

Efeitos Visuais Avançados

Skia4Delphi

Maximizando o impacto visual de sua interface de usuário em Delphi em todas as plataformas por meio de efeitos avançados com Skia4Delphi.

Agenda

- Skia4Delphi
- Shaders
- Um TCanvas incrementado
- Acrílico
- Fontes personalizadas

Skia4Delphi

- Biblioteca gráfica
 - Open source & Free
 - Cross-platform
 - Baseada no Google Skia, utilizado pelo Chrome, Flutter, Firefox, Xamarin
 - VCL, FMX e Console
- Renderizador para o FMX
- Recursos extras
 - SVG
 - Lottie
 - PDF
 - Imagens WebP
 - Configurações avançadas de textos
 - Padronização de linguagem shader
 - Entre outros!



github.com/skia4delphi/skia4delphi

Webinar anterior:
blogs.embarcadero.com/?p=140459

Shader

Código específico escrito para a GPU calcular a cor de cada pixel durante um desenho, com base na coordenada do pixel, algum algoritmo e/ou uniformes (dados de entrada ou imagens) fornecidos ao shader.

O shader é chamado internamente uma vez para cada pixel.

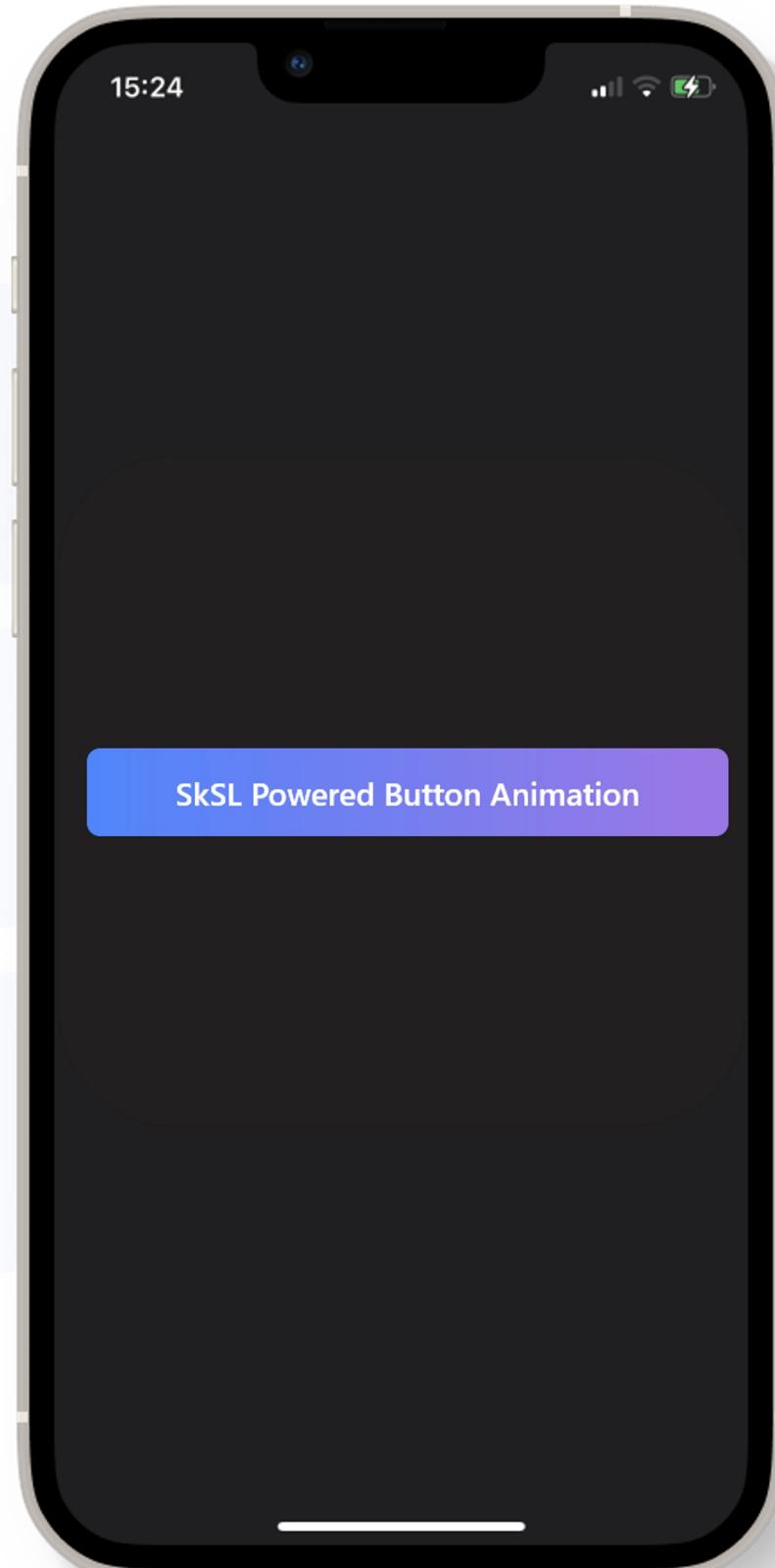


Shader na vida real

Shader não é usado apenas em jogos e forms 3D.
A imaginação é o limite.



App de álbuns, com
