Release Notes

Go Up to What's New

These Release Notes contain important information that might not appear in the main product documentation. We recommend that you read this page in its entirety.

Instal Unins and Upgra RAD Studio

Contents

Before you install, uninstall, or upgrade the product, read the following files, for your chosen language (`xx` can be en, de, fr or ja):

- `radstudio_install_xx.htm` : lists the system requirements, as well as installation and upgrade procedures.
- `radstudio_license_xx.rtf` : contains your Software License and Support agreement.
- `radstudio_deploy_xx.htm` : contains information about deployment.

Read the latest version of the `radstudio_install_xx.htm` file, available at the following locations:

- **Installation Notes (http://docwiki.embarcadero.com/RADStudio/Rio/en/Installation_Notes)**.
- The installation directory. The default location is: `C:\Program Files (x86)\Embarcadero\Studio\20.0`

Upgrading process may generate errors with GetIt packages. As workarounds you can manually uninstall the plugins before upgrade process, remove the matching entries in the registry under the CatalogRepository section, or install them again from GetIt if you see an error message when the IDE starts.

Previous Versions and Migration

RAD Studio Includes Licenses for Previous Versions

Licenses for previous versions of the product are included with your 10.3 Rio license.

- Delphi 10.3 Rio includes licenses for Delphi Tokyo, Berlin, Seattle, XE8, XE7, XE6, XE5, XE4, XE3, XE2, XE, 2010, 2009, 2007, and 7.
- C++Builder 10.3 Rio includes licenses for C++Builder Tokyo, Berlin, Seattle, XE8, XE7, XE6, XE5, XE4, XE3, XE2, XE, 2010, 2009, 2007, and 6.

For more information about licenses, see the `radstudio_license_xx.rtf` file in the RAD Studio installation directory.

Upgrading Projects When Opening

When using RAD Studio 10.3 Rio to open a project that was created with a previous version of RAD Studio, RAD Studio automatically upgrades the project. Upgraded projects are no longer compatible with previous versions of RAD Studio.

Always create a backup of a project before you open it with a newer version of RAD Studio.

Upgrading from Delphi 7

If you are upgrading a project that was created in Delphi 7 or another early release, and the project has a custom file extension, you need to choose Project > Options > Application, and set the **Target file extension** to match the type of your project (.exe, .dll, or .bpl).

GetIt Installer Issues on Single Core Win 7 VMs

If you are installing Delphi, C++Builder, or RAD Studio 10.3 on a single core virtual machine, running Windows 7, and are encountering an operation error during the web based GetIt installation, increase to multiple cores and restart the installation. If you have a Firewall blocking .7z files, GetIt installation will fail.

Windows

DLLs are exporting RTL methods

When linking a 64-bit application to a DLL's import library in the Project Manager, the library is placed on the linker's command line, connecting the linker to the `__CPPdebugHook` symbol in the DLL's import library, rather than its own internal RTL symbol. To prevent this from happening, link to the import library using the `#pragma link` or `#pragma comment` and remove the reference to the import library in the Project Manager. This forces the linker to look at the symbols in the import library at the end, hence linking to the correct `__CPPdebugHook` symbol in it's own RTL.

Issues when rebuilding an MSBuild for a second time

When rebuilding an MSBuild with or without the additional resource file for a second time, the check on whether to `build` the project version resource file or not happens before the `clean` step of rebuild.

To prevent this from happening, perform a `clean` followed by a `build` :

```
> msbuild /t:clean project.cbproj

> msbuild /t:build project.cbproj
```

Linux

Apache version requirements

Delphi for Linux supports Apache 2.4. Version 2.2 is not supported.

Wrong PAServer Error Message shown on Ubuntu Server 16.04 LTS

Enabling **Use launcher application** in the Run|Parameters dialog uses xterm as the default launcher application for Linux targets. When running PAServer on Ubuntu Server 16.04 LS, an error message is shown indicating that a file cannot be created. The error message is incorrect since it cannot launch xterm on Ubuntu Server 16.04 LTS which is expected. Xterm can be installed separately and is part of the client distributions of Ubuntu.

FireMonkey

Names Must Be Assigned to All Components Before Creating a New View in the Form Designer

If you open an older FireMonkey app in 10.3 Rio, you might encounter the following error message:

```
 Cannot inherit from form 'FormName".  It contains a component with a blank name
property.
```

The Views Inheritance system requires that you assign a name to all components before you can create a new View in the Form Designer.

iOS App Might Not Close/Reopen Properly After Running without Debugging on the Device

If you run your iOS app on the device without debugging and then kill the process of your application on the device, you will need to run the PAServer reset command if you want to run the process outside the IDE; type 'r' at the PAServer command prompt.

