RTI Launcher

User's Manual

Version 7.7.0

© 2015-2026 Real-Time Innovations, Inc.
All rights reserved.
June 2026.

Trademarks

RTI, Real-Time Innovations, Connext, Connext Drive, NDDS, the RTI logo, 1RTI and the phrase, "Your Systems. Working as one." are registered trademarks, trademarks or service marks of Real-Time Innovations, Inc. All other trademarks belong to their respective owners.

Copy and Use Restrictions

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form (including electronic, mechanical, photocopy, and facsimile) without the prior written permission of Real-Time Innovations, Inc. The software described in this document is furnished solely under and subject to RTI's standard terms and conditions available at https://www.rti.com/terms and in accordance with your License Acknowledgement Certificate (LAC) and Maintenance and Support Certificate (MSC), except to the extent otherwise agreed to in writing by RTI.

Third-Party Software

RTI software may contain independent, third-party software or code that are subject to third-party license terms and conditions, including open source license terms and conditions. Copies of applicable third-party licenses and notices are located at community.rti.com/documentation. IT IS YOUR RESPONSIBILITY TO ENSURE THAT YOUR USE OF THIRD-PARTY SOFTWARE COMPLIES WITH THE CORRESPONDING THIRD-PARTY LICENSE TERMS AND CONDITIONS.

Notices

Deprecations and Removals

Deprecated means that the item is still supported in the release, but will be removed in a future release. *Removed* means that the item is discontinued or no longer supported.

Any deprecations or removals noted in RTI's documentation serve as notice under the Real-Time Innovations, Inc., Maintenance Policy #4220 and/or any other agreements by and between RTI and customer regarding maintenance and support of RTI's software. RTI's current standard terms and support and maintenance policies are available at https://www.rti.com/terms.

Technical Support

Real-Time Innovations, Inc.
232 E. Java Drive
Sunnyvale, CA 94089

Phone: (408) 990-7444
Email: support@rti.com
Website: https://support.rti.com/

Contents

Chapter 1 Introduction

1.1 Paths Mentioned in Documentation 1

Chapter 2 Using RTI Launcher

2.1 Starting Launcher 3

2.2 License Management 4

2.2.1 Setting the License File 4

2.3 Launcher's Tabs 6

2.3.1 Learn Tab 9

2.3.2 Tools Tab 9

2.3.3 Services Tab 10

2.3.4 Utilities Tab 13

2.3.5 API Tab 15

2.3.6 Configuration Tab 15

2.3.7 What's New Tab 17

2.3.8 ADAS/Body Control Tab 18

2.3.9 Digital Cockpit/Telematics Tab 19

2.3.10 Simulation/Cloud Tab 20

2.3.11 Ecosystem Tab 21

2.3.12 User Tab 21

2.4 Help 22

2.5 Installing Target Libraries and RTI Components 23

2.6 Troubleshooting with Launcher 24

Chapter 3 Configuring Launcher

3.1 Launcher Tabs and Their Associated XML Files 26

3.1.1 Configuration file 27

3.2 How Launcher Finds Images 28

3.3 How Launcher Finds Buttons 28
3.4 Example: Adding a Button to Launcher's User Panel 29
3.5 Example: Adding a Tab to Launcher 31

Chapter 1 Introduction

RTI® Launcher is a graphical application that allows you to install, configure, and run *RTI Connext®* components. It automatically detects which RTI components are installed and enables their launch buttons, providing a convenient graphical interface to the configuration and run-time parameters.

You can use *Launcher* to install new components or target platforms into the current *RTI Connext* installation, see what is already installed, navigate to the documentation and web-based resources, create new source-code projects (via its integration with *rtiddsgen*), and run all the *RTI Connext* tools and command-line utilities.

Launcher is intended to be your "window" and "management console" for all your *RTI Connext* products. Keep it handy and use it as the "launch" interface whenever you need to start any RTI tool or service.

In addition, you can customize *Launcher* to run any program installed in your system, such as third-party tools that work with *RTI Core Libraries*, or commonly used editors. This can be easily done by changing the XML configuration file in the **resource/app/app_support/launcher/panels** subdirectory of your *Connext* installation directory.

1.1 Paths Mentioned in Documentation

The documentation refers to:

- **<NDDSHOME>**

  This refers to the installation directory for *RTI® Connext®*. The default installation paths are:
  - macOS® systems:
    /Applications/rti_connext_dds-7.7.0
  - Linux systems, non-*root* user:
    /home/<*your user name*>/rti_connext_dds-7.7.0

- Linux systems, *root* user:
  /opt/rti_connext_dds-7.7.0
- Windows® systems, user without Administrator privileges:
  <*your home directory*>\rti_connext_dds-7.7.0
- Windows systems, user with Administrator privileges:
  C:\Program Files\rti_connext_dds-7.7.0

You may also see **$NDDSHOME** or **%NDDSHOME%**, which refers to an environment variable set to the installation path.

Wherever you see **<NDDSHOME>** used in a path, replace it with your installation path.

Note for Windows Users: When using a command prompt to enter a command that includes the path **C:\Program Files** (or any directory name that has a space), enclose the path in quotation marks. For example:

```
"C:\Program Files\rti_connext_dds-7.7.0\bin\rtiddsgen"
```

Or if you have defined the **NDDSHOME** environment variable:

```
"%NDDSHOME%\bin\rtiddsgen"
```

- **API Reference HTML documentation**

  Default path: **<NDDSHOME>/doc/api/connext_dds/api_<language>**

- ***<path to examples>***

  By default, examples are copied into your home directory the first time you run *RTI Launcher* or any script in **<NDDSHOME>/bin**. This document refers to the location of the copied examples as ***<path to examples>***.

  Wherever you see ***<path to examples>***, replace it with the appropriate path.

  Default path to the examples:
  - macOS systems: **/Users/<*your user name*>/rti_workspace/7.7.0/examples**
  - Linux systems: **/home/<*your user name*>/rti_workspace/7.7.0/examples**
  - Windows systems: **<*your Windows documents folder*>\rti_workspace\7.7.0\examples**

    Where 'your Windows documents folder' depends on your version of Windows. For example, on Windows 10, the folder is **C:\Users\<*your user name*>\Documents**.

  Note: You can specify a different location for **rti_workspace**. You can also specify that you do not want the examples copied to the workspace. For details, see *Controlling Location for RTI Workspace and Copying of Examples* in the *RTI Connext Installation Guide*.

Chapter 2 Using RTI Launcher

2.1 Starting Launcher

To start *Launcher*:

- **On a Linux system**:

  You must have a graphical environment (such as X11, GNOME or KDE) running.
  - Open a command prompt and navigate to the *Connext* installation directory. Run the script **bin/rtilauncher**.
  - Alternatively, use your browser to navigate to your *Connext* installation directory's **bin** directory and double-click **rtilauncher.**
- **On a macOS system**:

  If you installed *Connext* in the **/Applications** directory, the LaunchPad utility should show *RTI Launcher* in the *Connext* folder (**rti_connext_dds-*version***). Click on it to open *Launcher*.
- **On a Windows system**:

  There are three ways to run *RTI Launcher* on a Windows system:
  - In the Start menu, expand the entry *RTI Connext* <*version*> and then select **RTI Launcher <*version*>**.
  - Use Windows Explorer to navigate to your *Connext* installation directory's **bin** folder (for example, **C:\Program Files\rti_connext_dds-*version*\bin.**, then double-click **rtilauncher.bat**.
  - From a command prompt, navigate to **<NDDSHOME>/bin** and run **rtilauncher.bat**.

2.2 License Management

Launcher itself does not require a license file. However, some of the *Connext* components may require a license file. *Launcher* will aid you in making sure a license file can be found by these applications when started from *Launcher*. *Launcher* will look for the license file named **rti_license.dat** in three locations in the order specified below:

- The path indicated by the **RTI_LICENSE_FILE** environment variable. The file does *not* need to be called **rti_license.dat** to use this method.
- The **RTI user home**. This is the **rti_workspace/***version* directory in the user's home folder. The specific location depends on the operating system.
- The Connext **installation directory**. This is the top-level directory where you installed the *Connext* bundle.

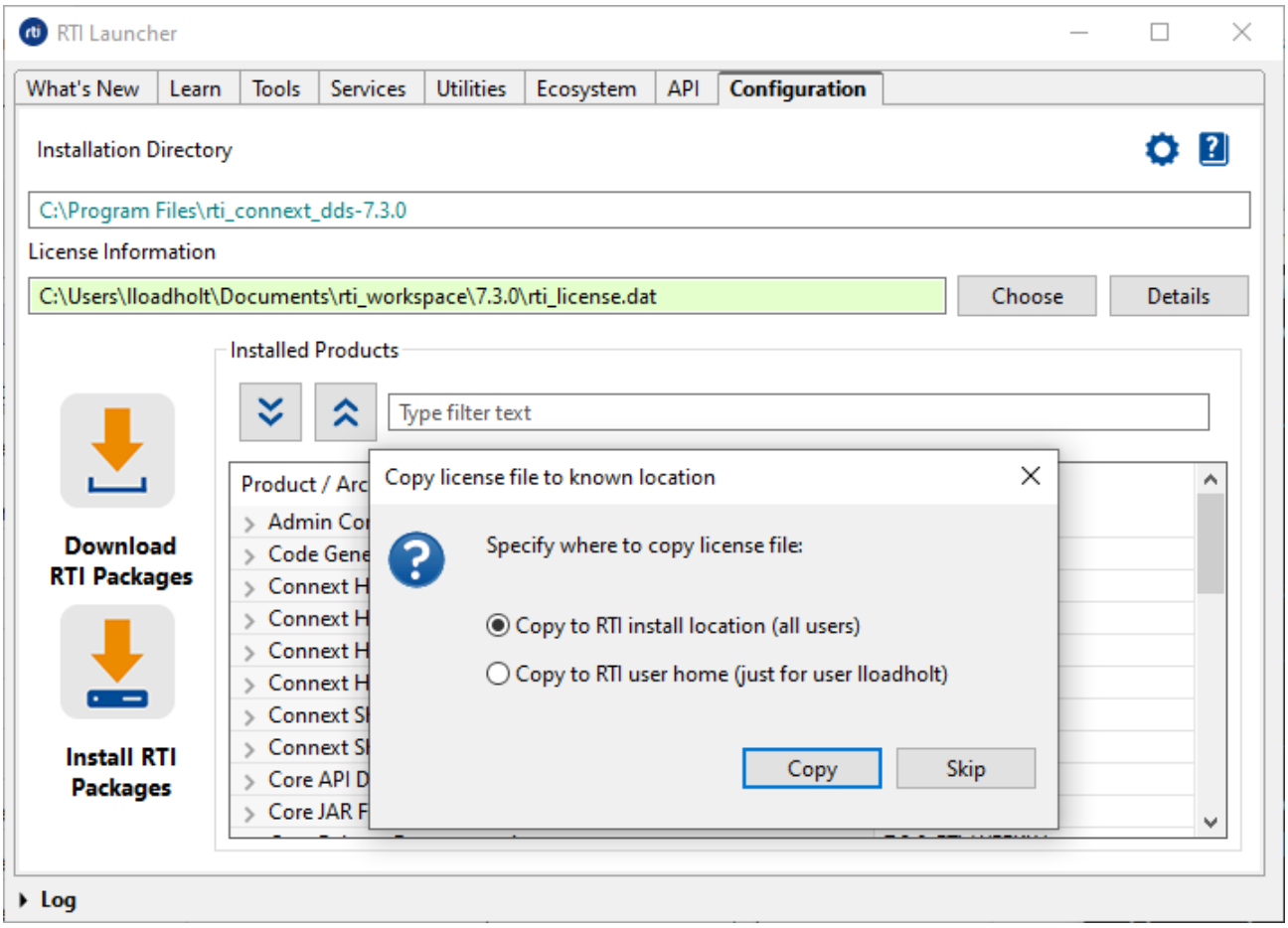
For example, if a free version (with "lm" in the bundle name) of *RTI Connext Professional* is installed in **C:\Program Files\rti_connext_dds-*version*** on a Windows system and there is no **RTI_LICENSE_FILE** environment variable set, *Launcher* will first look for **C:\Users*your user name*\Documents\rti_workspace*version*\rti_license.dat** (RTI user home). Then, if not found, it will look for **C:\Program Files\ rti_connext_dds-*version*\rti_license.dat**. See *License Management*, in the RTI Connext Installation Guide

2.2.1 Setting the License File

Some of *Launcher*'s tools may be license-managed. In this case, if *Launcher* has no knowledge of a valid license file, those tools will not become available until a valid license is provided. A warning message will be shown to indicate that the license file cannot be found.

