



DELPHI 12 – LA VERSIONE DEFINITIVA?

Tutte le features di Delphi 12 con gli update 12.1 e 12.2



PAOLO ROSSI

WINTech ITALIA - CTO

SENCHA & EMB. MVP



blog.paolorossi.net



paolo@paolorossi.net



twitter.com/awebguy



github.com/paolo-rossi



linkedin.com/in/paolo-rossi-pc



DelphiDay
italian conference

20 Novembre 2024
Padova



wintech
italia



GITHUB PROJECTS



github.com/paolo-rossi



Delphi JWT

JSON Web Token Library



WiRL

REST Library for Delphi



Linux Daemon

Real Linux daemons



Delphi Neon

JSON Serialization Library



OpenAPI-Delphi

OpenAPI 3.0 Library



NATS Delphi

NATS Client Library for Delphi

Introducing **RAD Studio 12**



e mbarcadero®



RAD Studio 12 Athens

Embarcadero and Idera

Embarcadero is owned by Idera, www.ideracorp.com

Part of the DevOps division along with other tools for developers



Agenda: A Mega Release For You Today!

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?!*

Live Q&A! That was a mega webinar...

- ***Now let's answer all your questions!***

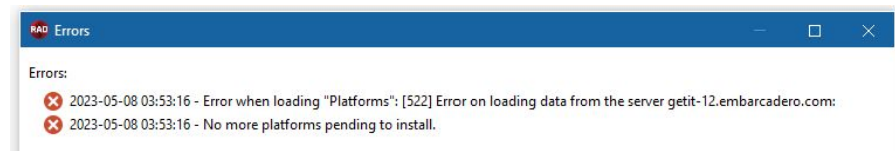
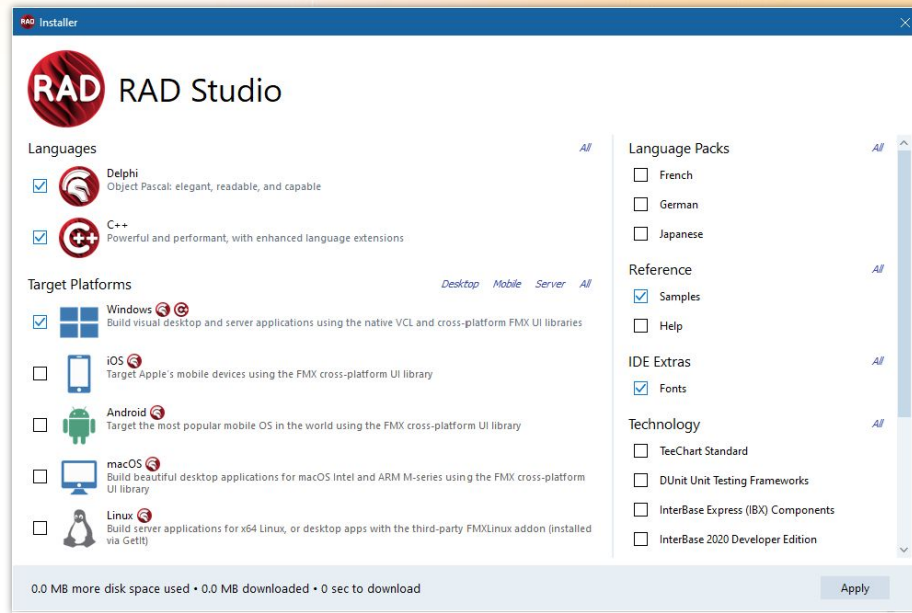


Installation and IDE

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 – done.
- 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

Structure

GetImageIndex: Integer

GetTabHintText: string

Close(var Allowed: Boolean)

GetTabColor: TColor

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

AllocCoTaskMemStr(const S: string): LPCWSTR

ApplicationMainHandle: HWND

CenterWindow(Wnd: HWND)

CopyData(Handle: THandle): THandle

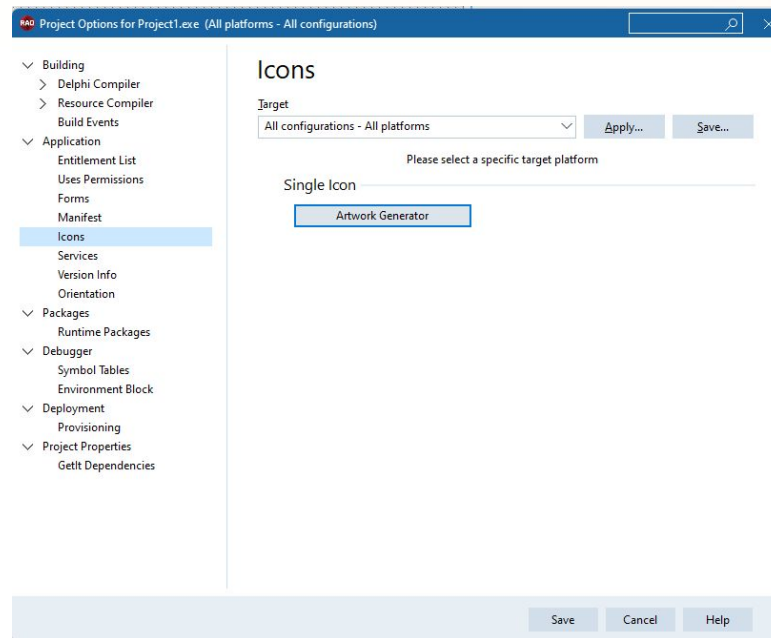
CreateMessageDialog(const Msg: string; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons): TForm

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

337 result(s) found.

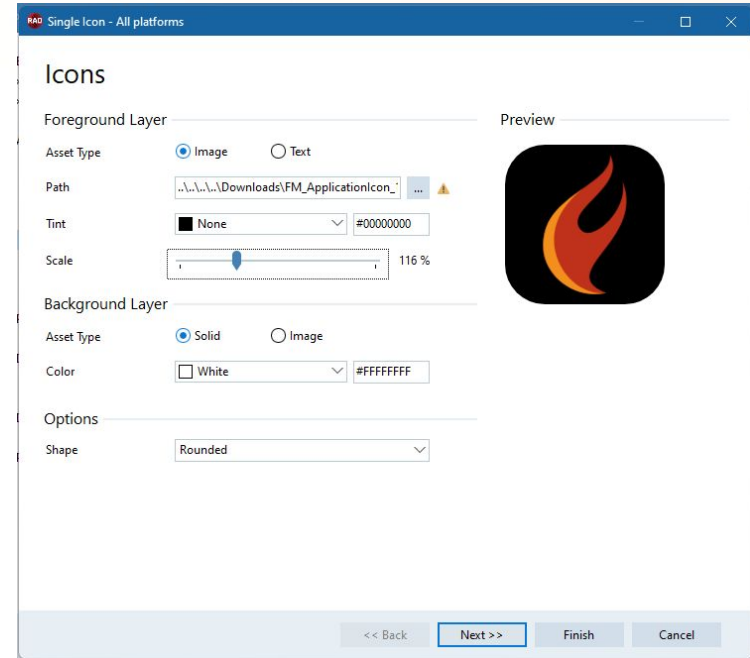
Multi Device Icon Generator

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



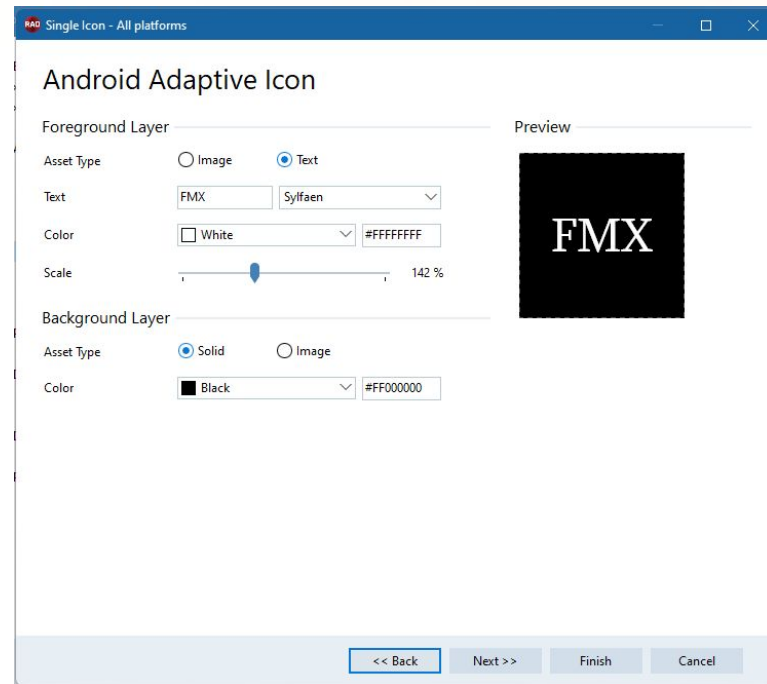
Icon Configuration

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



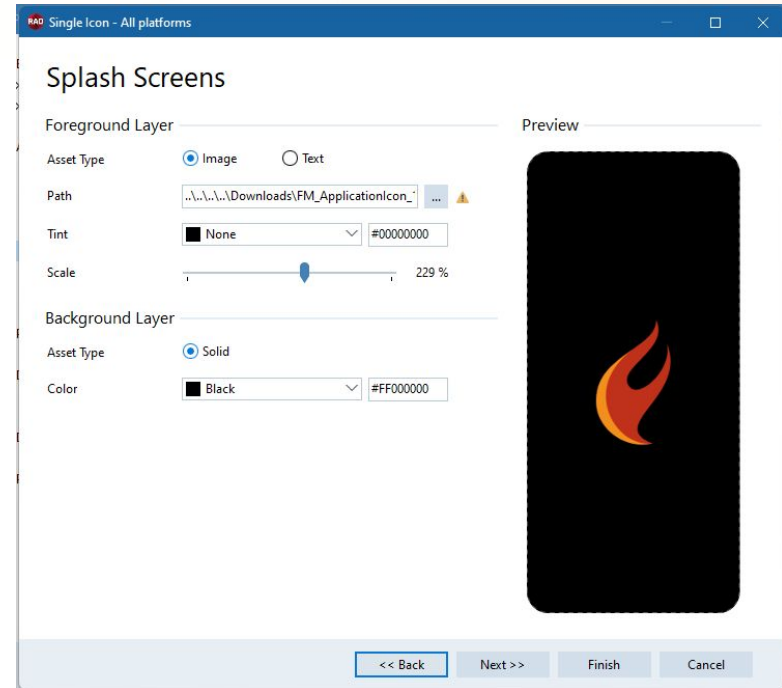
Android Adaptive Icon

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



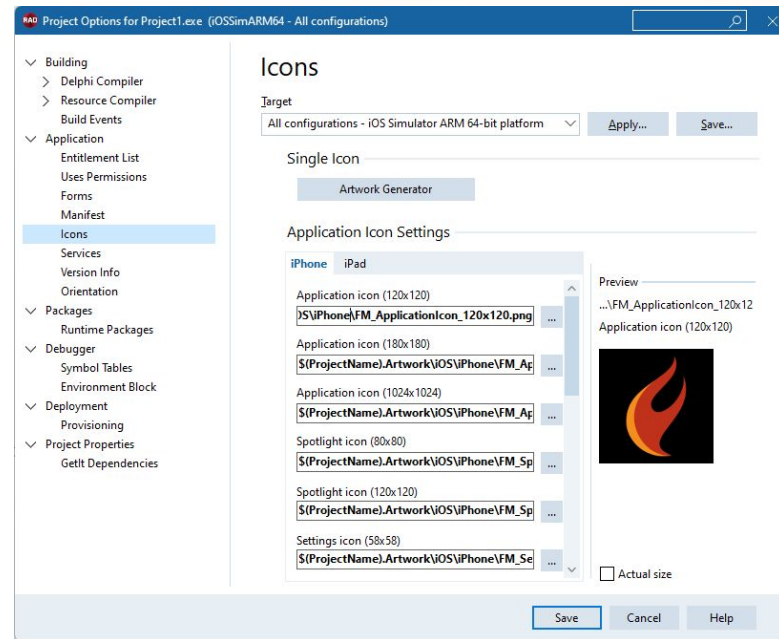
Splash Screens

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



Multi Device Icon Generator

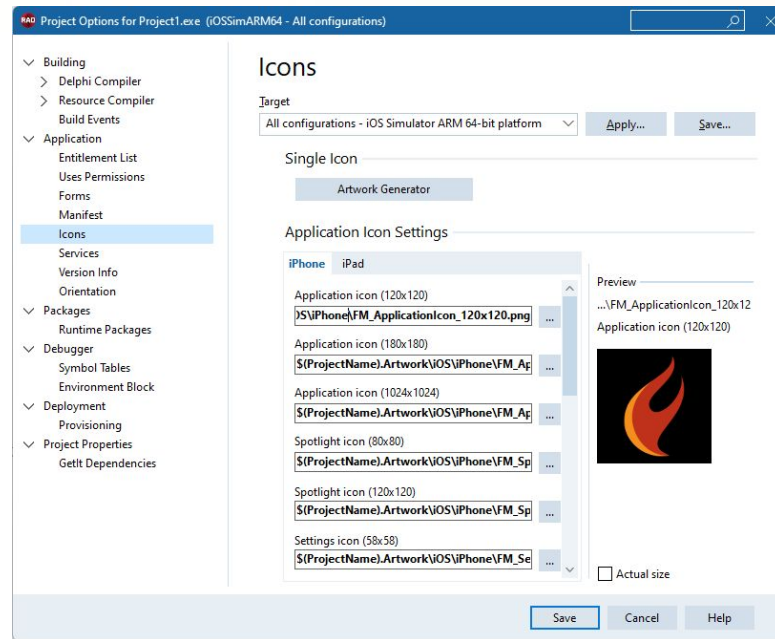
- All icons can be generated at once or individually
- Changes to single icons for specific platforms can be made easily



Multi Device Icon Generator

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

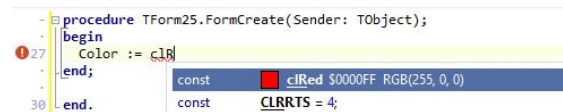
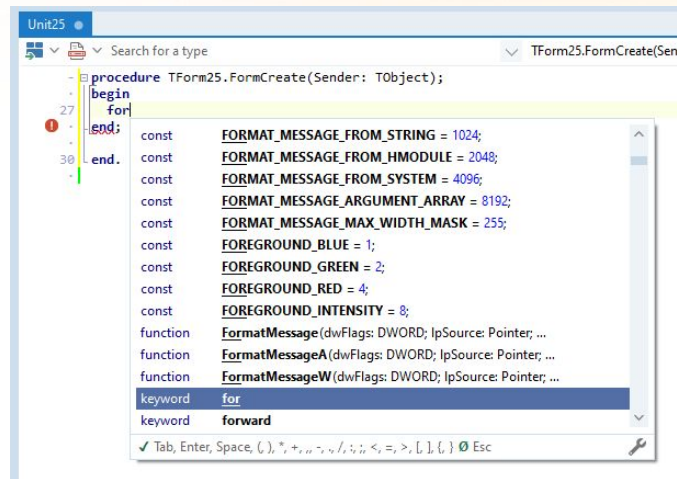
Keep in mind that the generated images are saved in a sub-folder of the project folder with the project name. Therefore, if you rename the project, you must manually rename the subfolder accordingly.



