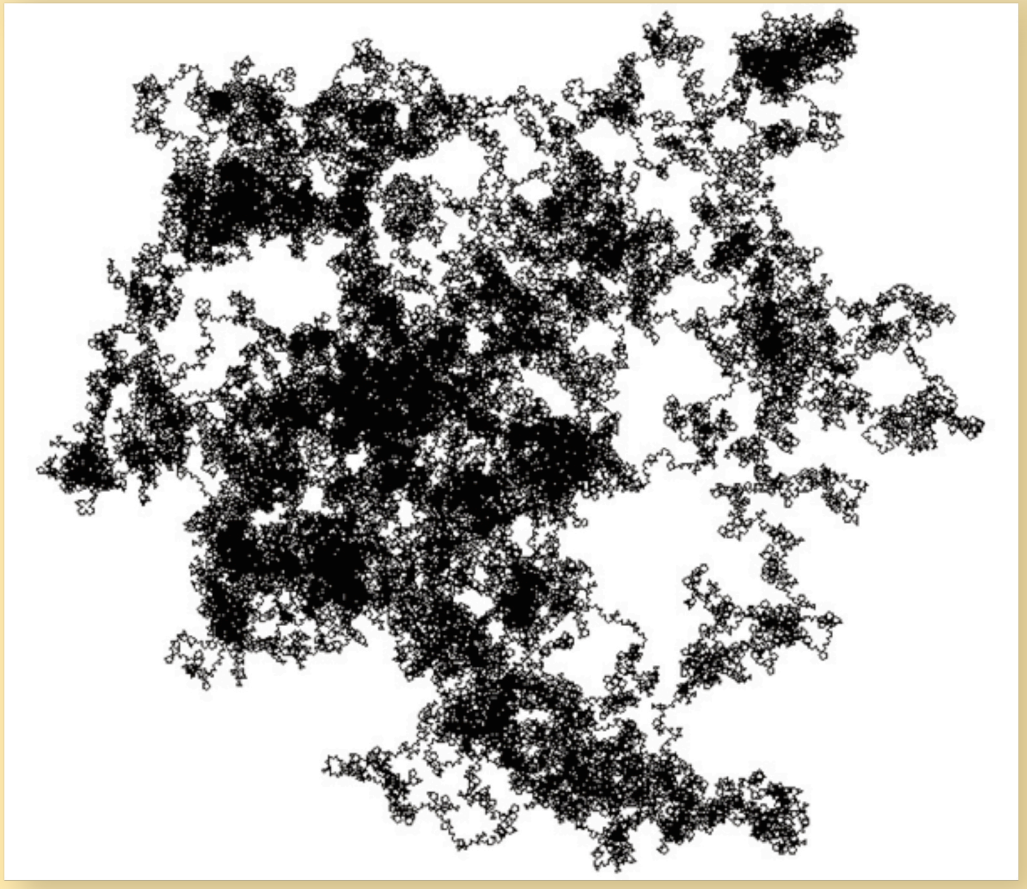
Aqui, a figura da tartaruga da sequência finita copiada 100011110 deve ser desenhada 9 vezes para voltar ao ponto e ao ângulo iniciais, portanto, n deve ser escolhido pelo menos 216.

SEQUÊNCIAS MAIS COMPLICADAS

As sequências periódicas são muito estruturadas. As sequências sem nenhuma estrutura são obtidas escolhendo-se aleatoriamente a[i], por exemplo, por

```
for i := 1 to n do a[i] := random(2);
```

Como era de se esperar, a figura da tartaruga resultante não mostra nenhuma estrutura e, normalmente, tem a seguinte aparência:



A SEQUÊNCIA THUE-MORSE

Figuras de tartarugas mais interessantes são obtidas a partir de sequências mórficas, definidas pela propriedade de que elas se produzem aplicando um determinado morfismo f, ou seja, mapeando cada 0 para uma sequência finita f(0) e cada 1 para uma sequência finita f(1). Por exemplo, a sequência de Thue-Morse

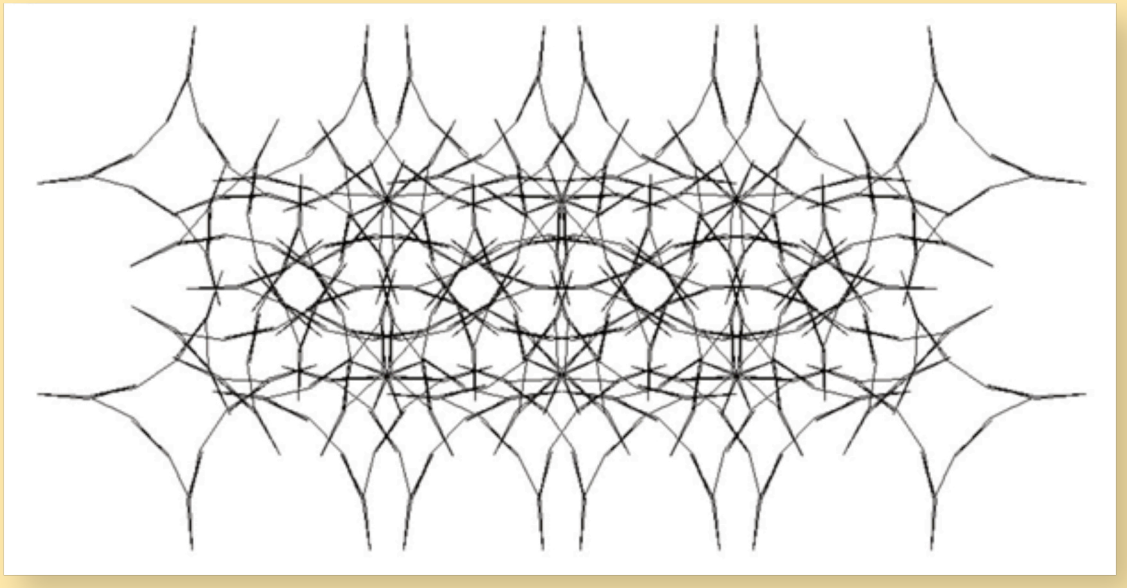
0110100110010110.....

é definida como a sequência mórfica que começa em 0 e se transforma em si mesma substituindo cada 0 por f(0) = 01 e cada 1 por f(1) = 10.

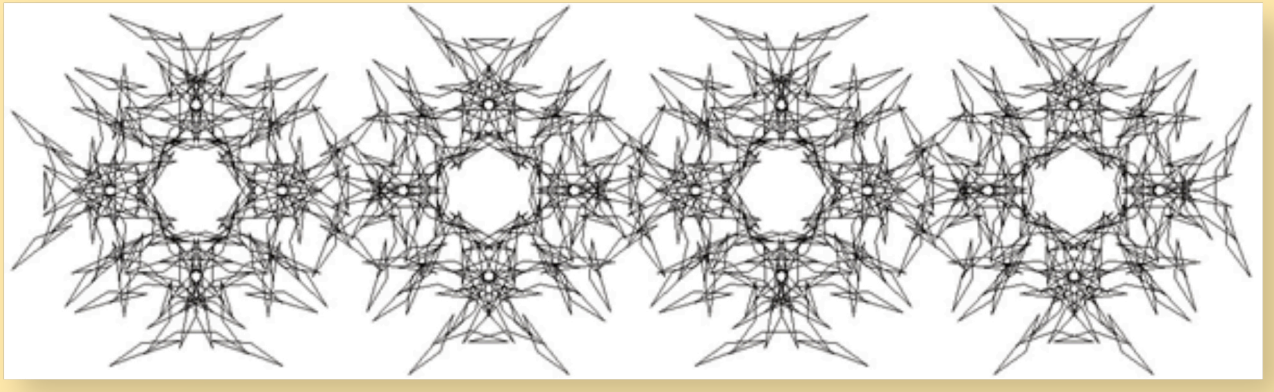
Pode-se argumentar (para a teoria, consulte [1,2]) que se 2kh0 e 2kh1 forem múltiplos de 360 graus, para algum k, então a figura da tartaruga da sequência de Thue-Morse será finita.