When you have a valid license, there are a few ways to tell *Launcher* about the license file:

- Use the *Launcher* Configuration tab's **Choose** button. This will open a file chooser dialog that will let you navigate to the license file. After choosing the file, *Launcher* will prompt you with the following dialog:



You have three choices:

- **Copy to RTI install location (all users):** The chosen license file will be copied to the *Connext* installation directory, with the name **rti_license.dat**. *Launcher* will use this license file not only for the current execution and the current user, but for subsequent executions by any user in the system with access to the *RTI Core Libraries* installation. Note that if you have installed into a system directory, selecting this option may require system privileges.
- **Copy to RTI user home (just for user <user>)**: The chosen license file will be copied to the current user's **rti_workspace/<*version*>** directory. The license file will thus only be visible for the current user in subsequent executions of *Launcher*.
- **Skip**: The chosen license file will be applied by *Launcher* only for the current execution of the tool. It will not be automatically detected by the tool in subsequent executions.

- Copy the file manually to one of the aforementioned locations where *Launcher* will search by default, and name the file **rti_license.dat**. As mentioned before, if the RTI license file is copied to the RTI user home, it will only be visible for this user in subsequent executions of the tool.

When the file is copied to the RTI install location directly, it will be visible for all users with access to that directory.

The file may be copied both when *Launcher* has already been started or when it is offline. If the license file is copied while *Launcher* is executing, *Launcher* will detect the change automatically and apply the license to the UI.

- Set the **RTI_LICENSE_FILE** environment variable with the path to the license file. When setting this variable, not only *Launcher* but all the RTI applications will use the license file environment variable when executing.

  Notice that when copied to one of the known locations, all RTI applications will also detect the license file.

After successfully setting a license file with *Launcher*, all the tools that were disabled because of the license will become available and usable.

2.3 Launcher's Tabs

Launcher has a Graphical User Interface with tabs for different application categories. The contents of the tabs are XML-defined. If a tab contains no entries (buttons), or none of the applications in a tab are supported on your platform, that tab will not appear in *Launcher*.

The following tabs are always visible to the user:

- 2.3.1 Learn Tab on page 9
- 2.3.2 Tools Tab on page 9
- 2.3.3 Services Tab on page 10
- 2.3.4 Utilities Tab on page 13
- 2.3.5 API Tab on page 15
- 2.3.6 Configuration Tab on page 15

Launcher also includes other tabs, which may or may not be visible, depending on what *Connext* bundles you have.

- 2.3.7 What's New Tab on page 17
- 2.3.8 ADAS/Body Control Tab on page 18
- 2.3.9 Digital Cockpit/Telematics Tab on page 19
- 2.3.9 Digital Cockpit/Telematics Tab on page 19

- 2.3.11 Ecosystem Tab on page 21
- 2.3.12 User Tab on page 21

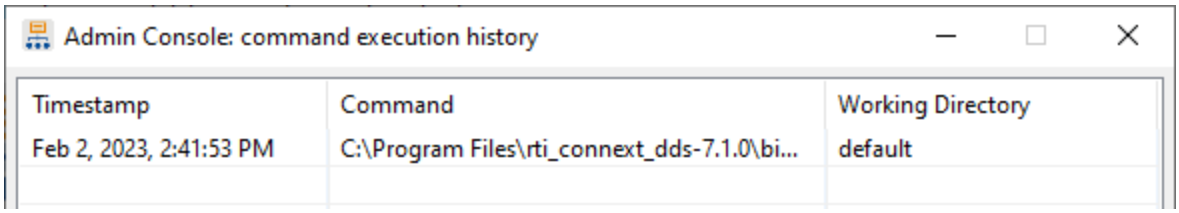
Note: All the tabs may or may not be visible if *Launcher* cannot find the resources needed to create the tab. For instance, if the **tools** folder is missing—or if there is a **tools** folder but no buttons configured inside it—the Tools tab will not display. See 3.1 Launcher Tabs and Their Associated XML Files on page 26 for details. You can also add new tabs to *Launcher*. See 3.5 Example: Adding a Tab to Launcher on page 31.

To start one of the applications in a tab, click on the application's button. Some applications will launch directly, such as *RTI Admin Console.* Other applications, like the ones in the Services or Utilities tab, will pop-up an application-specific dialog. This dialog allows you to configure the settings to run the application and then execute it.

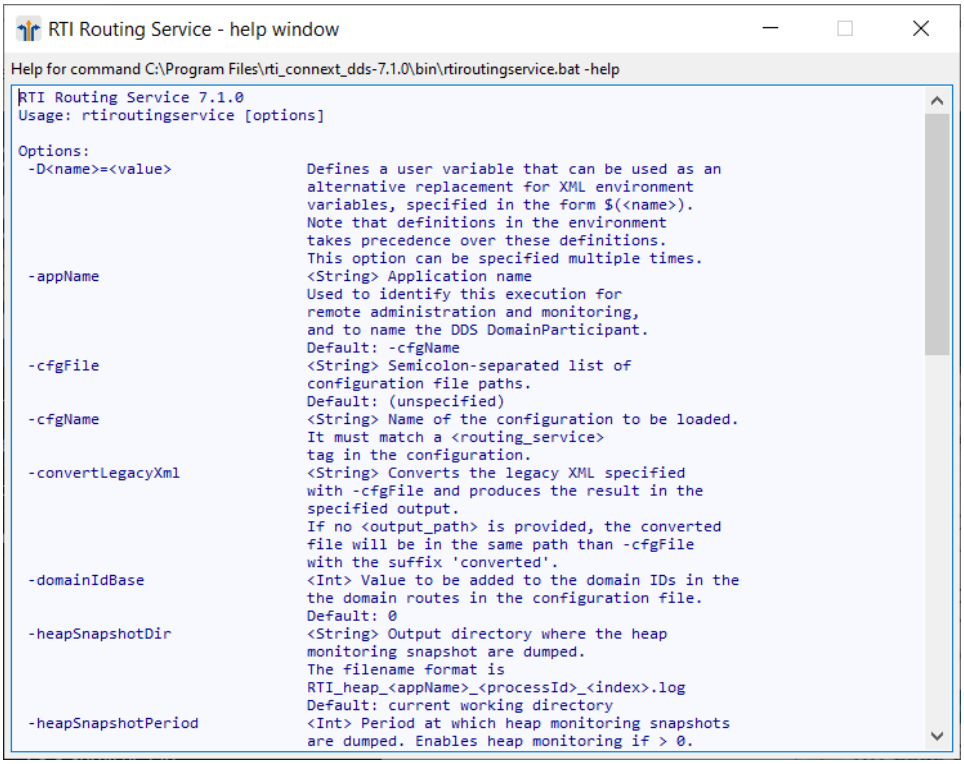
Right-click (or Command-Click on macOS systems) a button to open a context menu with additional information and options.

The context menu may differ between tools, but most include the folowing options:

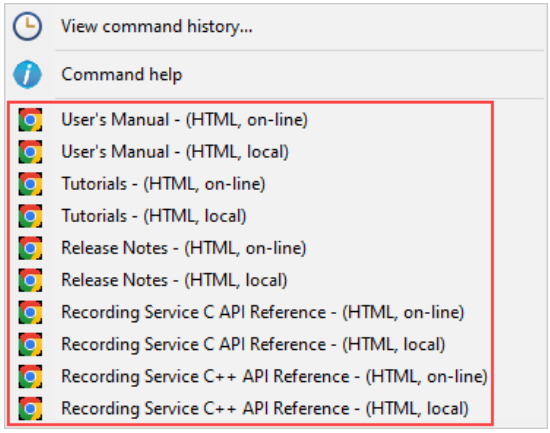
- **Install/Uninstall:** This option installs the tool if it is not yet installed, or uninstalls it if it is already installed.
- **View command history:** *Launcher* keeps track of all the successful command executions in the current *Launcher* session. When you click this menu item, it will open up the application's Command History dialog, which shows the command that was executed, when it was launched, and the working directory from where it was launched:



- **Command help:** Some applications provide usage notes regarding available command parameters. For these applications, **Command help** displays the available parameters.



- **Documentation:** These options link to on-line and local RTI documentation for the selected tool.



If a tool is not installed or cannot be used (due to an expired license, for instance), a different icon will appear and the button will not work.

Launcher automatically detects the presence of the applications in various ways. In the majority of cases, *Launcher* looks for installed applications in **<NDDSHOME>/bin**. For other applications, it looks in the Windows Registry. On Linux systems, Wireshark is found by using the UNIX *which* command.

Not all tools are available on all platforms. For example, *RTI Spreadsheet Add-in for Microsoft Excel* is only available on Windows systems. On non-Windows systems, its button still appears but will not work.

Note: Some applications in *Launcher* open a command prompt terminal. Whenever *Launcher* opens a command prompt terminal, it removes the content of the following environmental variables in order to successfully open some services:

- LD_LIBRARY_PATH on Linux systems
- DYLD_LIBRARY_PATH on macOS systems

2.3.1 Learn Tab

The Learn tab contains a broad range of technical resources to learn about the *RTI Connext* product line and its underlying Object Management Group (OMG) Data Distribution Service (DDS) technology. The Learn tab includes the following:

- **Shapes Demo** demonstrates *Connext* capabilities by publishing and subscribing to colored moving shapes.
- **Connext Chatbot** is an AI-based support tool that helps navigate documentation, understand *Connext* features, generate sample code, and troubleshoot issues with *Connext* applications.
- **Resources** contains links to useful resources for learning about DDS technology.

2.3.2 Tools Tab

The Tools tab contains applications that can be run directly, without requiring any additional application-specific setup. The Tools tab includes the following applications:

- **Admin Console** is a centralized tool for administering, supervising, and visualizing your distributed system. It provides control over RTI services such as *RTI Routing Service* and *RTI Recording Service*, offers a centralized way to receive and view log messages, detects configuration errors including type and QoS mismatches, and visualizes data. You can subscribe to the topics in the system to see the data being published in real time.
- **Monitor** provides a graphical view to the internal configuration and operation of *Connext* applications, including QoS and detailed statistics on connections, traffic, errors, and resource usage.
- **System Designer** is a graphical tool that allows you to define and configure *Connext* systems. You can use it as a user interface to XML-Based Application Creation, a mechanism that allows you to specify all aspects of a *Connext* system in XML format.
- **Wireshark** is a network packet analyzer. It can view and capture data packets in an RTI network, dissect them, and graph the output for easy analysis.

