What's New in RAD Studio 10.3 Rio

The RAD Studio 10.3 Rio release contains the following new and improved features.

Delphi Language

Inline Variable Declaration

The Delphi language in 10.3 has a fairly core change in the way it allows far more flexibility in the declaration of local variables. Until now, following classic Pascal language rules, all variable and constant declarations had to be done in a var or const block written before the beginning of a function, procedure or method code.

The new inline variable declaration syntax allows you to declare the variable or constant directly in the code and assign a value to it directly in the same statement. It is also possible to declare variables in a nested code block, with visibility and lifetime limited to that nested block.

Type inference

Variables declared inline also benefit from type inference. You do not need to specify a type for an inline variable with a direct assignment, as it can get inferred from the value assigned to it.

Traditional Memory Reference Counting in the Linux compiler

The Linux 64-bit compiler in 10.3 has been ‘reverted’ to using the non-ARC implementation of object memory management, matching exactly the Windows behavior.

The NEXTGEN define has been disabled for the Linux 64-bit compiler.

AnsiString / AnsiChar on Linux

RAD Studio 10.3 Rio has enabled support for the older-style AnsiChar/AnsiString data types on Linux. Use them with care, given Unicode is the preferred string type also on Linux and that Ansi code pages on Windows and Linux won’t match. However, this can help increase compatibility with existing low-level string management code.

C++

Clang Upgrade: C++17 support

C++Builder and RAD Studio 10.3 introduce an updated Clang-enhanced compiler with C++17 support for Windows 32-bit. Part of our progressive compiler upgrade, you can now use the latest C++ language features for more powerful and concise code, more compiler optimizations giving you faster code, and make use of more third-party C++ libraries to build more powerful applications.

- C++17 Clang-enhanced compiler and toolchain for Win32 (bcc32x and bcc32c).
- C++ runtime library (RTL) built with the updated Clang-enhanced compiler for Win32, including 2018 edition of Dinkumware STL.
- Also includes support for C++14.
- Improved code completion.

Clang is the default

In previous releases, a new C++ project defaulted to using the classic compiler for the Win32 platform. This has been changed, and the default is now the Clang-enhanced compiler.

Backwards compatibility

The Win32 compiler defaults to C++17 for all projects. However, in the Project Options > C++ Compiler page, you can choose to use an earlier language standard, including C++14, C++11, and even C++98/03.

What's new in C++17?

There are some great new features in C++17 that will enhance your productivity and your code.

For a good overview of what's new in C++17, read:

- This github document by Tony Van Eerd, which lists all features, with examples. It's a great read.
- This very detailed blog post also examines everything new in C++17.

As a result of upgrading from C++11 to C++17, RAD Studio 10.3 Rio also includes support for C++14.

- This Dr. Dobb's article has a good overview, with code, of new items in C++14.
- This github document by Anthony Calandra has info about what's new in all modern C++ versions, C++11 onwards.

New STL/Dinkumware version

C++Builder uses the Dinkumware STL to provide the C++ standard library for Windows. C++Builder 10.3 uses an updated, 2018 version of the the Dinkumware STL for both Win32 and Win64.

Better math performance for Win64

Many C++ math functions have a new implementation. On average, these methods are approximately twice as fast as in the previous release.

Better code completion for C++

Code completion for C++ Win32, using the updated Clang-enhanced compiler, is significantly faster and with better results than previous C++ code completion. It uses a Language Server Protocol server, cquery.

Code completion for this compiler is now asynchronous. Typing will not pause while completion is being calculated. It uses a Language Server Protocol server, cquery.

Error Insight for C++

Error Insight, which shows a preview of code issues inline in the editor by drawing a red underline under errors, is now available for C++. This uses the Language Server Protocol and cquery, and is available when using the new Clang-enhanced compiler for Win32.