Debugging on an iPhone XS

Debugging an application on an iPhone XS is currently not supported. Deployment to, and running on these devices are both functional.

macOS Mojave Development

Starting an application with debugging on Mojave is currently not supported. As a workaround, you can attach to a running application once it has started.

To achieve this, follow the steps below:

1. Choose **Run > Run Without Debugging**, or click the **Run Without Debugging** toolbar button. This launches the application.
2. Choose **Run > Attach to Process…**
3. In the **Attach to Process** dialog:

    - Ensure you have chosen **Remote** on the left hand side.

- Select **Embarcadero macOS 32-bit Debugger** in the Debugger combo box.
- Click the '...' button for the **Remote Host** field and select the connection profile for the macOS 32-bit platform.
- Select the application that you want to debug from the list of running processes.

The macOS 10.14 SDK no longer supports 32-bit applications. You can still build apps with the 10.14 SDK with Delphi, because Delphi resolves symbols without relying on the SDK. However, you cannot do so with C++Builder and you will need to use the 10.13 SDK for C++Builder applications.

If you want to develop macOS 32-bit applications using the macOS 10.13 SDK on a Mac running Mojave, you need to install XCode 9.x. We recommend installing both Xcode 9 and Xcode 10. You can switch between them using xcode-select (https://medium.com/@hacknicity/working-with-multiple-versions-of-xcode-e331c01aa6bc):

```
sudo xcode-select -s <xcode 9.x app>/Contents/Developer/
```

> ***Note:*** *This is only an issue when debugging Delphi or C++Builder 32-bit macOS applications, not when debugging Delphi 64-bit macOS apps with 10.3.2.*

macOS 64-bit

TMediaPlayer and macOS App Store

macOS 64-bit supports using TMediaPlayer in FireMonkey macOS applications. However, submitting applications to the macOS App Store using the TMediaPlayer component will result in a failure during the submission process since the component is based on an older, deprecated API which is no longer supported in the macOS App Store.

Inspecting / evaluating structures on macOS 64-bit

Structures passed as parameters by value into a method on macOS64 may display the wrong values in the debugger.

Expanding changed inspected values on macOS 64-bit

A variable which can be expanded when inspected to see the values of fields inside it will not show updated sub-values in the debugger UI after the value of the variable has changed.

Evaluation on macOS 64-bit is case-sensitive

For technical reasons, evaluation on macOS 64-bit (such as expressions in the Evaluate/Modify dialog) is case-sensitive. This is the same as iOS 64-bit. You may also need some C++-style syntax in some situations, such as using a `0x` prefix for hex values instead of `$`.

macOS 64-bit license update

When installing the macOS 64-bit platform for 10.3.2 Delphi or 10.3.2 RAD Studio (for the Delphi personality) on a system with an existing Delphi or RAD Studio license key you must follow the steps below according to your license type.

Workstation license owners:

1. In **RAD Studio IDE** select **Help > License Manager...**
2. From the **License Actions** panel, select **Register...**
3. Enter the existing serial number, login and password information and click the **Register button**
4. Once the step is completed, close the **License Manager**, restart the **IDE** and now the macOS 64-bit platform support should work as expected

Network license owners:

1. Ask your **license server (ELC) administrator** to re-host the existing license certificate and distribute the client license
2. From the RAD Studio/Delphi **Help** menu, select **License Manager...**
3. From the **License Actions** panel, select **Import...**
4. Point to the new client license and click the **Import** button
5. Restart RAD Studio/Delphi and now the macOS 64-bit platform support should work as expected

If you opened an existing Delphi 10.3 FireMonkey project using version 10.3.2 before updating the license, you need to manually edit the project's .dproj file in a text editor to add support for macOS 64-bit.This way the IDE is triggered to upgrade the project to support 64-bit macOS projects automatically.

1. First, shutdown the **RAD Studio 10.3.2 IDE**,
2. Locate the project's `.dproj` file in your project folder using MS Windows File Explorer.
3. Next, to save a back-up copy of the `.dproj` file, right-click the filename in the Explorer and select **Copy**.
4. Right-click again and select **Paste** to create the back-up copy of the `.dproj` file.
5. Next, right-click the file name in the Explorer and select **Open with...**. From the resulting dialog, select **Notepad**.
6. From Notepad, find the line that reads, **<ProjectVersion>18.7</ProjectVersion>** which is located on the fourth line from the top of the file.
7. Modify this line so it reads, **<ProjectVersion>18.6</ProjectVersion>**.
8. Now save the `.dproj` from Notepad, overwriting the original file.
9. Relaunch the RAD Studio IDE and open the project. macOS 64-bit should now be available under Target Platforms for your project.