shader de virada de
página realista, feito em
Delphi usando
Skia4Delphi e SkSL.



Botão animado feito com
shader, disponível nos
demos do Skia4Delphi
(FMX e VCL).

Location: Skia4Delphi\Samples\
Conference 2022 - Shader Button

Skia Shading Language

O Google Skia também criou sua própria linguagem de shader, mas para padronizar o desenvolvimento multiplataforma. Independente da biblioteca gráfica utilizada, independente da plataforma, agora é necessário 1 único código de shader.

A sintaxe do SkSL (Skia's Shading Language) é muito semelhante ao shading language do OpenGL, o GLSL.

Links úteis

- Shadertoy (maior repositório de GLSL): shadertoy.com
- Skia Shaders Playground: shaders.skia.org
- Documentação oficial: skia.org/docs/user/sksl
- Diferenças do GLSL: github.com/google/skia/tree/main/src/sksl#differences-from-gls
- Guia para conversão do GLSL: github.com/jimmckeeth/SkiaSimpleShaderViewer/blob/main/SkSL-Overview.md
- Visualizador báscio de SkSL escrito em Delphi: github.com/jimmckeeth/SkiaSimpleShaderViewer

Um TCanvas incrementado

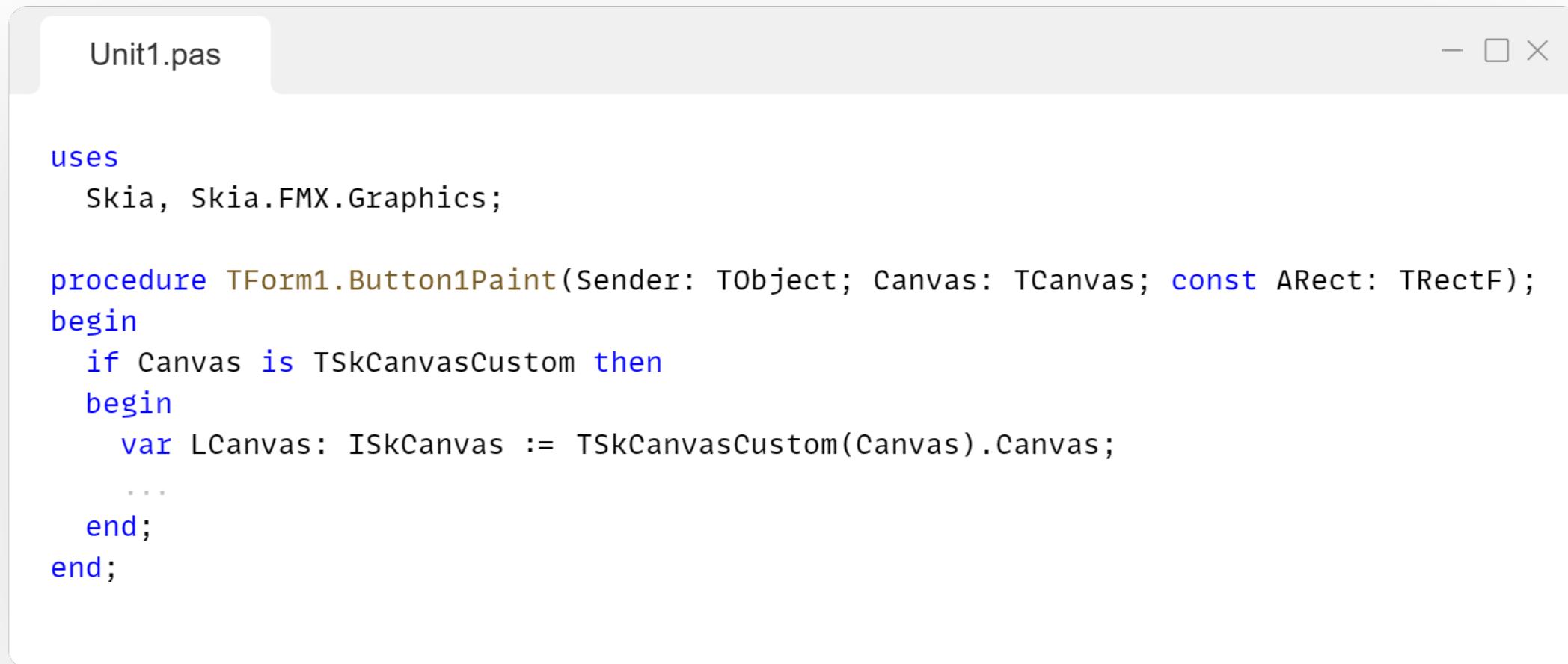
“Desbloqueie novos recursos do TCanvas!!”