- **PlotJuggler RTI Edition (Experimental)** enhances PlotJuggler, an open source tool for visualizing times series of real-time streaming data, with a *Connext* plugin. This plugin allows PlotJuggler to join the DDS databus, discover DDS Topics, and directly plot DDS data. If you click this button when *PlotJuggler RTI Edition* is not installed, *Launcher* downloads and installs the package in the *Connext* installation directory.

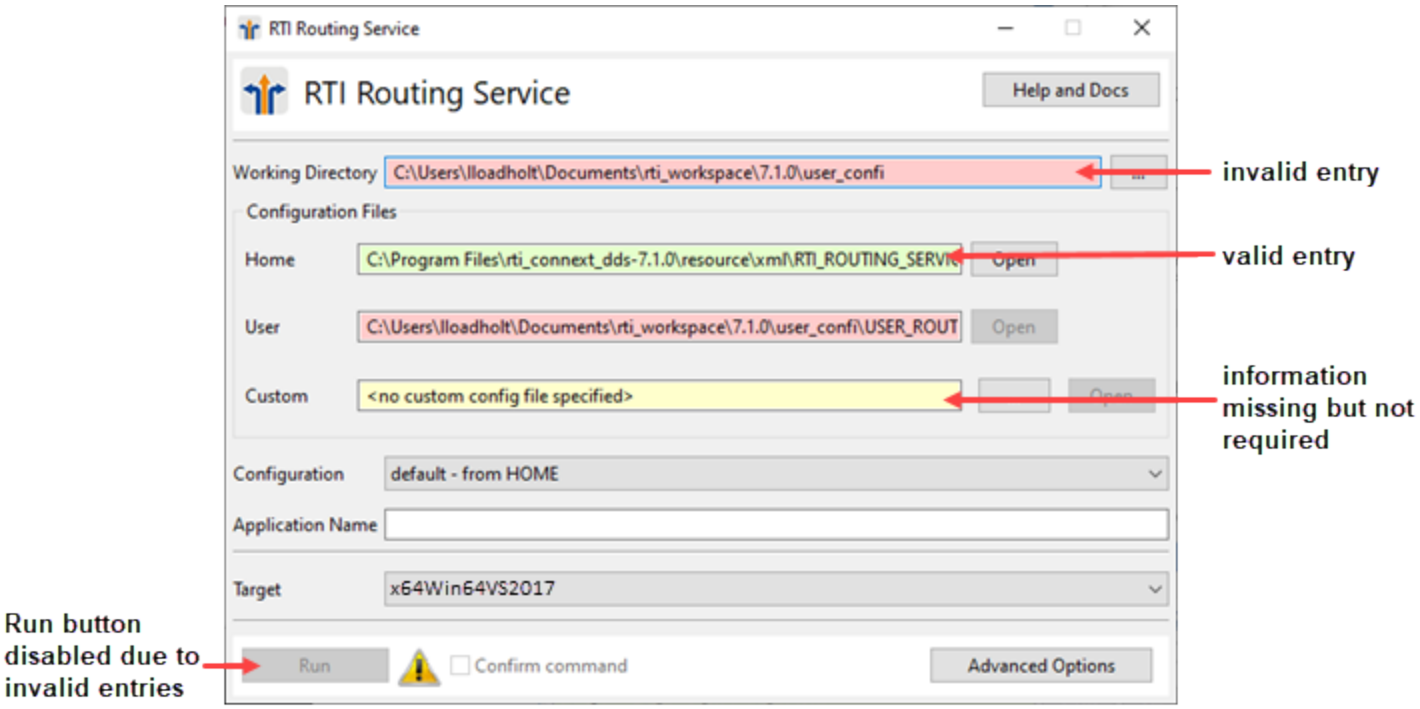
2.3.3 Services Tab

The Services tab provides buttons and dialogs to run RTI infrastructure services that require application-specific setup. The Services tab includes the following applications:

- **Routing Service** integrates disparate and geographically dispersed systems. It scales *Connext* applications across domains, LANs and WANs, including firewall and NAT traversal. It also supports *Connext*-to-*Connext* bridging by allowing you to make transformations in the data along the way.
- **Recording Service** records real-time data without having prior knowledge of the data-types or topics in the system.
- **Replay Service** replays recorded data by injecting it back into DDS. You can change data rates and QoS settings.
- **Persistence Service** saves data from *Connext* publishing applications to memory or permanent storage, so it can be delivered to subscribing applications that join the system at a later time—even if the publishing application has already terminated. Data can be persisted to files.
- **Web Integration Service** provides a simple, generic, standards-based interface that provides a transparent bridge between web-based services (HTTP/HTTPS) and unmodified *Connext* applications.
- **Cloud Discovery Service** is a stand-alone application that is needed when deploying *Connext* applications in dynamic environments where UDP/IP multicast is not available.
- **Queuing Service (experimental)** enables point-to-point messaging in *Connext*. It brokers interactions between message producers (*DataWriters*) and consumers (*DataReaders*), delivering each message (sample) to only one consumer

Dialogs in the Services and Utilities tabs also include validation for wrong parameters or paths. Text fields are highlighted with different colors depending on their validation status:

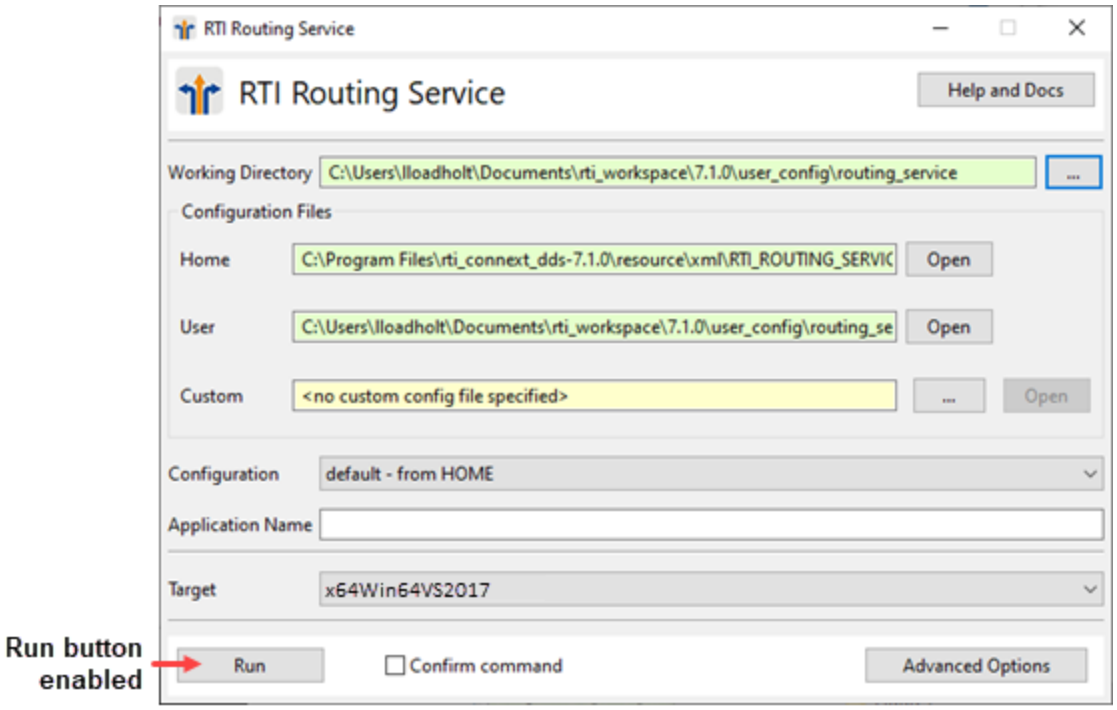
- Green indicates that the field has a valid entry.
- Yellow indicates that the information is missing, but not required.
- Red means the field contains invalid values (for example, a missing file).



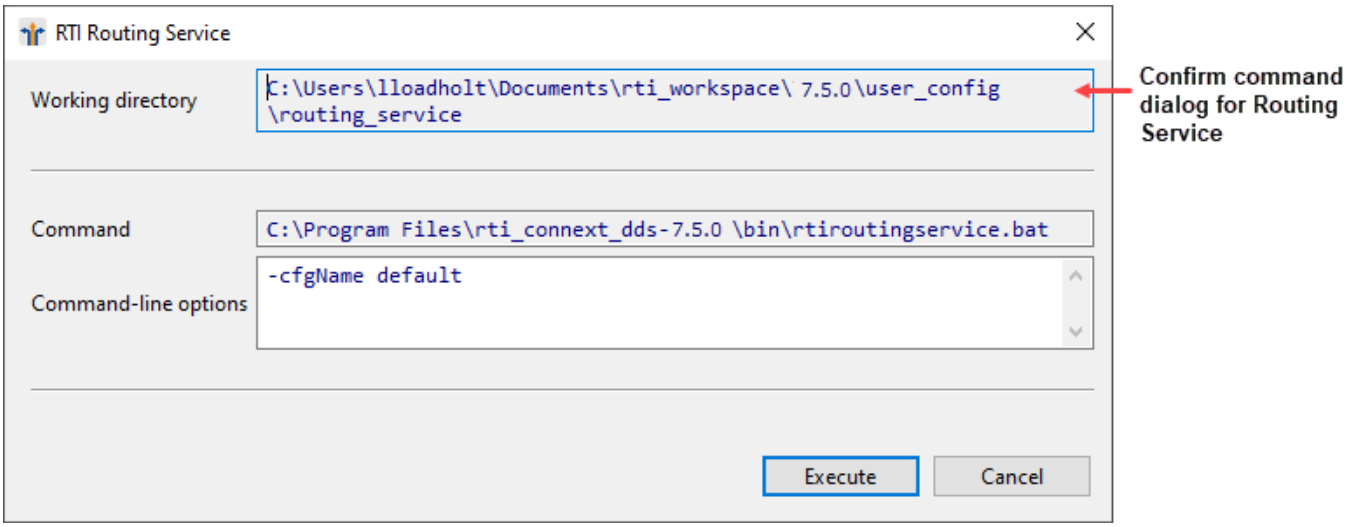
Hover over the warning icon to see an explanation of why the dialog is invalid.

When you are done configuring the service, click on the **Run** button to run the service. If the dialog has some invalid entries, the **Run** button will be disabled and a warning icon will appear next to it.

Note: The **Run** button will be available if at least one configuration file is valid. *Launcher* validates all the inputs again when you click on the **Run** button.



If you want to preview the command that will be used for execution, check the **Confirm command** checkbox. If the checkbox is checked, the name of the **Run** button will change to **Check**. If you click on **Check**, a pop-up dialog will show the command and command-line options that will be used and the working directory from which the command will be run.



The command-line options map to the fields you entered in the previous dialog (for example, the RTI Routing Service dialog above). You can add new command-line options or remove existing ones. See the service's documentation for more information about its full set of command-line options.

If the execution is successful, this command will be stored in the Command History of the service. See 2.3 Launcher's Tabs on page 6 for more about viewing command history.

For dialogs that have a Target field, *Launcher* runs the executable for the selected target. The choices for this field depend on which target bundles you have installed. To change the default Target value, either:

- Edit the file **$NDDSHOME/resource/scripts/rticommon_config** to add your selected platform to **connextdds_architecture** (on Linux and macOS systems) or **connextddsArchitecture** (on Windows systems).
- Or set the **CONNEXTDDS_ARCH** environmental variable to your selected target architecture.