FireDAC

Using Run-Time Packages with C++ FireDAC DataSnap Driver

If you use the **FireDAC DataSnap** driver in 64-bit C++ applications built with run-time packages, you might receive an error at run time. To avoid the error, you need to add DataSnapClient to the run-time packages, following these steps:

1. **Project > Options > Packages > Runtime Packages**.
2. In the **Runtime package import libraries** field, add **DataSnapClient** to the list.

This setting resolves the problem and adds **DataSnapClient.bpi** to the link line rather than **DataSnapClient.a**.

64-bit Windows C++ Application with a FireDAC Component Might Raise Error

The following error is raised when attempting to compile an application using a single FireDAC component, when the platform is Windows x64 with static linking:

```
c:\program files (x86)\embarcadero\studio\<n.n>\Bin\CodeGear.Cpp.Targets : warning :
Warning: Out of memory
c:\program files (x86)\embarcadero\studio\<n.n>\Bin\CodeGear.Cpp.Targets(2751,5): error
MSB6006: "ilink32" exited with code 2.
```

The workaround is not to link FireDAC statically in Win64 C++ apps.

DBX Enterprise Disabled in Database Explorer

Due to an issue, loading DBExpress Enterprise database drivers in Database Explorer is disabled by default. Database Explorer currently lists all of the FireDAC drivers you have and the DBExpress drivers in the professional edition. A fix will be provided in an upcoming release.

The DBExpress components remain available in the IDE and applications using DBExpress work as expected, as the issue is limited to the Database Explorer window.

> ***Note:*** *No longer applicable as of the 10.3.1 release.*

Database

Repopulating datasets

Repopulating a bound `TClientDataSet` or `TFDMemTable` may be slower than the initial time. To work around this issue, we have added a new `EmptyDataLink` method to `TBindSourceDB` .

Android

JDK Path Issues

Due to a bug (https://code.google.com/p/android/issues/detail?id=147561) of the Java Development Kit (JDK), the deployment path of a project must not contain non-ASCII characters for the Android target platform.

Android SDK/NDK Installation Steps

The Android SDK/NDK installation is not complete. You can either open **SDK Manager.exe** and verify that the correct download options are selected or (more reliably) use a command line silent install:

- Open command prompt and go to:

```
C:\Users\Public\Documents\Embarcadero\Studio\20.0\CatalogRepository\AndroidSDK-
2525_20.0.32429.4364\tools
```

- Issue Command:

```
android.bat update sdk --no-ui --all --filter platform-tools,build-tools-
28.0.2,android-26
```

To verify these steps have succeeded, open the **SDK Manager** in RAD Studio (Tools > Options > Deployment > SDK Manager), select **Android SDK 25.2.5 32-bit** from the **SDK Versions** list box and click the **SDK** tab in the **Properties** page. The SDK links should be correct and will have an exclamation point icon if the files they refer to are still missing.

Additional Android SDK Configuration Steps for C++Builder

The automatically configured SDK lacks a setting that prevents an error occurring in the string.h header file. To resolve this, open the RAD Studio IDE and perform the following steps:

1. Go to **File > Close All**
2. Go to **Tools > Options > Deployment > SDK Manager**.
3. Select **Android SDK 25.2.5 32-bit** from the **SDK Versions** list box and click **Delete**.
4. In the same window, click **Add...**
5. In the **Add a New SDK** dialog box, choose **Android** from the **Select a platform** dropdown list and **Add New...** from the **Select an SDK version** dropdown list. This action opens the **Create a new Android SDK** dialog.
6. Locate **Android SDK Base path** and **Android NDK Base path** from the following root directories, depending on your installation:

   - **Web / GetIt installation:** `c:\Users\Public\Documents\Embarcadero\Studio\20.0\CatalogRepository`
   - **Local/ISO installation:** `c:\Users\Public\Documents\Embarcadero\Studio\20.0\PlatformSDKs`

C++ Toolchains

Resolving Multiple Declarations for DWORD or ULONG in macOS Applications

In a multi-device application compiled for the macOS target platform, E2238 can occur because the macOS headers `iodbcunix.h` and `sqltypes.h` declare `typedef` statements for `DWORD` and `ULONG`, and the C++Builder `sysmac.h` file, which is included by `System.hpp`, also provides similar `typedef` statements. See E2238 for a workaround.

Specifying Dependent and Required Packages When Using a Component in a C++ Package

When using an existing component in a C++ package, the IDE does not automatically add the dependent (required) package for the component. You need to perform this step manually.