Debugging optimized builds

In the past, turning on debug info disabled compiler optimizations. In 10.3's updated Win32 compiler, debugging an optimized build is now supported. Note that in an optimized build, it is very common for expressions and variables to be unavailable to the debugger, because they are optimized out.

New libraries in GetIt

A number of widely used C++ libraries are available through GetIt.

Delphi and C++

C++/Delphi ABI compatibility

In previous releases, there were subtle platform differences in the ABI underlying method passing, especially for records between 4 and 8 bytes in size when passed by value or by reference. These are now resolved, and issues you may have seen migrating C++ Win32 code to Win64 when interacting with Delphi, especially when using event handlers, should be resolved. The canonical example is an event handler taking a TPoint parameter: in Win32 this showed correct values for the point’s x and y coordinates, but in Win64 reading x and y gave ‘junk’ values. This no longer occurs.

Most changes affect fastcall, but RAD Studio 10.3 Rio also includes changes to cdecl and other calling convention support for full compatibility, including on mobile platforms.

IDE

RAD Studio 10.3 Rio has an improved look and feel of the main window and several key dialogs for a more modern style with an emphasis on readability and clarity.

Two new themes

RAD Studio 10.3 includes a new light theme and a revision of the dark theme, with an emphasis on usability for long periods of time.

Project and IDE Options

The IDE and Project Options dialogs have a cleaner look with controls aligned and spaced regularly. Instead of a gray background, the windows look more modern with a white background, a caption for each settings page, and a full-width selection for the options tree.

New categories have been introduced in the options tree and some options have been moved to new categories. For example, in the Project Options dialog, the application icons or images are no longer on the Application page, but in **Application > Icons**.

Both Options dialogs can be searched using a new search box in the title bar.

IDE Main Window

RAD Studio 10.3's main window has a cleaner look, it is more readable and aligned. Some changes include:

- Excess borders and lines have been removed. The UI differentiates sections based on background colors, not on border lines.
- Readability and click target enhancements. Editor tabs are larger and the font is white on a dark background, making them easier to read and click, and dockable windows have a larger title bar.
- Focus changes in the IDE are very clear. The current focused area, whether it is an editor tab or a docked window, has a strong blue background. It is now immediately clear where keyboard focus is located.
- Some windows have been renamed, including the **Project Manager** which is now **Projects**, and the **Tool Palette** which is now **Palette**.
- IDE Insight, allowing IDE-wide search, is in the IDE's title bar.

New Items Dialog

The New Items dialog window lists items in a scrollable list, with a name and description. To ease finding items, some items are now in multiple categories (for example a dynamic library is now present in Windows and Multi Device), and the search has been moved to the title bar.

Compile Dialog

The Compile dialog now makes the number of hints, warnings, and errors clear. A new layout makes it more readable.

GetIt Package Manager Dialog

The GetIt Package Manager now shows items in a scrolling list, instead of pages. It also has a cleaner look, and search has been moved to the title bar.

Additional IDE Improvements

Form Loading / LiveBindings Design Time IDE Optimizations

In RAD Studio 10.3, loading forms using LiveBindings or with many controls is now significantly faster. Load times decreased from thirty seconds (for a very large form) to a few seconds only.

Fix Pack Improvements

RAD Studio 10.3 Rio includes some of Andreas Hausladen's "IDE Fix Pack" fixes and improvements. Most of the incorporated fixes are focused around the IDE and include the following:

- High resolution timer caused by using Virtual Treeview.
- FindHInstance optimisations.
- Debugger using CREATE_DEFAULT_ERROR_MODE when creating the process.
- TStringList IndexOf and Name optimisations.

VCL Integrated Translation Architecture and Tooling

Although VCL Integrated Translation architecture and tooling are still included in 10.3, no further improvements to these tools were made and they will be removed in a future release. You should reduce the dependency on these tools and migrate to a different set of translation tools.