Acontece que, muitas vezes, eles produzem belas imagens. Por exemplo, escolhendo h0 = 22,5 = 360/16

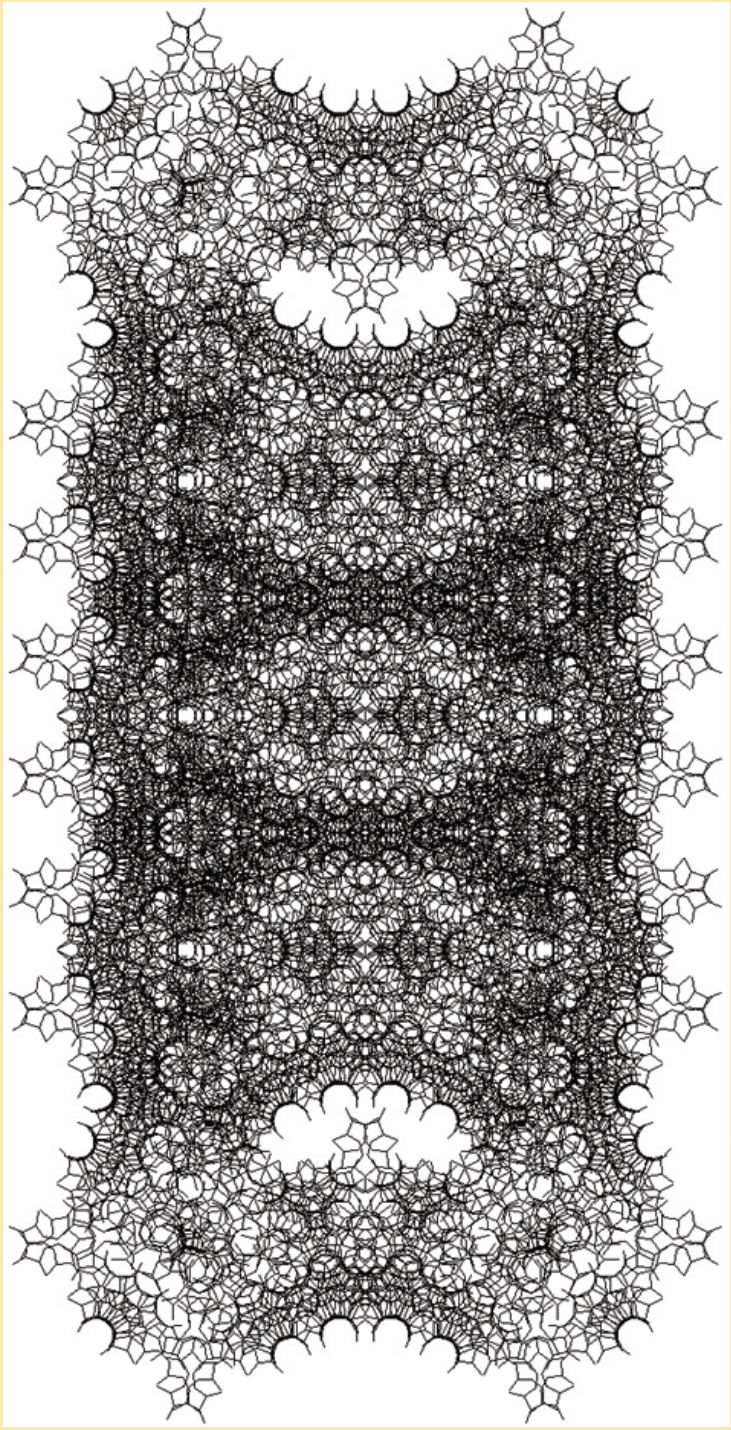
e h1 = 177,1875 = 63 x 360/128 produz a seguinte figura de tartaruga:



A escolha de outros ângulos que também satisfaçam essa condição produz as seguintes figuras de tartaruga:



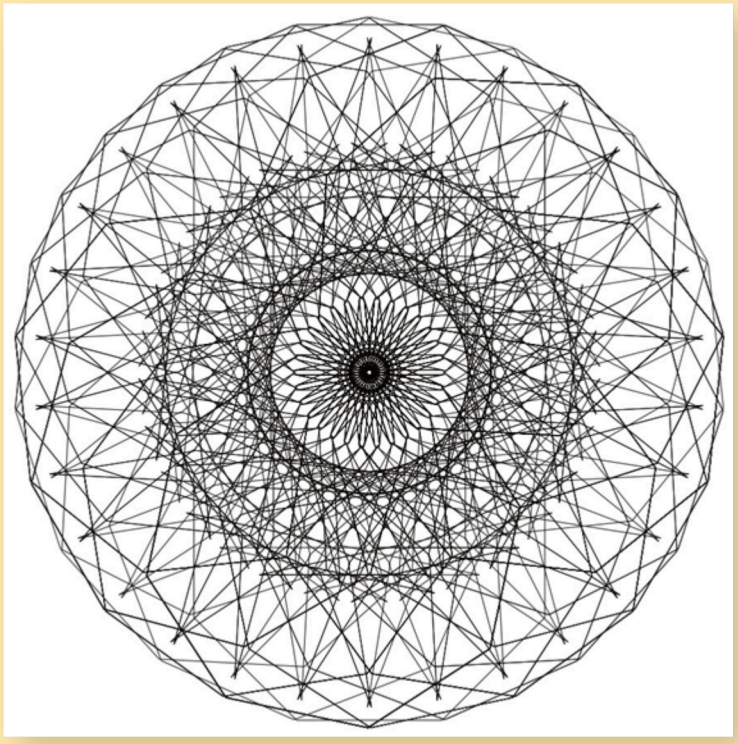
e → (veja a próxima página)

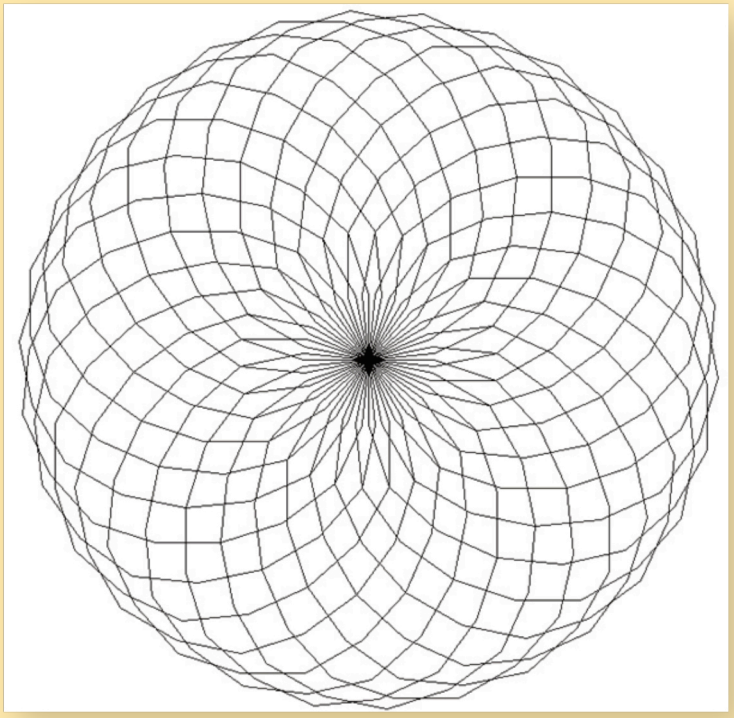


Para todas essas figuras de tartaruga da sequência de Thue-Morse, o número total de segmentos desenhados é uma potência de 2, para esses três exemplos, respectivamente 210 = 1024, 211 = 2048 e
216 = 65536. Para todos esses exemplos, o número n no programa deve ser, no mínimo, o número correspondente. A geração da parte inicial de qualquer sequência mórfica pode ser feita com apenas algumas linhas de código. Por exemplo, para os exemplos acima, o código completo para desenhar a curva da tartaruga pode ser o seguinte:

```
a[1]:= 0;
a[2]:= 1;
i   := 2;
j   := 1;
while i < n do
  begin
    i := i+1; j := j+1;
    if a[j] = 0
    then
      begin
        a[i]:= 0;
        i   :=i+1;
        a[i]:= 1;
      end
    else
      begin
        a[i]:= 1;
        i   :=i+1;
        a[i]:= 0;
      end;
  end;
for i := 1 to n do
  begin
    if a[i]  := 0
    then h   := h+h0
    else h   := h+h1;
    x := x + u * cos(h);
    y := y + u * sin(h);
    lineto(round(x),round(y));
  end;
```

Todos os três exemplos que demos (e muitos outros) são obtidos por diferentes inicializações das variáveis n, h0, h1, h, x, y e u. Essencialmente, a imagem é obtida pela escolha de n, h0 e h1; os valores iniciais de h, x, y e u são normalmente escolhidos depois de brincar com esses valores até que a imagem resultante se encaixe perfeitamente na tela.





OUTRAS SEQUÊNCIAS MÓRFICAS

Ao escolher outras sequências finitas f(0) e f(1), nas quais f(0) deve começar em 0 e consistir em pelo menos dois símbolos, obtemos outras sequências mórficas, dando origem a muitas outras figuras de tartarugas interessantes. Algumas delas são finitas, assim como os exemplos que demos para a sequência de Thue-Morse.

Novamente, para conhecer a teoria subjacente, consulte [1,2]. Como primeiro exemplo, consideramos f definida por f(0) = 010 e f(1) = 11, produzindo a sequência mórfica

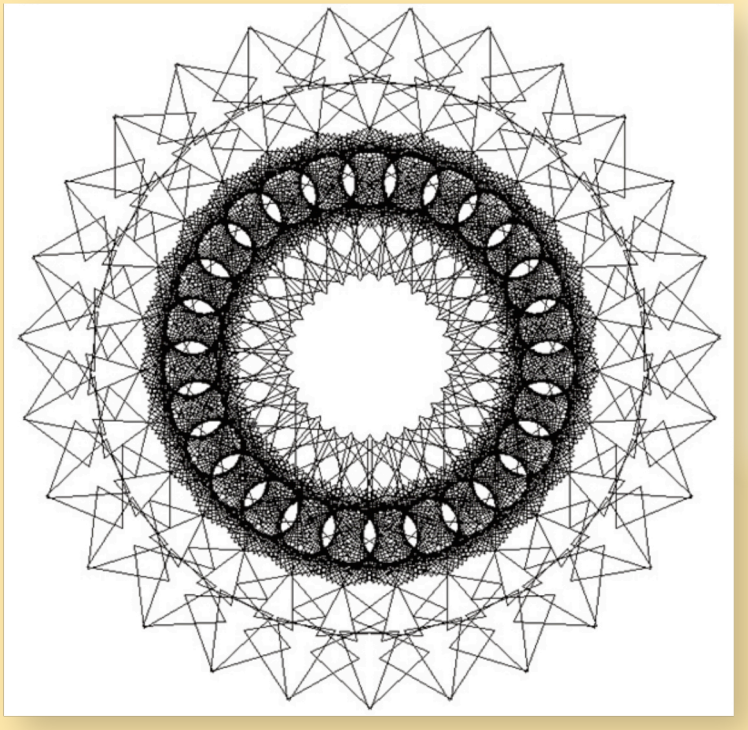
0101101011101011010....

sendo a única sequência que começa em 0 e mapeia para si mesma quando simultaneamente todo 0 é substituído por f(0) = 010 e todo 1 por f(1) = 11.