Delphi LSP

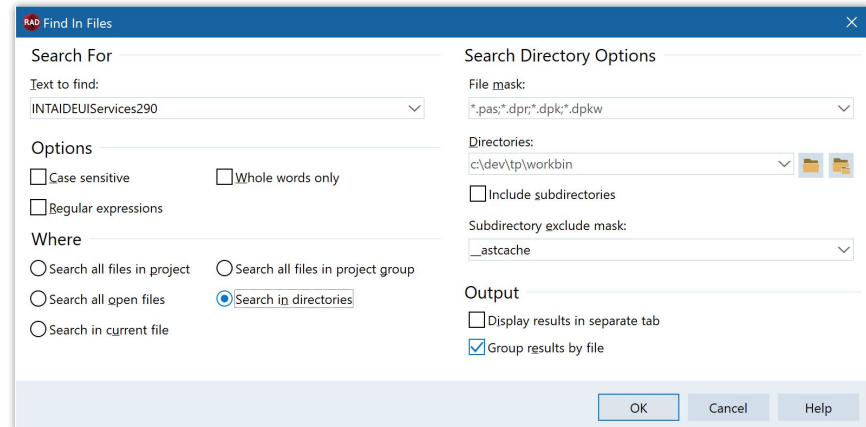
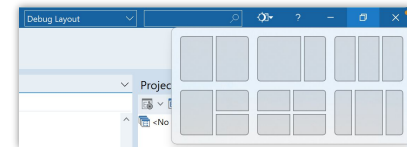
Improvements focused on stability, especially closing or switching projects. Plus:

- 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



IDE: Smaller Features

- New Win32/WinAPI App Wizard for C++
- You can reopen a tab with the state (scroll pos, cursor pos, region collapsing, bookmarks, etc) exactly as it was (via a setting on the Options > IDE > Saving and Recovering page)
- Editor tabs now have a Copy File Path menu item, for full / filename only / path only
- The VCL designer will now show custom colors set for controls even when using the 'Mimic System Style' setting
- The IDE now supports Windows 11 Snap Layouts in the title bar
- Searching lets you exclude folders (with defaults set up for C++ and Delphi)



IDE Quality

Lots of quality work, but especially want to note around:

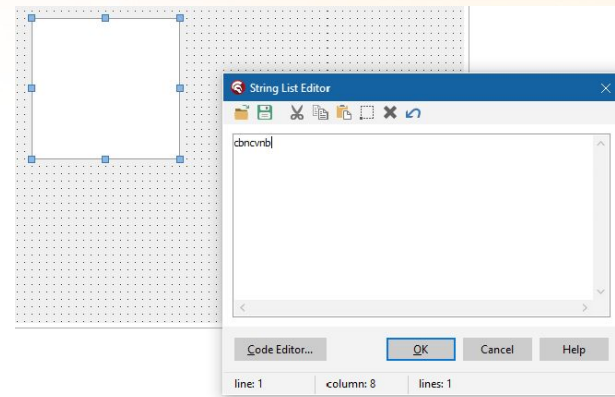
- Multiple editor windows
- Searching (eg unwanted behaviour for editor double clicks, or whole word searches, etc)
- The VCL form designer including in high DPI mode

GDI resource usage should be radically decreased, eg from ~4000 to ~1500 with several component packs installed; we make the ability to implement the same thing available in the VCL

New VCL Designers (from KSVC)

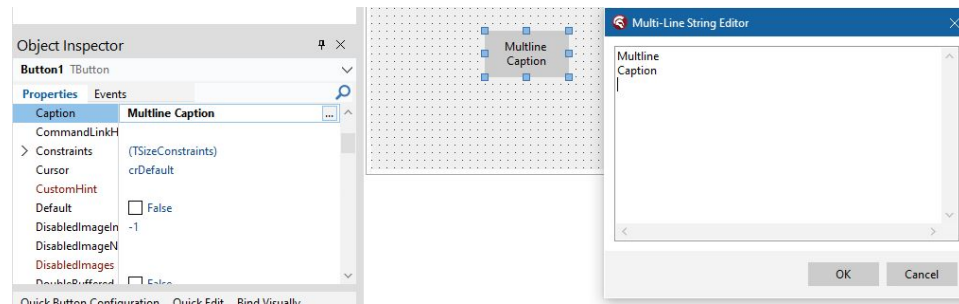
New StringList Editor

- 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



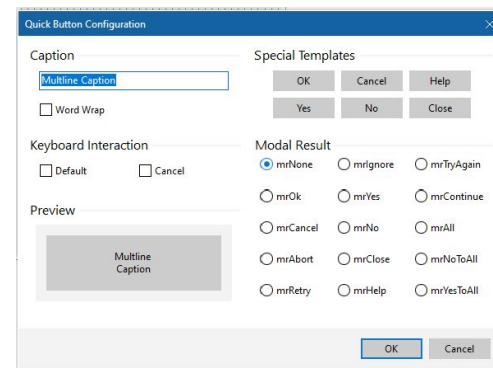
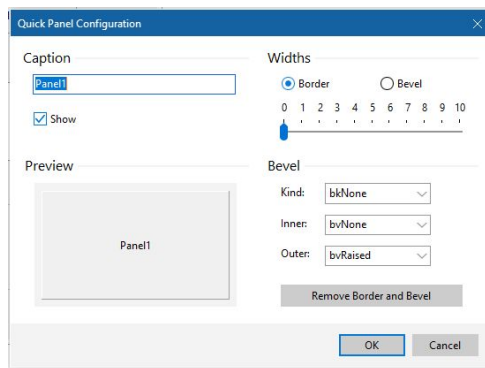
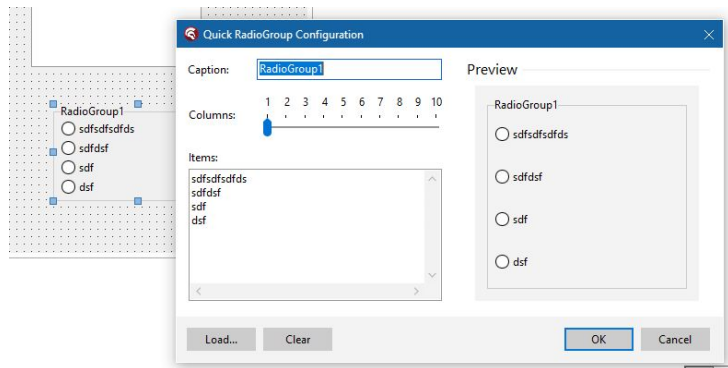
New VCL Quick Editors (from KSVC)

Similar to other “Quick” editors already in the IDE

Quick RadioGroup
Configuration
Configuration

Quick Panel
Configuration

Quick Button
Configuration



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

ToolsAPI

We love the ToolsAPI: it lets our amazing developers build amazing things!

New APIs building on last release's editor info and editor painting APIs:

- Line state (eg, 'inactive breakpoint', 'compiled with blue dot', etc)
- Cell state (eg, selected, hotlinkable, search match, syncedit, brace match, etc)

Should assist fully custom painting.

New APIs for reading and writing to the editor

- Much higher performance than existing stream/file-like APIs; random access supported

Misc improvements: assist setting up IDE title bar, getting editor token, themed input and message dialogs (same as IDE uses), plus some quality

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)

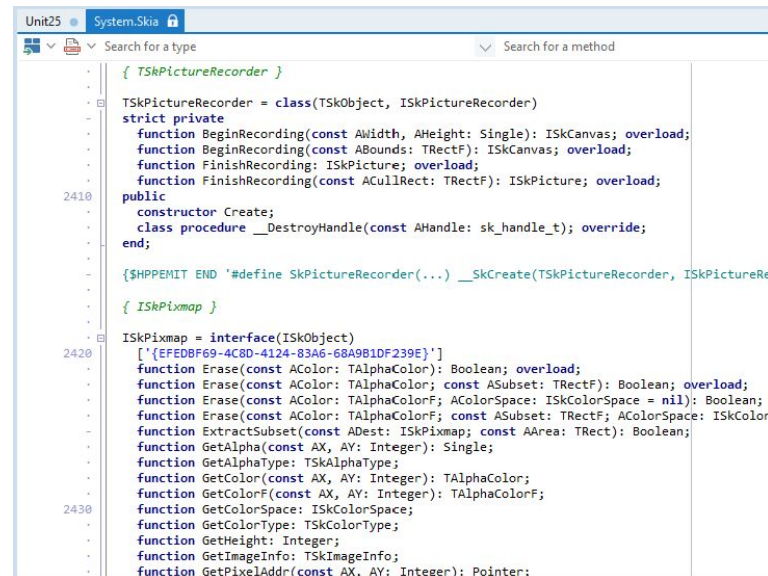
Feature	Details
Text	Multiple styles, max lines, line spacing, justified text, text outline, gradient, and decorations
Right-To-Left languages	Rendering of texts in Persian, Arabic, Hebrew, etc
PDF	Generation of vectorized PDF
Unicode	Graphemes parser
Filters	Color, mask, and image filters
Clippings	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



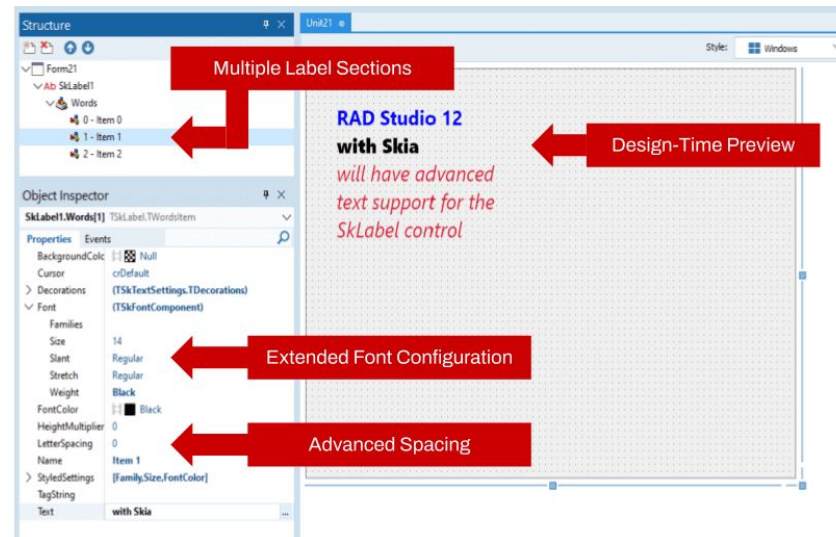
Skia in RAD Studio

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

2. UI Controls

(for both FMX and VCL)

- TSkAnimatedImage
- TSkLabel
- TSkPaintBox
- TSkSvg



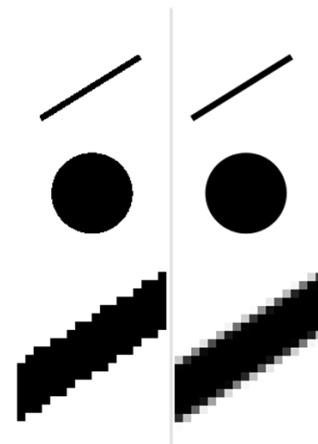
Skia in RAD Studio

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



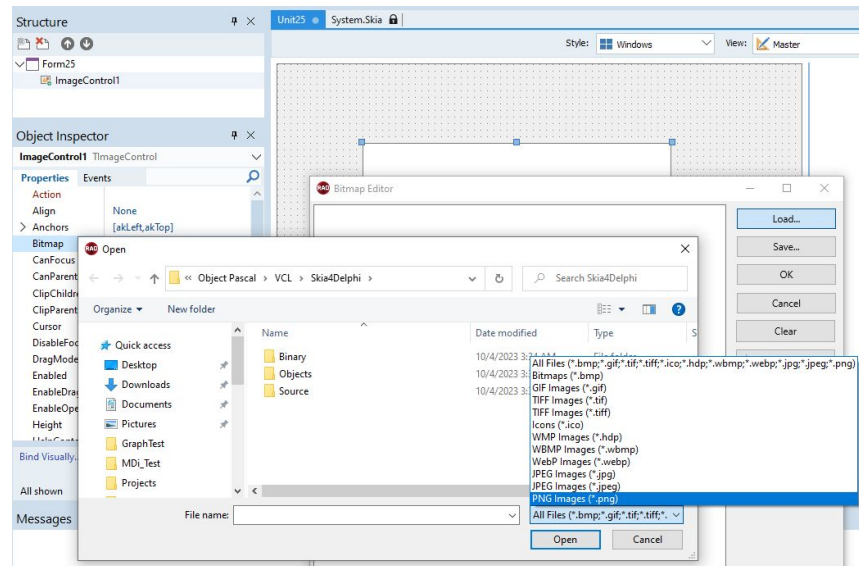
Improved drawing with anti-aliasing in Skia (on the right)

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

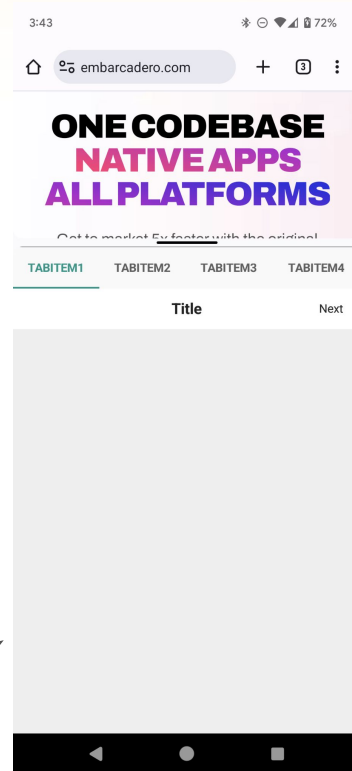
- Refactoring Virtual Keyboard: moved the logic for toolbar alignment/animation
- Improved the magnifier glass position for TMemo

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



UI Libraries: VCL

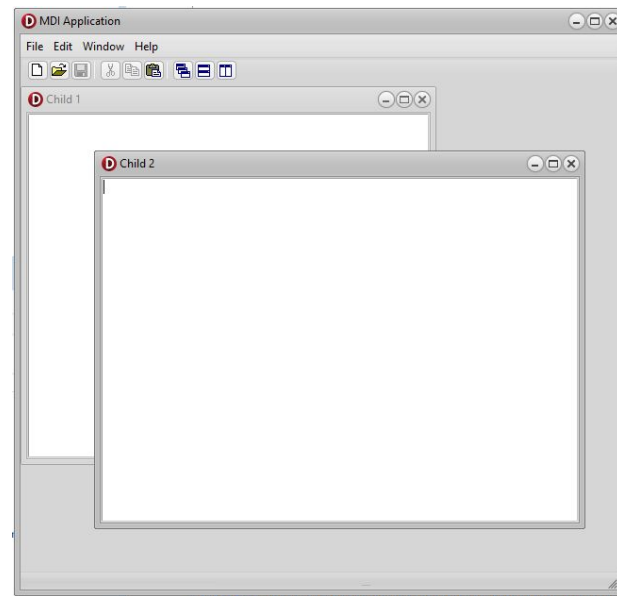
The background of the slide is an abstract composition. It features a gradient from a deep blue on the left to a dark red on the right. Overlaid on this are several large, semi-transparent, wavy shapes in shades of red and blue. Thin, white, hand-drawn style lines are scattered across the entire background, some forming rectangular outlines and others being random strokes.

