

Product Address

RAD Studio

12.x Athens

About me

Marco Cantu

Senior Product Manager, RAD Studio

marco.cantu@embarcadero.com

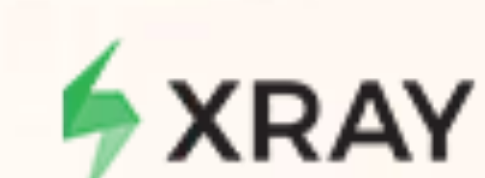
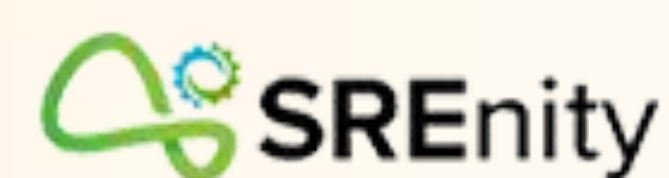
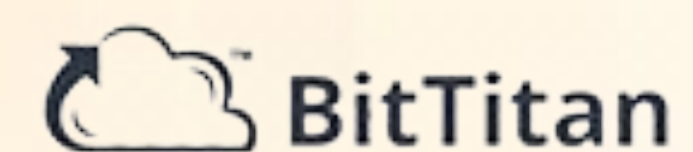
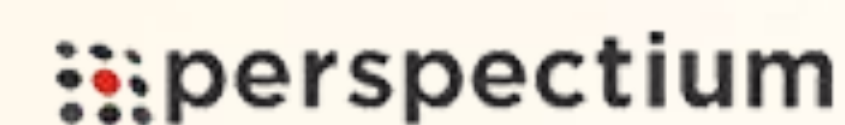
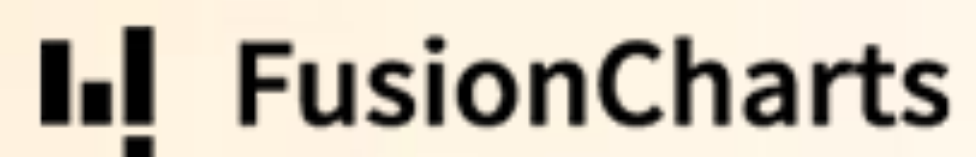
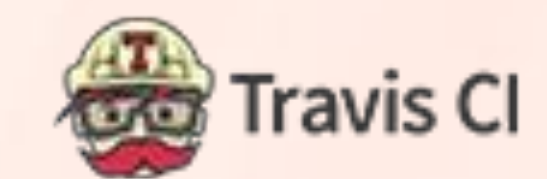
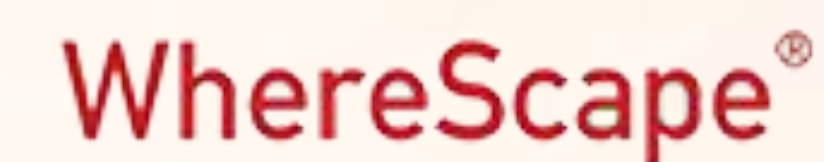
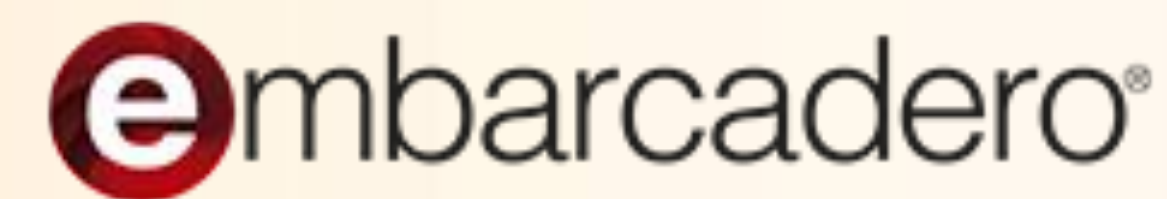
Delphi books author and guru



Embarcadero and Idera

Embarcadero is owned by Idera, www.ideracorp.com

Part of the DevOps division along with other tools for developers

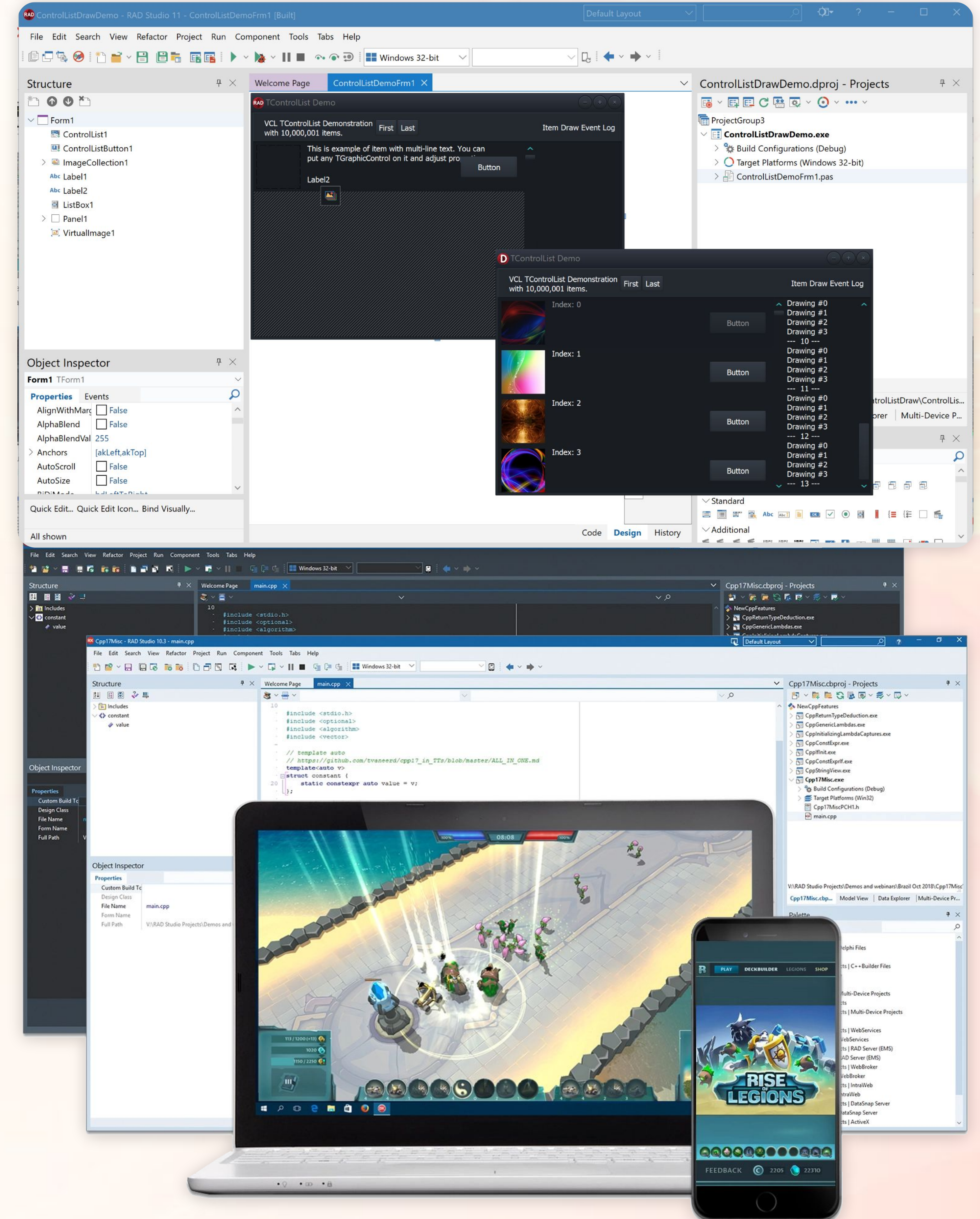


What is RAD Studio?

The ultimate IDE for building

- multi-platform
- high-performance
- native applications

in Modern Object Pascal
and Extended C++
with powerful visual design tools
and integrated toolchains



What is Great in RAD Studio?



Developer Productivity

Shipping is a feature. Get to market 5x faster.



Fast Native Apps

Native compilers give your apps the speed they need.

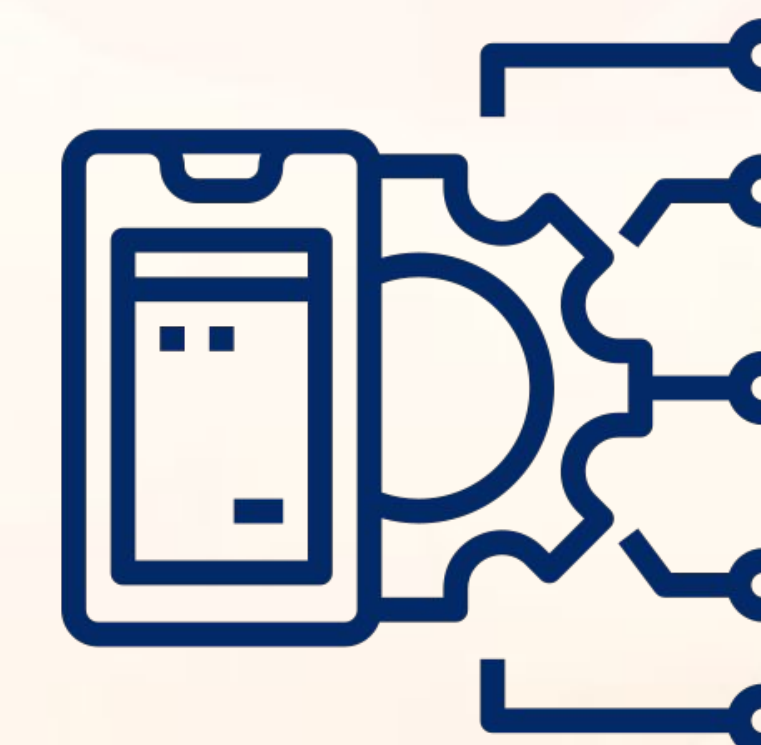
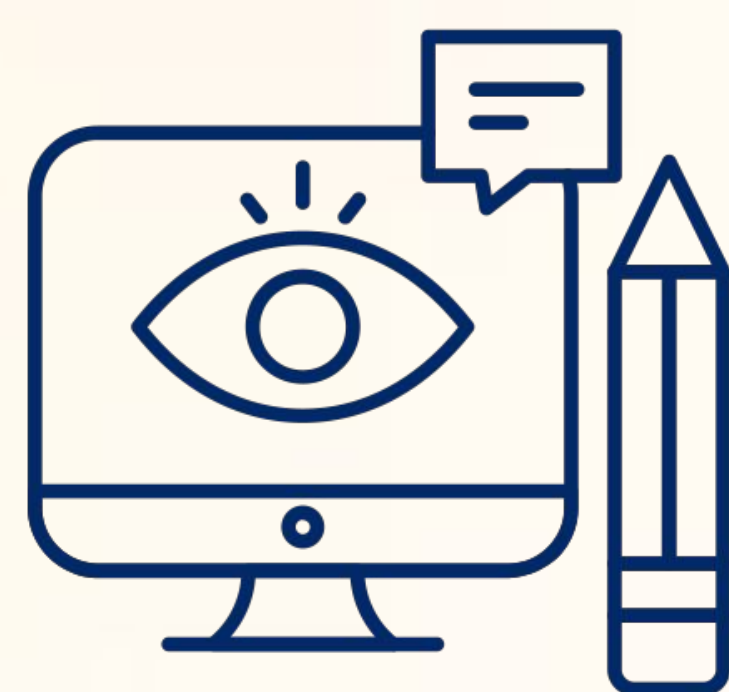


Strong Community

Technology Partners, MVPs, trainers, authors and developers – worldwide

Visual Designers

Forget prototyping in another tool, just design it and finish it in one IDE



Platform API Access

Gives you access to all the platform APIs on all platforms.

Database Access

Key to Delphi's initial design, database access is easy and fast.



Backward Compatibility

You have an investment in your code, keep your code relevant.

RAD Studio, C++ Builder, Delphi

12.1 Athens

released April 2024



RAD Studio 12 Athens was a *Mega Release*

1. C++

- Visual Assist
- New Clang Tech Preview

2. Installer & IDE

- Multidevice Icons
- DelphiLSP
- VCL Designers, IDE Misc, ToolsAPI

3. FireMonkey & Skia

- Skia UI Controls and rendering and graphic formats and way more (and VCL as well)
- TMemo and TEdit

4. VCL

- MDI and Tab based UIs
- Fonts and Screen, Components improvements

4. Data

- JSON Data Binding
- FireDAC Secure Coding
- Sqids, RAD Server, HTTP, REST

5. Delphi

- String literals improvements
- Platforms (Windows, Android)
- Circular Uses Statements, Floating Point

6. Quality

- *How many bug fixes this release?!*

The focus of RAD Studio 12.1 was on improving the quality of the many new features added in RAD Studio 12 Athens and on completing the Clang upgrade (moving it from preview to full feature)

RAD 12.1 Summary: Focus on Quality

- **Some substantial features**
 - Editor split views and other IDE enhancements
 - The delivery of the new C++Builder Win64 Clang compiler
 - Android API level 34 support
- **Quality highlights include**
 - C++ Builder VA integration
 - VCL and extensive FireMonkey quality
 - RTL, database, etc
 - Includes the 12.0 Patch 1 Fixes (released in January)



Agenda

- IDE Improvements
- Win64 Clang Toolchain
- Delphi Language and RTL
 - Android API Level 34
- UI Libraries: FireMonkey
- UI Libraries: VCL (and Windows)
- All About Data FireDAC & Web
- Intrastructure
- Summary



Before we get into it...

- Are you using Delphi or C++ (or both)?



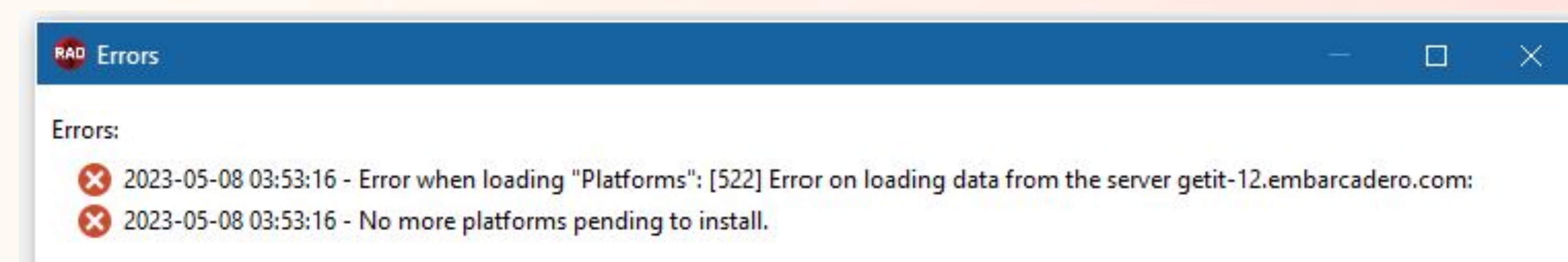
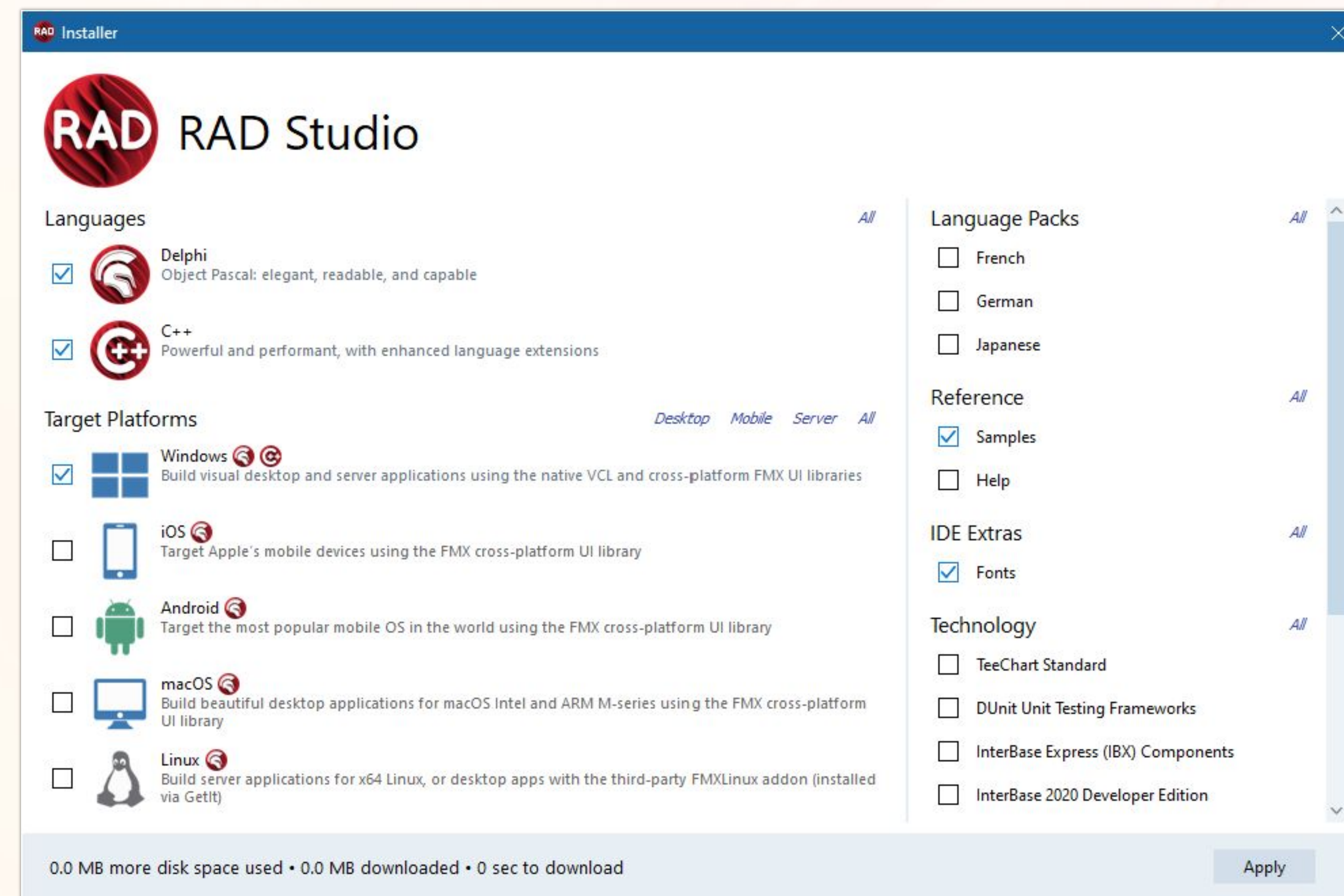
- Are you **primarily** using 12.x?
Or 11.x?
Or older versions?

IDE Improvements

New installer UX

Significantly reworked and simplified installer: much easier to choose what will be installed

- Simplify installation choices.
- Everything is on one page: Choose languages, platforms, extras.
- Better info on the effect of changes
- Improved error messages, easier access to logs
- Uses VCL styles, supports high DPI



Syntax Highlighting throughout the IDE

Highlighting added to

- Call Stack
- Debugger tooltips
- Error Insight messages
- Structure view
- Navigation toolbar

Watch List - Thread 9628	
Watch Name	Value
<input checked="" type="checkbox"/> Caption	'Form3'
> <input checked="" type="checkbox"/> ClientRect	(0, 0, 1248, 882, (0, 0), (1248, 882))
<input checked="" type="checkbox"/> @Self	\$9BF800

Call Stack - Thread 4304

- ➔ Unit3.TForm3.**FormCreate**(\$3382820)
- Vcl.Forms.TCustomForm.**DoCreate**
- Vcl.Forms.TCustomForm.**AfterConstruction**
- System.**_AfterConstruction**(\$3382820)
- Vcl.Forms.TCustomForm.**Create**(\$33ACD10)
- Vcl.Forms.TApplication.**CreateForm**(TForm3,(no value))
- Project3.**Project3**
- :75027ba9 KERNEL32.BaseThreadInitThunk + **0x19**
- :76fab79b ntdll.RtlInitializeExceptionChain + **0x6b**
- :76fab71f ntdll.RtlClearBits + **0xbf**

AllocCoTaskMemStr(const S: string): LPCWSTR

ApplicationMainHandle: HWND

CenterWindow(Wnd: HWND)

CopyData(Handle: THandle): THandle

CreateMessageDialog(const Msg: string; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons): TForm

CreateMessageDialog(const Msg: string; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons): TForm

< 337 result(s) found. >

Structure

 **GetImageIndex:** Integer

 **GetTabHintText:** string

 **Close**(var Allowed: Boolean)

 **GetTabColor:** TColor

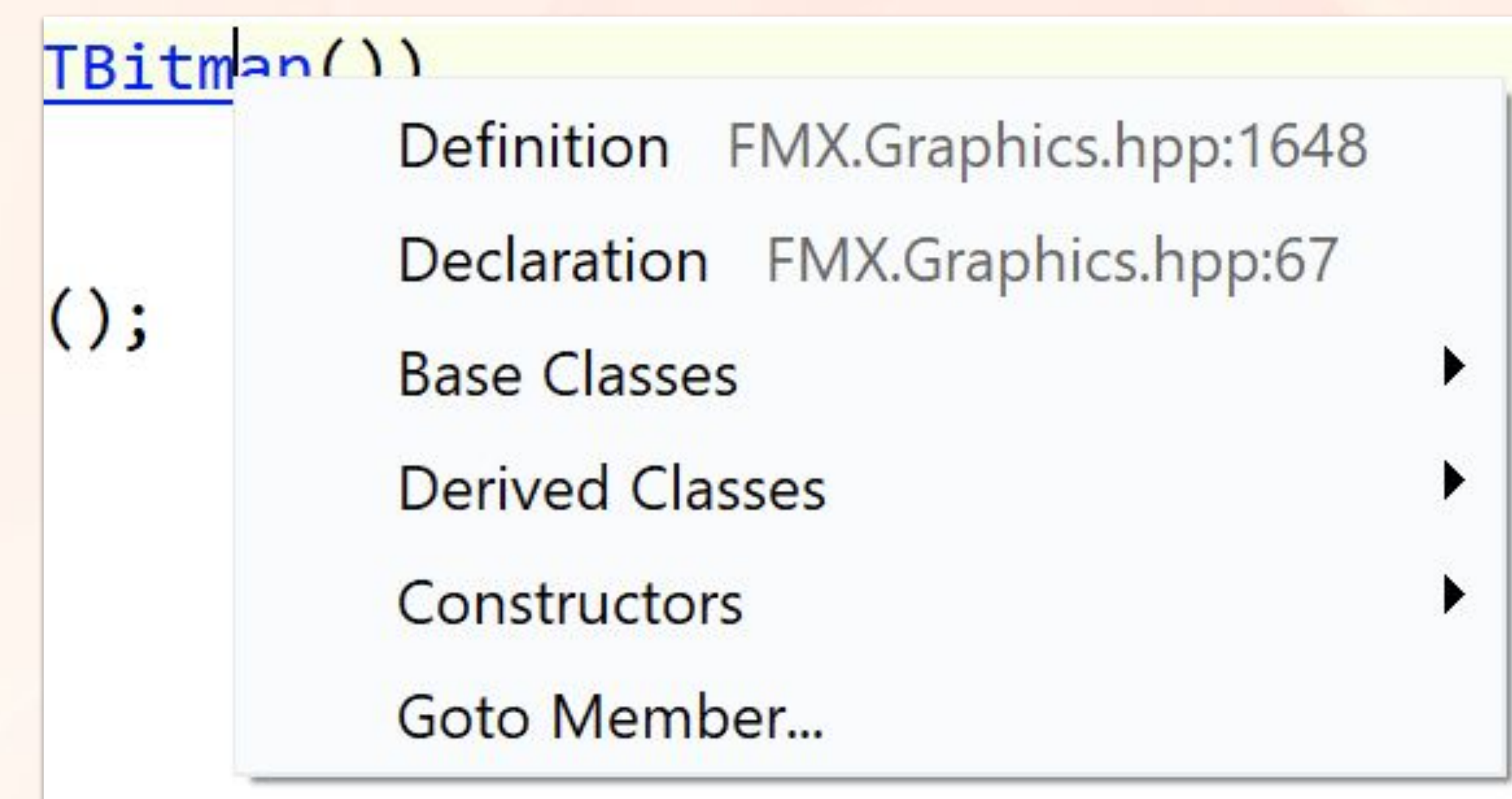
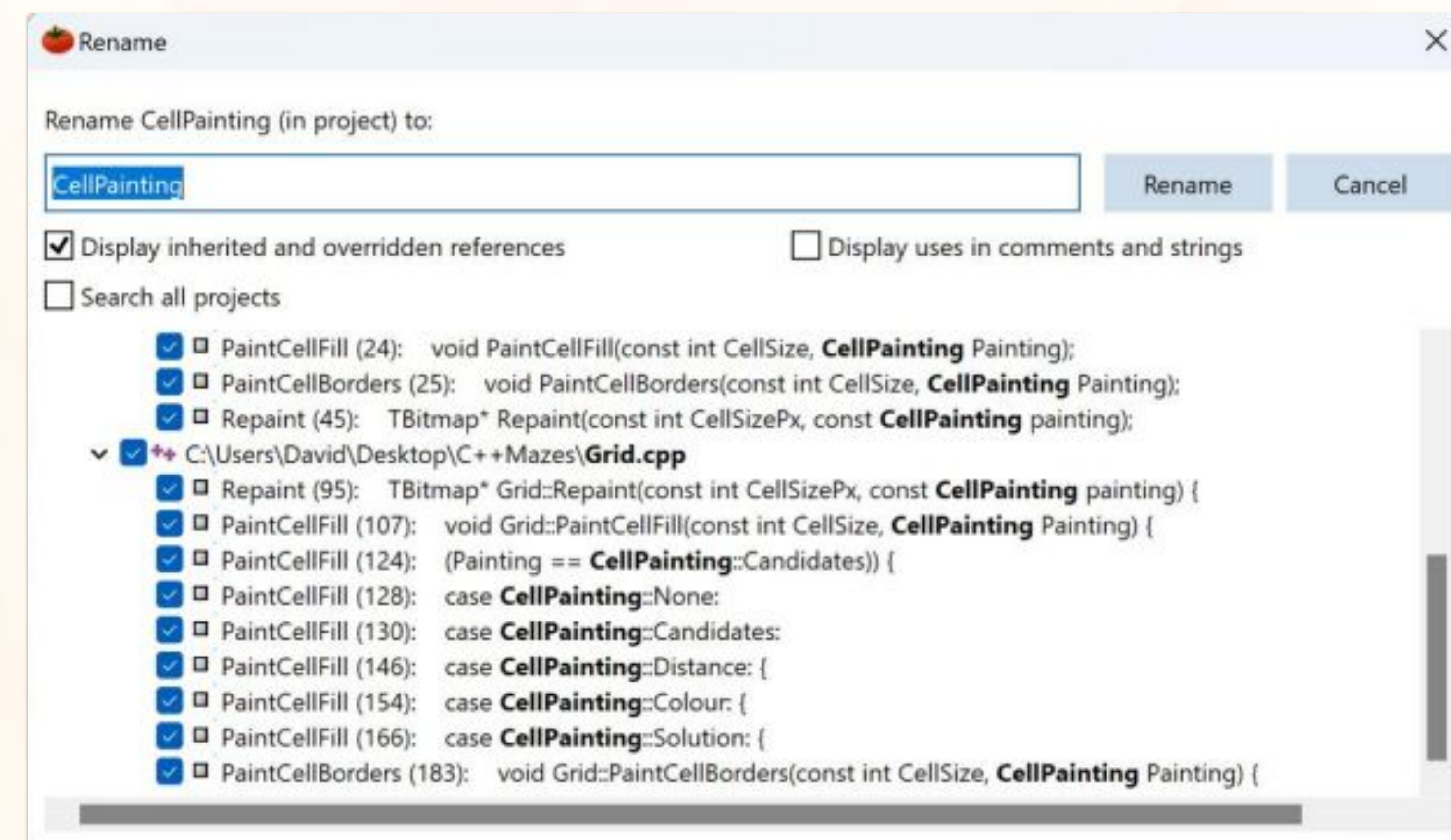
 **LoadViewState**(const Desktop: TCustomIniFile; const ViewDeskSection: string)