VCL

High DPI Image List Support

With the new VCL High DPI ImageList control in 10.3, developers building new VCL Windows applications or updating existing apps for high DPI displays can fully support multi-resolution, pixel perfect images on all controls, as well as any custom drawing requiring scaled images for multiple resolution monitors. You can achieve this by using the **TImageCollection** component in combination with the **TVirtualImageList**component.

These paired components separate the concept of a collection of images (where each logical image can have multiple resolutions) from a list of images at a single specific size used for a control.

TVirtualImageList is fully compatible with and is a drop-in replacement for traditional image lists, including providing a HIMAGELIST handle, and can be used by both VCL controls and any code using Windows API image list calls.

TImageCollection supports images with alpha channels, including PNGs. You can also load old-style color-keyed transparency bitmaps and there are migration tools to assist converting traditional TImageLists to the new image collection and virtual image list.

Per Monitor V2 support

RAD Studio 10.3 includes Per Monitor V2 support for VCL. This allows VCL applications to scale correctly for all Windows scaling, and to respond to DPI scaling changes between different screens. To enable this for your application, go to **Project > Options > Application > Manifest** and in the **DPI Awareness** section choose **Per Monitor V2**.

Win 10 Features - Expanded Store Support/APIs

10.3 increases the number of Windows APIs that VCL and FireMonkey developers can use in their applications. This includes a number of key WinRT APIs and recent Windows 10 APIs, including ready-to-use components for in-app purchases and trials in the Windows 10 Store.

WinRT API Updates

RAD Studio 10.3 includes updates to the WinRT API Object Pascal header declarations, including support for many APIs added since the first release of Windows 10. These APIs follow the same declaration model introduced in RAD Studio 10 Seattle and can be found in the following units (under the System Windows WinRT source folder):

```
WinAPI.ApplicationModel.Background.pas
Winapi.ApplicationModel.Contacts.pas
WinAPI.ApplicationModel.Core.pas
WinAPI.ApplicationModel.DataTransfer.pas
WinAPI.ApplicationModel.pas Winapi.CommonNames.pas
WinAPI.CommonTypes.pas WinAPI.DataRT.pas
Winapi.Devices.AllJoyn.pas
Winapi.Devices.Bluetooth.Advertisement.pas
WinAPI.Devices.Bluetooth.pas
WinAPI.Devices.Enumeration.pas
Winapi.Devices.Geolocation.pas Winapi.Devices.Midi.pas
WinAPI.Devices.pas Winapi.Devices.PointOfService.pas
Winapi.Devices.Scanners.pas Winapi.Devices.Sensors.pas
Winapi.Devices.Sms.pas
WinAPI.Foundation.Collections.pas WinAPI.Foundation.pas
WinAPI.Foundation.Types.pas WinAPI.Gaming.pas
WinAPI.Globalization.pas WinAPI.GraphicsRT.pas
WinAPI.Management.pas Winapi.Media.Devices.pas
Winapi.Media.MediaProperties.pas WinAPI.Media.pas
WinAPI.Networking.Connectivity.pas
WinAPI.Networking.NetworkOperators.pas
WinAPI.Networking.pas WinAPI.Networking.Proximity.pas
Winapi.Networking.PushNotifications.pas
WinAPI.Networking.Sockets.pas WinAPI.Networking.Vpn.pas
Winapi.Perception.pas WinAPI.Security.Credentials.pas
WinAPI.Security.Cryptography.pas WinAPI.Security.pas
Winapi.ServicesRT.pas Winapi.ServicesRT.Store.pas
WinAPI.Storage.pas WinAPI.Storage.Streams.pas
WinAPI.SystemRT.pas WinAPI.UI.Composition.pas
WinAPI.UI.Core.pas WinAPI.UI.Input.Inking.pas
WinAPI.UI.Input.pas WinAPI.UI.Notifications.pas
WinAPI.UI.pas WinAPI.UI.Text.pas
WinAPI.UI.ViewManagement.pas WinAPI.UI.WebUI.pas
WinAPI.UI.Xaml.pas WinAPI.WebRT.pas
```

Windows API Updates

This Windows API update includes different areas. One is for new High-DPI related APIs, including DPI process and thread awareness, DPI settings for monitors and windows, high DPI themes, and more.