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:

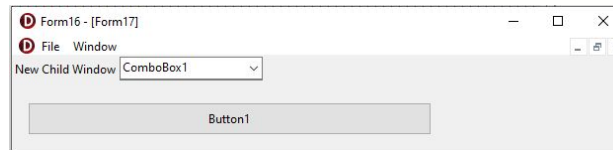
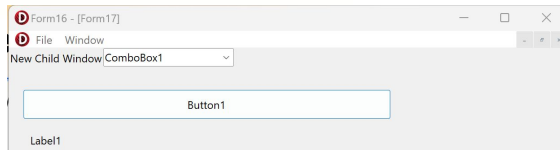
- Paint a custom border for MDI child window
- Support for a custom border for forms with Parent Assigned property (also outside of the MDI scenario)

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

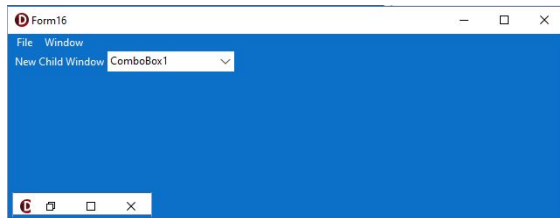


Reworked VCL MDI architecture (Examples)

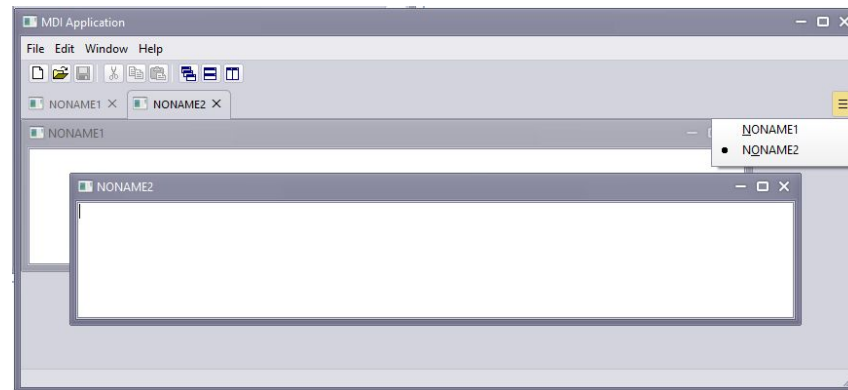
- Maximized child in unstyled app (11.3 -> 12)



- Minimized styles (12)

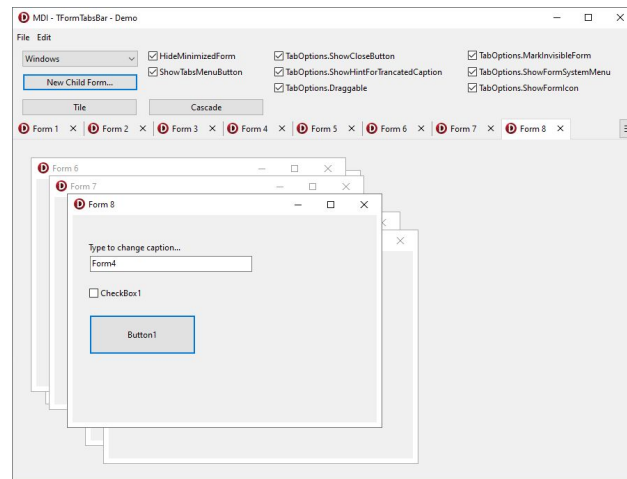


- App Generated by the MDI Wizard -> with styles applied



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
 - TFormVisualManager 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)

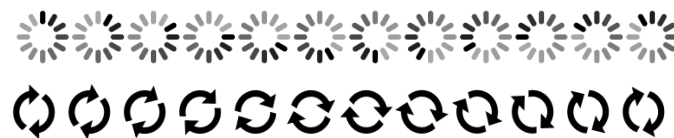
```
for ACtrl in AControl.GetControls(AMyFilter) do  
begin  
    ...  
end;
```

Additional VCL Features (1/2)

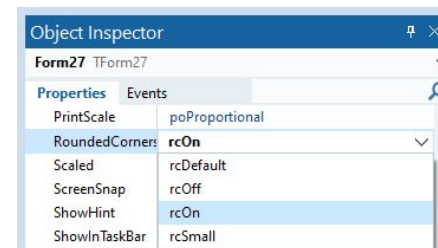
- Tile View support to TListView
 - New vsTile in TViewStyle
 - TileOptions property to adjust tile items
 - TileColumns property to adjust display of sub-item text from columns
- TNumberBox control
 - nbmInt64 mode accepting 64-bit numbers in input
 - Considers MinValue and MaxValue even if they are 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



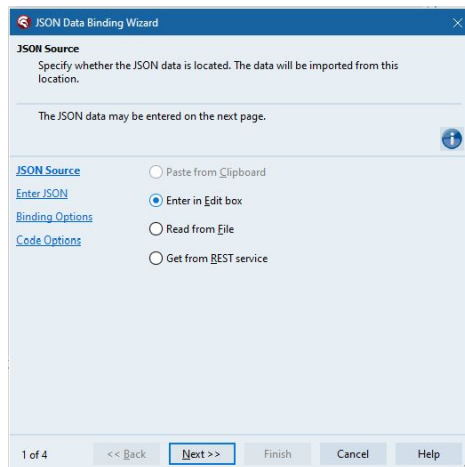


All About Data FireDAC & Web

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



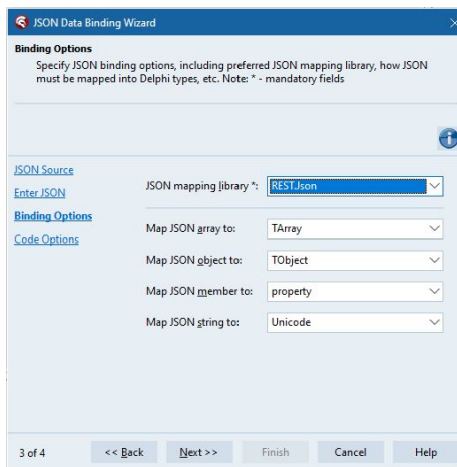
JSON Data Binding Wizard

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 in Clipboard
[Enter JSON](#) ☒ Enter in Edit box
[Binding Options](#) ☐ Read from File
[Code Options](#) ☐ 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](#)
[Enter JSON](#)
[Binding Options](#)
[Code Options](#)

JSON mapping library*: REST.Json

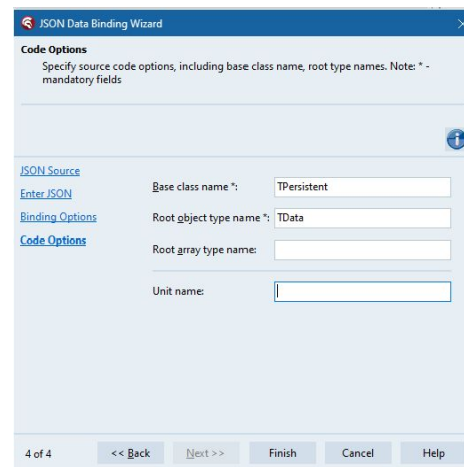
Map JSON array to: TArray

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](#)
[Enter JSON](#)
[Binding Options](#)
[Code Options](#)

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

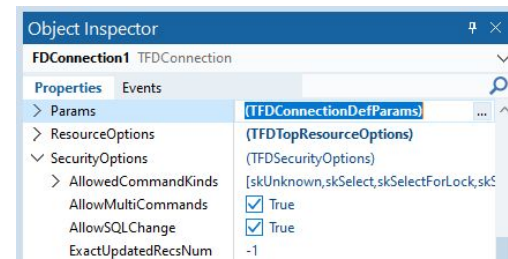
```
{
  "id": 1,
  "name": "Leanne Graham",
  "username": "Bret",
  "email": "Sincere@april.biz",
  "address": {
    "street": "Kulas Light",
    "suite": "Apt. 556",
    "city": "Gwenborough",
    "zipcode": "92998-3874",
    "geo": {
      "lat": "-37.3159",
      "lng": "81.1496"
    }
  },
  "phone": "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;
•
•   TJSONMapper<Users>.SetDefaultLibrary('System.JSON.Serializers');
39   var LUsers := Users.FromJSON(RESTResponse1.JSONText);
40
•
•   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;
• end;
```

FireDAC More Secure Coding

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

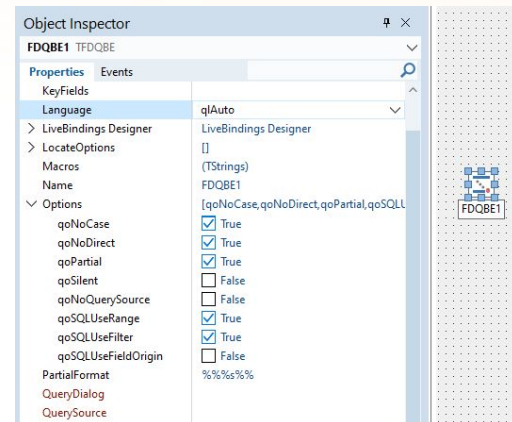
- AllowedCommandKinds: restricts the categories of SQL operations a query can perform
- AllowMultiCommands: disables the execution of multiple SQL commands in a single query (if the underlying DB supports it)
- AllowSQLChange: can prevent the code to change the text of a FireDAC SQL query at runtime
- 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)



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

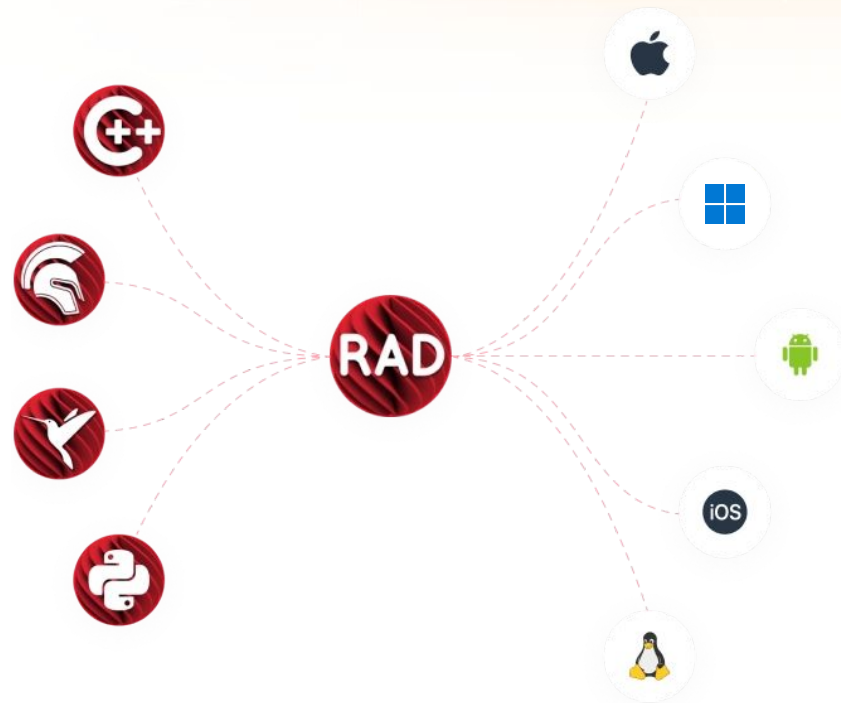
- Performance improvements by embedding FastMM5
- Paging allows page size override in client request
- Session authentication improvements



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)