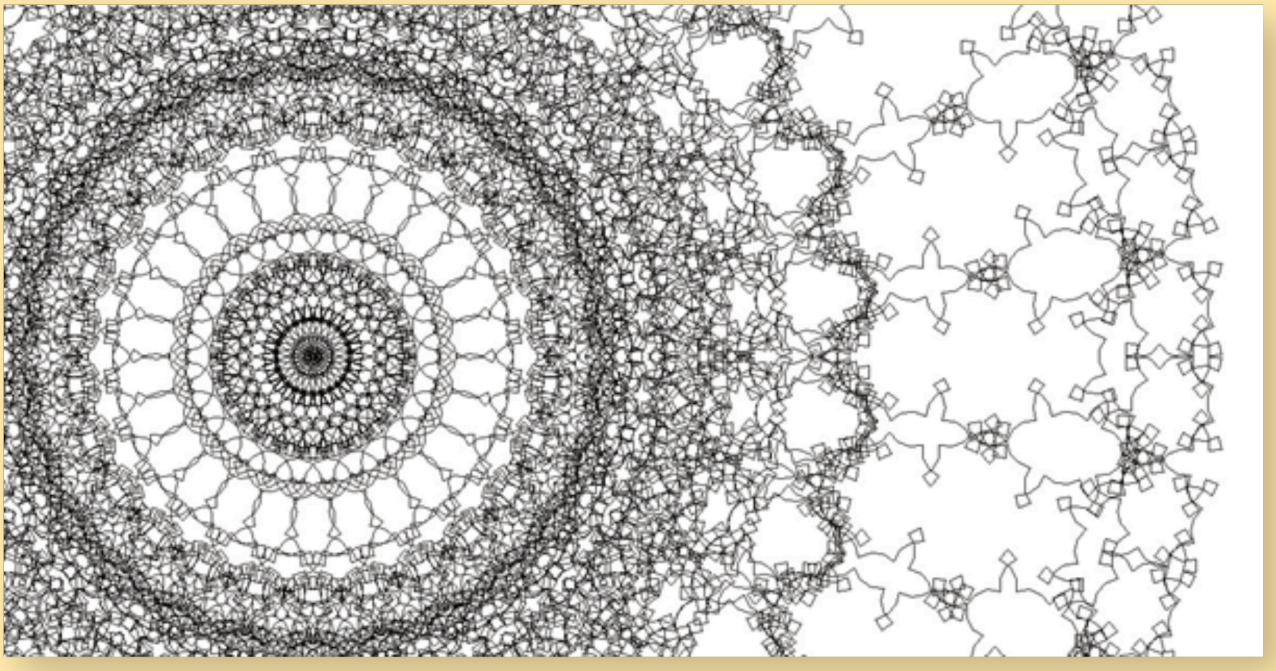
Escolhendo h0 = 168o e h1 = -45o e n suficientemente grande, obtém-se a figura da tartaruga mostrada no topo esquerdo desta página. Mais precisamente, essa figura da tartaruga consiste em 840 segmentos.

Ao escolher n = 840, nem todos os segmentos serão desenhados, pois alguns segmentos já estão duplamente desenhados, mas ao escolher n = 1.300, o senhor terá a figura completa: a continuação apenas desenhará por cima dos segmentos existentes.

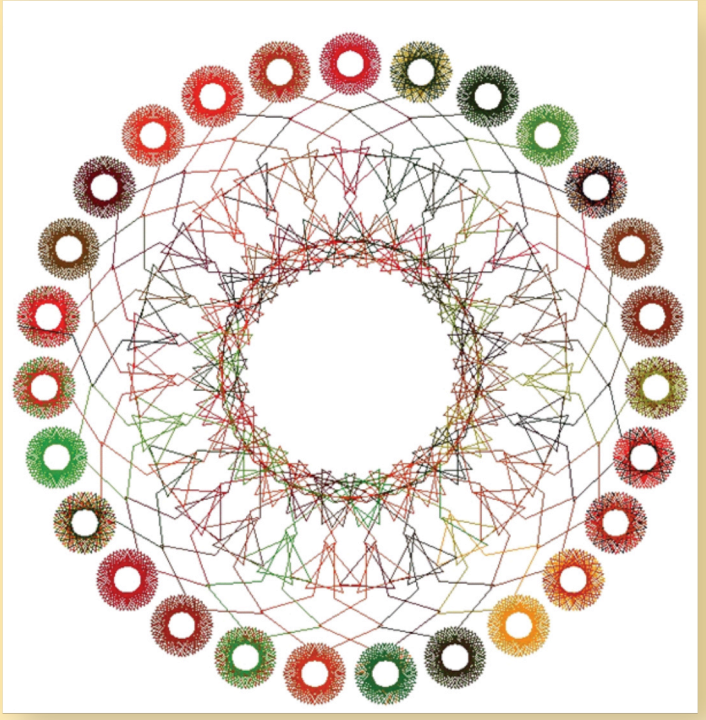
Outras opções de f(0), f(1), h0 e h1 produzem as figuras de tartaruga mostradas à esquerda abaixo e na parte superior da próxima página.



Para todos esses exemplos, o programa é apenas uma pequena modificação do programa que fornecemos para a sequência de Thue-Morse: apenas o primeiro loop é modificado para gerar a sequência mórfica desejada.
Além disso, os ângulos h0 e h1 devem ser escolhidos de modo que os requisitos para os teoremas que concluem a finitude de [1], [2] sejam atendidos,
e os valores iniciais de h, x, y e u devem ser escolhidos de forma que a figura resultante se encaixe perfeitamente na tela.
Se o senhor escolher u muito grande de propósito, a figura resultante será ampliada, como no exemplo mostrado abaixo.



Um programa Lazarus com uma interface de usuário para inserir todos os parâmetros para criar todas as figuras de tartaruga que fornecemos, incluindo todas as fontes e vários exemplos, está disponível em `https://github.com/hzantema/turtle-graphics`
Até agora, a cor da caneta não foi especificada, sendo que o padrão é preto. Ao modificar a cor da caneta no segundo loop para aumentar i, é possível obter figuras como as seguintes:



Em todos esses exemplos, cada segmento é desenhado na tela pelo comando
lineto(round(x),round(y)); Para mostrar apenas a imagem na tela, isso funciona bem, mas não oferece alta resolução devido ao arredondamento.
Para imprimir em alta resolução ou oferecer o recurso de zoom, deve-se mudar dessa abordagem de bitmap para uma abordagem baseada em vetor, simplesmente removendo os comandos de arredondamento e substituindo o comando lineto pelo comando correspondente em uma configuração baseada em vetor. Então, uma compilação de exemplos como a apresentada neste documento pode ter a seguinte aparência:
(veja a figura na página 9 do artigo)