Visual Assist in C++ Builder

An amazing initial integration

- Code completion: fast, heuristic
- Refactoring: Rename, Add Include, and create implementation/declaration
- Navigations: Find References, Find Symbol, toggle impl/decl, Go To Member
- And Go To Related: the single most amazing feature ever

Code Insight (completion etc) uses VA by default

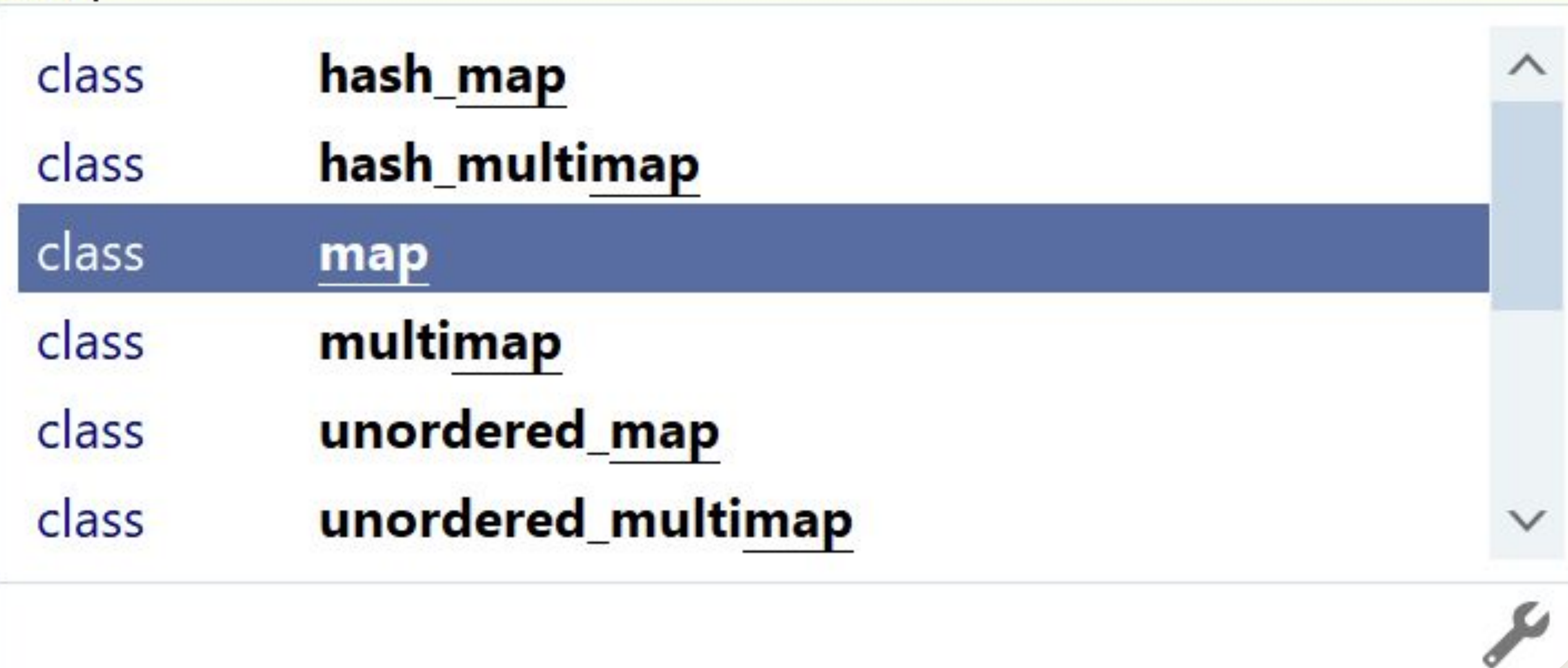


1. Code Completion

Fast code completion: very speedy to display
Completely different engine
Understands types, but also *heuristic* in results

```
#include <iostream>
#include <tchar.h>

int _tmain(int argc, _TCHAR* argv[])
{
    std::map|
}
```



```
PaintGrid();
Ac|
```



Even suggests items not #include-d,
from headers VA knows about!

Plus: tooltips, parameter completion...

2. Refactorings

You have code and want to:

- Rename a method or type, safely, everywhere it's used in your project group
- You wrote code and now it won't compile and you have to scroll to the top to add a `#include`. If only this could be done for you...
- You wrote a method and want the declaration automatically added – or vice versa

Visual Assist does these!

3. Navigations

You want to move around your code!

- Move to a type or a method definition
- See what uses a method or class
- Navigate to a symbol implementation
- See the ancestors of a type
- List the descendant methods override of a virtual method

Visual Assist does these!

Visual Assist work in 12.1

Focused on quality

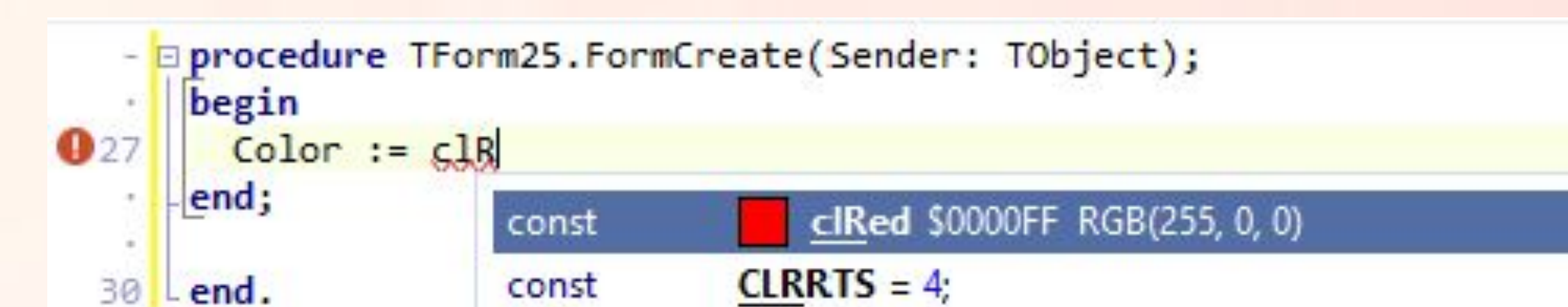
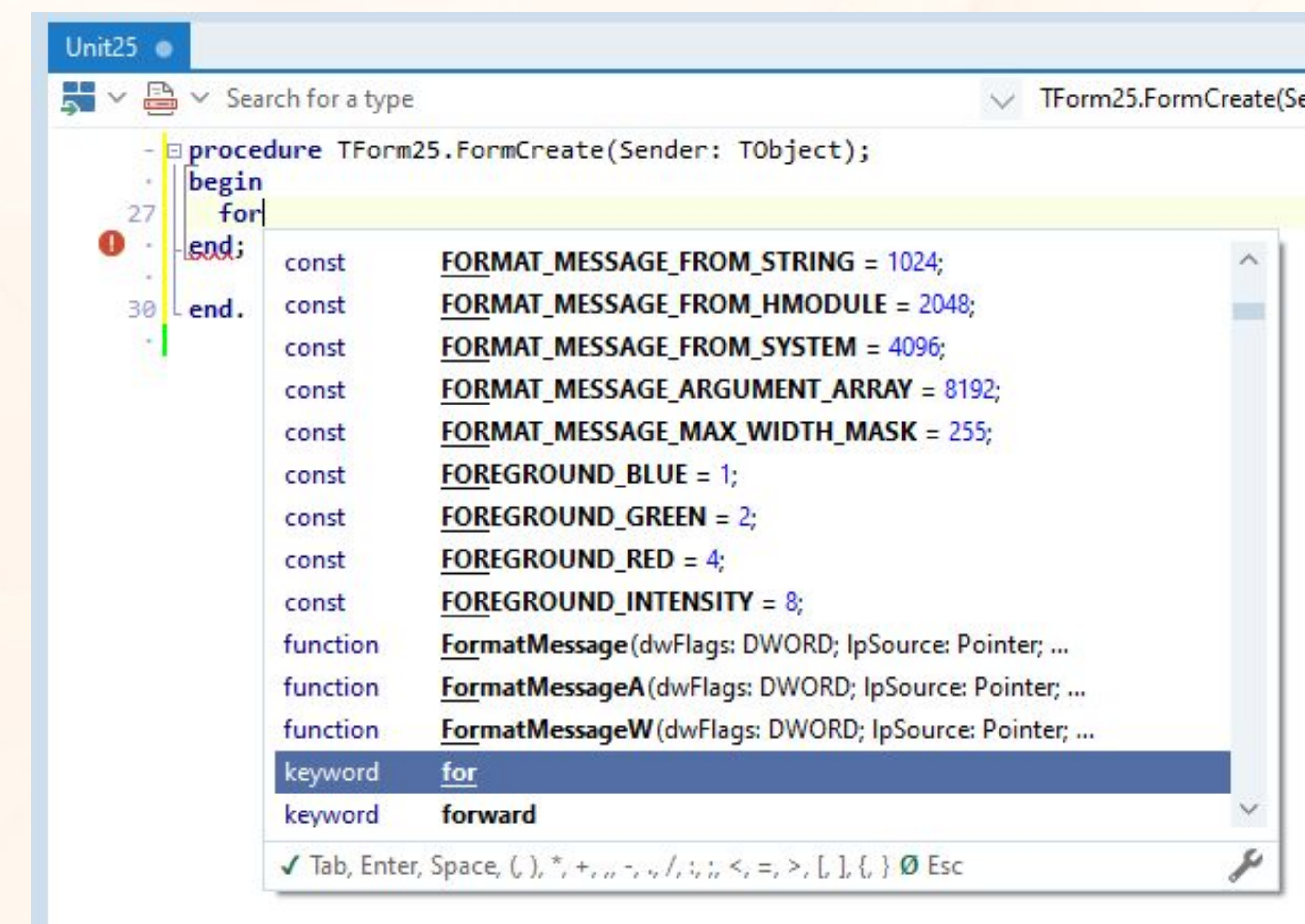
- Expanding rename refactoring to rename across code and visual designers
 - Renaming a component will include renaming it in the designer
 - Renaming now also renames event handlers
- Solved an issue where VA was left with a cache if a file was closed unsaved
- Code completion on object properties is working correctly
- Improved `#include` preprocessor directive autocomplete with `<` and `“”`
- Minor UI improvements on menus

A great feature for C++Builder 12. More to come in 12.x releases!

Delphi LSP (12.0)

Improvements focused on stability, especially closing or switching projects. Plus:

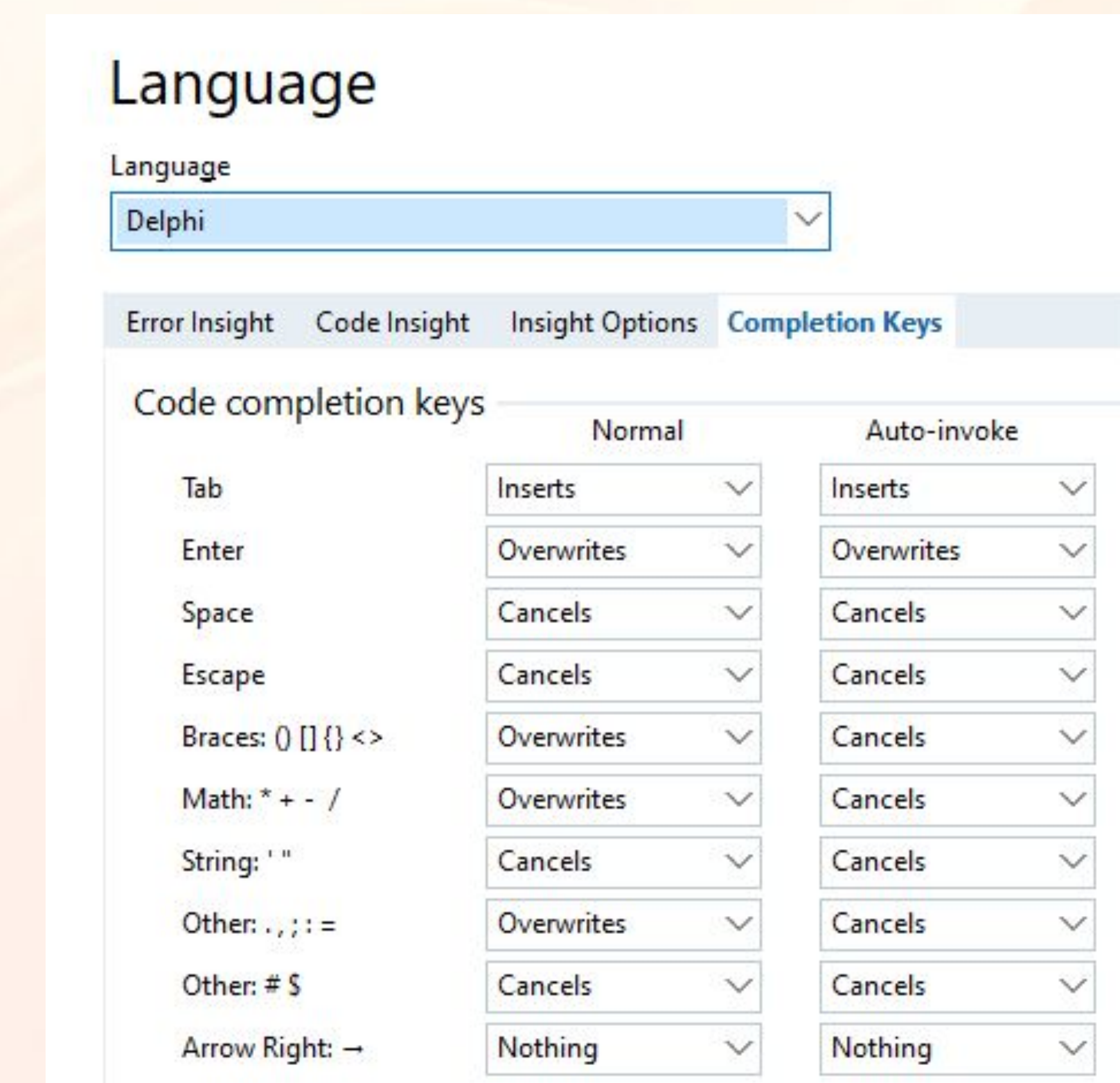
- Autocompleting [] for array types
- Improved color constant representation
- Re-introduced code completion's auto-invoke feature (off by default in this release)
- Language keywords included in the completion list
- Templates still included, but in correct locations
- New menu item to restart the LSP server
- The code completion window can show the keys that affect completion



Delphi LSP Quality and Improvements (12.1)

Goal: making auto-completion work smoothly and avoid anything unexpected

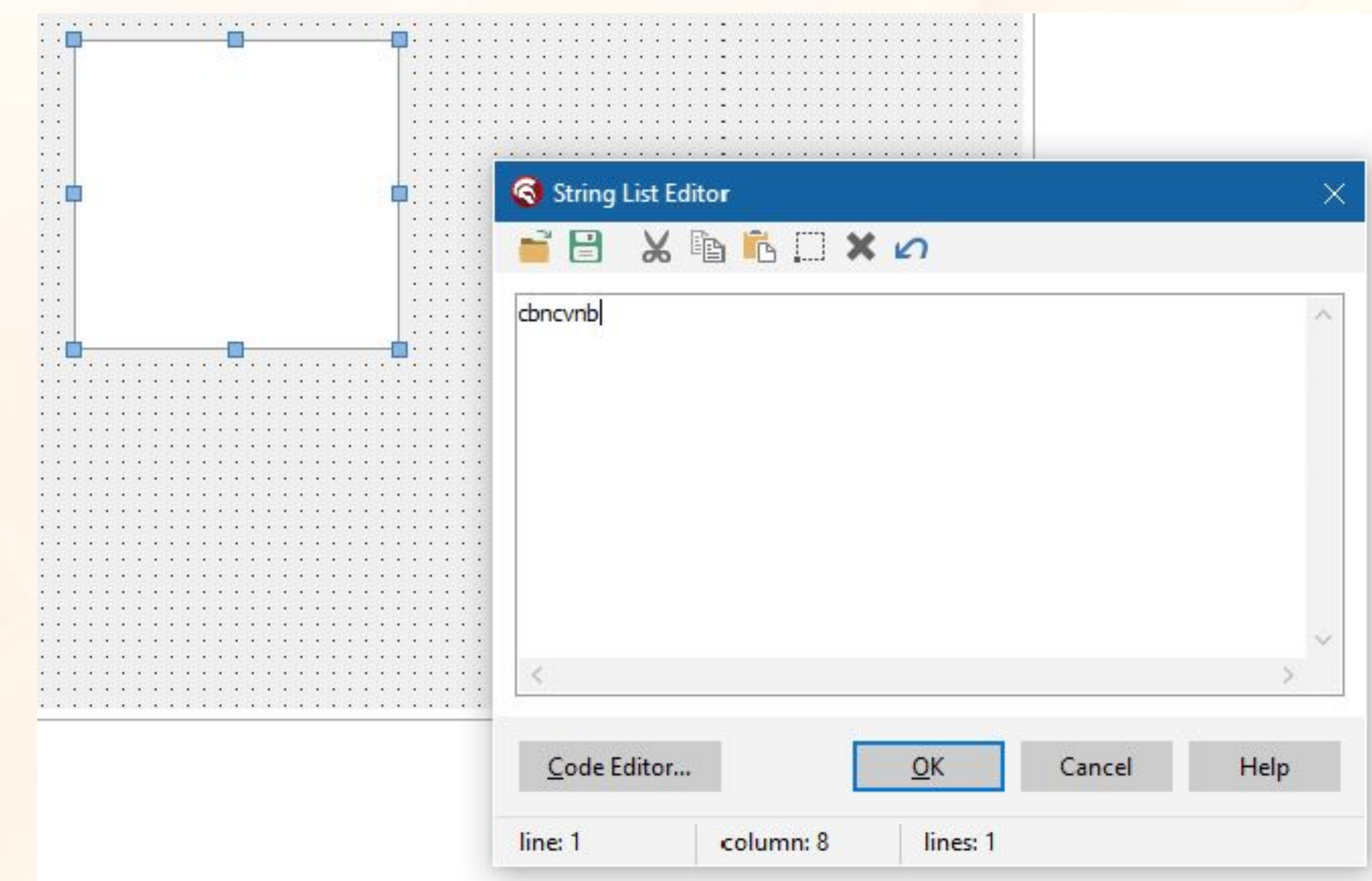
- Ability to customize the behavior of specific keystrokes or ‘*acceptance keys*’ when code completion is shown
 - When shown normally
 - When auto-invoked
- Customers can configure completion to match preferred behavior
- Added a number of improvements to Delphi LSP overall
 - Several fixes for class completion
 - Better code completion inside generic classes in more situations



New VCL Designers (from Ksvc)

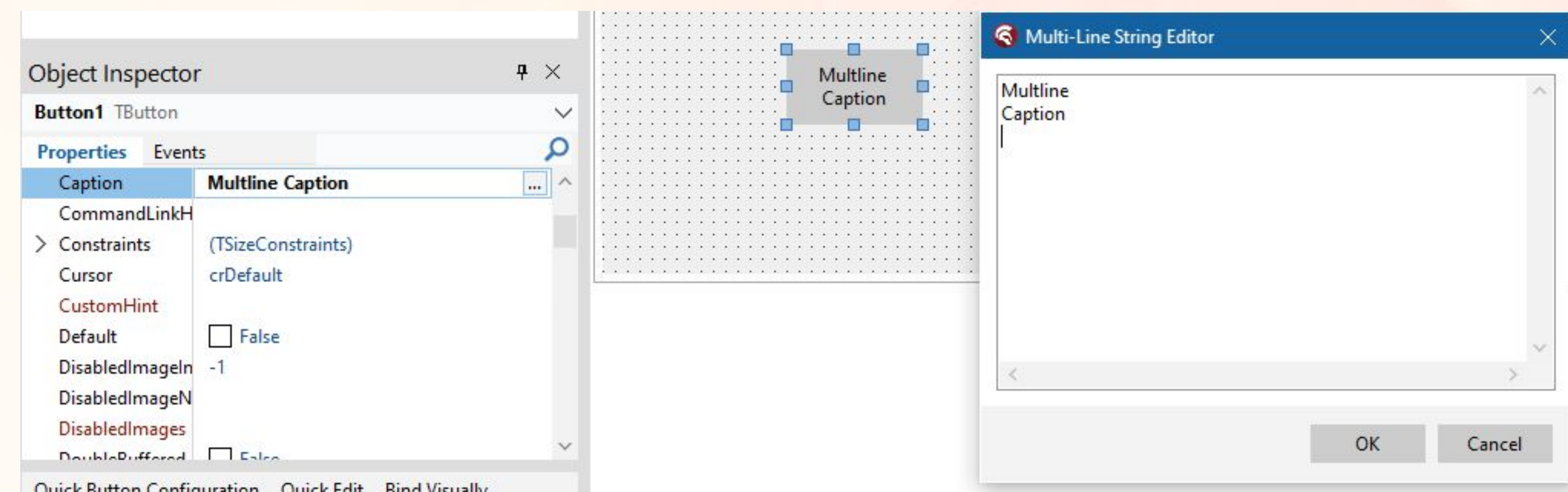
New StringList Editor

- Toolbar to load and save to text files, standard clipboard operations, plus a status bar with position information



Multiline String Editor

- Offering the ability to display multiple lines of text in their Caption (or other) properties