Another area relates with support for the new “pen input” and WM_POINTER_xxx messages and related APIs and data structures.

10.3 also has rearranged some duplicated core type declarations. For example, PUint32 is not in System.pas. Also, some Windows handle types have been moved and have a slightly different declaration.

TWindowsStore

TWindowsStore component allows you to connect a Windows application with the Windows Store and use features such as getting a list of apps owned by the user, available add-ons, purchased add-ons, and handle trial mode.

The TWindowsStore component is supported for Windows platforms only and RAD Studio 10.3 Rio includes a VCL version and a FireMonkey version.

TWindowsStore is a component wrapper for the TWindowsStoreCore.

Additional VCL Improvements

- Support for colored fonts in TDirect2DCanvas.
- Exposing of wm_pointer related messages and APIs.
- Sharing Contract extensions.

FireMonkey

Android API level 26 or higher

RAD Studio 10.3 Rio includes support for Android API version 26 development, as required by Google for new Play Store applications starting August 2018 and for updates starting November 2018. Changes include:

- New information in the manifest (with the proper API level).
- The default configuration uses a recent version of the SDK/NDK.

You must update the SDK to include a recent platform library, otherwise your Android apps won't build.

New Android Permission Model

Recent versions of the Android API have changed the mechanism to request permissions. The user can call PermissionsService.RequestPermissions (from the new System.Permissions unit) and pass it a reference to a routine that will be called with the users responses, and optionally a routine to display a rationale for the requested permission. See Android Permission Model for more information.

Android Z-Order

Android Z-Order in 10.3 provides support for using FireMonkey styled controls such as buttons, labels, and checkboxes with natively rendered controls like the browser and map control on the same form without the native control covering the styled control.

Android native controls

10.3 includes a number of new capabilities designed to support the use of native controls together with styled FireMonkey controls on the same Android form.

One of the changes is the use of the material design theme for Android native controls in FireMonkey applications. Native controls such as TWebBrowser, TMapView, and other new native controls will use material design theme on devices running Android version 5.0 or later.

The following list contains all the controls that currently support native presentation:

- TCalendar
- TEdit
- TSwitch

iOS 12 support

RAD Studio 10.3 includes iOS 12 support for targeting devices and building App Store and Enterprise applications.

Mojave support

RAD Studio 10.3 includes building 32-bit applications that run on macOS 10.14 Mojave. Delphi supports targeting the 10.14 SDK.

Additional FireMonkey Improvements

- Support for Unicode Emoji.

FireDAC and Database

Improvements for MySQL

- Support for MySQL v 8.0.
- Support for MariaDB 10.3.

Improvements for SQL Server

- Support for SQL Server 2017.

Improvements for PostgreSQL

- Support for PostgreSQL v 10, including new:
  - Identity columns.
  - Macaddr8.
  - Password encryption.
- New GUIDEndian=Little|Big connection parameters.

Improvements for Firebird

- Support for Firebird v 3.0, including new:
  - local connection protocol.
  - FB$OUT package.
  - isc_database_info() information items.
  - Statistics feature in gbak output.
  - Support for statement length > 64Kb.
- New TFDFBOnlineValidate component.
- New GUIDEndian=Little|Big connection parameters.

Improvements for MongoDB

- New TimeZone=Local|UTC connection parameter.

Improvements for InterBase

- Support for InterBase v 2017, including new:
  - TRUNCATE command.
  - Support for TRUNCATE in Change Views.
  - Transaction wait tim.e

- New GUIDEndian=Little|Big connection parameters.

Improvements for SQLite

- Support for 3.23.1.
- Improved support for Linux platform, including new:
  - Encryption support.
  - Collation support.

Improvements for SQL Anywhere

- TFDPhysASADriverLink.ToolHome new property.
- Added support for TFDEventAlerter.