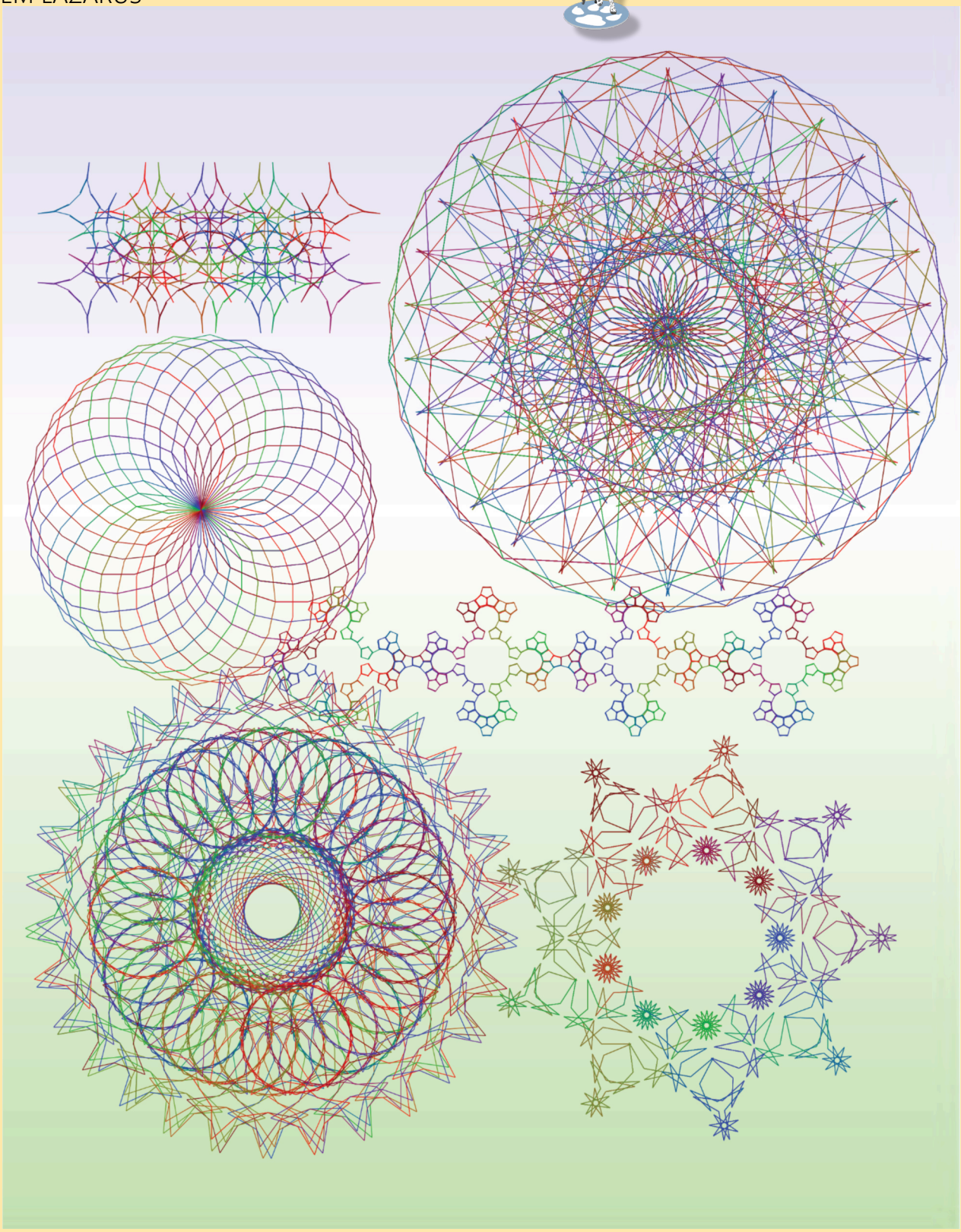
CONCLUSÃO

Neste artigo, mostramos como várias figuras notáveis podem ser desenhadas por um programa Lazarus disponível gratuitamente, no qual o código subjacente tem um tamanho muito limitado.

REFERÊNCIAS

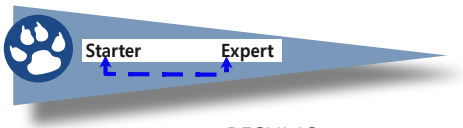
[1] H. Zantema, Turtle graphics of morphic sequences (Gráficos de tartaruga de sequências mórficas), Fractals, 2016, volume 24, número 1, versão preliminar disponível em https://www.win.tue.nl/~hzantema/turtle.pdf

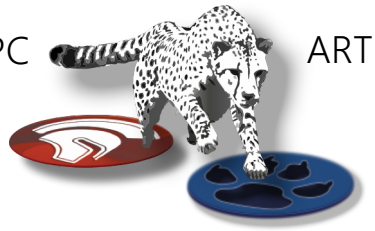
2] H. Zantema, Playing with infinity: turtles, patterns, and pictures [Brincando com o infinito: tartarugas, padrões e imagens]. A ser publicado em 2024 na CRC Press, grupo Taylor and Francis, cerca de 250 páginas. Traduzido da versão holandesa Spelen met oneindigheid: verrassende figuren en patronen, Noordboek 2023



ACOMPANHANDO O DELPHI NO FPC

POR MICHAEL VAN CANNEYT





RESUMO
A compatibilidade com o **Delphi** sempre foi importante para o Free Pascal.
No entanto, em certos aspectos, essa compatibilidade estava faltando.
Recentemente, muito trabalho foi feito para melhorar a compatibilidade do **FPC** com o **Delphi**.

① INTRODUÇÃO

A equipe do compilador Free Pascal sempre considerou a compatibilidade com o Delphi como um recurso importante do compilador Free Pascal.
Essa compatibilidade se aplica primeiramente e principalmente à linguagem **Pascal**: se o **Delphi** pode compilá-la, então o **Free Pascal** também deve. Por muito tempo, o **Delphi 7** foi a versão com a qual o Free Pascal era mais compatível, mas, obviamente, o **Delphi** evoluiu:
Foram adicionados recursos de linguagem moderna, como **Generics**, funções anônimas, atributos e **RTTI** estendido.

Esses recursos foram desenvolvidos no FPC - alguns há mais tempo, outros recentemente, mas muitos desses recursos ainda não foram incorporados em uma versão oficialmente lançada do **Free Pascal**. Para usá-los, você deve usar a versão de desenvolvimento do **FPC**.

A compatibilidade com o **Delphi** também se aplica às unidades básicas que são distribuídas com o **Free Pascal:**
As unidades **RTL** básicas do **Delphi** também podem ser encontradas no **Free Pascal**.
É claro que o **Delphi** e o **Free Pascal** evoluem ao longo do tempo, e algumas das diferenças nas unidades **RTL** dificultam a manutenção do código funcionando tanto no **Delphi** quanto no Free Pascal.

- unidades **RTL** tornam difícil manter o código funcionando tanto no **Free Pascal** quanto no Delphi: Algumas unidades básicas não estão presentes. O número de unidades na **RTL** do Delphi se expandiu consideravelmente. A funcionalidade básica é oferecida em `System.IOUtils`, `System.JSON`, `System.Threading` etc.
  Essas unidades foram adicionadas ao **FPC** recentemente.
- O **Delphi** passou a usar unidades pontilhadas (**namespaced**):
  Os prefixos `System`, `Data`, `Vcl`, `FMX`
  são usados em toda a base de código do **Delphi** há muitos anos.
- O Delphi mudou há muitos anos o tipo de alias 'String' para que ele seja um alias para **UnicodeString**: um tipo de cadeia de caracteres em que cada elemento da cadeia (*um caractere*) é um caractere **UTF-16 - ele usa 2 bytes**.

Não é preciso dizer que a equipe do **Free Pascal** sempre precisa se atualizar em relação ao **Delphi** - o requisito de compatibilidade é uma via de mão única.
Isso representou um dilema para a equipe do FPC:
Em primeiro lugar, o **FPC deseja manter a compatibilidade retroativa com ele mesmo.**
Algo que é considerado igualmente (se não mais) importante do que a compatibilidade com o **Delphi**. Claramente, mudar o tipo de string e começar a usar namespaces nos nomes das unidades torna o código incompatível com as versões anteriores.

COMO RESOLVER ISSO?

② A SOLUÇÃO

Depois de muitos anos, uma solução para esse dilema foi implementada.
A ideia é simples: usar a mesma base de código para recompilar todo o código **RTL e de pacotes** do **FPC** duas vezes.

- Uma compilação é feita com configurações que são **compatíveis com as versões anteriores** do próprio FPC: sem nomes pontilhados, string é a string de 1 byte.
- Uma segunda compilação é feita com configurações que geram nomes com pontos, e em que string é a string de 2 bytes.

Isso resulta em dois conjuntos de unidades, que não podem ser misturados. O usuário precisa escolher qual conjunto de unidades ele deseja usar:

- **O conjunto de unidades compatível com versões anteriores do Free Pascal,**
- **O conjunto de unidades compatível com o "Delphi recente"**.

O último foi apelidado de "unicode rtl".
Há uma terceira opção, que é compilar tudo com um conjunto pessoal de configurações: mais sobre isso mais tarde.
Para colocar essa solução em prática, foi necessário algum trabalho no compilador:

- O compilador teve que mudar sua visão sobre o que constitui o tipo `Char` básico - Até recentemente, ele era codificado como um caractere de 1 byte e `AnsiChar` era um alias para `Char`.

Agora, `AnsiChar` é um tipo básico e `Char` é um alias definido na unidade do sistema.
Isso permite que a palavra-chave (**Keyword**) `String` seja uma `AnsiString` ou uma `UnicodeString` e `Char` sempre corresponderá ao tipo de um único caractere em uma `String`.

- O "destino" de um compilador era, até recentemente, definido como uma combinação da CPU de destino e o **sistema operacional de destino**.

Isso significa que, por exemplo, o **i386-win32** e o **x86 64-win64** são duas plataformas Windows: uma de **32 bits e outra de 64 bits**.
O Linux conhece ainda mais plataformas, assim como o Darwin (macos) já tem 3 plataformas. Essa definição de plataforma precisava ser ampliada, e a noção de "SubTarget foi introduzida no compilador.

Ambas as alterações foram fundamentais para possibilitar a criação dos 2 conjuntos de unidades para o usuário final.

③ O SUBALVO

A noção de um alvo de compilador foi ampliada: ele precisa encapsular uma CPU, um sistema operacional e, opcionalmente, também um conjunto arbitrário de configurações. Para o rtl unicode compatível com Delphi, esse "conjunto arbitrário de configurações" seria:

- usar strings **UTF16**.
- Usar nomes "**dotted**" (com pontos).