[illegible]

```
26  {$TEXTBLOCK CRLF JSON}
    *
    *  const
    *      strJSON = '''
    *          [
    *              {"id" : "1", "name" : "Large"},
30      *              {"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

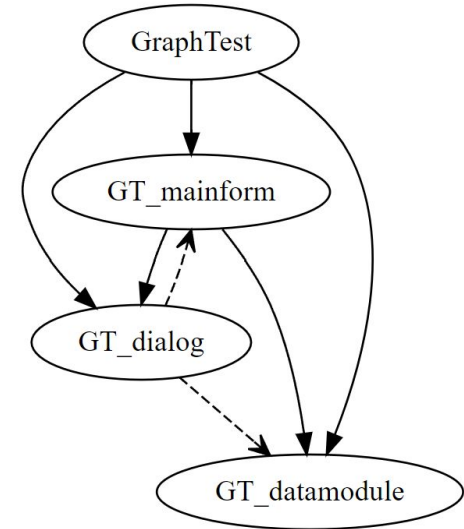
Support **NaN comparisons** as required by IEEE (Win32 compiler)

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



Delphi Support for Android API 33

- Android API Level 33
 - Required for new apps published on Google Play Store
- Full support for latest Android
 - Including updates to some core Android APIs
 - Android Scoped Storage model support
 - IFMXCameraService and IFMXTakenImageService updated for this
 - IFMXExtendedClipboardService updated to prevent sensitive content
 - Cleaned up Uses Permissions page
 - New Android Splash Screen
 - Google Play Billing library 6.0.1
 - New built-in .dex.jar files

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

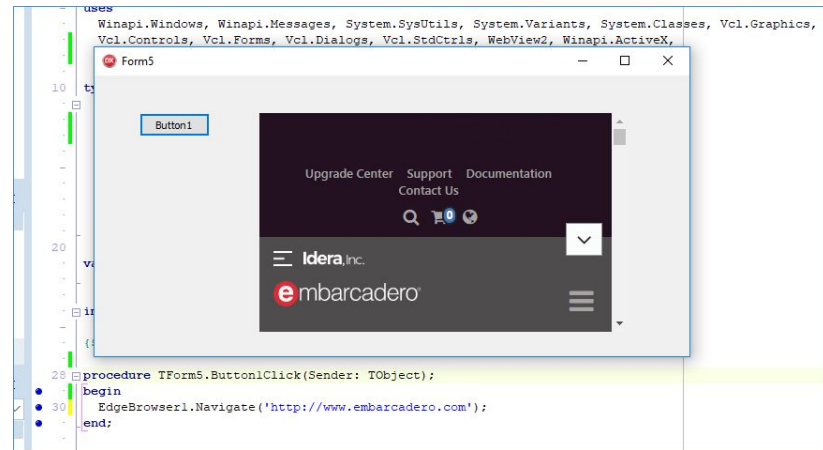
+ Install

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



Changes to Floating-Point Exceptions

- Disabling Floating-Point Exceptions on All Platforms
 - The RTL used floating-point number exceptions since the early days
 - Modern operating systems and their official dev tools, including Windows, macOS, and Linux, decided not to use the FP exception model
 - This is causing a mismatch with platform libraries, we are solving by adapting our model
 - Old behavior can be restored via `System.math.SetExceptionMask`



Quality and Stability

Quality Improvements and Feature Extensions in All Subsystems

Extensive quality work and many small improvements, in

- IDE
- Debuggers
- Platform toolchains and compilers
- Delphi LSP
- Delphi RTL
- FireDAC database
- FireMonkey
- VCL
- RAD Server... and all other subsystems in RAD Studio

Quality, Quality

- RAD Studio 12.0 is providing a **fix** for:
 - **1,027** issues reported by customers on Quality Portal (quality.embarcadero.com)
 - 877 bug reports
 - 150 feature requests
 - 1,809 issues in total including internal and public reports and features
 - 1,471 bugs reports
 - 338 feature requests

This doesn't include duplicates, issues that no longer apply or we decided not to fix. The overall total of QP issues addressed is **1,264**

RAD Studio, C++Builder, Delphi

12.1

released April 2024



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)



Round up of product features and directions

- 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
- Infrastructure
- Summary



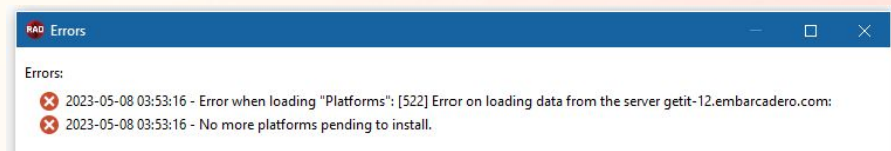
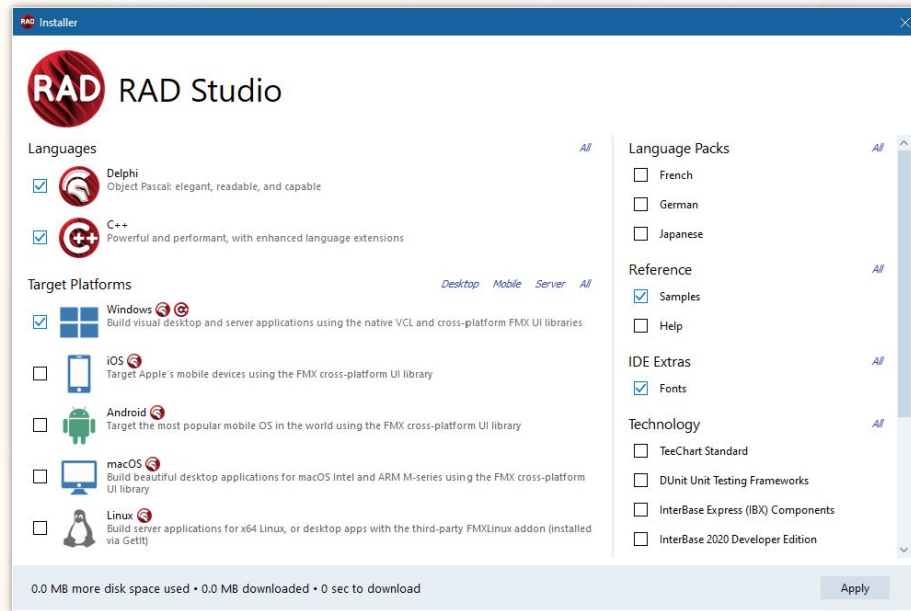
IDE Improvements

The background of the slide is an abstract composition of soft, wavy lines and semi-transparent geometric shapes. The color palette is warm, featuring various shades of yellow, orange, and pink. The overall effect is a modern, flowing, and layered visual texture.

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

Structure

 **GetImageIndex**: Integer
 **GetTabHintText**: string
 **Close**(var Allowed: Boolean)
 **GetTabColor**: TColor
 **LoadViewState**(const Desktop: TCustomIniFile; const ViewDeskSection: string)

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

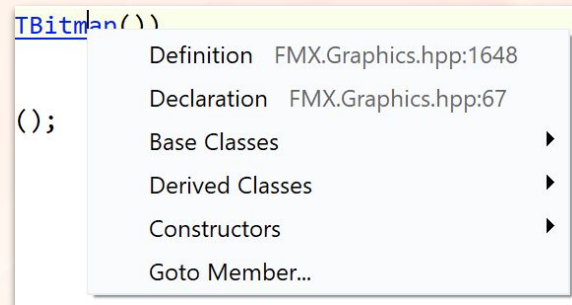
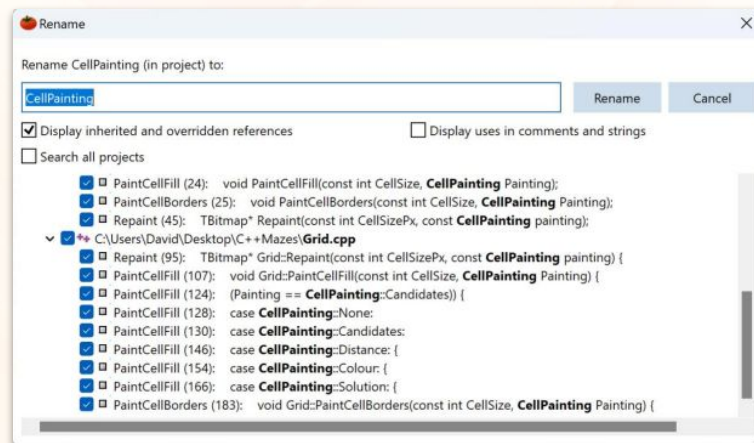
337 result(s) found.

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

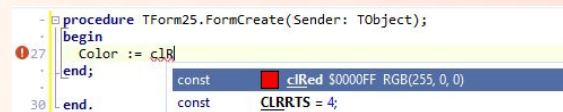
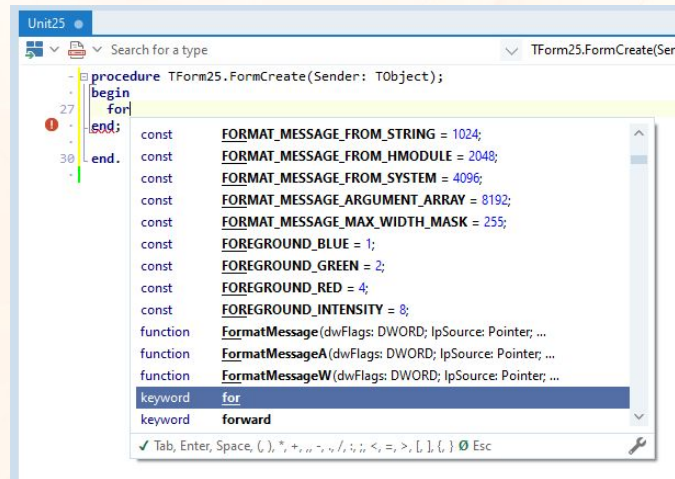
Code Insight (completion etc) uses VA by default



Delphi LSP (12.0)

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

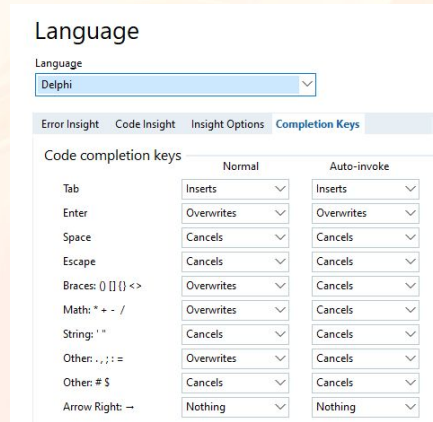
- Autocompleting [] for array types
- Improved color constant representation
- Re-introduced code completion's auto-invoke feature (off by default in this release)
- Language keywords included in the completion list
- Templates still included, but in correct locations
- New menu item to restart the LSP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

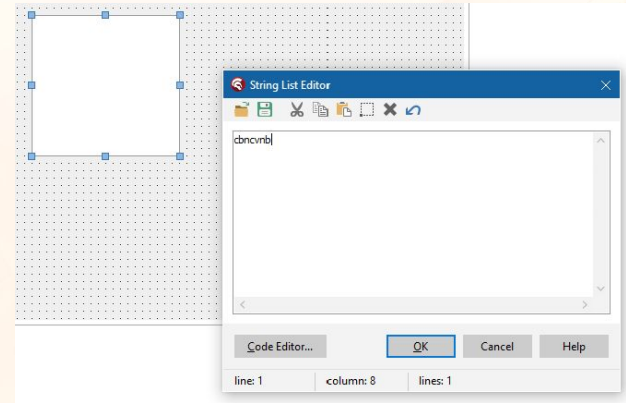
- Ability to customize the behavior of specific keystrokes or ‘acceptance keys’ when code completion is shown
 - When shown normally
 - When auto-invoked
- Customers can configure completion to match preferred behavior
- Added a number of improvements to Delphi LSP overall
 - Several fixes for class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



New VCL Designers (from KSVC)

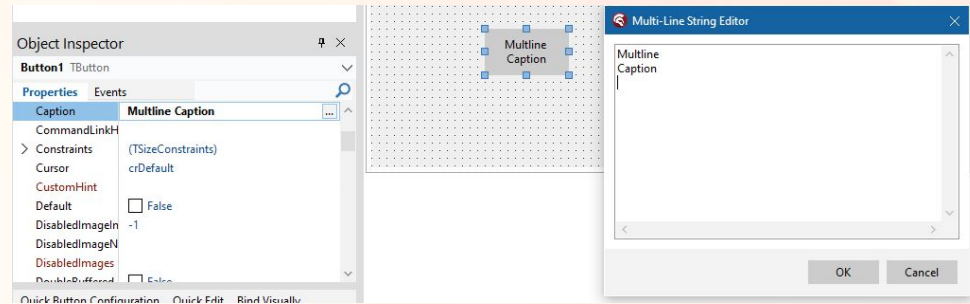
New StringList Editor

- 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)

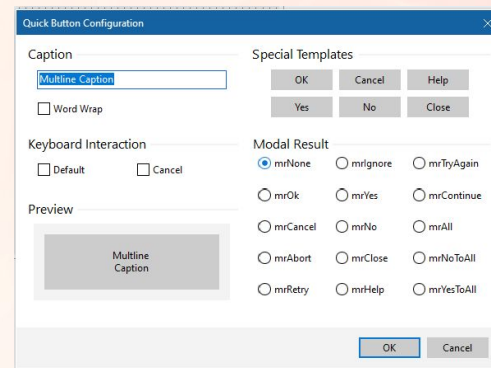
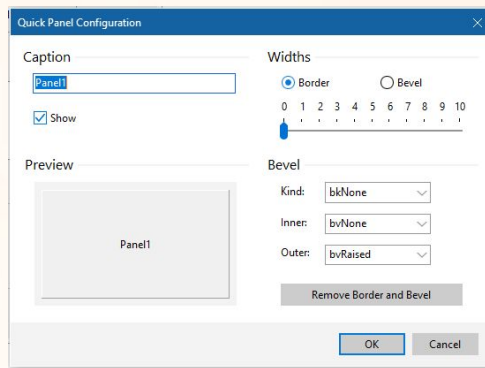
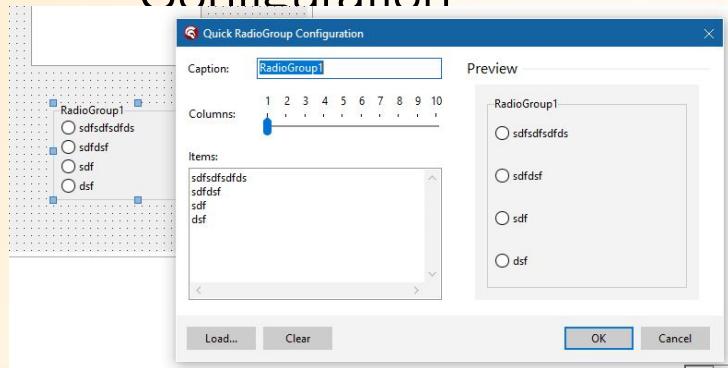
Similar to other “Quick” editors already in the IDE

Quick RadioGroup

Quick Panel

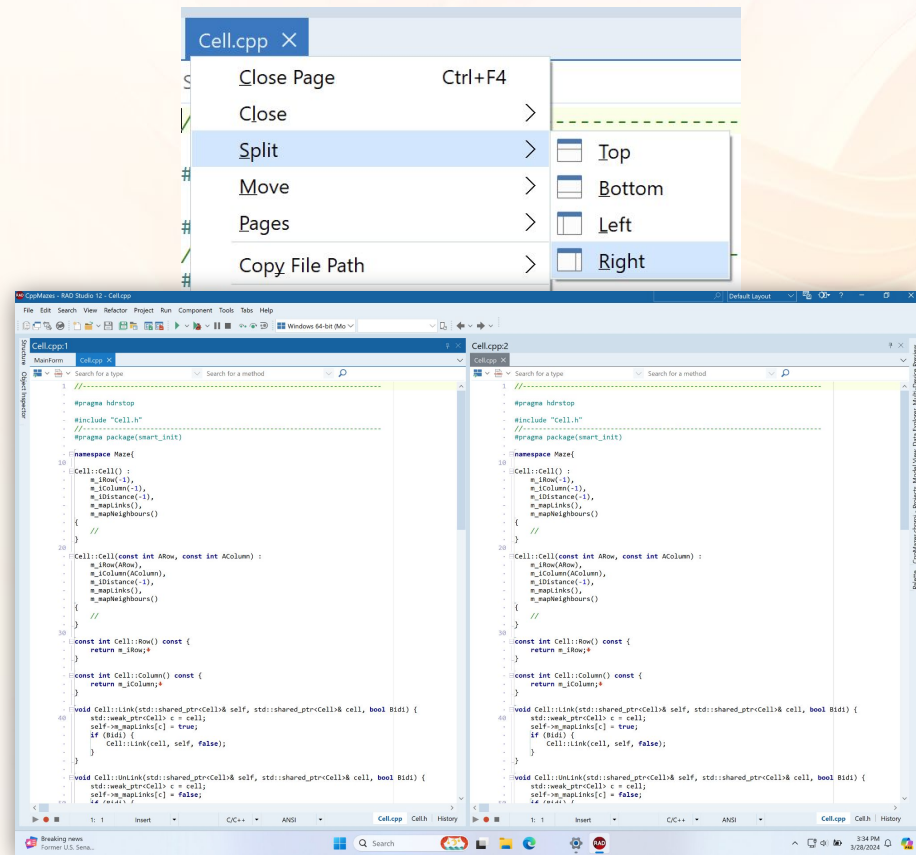
Quick Button

Configuration



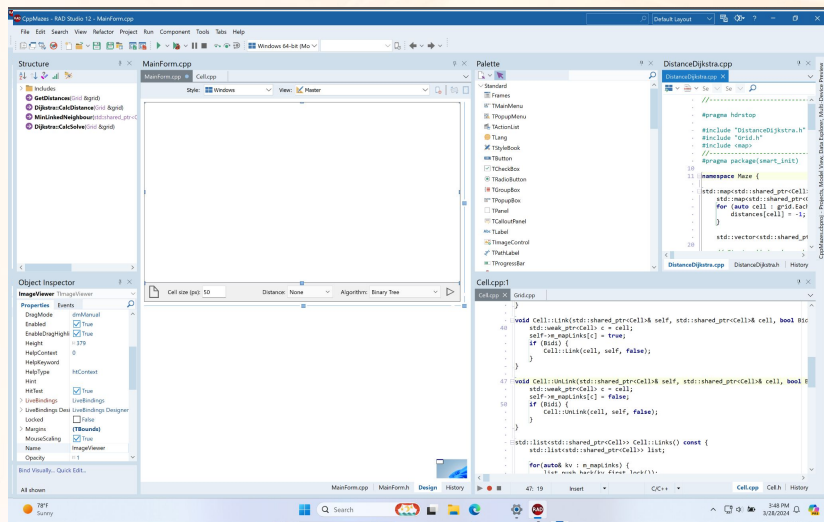
Editor split views

- 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...



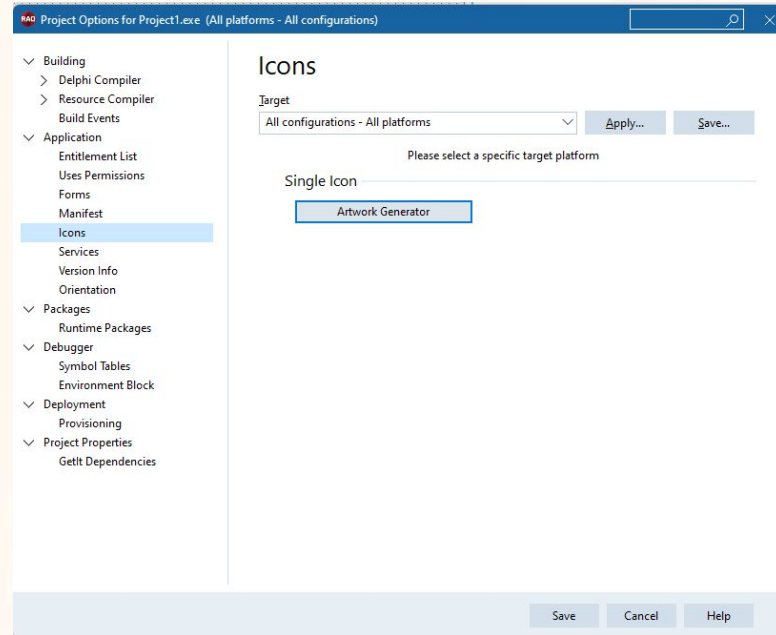
Why editor split views?

- 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



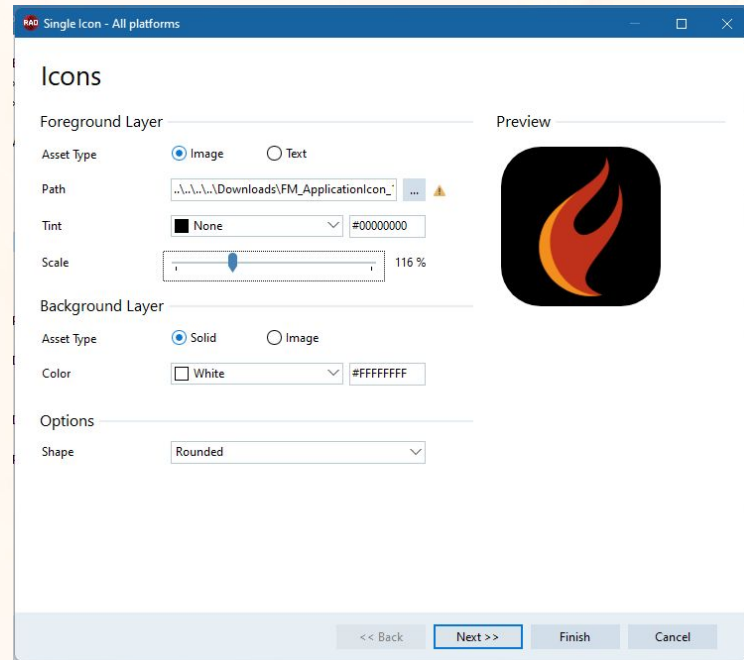
Multi Device Icon Generator

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



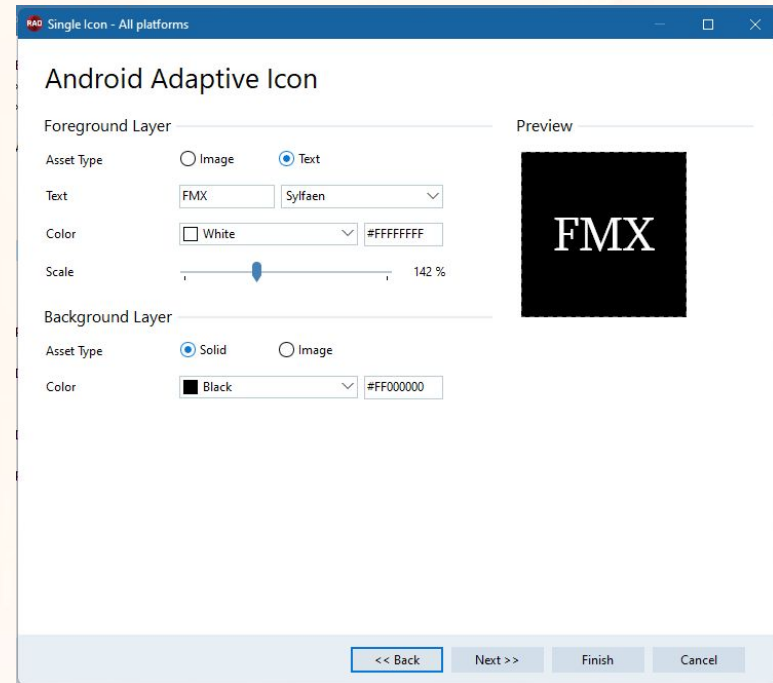
Icon Configuration

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



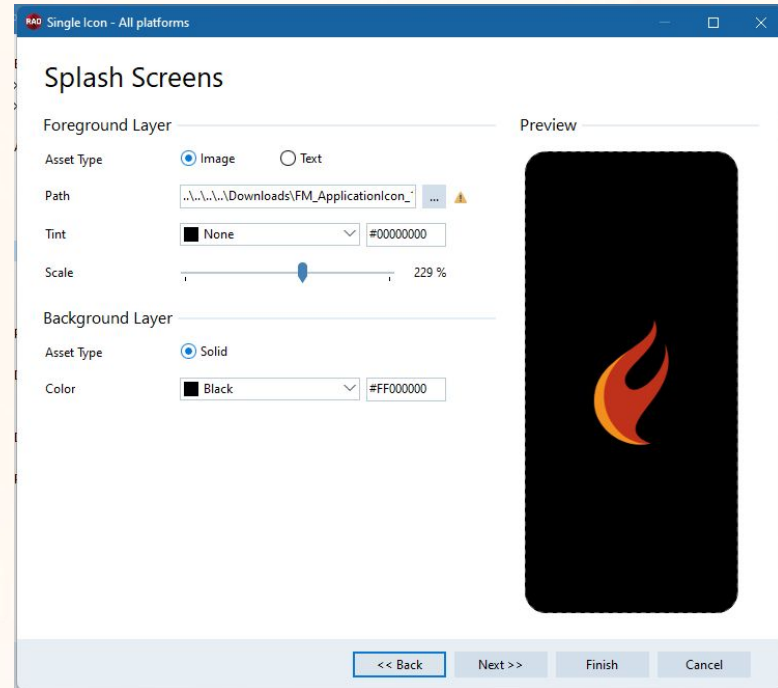
Android Adaptive Icon

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



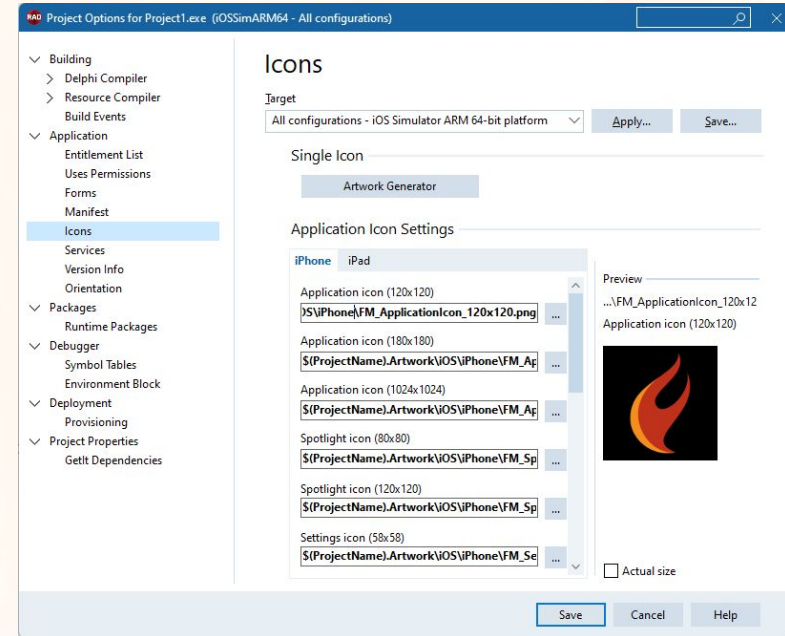
Splash Screens

- 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



Splash with multiple items



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



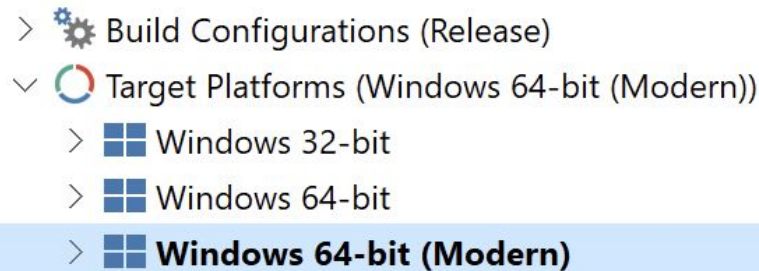
Win64 Clang Toolchain

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

- 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

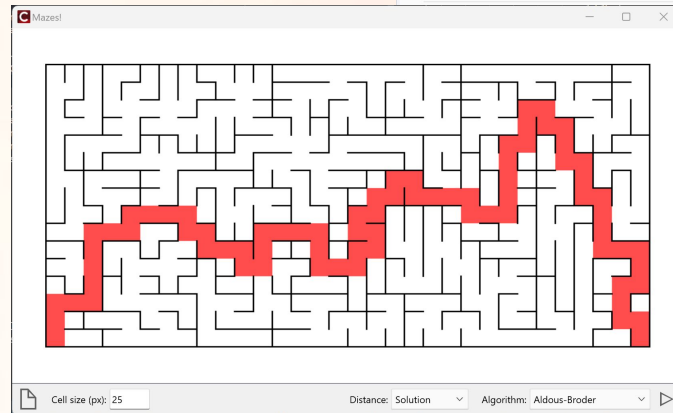
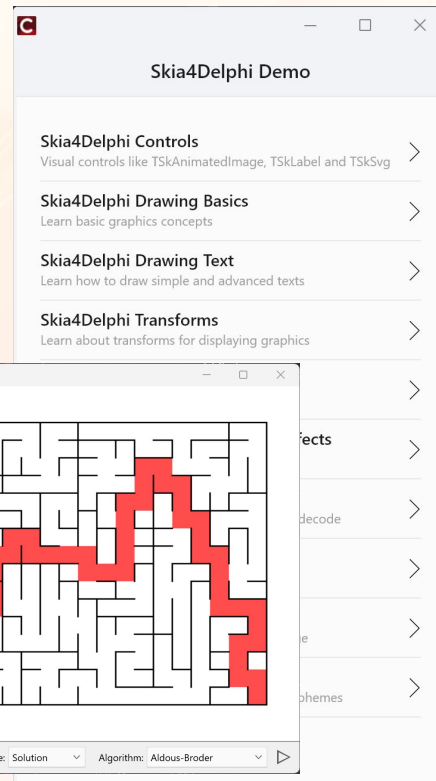


The value of C++Builder with a high quality toolchain for modern C++

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
 - I.e. 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*

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

Feb 2024: Behind the Build for 12.1:

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



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



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

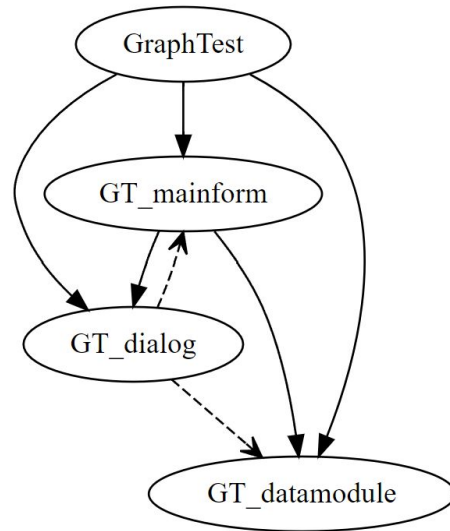
Support **NaN comparisons** as required by IEEE (Win32 compiler)

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



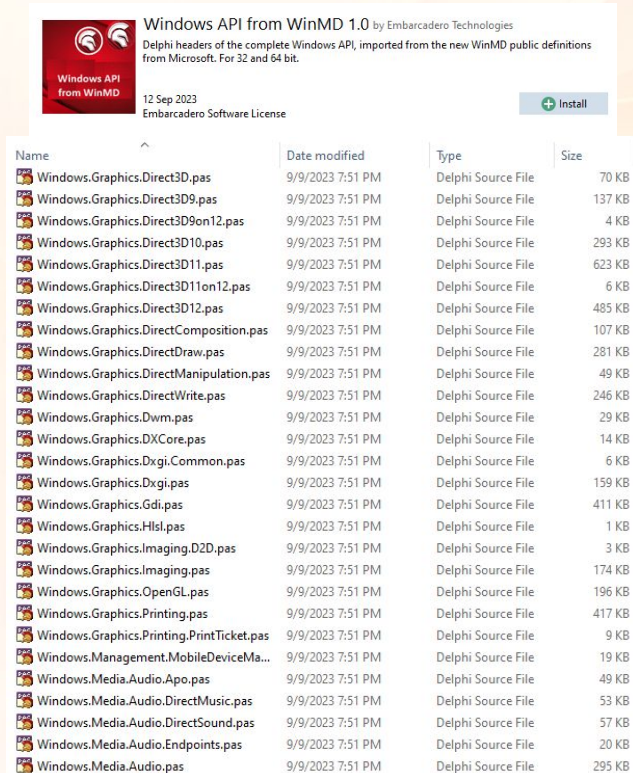
Complete Windows API Integration

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

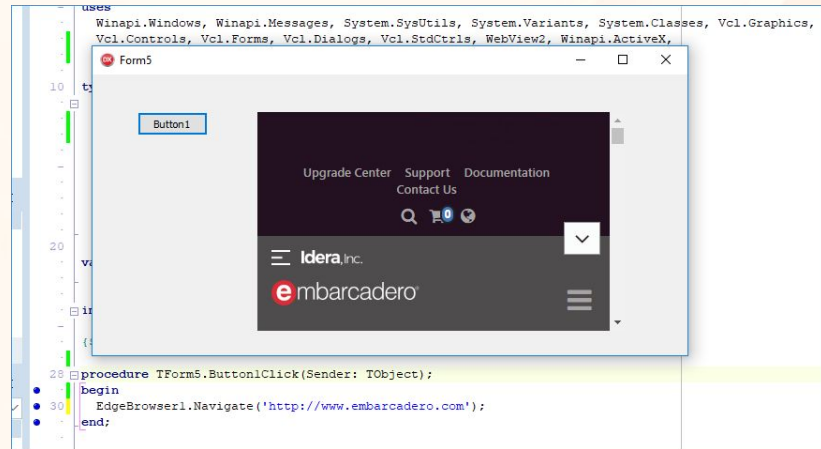
+ Install

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

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)

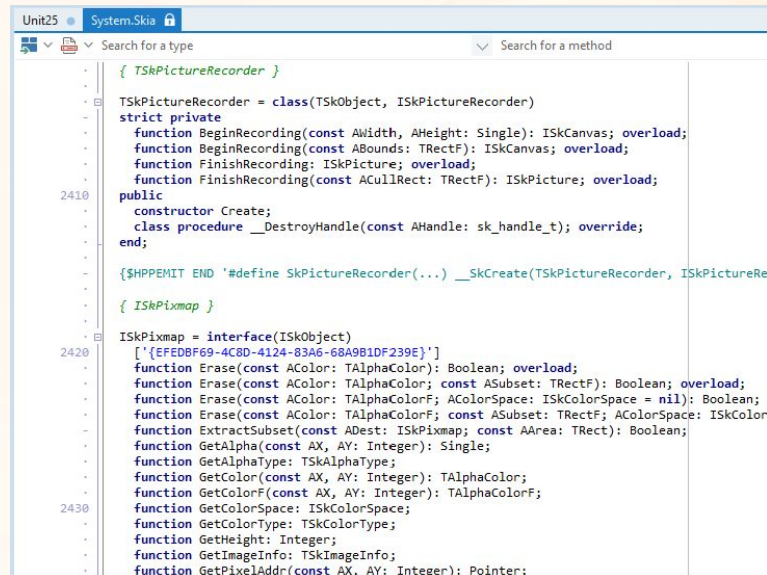
Feature	Details
Text	Multiple styles, max lines, line spacing, justified text, text outline, gradient, and decorations
Right-To-Left languages	Rendering of texts in Persian, Arabic, Hebrew, etc
PDF	Generation of vectorized PDF
Unicode	Graphemes parser
Filters	Color, mask, and image filters
Clippings	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



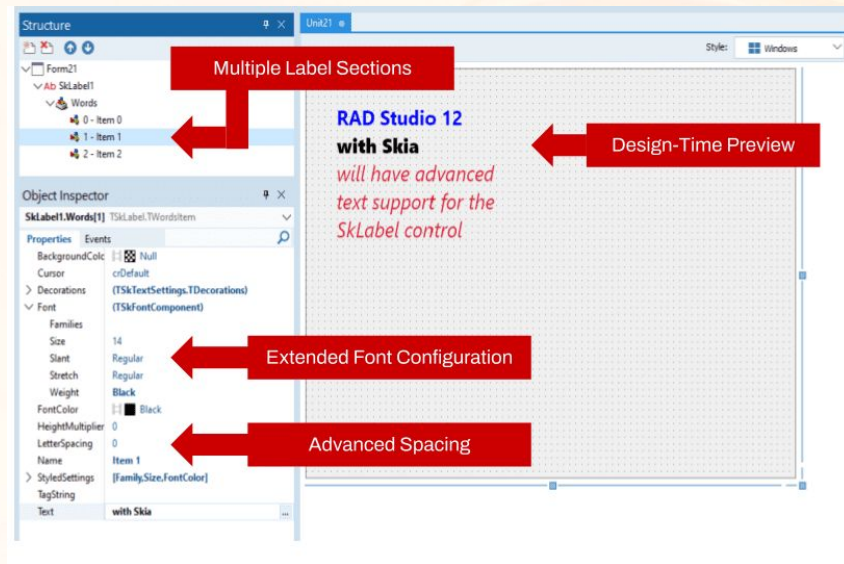
Skia in RAD Studio

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