Without the relevant dependent (required) packages, the C++ package might fail to link because the linker cannot locate .obj files that must be initialized when that component is used. To avoid such linker errors, you need to add the package of the component and its dependencies to the **Requires** node of your C++ package (in the Projects Window).

The following example illustrates how to determine the required packages for a component and eliminate the **Unable to open 'xxx.obj** linker errors by creating a dummy VCL Forms Application that uses the component.

By default, a newly-created C++ Multi-Device Application links with the .rtl and .fmx packages, as shown by the *.bpi files listed on the linker line (viewable in **View > Messages**):

Ilink32 command line:

```
c:\BDSLoc\bin\ilink32.exe ... -aa -V5.0 -Tpe  c0wfmx32w rtl.bpi fmx.bpi memmgr.lib
sysinit.obj .\Win32\Debug\Project30.obj .\Win32\Debug\Unit26.obj ,
.\Win32\Debug\Project30.exe , .\Win32\Debug\Project30.map , import32.lib cp32mti.lib ,
, Project30.res
```

When you add new components, the IDE calculates dependent packages and adjusts the link line. To see this, add the component you want to use in a C++ Package to the Multi-Device Form. This example uses the component TFDSchemaAdapter. Now rebuild the application. You will see that several new .bpi files have been added to the link line:

Ilink32 command line:

```
c:\BDSLoc\bin\ilink32.exe ... -aa -V5.0 -Tpe  c0w32w rtl.bpi fmx.bpi
FireDACCommonDriver.bpi FireDACCommon.bpi xmlrtl.bpi dbrtl.bpi FireDAC.bpi
FireDACSqliteDriver.bpi memmgr.lib sysinit.obj .\Win32\Debug\Project30.obj
.\Win32\Debug\Unit26.obj , .\Win32\Debug\Project30.exe , .\Win32\Debug\Project30.map ,
import32.lib cp32mti.lib , , Project30.res
```

These new .bpi files are the packages on which your component depends. In the case of TFDSchemaAdapter, the .bpi files are:

```
FireDACCommonDriver.bpi FireDACCommon.bpi xmlrtl.bpi dbrtl.bpi FireDAC.bpi
FireDACSqliteDriver.bpi
```

So to use the **TFDSchemaAdapter** component in a C++ Package, you need to explicitly add each one of these bpi files as 'Required' references in the IDE (see Packages (Delphi)).

Downloading DirectX Header Files

We only ship DirectX headers inside the Microsoft Windows Platform SDK. If you encounter errors such as "**d3d.h file not found**", the HPP generated for that Delphi unit contains `#include <D3D*.hpp>` . You can do either of the following:

- Add a `{$NOINCLUDE Winapi.D3DX9}` directive to the unit that uses the D3D unit and regenerate the HPP file of that unit. The regenerated HPP will not contain the `#include <Winapi.D3DX9.hpp>` .
- Download the DirectX SDK that the D3D header relies upon. You can download the DirectX SDK for free from Microsoft. For more information, see Where is the DirectX SDK? (Windows) (http://msdn.microsoft.com/en-us/library/windows/desktop/ee663275(v=vs.85).aspx).

C++ code completion

Code completion for the Clang-based compilers was significantly overhauled in 10.3.2, but there are some known cases where it can fail. The most common one is where the 'include what you use' principle is not implemented: that is, a .cpp file or its header (or other headers included in files either includes) do not contain all includes required to build. Often this is because of an injected precompiled header which does include the required files. The language server regards each .cpp file as a standalone file and processes it without the precompiled header. A good guide is to check **Error Insight** and resolve the errors it displays.

Tooltip Insight and Go To Definition, which use indexing, are not affected.

IDE

In the Multi-Device Form Designer You Can Only Delete a Component from the Master View

When you try to delete a component, such as a TButton, from a different view (not the **Master** view), the following error message appears:

```
 Selection contains a component, Button1, introduced in an ancestor and cannot be
deleted
```

The solution is to delete the component from the **Master** view. For more information about views, see Form Designer.

Renamed iOS Project Might Fail at Run Time or When Debugging

Renaming a project in the IDE can cause an iOS app to fail at run time and debug time. To correct the issue, do the following:

1. Select **Project > Deployment**.
2. In the Deployment Manager, click the **Revert To Default** speed button.

Version Info Settings Inheritance

To include version information in a project targeting Windows, do not change the key values on the **All Platforms** target. Instead, change the values individually for each platform (32-bit and 64-bit). The targets do not inherit the values properly from the **All platforms** target.