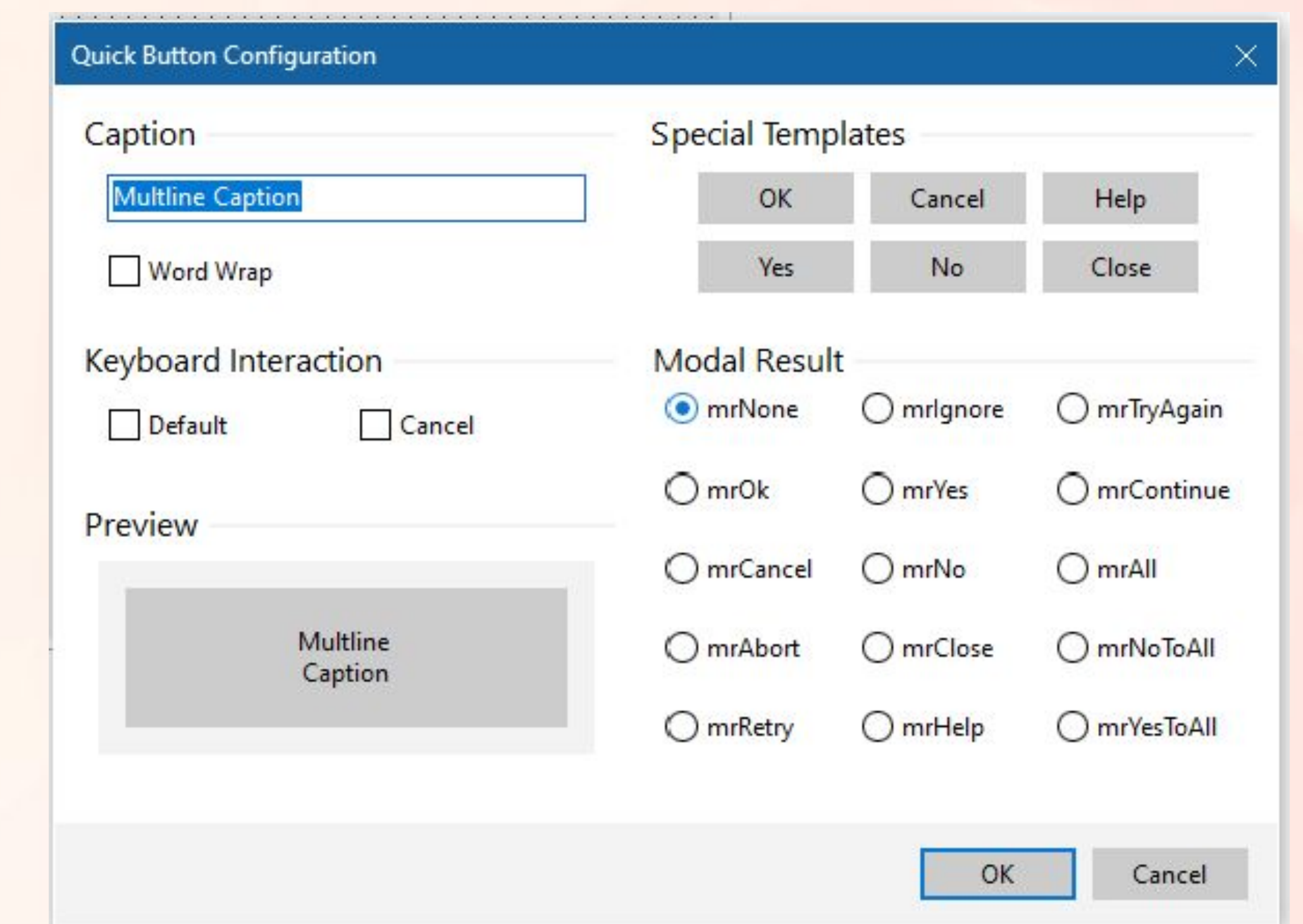
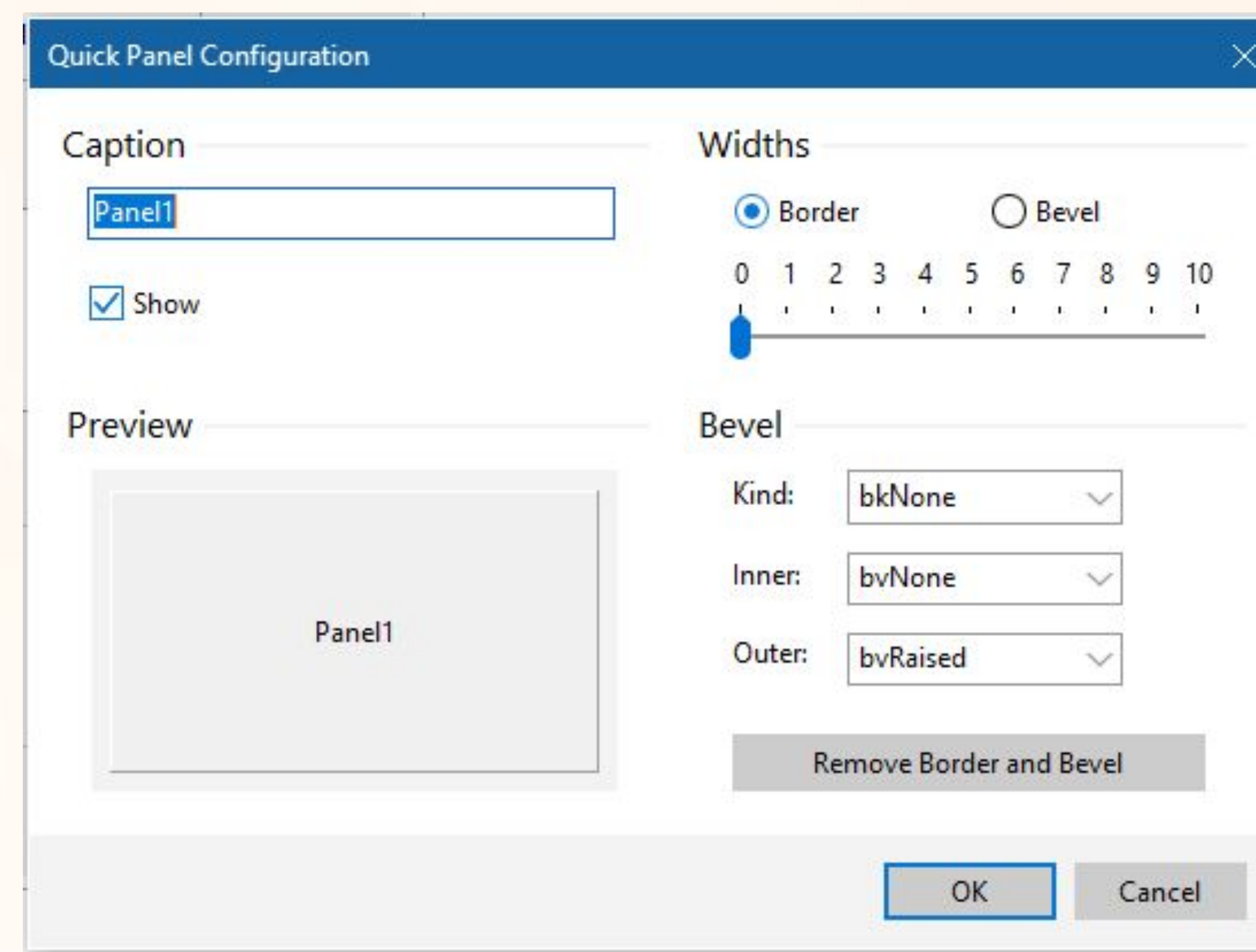
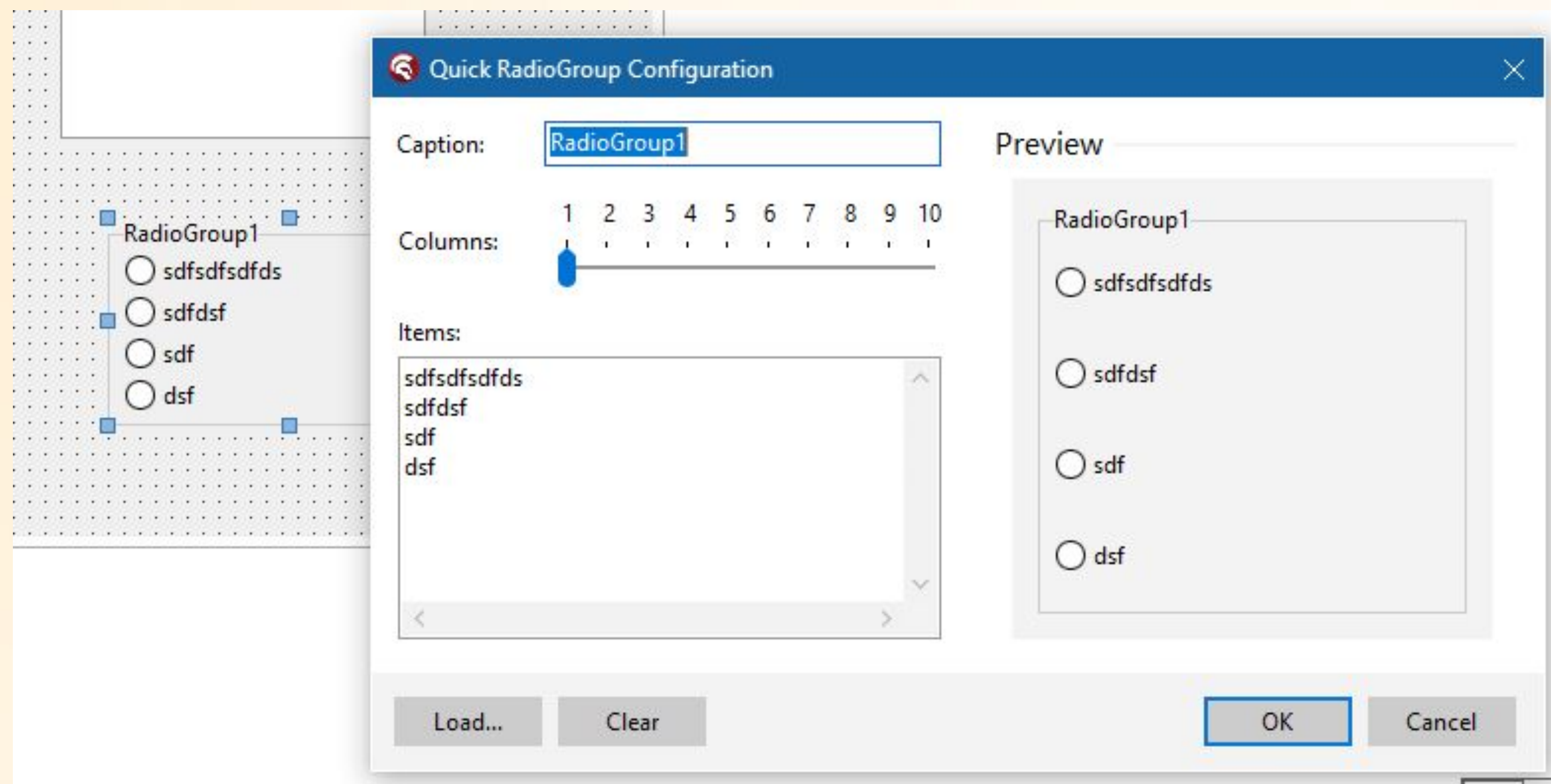
New VCL Quick Editors (from Ksvc)

Similar to other “Quick” editors already in the IDE

Quick RadioGroup
Configuration

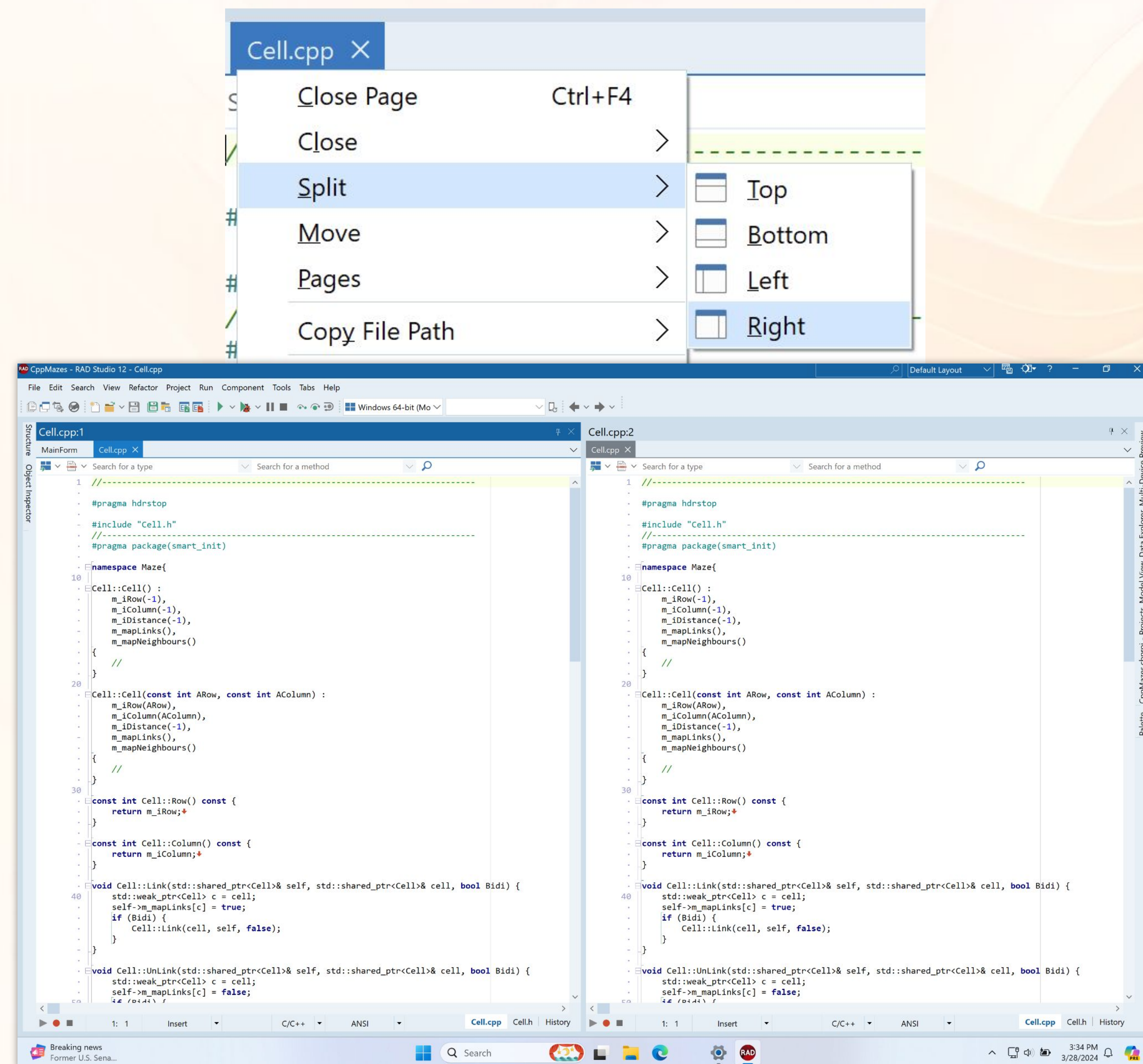
Quick Panel
Configuration

Quick Button
Configuration



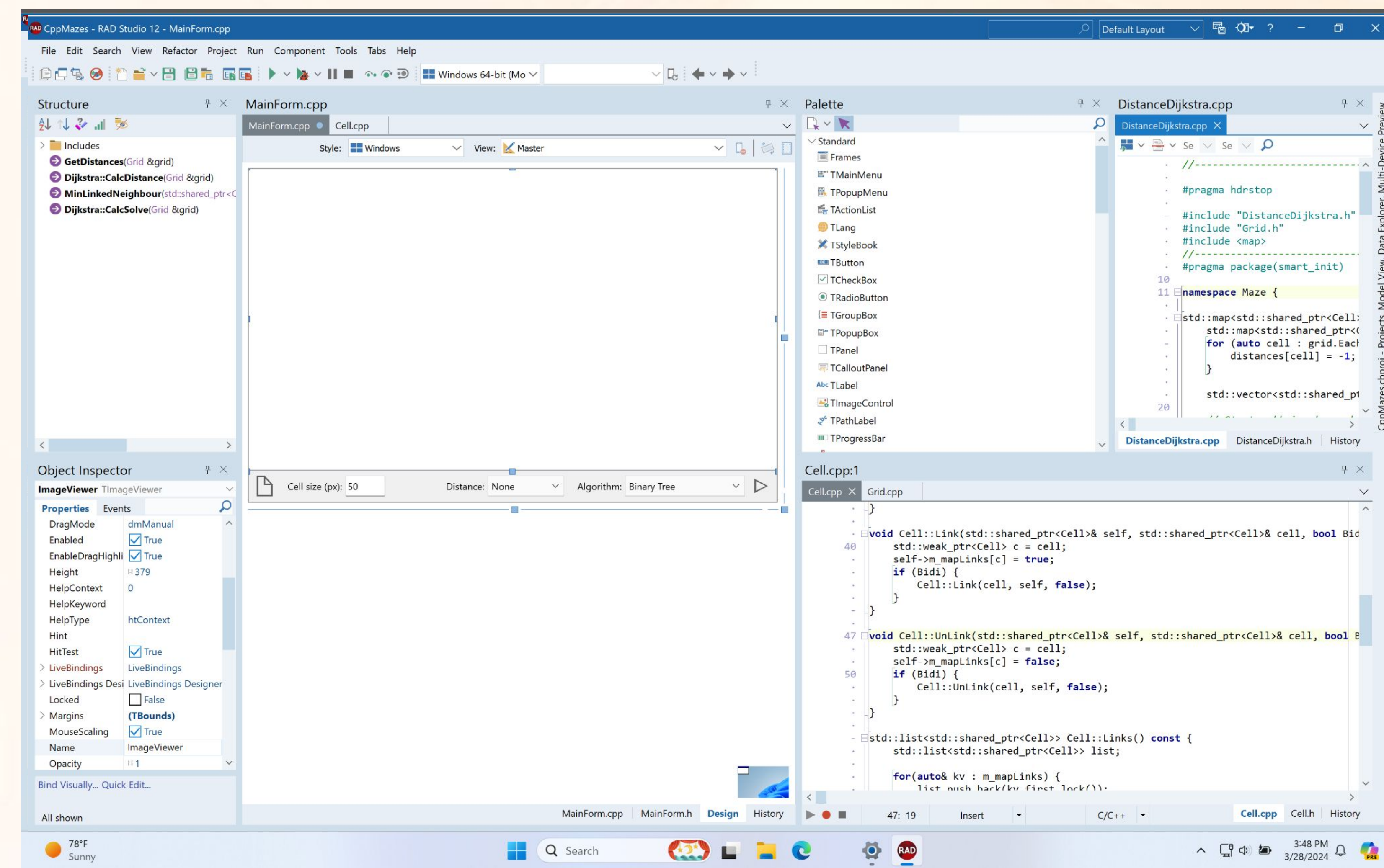
Editor split views

- Multiple editors (**tab groups!**) side by side
 - Or above/below each other
 - Or any combination!
- Split editors allow **same file** to be edited next to itself
 - Allowing you to edit the same file in multiple locations in code
 - Keep both **design & code** on screen
- Split editors allow many files in many **tab groups**, docked, windowed...



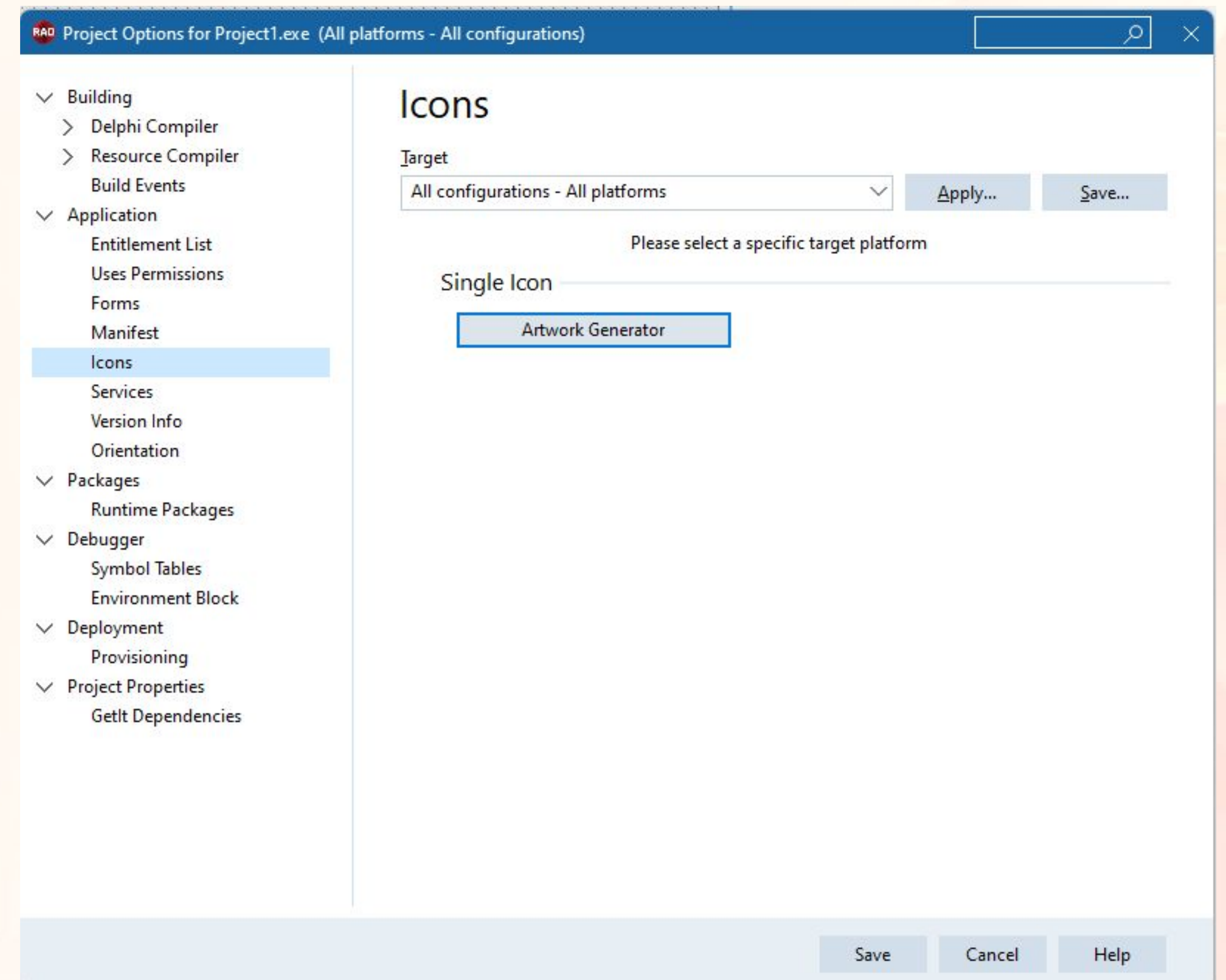
Why editor split views?

- Productivity:
 - Incredibly useful for coding
 - More than one editor in any IDE window
 - See two or more things at once
- Very useful for code and form design at the same time, great for the developers who miss the old undocked designer
- Usability
 - Focus is clear - solid blue bar



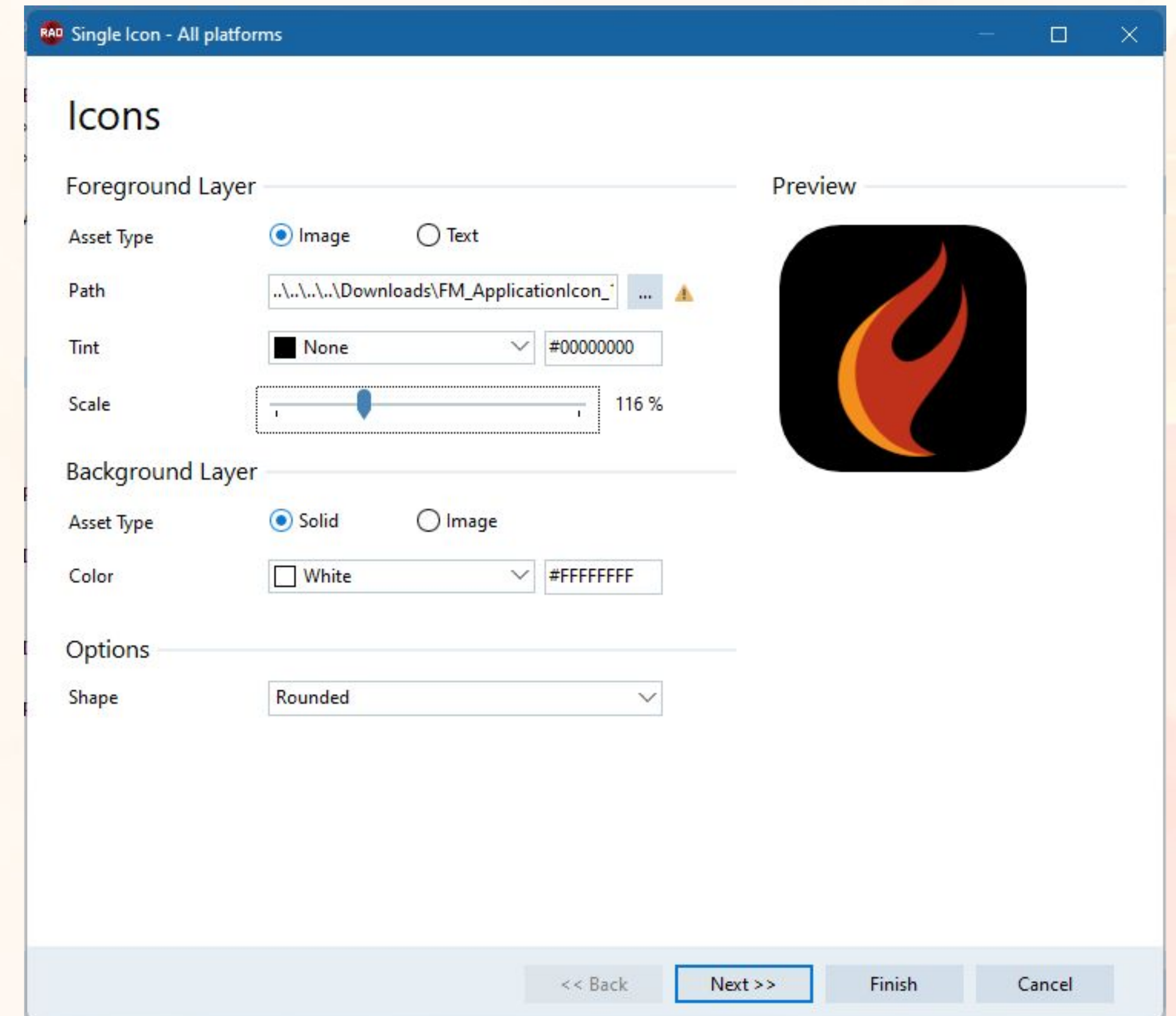
Multi Device Icon Generator

- Generates icons for all screen resolutions
- Generates splash screens
- Generates Android's adaptive icon
- Embedded in the IDE as a multi step wizard



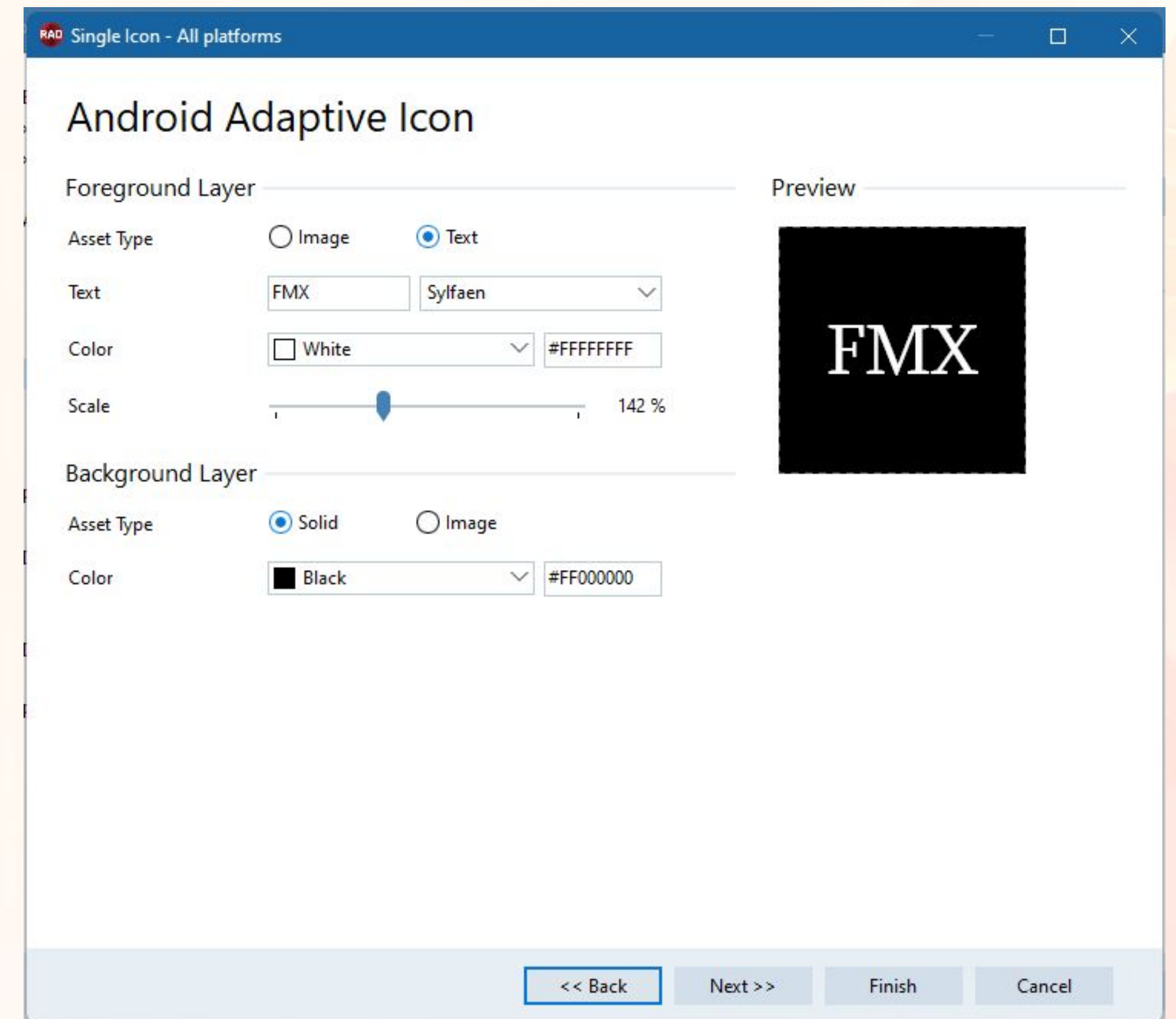
Icon Configuration

- Can be created based on a image or text
- Color and scale can be adjusted
- Icon can be rounded, square...