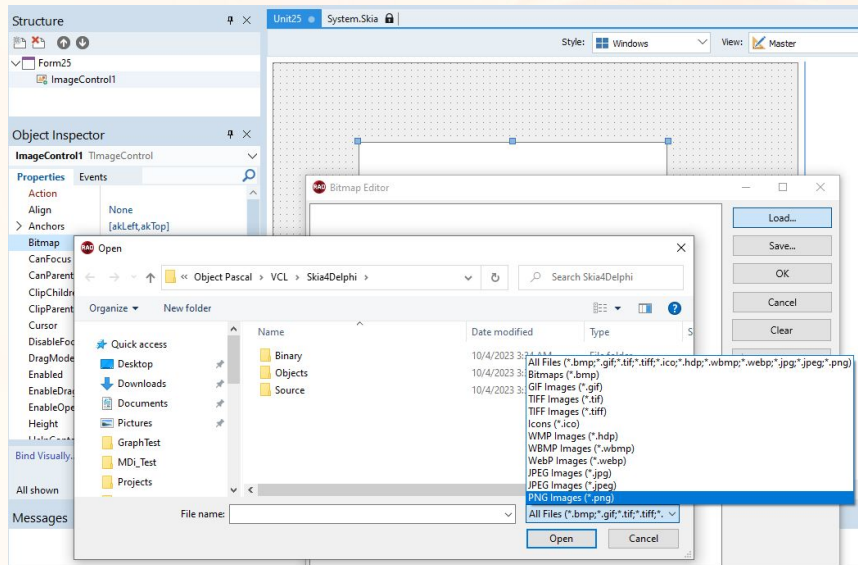
Improved drawing with anti-aliasing in Skia (on the right)

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

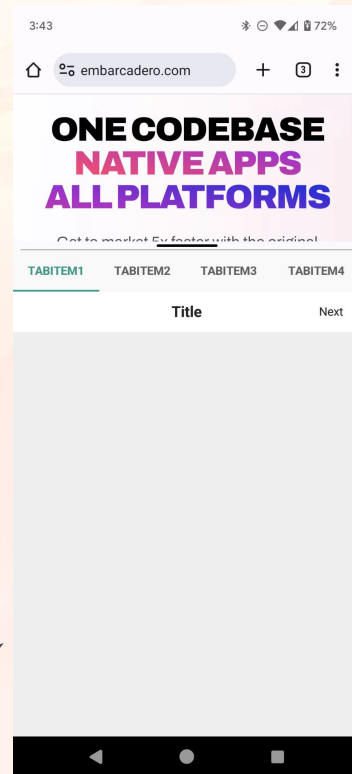
- Refactoring Virtual Keyboard: moved the logic for toolbar alignment/animation
- Improved the magnifier glass position for TMemo

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

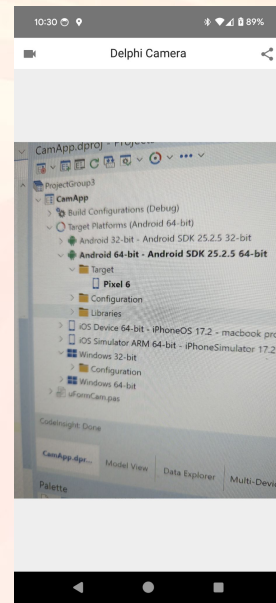
Additional FireMonkey Improvements

- 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



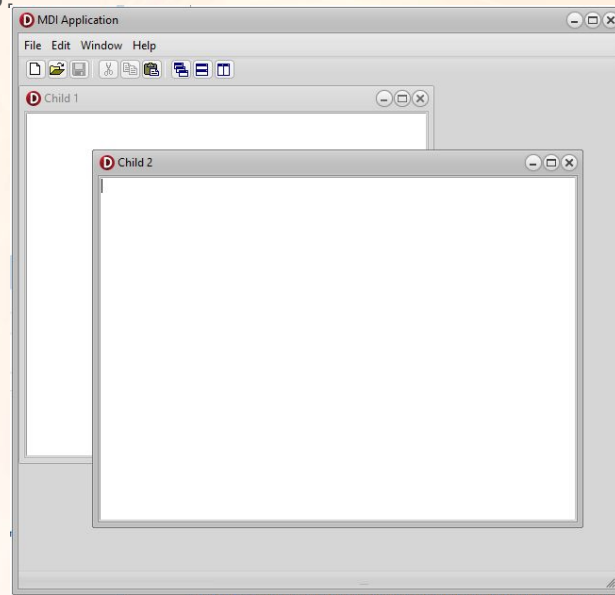
UI Libraries: VCL

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:

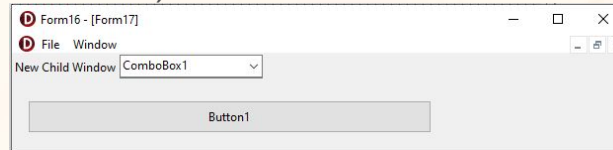
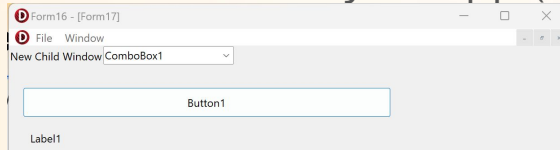
- Paint a custom border for MDI child window
- Support for a custom border for forms with Parent Assigned property (also outside of the MDI scenario)

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

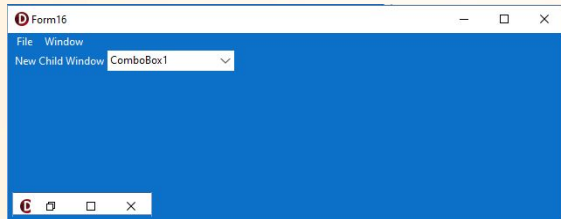


Reworked VCL MDI architecture (Examples)

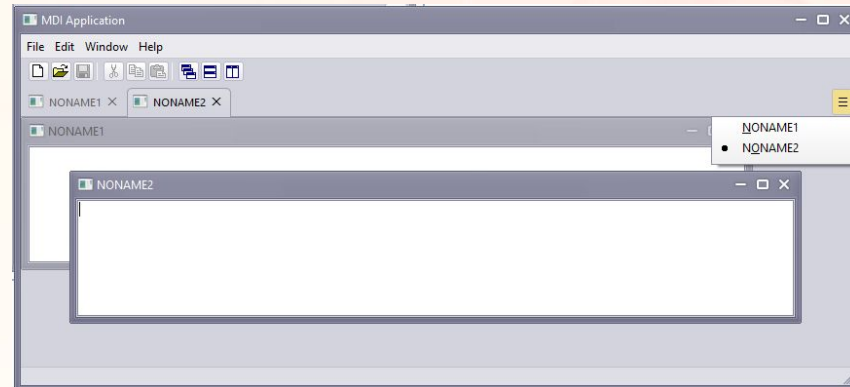
- Maximized child in unstyled app (11.3 -> 12)



- Minimized styles (12)

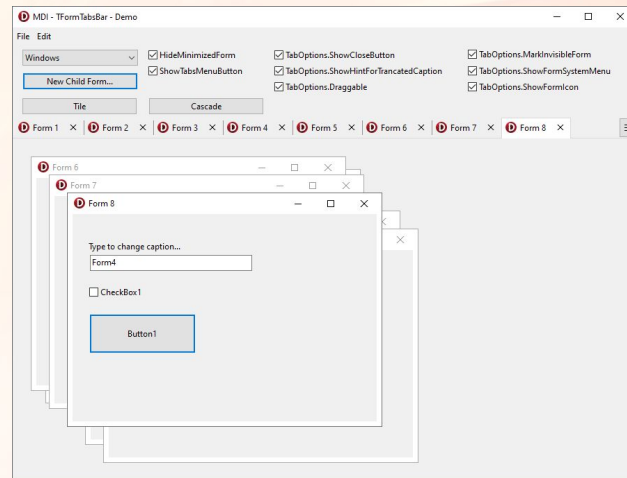


- App Generated by the MDI Wizard -> with styles applied



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



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)

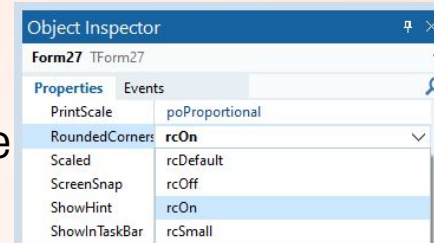
- **Tile View support to TListView**
 - New vsTile in TViewStyle
 - TileOptions property to adjust tile items
 - TileColumns property to adjust display of sub-item text from columns
- **TNumberBox control**
 - nbmInt64 mode accepting 64-bit numbers in input
 - Considers MinValue and MaxValue even if 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

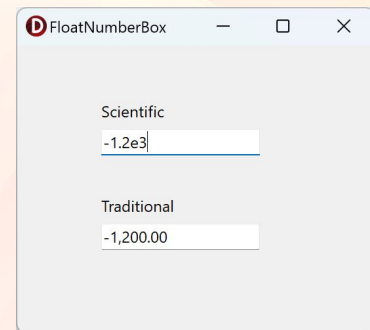



- ShowInTaskbar property for TForm (offering more flexibility in taskbar management)
- Desktop Windows Manager Enhancements
 - Win11 RoundedCorners and EnableImmersiveDarkMode
- New Double Buffering Mode



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



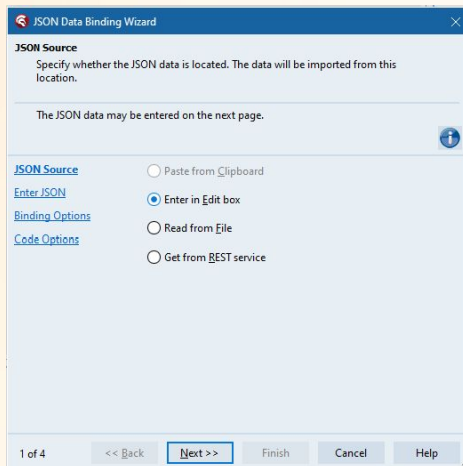


All About Data FireDAC & Web

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



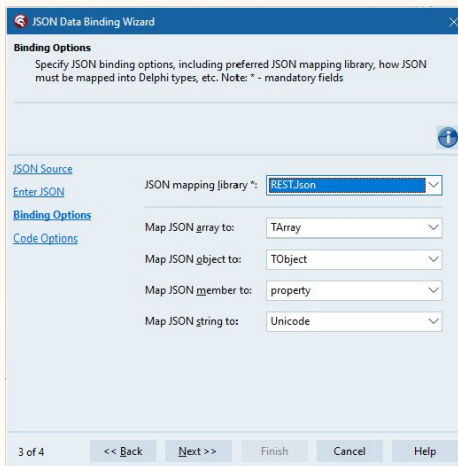
JSON Data Binding Wizard

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 JSON](#) ☒ Enter in Edit box
[Binding Options](#) ☐ Read from File
[Code Options](#) ☐ 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](#)
[Enter JSON](#)
[Binding Options](#)
[Code Options](#)

JSON mapping library: REST.Json

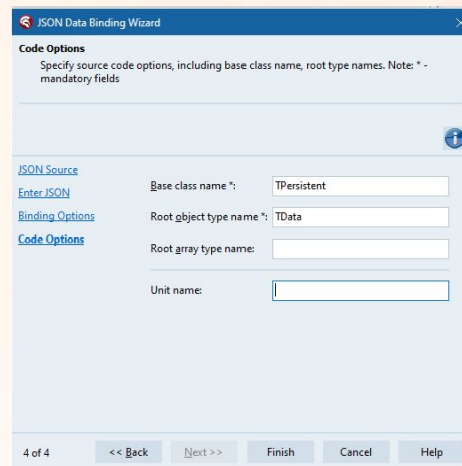
Map JSON array to: TArray

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](#)
[Enter JSON](#)
[Binding Options](#)
[Code Options](#)

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;

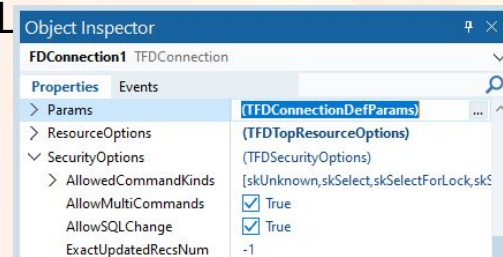
  TJSONMapper<Users>.SetDefaultLibrary('System.JSON.Serializers');
  var LUsers := Users.FromJSON(RESTResponse1.JSONText);

  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);
    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

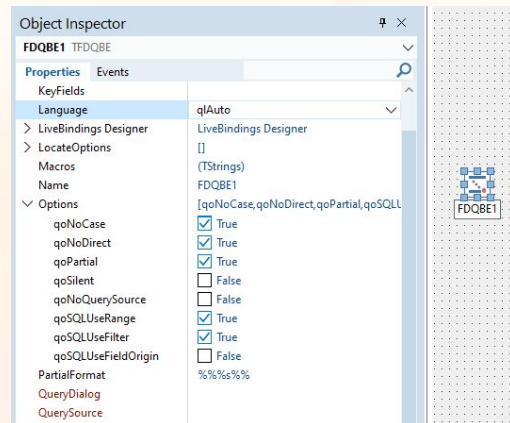
- AllowedCommandKinds: restricts the categories of SQL operations a query can perform
- AllowMultiCommands: disables the execution of multiple SQL commands in a single query (if the underlying DB supports it)
- AllowSQLChange: can prevent the code to change the text of a FireDAC SQL query at runtime
- 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 TFD FBBBackup 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

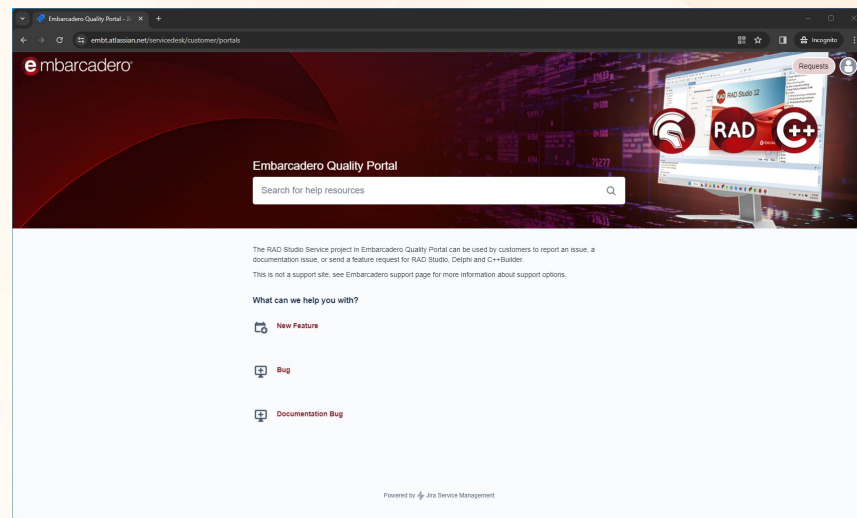
The background is a vibrant, abstract composition. It features a series of overlapping, wavy lines that create a sense of movement and depth. These lines are primarily in shades of orange, yellow, and pink, with some areas appearing more saturated than others. Interspersed among these organic shapes are several geometric elements, including semi-circles, rectangles, and triangles, some of which are solid colors and others that are semi-transparent, allowing the underlying patterns to show through. The overall effect is a modern, artistic, and energetic visual field.

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 is an abstract composition of soft, wavy lines and overlapping geometric shapes. The color palette is warm, featuring shades of yellow, orange, and pink. The lines flow across the frame, creating a sense of movement and depth. Some areas are more saturated, while others are lighter, creating a layered effect.

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 foxes 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)





What's New in <RAD Studio 12.2>

Sponsored by





RAD



What's New in <RAD Studio 12.2>

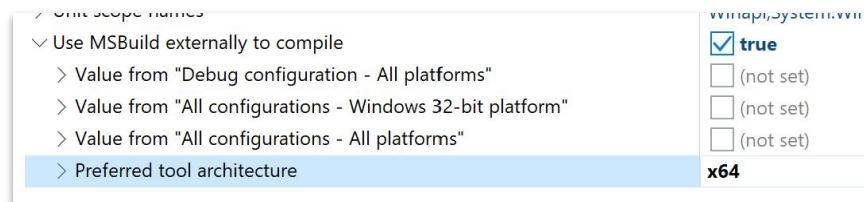
64-bit Version of Delphi Windows compilers

64-bit binaries (large memory address) for Delphi Windows compilers for both Win32 and Win64 **command line compilers**

- Can be enabled in the IDE, when using *external MSBuild*
- Allows building **very large** applications as a single executable
- Available only in **Enterprise** and **Architect**

64-bit binary (large memory address)
version of the **DelphiLSP** executable

- Can be activated in the IDE
- Also available only in Enterprise and Architect



Notice that the new C++ Clang compiler has been a 64-bit binary since release



RAD



What's New in <RAD Studio 12.2>

Smart CodeInsight: AI Powered Coding

Use **AI LLMs** for coding in the RAD Studio IDE

- An **open architecture** for AI plugins

Ready-to-use support for **three online solutions**

- OpenAI
- Gemini by Google
- Claude by Anthropic

and one **offline solution**

- Ollama

Ready to use docker image, works in WSL2





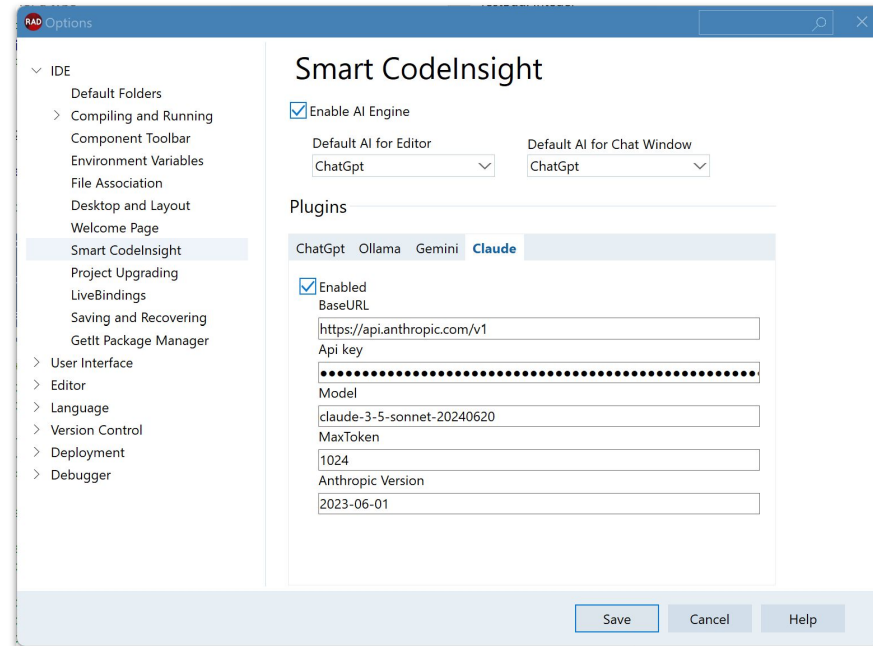
RAD



What's New in <RAD Studio 12.2>

Smart CodeInsight: AI Powered Coding

- Everything is **optional** and **disabled** by default
- **Multiple** providers, so you can **choose** which one or ones **you want** to enable
- A locally installed, **offline** solution for **maximum privacy**
- You decide **if** and **how** to use **AI**
... and you can even **create additional plugins** for your preferred solution





RAD



What's New in <RAD Studio 12.2>

Smart CodeInsight: AI Powered Coding

RAD Project21 - RAD Studio 12 - Unit25

File Edit Search View Refactor Project Run Component

Smart CodeInsight

Answer>

In Delphi, "strict private" is a visibility specifier used in object-oriented programming to control access to class members (fields, methods, properties). It is one of the most restrictive visibility levels available in Delphi. Here's what you need to know about "strict private":

1. Scope: Members declared as "strict private" are only accessible within the class where they are declared.
2. Inheritance: Strict private members are not inherited by subclasses and cannot be accessed by descendant classes.
3. Unit scope: Unlike regular "private" members, strict private members are not accessible even within the same unit where the class is defined.
4. Usage: It's typically used for implementation details that should be completely hidden from outside the class and its descendants.

Claude>What's the meaning of strict private in Delphi?

Claude

Structure **Smart CodeInsight**

IDE: chat window and editor commands