Definir um subtarget significa simplesmente dar um nome a esse conjunto de configurações. O subtarget é sempre especificado para o compilador com uma opção de linha de comando: -t.
Portanto, se você quisesse compilar um arquivo para o subtarget '**unicodertl**', especificaria:

```
fpc -Mdelphi -tunicodetrl myproject.pas
```

O nome do subtarget é então usado de várias maneiras: A primeira maneira pela qual o nome do subtarget é usado é no compilador. Quando o compilador está compilando para um determinado destino, ele define uma macro fpctarget. O valor dessa macro é a combinação da CPU e do sistema operacional de destino, separados por um traço.

Continuação Capítulo 3

Essa macro é usada no arquivo de configuração do compilador:

```
-Fu/usr/local/lib/fpc/$fpcversion/units/$fpctarget
-Fu/usr/local/lib/fpc/$fpcversion/units/$fpctarget/*
-Fu/usr/local/lib/fpc/$fpcversion/units/$fpctarget/rtl
```

Portanto, quando o compilador versão 3.1.1 está compilando para i386-linux, o compilador "vê" a seguinte configuração:

```
-Fu/usr/local/lib/fpc/3.3.1/units/i386-linux
-Fu/usr/local/lib/fpc/3.3.1/units/i386-linuxt/*
-Fu/usr/local/lib/fpc/3.3.1/units/i386-linux/rtl
```

Agora, a opção "Subtarget" é introduzida, e um dos efeitos de seu uso é que ele adiciona o nome do subtarget à macro fpctarget. Isso significa que quando o compilador versão 3.1.1 estiver compilando para i386-linux e o subtarget 'unicodertl', a configuração se torna

```
-Fu/usr/local/lib/fpc/3.3.1/units/i386-linux-unicodertl
-Fu/usr/local/lib/fpc/3.3.1/units/i386-linuxt-unicodertl/*
-Fu/usr/local/lib/fpc/3.3.1/units/i386-linux-unicodertl/rtl
```

Os caminhos de pesquisa do compilador são subitamente diferentes, dependendo do subtarget. Como dito anteriormente, o subtarget é um nome de um conjunto de configurações. Isso é concretizado pelo segundo efeito do uso do subtarget, e explica por que ele precisa ser definido na linha de comando:

Ao procurar a configuração do compilador, o compilador também procurará um arquivo de configuração que tenha o subtarget incluído em seu nome.
Ao compilar com o subtarget `unicodertl`

```
fpc -Delphi -tunicodetrl myproject.pas
```

O compilador, nos mesmos locais em que procura por fpc.cfg, também procurará por um arquivo chamado

```
fpc-unicodertl.cfg
```

Em plataformas do tipo Unix, ele também procurará o arquivo "oculto" usual no diretório inicial do usuário:

```
.fpc-unicodertl.cfg
```

Esse segundo arquivo de configuração contém o conjunto de configurações que define o subtarget. Há dois aspectos a serem observados sobre esse arquivo de configuração extra:

❶ Ele deve existir. Se não existir, o compilador exibirá um erro (*no entanto, ele pode estar vazio*).
❷ É sempre carregado, mesmo que a opção para não carregar os arquivos de configuração padrão `(-n)` for especificada.

Então, como isso é usado para criar um rtl compatível com o **Delphi**?
Um arquivo de configuração com o nome **`fpc-unicodertl.cfg`** é criado com o seguinte conteúdo:

```
-dUNICODERTL
-Municodestrings
```

Esse arquivo de configuração é então usado para compilar a **RTL** e os pacotes.
Ele define **`UNICODERTL`** e especifica o modeswitch `unicodestrings`, que instrui o compilador que a String deve ser interpretada como `UnicodeString`.

É claro que muitas unidades contêm algum código de baixo nível em que o tamanho do tipo **`Char`** é muito importante.

Os códigos-fonte foram alterados quando necessário para que o código compile e funcione corretamente com ambas as definições dos tipos **`String`** e **`Char`**.

Continuação Capítulo 3

Não apenas os códigos-fonte foram alterados, mas também o processo de compilação e instalação foi ligeiramente modificado:
Vimos anteriormente que, ao especificar -tunicodertl no alvo **i386-linux,** o compilador procurará por unidades compiladas nos seguintes diretórios:

```
-Fu/usr/local/lib/fpc/3.3.1/units/i386-linux-unicodertl
-Fu/usr/local/lib/fpc/3.3.1/units/i386-linuxt-unicodertl/*
-Fu/usr/local/lib/fpc/3.3.1/units/i386-linux-unicodertl/rtl
```

Isso significa que as unidades também devem ser instaladas lá ao instalar as unidades compiladas.
Os makefiles para o **RTL** e os pacotes foram adaptados para atender a isso.
Agora há uma variável `SUB_TARGET` que configura os **Makefiles** para chamar o compilador com o subtarget especificado. Ao instalar unidades, o nome do subtarget será anexado ao nome do diretório nome do diretório onde as unidades estão instaladas.

④ USO DE NAMESPACES EM NOMES DE ARQUIVOS

Para criar unidades com namespaces e as mesmas unidades, mas sem namespaces, um pequeno truque é usado. Deve ficar claro que a equipe do **Free Pascal** não pode manter dois conjuntos de unidades.
A solução implementada usa um conjunto de arquivos, mas usa um mecanismo de inclusão para criar o segundo arquivo (**namespaced**). Como exemplo, aqui está o arquivo da unidade `'System.Math':`

```
unit System.Math;

{$DEFINE FPC_DOTTEDUNITS}
{$i math.pp}
```

The `math.pp` file is changed

```
{$IFNDEF FPC_DOTTEDUNITS}
unit Math;
{$ENDIF FPC_DOTTEDUNITS}

interface
uses

{$IFDEF FPC_DOTTEDUNITS}
  System.SysUtils;
{$ELSE FPC_DOTTEDUNITS}
  sysutils;
{$ENDIF FPC_DOTTEDUNITS}
```

OBSERVE QUE A CLÁUSULA USES É DIFERENTE.
Por exemplo, sempre que um identificador totalmente qualificado é usado (*ou seja, um identificador que inclui o nome da unidade*), o nome teve de ser corrigido.
Esse sistema permite que a equipe do **FPC** compile uma **RTL** com nomes de arquivos namespaced, e uma **RTL** com nomes de arquivos compatíveis com versões anteriores.
O exercício acima foi feito para todas as unidades distribuídas pelo **FPC**:
Para cada unidade, uma versão com namespaced foi implementada. Para as unidades que existem no **Delphi**, o nome de arquivo **namespaced** é idêntico ao nome no **Delphi**.

Para as unidades que não têm um equivalente no **Delphi**, foi aproveitada a oportunidade para tornar os nomes de arquivos mais consistentes.

Todos os programas makefiles e `fpmake` na distribuição do **Free Pascal** foram adaptados para que eles compilem as unidades namespaced quando a opção `FPC_DOTTEDUNITS=1` for fornecida na linha de comando **make**.

Continuação Capítulo 4

Até agora, a política da equipe do **Free Pascal tem** sido a de colocar em minúsculas os nomes de arquivos de unidades em sistemas **de arquivos que diferenciam maiúsculas de minúsculas.** sistemas de arquivos. Isso significa que você pode ser desleixado na cláusula **uses:** o caso do nome da unidade não importava, pois o compilador procurava uma versão em minúsculas de um arquivo além de procurar um arquivo com o mesmo caso usado na cláusula uses.
Essa prática foi abandonada para unidades namespaced: **O nome da unidade agora deve ter o correto em sistemas de arquivos que diferenciam maiúsculas de minúsculas**.

COMO ENCONTRAR A VERSÃO NAMESPACED DE UM NOME DE UNIDADE?
Há um arquivo que especifica, para cada unidade não namespaced, o nome namespaced da unidade.

Esse arquivo pode ser usado para alterar os nomes das unidades.
Não é necessário fazer isso manualmente, há uma ferramenta que fará isso para você. Mais informações sobre essa ferramenta mais adiante.
O leitor astuto deve ter notado que a geração de unidades namespaced não é feita usando o conceito recém-introduzido de subtargets. De fato, os dois recursos são ortogonais.
O motivo é que não existe um subtarget "especial". **Você cria quantos subtargets quiser, e, para cada subtarget, você pode criar uma versão namespaced da RTL ou uma versão compatível com as versões anteriores.**
Isso significa que seria possível gerar uma **RTL** sem unidades namespaced, mas com `String` igual a **UnicodeString**, ou uma **RTL** com namespaced com `String=AnsiString`.
A escolha da equipe do **FPC** é determinada pelo requisito de ter uma **RTL** compatível com o **FPC** e uma **RTL** compatível com o **Delphi**.

⑤ CRIAÇÃO DA RTL COMPATÍVEL COM O DELPHI

Quando a próxima versão principal do Free Pascal for lançada, a equipe do FPC criará as duas RTLs mencionadas acima, mas você já pode aproveitar essa maior compatibilidade com o Delphi hoje mesmo compilando o compilador e a RTL compatível com o Delphi.

COMO FAZER ISSO?