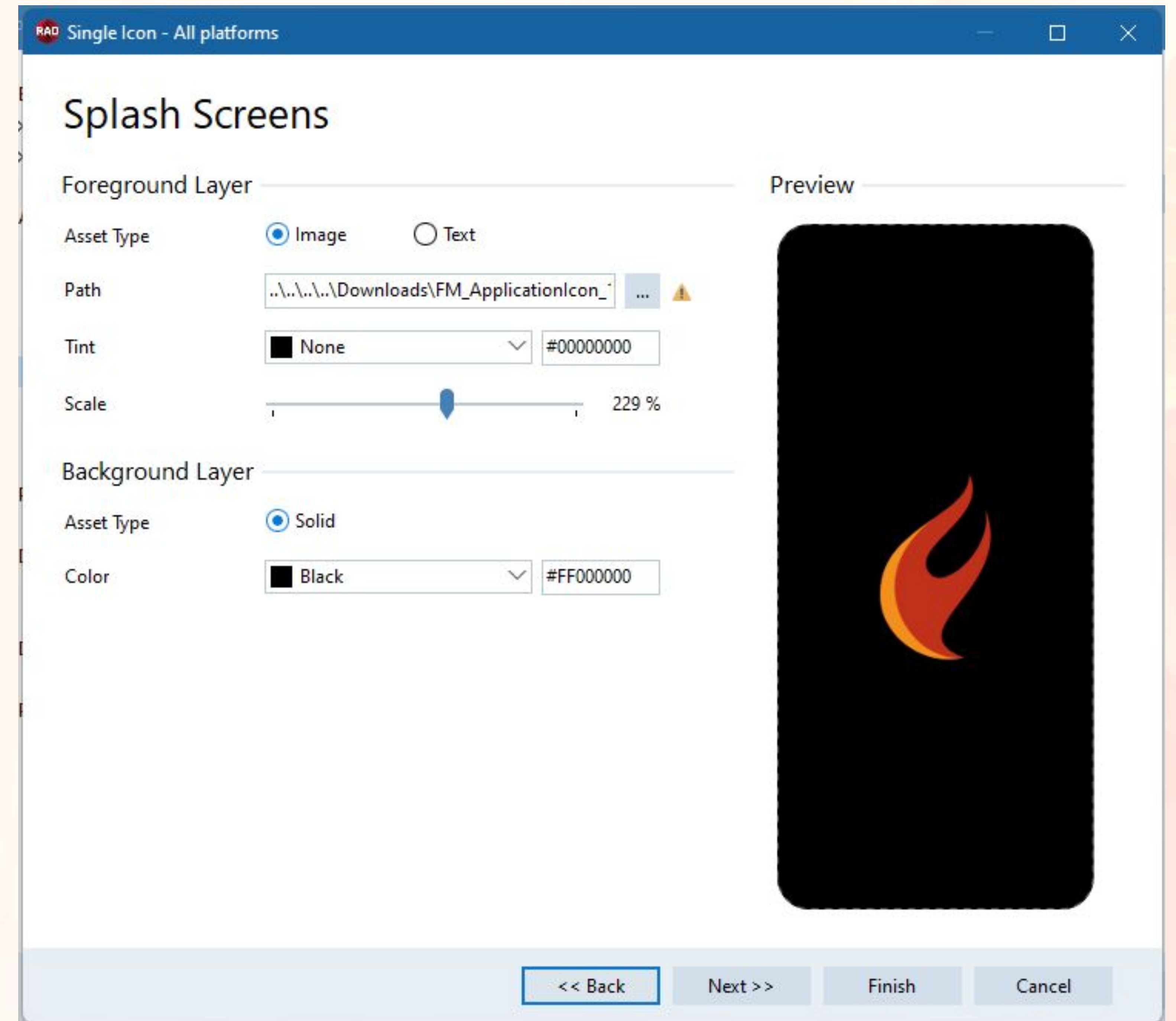
Android Adaptive Icon

- Similar options to a standard icon
- If you want to use this option with an image, you will need to use a SVG



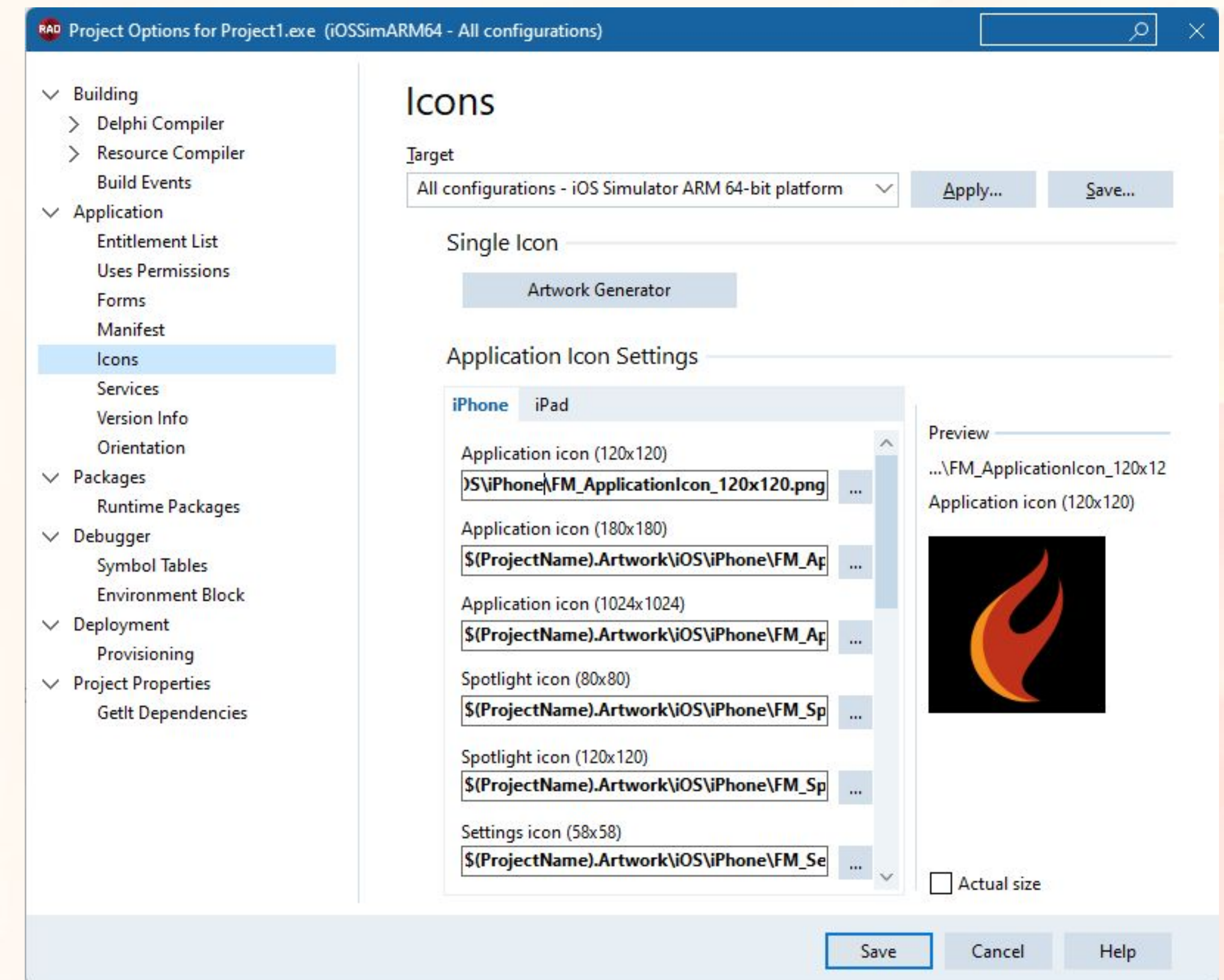
Splash Screens

- Based on your upload image
- Generate splash screens for light and dark theme



Multi Device Icon Generator

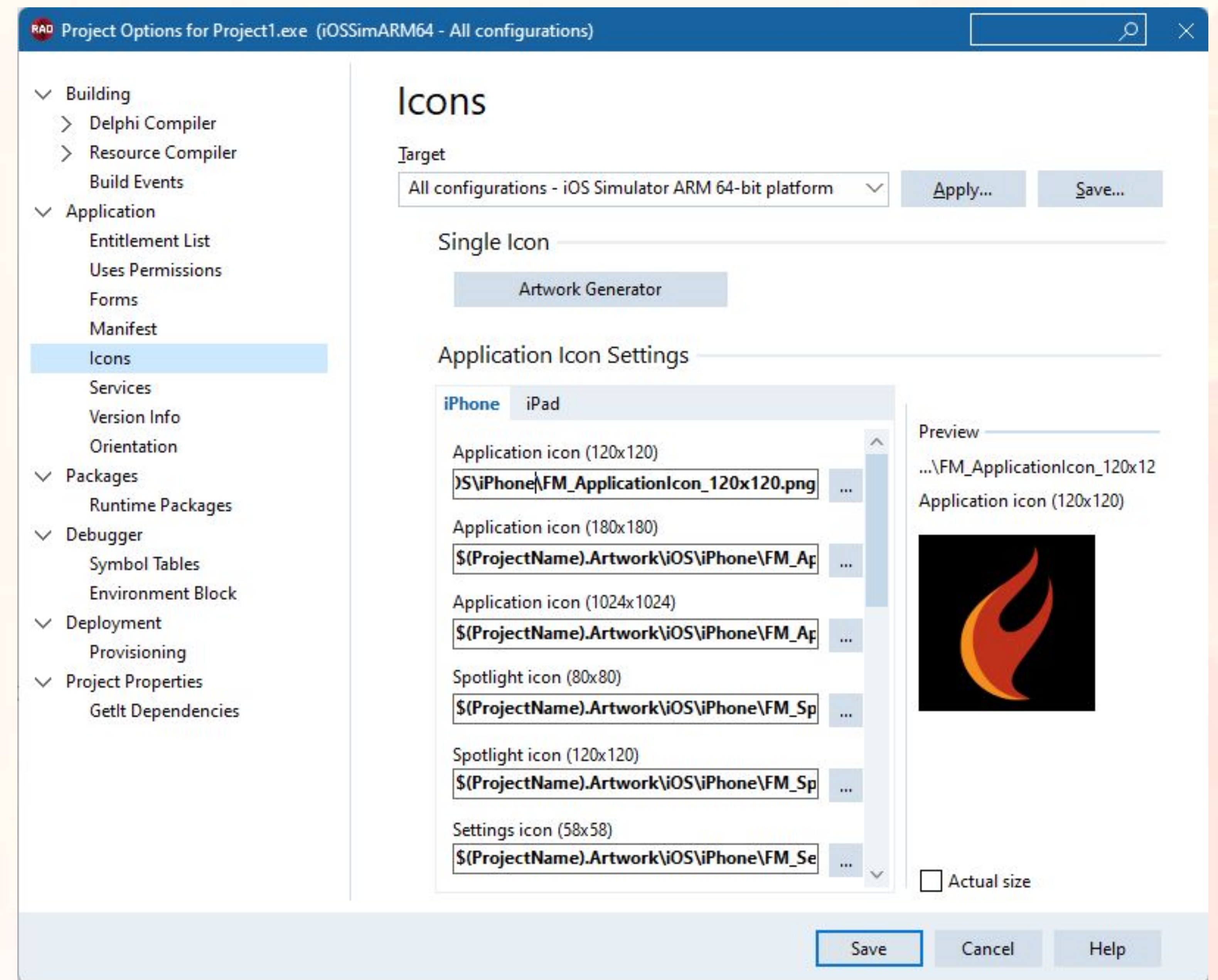
- All icons can be generated at once or individually
- Changes to single icons for specific platforms can be made easily



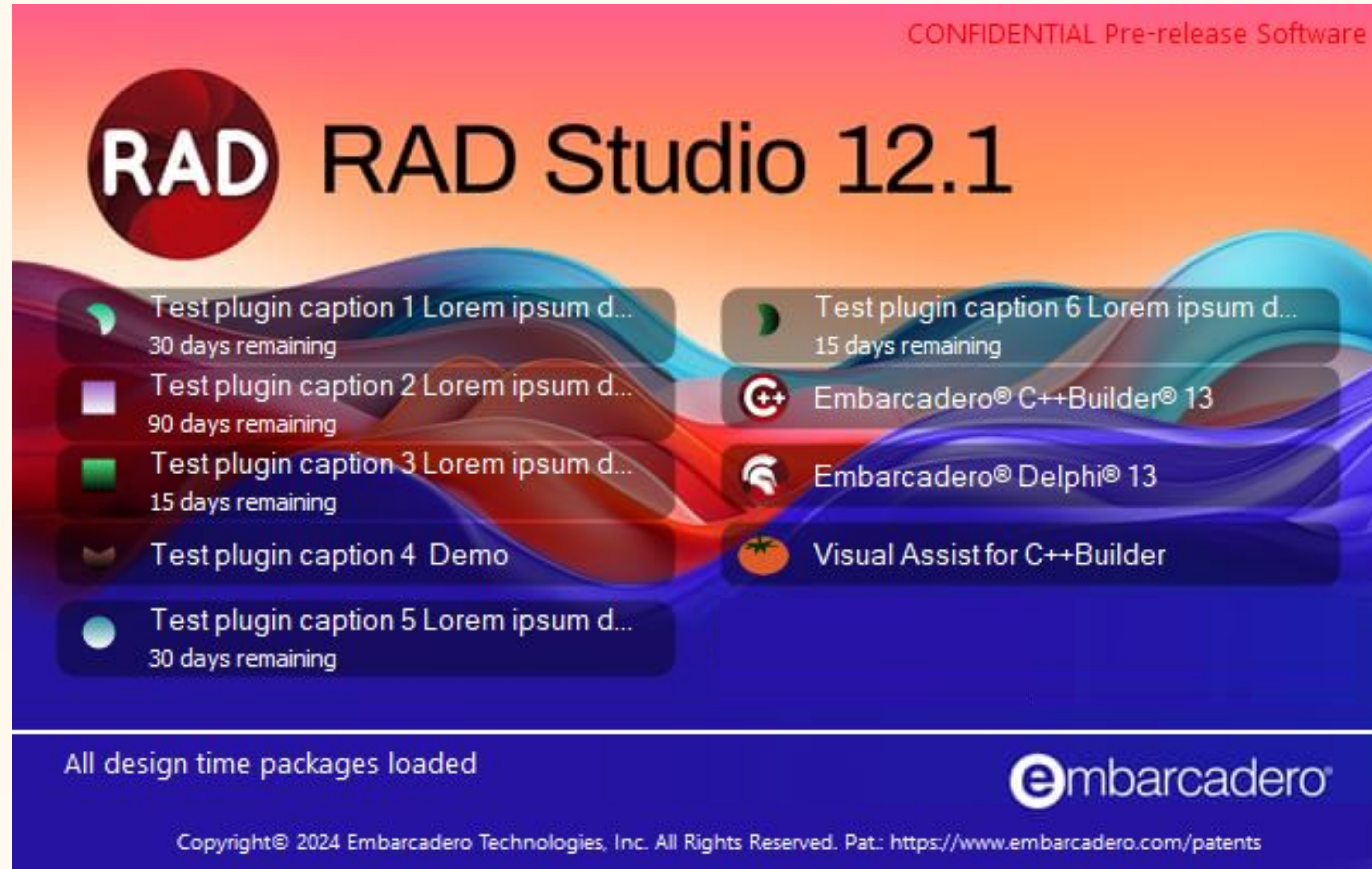
Multi Device Icon Generator

- All icons can be generated at once or individually
- Changes to single icons for specific platforms can be made easily

Keep in mind that the generated images are saved in a sub-folder of the project folder with the project name. Therefore, if you rename the project, you must manually rename the subfolder accordingly.



Splash with multiple items



Today, this expands to three columns, and eventually a horizontal scrollbar.

Debuggers

The new C++ Win64 platform in Preview uses LLDB 15 and PDB debug format, making it much more accessible to other tooling

Other changes:

- PAServer messages show in the Messages pane
- OS thread handle shows in the Threads view
- ‘Debugger Not Responding’ improvements: used to be a single long timeout; now much shorter, but grows each time you press the new Wait button



Win64 Clang Toolchain

New Win64 Clang compiler

- Building a new C++ Win64 compiler based on a recent version of Clang
- Support for the most recent C++ language features and a much more stable compiler and toolchain
- A new foundation for the future of our C++ compilers
- In 12.0 we shipped a Preview of this new C++ toolchain for Win64, available as a command line compiler
- in 12.1 the compiler became a full feature, with IDE integration and libraries support

C++ Clang 15 Compiler Goals

- Modern C++ standards
 - Run C++ code you find online, use third party libraries
 - Use modern safe coding standards
 - Write code you expect to work and have it work
- Performant apps
- Excellent debugging
- Excellent toolchain tools, eg the linker
- Matches platform standards (eg COFF and PDB)
- Supports existing code with C++Builder-isms and links with RAD Studio Delphi libraries (of course!)

Clang Upgrade in C++Builder 12.1

A new foundation for C++Builder and RAD Studio

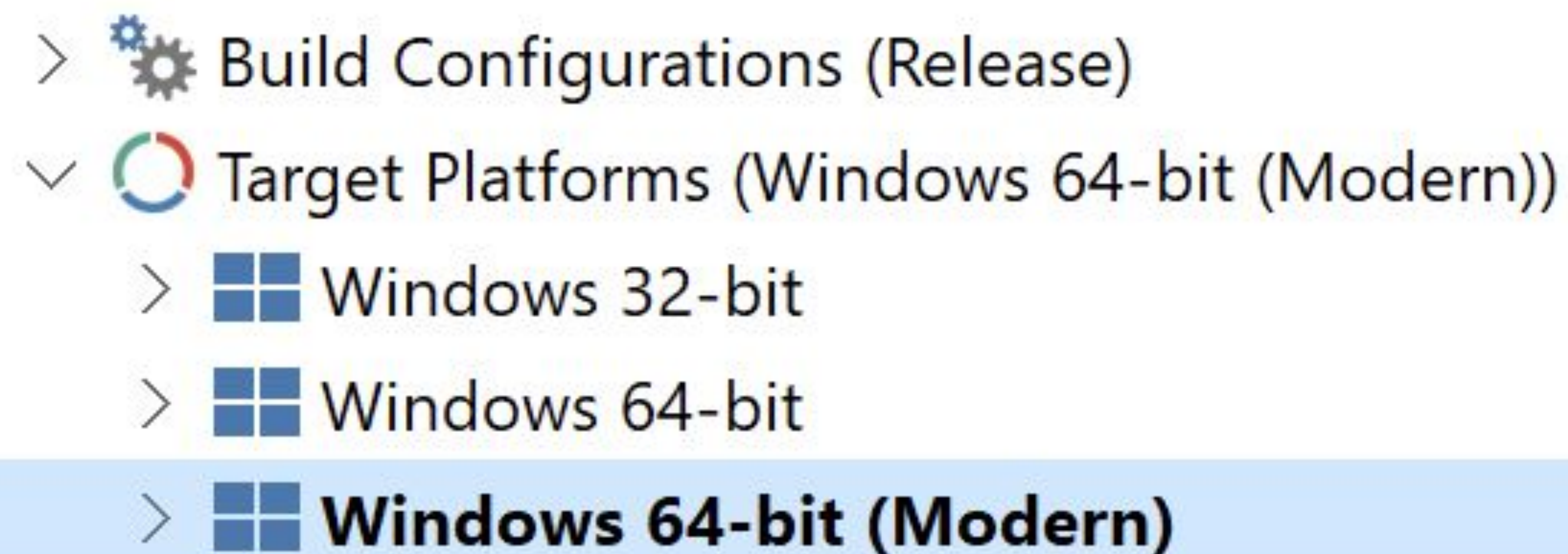
- New version of Clang
- Revision of the entire toolchain
- Focus on *platform conventions*
- Focus on *doing it right*
- Focus on *quality*
 - STL, linker etc: a must-use



The value of C++Builder with a high quality toolchain for modern C++

Clang Upgrade in C++Builder 12.1

- Shipping the first official version
 - Targets Win64
 - Existing Win64 compiler remains available in-IDE to help migration
 - Build your VCL and FMX apps
 - New toolchain: compiler, linker, STL, C++ RTL
 - Excellent compatibility with C++ standards and 3rd party libraries
 - Very exciting: performance, standards, quality



The value of C++Builder with a high quality toolchain for modern C++

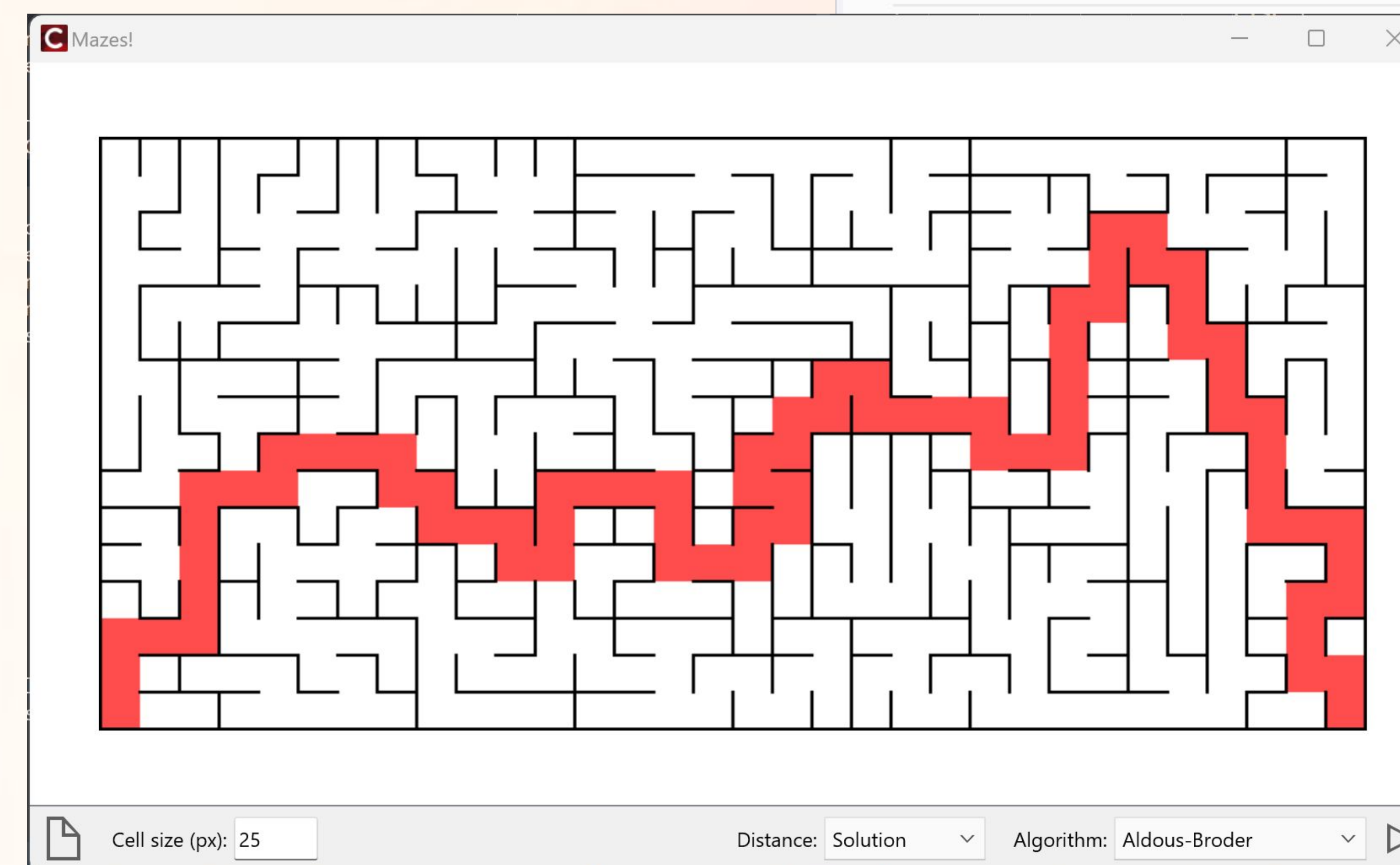
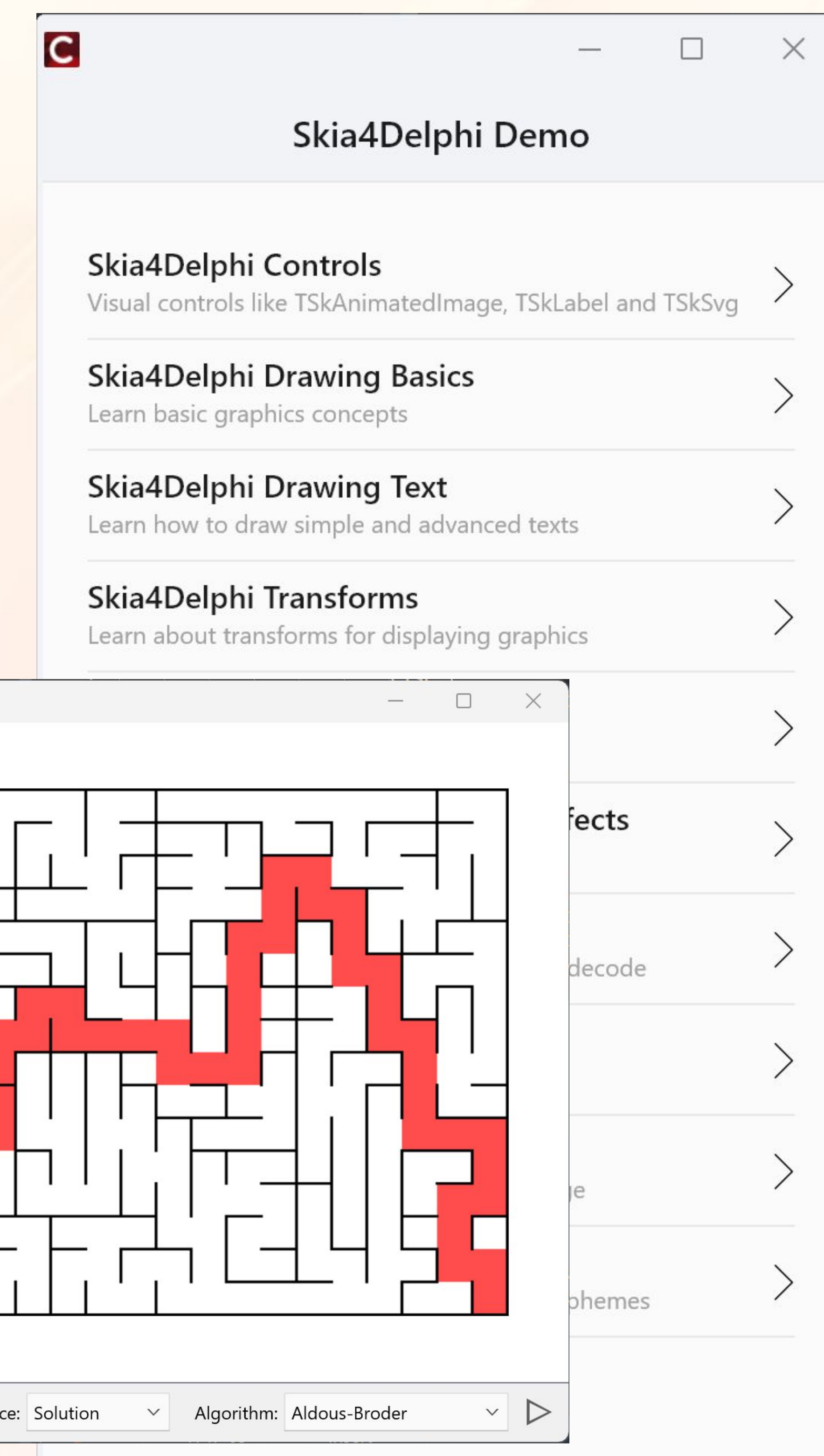
Technical Info of the new Clang Toolchain

- General goal: follow Windows platform standards
 - Leads to COFF, UCRT, PDB, etc
- Uses the Itanium C++ ABI
- COFF object file format, PDB debug format
 - Not link-compatible with a VC++ COFF object file, different C++ ABI
(goal: *source compatible*, handle any C++ code)
- C RTL: Uses the Windows component: Universal C Runtime (UCRT)
- C++ RTL, with enormous amount of exception handling work - 700+ new tests + more
- STL: LLVM's libc++
- Linker: LLVM's lld (used for Chrome)
- 64-bit binaries (even in IDE) – enormous memory space

Clang Toolchain: VCL and FMX apps

- VCL applications
- FMX applications
- DLLs, LIBs, console apps, etc
 - Can be 'pure C++' for these if you wish
- In 12.0, Delphi packages (components) are statically linked – but much faster linker to make up for it
 - Ie the Delphi default! Never had a fast enough linker to do it before
 - These smaller features, like dynamic package linking and CMake support, coming soon

Skia in VCL used from C++



FMX 'Mazes' app

Clang Upgrade: Why It Matters to You!