```
26 function TestBug: Integer;
27 var
28   P: Pointer;
29 begin
30   GetMem(P, 100);
31   Result := 100;
32 end;
33
34 {///***** Find Bugs *****/}
35 1. Memory Leak: The allocated memory isn't freed after been used. After calling GetMem(P, 100),
36   the programmer should call FreeMem(P) before the function ends to avoid Leaking memory
37
38 2. Uninitialized pointer: The allocated memory is not initialized. In order to avoid this,
39   it is recommended to initialize the memory after allocating it. The allocated memory can be
40   using FillChar or ZeroMemory functions.
41
42 3. Unused pointer: The pointer 'P' was allocated memory but it's not used anywhere in the function.
43   This failure is not handled in the code.
44
45 The improved version of the code could look like this:
46 ...delphi
47 function TestBug: Integer;
48 var
49   P: Pointer;
50 begin
51   GetMem(P, 100);
52   try
53     // use 'P' for some operations here...
54     // ...
55     FillChar(P^, 100, 0);
56     Result := 100;
57   finally
58     FreeMem(P);
59   end;
60 end;
61 ...
62 }
```

Smart CodeInsight

- Open File at Cursor Ctrl+Enter
- Topic Search F1
- Complete Class at Cursor Ctrl+Shift+C
- Add To-Do Item... Ctrl+Shift+T
- Cut Ctrl+X
- Copy Ctrl+C
- Paste Ctrl+V
- Toggle Bookmarks
- Goto Bookmarks
- Clear Bookmarks
- Debug
- Toggle Comment Ctrl+/- Fold
- Unfold
- Refactor
- Find
- Find Symbols
- Sync Prototypes Ctrl+Alt+Shift+P
- Surround
- Add to Repository
- View Messages
- Read Only
- Editor Options

AI Chat

- Find Bugs
- Explain the Code
- Add Comment
- Complete the Code
- Optimize Code
- Add Unit Test
- Convert to Assembly
- Convert to Delphi
- Convert to C++



RAD




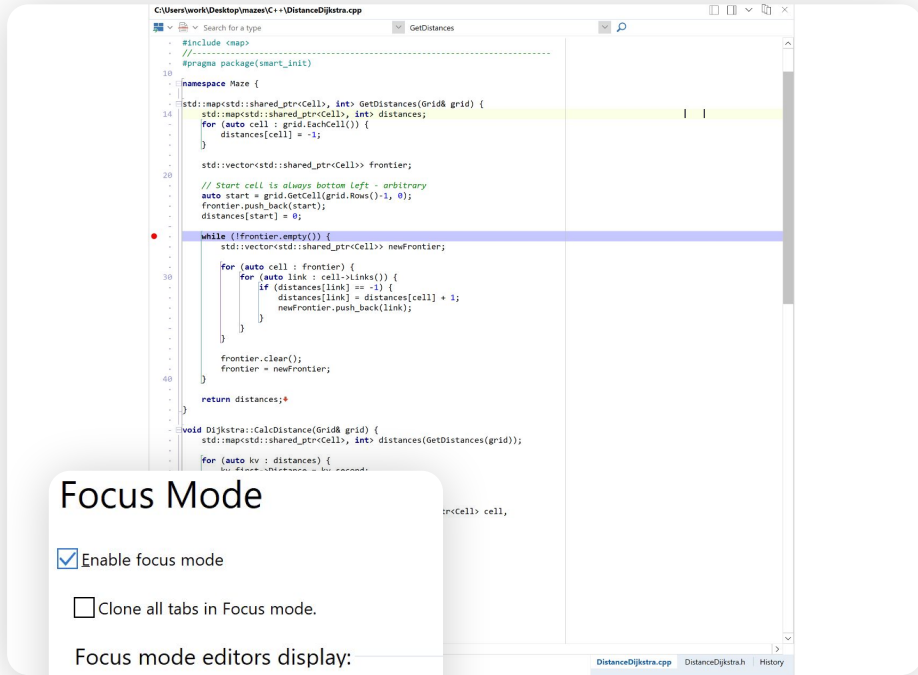
What's New in <RAD Studio 12.2>

Editor: Focus Mode

Progression from 12.1's split views,
and known as 'zen mode' in other
IDEs

Focus Mode leaves the IDE with **only**
your code editor

- Removes other UI chrome
- Multiple side by side editors
- 'Code only'
- Distraction free... *focus!* 



Focus Mode

☒ Enable focus mode

☐ Clone all tabs in Focus mode.

Focus mode editors display:

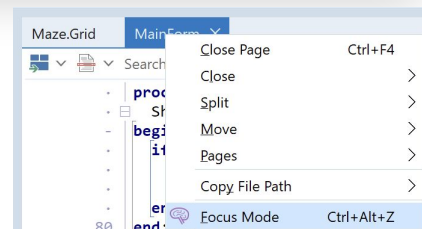
☒ Bookmarks

☐ Right margin

☐ Navigation toolbar

☐ Code folding

☒ Debug Info (Breakpoints, Compiled lines)





RAD



What's New in <RAD Studio 12.2>

Z:\projects\Embarcadero\mazes\mazes\Delphi\MainForm.pas

