

HCLSoftware

What's New in DevOps Model RealTime 12.1.1

updated for the DevOps 2024.12 release

Overview

- ▶ Model RealTime 12.1 is based on Eclipse 2024-06 (4.32)
- ▶ Two brandings of the product are available (HCL and IBM). There is no differences in functionality between them.
 - The only difference is in the licensing mechanism and branding (e.g. documentation)



DevOps Model RealTime
Version: 12.1.1 (part of DevOps 2024.12)
Build: 12.1.1.v20241210_1545

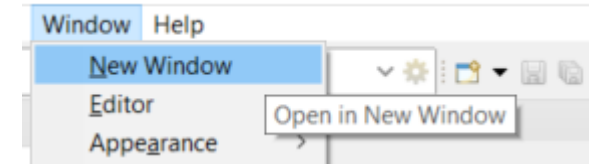
(c) Copyright IBM Corporation 2004, 2016. All rights reserved.
(c) Copyright HCL Technologies Ltd. 2016, 2024. All rights reserved.
Visit <https://model-realtime.hcldoc.com/help/topic/com.ibm.xttools.rsarte.webdoc/users-guide/overview.html>

Eclipse 4.32 (2024.06)

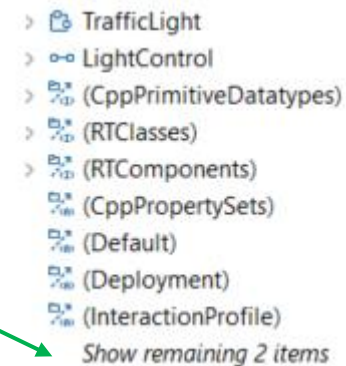
- ▶ Compared to Model RealTime 12.0, version 12.1 includes new features and bug fixes from 4 quarterly Eclipse releases:
 - 2023.09 (<https://www.eclipse.org/eclipse/news/4.29/platform.php>)
 - 2023.12 (<https://www.eclipse.org/eclipse/news/4.30/platform.php>)
 - 2024.03 (<https://www.eclipse.org/eclipse/news/4.31/platform.php>)
 - 2024.06 (<https://www.eclipse.org/eclipse/news/4.32/platform.php>)
- ▶ For full information about all improvements and changes in these Eclipse releases see the links above
 - Some highlights are listed in the next few slides...

Eclipse 4.32 (2024.06)

- ▶ Better support for opening additional workbench windows when using multiple monitors
 - Now a new workbench window is opened on the same monitor where the current workbench window is located
 - Previously it opened on the primary monitor, which could be "far away" from the current workbench window
- ▶ The number of items shown in many views can now be limited to improve UI performance
 - Controlled by a new preference **General - Initial maximum number of elements shown in views** (by default 1000)
 - Affects for example the Problems and the Project Explorer views
 - Helps avoiding performance problems when a huge number of elements are shown in these views
 - You can disable this feature by setting the preference to 0