Quality

- Linker problems? (Eg memory issues) -> *Use this toolchain*
- STL problems? (Using 3rd party C++ code or modern C++ code?) -> *Use this toolchain*
 - Much better STL! Things that didn't work before, do work now -> *Use this toolchain*
- Compiler stability (ICEs) etc -> *Use this toolchain*

Compatibility

- Want to use other tools? (Eg other debuggers) -> *Use this toolchain*
- Use C++ code from anywhere -> *Use this toolchain*

Modernisation

- Clang 15, much more recent.

Performance

- Linker about 4x faster -> *Use this toolchain*

Feb 2024: Behind the Build for 12.1:

[youtube.com/watch?](https://youtube.com/watch?v=Ps5pW5uhmMw)

[v=Ps5pW5uhmMw](https://youtube.com/watch?v=Ps5pW5uhmMw)

Our foundation for C++'s future – here for you now.



Delphi Language and RTL

Delphi Target Platforms

- Android 14
 - Android 10, 11, 12, 13
- iOS 17
 - iOS 16, no debugging support for iOS 17
- macOS Sonoma
 - macOS Ventura and Monterey
- Windows 11
 - Also Windows 10, 7 (SP1 +)
 - Windows Server 2019 and 2022
- Ubuntu 22
 - Ubuntu 20.04, RedHat 8, WSL2




Delphi Extended String Literals

- **Long String Literals**

- String literals not limited to 255 chars any more

- **Multiline String Literals**

- String literals encompassing multiple lines
- Delimited by a triple quote (""")
 - Or by five quotes to embed a triple quote
- Delimiters on separate lines
- Closing delimiter determines indentation
- `$TEXTBLOCK` compiler directive to control line breaks (CR, LF, CRLF, Platform Native)



```
27 9012345678901234567890123456789012345678901234567890; // 600 chars string
30
```



```
26 {$TEXTBLOCK CRLF JSON}
  const
  strJSON = '''
    [
30     {"id" : "1", "name" : "Large"},
     {"id" : "2", "name" : "Medium"},
     {"id" : "2", "name" : "Small"}
    ]
  ''';
```


More Delphi Language Improvements

NativeInt as a Weak Alias

- NativeInt becomes a “weak alias” of Integer or Int64, depending on the platform
- You cannot use it any more as a separate type: You get overload error on declaration, not ambiguous error on call
- Related: Further improvements in RTL support for Win64 types as needed

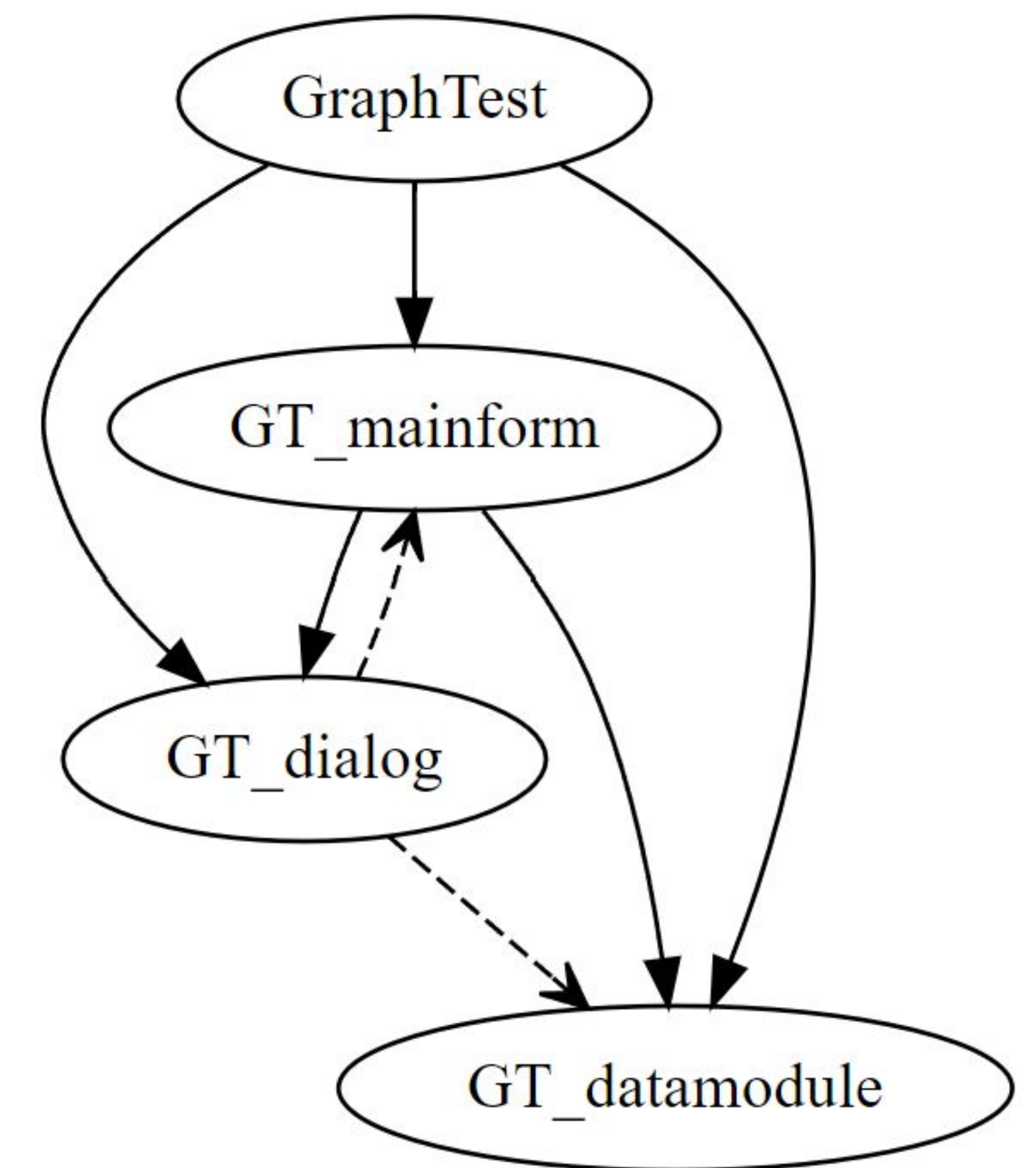
Support **NaN comparisons** as required by IEEE (Win32 compiler)

Improved warnings in generic classes (like in the non-generic counterpart)

Circular Uses Statements in Delphi

GraphViz representation

- Helps understanding “implementation” circular unit references, which put some stress on the compiler
- Options to export a GraphViz file with the complete unit dependency (including circular references in the implementation, marked as dashed lines)
- You can exclude unit families



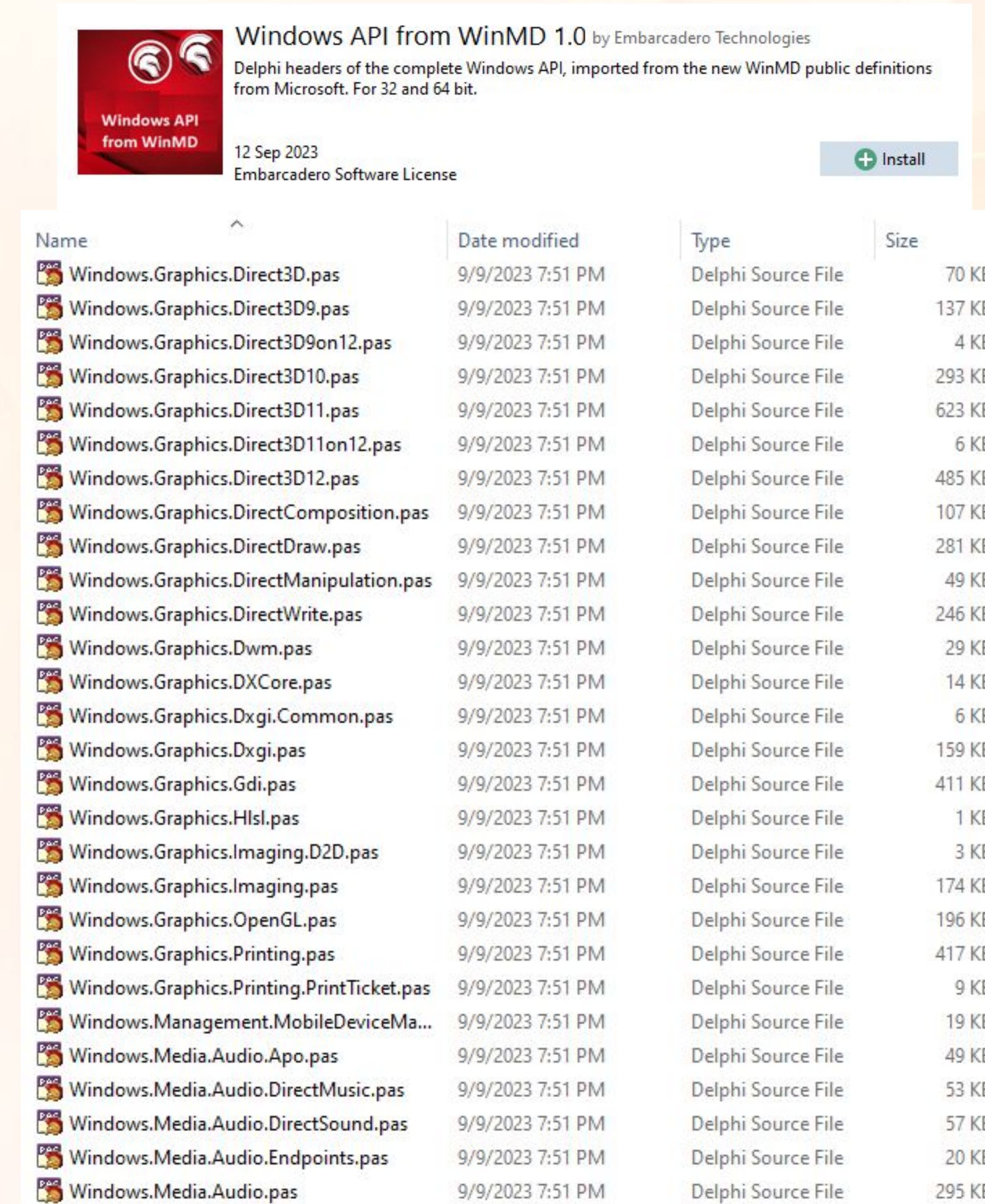
Complete Windows API Integration

Microsoft has (finally) provided complete metadata for the classic Windows API in WinMD format

- WinMD metadata available at <https://github.com/microsoft/win32metadata>
- We used it to generate over 300 units with 41 MB of Delphi code
- Available in GetIt

We have also refreshed to the latest version:

- The WinRT / Windows App SDK API
- The WebView 2 control API



Windows API from WinMD 1.0 by Embarcadero Technologies
Delphi headers of the complete Windows API, imported from the new WinMD public definitions from Microsoft. For 32 and 64 bit.

12 Sep 2023
Embarcadero Software License

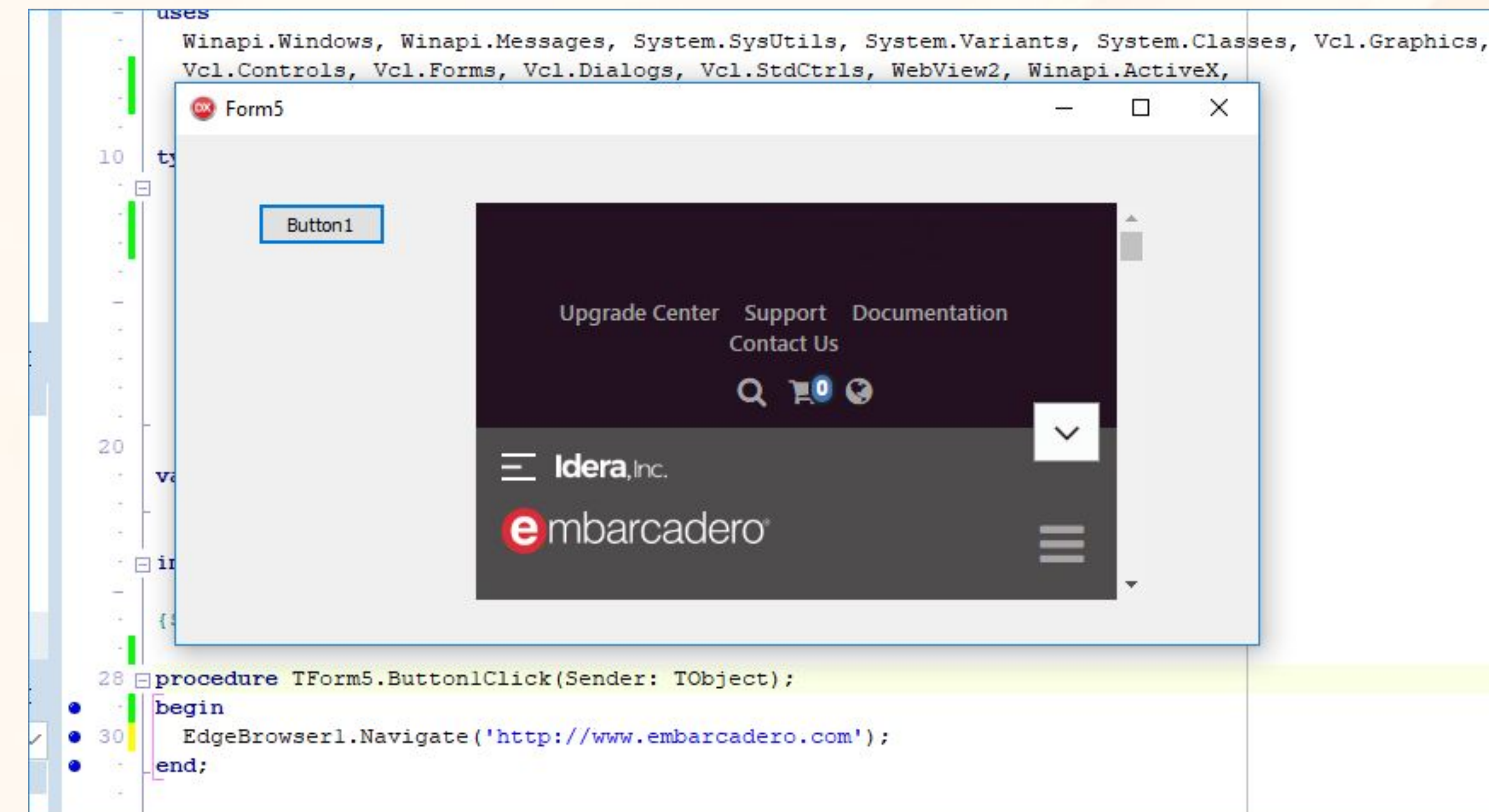
+ Install

Name	Date modified	Type	Size
Windows.Graphics.Direct3D.pas	9/9/2023 7:51 PM	Delphi Source File	70 KB
Windows.Graphics.Direct3D9.pas	9/9/2023 7:51 PM	Delphi Source File	137 KB
Windows.Graphics.Direct3D9on12.pas	9/9/2023 7:51 PM	Delphi Source File	4 KB
Windows.Graphics.Direct3D10.pas	9/9/2023 7:51 PM	Delphi Source File	293 KB
Windows.Graphics.Direct3D11.pas	9/9/2023 7:51 PM	Delphi Source File	623 KB
Windows.Graphics.Direct3D11on12.pas	9/9/2023 7:51 PM	Delphi Source File	6 KB
Windows.Graphics.Direct3D12.pas	9/9/2023 7:51 PM	Delphi Source File	485 KB
Windows.Graphics.DirectComposition.pas	9/9/2023 7:51 PM	Delphi Source File	107 KB
Windows.Graphics.DirectDraw.pas	9/9/2023 7:51 PM	Delphi Source File	281 KB
Windows.Graphics.DirectManipulation.pas	9/9/2023 7:51 PM	Delphi Source File	49 KB
Windows.Graphics.DirectWrite.pas	9/9/2023 7:51 PM	Delphi Source File	246 KB
Windows.Graphics.Dwm.pas	9/9/2023 7:51 PM	Delphi Source File	29 KB
Windows.Graphics.DXCore.pas	9/9/2023 7:51 PM	Delphi Source File	14 KB
Windows.Graphics.Dxgi.Common.pas	9/9/2023 7:51 PM	Delphi Source File	6 KB
Windows.Graphics.Dxgi.pas	9/9/2023 7:51 PM	Delphi Source File	159 KB
Windows.Graphics.Gdi.pas	9/9/2023 7:51 PM	Delphi Source File	411 KB
Windows.Graphics.Hlsl.pas	9/9/2023 7:51 PM	Delphi Source File	1 KB
Windows.Graphics.Imaging.D2D.pas	9/9/2023 7:51 PM	Delphi Source File	3 KB
Windows.Graphics.Imaging.pas	9/9/2023 7:51 PM	Delphi Source File	174 KB
Windows.Graphics.OpenGL.pas	9/9/2023 7:51 PM	Delphi Source File	196 KB
Windows.Graphics.Printing.pas	9/9/2023 7:51 PM	Delphi Source File	417 KB
Windows.Graphics.Printing.PrintTicket.pas	9/9/2023 7:51 PM	Delphi Source File	9 KB
Windows.Management.MobileDeviceMa...	9/9/2023 7:51 PM	Delphi Source File	19 KB
Windows.Media.Audio.Apo.pas	9/9/2023 7:51 PM	Delphi Source File	49 KB
Windows.Media.Audio.DirectMusic.pas	9/9/2023 7:51 PM	Delphi Source File	53 KB
Windows.Media.Audio.DirectSound.pas	9/9/2023 7:51 PM	Delphi Source File	57 KB
Windows.Media.Audio.Endpoints.pas	9/9/2023 7:51 PM	Delphi Source File	20 KB
Windows.Media.Audio.pas	9/9/2023 7:51 PM	Delphi Source File	295 KB

Windows Platform Integration

Edge Browser

- UserAgent is available in ICoreWebView2Settings
- ICoreWebView2Profile2 contains methods ClearBrowsingData, ClearBrowsingDataAll, and ClearBrowsingDataInTimeRange
- New OnDownloadStarting event
- New NavigateWithWebResourceRequest method
- Edge component surfaces the Print and ShowPrintUI operations



Android API Level 34

Android API level 34 support

- Updating the supported Android API to level 34
 - This has been done in advance of the August 2024 deadline for Google Play Store app submission
 - The default `targetSdkVersion` manifest attribute is now 34
- This change required (more technical details in next slide)
 - Updates to the Android SDK
 - Introduction of some additional Android platform tools
 - Update of the Java runtime
 - Improvements to the app packaging process
 - Updates to the uses permissions
 - For example: detect screen capture, foreground services

Android API level 34 support (more info)

- Eclipse Temurin JDK OpenJDK 17 (Hotspot) JVM
- Command-line Tools Android SDK components from Android SDK
- Improvements in the Android packaging:
 - Android DEX operations (DEX compilation and DEX merging) are now incremental via a new command-line tool called dexter
 - Use Google's AAPT2 tool (more control over the packaging process)
 - Packager command-line tool to package resource files, etc
 - New packaging project options
- Noticeable performance boost in the generation of .apk files
 - Due to the use of Google's zipflinger and signflinger libraries

Android API level 34: Why Does it Matter to You?

- We are supporting a future Google requirement for the store
- We have made packaging apps and deploying them to Android higher quality and faster, using the most recent tooling from Google and the platform SDK – the same tools the “native” Android Studio uses

UI Libraries: FireMonkey

What's New in FireMonkey? Really a lot!

- Skia Graphic Library library support
 - For all target platform, both Delphi and C++Builder
- Completed the redesign of styled TEdit/TMemo
- Android API 33 Support (see later)
- New Icon and Splash screen designer (covered earlier)
- Split Views for mobile platforms
- Plus smaller features and quality

Skia and FireMonkey

Why Skia?

Skia is a multi-platform 2D graphics library

Skia offers high performance and high quality rendering

Skia supports advanced graphic operations across all platforms

Skia offers support for many image and video formats

Skia is becoming the new foundation for FireMonkey rendering

What is Skia?

“Skia is an open source 2D graphics library which provides common APIs that work across a variety of hardware and software platforms. It serves as the graphics engine for Google Chrome and ChromeOS, Android, Flutter, and many other products.” - skia.org



Skia logo source: https://en.wikipedia.org/wiki/Skia_Graphics_Engine

How we leverage Skia?

“Skia4Delphi is a cross-platform 2D graphics API for Delphi and C++Builder based on Google's Skia Graphics Library. Provides common 2D APIs by abstracting complexities in implementing low-level libraries used behind, such as OpenGL, Vulkan, DirectX, and Metal, among others”



Main Features of Skia (1/2)

Feature	Details
2D drawing	Shapes, paths, and texts
SVG	Render and creation
Image decoders	BMP, GIF, ICO, JPG, PNG, WBMP, WEBP, and raw images
Image encoders	PNG, JPG, and WEBP
Animations player	Lottie, Telegram Stickers, animated GIFs, and animated WEBP
Anti-aliasing	High draw quality, no jagged edges
Font	Font weight, families fallbacks, and custom font (without installation), ligature

Main Features of Skia (2/2)

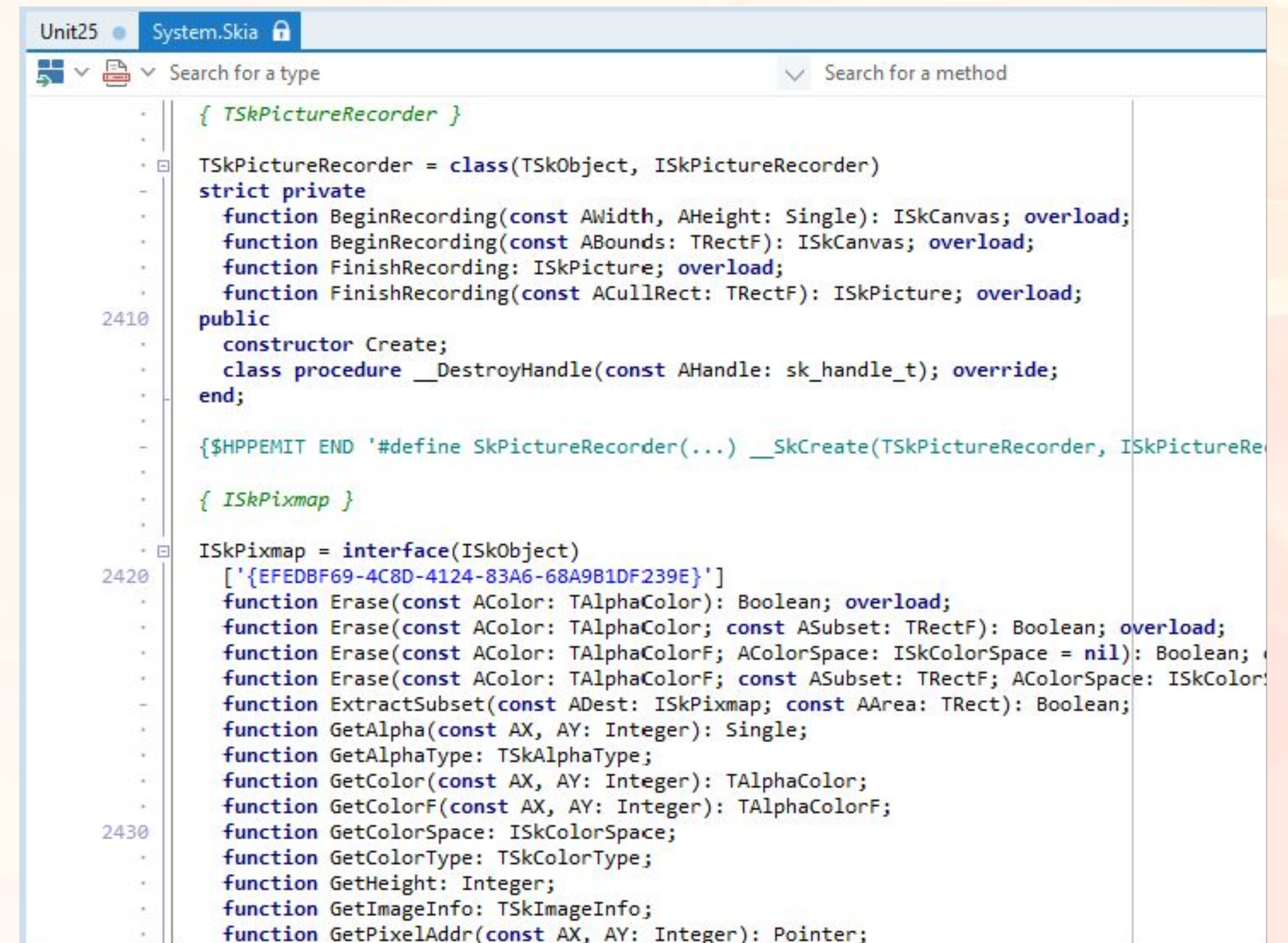
Feature	Details
Text	Multiple styles, max lines, line spacing, justified text, text outline, gradient, and decorations
Right-To-Left languages	Rendering of texts in Persian, Arabic, Hebrew, etc
PDF	Generation of vectorized PDF
Unicode	Graphemes parser
Filters	Color, mask, and image filters
Clippings	Support for many advanced clipping operations such as paths and shaders
Gradients	Linear, radial, sweep, and conical gradients
Shader	Creation of shaders to execute specific draws directly on the GPU, through a single shader language (SkSL)