```
Search for a type TForm1.RecreateGrid

procedure TForm1.edtCellSizeKeyUp(Sender: TObject; var Key: Word; var KeyChar: Char;
  Shift: TShiftState);
begin
  if Key = vkReturn then begin
    RecreateGrid;
    PaintGrid;
  end;
80 end;

procedure TForm1.FormShow(Sender: TObject);
begin
  PaintGrid;
end;

procedure TForm1.RecreateGrid;
var
  R, C : Integer;
90 begin
  C := Trunc(ImageViewer.Width / CellSize) - 1;
  R := Trunc(ImageViewer.Height / CellSize) - 1;

  // Limit the small size
  C := Max(C, 2);
  R := Max(R, 2);

98 FGrid := CreateGrid(R, C);
end;

100 function TForm1.CellSize : Integer;
begin
  Result := 50;
  TryStrToInt(edtCellSize.Text, Result);
end;

procedure TForm1.PaintGrid;
var
  CellDistPainting : TCellPainting;
110 begin
  CellDistPainting := TCellPainting(cmbDistance.ItemIndex);
  ImageViewer.Bitmap.Assign(FGrid.Repaint(CellSize, CellDistPainting));
  Invalidate;
end;

procedure TForm1.btnNewClick(Sender: TObject);
begin
  RecreateGrid;
  PaintGrid;
120 end;

function TForm1.GetMazeGenerationProc : TMazeGenerationProc;
begin
  Result := nil;
  case cmbAlgorithm.ItemIndex of
    0 : Result := BinaryTree;
    1 : Result := AldousBroder;
    2 : Result := RecursiveBacktracker;
  end;
130 end;

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

Code History



RAD



What's New in <RAD Studio 12.2>

Z:\projects\Embarcadero\mazes\mazes\Delphi\MainForm.pas

```
10  type
    TForm1 = class(TForm)
        ToolBar: TToolBar;
        btnNew: TSpeedButton;
        cmbDistance: TComboBox;
        btnPlayStepped: TSpeedButton;
        btnPlay: TSpeedButton;
        ImageViewer: TImageViewer;
        edtCellSize: TEdit;
        btnStop: TSpeedButton;
        cmbAlgorithm: TComboBox;
        Label1: TLabel;
        Label2: TLabel;
        Label3: TLabel;
        procedure btnNewClick(Sender: TObject);
        procedure edtCellSizeKeyUp(Sender: TObject; var Key: Word; var KeyChar: Char;
            Shift: TShiftState);
        procedure FormShow(Sender: TObject);
        procedure btnPlaySteppedClick(Sender: TObject);
        procedure btnPlayClick(Sender: TObject);
        procedure btnStopClick(Sender: TObject);
        procedure cmbDistanceChange(Sender: TObject);
    private
        FGrid: IGrid;
        FStopPlaying: Boolean;
        procedure RecreateGrid;
        procedure PaintGrid;
        function CellSize: Integer;
        procedure DoDijkstra;
        function GetMazeGenerationProc: TMazeGenerationProc;
    public
        { Public declarations }
        constructor Create(AOwner: TComponent); override;
        destructor Destroy; override;
    end;

    var
        Form1: TForm1;

50 implementation

    {$R *.fmx}

    uses
        System.Math, Maze.DistanceDijkstra;

    { TForm1 }

    constructor TForm1.Create(AOwner: TComponent);
60 begin
        inherited;
        FStopPlaying := false;

        RecreateGrid;
    end;

    destructor TForm1.Destroy;
    begin
70 inherited;
```

Code History

Z:\projects\Embarcadero\mazes\mazes\Delphi\MainForm.pas

```
procedure TForm1.edtCellSizeKeyUp(Sender: TObject; var Key: Word; var KeyChar: Char;
    Shift: TShiftState);
begin
    if Key = vkReturn then begin
        RecreateGrid;
        PaintGrid;
    end;
80 end;

procedure TForm1.FormShow(Sender: TObject);
begin
    PaintGrid;
end;

procedure TForm1.RecreateGrid;
var
    R, C: Integer;
90 begin
    C := Trunc(ImageViewer.Width / CellSize) - 1;
    R := Trunc(ImageViewer.Height / CellSize) - 1;

    // Limit the small size
    C := Max(C, 2);
    R := Max(R, 2);

98 FGrid := CreateGrid(R, C);
end;

100 function TForm1.CellSize: Integer;
begin
    Result := 50;
    TryStrToInt(edtCellSize.Text, Result);
end;

procedure TForm1.PaintGrid;
var
    CellDistPainting: TCellPainting;
110 begin
    CellDistPainting := TCellPainting(cmbDistance.ItemIndex);
    ImageViewer.Bitmap.Assign(FGrid.Repaint(CellSize, CellDistPainting));
    Invalidate;
end;

procedure TForm1.btnNewClick(Sender: TObject);
begin
    RecreateGrid;
    PaintGrid;
120 end;

function TForm1.GetMazeGenerationProc: TMazeGenerationProc;
begin
    Result := nil;
    case cmbAlgorithm.ItemIndex of
        0: Result := BinaryTree;
        1: Result := AldousBroder;
        2: Result := RecursiveBacktracker;
    end;
130 end;

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

Code History



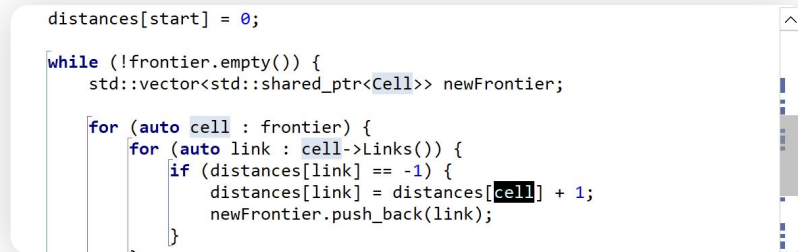
RAD



What's New in <RAD Studio 12.2>

Editor: Scrollbar Annotations


- Editor Scrollbar Annotations
 - See at a glance sections of code modified, with bookmarks, errors, warnings...
- Delphi Inline Assembly Highlighting
- Delphi LSP: code completion optionally in editor colors
- Split Editor Improvements around resizing, stability and IDE toolbars in high DPI quality



```
distances[start] = 0;

while (!frontier.empty()) {
    std::vector<std::shared_ptr<Cell>> newFrontier;

    for (auto cell : frontier) {
        for (auto link : cell->Links()) {
            if (distances[link] == -1) {
                distances[link] = distances[cell] + 1;
                newFrontier.push_back(link);
            }
        }
    }
}
```



```
procedure TForm1.FormCreate(Sender: TObject);
begin
    Self.
end;
end.
```

procedure TForm1.FormCreate(Sender: TObject);
begin
Self.
end;
end.

procedure ArrangeIcons;
procedure Cascade;
procedure Next;
procedure Previous;
procedure Tile;
property ActiveMDIChild: TForm;

✓ Tab Enter — ↑ ↓ ✗ Space " # \$ ' () * + , - . : ; < = > [] { } Esc →

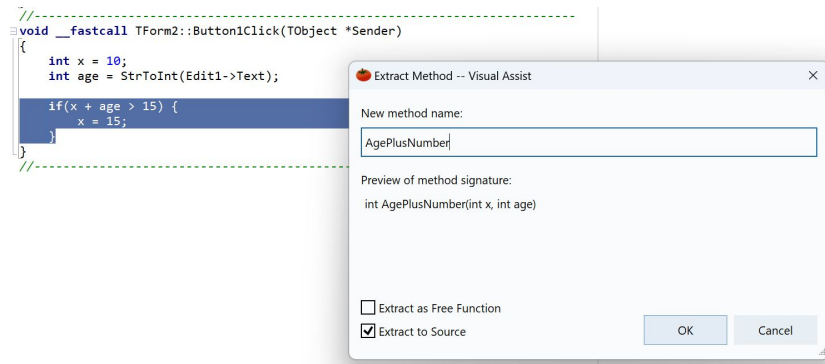
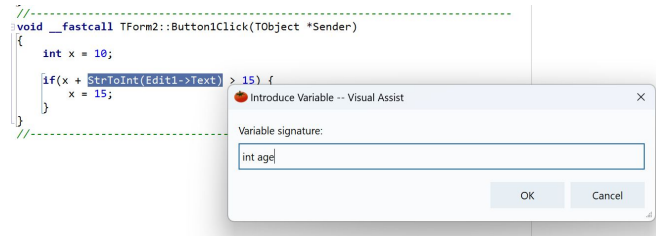


C++Builder Visual Assist Integration

The C++Builder **Visual Assist** integration has been expanded with three new refactorings

- Introduce Variable
- Create From Usage
- Extract Method

Plus **significant** quality and stability improvements



whole **tomato**
Visual Assist



RAD

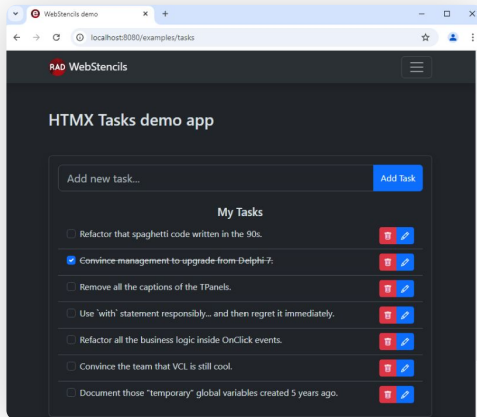


What's New in <RAD Studio 12.2>

WebStencils Template Library

Server-side script-based integration and processing of HTML files for WebBroker and RAD Server. You can develop modern websites

- Based on any JavaScript library, CSS, and web technology
- Powered by the data extracted and processed by a RAD Studio server-side application



- Foundation for HTMX as a web development solution.
- HTMX pages benefit from server-side code generation and hook into REST servers for content updates



WebStencils Template Library

Core Web Scripting Syntax

- Uses **@** for special processing of symbols and **{ }** to define blocks
- **@object.value** to read data from server objects
- **@keywords** for special processing
- Supports simple expressions

```
ProcessorGeneric.AddVar( 'obj2',  
    TSimpleObject.Create( 'marc', 9 ) );
```

<h4>Obj2</h4>