No FMX, quando o Skia é o renderizador da aplicação (GlobalUseSkia = True), todos os TCanvas da aplicação (controles, forms, bitmaps), são do tipo TSkCanvasCustom.

Então, em qualquer evento paint, podemos castar qualquer Canvas e acessar o SkCanvas (Canvas do Google Skia), que possui inúmeras funcionalidades extras do TCanvas padrão, tais como:

- Clipping: excluir ou adicionar regiões desenháveis
- Efeitos extras
- Filtros de cores extras
- Máscaras de alfa
- Blending: composição dos pixels dos novos desenhos com os existentes
- Shader: permite, por exemplo, trocar a cor de todos os desenhos adiante por um gradiente

Um TCanvas incrementado



The screenshot shows a Delphi IDE window titled "Unit1.pas". The code in the editor is as follows:

```
Unit1.pas

uses
  Skia, Skia.FMX.Graphics;

procedure TForm1.Button1Paint(Sender: TObject; Canvas: TCanvas; const ARect: TRectF);
begin
  if Canvas is TSkCanvasCustom then
  begin
    var LCanvas: ISkCanvas := TSkCanvasCustom(Canvas).Canvas;
    ...
  end;
end;
```

Acessando SkCanvas a partir de um TCanvas:

- Apenas FMX
- Renderizador do Skia ativado (*GlobalUseSkia := True; no dpr*)
- **Controles/Formas:** Apenas durante procedimentos/eventos de paint
- **Bitmaps:** Apenas dentro do BeginScene

Efeito Acrílico

Uma tendência crescente para a interface do usuário em 2023 é o *Glassmorphism*, que é um estilo de design que, como o nome indica, usa as propriedades do vidro para aprimorar os designs. Esse efeito também é conhecido como acrílico ou vidro fosco.

O Skia4Delphi trouxe inúmeros recursos gráficos para o Delphi, adicionando várias novas opções de efeitos, incluindo este efeito acrílico.