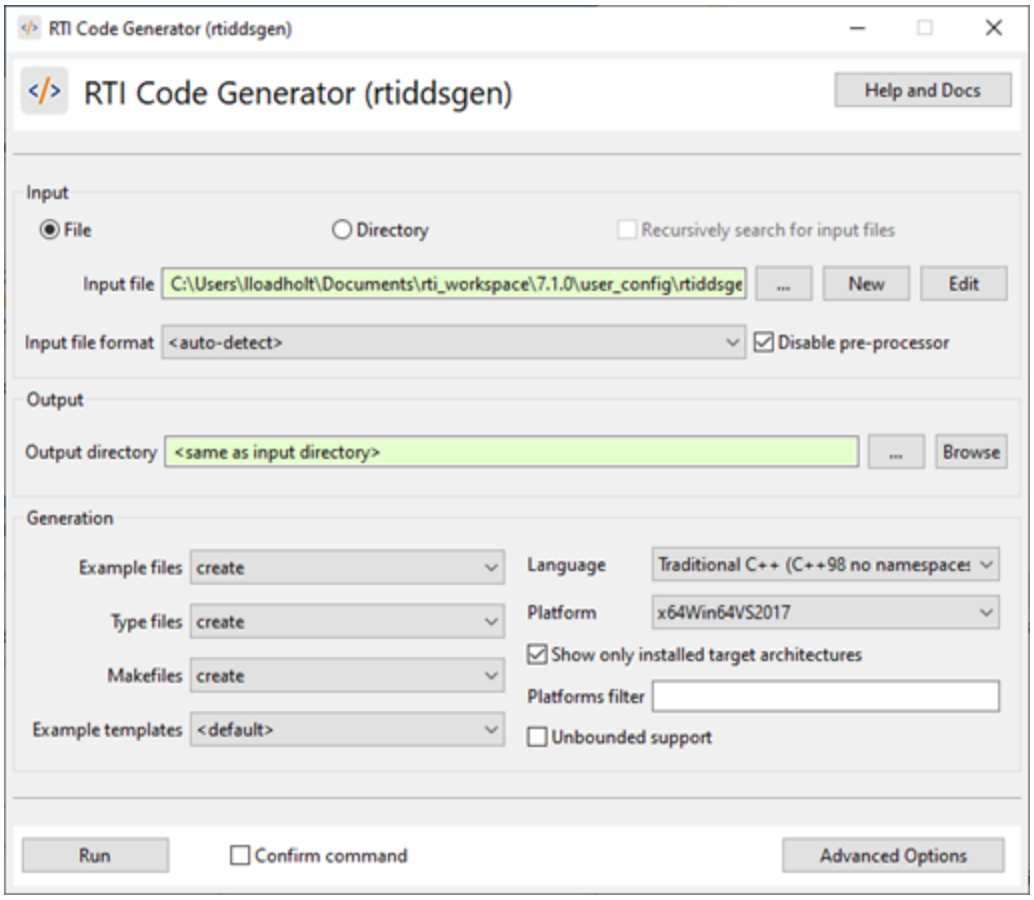
Note: The value set in the **rticommon_config** file takes precedence over the environmental variable.

2.3.4 Utilities Tab

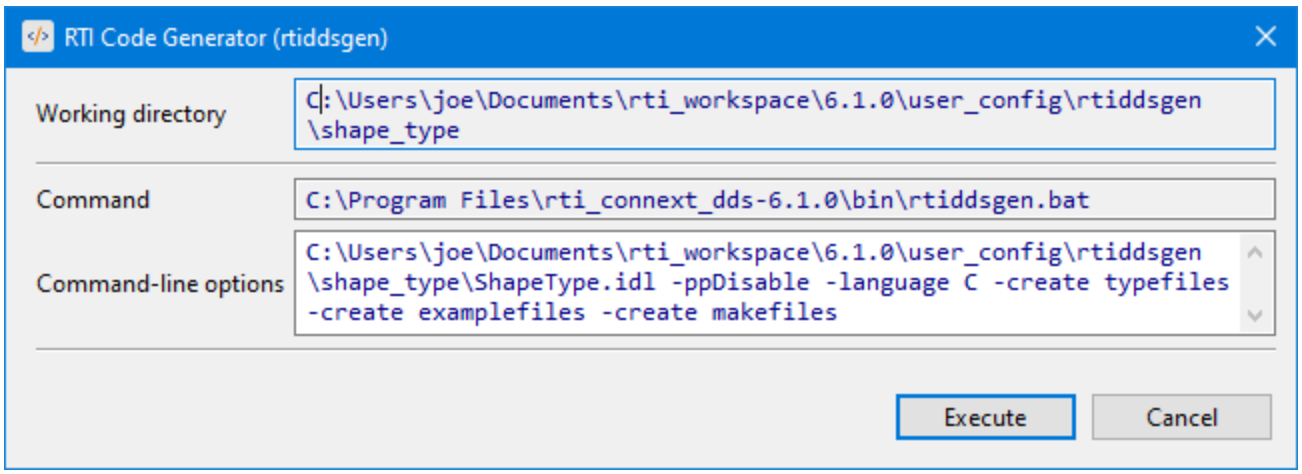
The Utilities tab contains helper applications provided with *Connext* to assist you when testing, designing, or inspecting your DDS system.

- **Code Generator**, also known as *rtiddsgen,* generates the code to serialize and deserialize application data on the wire from an IDL (or XML) description of the data-type. It also generates example code and makefiles/project files so you can quickly build a working application.
- **DDS Ping**, also known as *rtiddsping*, publishes (and subscribes to) DDS ping messages to test system connectivity.
- **DDS Spy**, also known as *rtiddsspy*, can subscribe to any DDS Topic and print out all the data being published.
- **Data Type Convert** converts data-type formats between IDL and XML.
- **Record Database Converter** converts database files recorded by *Recorder* into readable, commonly accepted formats such as a CSV file and JSON or XCDR (in SQLite format).
- **Connext Professional Terminal** opens a terminal window configured with the environment variables necessary for building *Connext* applications.
- **Perftest** is a command-line tool intended to measure minimum latency, maximum throughput, and load latency. This tool is highly configurable, enabling you to closely match your specific environment. If you click this button when Perftest is not installed, *Launcher* downloads and installs it to your examples folder at **\utilities\perftest\<perfest_version>**.

When you click on a utility button, a configuration dialog will pop up. Each utility has a different dialog, customized for its own needs. For example, the Code Generator dialog looks like this:



The Code Generator dialog can generate code for *Connext*. You can see the command line by selecting the **Confirm command** checkbox and clicking **Check**.



2.3.5 API Tab

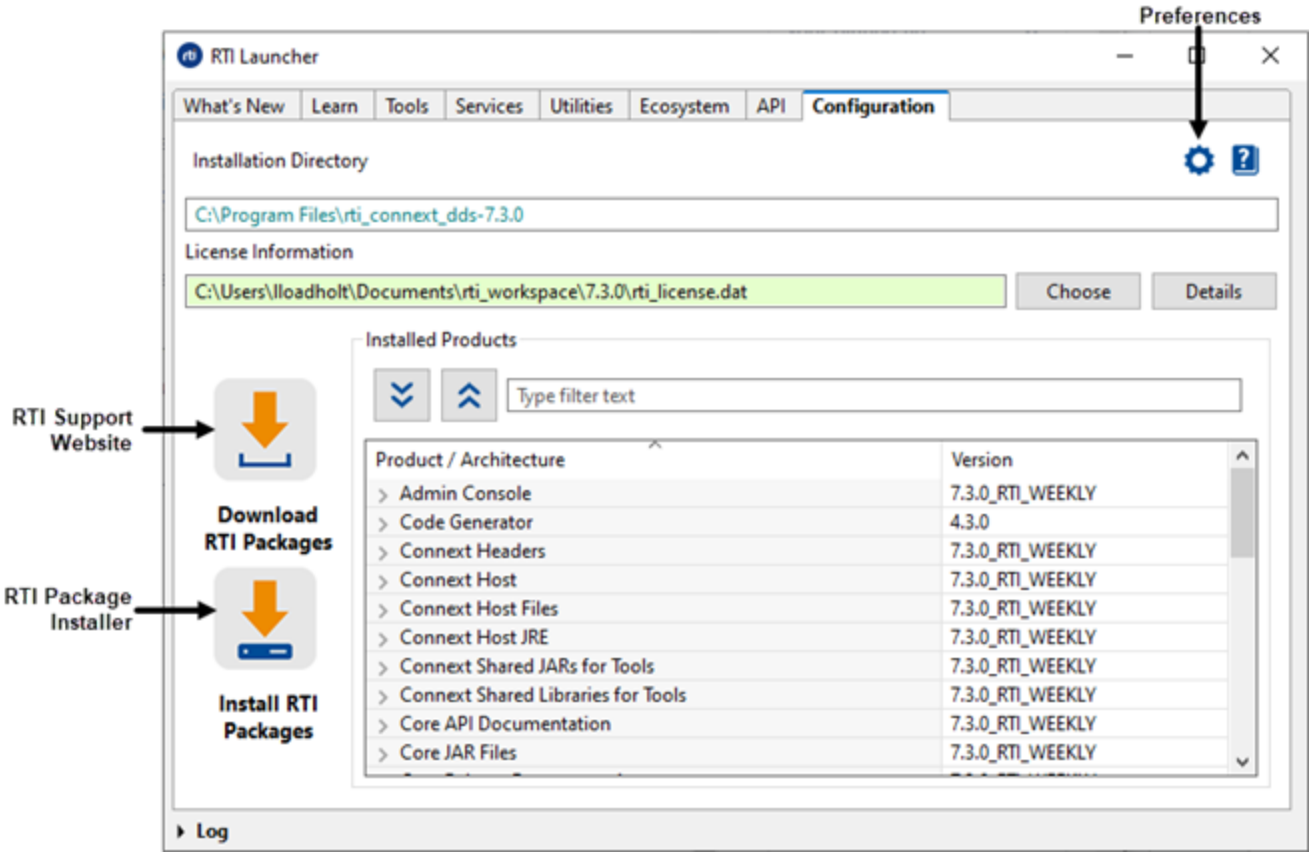
The API tab provides easy access to API documentation for all the languages supported by *Connext*: C, C++, Java, C#, and Python. This tab also includes links to *RTI Connector* and experimental gateways.

- **Connector for JavaScript** provides a quick and easy way to write applications that publish and subscribe to the *RTI Connext* DDS databus in JavaScript (with Node.js) and other languages.
- **RTI Connext Nano (experimental)** is a standards-compliant implementation of the OMG DDS-XRCE™ protocol, enabling resource-constrained devices to join DDS domains.
- **RTI Connext Gateway (experimental)** is an open framework, based on the *RTI Routing Service* SDK, that enables you to easily add new adaptors, processors, and transformations to a DDS databus.
- **OPC UA/DDS Gateway (experimental)** provides a generic, standards-based solution that provides a transparent bridge between OPC UA and DDS applications.