Skia in RAD Studio

Different levels of usage for both Delphi and C++Builder

1. Skia API

Access to the pure Skia library, through a single unit: Skia.pas or Skia.hpp



```
Unit25 System.Skia
Search for a type Search for a method
{ TSkPictureRecorder }
TSkPictureRecorder = class(TSkObject, ISkPictureRecorder)
strict private
function BeginRecording(const AWidth, AHeight: Single): ISkCanvas; overload;
function BeginRecording(const ABounds: TRectF): ISkCanvas; overload;
function FinishRecording: ISkPicture; overload;
function FinishRecording(const ACullRect: TRectF): ISkPicture; overload;
2410 public
constructor Create;
class procedure __DestroyHandle(const AHandle: sk_handle_t); override;
end;
{$HPPEMIT END '#define SkPictureRecorder(...) __SkCreate(TSkPictureRecorder, ISkPictureRe
{ ISkPixmap }
ISkPixmap = interface(ISkObject)
2420 ['{EFEDBF69-4C8D-4124-83A6-68A9B1DF239E}']
function Erase(const AColor: TAlphaColor): Boolean; overload;
function Erase(const AColor: TAlphaColor; const ASubset: TRectF): Boolean; overload;
function Erase(const AColor: TAlphaColorF; AColorSpace: ISkColorSpace = nil): Boolean;
function Erase(const AColor: TAlphaColorF; const ASubset: TRectF; AColorSpace: ISkColor
function ExtractSubset(const ADest: ISkPixmap; const AArea: TRect): Boolean;
function GetAlpha(const AX, AY: Integer): Single;
function GetAlphaType: TSkAlphaType;
function GetColor(const AX, AY: Integer): TAlphaColor;
function GetColorF(const AX, AY: Integer): TAlphaColorF;
2430 function GetColorSpace: ISkColorSpace;
function GetColorType: TSkColorType;
function GetHeight: Integer;
function GetImageInfo: TSkImageInfo;
function GetPixelAddr(const AX, AY: Integer): Pointer;
```

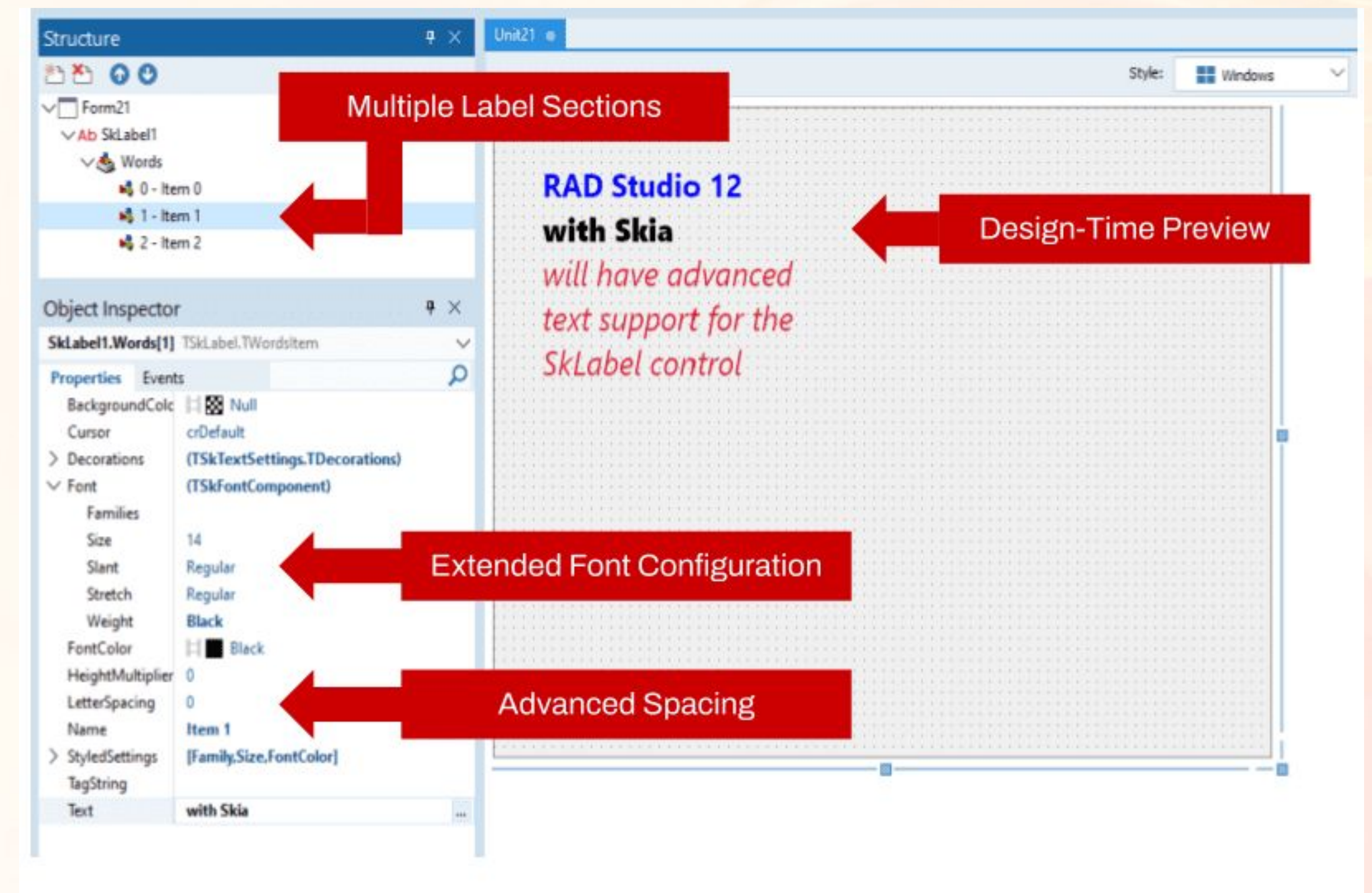

Skia in RAD Studio

Different levels of usage for both Delphi and C++Builder

2. UI Controls

(for both FMX and VCL)

- TSkAnimatedImage
- TSkLabel
- TSkPaintBox
- TSkSvg



Skia in RAD Studio

Different levels of usage for both Delphi and C++Builder

3. App and Styles Rendering

Replacement of FMX graphic engine with Skia. Better performance, improved anti-aliasing

FMX.Skia.GlobalUseSkia := True



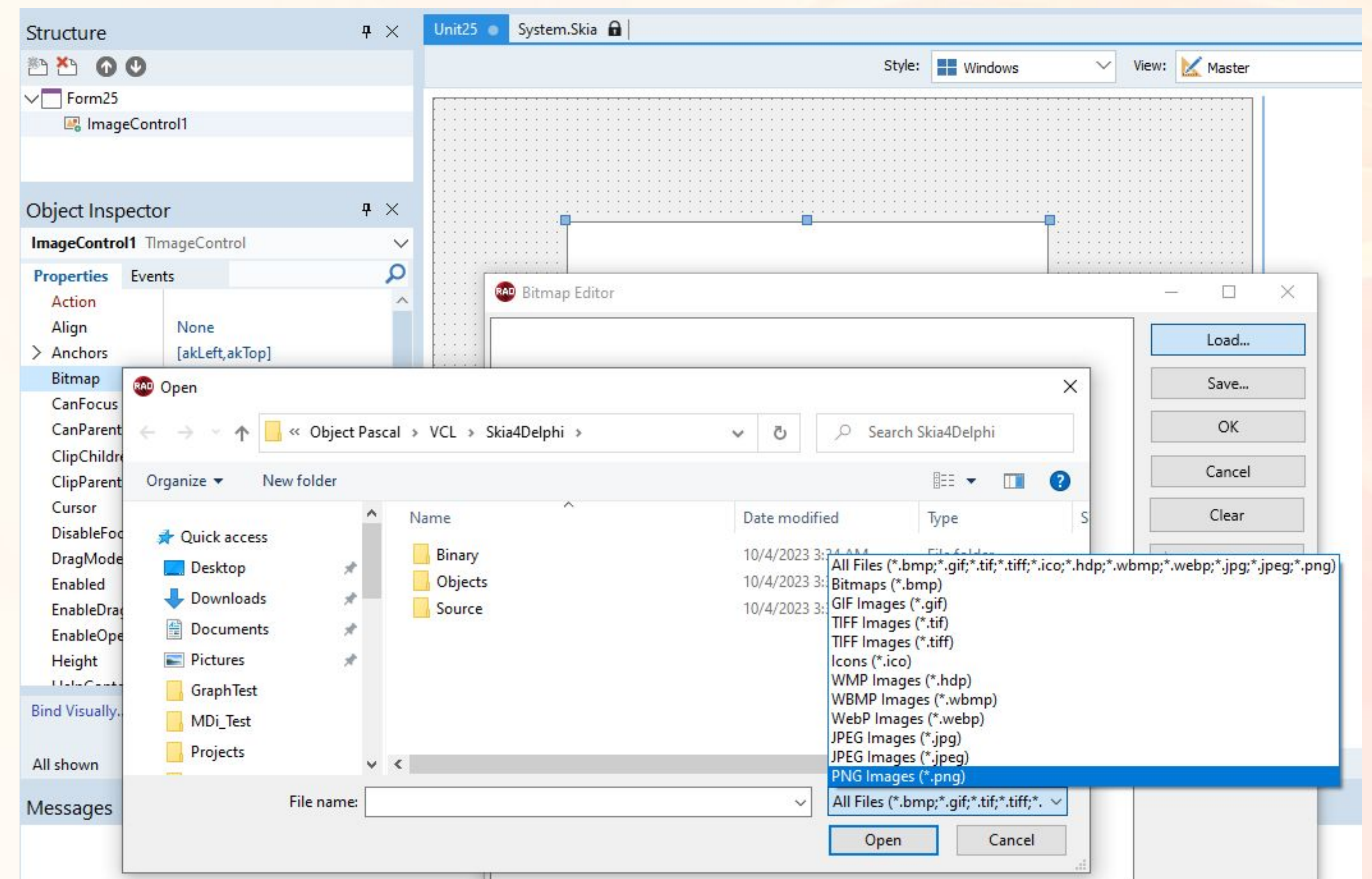
Improved drawing with anti-aliasing in Skia (on the right)

Skia in RAD Studio

Different levels of usage for both Delphi and C++Builder

4. Codecs for image controls

New image codecs Skia supports are registered in the frameworks, FMX or VCL image controls can directly load and save new image formats like WebP



New UI Controls (FMX/VCL) Based on Skia

- TSkAnimatedImage
 - Supports Lottie file, Telegram Sticker, Animated GIF, Animated WebP
- TSkLabel
 - Font weight, Font slant, multiple styles in text, BiDi (Right-to-Left)
 - Justify horizontal alignment, and much more
- TSkPaintBox & TSkAnimatedPaintBox
 - Paint with Skia APIs directly on the screen with the OnDraw event
- TSkSVG
 - Display SVG easily

FireMonkey Skia support for all platforms

For both in C++Builder and Delphi

FireMonkey, but also VCL

Leveraging the Skia4Delphi open-source project

Including additional capabilities not found in the open-source project:

- Vulkan support
 - Enhanced graphical performance and energy efficiency on Android compared to OpenGL ES
- Skia Shading Language (SKSL) for effects and filters
- WebP Encoder
- Native printer for Windows and PDF printing for all platforms

Used also for new icon and splash wizard

FireMonkey: TEdit/TMemo Rework

Significantly refactored the TEdit/TMemo controls starting with desktop platforms (Windows and macOS) and also planning to include mobile ones

General FMX changes

- New TEdit property AutoSelect
- New TEdit CharCase
- Addressed several undo/redo issues using the recently added TUndoManager
- Addressed TEdit caret position problems, particularly on Windows
- TMemo implements IME Text inputting like in native TEdit
- TEdit takes into account the foreground brush resource
- Improved Shift-clicking selection

FireMonkey: TEdit/TMemo Rework

For the iOS platform

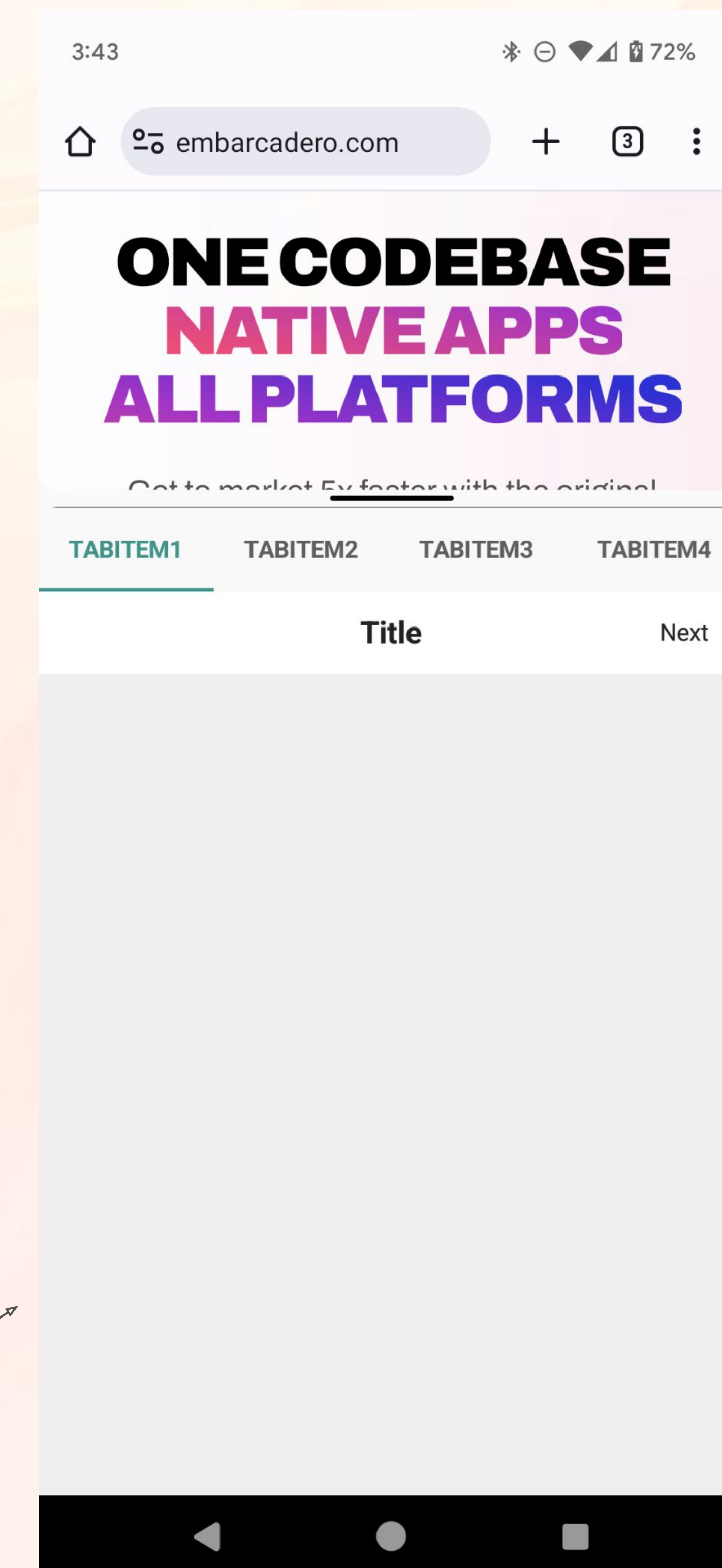
- Refactoring Virtual Keyboard: moved the logic for toolbar alignment/animation
- Improved the magnifier glass position for TMemo

For the Android platform

- Improved interactive text selection
- Adjusted selection Point position and place the left below text selection
- Added selection Point for position cursor in TMemo and TEdit
- Improved the PointInObjectLocal handles the area using a style with an offset of objects
- Updated platform and premium styles to support improved text selection
- Added CaretPoint for text-input controls for Android
- Support for finger slide gestures to move the cursor on TMemo on Android

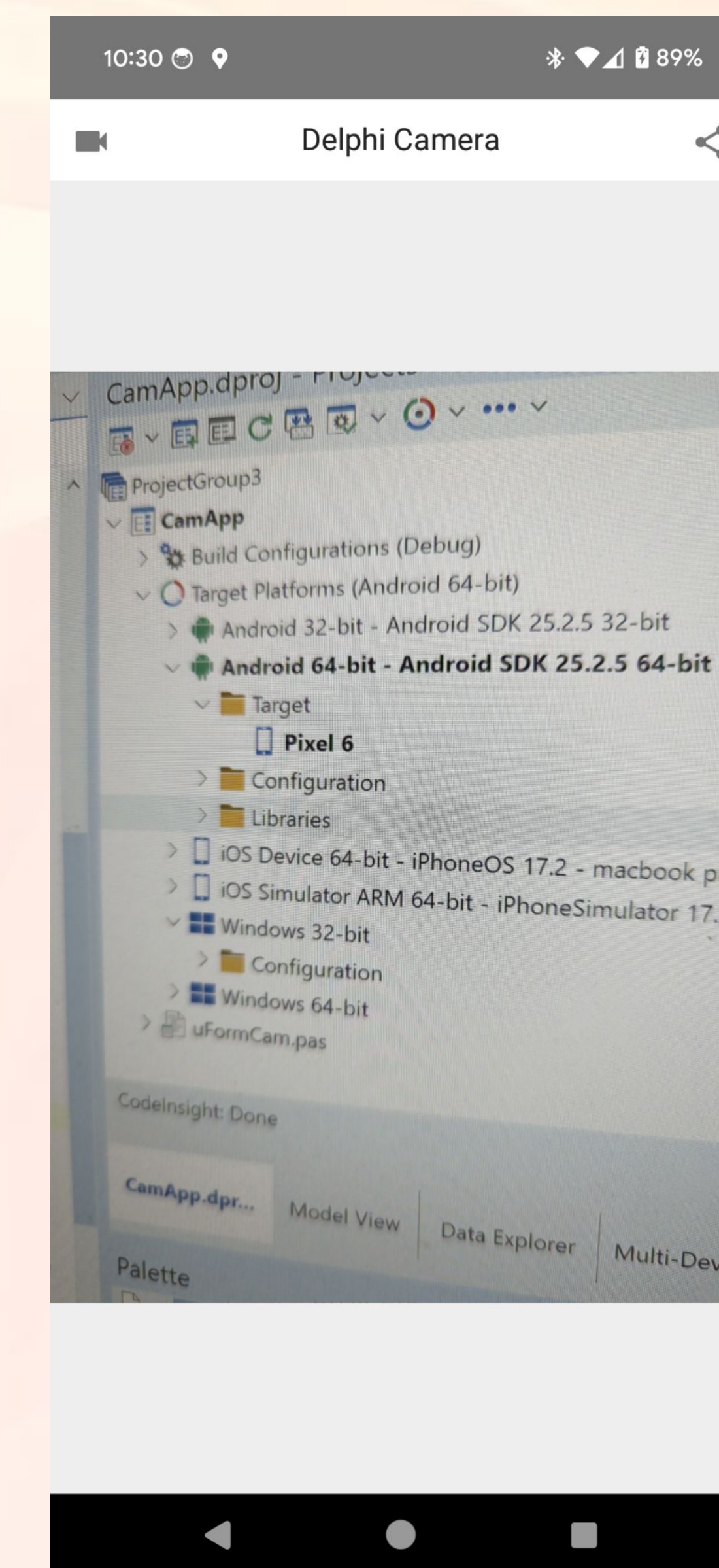
Additional FireMonkey Improvements

- New TWinFormPositionerService to manage the screen(s) layout and positions
- Extensive refactoring of the TCommonCustomForm.KeyDown code, which handles accelerator keys, dialog keys, tab keys
- Changes to double click management
- New IFMXPlatformPropertiesService
 - Replacing IFMXDefaultPropertyValueService and IFMXTextEditingService
- Introduction of a new universal TFontManager
- New Windows 11 style
- Split Views / Split Screen support for Mobile: see image here



Extensive FireMonkey quality work in 12.1

- On macOS, transparent bitmaps saved to non-transparent formats behave like on Windows (i.e., the transparent color will turn black).
- Further improvements for TMemo component (9 different extensions), TEdit (7 different extensions), FMX grids, FMX effects (with Skia), Keyboard shortcuts on desktop, the Flow Layout, Treeview, Listbox...
- Addressed issues in Android actions for library access and photo sharing, plus camera service



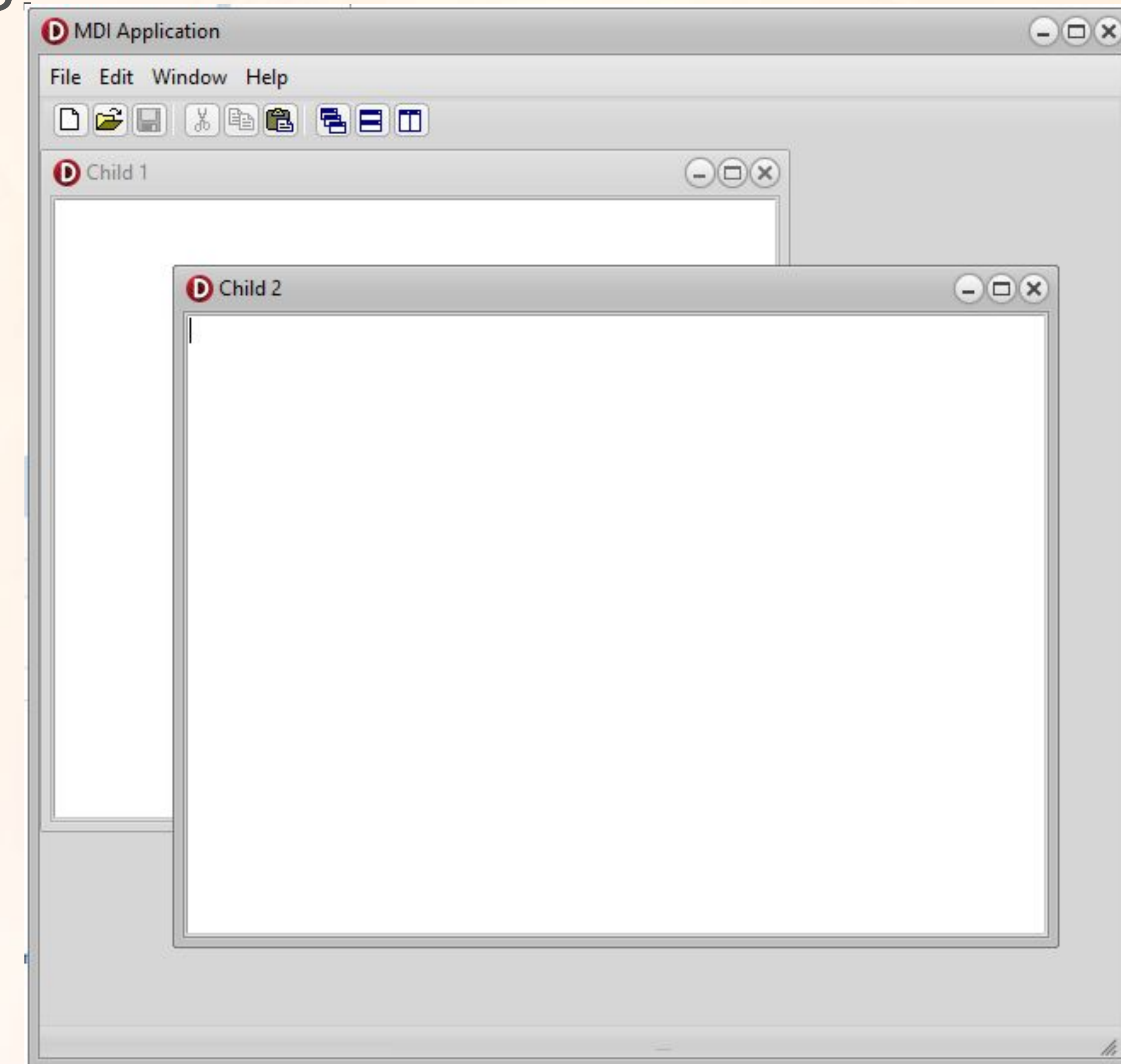
UI Libraries: VCL

Reworked VCL MDI architecture

Reworked the VCL MDI (Multi Document Interface) to address a number of platforms issues with HighDPI and VCL issues with styles:

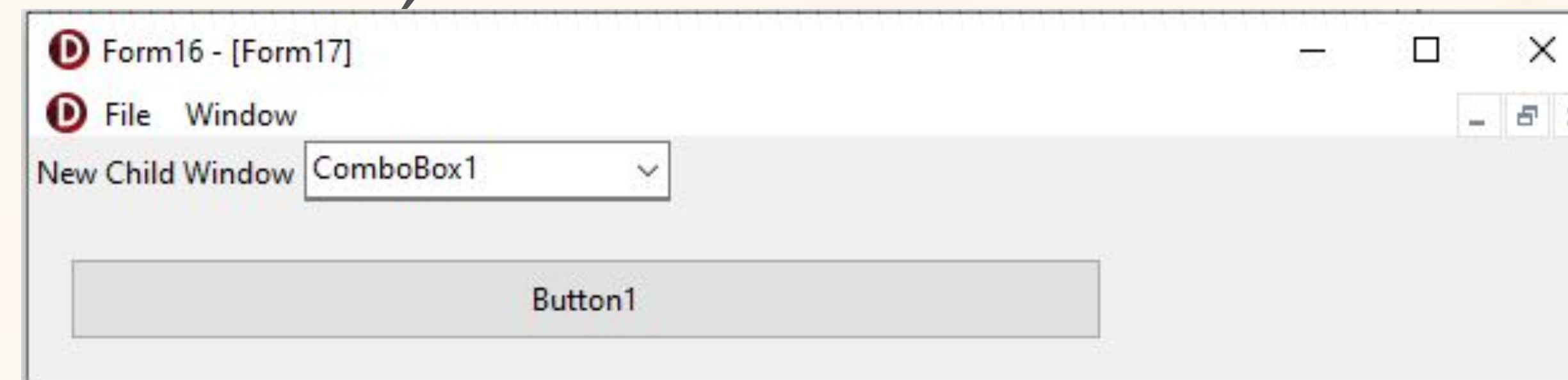
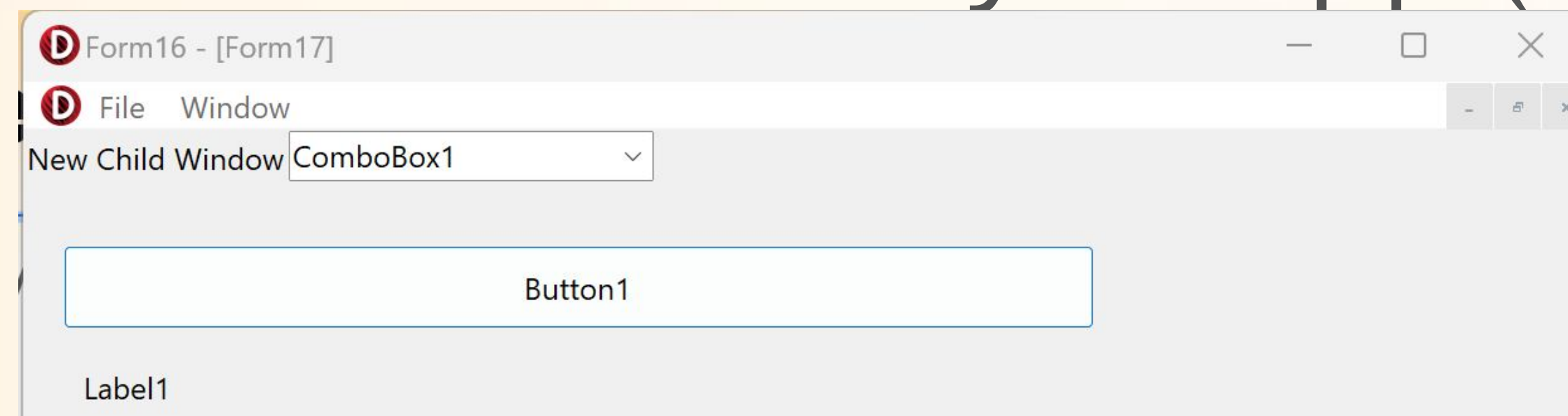
- Paint a custom border for MDI child window
- Support for a custom border for forms with Parent Assigned property (also outside of the MDI scenario)