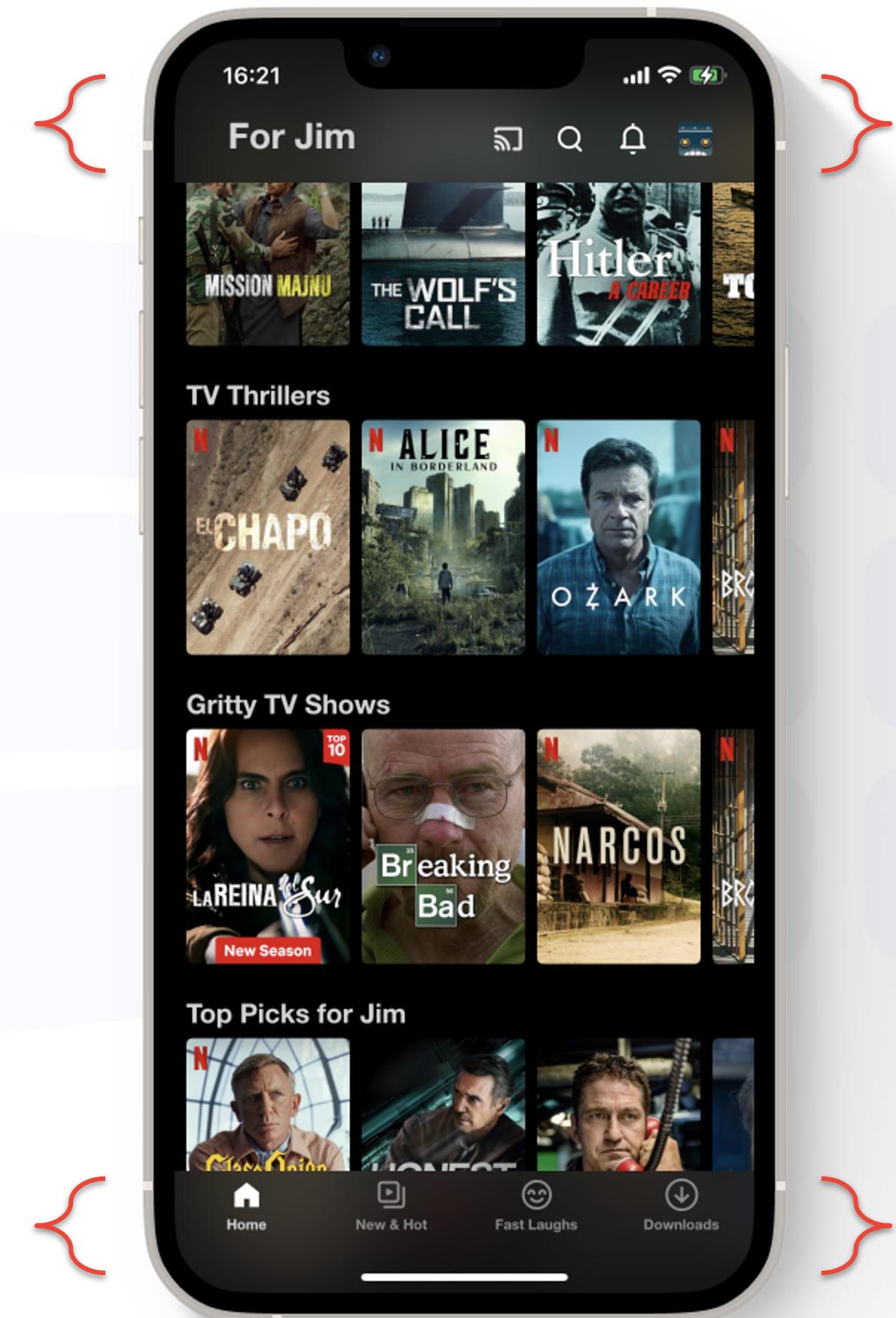
Retângulo semi-transparente



Retângulo semi-transparente +
Backdrop blur