2.3.6 Configuration Tab

The Configuration tab provides general information regarding the current *Connext* installation. This tab:

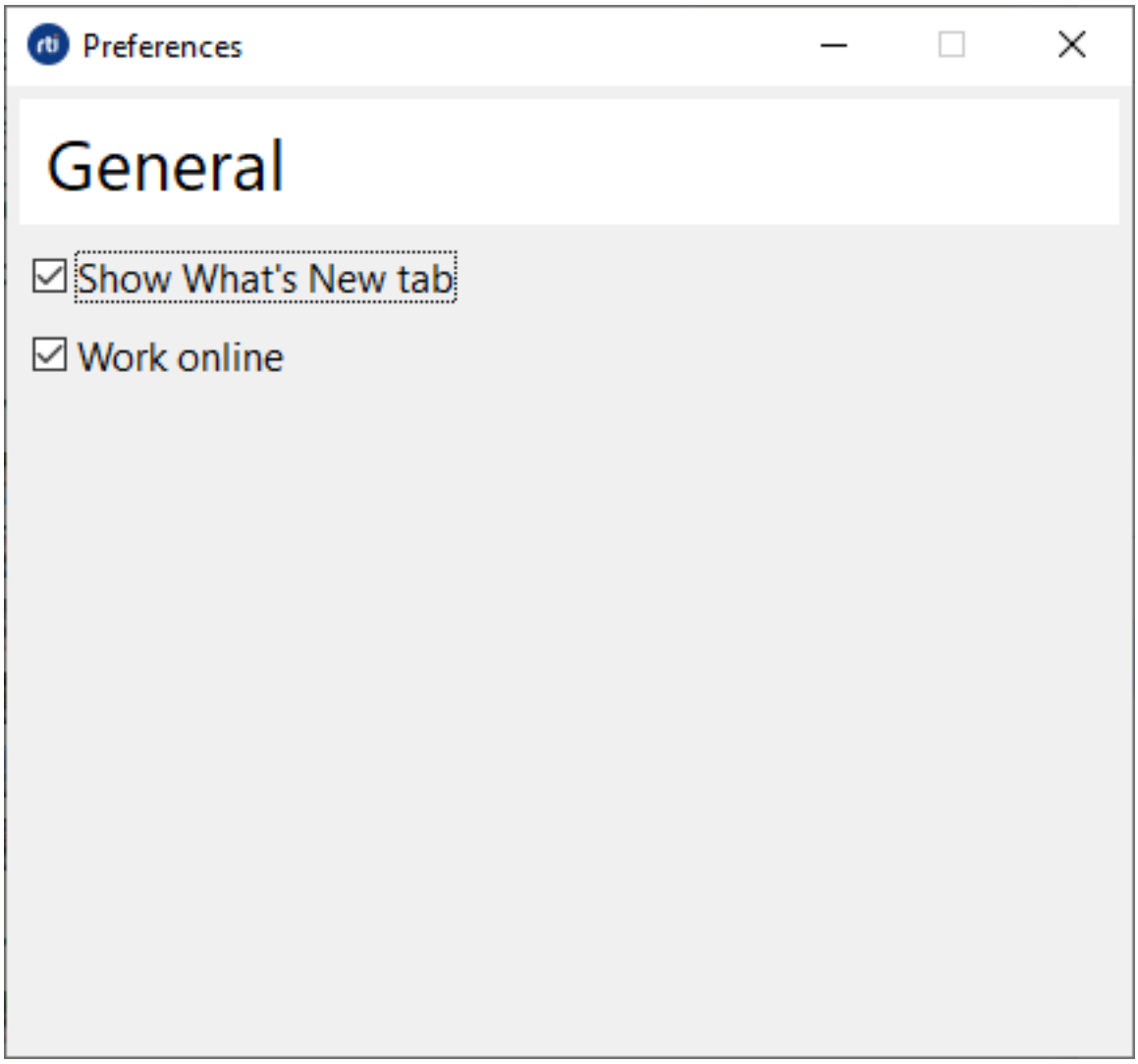
- Displays the current installation directory for *Connext*.
- Displays the license file and allows changes to its location. A license may not be necessary depending on the bundle installed. *Launcher* detects the need for a license file on a per-product basis.
- Displays a table with the installed products and libraries, and their installed versions. This table can be filtered by using the filter field. The table can also be fully expanded/collapsed by using the two buttons next to the filter field.
- Provides access to the RTI Package Installer through the **Install RTI Packages** button, to install additional components and libraries into an existing *Connext* installation.
- Provides access to the RTI Support website via the **Download RTI Packages** button.
- Provides access to the **Preferences** dialog.



2.3.6.1 Preferences Dialog

This dialog is used to configure general settings in *Launcher*:

- Hide or show the What's New tab.
- "Work online" means that *Launcher* will get the latest blog posts or news from the RTI Website when displaying the What's New tab. Disabling this option means *Launcher* will use local content that was stored when you last accessed the internet (when you downloaded the current release, or the last time the buttons were clicked and the content was stored from the internet). The local content is stored in **rti_workspace/*version*/user_config/launcher/rss**.



The **Preferences** settings are stored in the file **launcher.properties**, in **rti_workspace/*version*/user_config/launcher**.

2.3.7 What's New Tab

The What's New tab includes three different buttons used to select the content to display:

- **What's New** shows a brief summary of the new features in the current version of *RTI Connext Professional*.
- **News** shows the latest news from https://www.rti.com/news.
- **Blogs** shows the latest blog posts from https://www.rti.com/blog.

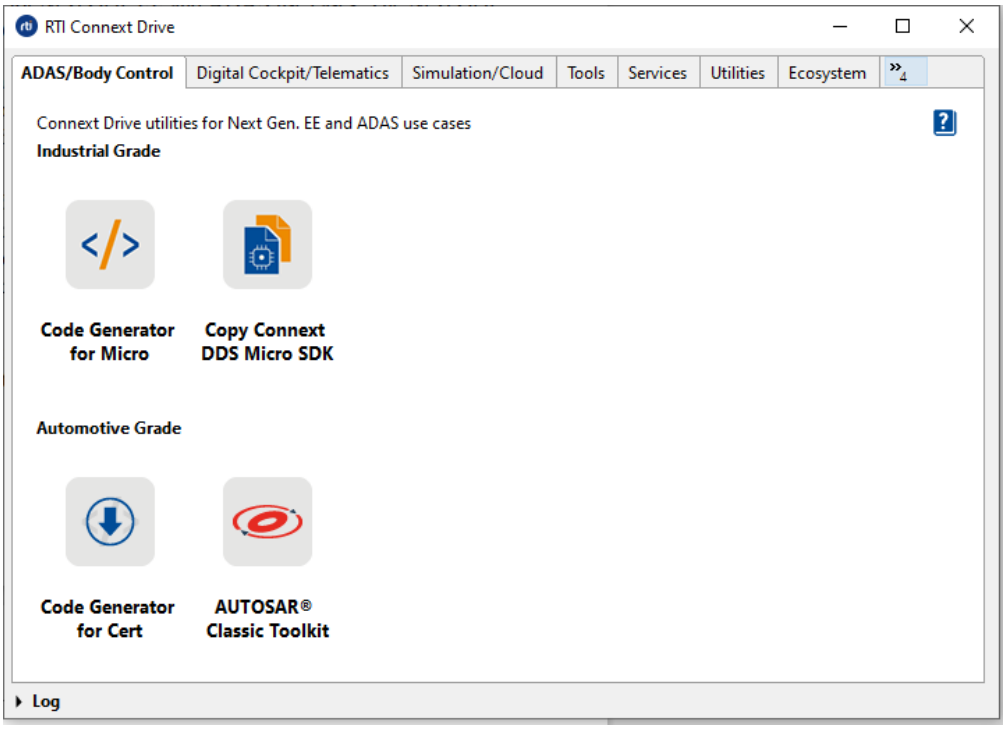
This tab requires an internet connection to show the latest news and blog posts from the RTI website. *Launcher* updates this content each time the buttons are clicked, as long as "Work online" is checked in the **Preferences** dialog. See 2.3.6.1 Preferences Dialog on page 16 for more information.

You can hide this tab by unchecking the option "Show What's New tab" in the Preferences dialog. You can also configure *Launcher* to not require an internet connection by unchecking the option "Work online." See 2.3.6.1 Preferences Dialog on page 16 for more information on these options.

2.3.8 ADAS/Body Control Tab

The ADAS/Body Control tab provides utilities for Next Gen. EE and ADAS use cases. The ADAS/Body Control tab is only visible when **Connext Drive** is installed. It includes the following applications:

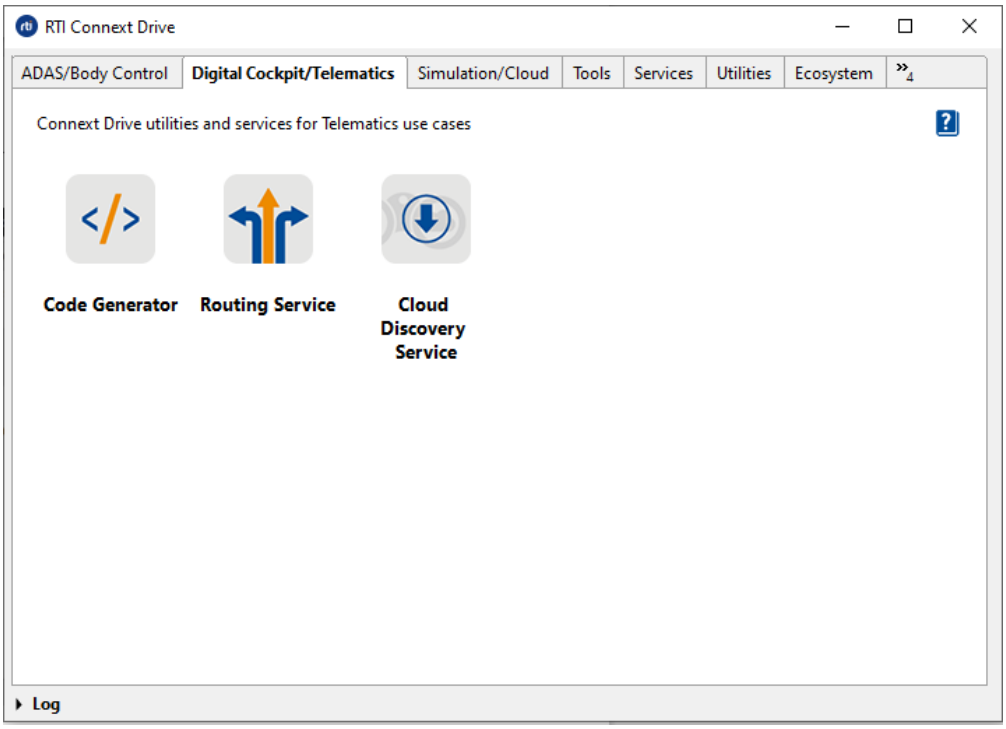
- **Code Generator for Micro**, also known as rtiddsgen, generates the code for *Connext Micro* 2.4.14 to serialize and deserialize application data on the wire from an IDL (or XML) description of the data-type. It also generates example code and makefiles/project files so you can quickly build a working application.
- **Copy Connext Micro SDK** copies the *Connext* Micro SDK (source code + code generator) to a specified directory. The context menu also allows you to copy any version of *Connext Micro* that has been installed, or to copy the source path of any version of *Connext Micro* (the path where the SDK was copied, or the original folder if the SDK has never been copied).
- **Code Generator for Cert**, also known as rtiddsgen, generates the code for *ConnextCert* 2.4.12.1 to serialize and deserialize application data on the wire from an IDL (or XML) description of the data-type. It also generates example code and makefiles/project files so you can quickly build a working application. By default, this button will be disabled until the *Cert* package is installed.
- **AUTOSAR® Classic Toolkit**, is a set of tools supporting automatic conversion of data type definitions across standard formats and generation of supporting C code for data conversion and marshaling between the RTE and DDS communication frameworks.