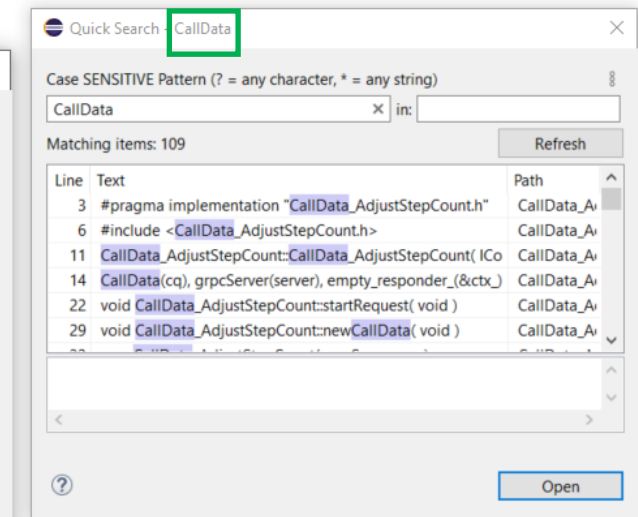
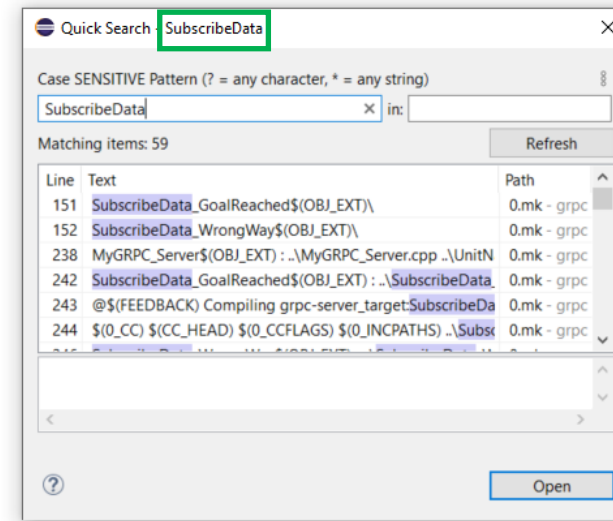
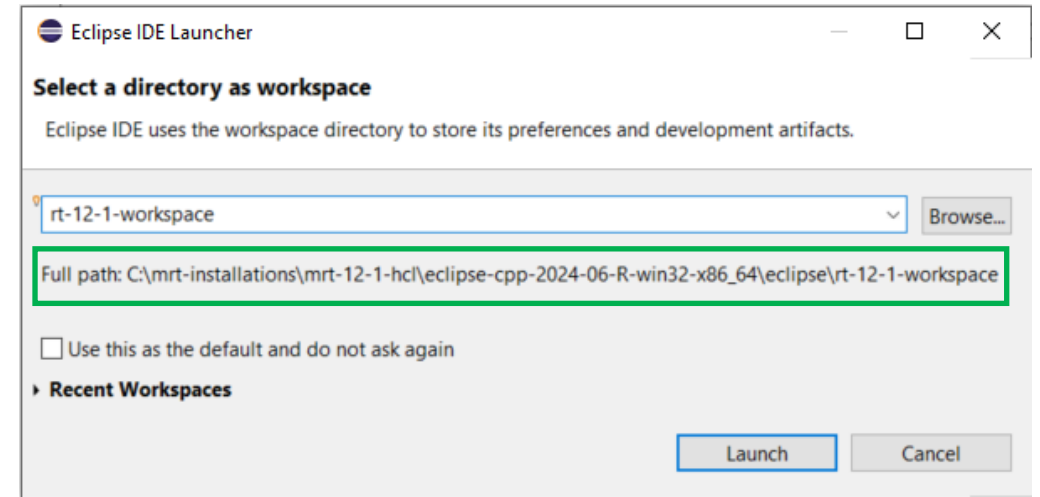


click here to show more items



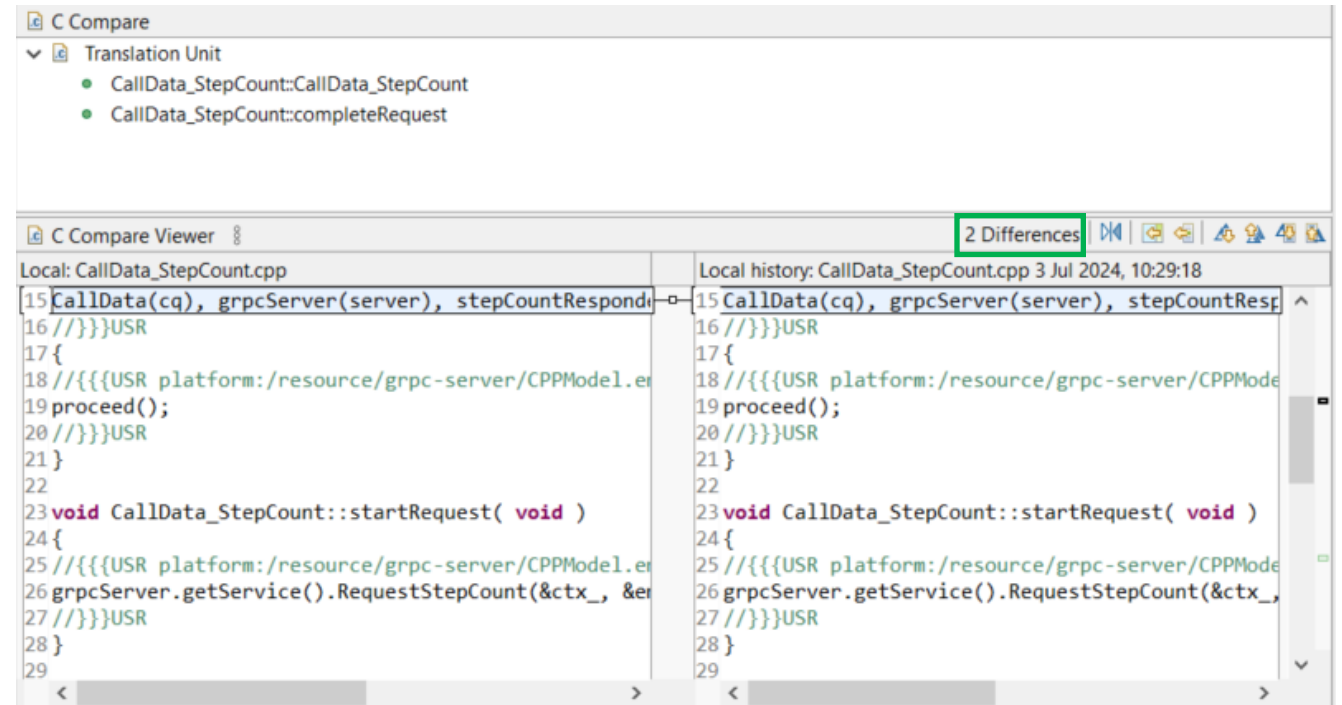
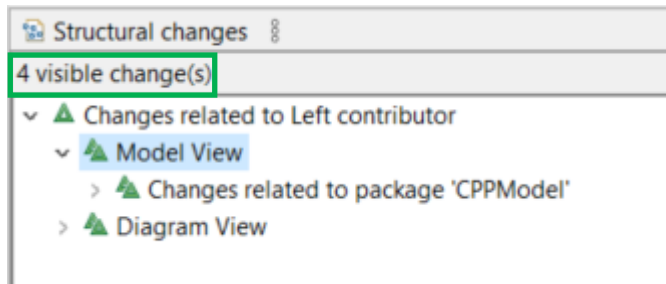
Eclipse 4.32 (2024.06)