Updated code generated by the MDI wizard (with support for tabbed UI)

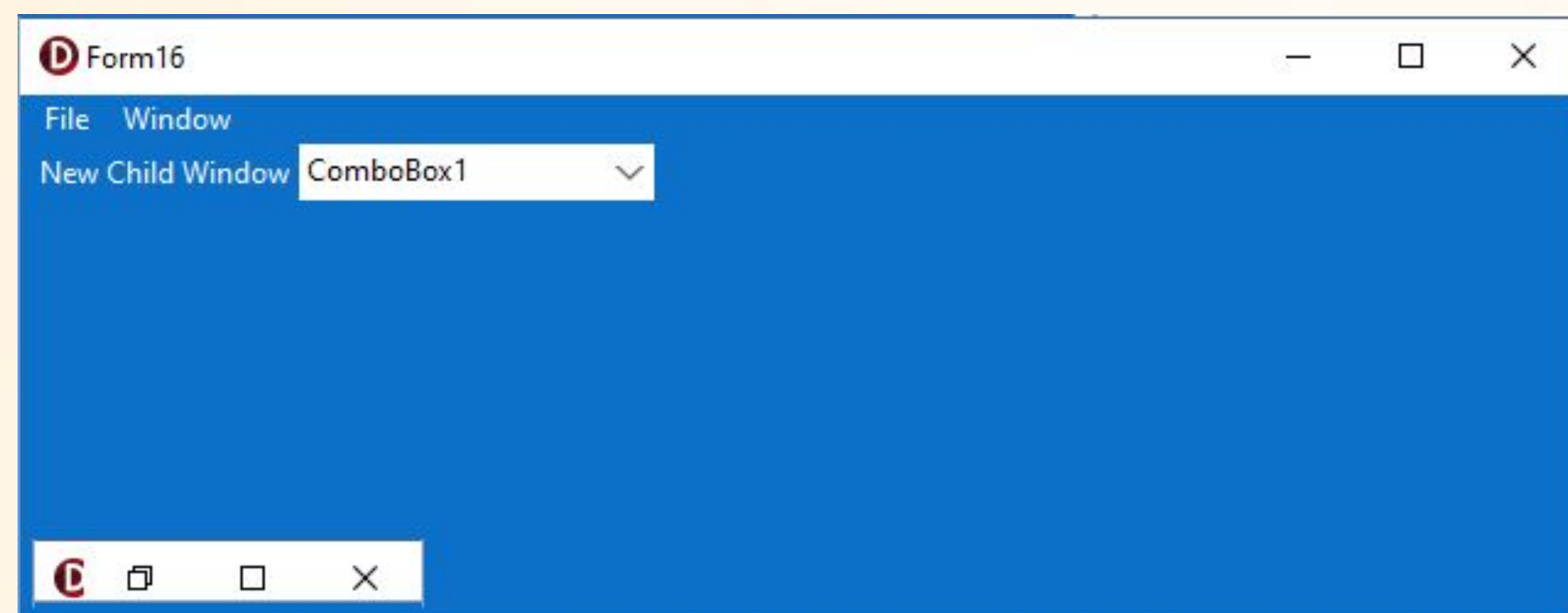


Reworked VCL MDI architecture (Examples)

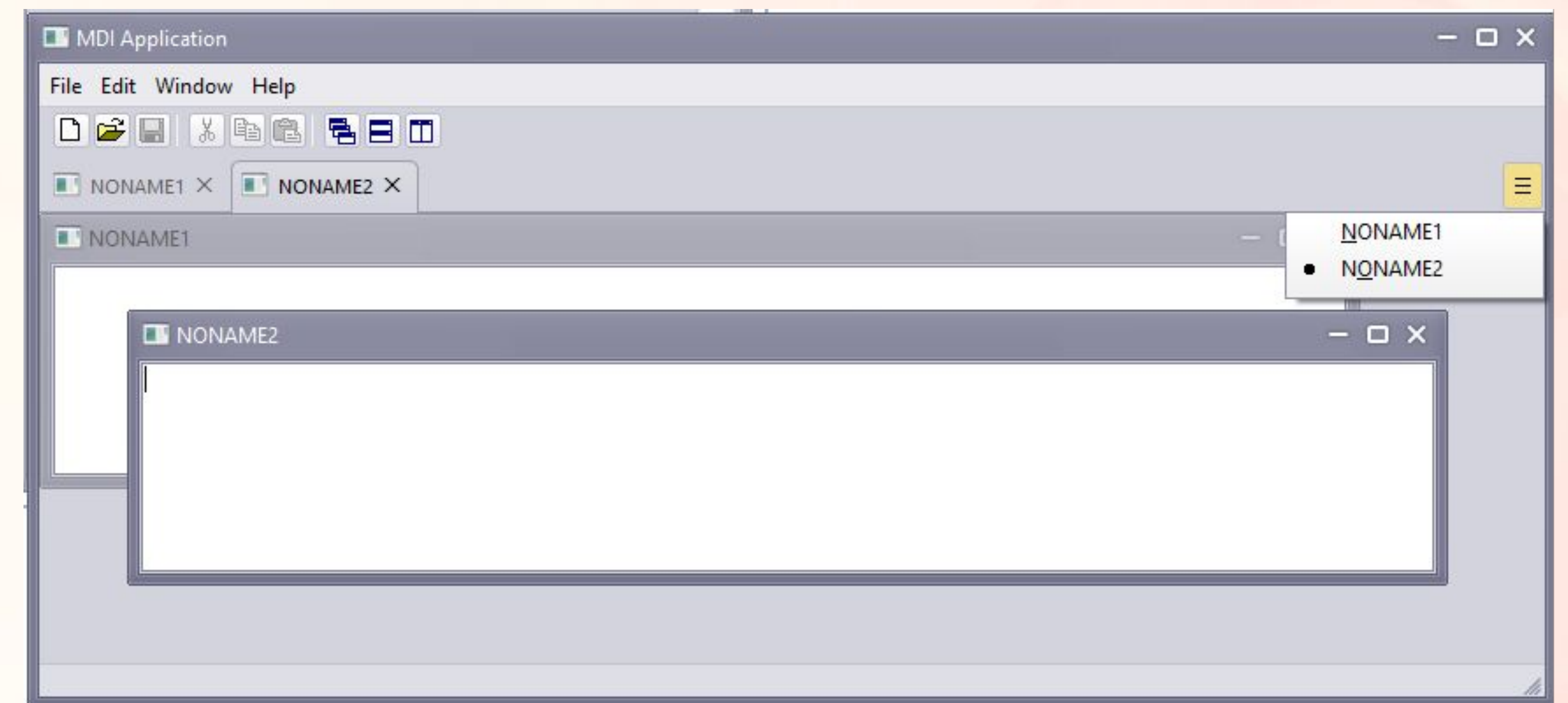
- Maximized child in unstyled app (11.3 -> 12)



- Minimized styles (12)

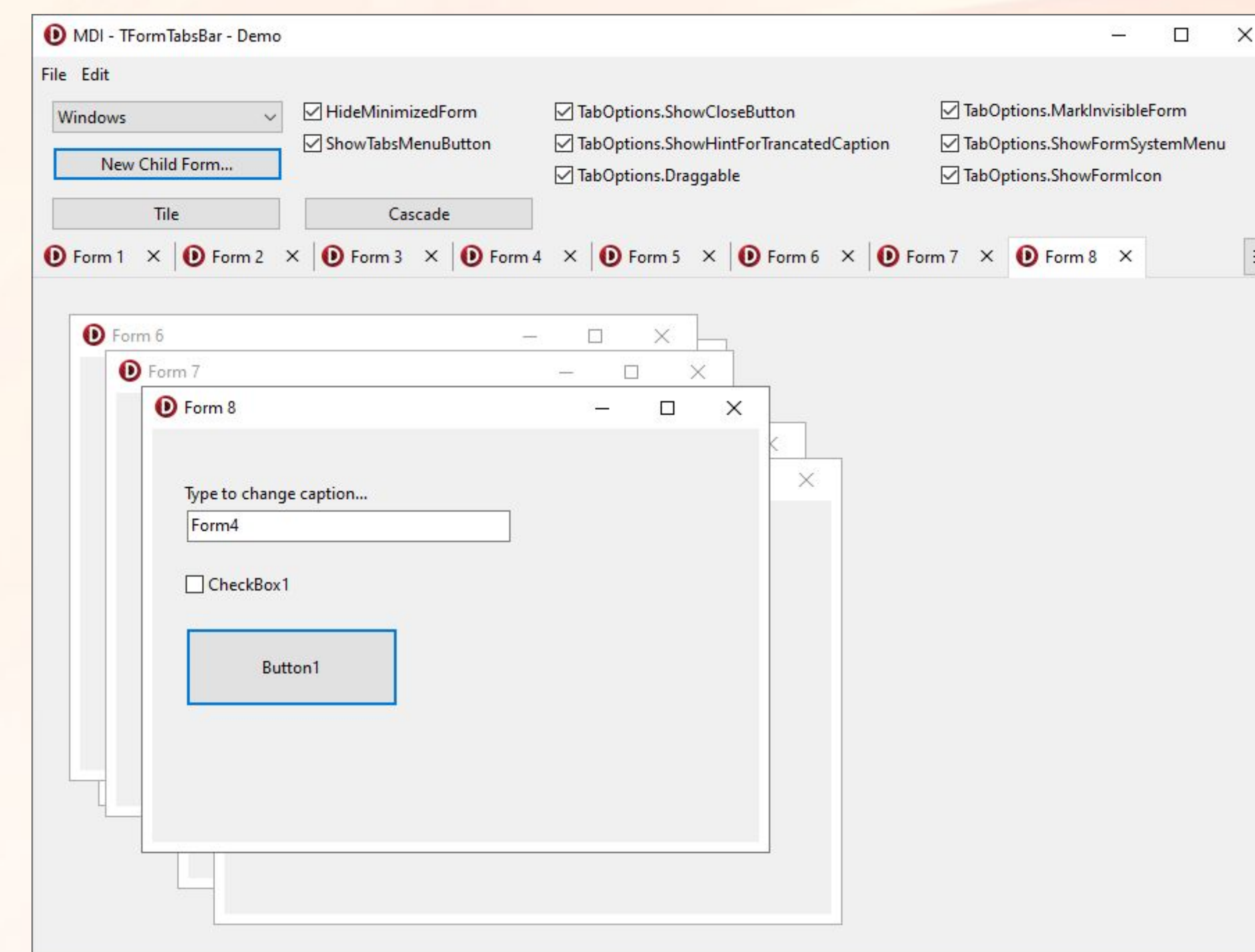


- App Generated by the MDI Wizard -> with styles applied



VCL: New Multi-tab architecture

- Goals
 - Migrate MDI apps to multi tab (full client area or partial, like in MDI)
 - Create new apps with a modern architecture out-of-the-box
- Introducing
 - IFormVisualManager Interface
 - VisualManager property of the TCustomForm class (to associate a form with its “*visual manager*” host)
 - TFormTabsBar Control
 - Ready-to-use solution for hosting multiple child forms in a modern tab-like user interface
 - Includes support for dragging a tab content out of the tab and in a separate window



VCL: Fonts and Screens

Reworked support for VCL TFont scaling independent from DPI scaling

- **TFont.Size** property now adapts to different DPIs
- **TFont** class new properties and methods:
 - property `IsDPIRelated`: Boolean // enables using `TFont.PixelsPerInch` property
 - property `IsScreenFont`: Boolean // for global fonts in `TScreen` class
 - procedure `ChangeScale` // called to initialize and scale any DPI-related font
 - procedure `ScaleForDPI` // used in code to adapt a font to any DPI

TControl Enumerator

Offers various filters to navigate child controls (which can be combined)

- ceftAll - all controls, including controls of the child controls (recursive)
- ceftEnabled - control should be enabled.
- ceftDisabled - control should be disabled.
- ceftVisible - control should be visible.
- ceftInvisible - control should be invisible.
- ceftCustom - Uses CanEnumerateControl method (OnEnumerateControl event)

```
for ACtrl in AControl.GetControls(AMyFilter) do  
begin  
    ...  
end;
```


Additional VCL Features (1/2)

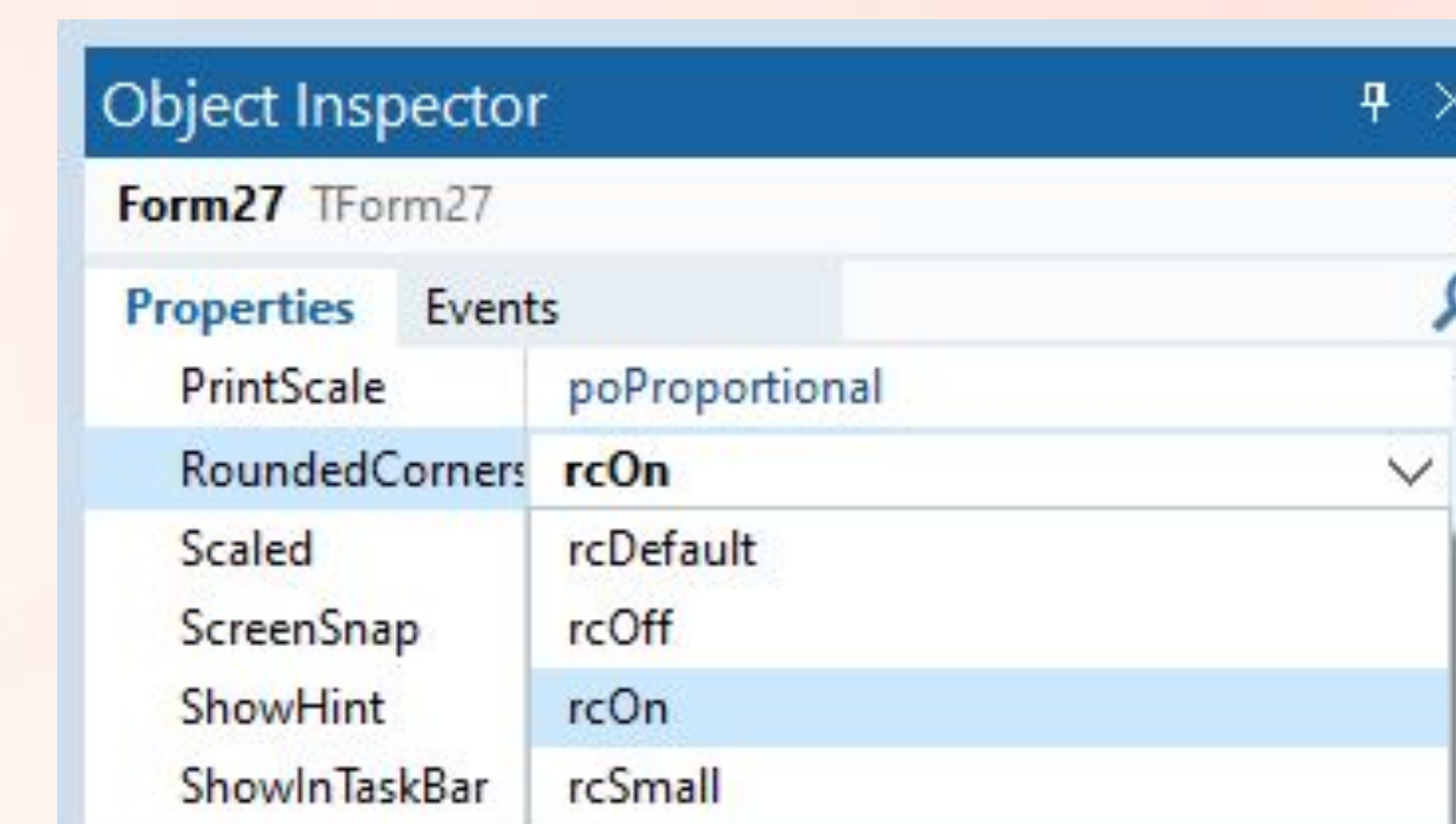
- **Tile View support to TListView**
 - New vsTile in TViewStyle
 - TileOptions property to adjust tile items
 - TileColumns property to adjust display of sub-item text from columns
- **TNumberBox control**
 - nbmInt64 mode accepting 64-bit numbers in input
 - Considers MinValue and MaxValue even if they are equal
- **TControlList Control**
 - Better support TControlList multiple selections
 - SmoothMouseWheelScrolling property
- **The TitleBar control adds support for Windows 11 snap layouts**

Additional VCL Features (2/2)

- TWICImage and TImageCollection now have a Dormant() method that removes GDI handles from system memory
- Significant rework of TActivityIndicator: new types (RotatingLines and Refresh) bitmap sizes, and custom colors

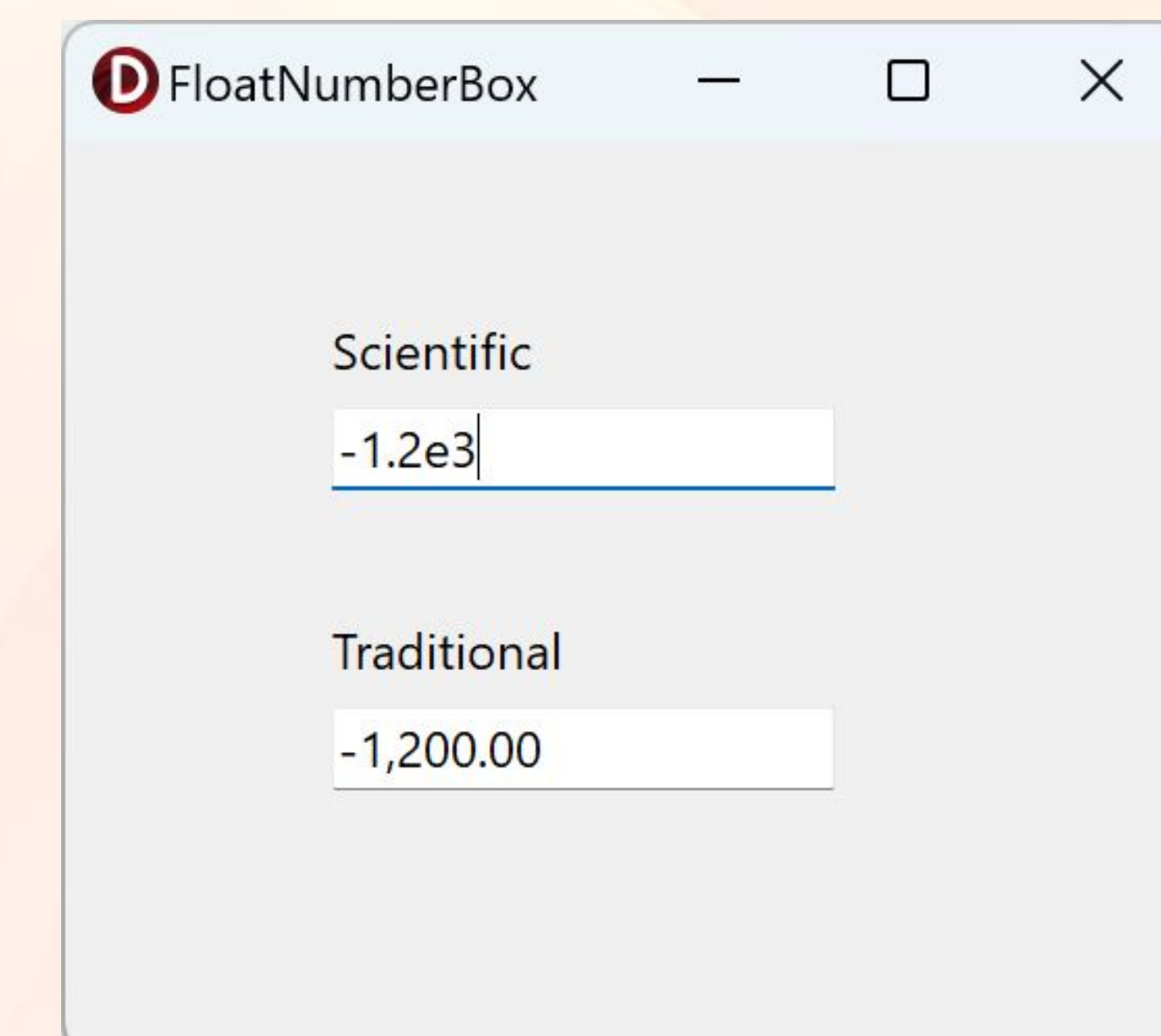



- ShowInTaskbar property for TForm (offering more flexibility in taskbar management)
- Desktop Windows Manager Enhancements
 - Win11 RoundedCorners and EnableImmersiveDarkMode
- New Double Buffering Mode



VCL quality in 12.1

- Improvements for recent MDI redesign
 - Including MDI menu merge and styles, tiling, and more
- Enabled VCL TEdit to work with Windows *Ease of Access Text Cursor*
 - Re-enabled accessibility to the IDE
 - Using JAWS with the IDE now works by default
- NumberBox improvements
 - Better support for negative values
 - In nbmFloat mode, support for exponent, as in '-1.2e3'
- ProgressBar State is now supported under VCL Styles
- Addressed a few issues in Konopka Signature VCL Controls (KSVC)
- Additional work on styles, High DPI, TWICImage and much. much more



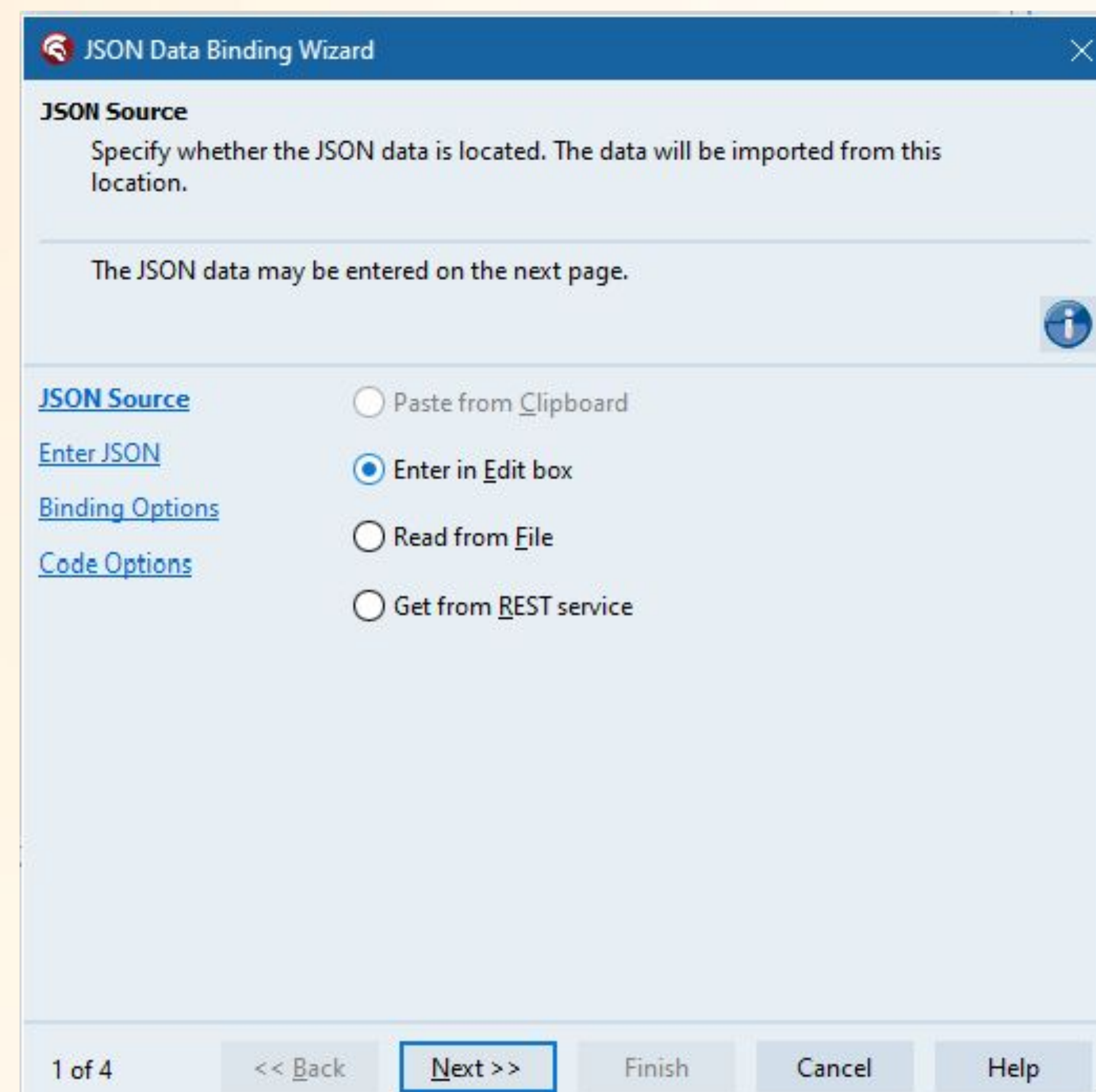


All About Data FireDAC & Web

JSON Data Binding Wizard

New Wizard to create Delphi classes mapped to JSON data structures

- Starts from JSON text
- Supported libraries are REST.Json and System.JSON.Serializers (more can be added)
- Multiple “Binding Options” specify how to map JSON elements to Delphi elements
- “Code Options” specify how to name Delphi entities, how to use properties



JSON Data Binding Wizard

JSON Source
Specify whether the JSON data is located. The data will be imported from this location.

The JSON data may be entered on the next page.