2.3.9 Digital Cockpit/Telematics Tab

The Digital Cockpit/Telematics tab provides utilities for telematics use cases. The Digital Cockpit/Telematics tab is only visible when **Connext Drive** is installed, and includes the following applications:

- **Code Generator**, also known as rtiddsgen, generates the code to serialize and deserialize application data on the wire from an IDL (or XML) description of the data-type. It also generates example code and makefiles/project files so you can quickly build a working application.
- **Routing Service** integrates disparate and geographically dispersed systems. It scales *Connext* applications across domains, LANs and WANs, including firewall and NAT traversal. It also supports *Connext*-to-*Connext* bridging by allowing you to make transformations in the data along the way.
- **Cloud Discovery Service** is a standalone application that is used to deploy *Connext* applications in dynamic environments where UDP/IP multicast is not available.

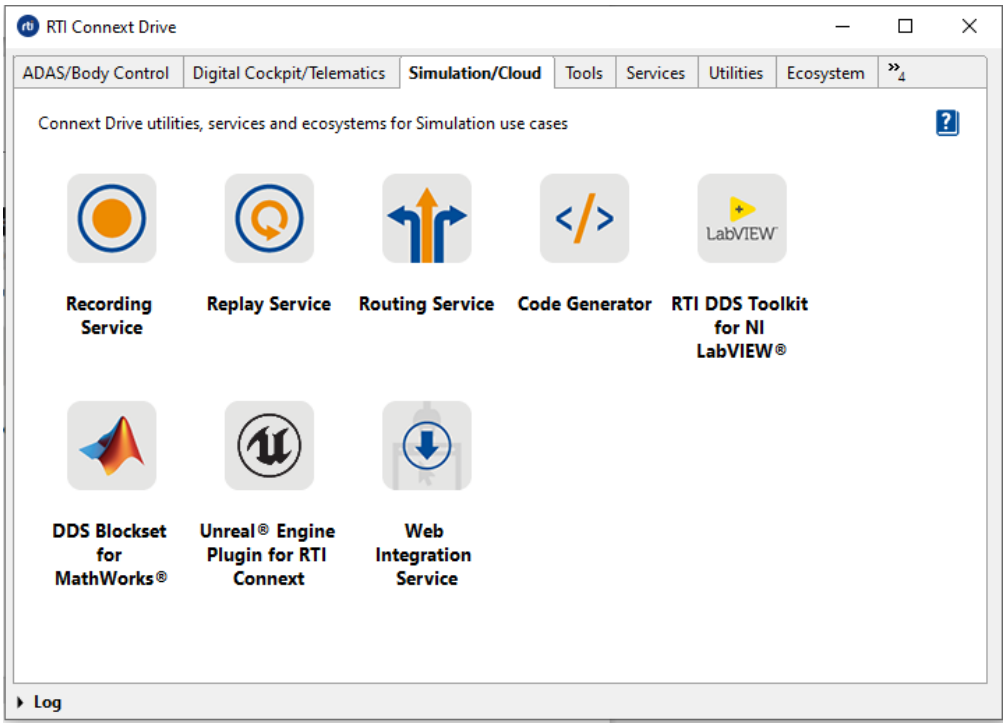


2.3.10 Simulation/Cloud Tab

The Simulation/Cloud tab provides utilities, services and ecosystems fitting simulation use cases needs. The Simulation/Cloud tab is only visible when **Connext Drive** is installed, and includes the following applications:

- **Recording Service** records real-time data without having prior knowledge of the data-types or topics in the system. (This service is also available by using Recording Console in the Tools tab.)
- **Replay Service** replays recorded data by injecting it back into DDS. You can change data rates and QoS settings. (This service is also available by using the Recording Console in the Tools tab.)
- **Routing Service** integrates disparate and geographically dispersed systems. It scales *Connext* applications across domains, LANs and WANs, including firewall and NAT traversal. It also supports *Connext*-to-*Connext* bridging by allowing you to make transformations in the data along the way.
- **Code Generator**, also known as rtiddsgen, generates the code to serialize and deserialize application data on the wire from an IDL (or XML) description of the data-type. It also generates example code and makefiles/project files so you can quickly build a working application.
- **RTI DDS Toolkit for NI® LabVIEW®** provides VI blocks that allow you to easily publish and share data between multiple LabVIEW applications, NI Linux Real-Time controllers and third-party applications that use DDS.

- **DDS Blockset for MathWorks®** provides apps and blocks for modeling and simulating software applications that publish or subscribe to Data Distribution Service (DDS) using *RTI Connext*.
- **Unreal® Engine Plugin for RTI Connext** is built on the Data Distribution Service (DDS) standard, the leading data-centric publish/subscribe connectivity standard for integrating distributed real-time applications, extending the benefits of *Connext* solutions to Unreal Engine projects
- **Web Integration Service** provides a simple, generic, standards-based interface that provides a transparent bridge between web-based services (HTTP/HTTPS) and unmodified *Connext* applications.



2.3.11 Ecosystem Tab

This tab is reserved for applications from RTI's partners. For information about RTI's strategic partners, see the RTI Partners website: https://www.rti.com/partners. This tab may be empty in some releases and, therefore, not shown at all.

Products downloaded by using the Ecosystem tab in *RTI Launcher* are not part of RTI; contact the third-party provider for more information about each product.

2.3.12 User Tab

The User tab is reserved for customizing *Launcher*. Its purpose is to allow users to integrate their preferred applications in *Launcher*. The User tab is not visible the first time *Launcher* is run; it's empty by

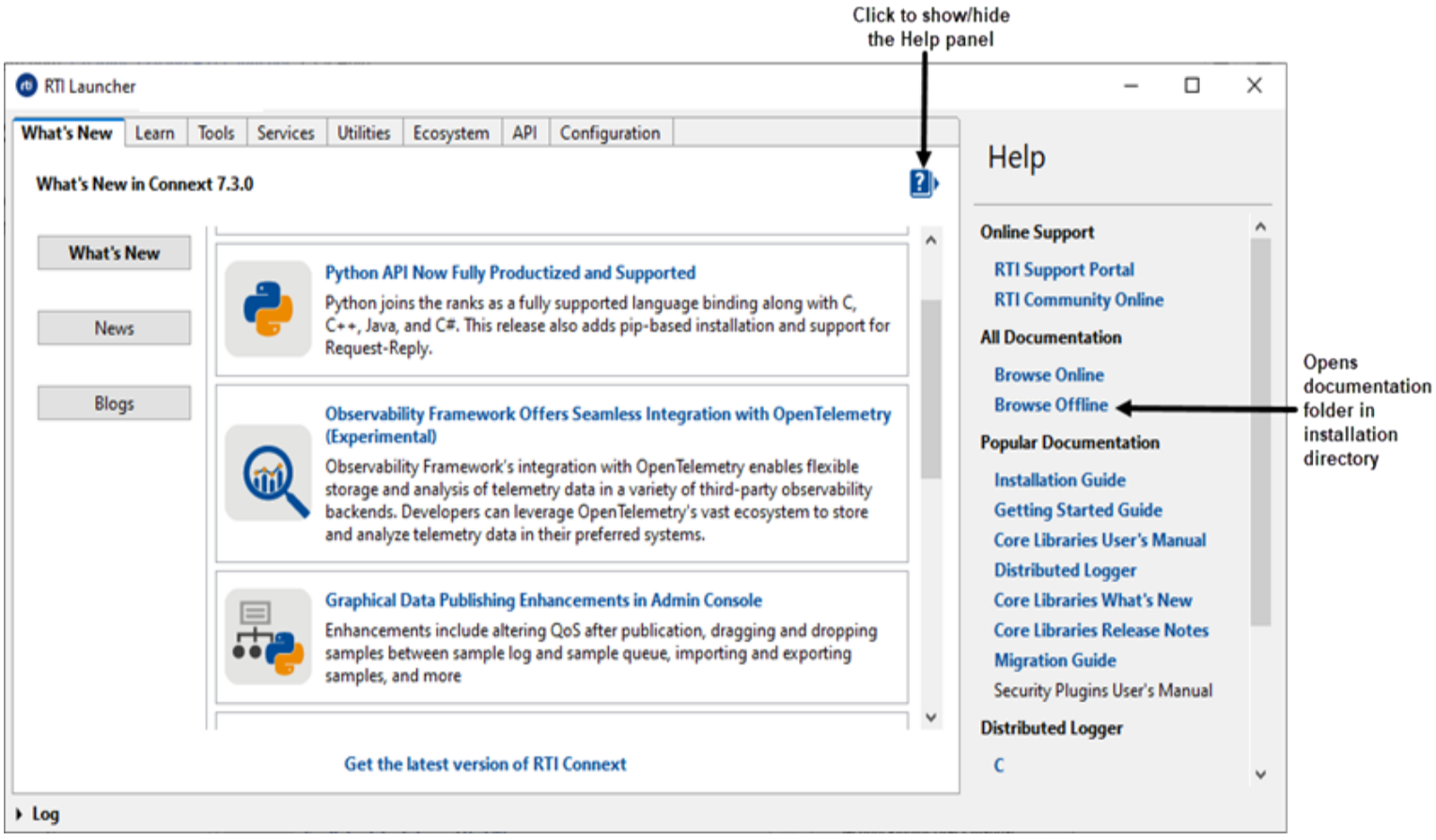
default, and *Launcher* doesn't show tabs that are empty. The folder that defines this tab is in the RTI workspace in your home folder: **rti_workspace/*version*/user_config/launcher/panels/user**. Inside this folder, you will find two files:

- **configuration.xml**: used to configure the name of the tab, add a description to the tab, and alter the order of the buttons.
- **example.xml**: used to add a shortcut button to any application in the system.

You can modify the **user** folder to add shortcuts for any application in the system. See Chapter 3 Configuring Launcher on page 26 for more details on how to do this.

2.4 Help

By default, the help content, which includes links to documentation and other information such as the RTI Customer Portal and RTI Community website, is shown. Each tab has a button in the upper right hand corner which is used to show or hide the help content.



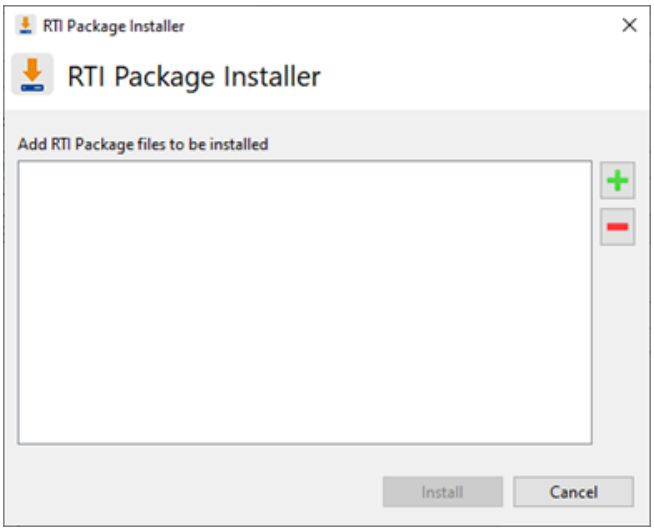
The **Browse Offline** link opens the README file in your *Connext* installation directory. Links to specific guides and API references also open those documents in the installation directory.

2.5 Installing Target Libraries and RTI Components

Launcher uses the RTI Package Installer to install new libraries and components. The Package Installer accepts files in RTIPKG format (***.rtipkg**). RTI's target libraries and add-on components are available in this RTIPKG format when downloaded from the RTI download portal.