Other FireDAC Changes

- Added TFDBatchMoveJSONWriter.
- Optimized TFDBatchMove and related components.

Database Improvements

- New TClientDataSet.IncludeBlobsInDelta property.

Improvements for DataSnap

- New TDSRestConnection.SecureProtocols property.
- DataSnap REST now explicitly specifies "Content-Type=application/json".

Improvements for REST

- Improved awareness of different MIME types.
- Added TRESTClient.OnNeedClientCert property.
- Added TRESTClient.OnAuthEvent property.
- Added TRESTClient.RedirectsWithGET property.
- Added TRESTClient SecurityProtocols property.
- Indy replaced with THTTPClient.
- Added TRESTRequestParameterOption.poFlatArray, poPHPArray, poListArray options.
- Added TRESTRequestParameter.AddValue method.
- Added TRESTRequestParameter.SetStream method.
- Added TRESTRequestParameter.Stream property.

- Added TRESTRequestParameter.StreamOwner property.
- Added TRESTRequestParameter.Bytes property.
- Added TRESTRequestParameterList.AddItem.
- Modified TRESTRequestParameterKind.pkGETorPOST behavior.
- Added TRESTRequestParameterKind.pkQUERY value.
- Added TRESTRequestParameterList.AddBody methods to replace TBodyParams class.
- Added AOwnsObject: TRESTObjectOwnership parameter for TCustomRESTRequest.TBody.Add methods.
- Added AContentType: TRESTContentType argument for TCustomRESTRequest.DoPrepareQueryString.
- Added AContentType: TRESTContentType, ABodyStreamOwner: Boolean arguments for TCustomRESTRequest.DoPrepareRequestBody.
- Optimized THTTPClient on Windows platform.
- Added TMultipartFormData.AddStream and AddBytes.
- Added TNetHTTPClient.Put / THTTPClient.Put overload methods for TMultipartFormData, FileName, TStrings.
- Added TAcceptValueList class to handle values considering HTTP Accept-Xxxx headers.
- Added TMimeType class to handle MIME types.

Improvements for Cloud

- Improved support for Azure storage emulator.
- Updated AWS S3 support to cover recently added regions.
- Added TAmazonConnectionInfo.Region property.
- Added "BucketRegion: TAmazonRegion = amzrNotSpecified" parameter for most of bucket / object related TAmazonStorageService methods.
- Improved general performance of Cloud support.

RAD Server

Performance enhancements

RAD Studio 10.3 includes the following performance enhancements for RAD Server:

New EndPoint Attributes for Content-Type and Accept based mapping

To support better resource mapping, which doesn't depend only on the URL but also on Accept and Content-Type HTTP request headers. This means you can have two distinct methods mapped to the same URL and HTTP verb, but still returning different types of data depending on the request.

HTTP Verb to Custom Method Name Mapping

In past versions of RAD Server, the system would generate custom mappings for HTTP verbs (GET, POST, etc) to method names. This remains the default, but you can also map an HTTP verb to a method with a custom name, using the new EndpointMethod attribute.

Ability to Delegate Processing of a Request to a Custom Class or Component

Add ability to RAD Server custom resources API to delegate request processing to resource module Delphi fields, which are custom endpoint publisher classes / components.

Helper Components for JSON Processing

Taking advantage of the new ability to delegate processing to a component, RAD Studio 10.3 introduces new components for simplifying the JSON processing work, particularly when database queries are involved. The components can be added to a class mapped to a RAD Server resource (or a data module), and the HTTP methods can be mapped to them without writing any code.

Additional RAD Server Enhancements

- RAD Server performance was significantly improved with ten-fold increase in throughput for simple operations (part of the related fixes were already delivered as a patch for 10.2.3)

RTL

The Delphi RTL in 10.3 has significant performance improvements and enhanced standards compatibility for JSON and HTTP.