JSON Source

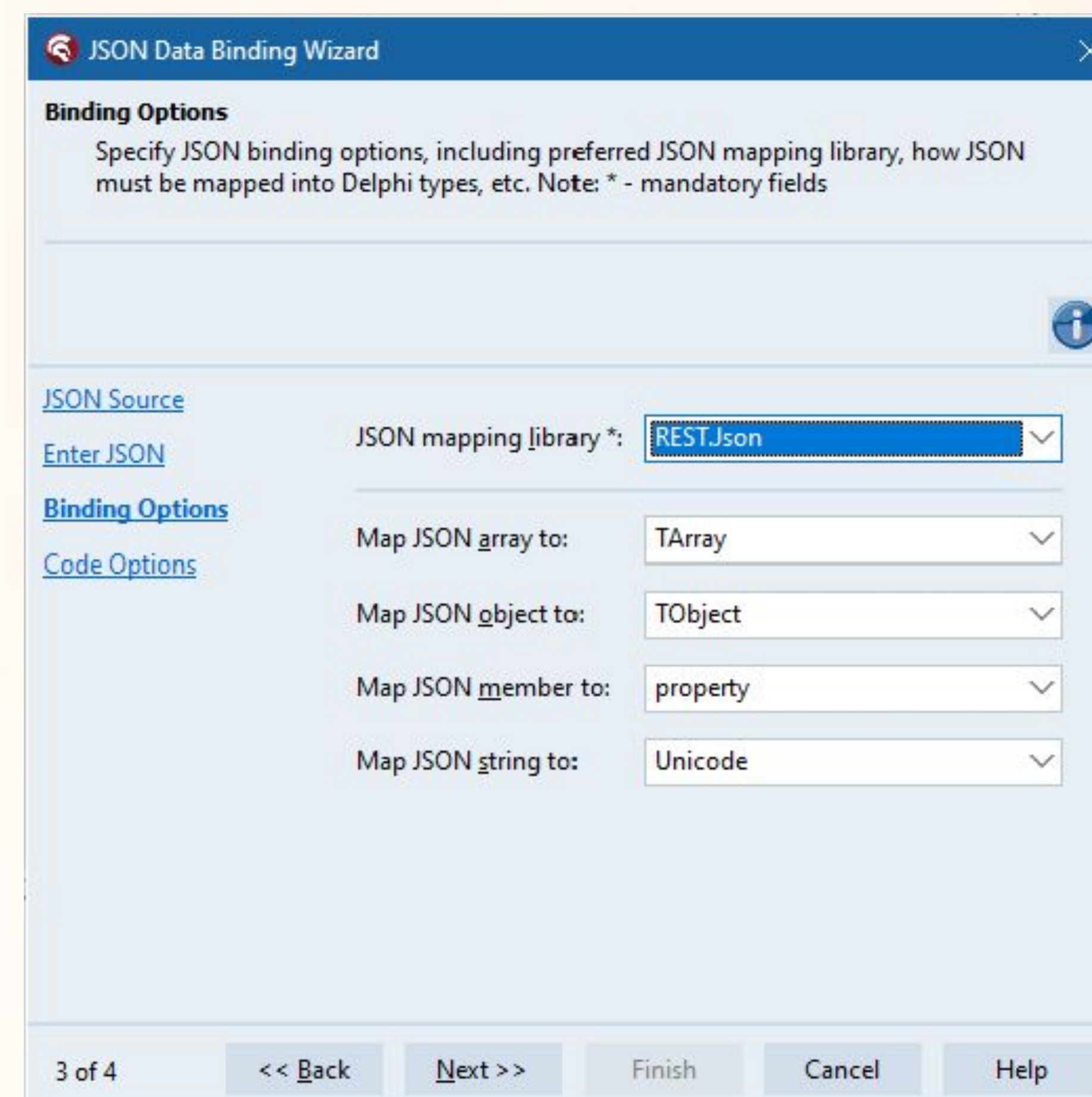
Paste from Clipboard

Enter in Edit box

Read from File

Get from REST service

1 of 4 << Back Next >> Finish Cancel Help



JSON Data Binding Wizard

Binding Options
Specify JSON binding options, including preferred JSON mapping library, how JSON must be mapped into Delphi types, etc. Note: * - mandatory fields

JSON Source

JSON mapping library *: REST.Json

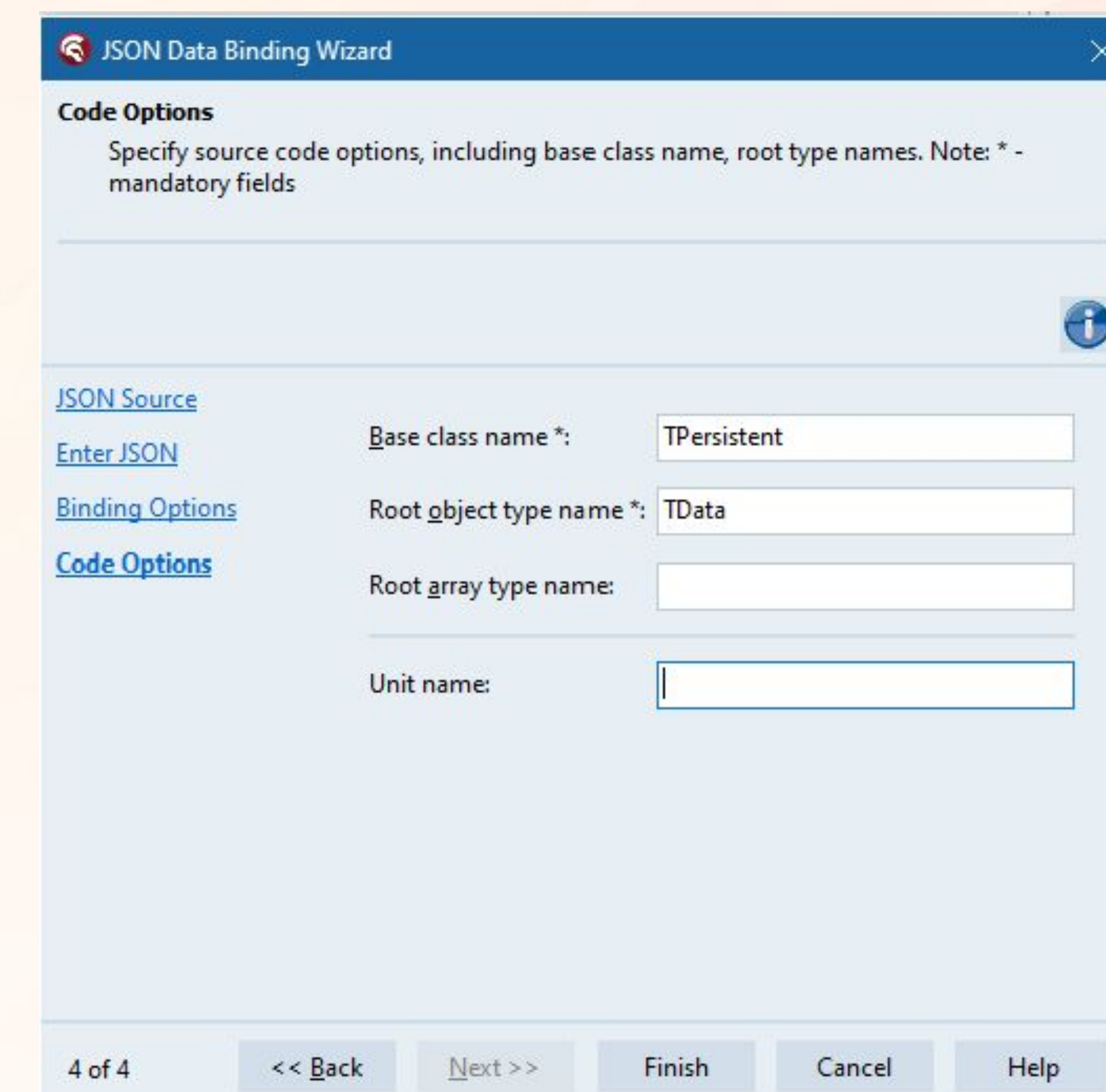
Map JSON array to: TArray

Map JSON object to: TObject

Map JSON member to: property

Map JSON string to: Unicode

3 of 4 << Back Next >> Finish Cancel Help



JSON Data Binding Wizard

Code Options
Specify source code options, including base class name, root type names. Note: * - mandatory fields

JSON Source

Base class name *: TPersistent

Root object type name *: TData

Root array type name:

Unit name:

4 of 4 << Back Next >> Finish Cancel Help

JSON Data Binding Wizard

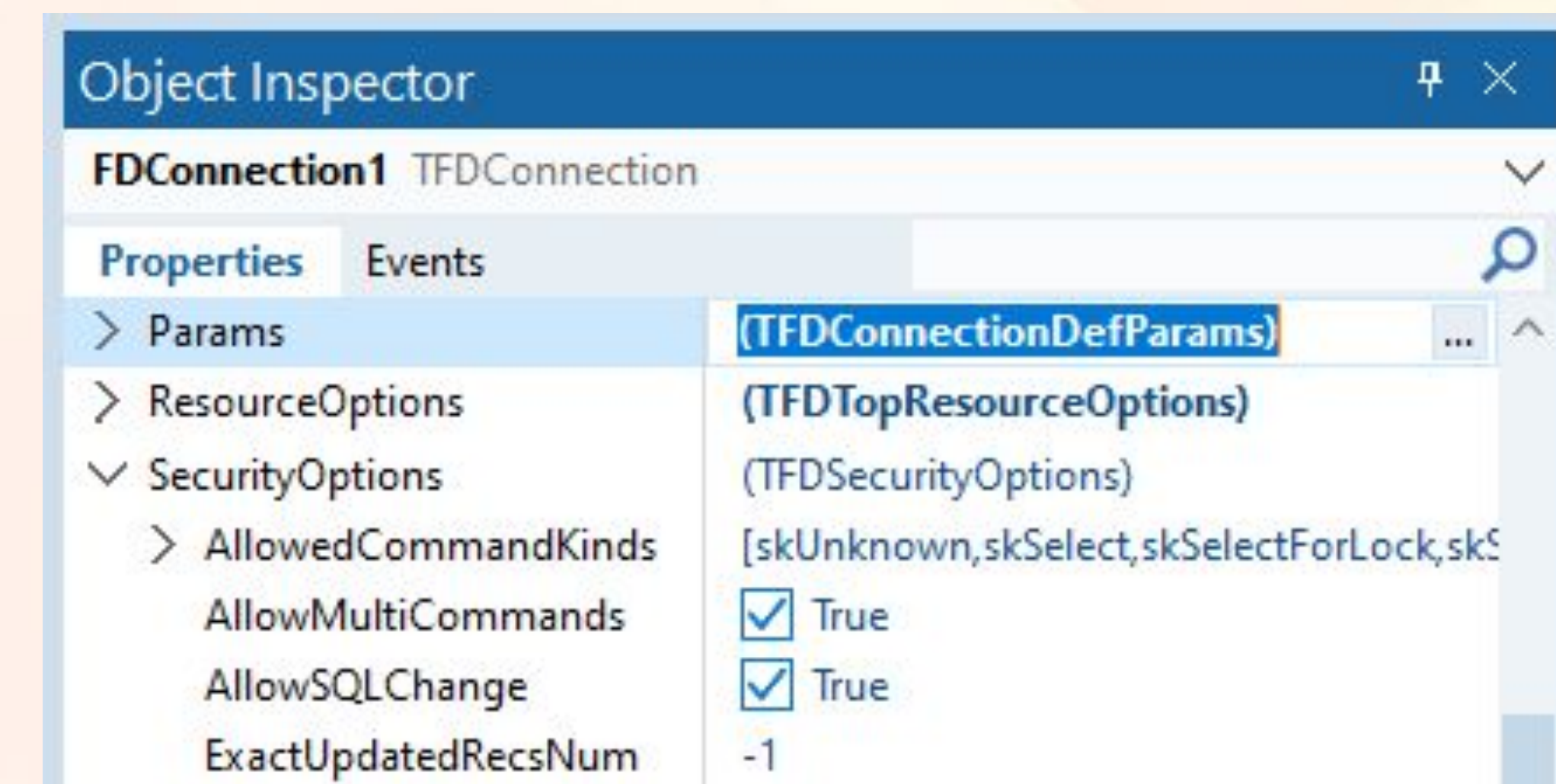
```
{
  "id": 1,
  "name": "Leanne Graham",
  "username": "Bret",
  "email": "Sincere@april.biz",
  "address": {
    "street": "Kulas Light",
    "suite": "Apt. 556",
    "city": "Gwenborough",
    "zipcode": "92998-3874",
    "geo": {
      "lat": "-37.3159",
      "lng": "81.1496"
    }
  },
  "phone": "1-770-736-8031 x56442",
  "website": "hildegard.org",
  "company": {
    "name": "Romaguera-Crona",
    "catchPhrase": "Multi-layered client-server neural-net",
    "bs": "harness real-time e-markets"
  }
}
```

```
uses uJsonMapping, System.JSON;
.
.
- procedure TForm1.Button1Click(Sender: TObject);
. begin
.   RESTRequest1.Execute;
.
.
39   TJSONMapper<Users>.SetDefaultLibrary('System.JSON.Serializers');
40   var LUsers := Users.FromJSON(RESTResponse1.JSONText);
.
.
.   Memo1.Clear;
.   for var i: integer := Low(LUsers.Dataset) to High(LUsers.Dataset) do
.     begin
-     Memo1.Lines.Add(LUsers.Dataset[i].name);
.     Memo1.Lines.Add(LUsers.Dataset[i].username);
.     Memo1.Lines.Add(LUsers.Dataset[i].email);
.     Memo1.Lines.Add(LUsers.Dataset[i].address.street);
.     Memo1.Lines.Add(LUsers.Dataset[i].address.suite);
50    Memo1.Lines.Add(LUsers.Dataset[i].address.city);
.     Memo1.Lines.Add(LUsers.Dataset[i].address.zipcode);
.     Memo1.Lines.Add(LUsers.Dataset[i].address.geo.lat.ToString);
.     Memo1.Lines.Add(LUsers.Dataset[i].address.geo.lat.ToString);
.     Memo1.Lines.Add(LUsers.Dataset[i].phone);
-     Memo1.Lines.Add(LUsers.Dataset[i].website);
.     Memo1.Lines.Add(LUsers.Dataset[i].company.name);
.     Memo1.Lines.Add('-----');
.
.   end;
. end;
```


FireDAC More Secure Coding

FireDAC SecurityOptions property: new features to help write more secure code

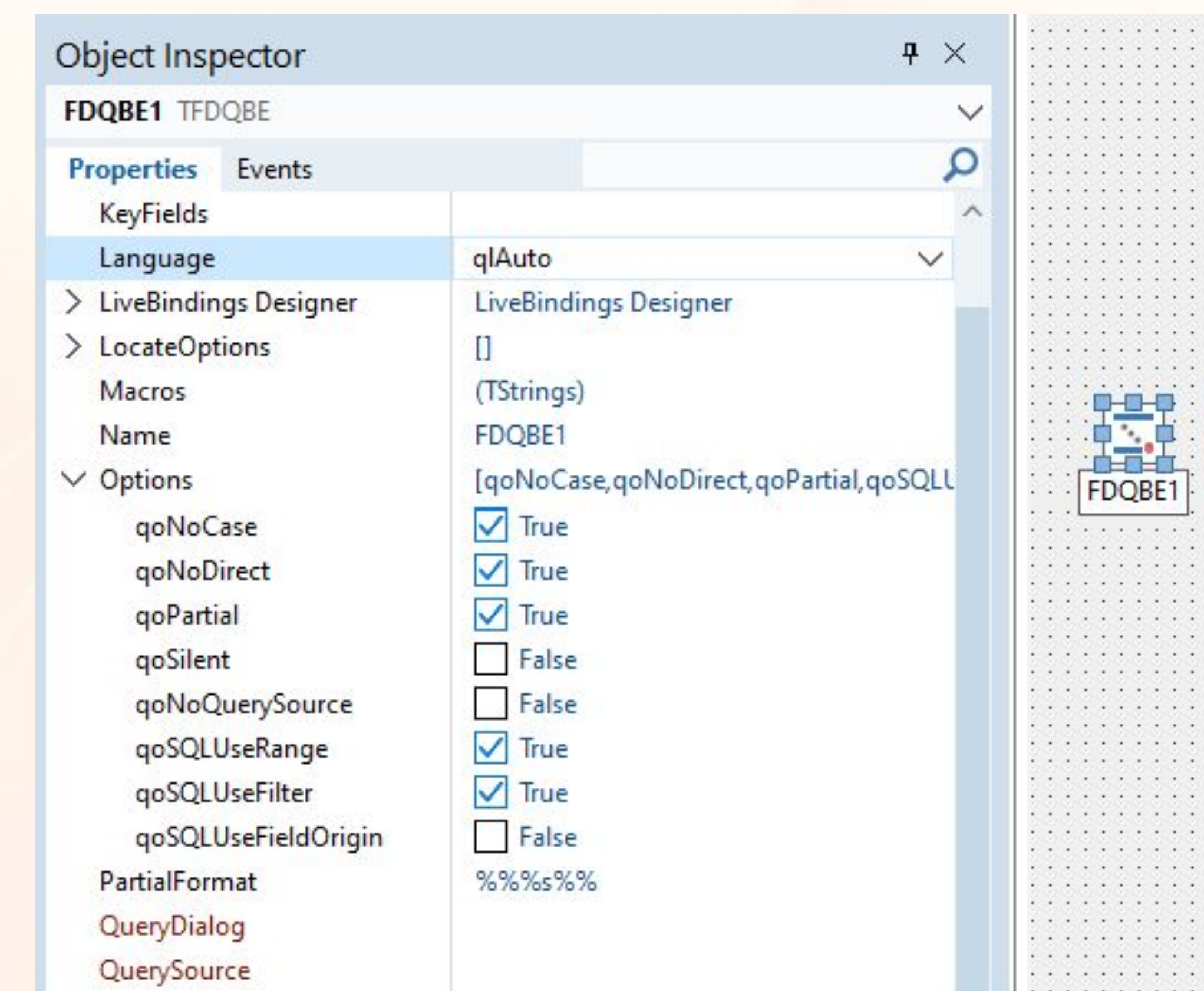
- AllowedCommandKinds: restricts the categories of SQL operations a query can perform
- AllowMultiCommands: disables the execution of multiple SQL commands in a single query (if the underlying DB supports it)
- AllowSQLChange: can prevent the code to change the text of a FireDAC SQL query at runtime
- ExactUpdatedRecsNum: if > 0 , enables extra checked on the number of records updated by a query execution



These features can help, but writing secure database applications still requires developers to follow a number of best practices

Data and FireDAC

- Added also several JSON streaming improvements
- FireDAC support for QBE (query by example)
 - Use data aware UI controls so that an end user can define data filters
- IBLite/IBToGo for iOS Simulator
- DB RTL: improved TBlobField display logic
- SpellChecking in TDBRichEdit
- FireDAC SQLite Version Update (SQLite 3.42)
 - No encryption support outside of the paid SQLite EE
 - In alternative, use version 3.31.1 with FireDAC encryption (FDE)



DataBase & FireDAC in 12.1

- **FireDAC adds official support for**
 - Firebird version 5
 - Includes support for parallel operation in TFDFBBackup for Firebird 5
 - PostgreSQL version 16
 - InterBase 2020 Update 5 versions of IBToGo and IBLite for Android
 - SavePoint / RestorePoint support for a cloned TFDMemTable
- **Database RTL**
 - Cleanup in DataSet's OnValidate event
 - Some TDBNavigator fixes

Two words on InterBase

- InterBase developer edition included with RAD Studio
 - 2020 updates continuously released
 - IBConsole UI revamp (it's a VCL application!)
 - VAR contracts available
- Embedded versions:
 - IBLite free to distribute
 - IBToGo free for mobile (in Enterprise)
 - Unlimited deployment all platforms with yearly subscription

RAD Server and Sqids

- Sqids are an improved version of Hashids, see <https://sqids.org>
- Replace physical IDs in a URL with a series of letters
 - Similar to YouTube video IDs
 - Uses a standard algorithm available in many programming languages
 - Can be customized via properties
 - Includes a list of illegal words (multi language, but you can provide yours)
- Display record with ID 12345:
 - From /customer/12345 to /customer/tyyksole
- Fully integrated and ready-to-use in RAD Server
 - When a TRESTRequest.Resource property parameter name starts with "#" the value will be Sqids encoded
- Available as a standalone feature (class TSqidsEncoding)

HTTP, REST, and More on RAD Server

HTTP and REST Client Library

- Added support for HTTP DELETE method with content
- Improved redirect and cookie management
- Support for large files in TMultipartFormData
- RESTRequest accepts multi-part ContentType
- Curl support on Windows and macOS (via libcurl)

RAD Server

- Performance improvements by embedding FastMM5
- Paging allows page size override in client request
- Session authentication improvements

REST and HTTP in 12.1

- TRestClient supports using certificates from the local machine list
- Certificates support in TCurlHttpRequest
 - We recently introduced support for Curl also on Windows (beside Linux)
- Support for web hex color syntax in StringToColor function
- The ability for the REST Debugger to copy values from table cells
 - The REST debugger is a very nice, standalone, free tool by Embarcadero
- Ability to change the username of RAD Server users

- Improvements in JSON objects serialization and TJSONNumber decimal value handling with scientific notation

Infrastructure

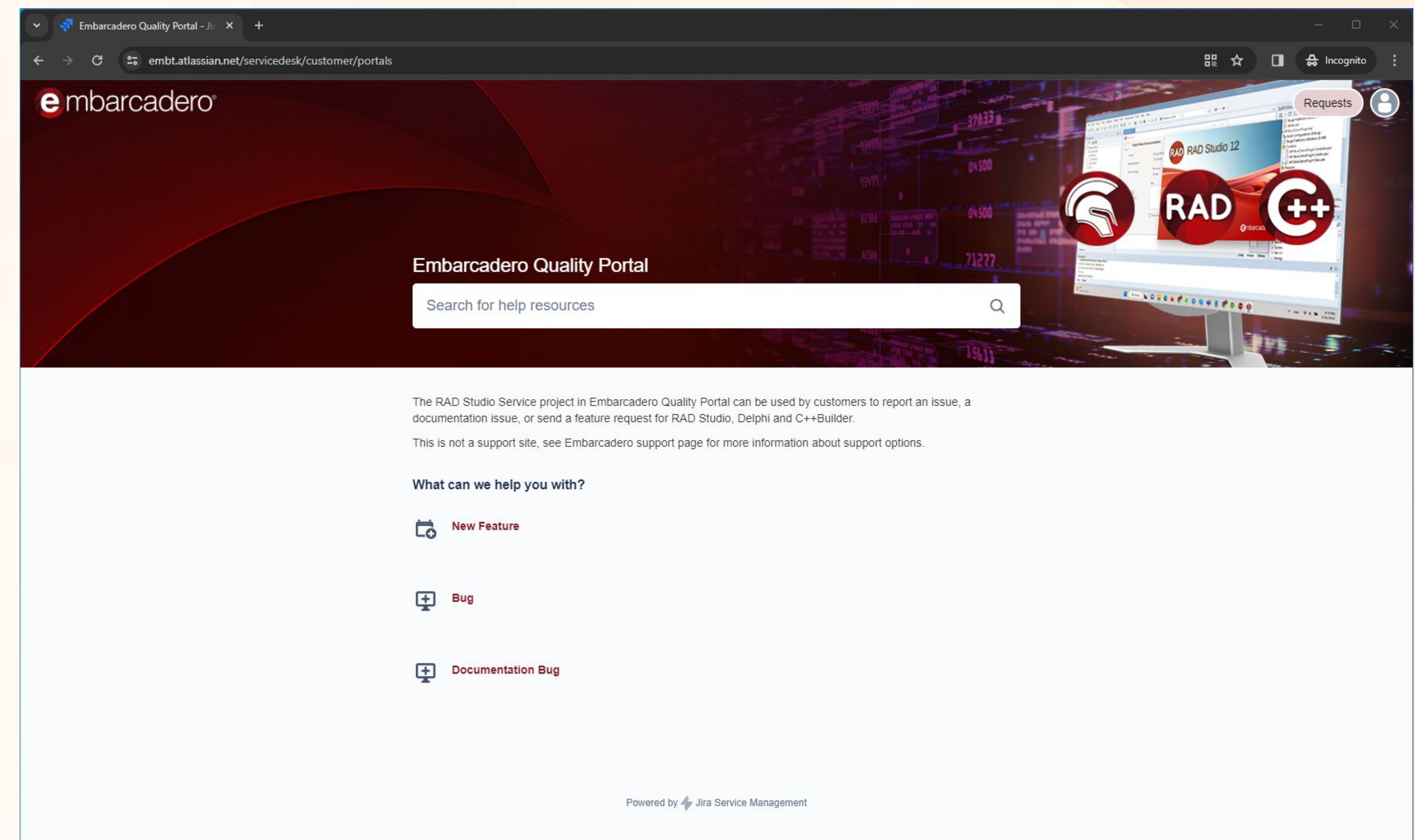
The background features a complex, layered composition of abstract shapes and wavy lines. The color palette is warm, ranging from pale yellow and light pink to vibrant orange and deep magenta. The shapes are semi-transparent and overlap, creating a sense of depth and movement. The overall aesthetic is modern and dynamic, typical of a corporate or technical presentation.

New GetIt Servers

- Now hosted on AWS
 - High availability
 - Faster, much faster
- For 12.x, 11.x and now also 10.4
 - The older ones still internal

New Embarcadero Quality Portal

- <https://qp.embarcadero.com> (takes you to <https://embt.atlassian.net/servicedesk/customer/portals>)
- Powered by Jira Service Manager
- Integrated with the Embarcadero R&D Jira instance
- Hosted by Atlassian Cloud
- Self service accounts
- New Feature requests, Bug reports and Documentation Bug reports



Summary

The background features a complex, layered composition of abstract shapes and wavy lines. The color palette is warm, ranging from pale yellow and white to vibrant orange and pink. The shapes are semi-transparent and overlap, creating a sense of depth and movement. The overall aesthetic is modern and artistic.

With all the features, are we neglecting quality?

- **Short answer: No!**
- RAD Studio 12.0 is providing a fix for:
 - 1,027 issues reported by customers on Quality Portal
 - 877 bug reports and 150 feature requests
- RAD Studio 12.1 adds fixes for:
 - Over 300 customer reported issues in Quality Portal



RAD Studio 12 Athens was a *Mega Release*

1. C++

- Visual Assist
- New Clang Tech Preview

2. Installer & IDE

- Multidevice Icons
- DelphiLSP
- VCL Designers, IDE Misc, ToolsAPI

3. FireMonkey & Skia

- Skia UI Controls and rendering and graphic formats and way more (and VCL as well)
- TMemo and TEdit

4. VCL

- MDI and Tab based UIs
- Fonts and Screen, Components improvements

4. Data

- JSON Data Binding
- FireDAC Secure Coding
- Sqids, RAD Server, HTTP, REST

5. Delphi

- String literals improvements
- Platforms (Windows, Android)
- Circular Uses Statements, Floating Point

6. Quality

- *How many bug fixes this release?!*

The focus of RAD Studio 12.1 was on improving the quality of the many new features added in RAD Studio 12 Athens and on completing the Clang upgrade (moving it from preview to full feature)

RAD 12.1 Summary: Focus on Quality

- **Some substantial features**
 - Editor split views and other IDE enhancements
 - The delivery of the new C++Builder Win64 Clang compiler
 - Android API level 34 support
- **Quality highlights include**
 - C++ Builder VA integration
 - VCL and extensive FireMonkey quality
 - RTL, database, etc
 - Includes the 12.0 Patch 1 Fixes (released in January)





Q & A