A primeira coisa a fazer é instalar o git e baixar os códigos-fonte do compilador. Esse mecanismo foi tratado em profundidade em artigos anteriores. Supondo que o **git** esteja instalado e em seu PATH, em uma janela de terminal (*na linha de comando*) você pode clonar os códigos-fonte do FPC:

```
git clone https://gitlab.com/freepascal.org/fpc/source.git fpc
```

Uma vez feito isso, você pode compilar e instalar o FPC mais recente:

```
cd fpc
make clean all PP=/caminho/para/FPC/3.2.2
make install PP=/caminho/para/FPC/3.2.2
```

A variável PP deve apontar para o binário fpc da versão 3.2.2 do FPC (esse é um requisito).
O local onde esse binário fpc está instalado depende de seu sistema. O procedimento acima compilará e instalará a versão 3.3.1 do FPC. Você também pode usar o FPCUpdeluxe para isso.

Quando isso for feito, você deverá observar onde a nova versão do FPC 3.3.1 foi instalada. A próxima etapa é criar o arquivo de configuração fpc-unicodertl.cfg em um dos locais onde o compilador procura o arquivo de configuração:

```
-dUNICODERTL
-Municodestrings
```

A estratégia mais fácil é procurar o fpc.cfg existente e criar o arquivo acima ao lado dele.

Continuação Capítulo 5 

Em seguida, no mesmo terminal em que os comandos anteriores foram digitados, os seguintes comandos make a seguir criarão e instalarão o rtl unicode.
Esses comandos devem ser executados no mesmo diretório que os comandos "**make**" anteriores.

```
make -C rtl clean all PP=/path/to/FPC/3.3.1 SUB_TARGET=unicodertl FPC_DOTTEDUNITS=1
make -C rtl install PP=/path/to/FPC/3.3.1 SUB_TARGET=unicodertl FPC_DOTTEDUNITS=1

make -C packages clean all PP=/path/to/FPC/3.3.1 SUB_TARGET=unicodertl FPC_DOTTEDUNITS=1
make -C packages install PP=/path/to/FPC/3.3.1 SUB_TARGET=unicodertl FPC_DOTTEDUNITS=1
```

Depois de fazer isso, você pode compilar o código Delphi recente usando a nova opção de linha de comando '`-tunicodertl`' da linha de comando.

⑥ CONVERSÃO DE CÓDIGO PARA USAR UNIDADES COM NAMESPACED

E SE VOCÊ QUISER ATUALIZAR SEU CÓDIGO E USAR O RTL COMPATÍVEL COM O DELPHI?

Conforme mostrado na *página 5 deste artigo*, a **cláusula uses** de um projeto deve ser alterada. Se você tiver muitas unidades, isso significa muito trabalho.
Felizmente, as ferramentas que foram usadas para criar as unidades de uso duplo do **FPC** estão disponíveis para você.

Na **árvore de código-fonte do FPC**, no diretório `utils/dotutils`, há um programa chamado prefixunits. Você pode compilá-lo na linha de comando ou usando o **Lazarus** e usá-lo para converter unidades para o novo mecanismo usado.

A linha de **comando a seguir** converterá a unidade myunit.pas em company.myunit.pas usando o mesmo mecanismo mostrado acima: **`prefixunits -b -k known.txt c:\Temp\myunit.pas -n company`**
Um novo arquivo varcompany.myunit.pas será gravado e incluirá myunit.pas.
A cláusula uses em myunit.pas será reescrita usando a definição condicional, conforme mostrado acima. A opção -b diz ao programa para fazer um backup; a opção -k deve ser usada para apontar para o arquivo que **mapeia a unidade antiga** os **novos nomes**.

Esse arquivo está presente no diretório dotutils, ao lado dos códigos-fonte da ferramenta prefixunits. Você também pode decidir usar apenas os novos nomes de unidades.

Nesse caso, especifique o sinalizador **`'-r'`** (*para 'substituir'*) **:**

```
prefixunits -r -b -k known.txt c:\Temp\myunit.pas -n company
```

Nesse caso, nenhum novo arquivo é criado e a cláusula uses em myunit.pas será substituída por uma cláusula de unidade que usa apenas os nomes de unidade pontilhados.

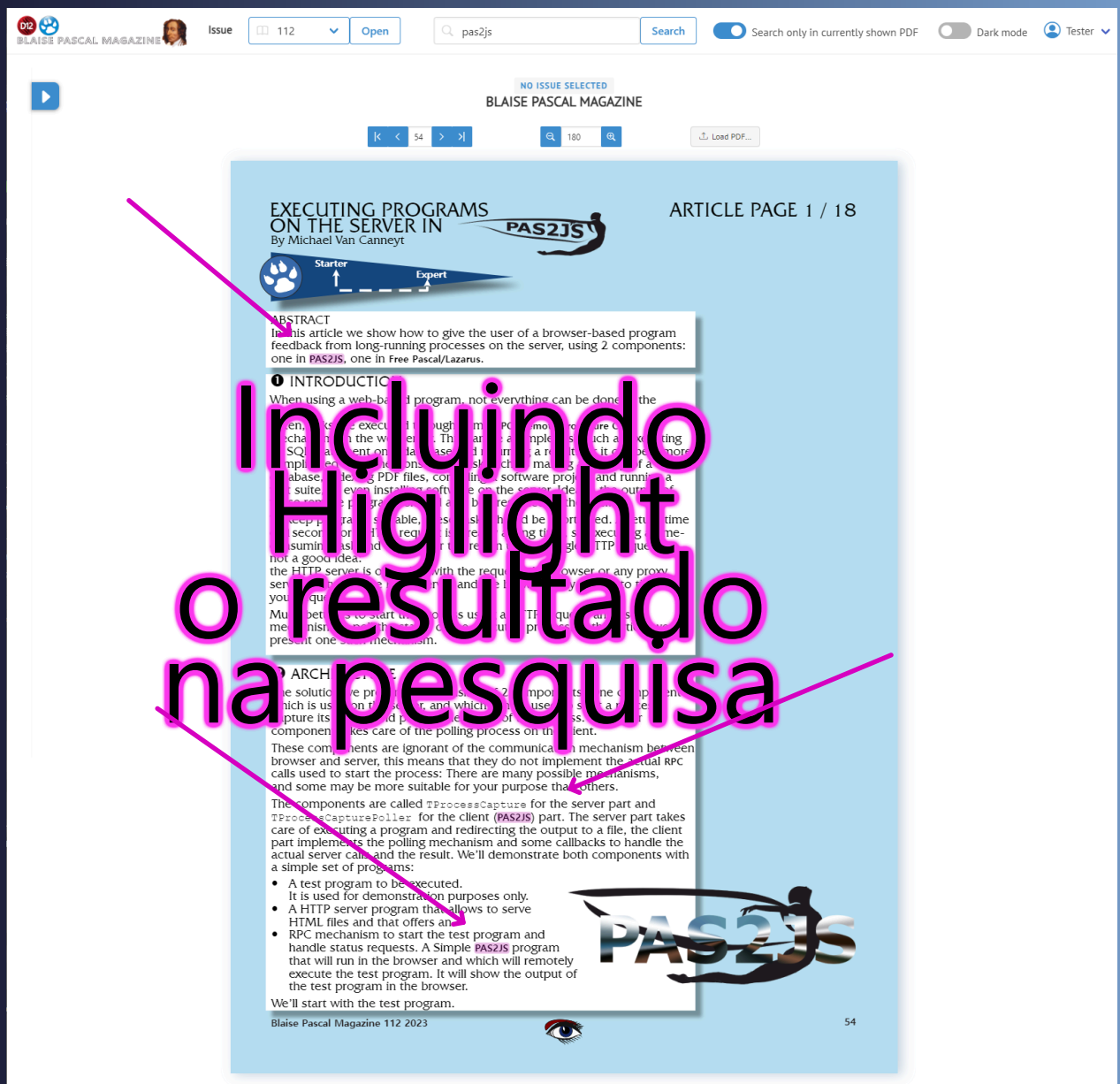
⑦ E QUANTO AO PAS2JS?

Para o PAS2JS, a mesma operação de namespacing foi realizada.
A opção -t também foi adicionada às opções de linha de comando do transpilador, portanto, o funcionamento dos dois compiladores é o mesmo.Para melhorar a compatibilidade com o Delphi, **foi adicionado o suporte à string de várias linhas do Delphi 12**, além do suporte à string de **várias linhas já existente no além do suporte já existente** à string de várias linhas no 

⑧ CONCLUSÃO

Muito trabalho foi feito para tornar o FPC mais compatível com o Delphi: novos recursos de linguagem, nomes de unidades pontilhadas. O trabalho foi amplamente patrocinado por uma empresa que deseja compilar seu programa Delphi com o Free Pascal sem precisar alterar os códigos-fonte do programa, alterar os códigos-fonte do programa. Como resultado, a compatibilidade do Delphi com o Free Pascal recebeu um impulso e todos os usuários do Free Pascal agora podem desfrutar de uma compatibilidade aprimorada com o Delphi. **Ao contrário das várias transições feitas no Delphi, que poderiam quebrar a compatibilidade com versões anteriores, a equipe do FPC decidiu que o FPC continua compatível com ele mesmo.** Como resultado, os usuários agora podem escolher se querem usar essas unidades mais recentes ou não: a escolha é sempre deles.