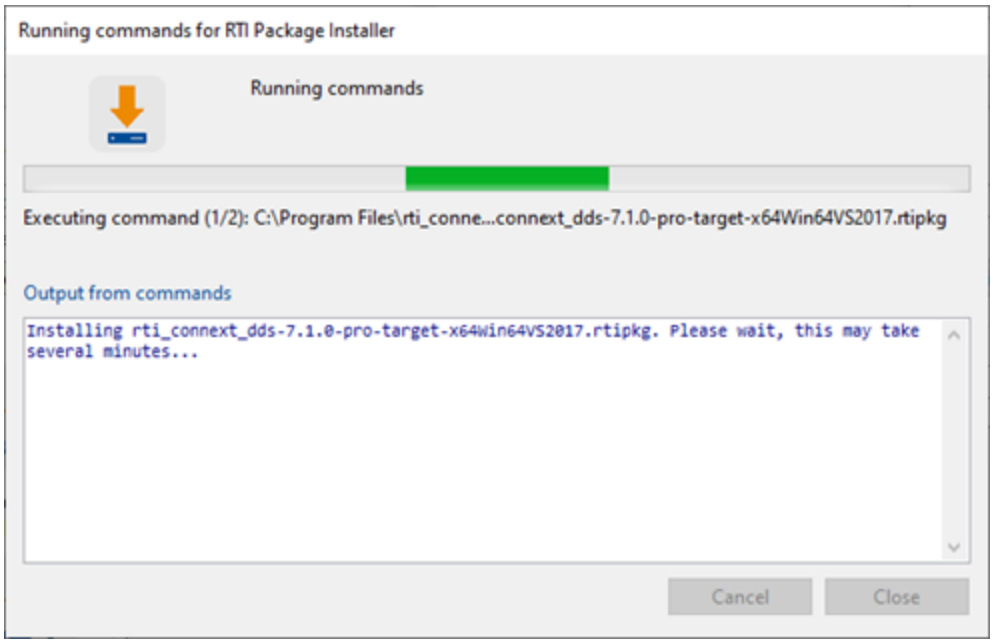
The RTI Package Installer executable is in the *Connext* installation directory, in the **bin** directory. It can also be executed from the command-line prompt.

In *Launcher*, the RTI Package Installer is in the Configuration tab. When you click on it, a dialog will pop up, like this:



At this point, select the RTIPKG files to be installed. If you click on the '+' button, a file chooser dialog will appear, allowing you to navigate to the file(s) you want to install. You can also drag and drop **.rtipkg** files into this window.

Once the files are selected, click the **Install** button. The install commands are launched sequentially. A progress dialog will appear:

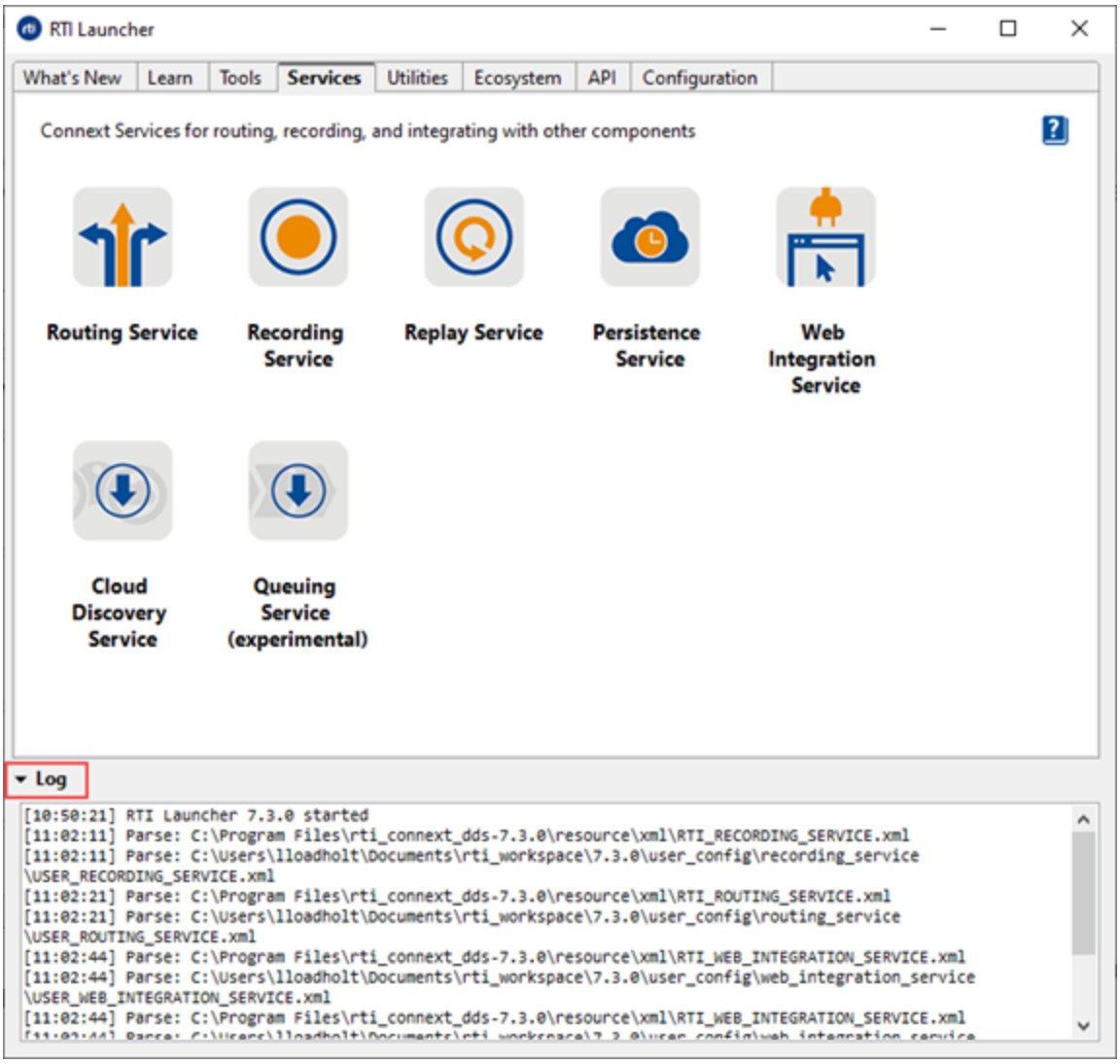


Note: The drag and drop feature won't work if you opened *Launcher* as an administrator on a Windows system.

2.6 Troubleshooting with Launcher

Launcher provides two debugging or troubleshooting mechanisms:

- **Log view**: *Launcher* produces log messages when it performs an action, an error occurs, etc. You can access these log messages from any of the tabs by clicking on the expandable 'Log' view. When clicked, the view will expand and show all the messages produced up to that point:



- **Log files**: *Launcher* logs output and errors to files in a special directory. This directory can be found in the user home's RTI workspace directory: **rti_workspace/<version>/user_config/launcher/logs**. *Launcher*'s execution log files are named with the following pattern: **launcher-log-<timestamp>.txt.** Every execution of *Launcher* creates a different log file.

For applications not launched in command-prompt windows—like the applications in the Tools tab—*Launcher* also logs their output into text files for application troubleshooting. These files are stored in the same place as the *Launcher* logs (**rti_workspace/*version*/user_config/launcher/logs**). They are named following this pattern: ***log-<executable name>-<timestamp>.out***.

Chapter 3 Configuring Launcher

3.1 Launcher Tabs and Their Associated XML Files

Launcher's tabs are defined using XML inside a folder. There is a folder for each tab. Each folder contains an XML file for each button and an optional configuration file that is used to configure the name of the tab, the description of the tab, and the order of the buttons. These are predefined XML files.

You should not modify the predefined XML files except the configuration file (configuration.xml), which is used to configure the order of the buttons in a tab. The predefined XML files are in the *RTI Connext* installation directory, in **<NDDSHOME>/resource/app/app_support/launcher/panels**. They configure the following tabs:

- Learn
- Tools
- Services
- Utilities
- Ecosystem
- API

There are other *Launcher* configuration files that are intended to be customized by users. You will find these files in your RTI workspace[a] : **rti_workspace/<*version*>/user_config/launcher/panels** directory. You can use these files to configure the following tabs:

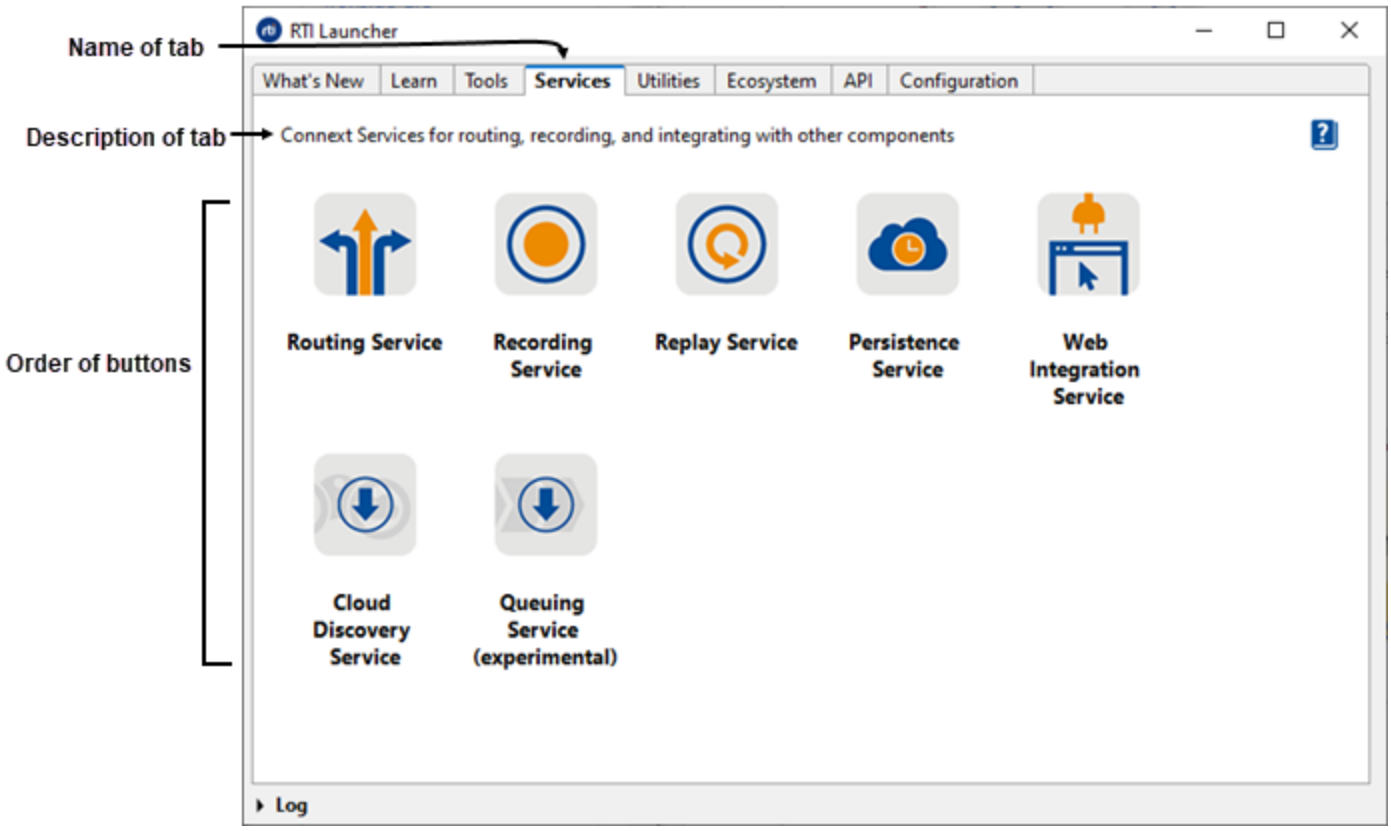
- User (see 3.4 Example: Adding a Button to Launcher's User Panel on page 29)

[a] See 1.1 Paths Mentioned in Documentation on page 1.