InterBase

InterBase 2017 Edition in RAD Studio 10.3 Rio

This note pertains to users who have both an earlier version of RAD Studio and RAD Studio 10.3 Rio installed on the same machine. In this note, the mention of RAD Studio also implicitly includes Delphi and C++Builder editions.

Each version of RAD Studio includes a license for a version of InterBase:

Product	InterBase Version
RAD Studio XE3	InterBase XE3 Developer Edition
RAD Studio XE4	InterBase XE3 Developer Edition
RAD Studio XE5	InterBase XE3 Developer Edition
RAD Studio XE6	InterBase XE3 Developer Edition
RAD Studio XE7	InterBase XE3 Developer Edition
RAD Studio XE8	InterBase XE7
RAD Studio XE9	InterBase XE7
RAD Studio Berlin	InterBase XE7
RAD Studio Tokyo	InterBase XE7
RAD Studio Rio	InterBase 2017

Since all these RAD Studio license suites are visible system-wide, only one license of InterBase can be used at any time.

For example, if you are running the "InterBase XE3 Developer Edition" that comes with RAD Studio XE6, you cannot start a simultaneous instance of the "InterBase XE3 Developer Edition" that comes with RAD Studio XE7. You cannot start a simultaneous instance of the InterBase 2017 edition that comes with RAD Studio 10.3 Rio either. When you try, you receive an error dialog that states "InterBase licensing error", and the InterBase log shows "Registration file error: License is in use by another instance of InterBase".

For more information about running multiple instances of InterBase, see **Multiple Instances** section in http://docs.embarcadero.com/products/interbase/IBXE7/OpGuide.pdf.

Workaround for InterBase Licensing Errors

1. Stop all your instances of InterBase. Also, if you have set up this instance as a Windows Service, please disable it in the system Service Control Panel.
2. Start the instance of InterBase that you want to use. It should now launch successfully with the proper license.

The aforementioned RAD Studio versions and the applications built by them, can also work with the updated InterBase 2017 edition installed with RAD Studio 10.3 Rio. Have your older IDE tools and applications connect to your database via TCP loopback to this InterBase instance. For example:

```
localhost/gds_db:<dbpath>
```

In older versions of RAD Studio, you might also want to select Tools > Options > Environment Options > Environment Variables and add the following new "User overrides" entries for making local client connections.

Variable	Value
IB_Protocol	gds_db
InterBase	C:\Program Files (x86)\Embarcadero\Studio\20.0\InterBase2017

Building IBLite applications for Android requires updating the NDK path

To be able to build IBLite and IBToGo applications for Android, change the NDK API location to version 26 (19 and later can work as well.) To do this, go to Tools > Options > Deployment > SDK Manager, select **Android SDK 25.2.5 32-bit** from the **SDK Versions** list box, click the **NDK** tab in the **Properties** page, and modify the following paths:

NDK API location:

```
C:\Users\Public\Documents\Embarcadero\Studio\20.0\CatalogRepository\AndroidNDK-
17b_20.0.32429.4364\platforms\android-19
```

C++ Builder NDK Library path:

```
C:\Users\Public\Documents\Embarcadero\Studio\20.0\CatalogRepository\AndroidNDK-
17b_20.0.32429.4364\platforms\android-19\arch-arm\usr\lib
```

Delphi NDK Library paths:

```
C:\Users\Public\Documents\Embarcadero\Studio\20.0\CatalogRepository\AndroidNDK-
17b_20.0.32429.4364\platforms\android-19\arch-arm\usr\lib
C:\Users\Public\Documents\Embarcadero\Studio\20.0\CatalogRepository\AndroidNDK-
17b_20.0.32429.4364\sources\cxx-stl\gnu-libstdc++\4.9\libs\armeabi-v7a
```

> ***Note:*** *No longer applicable as of the 10.3.1 release.*

External Software

Issues might occur for applications debugged in an environment that includes a third-party keyboard switcher such as the **Yandex Punto Switcher**. If the application is closed by Windows, and an access violation is raised by the application, please ignore this access violation error.

Translation Tools

RAD Studio 10.3 Rio officially deprecates and removes from active support the translation tools available in the product, both for VCL (Integrated Translation Editor) and for FireMonkey (TLang). Both features are still available in Rio but it is recommended to reduce your project reliance on them. RAD Studio is teaming with third-party vendors to offer alternative options.

See Also

- Installation Notes
- New features and fixed issues

Retrieved from "http://docwiki.embarcadero.com/RADStudio/Rio/e/index.php?title=Release_Notes&oldid=270174"

This page was last edited on 19 September 2019, at 11:07.