LIB-STICK ON USB CREDIT CARD BLAISE PASCAL MAGAZINE



Incluindo Higlight o resultado na pesquisa

LIBRARY-STICK USB-CARD: ALL ISSUES / CODE INCLUDED. ENGLISH VERSION
SAME INTERFACE AS THE INTERNET LIBRARY B$ 500 OFFER

ADICIONANDO UMA CAMADA DE TEXTO A ARQUIVOS PDF SOB O LINUX

POR HELMUT ELSNER

RESUMO

Para a maioria das impressoras em LINUX, não há ferramentas fornecidas pelo fabricante para obter arquivos PDF pesquisáveis ao digitalizar documentos. Os textos são incorporados apenas como imagens nos PDFs. Isso torna difícil encontrar algo em suas digitalizações.
É claro que também existem ferramentas correspondentes no LINUX, mas elas geralmente só podem ser operadas a partir da linha de comando. O "PDFsandwich" é uma delas, e muito boa. Queremos criar uma GUI para essa ferramenta para adicionar uma camada de texto a um ou mais arquivos PDF com apenas alguns cliques.

PREPARAÇÃO

Instale as ferramentas necessárias:

1. ImageMagick (*geralmente já faz parte da distribuição LINUX*)
2. Pdfsandwich
3. tesseract-ocr

sudo apt install pdfsandwich tesseract-ocr-de -y

Dê ao ImageMagick direitos de leitura|gravação para arquivos PDF:

`sudo nano /etc/ImageMagick-6/policy.xml`

Substitua a linha **`<policy domain="coder" rights="none" pattern="PDF" />`** por

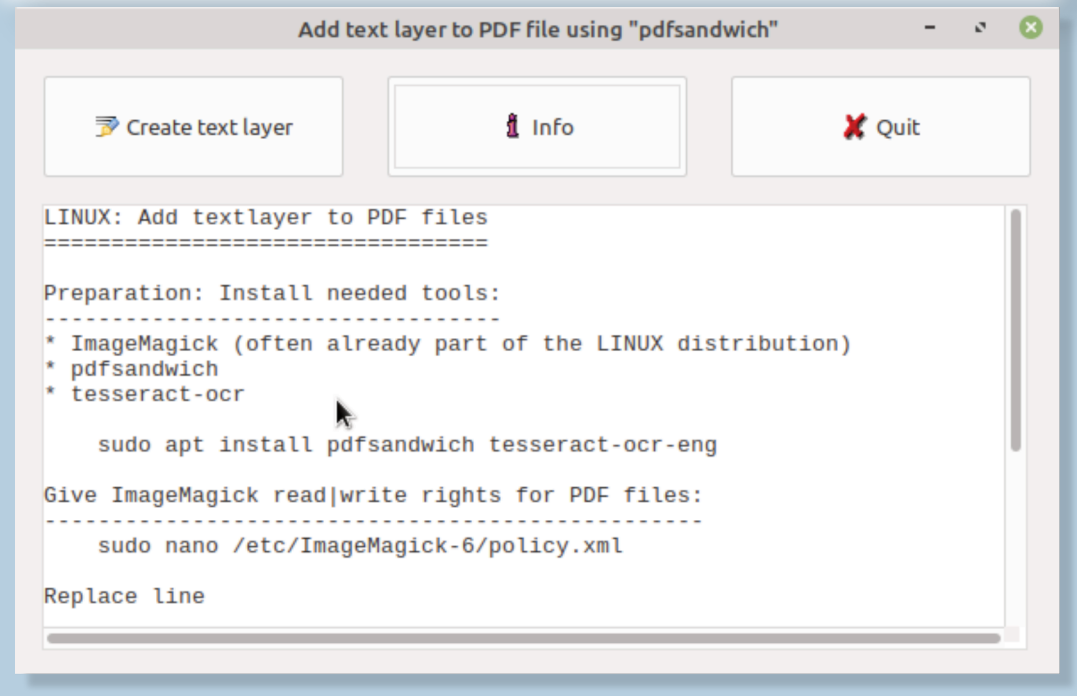
`<policy domain="coder" rights="read|write" pattern="PDF" />`

salve com Ctrl+O e feche com Ctrl+X.

Descompacte o arquivo baixado e coloque-o em algum lugar em seu diretório pessoal, por exemplo em um diretório **`/tools`** se quiser criar um arquivo .desktop para ele.

USO DO PDFTXT

Essa GUI é simples e (espera-se) autoexplicativa:



Com "**Create text layer**" (*Criar camada de texto*),
você é solicitado a selecionar arquivos **PDF** para processamento. Você pode selecionar vários arquivos **PDF** e processá-los de uma só vez.
Também é possível arrastar e soltar arquivos **PDF** do gerenciador de arquivos para a janela do programa e processar todos eles. O resultado do processamento é salvo no mesmo diretório que o arquivo original em um arquivo indicado com "**_ocr**" (**OptiCalRecognition**) no nome do arquivo.

ESTRUTURA DO PROGRAMA

FUNÇÃO DE AJUDA SIMPLES:
Na verdade, precisaríamos apenas de um botão para executar a ação, mas um pouco mais de informação é sempre é útil. Decidi exibir apenas um arquivo de texto em um `TMemo` como ajuda. É claro que é preciso verificar se o arquivo de texto com as instruções de instalação está localizado onde esperamos que ele esteja - no diretório do programa (`Application.Location`).

```
procedure TForm1.btnInfoClick(Sender: TObject); {Button Info}
begin
  if FileExists(Application.Location+infofile) then
   begin
     Memo1.Lines.LoadFromFile(Application.Location+infofile);
   end else
   begin
     Memo1.Lines.Add(infofile+errInfo);
     Memo1.Lines.Add(hntInfofile);
   end;
end;
```

PROCESSAR ALGUNS ARQUIVOS PDF:
Para processar um ou mais arquivos **PDF**, há o botão "**Create text layer**" (*Criar camada de texto*).
Para poder selecionar mais de um arquivo, devemos definir a opção `ofAllowMultiSelect` em `TOpenDialog`. Em seguida, podemos começar a trabalhar com a lista de arquivos.

```
procedure TForm1.btnAddTxtClick(Sender: TObject);
var
  i: Integer;
begin
  OpenDialog1.Title:=capOpenPDF; {Option ofAllowMultiSelect must be set}
  if OpenDialog1.Execute then
    begin
      Screen.Cursor:=crHourGlass;
      btnClose.Enabled:=false;
      Memo1.Lines.Clear; {Empty log output}
    try
      for i:=0 to OpenDialog1.Files.Count-1 do
        PDFAddText(Opendialog1.Files[i]);
    finally
      btnClose.Enabled:=true;
      Screen.Cursor:=crDefault;
    end;
  end;
end;
```

CHAMAR UM COMANDO DE TERMINAL COM TPROCESS:
Chamamos a rotina de processamento para cada arquivo. Usamos a extensão do arquivo para verificar se os arquivos são **PDF**.
Esse procedimento cria um processo (`pdftxt: TProcess;`) para executar o comando de terminal desejado, ou seja, "**pdfsandwich**", capturar a saída do comando e gravá-la no `TMemo` como um registro.
O comando também recebe uma série de parâmetros, incluindo o nome do arquivo **PDF** a ser processado (`pdftxt.Parameters.Add(fn);`).

```
procedure TForm1.PDFAddText(fn: string); {Process one PDF file}
var
  pdftxt: TProcess;
  outlist: TStringList;
  i: integer;
begin
  if Lowercase(ExtractFileExt(fn))=pdfext then
    begin
      Memo1.Lines.Add(fn);
      Memo1.Lines.Add('');
      outlist:=TStringList.Create;
      pdftxt:=TProcess.Create(nil);
      Application.ProcessMessages;
    try
      pdftxt.Executable:=app;
      pdftxt.Parameters.Add(paralang);
      pdftxt.Parameters.Add(Sprache);
      pdftxt.Parameters.Add(fn);
      pdftxt.Options:=pdftxt.Options+[poWaitOnExit, poUsePipes];
      pdftxt.Execute;
      outlist.LoadFromStream(pdftxt.Output); {Get commandline output}
    for i:=0 to outlist.Count-1 do {Append to log}
      Memo1.Lines.Add(outlist[i]);
      Memo1.Lines.Add(separ);
      Memo1.Lines.Add('');
    finally
      pdftxt.Free;
      outlist.Free;
    end;
    end else begin
      Memo1.Lines.Add(ExtractFileName(fn)+errPDF);
    end;
end;
```

ARRASTAR E SOLTAR:
Para tirar o máximo proveito de uma **GUI (GraphicalUserInterface)**, também deve ser possível arrastar e soltar.
Para isso, a opção `AllowDropFiles` deve ser definida como verdadeira para o formulário.
Em seguida, podemos arrastar qualquer número de arquivos **PDF** para a janela do programa para processamento.