Note: You can add more tabs to Launcher. See 3.5 Example: Adding a Tab to Launcher on page 31 for more information.

3.1.1 Configuration file

The configuration file is an optional file used to configure the following content in each tab:



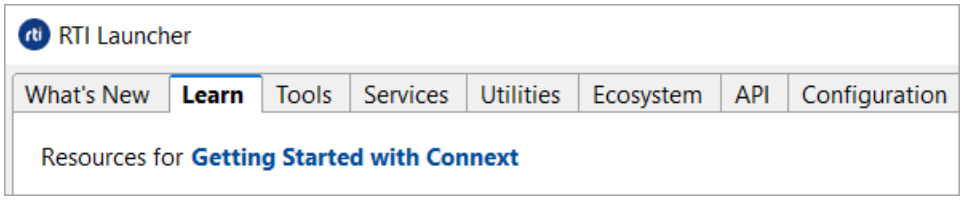
- **Name of the tab:** If not specified, Launcher will use the name of the folder.

```
<name>Services</name>
```

- **Description of the tab:** Displayed at the beginning of each tab. The description can be plain text or a link to any resource, such as a PDF or website. The following shows how the Learn tab has been configured to show its description:

```
<description>
    <element>
        <message>Resources for </message>
    </element>
    <element>
        <message>Getting Started with Connext</message>
        <link>manuals/rti_connext_dds-<version>/doc/manuals/connext_dds_
professional/getting_started_guide/index.html</link>
    </element>
</description>
```

The above description has two parts, both plain text; however, the second one is also a link to the *RTI Connext Getting Started Guide*:



- **Organization of buttons in the tab:** If not specified, the buttons will be organized alphabetically by name. To organize the buttons, specify the name of the file and not the name of the button itself. You can modify the configuration file of all the tabs in *Launcher* to reorganize the buttons.

```
<order>
    <name>RoutingService.xml</name>
    <name>RecordingService.xml</name>
    <name>ReplayService.xml</name>
    <name>PersistenceService.xml</name>
    <name>DataBaseIntegration.xml</name>
    <name>QueuingService.xml</name>
    <name>WebIntegrationService.xml</name>
</order>
```

  For example: consider two buttons in the user tab, Eclipse.xml and Notepad.xml. Launcher adds first the Eclipse button and then the Notepad button, alphabetically by name. If you rename the file Eclipse.xml to Z.xml, however, the first button will now be the Notepad button (Eclipse will be the second button). Alternatively, use the <order> tag to explicitly choose the order of the buttons.

3.2 How Launcher Finds Images

Launcher looks for the image files to be shown for the applications in the following places:

- In the *Connext* installation directory, in **<NDDSHOME>/resource/app/app_support/launcher/icons**. This folder is used for RTI-related applications and applications by RTI's partners (shown in the Ecosystem tab).
- In your RTI workspace directory, under **rti_workspace/***version***/user_config/launcher/icons**. This folder may not exist after installing and first running the tool. To add user-related images, you have to create this folder and add your icon image files there.
- Absolute paths. *Launcher* can detect files described as an absolute path to the image.

3.3 How Launcher Finds Buttons

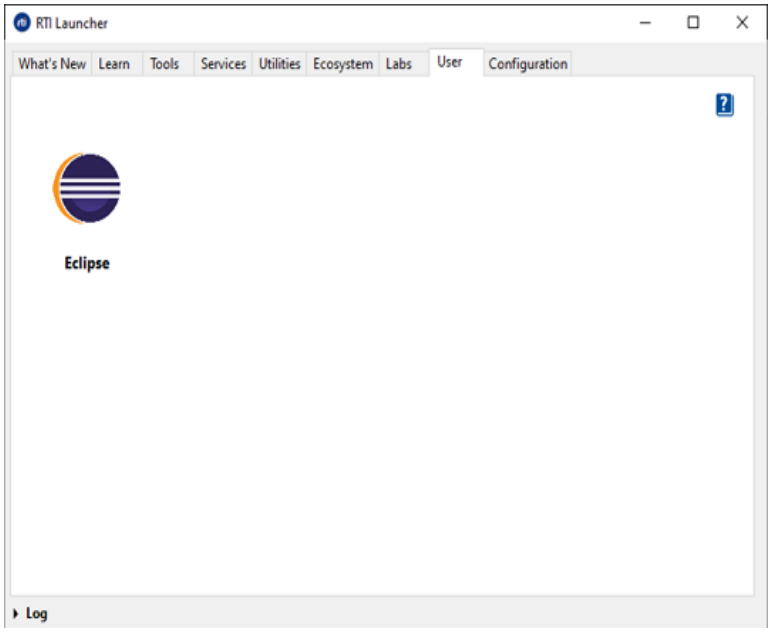
The buttons that *Launcher* displays (in any tab) must be in either of these locations:

- In the *Connext* installation directory, in **<NDDSHOME>/resource/app/app_support/launcher/buttons**. This folder is used for applications from RTI and RTI's partners (shown in the Ecosystem tab).
- In your RTI workspace directory, under each panel. For example, **rti_workspace/version/user_config/launcher/panels/user**.

3.4 Example: Adding a Button to Launcher's User Panel

The User tab is not visible the first time you run *Launcher*, because it's empty by default and *Launcher* doesn't show tabs that are empty. This tab is intended to be configured by the user.

This section shows how to add a button to the User tab's XML file so that it appears in *Launcher*. We'll use the Eclipse development environment as an example:



The User tab is defined in the **user** folder found in the *Launcher* configuration folder in the user home's RTI workspace. Inside this folder are two files, **configuration.xml** and **example.xml**.

The first time you open **example.xml**, it contains:

```
<?xml version="1.0" encoding="UTF-8"?>
<launcher xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:noNamespaceSchemaLocation="launcher_panel.xsd">
    <!-- Add a custom menu item here -->
</launcher>
```

You can change the name of the file **example.xml**; however, take into account that by default *Launcher* uses the name of the files to organize the buttons in the Tab alphabetically (each file represents a button). You can also reorganize the buttons using the configuration file as explained in 3.1 Launcher Tabs and Their Associated XML Files on page 26.

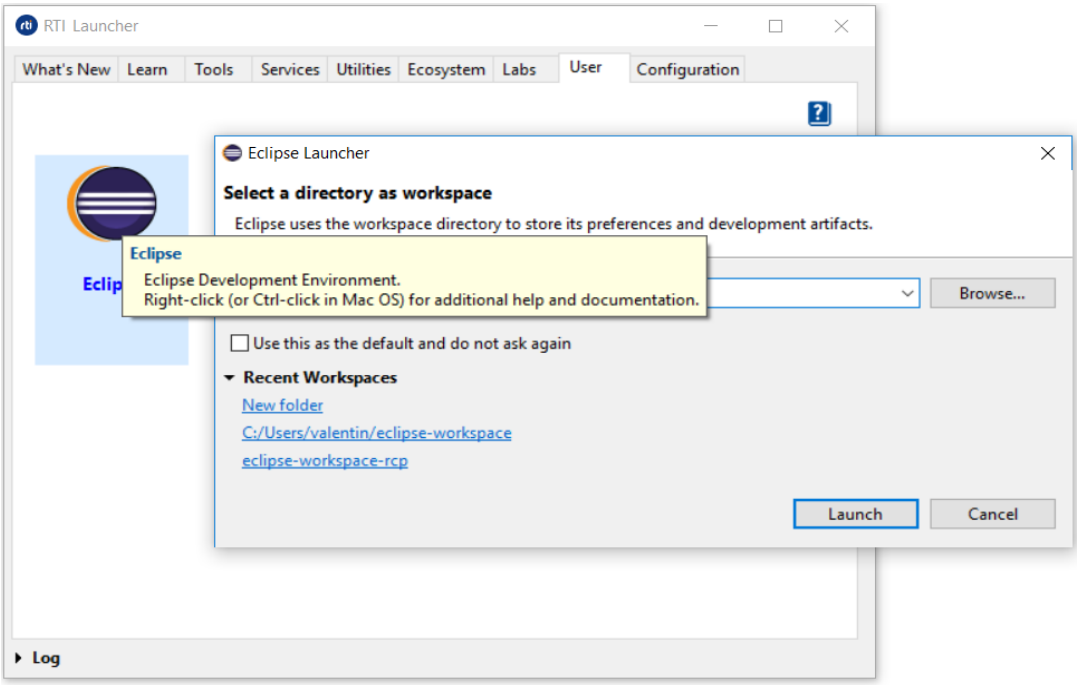
All the tab/panel configuration files start with the XML tag <launcher>. Let's add a button for running Eclipse to the user panel definition. Buttons are defined in a tag called <button>:

```
<?xml version="1.0" encoding="UTF-8"?>
<launcher xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:noNamespaceSchemaLocation="launcher_panel.xsd">
    <!-- Add a custom menu item here -->
    <button name="eclipse">
        <icon>eclipse256.png</icon>
        <tooltip>Eclipse:
            Eclipse Development Environment.
        </tooltip>
        <legend>Eclipse</legend>
        <toolexec>C:\Eclipse\eclipse.exe</toolexec>
        <popupmenu>
        </popupmenu>
    </button>
</launcher>
```

This is how the different tags are interpreted by *Launcher*:

- **'name' attribute**: Internal name of the button. No specific rule for it.
- **icon**: Path or name of the icon file that will be used to show the application in the panel. Preferred image file formats for *Launcher* are PNG and GIF (*Launcher* may not be able to correctly render the file if another format is provided, e.g., vectorial). It can contain an absolute path to the file, or it can refer to the name of the file, in which case the image file will be looked for in one of the image locations discussed in 3.2 How Launcher Finds Images on page 28. In our case, we copied the eclipse 256 logo file (eclipse256.png) into **rti_workspace/<*version*>/user_config/launcher/icons**. If the **icons** sub-directory doesn't exist, it can just be created normally.
- **tool-tip**: Required field. It is used to define a title for the application and a brief description. This information will be shown to the user as a balloon tool-tip when hovering over the icon. The format of this field must be: **{Title}:{Description}** (the ':' is necessary; *Launcher* may fail to create the panel if not found).
- **legend**: Title for the tool that will be shown right below the icon.
- **toolexec**: Path to the executable to run when the user clicks on the application icon.
- **popupmenu**: Required field, needs to be present, but it may be empty, like in our example. This tag is used to describe entries in the context menu that will be shown when the user right-clicks on the icon.

Once the **example.xml** file has been modified and saved, restart or start *Launcher* for the changes to take effect. You should be able to see the User tab now, and in it there will be a launch button for the Eclipse Development Environment.



Launcher's XML format for panels is described in the XSD file **launcher_panel.xsd**. This file is located in the same directory as the **user** folder (**rti_workspace/*version*/user_config/launcher/panels**). This file provides complete reference for defining buttons for *Launcher* in XML.

3.5 Example: Adding a Tab to Launcher

The User tab is not the only customizable tab in *Launcher*. To add a new tab to *Launcher*, create a folder in the following directory: **rti_workspace/*version*/user_config/launcher/panels**. Name the folder with the name you want to give the tab.

Launcher won't display the tab until you add at least one button, because *Launcher* doesn't show tabs that are empty. See 3.4 Example: Adding a Button to Launcher's User Panel on page 29.