```
@if obj2.ValueBelowTen {  
    <p>@obj2.name <span> has a value of </span>  
        @obj2.value, which is under 10</p>  
}
```

Obj2

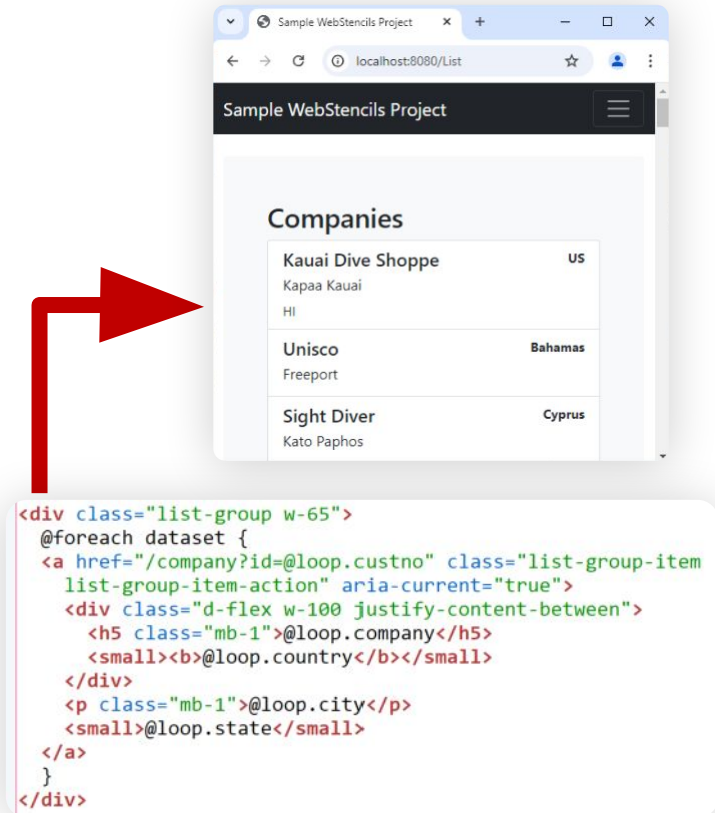
marc has a value of 9, which is below 10



WebStencils Template Library

Data access for the script

- RTTI based (but not only)
- `@object.value`
- `OnValue` event handler
- Use objects, lists, or datasets registered as **script variables**
- Define modules variables with attributes





RAD



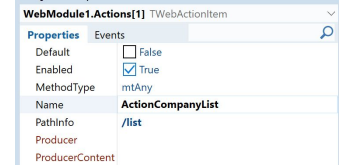
What's New in <RAD Studio 12.2>

From Simple WebBroker Scenario to Mapping

WebBroker Actions

- Connect action to WebStencilsProcessor
- Use OnAction handler to:
 - Initialize the processor script variables
 - Run the processor
 - Return the resulting HTML
- In the script, use the variables to generate the HTML

Both WebBroker and RAD Server offer a mapping layer to associate a URL with a file script



```

- procedure TWebModule1.WebModule1WebActionItem1Action( Sender : TObject;
-   Request : TWebRequest; Response : TWebResponse; var Handled : Boolean );
- begin
200 ClientDataSet1.Open;
-   ProcessorCompanyList.AddVar( 'dataset', ClientDataSet1, False ); // do not destroy
-   Response.Content := ProcessorCompanyList.Content;
-   ClientDataSet1.Close;
-   Handled := True;
- end;

@foreach dataset {

  <a href="/company?id=@loop.custno" class="list-group"
  <div class="d-flex w-100 justify-content-between">
    <h5 class="mb-1">@loop.company</h5>
    <small>@loop.country</small>
  </div>
  <p class="mb-1">@loop.city</p>
  <small>@loop.state</small>
</a>

}

```



RAD



What's New in <RAD Studio 12.2>

WebStencils Keywords

@query - access HTTP parameters

@* .. *@ - comment

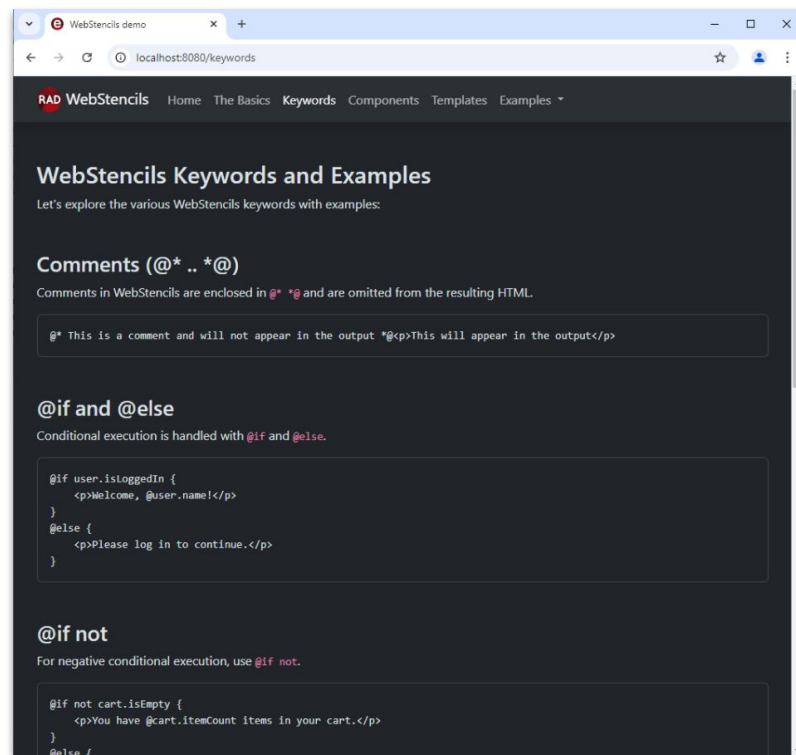
@if [not] object.value { ... } [else { ... }]

@ForEach (var object in list) { ... }

@ForEach object { ... }

@Import

@Scaffolding





RAD

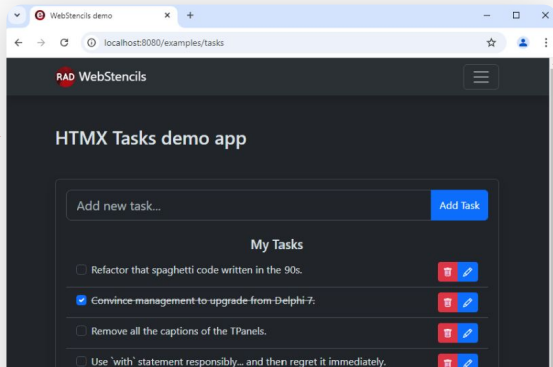


What's New in <RAD Studio 12.2>

WebStencils Template structure

```
<body data-bs-theme="dark">
  @Import layouts/navBar
  @RenderBody
  @Import layouts/footer
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@
    integrity="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLES
    crossorigin="anonymous"></script>
</body>
```

```
<li class="list-group-item list-group-item-action d-flex" id="task-@Task.Id">
  <input class="form-check-input me-2"
    hx-put="/tasks/toggleCompleted?id=@Task.Id"
    hx-target="#task-@Task.Id"
    hx-swap="outerHTML"
    type="checkbox"
    id="Checkbox-@Task.Id"
    @if ( Task.completed ) { checked }>
  <span class="task-text @if( Task.completed ) { text-decoration-line-through }"
    hx-trigger="click"
    hx-put="/tasks/toggleCompleted?id=@Task.Id"
    hx-target="#task-@Task.Id"
    hx-swap="outerHTML">@Task.Description</span>
```



@LayoutPage

Indicates which template file to use

@RenderBody

Placeholder in a template file where to place actual content

@ExtraHeader { ... }

Optional block of code to be added as extra header information

@RenderHeader

Where to add the extra header of the page



RAD



What's New in <RAD Studio 12.2>

WebStencils Template Library Components

TWebStencilsProcessor

Processes an individual file

Standard WebBroker Producer

Properties InputFilename, InputLines...

TWebStencilsEngine

Provides shared settings to individual

TWebStencilsProcessor components

Create WebStencilsProcessor comp, if needed

Properties: Dispatcher of type IWebDispatch,
RootDirectory, DefaultFileExt...

#	Company	Last Name	First Name	City
1	Realfire	Birdseye	Carmine	Pittsburgh
2	Nlounge	Andree	Josy	Bobigny
3	Browsezoom	Doreward	Korrie	Kalanchak
4	Quinu	Tomeo	Karyn	Kasli
5	Skiptube	Salleir	Alia	Añelo
6	Cogibox	Hallard	Harley	Xinfengjie
7	Layo	Maunton	Dukey	Sadabe
8	Skyba	Ridout	Fowler	Jeding
9	Pixonyx	Earney	Janessa	Maharagama
10	Trilia	Daybell	Jaclyn	Qa'en
11	Yotz	Dunlea	Ase	Lumbang
12	Gabspot	Beccera	Mariam	Simnas
13	Edgetag	Croom	Agata	Sidomukti
14	Wikibox	Tomasino	Roxy	Maoqitun
15	Yata	Feldbaum	Kirsteni	El Corozo



Improved Mobile Development

Significant improvements for Delphi mobile development

Apple platforms

- Linker updated to the most recent Apple version
- Link iOS static libraries written in Xcode, like **Firestore iOS SDK**

Android support

- Options to **compress Dex files** and **compress native library files**
- The **targetSdkVersion** of the manifest file can be configured in the IDE
- Import Android libraries ('.aar' files)
- Works with the **Facebook SDK** for Android and **Firestore Analytics**

Quality in 12.2

Visible
improvements in all
product areas





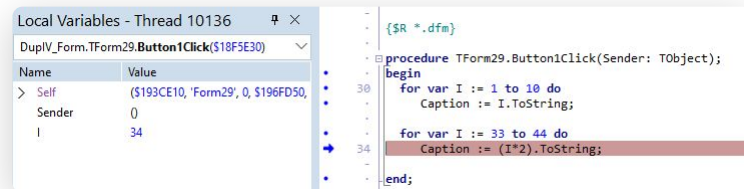
RAD



What's New in <RAD Studio 12.2>

Debugger Quality

- Several improvements in the various **RAD Studio debuggers**
- **Delphi debugger** handles repeated inline variables with the same name
- Improved the **debugger error handling** and the remote debuggers communication timeouts.





DelphiLSP Quality

The DelphiLSP engine improvements

- **Symbol Hints** Hints work on most symbols in the class interface, including fields (was not available in 12.1)
- **Local Constants and Variables** Hovering over local constants and variables can now resolve their definitions
- **Forwarded Type Declarations** The engine now resolves forwarded type declarations in some cases

```
1 unit Unit2;
2
3 interface
4
5 uses
6   Winapi.Windows, Winapi.Messages, System.SysUtils, SysUtils,
7   Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Menus;
8
9 type
10  TForm2 = class(TForm)
11    MainMenu1: TMainMenu;
12  pr var TForm2.MainMenu1: TMainMenu - Unit2.pas (11)
13  public
14    { Public declarations }
15  end;
16
17 var
18   Form2: TForm2;
19
20 implementation
21
22 {$R *.dfm}
23
24 end.
```



Delphi RTL, VCL, and FireMonkey Quality

Delphi RTL

- New **TParallelArray** class (offers faster parallel operations, like sorting)
- New **TOrderedDictionary** class
- Support for the **REG_NONE** type in **TRegistry** (key with no type)

VCL

- Primarily quality work around **frames** scaling, **FormsTabBar**, styling issues, High DPI issues. *Quite extensive work.*

FireMonkey

- Further work on Android deployment, intensive quality work for many areas, with additional focus on edit and memo controls on mobile





RAD



What's New in <RAD Studio 12.2>

RAD Server Improvements

- **WebStencils** template engine support   (*RAD Server can do Web!*)
- Reworked RAD Server Push Notification (and Firebase APNS)
 - Added required support for HTTP/2 (uses libCurl also on Win)
- RAD Server **Swagger Documentation** enhancements
 - New **IEMSApiDocumentService** interface (customization of the metadata in the generated swagger doc files)
 - New **IEMSApiDocumentFilter** interface (filter the API being documented)
 - Swagger support for YAML format in addition to JSON
 - **APIDocAttributes** demo updated to use Delphi multiline string literals



Quality Overall

Customer reports fixed

- Addressed approximately 350 customer reported issues
- Implemented 28 customer feature requests
- Reported on the old or the new portal

Addressed issues in all areas, the top ones are

- | | |
|--------------|-------------------|
| ○ IDE | ○ Compiler\C++ |
| ○ VCL | ○ RTL\Delphi |
| ○ FireMonkey | ○ Compiler\Delphi |



Why adopt
version 12?

1



DELPHI RELEASES

- Longer release cycle
 - ◆ 2 years between major releases?
 - ◆ More updates (2, 3, ...)
 - ◆ Updates pack more features, not only bugfixes
 - No dcu interface breakage rule
- A major release tends to be more stable (hopefully)
 - ◆ With more updates



DELPHI DEFINITIVE VERSION?

- It's not about choosing the **best version ever**
 - ◆ But moving your code away from older versions
- What qualify as **definitive version**?
 - ◆ IDE Stable enough
 - ◆ Right features for you
- Think of changing your platform every 2 years
 - ◆ Avoid the **ancient-version** lock-in
 - ◆ Better **some work** every 2 years than **immense work** after, let's say, 10 years



Language features

2



Delphi 2005

- for ... in loops
- inline keyword
- nested types
- nested constants
- {\$REGION}/{\$ENDREGION} directives



Delphi 2007

- Enhanced records
- Operator overloading
- Static methods and properties
- FastMM is the default memory manager
- Strict private/protected visibility keyword
- Final keyword for virtual methods



Delphi 2009

- Generics
- Anonymous methods
- Smart Pointers
- Support for nested exceptions and exception tracing
- Exit function can take a parameter specifying a result
- Deprecated keyword can now have additional text



Delphi 2010

- New Delphi RTTI (Run Time Type Information)
- Attributes
- The as and is operators can be used on interfaces
- Casting can be performed on an interface: `TObject(MyInterface)`
- New delayed directive to load external libraries at first call
- Class Constructor/Destructor



Delphi XE

- Regular Expressions Library
- TTimeZone, TBinaryReader & TBinaryWriter
- Refactoring and improvements on TThread and Sync classes
 - ◆ TThread.Suspend and TThread.Resume deprecated
 - ◆ Proper thread synchronization based on TEvent and TMutex
 - ◆ New classes for thread synchronization
 - TInterlocked, TSpinWait, TSpinLock



Delphi XE2

- FireMonkey
- Support for Win64
- LiveBindings
- Full unit scope names are now required in your uses clause
- New System.Zip unit



Delphi XE3

- Record helpers for built-in types
- Atomic intrinsic functions:
 - ◆ AtomicExchange()
 - ◆ AtomicIncrement()
 - ◆ AtomicCmpExchange()
 - ◆ AtomicDecrement()
- Introduction of the [ref] attribute



Delphi XE4

- New dependency directive
 - ◆ Specifying the dependencies of an external library



Delphi XE5

- New REST Client Library added
- System.Types
- Operator overloading for classes
 - ◆ Only for the NextGen compilers Android/iOS



Delphi XE6

- JSON framework refactored
 - ◆ System.JSON introduced
 - ◆ REST.Json introduced



Delphi XE7

- String-Like Operations Supported on Dynamic Arrays
- Parallel Library added to the RTL
- TNetEncoding



Delphi XE8

- New HTTP Client API
- New Hash API
- Asynchronous Programming Library
- New integer types
 - ◆ FixedInt, FixedUInt => 32-bit integer types on all platforms
- New platform dependent integer types
 - ◆ LongInt, LongWord => 64-bits on iOS-64, 32-bits on all other platforms



Delphi 10

- Support for Android 5.1.1 and iOS 8.4
- Improved OSX exception handling
-



Delphi 10.1

- Native support for `Utf8String` and `RawByteString` type on all platforms
- The `[weak]`, `[unsafe]` and `[volatile]` attributes are supported on all compilers
- Class and record helpers cannot access private members of the classes or records they extend



Delphi 10.2

- Support for Linux server apps (Intel 64-bit using LLVM and ARC).
- Assigning a dynamic arrays to a pointer using the @ operator is only allowed when hard-casting the array.
- More flexible namespace resolution of unit names



Delphi 10.3

- Inline Variable Declaration
- Type inference
- AnsiString / AnsiChar on Linux
- Traditional Memory Reference Counting in the Linux compiler



Delphi 10.4

- Custom managed records
- Unified memory management on all platforms - full ARC compiler has been removed and all compilers
- Support for Android, macOS 64-bit



Delphi 11

- Binary Literals and Digit Separators
- New record helpers: TDateTimeHelper and TCurrencyHelper
- Inline assembler support for AVX instructions (AVX-512)
- macOS ARM 64-bit target platform



Delphi 12

- Long and multiline string literals
- NativeInt/NativeUInt have been downgraded to weak aliases
- The Delphi32 compiler now handles NaN comparisons according to IEEE754



THANK YOU