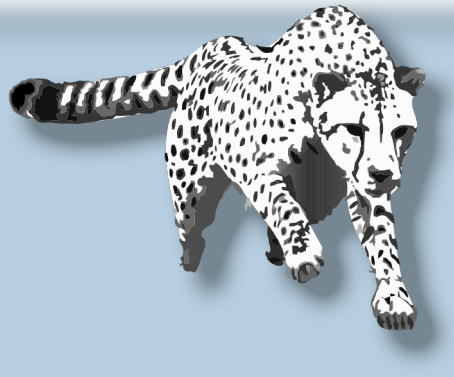
```
procedure TForm1.FormDropFiles(Sender: TObject; const FileNames:array of string);
var
  i: integer;
begin
  Screen.Cursor:=crHourGlass;
  btnClose.Enabled:=false;
  Memo1.Lines.Clear;
  Application.BringToFront; {Set focus to this program}
  try
    for i:=0 to high(FileNames) do
      PDFAddText(FileNames[i]);
  finally
    btnClose.Enabled:=true;
    Screen.Cursor:=crDefault;
  end;
end;
```

USABILIDADE

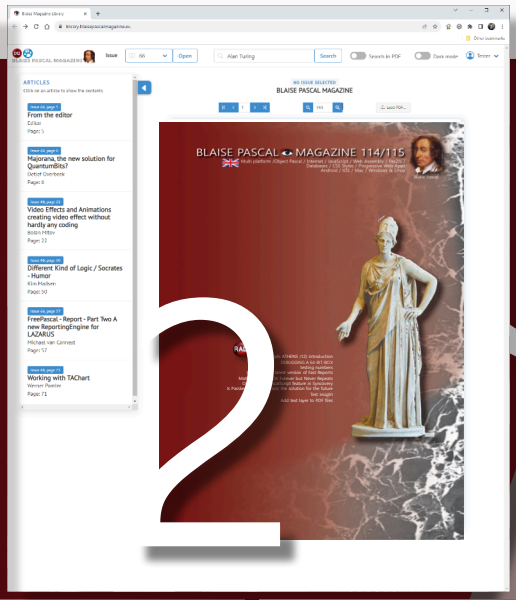
Obviamente, há também uma pequena ajuda na propriedade Hint para cada elemento de controle. Para alterar o tamanho da fonte no TMemo com o **mousewheel + Ctrl**, adicionamos um procedimento para reagir aos eventos `OnMouseWheelDown` e `OnMouseWheelup`.

```
procedure TForm1.Memo1MouseWheelDown(Sender: TObject; Shift: TShiftState;
                                     MousePos: TPoint; var Handled: Boolean);
begin
  if ssCtrl in Shift then
    Memo1.Font.Size:=Memo1.Font.Size-1;
end;

procedure TForm1.Memo1MouseWheelUp(Sender: TObject; Shift: TShiftState;
                                   MousePos: TPoint; var Handled: Boolean);
begin
  if ssCtrl in Shift then
    Memo1.Font.Size:=Memo1.Font.Size+1;
end;
```

É isso aí. Usando esse princípio, podemos criar uma pequena ferramenta com uma GUI para todos os tipos de comandos de terminal complicados dos quais não conseguimos nos lembrar.
O projeto está aqui: **`https://github.com/h-elsner/PDFtext`**



1



2

3

LAZARUS HANDBOOK

4

POCKET Edition +shipment

LAZARUS HANDBOOK PDF

5

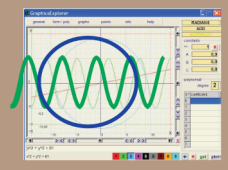
6

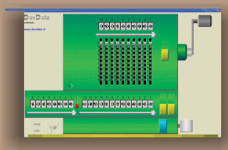
LEARN TO PROGRAM USING LAZARUS

HOWARD PAGE-CLARK

DAVID DIRKSE

including 50 example projects





7

BLAISE PASCAL MAGAZINE

COMPUTER (GRAPHICS) MATH & GAMES IN PASCAL

3

1. Assinatura de um ano
2. Visualização da revista na Internet
3. O mais novo LIB Stick
   - Todas as edições 1-111
   - No cartão de crédito
4. Manual de bolso do Lazarus
5. PDF do LH incluindo o código
6. Livro Aprenda a programar
   - usando o Lazarus PDF, incluindo 19 lições e projetos
7. Livro Computação Gráfica Matemática e jogos
   - PDF incluindo ±50 projetos

SUPER PACOTE 7PRODUTOS

PREÇO R$ 500

PREÇO NORMAL R$ 1200

Doe para a Ucrânia e obtenha uma licença gratuita em :

https://components4developers.blog/2022/02/26/donate-to-ukraine-humanitarian-aid/

Se o senhor for de origem ucraniana, poderá obter uma assinatura gratuita da Blaise Pascal Magazine, e também lhe daremos uma versão em pdf gratuita do Lazarus Handbook. O senhor precisa nos enviar seu nome ucraniano e seu endereço de e-mail ucraniano (que ainda funcione para o senhor), para que seja comprovado que o senhor é ucraniano de verdade. Envie-os para editor@blaisepascal.eu e o senhor receberá o livro e a assinatura.

BLAISE PASCAL MAGAZINE

Multi platform /Object Pascal / Internet / JavaScript / Web Assembly / Pas2Js / Databases / CSS Styles / Progressive Web Apps Android / IOS / Mac / Windows & Linux

COMPONENTS 4 DEVELOPERS

Faça uma doação para a Ucrânia e obtenha uma licença gratuita em:
https://components4developers.blog/2022/02/26/donate-to-ukraine-humanitarian-aid/

COMPONENTS 4 DEVELOPERS

D11

Doe para a Ucrânia e obtenha uma licença gratuita em :
https://components4developers.blog/2022/02/26/donate-to-ukraine-humanitarian-aid/

kbmMW Professional e Enterprise
NEW EDITION V. 5.23
kbmMemTable
NOVA EDICAO V. 7.99.00
Standard e Professional Edition

O 5.23.00 é uma versão que contém novidades, refinamentos e correções de bugs, além do suporte ao SSL v 3, suporte ao WebSoft, outras melhorias no SmartBind, novos algoritmos de hashing de alta performance, melhorias no RemoteDesk para amostragem e muito mais.
Essa versão exige o uso do kbmMemTable v. 7. 9 8 .0 0 ou mais recente.

- Suporte ao RAD Alexandria
- Win32, Win64, Linux64, Android, IOS 32, IOS 64 e
- Suporte a cliente e servidor OS X
- Servidor de aplicativos nativo de alto desempenho 100% definido pelo desenvolvedor servidor de aplicativos definido pelo desenvolvedor
- Suporte completo para balanceamento de carga centralizado e distribuído balanceamento de carga centralizado e distribuído e fail-over
- Suporte avançado a ORM/OPF, incluindo suporte a bancos de dados existentes
- Suporte avançado a registro em log
- Estrutura de configuração avançada
- Suporte avançado a agendamento para acesso fácil a programação de vários threads
- Serviço inteligente avançado e clientes para facilitar a publicação muito fácil da funcionalidade
- Funções aleatórias de alta qualidade.
- Geradores de senhas pronunciáveis de alta qualidade.
- Compactação de LZ4 e Jpeg de alto desempenho
- Estrutura completa de notação de objetos, incluindo suporte total a suporte total a YAML, BSON, Messagepack, JSON e XML
- Marshalling avançado de objetos e valores de e para
- YAML, BSON, Messagepack, JSON e XML
- Suporte a transporte TCP nativo de alto desempenho
- Transporte HTTPSys de alto desempenho para Windows.
- Suporte a CORS em serviços REST/HTML.
- Suporte nativo a clientes PHP, Java, OCX, ANSI C, C# e Apache Flex!

O kbmMemTable é a tabela de memória mais rápida e com mais recurso para produtos Embarcadero.

- Suporta facilmente grandes conjuntos de dados com milhões de registros
- Suporte fácil a streaming de dados
- Opção de usar o mecanismo SQL nativo
- Suporte a transações aninhadas e desfazer
- Construção nativa e rápida em M/D, agregação/agrupamento
- Recursos de seleção de intervalo
- Recursos avançados de indexação para desempenho extremo

- Novo: suporte total a Web-socket.
  A próxima versão do kbmMW Enterprise Edition incluirá várias novidades e aprimoramentos.
  Um deles é o suporte total a Web-sockets.
- Nova estrutura de internacionalização sensível ao contexto I18N para tornar seus aplicativos multilíngues.
- Novo suporte ORM LINQ para exclusão e atualização.
- Suporte a comentários em YAML.
- Novo suporte a StreamSec TLS v4 (por StreamSec).
  Muitos outros aprimoramentos e correções de recursos.

Acesse http://www.components4developers.com
Para obter mais informações sobre a kbmMW

- Acesso a banco de dados unificado e de alta velocidade (mais de 35 APIs de banco de APIs de banco de dados compatíveis) com pooling de conexões, metadados e cache de dados e metadados em todas as camadas
- Acesso de vários cabeçalhos ao servidor de aplicativos, via REST/AJAX, binário nativo, Publish/Subscribe, SOAP, XML, RTMP a partir de navegadores da Web, dispositivos incorporados, servidores servidores de aplicativos vinculados, PCs, dispositivos móveis, sistemas Java e muitos outros clientes
- Suporte completo para hospedagem de aplicativos baseados em FastCGI (normalmente PHP/Ruby/Perl/Python)
- Suporte nativo completo ao AMQP 0.91 (Advanced Message Queuing Protocol)
- Área de trabalho remota completa, segura e com marca, de ponta a ponta, com vídeo HD quase em tempo real, suporte a 8 monitores, detecção de textura, compactação e compartilhamento de área de transferência.
- Pacote do kbmMemTable Professional, que é a tabela de memória mais rápida e mais rica em recursos em tabela de memória para produtos Embarcadero.

COMPONENTS 4 DEVELOPERS