- ▶ The launch dialog now shows the resolved path of the workspace that will be opened
 - Useful if you use relative paths or the ~ character on Linux/MacOs (for specifying the home folder)
 - Also tells you if you happen to use a character in the workspace name that is not valid on your operating system
- ▶ Multiple Quick Search dialogs can now be more easily distinguished since the search term is printed in their titles
 - Useful if you tend to keep a few of those modeless dialogs open for repeated searches

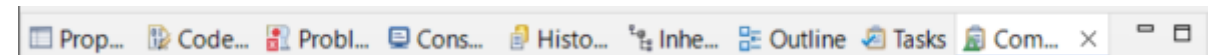


Eclipse 4.32 (2024.06)

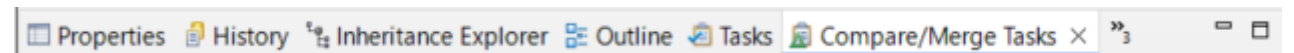
- ▶ The Compare/Merge editor for text files now shows the total number of differences
 - This was already before shown in the Compare/Merge editor for models



- ▶ A new preference can be set to avoid very short titles when a large number of views appear in the same view stack
 - **General - Appearance - Always show full titles**
 - Another new preference allows to hide the view icons to further save space



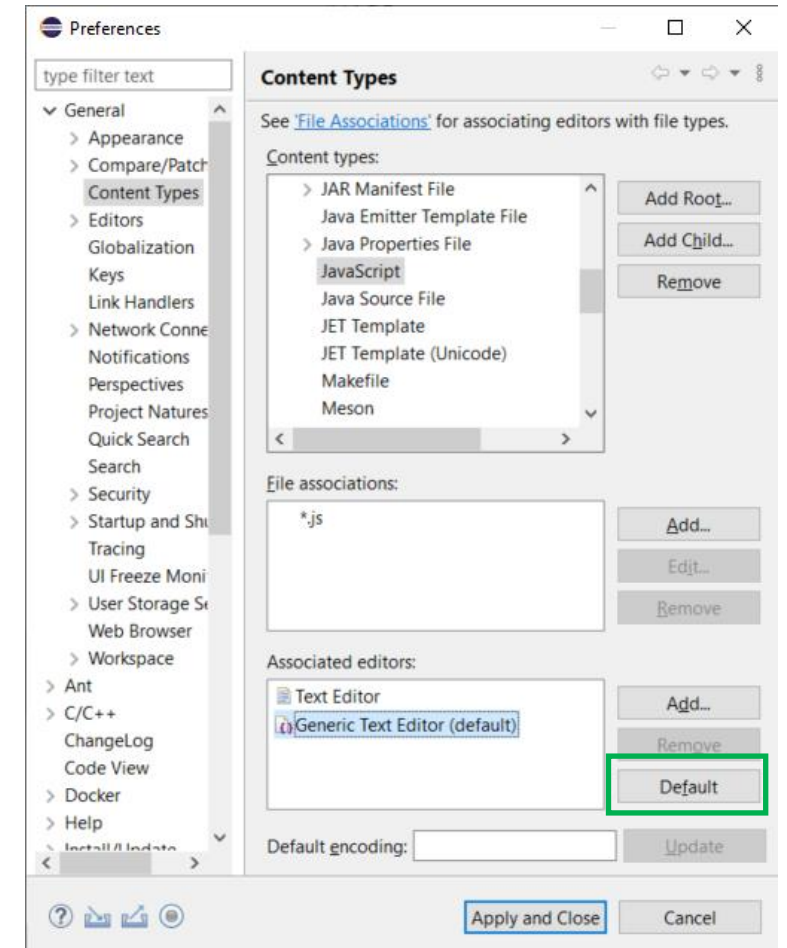
before (and still the default)



now (with new preference set)