Acrílico da Netflix

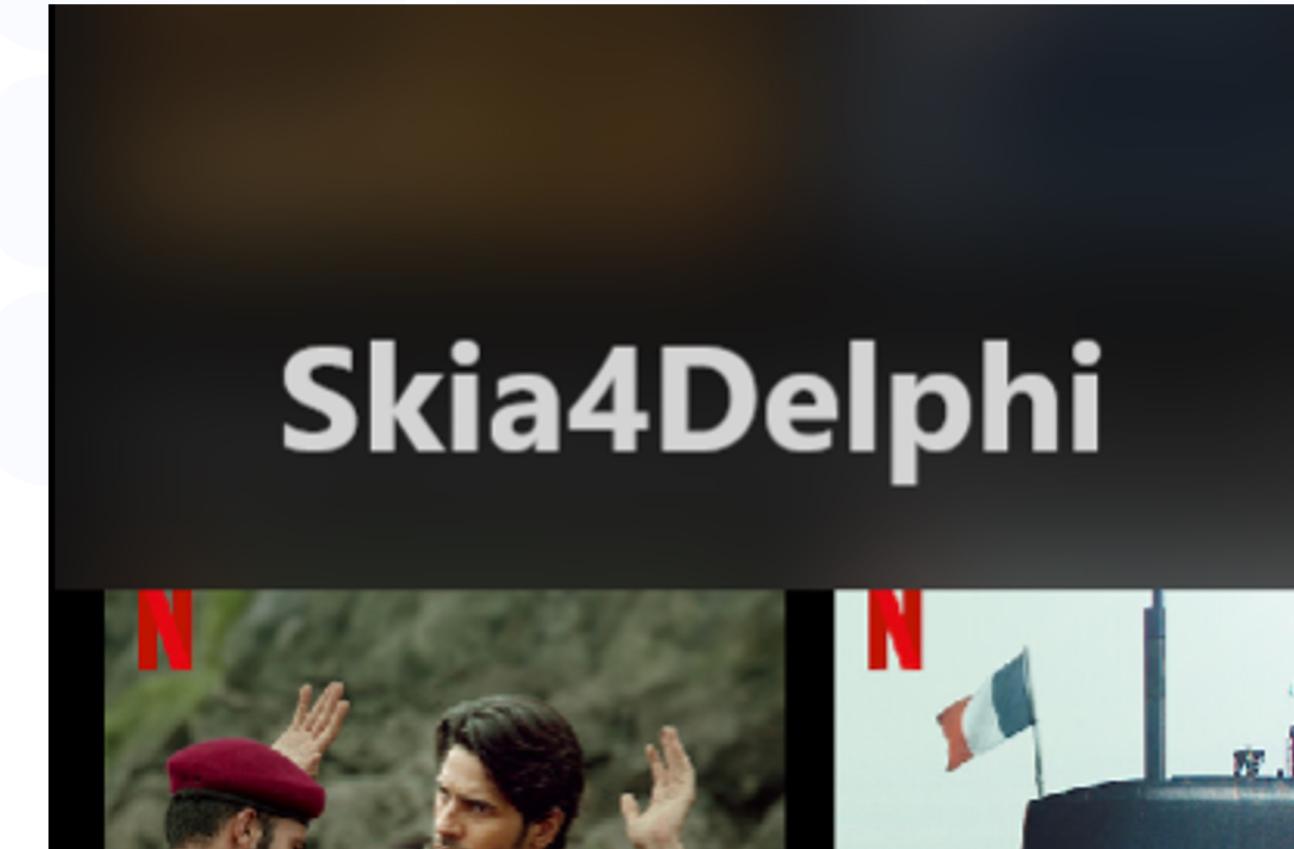
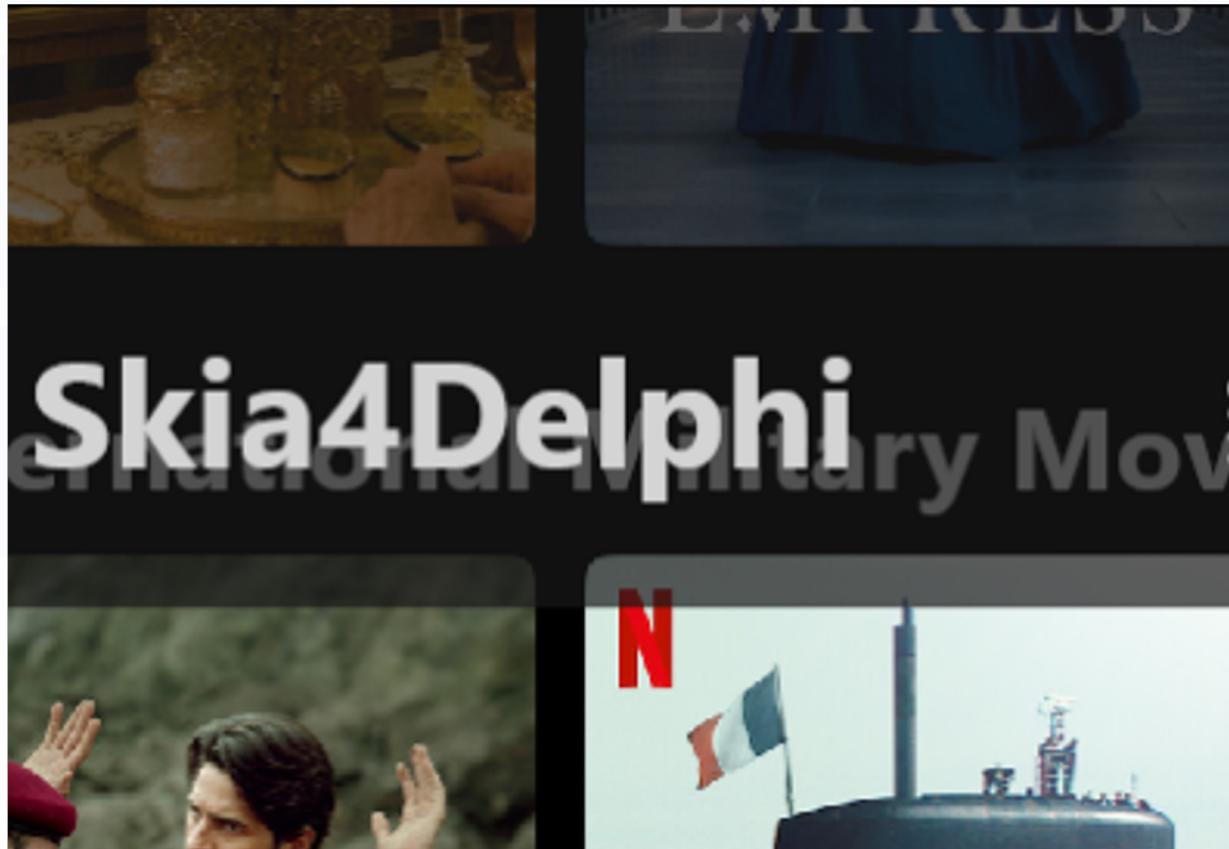
Nós recriamos o efeito acrílico do aplicativo Netflix para demonstrar como é simples criar aplicativos de qualidade incrível em Delphi com muito pouco código.