Data Structures Growth Strategy

Several data structures (TStringList, TList, TList <T>, TQueue<T>, TStack<T>) have now a flexible growth strategy when they are full and need to be expanded, compared to the x2 strategy of the past. The growth strategy can be replaced. The new growth strategy is implemented in a shared global function, declared in SysUtils.pas:

```
function GrowCollection(OldCapacity, NewCount:
Integer): Integer;
```

Additionally, you can customize the implementation by writing a new compatible function and calling the global SetGrowCollectionFunc procedure.

If you install a custom “grow collection function” in a runtime package, remember to set it to `nil` upon exit, or the runtime might try to call the non-existent function after the packages have been unloaded.

TStringBuilder Changes

The class has had several changes with the goal of improving its performance, including a similar change in memory growth strategy, the removal of some redundant code, and an overall implementation cleanup.

The TStringBuilder enumerator has been optimized.

There is also an additional parameter for the TStringBuilder.ToString method. The signature is ToString (UpdateCapacity: Boolean). ToString(True) will give better performance if no more modifications are expected for TStringBuilder, as it reduces the amount of data being copied.

JSON Improvements

Significant JSON processing and parsing improvements for correctness and performance.

Lists-Related Improvements

- TList and TDictionary have new public properties to make their comparers (the definition of their comparison operations for sorting) accessible after initialization.
- Added TryAdd method to TDictionary<TKey,TValue>.
- Added ExtractAt(Index: Integer): T to TObjectList<T>.

- Improved TList<T>.IndexOf performance.
- Improved general TList<T>, TQueue<T>, TStack<T> performance for adding items to the lists. Standard TList<T>.Add should be ~30% faster.
- Optimized TList, TStrings, TComponent, TCollection, TList<> enumerators. Now empty "for in" loop is 2.5 to 4 times faster.
- As part of the collection enumerator types, RAD Studio formally defines the enumerator status when the iteration has completed: “The enumerator state is not valid after MoveNext returned False and enumerator must be released or recreated and should not be accessed any further”.

Additional RTL changes

- TStringHelper.Split now produces the same results of the global SplitString function from StrUtils.
- zlib upgraded to 1.2.8 with additional fixes (and now compiled with RAD Studio C++ compiler for 64-bit).
- PCRE upgraded to 8.42 and includes UTF-16 support on Windows (and now compiled with RAD Studio C++ compiler).
- Unicode table (System.Character unit) supports Unicode v11.0.
- The TStringHelper.Split method has different optional behavior based on the new TStringSplitOptions.ExcludeLastEmpty option.
- Reorganization of the order of the procedures and functions SysUtils unit to better support inlining.
- Float32 and Float64 aliases added in System unit.

Sending Long Strings with AppTethering

The current implementation of SendString / AsString operations in the TTetheringAppProfile class (used for AppTethering) are limited to a length of roughly 1,400 characters. RAD Studio 10.3 includes the following changes and additions, allowing for long strings while maintaining compatibility:

- New TTetheringAppProfile SetLongString and SendLongString method use a stream rather than a string for the underlying communication.
- TResourceValue.AsString can now return a string value of received stream, if it is available. If not then returns value of received string. This was it works both for a regular string and a “long string”.

It is recommended you use TTetheringAppProfile.SendString only if length of string is less than ~1.3Kb, otherwise TTetheringAppProfile.SendLongString must be used.

TMemIniFile Optimization

RAD Studio 10.3 Rio has optimized the TMemIniFile implementation. Reading and constructing a TMemIniFile is 10 to 25 times faster and consumes half of the memory. Other TMemIniFile operations are improved too and they are 50 to 100% faster compared to the previous implementation.

Also, 10.3 includes the ability to load a TMemIniFile from a stream, with two additional overloaded constructors:

- TMemIniFile.Create(Stream).
- TMemIniFile.Create(Stream, UseLocale).

These constructors parameters remain available in the class and are exposed in new properties:

- property Stream
- property UseLocale