Normal

VS

Acrílico

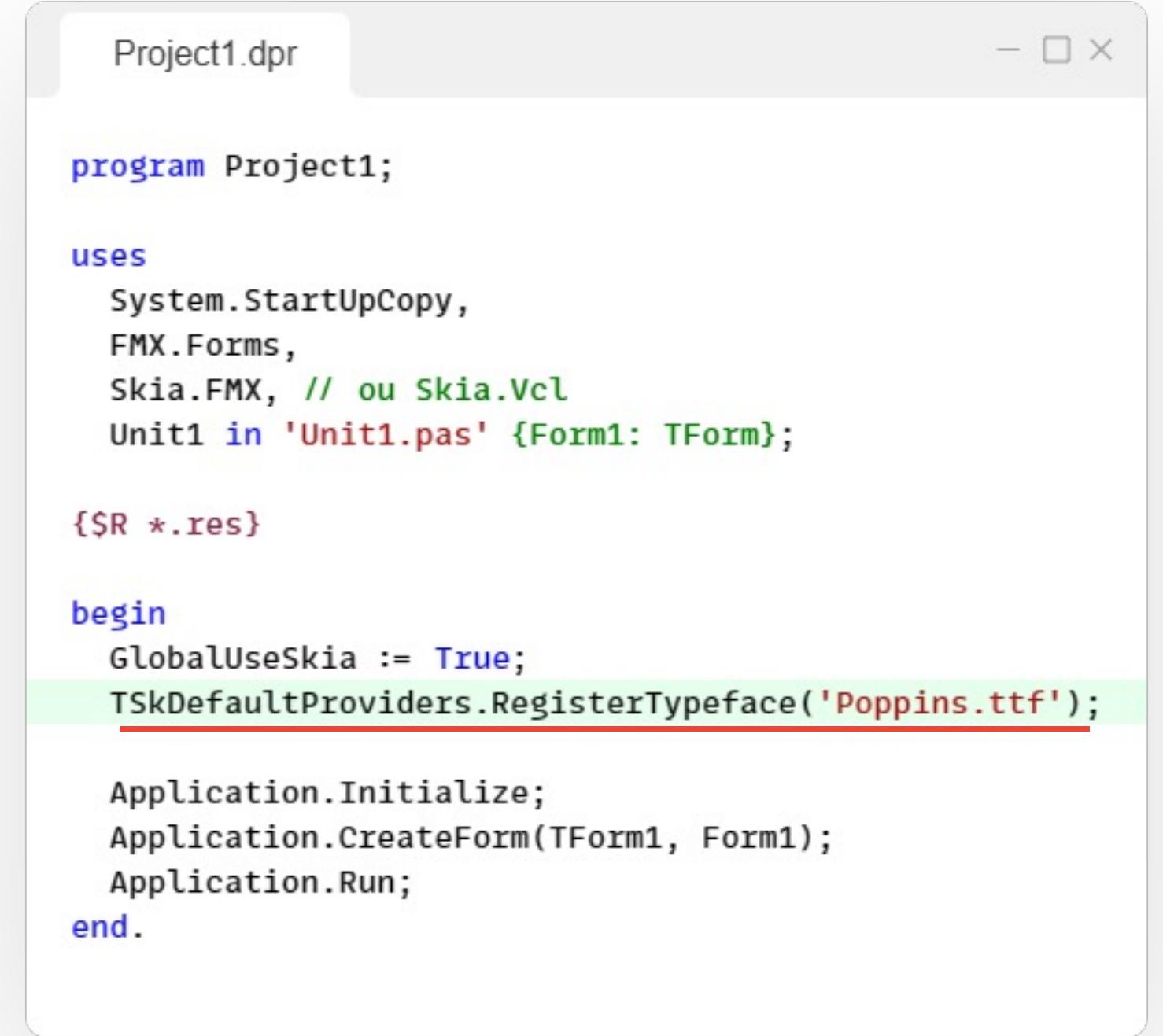


Demo Acrílico

Fontes personalizadas

Adicione apenas 1 linha na inicialização do app e seu app reconhecerá a fonte personalizada.

- Controles do Skia, como o TSkLabel, reconhecerão a nova fonte (**VCL e FMX**)
- Todos os controles do FMX reconhecerão a fonte **caso o Skia seja o renderizador do app**
(GlobalUseSkia := True;)



```
Project1.dpr

program Project1;

uses
  System.StartUpCopy,
  FMX.Forms,
  Skia.FMX, // ou Skia.Vcl
  Unit1 in 'Unit1.pas' {Form1: TForm};

{$R *.res}

begin
  GlobalUseSkia := True;
  TSkDefaultProviders.RegisterTypeface('Poppins.ttf');

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

Obrigado!

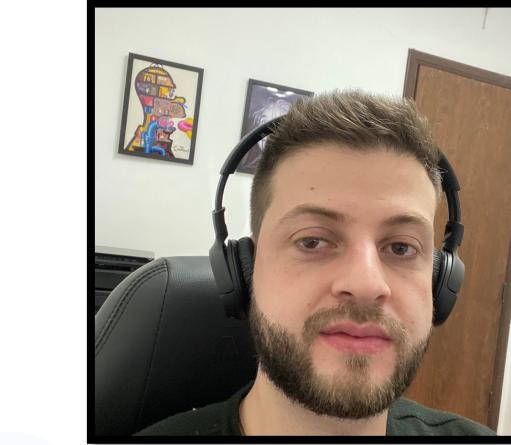
Skia4Delphi



[@skia4delphi](https://github.com/skia4delphi)



[@skia4delphi](https://t.me/skia4delphi)



Vinícius Felipe Botelho Barbosa



[@viniciofbb](https://github.com/viniciofbb)



[@viniciofbb](https://t.me/viniciofbb)



[@viniciofbb](https://www.linkedin.com/in/viniciofbb)



viniciusfbb2@gmail.com