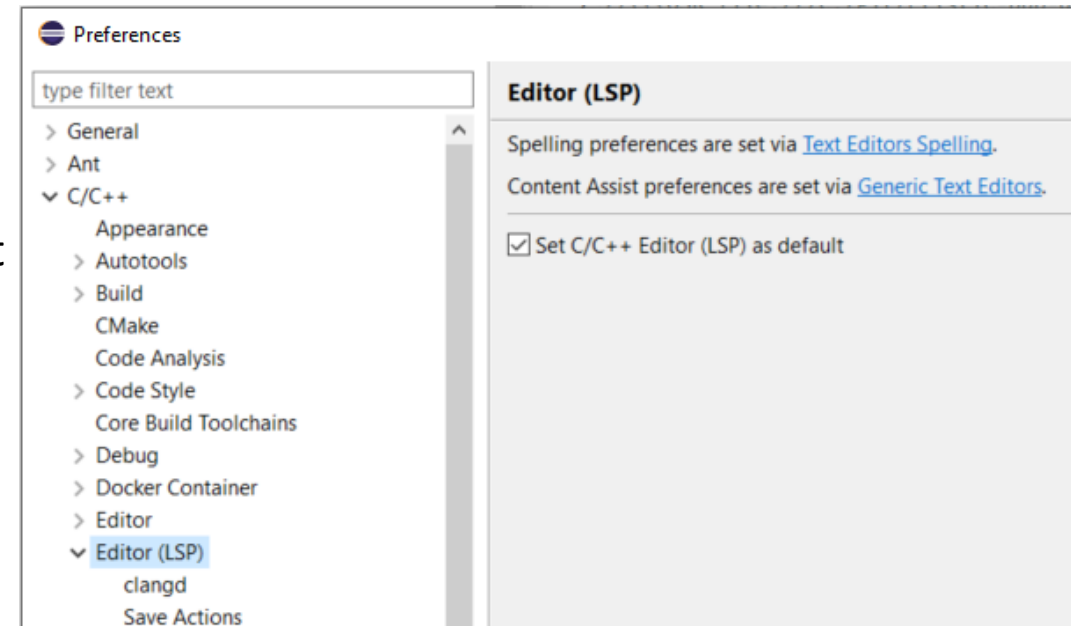
Eclipse 4.32 (2024.06)

- ▶ It's now possible to change the default editor for a certain file content type
 - For example, if you use Build Variants, but don't have a JavaScript plugin installed, you can now create a content type for the *.js file extension and set the Generic Text Editor as the default editor
 - Previously it was necessary to always use the **Open With** command which was easy to forget
- ▶ Restart of Eclipse now works the same as an exit followed by a relaunch
 - For example, changes in `eclipse.ini` are now picked up when restarting



CDT 11.6 (included as part of Eclipse 2024.06)

- ▶ Information about the CDT improvements can be found here:
<https://github.com/eclipse-cdt/cdt/blob/main/NewAndNoteworthy/CDT-11.3.md>
<https://github.com/eclipse-cdt/cdt/blob/main/NewAndNoteworthy/CDT-11.4.md>
<https://github.com/eclipse-cdt/cdt/blob/main/NewAndNoteworthy/CDT-11.5.md>
<https://github.com/eclipse-cdt/cdt/blob/main/NewAndNoteworthy/CDT-11.6.md>
- ▶ **Note:** Eclipse 2024.06 also contains another plugin for C/C++ development called [CDT-LSP](#) which makes use of the Clang language server (clangd)
 - You have to enable this feature in the Preferences - by default CDT is still used for new projects
 - You can use both CDT and CDT-LSP in the same workspace for different C++ projects
 - Model RealTime does not yet support CDT-LSP (i.e. target projects are still always using CDT, and the Code View/Editor also uses CDT)



Newer EGit Version in the EGit Integration

- ▶ The EGit integration in Model RealTime has upgraded EGit from 6.6 to 6.10
 - This is the recommended and latest version for Eclipse 2024.06
- ▶ This upgrade provides several new features and bug fixes
 - For detailed information about the changes see
 - https://wiki.eclipse.org/EGit/New_and_Noteworthy/6.7
 - https://wiki.eclipse.org/EGit/New_and_Noteworthy/6.8
 - <https://projects.eclipse.org/projects/technology.egit/releases/6.9.0>
 - <https://projects.eclipse.org/projects/technology.egit/releases/6.10.0>

Java 21

- ▶ Eclipse 2024.06 requires a JVM for Java 21 or newer
 - It comes with a Java 21 JVM included which will be used by default
 - Model RealTime therefore also requires Java 21 and it's recommended to use that exact version. A warning dialog will appear on start-up in case another JVM version is used.
 - This check can be disabled by setting the system property `com.ibm.xtools.umltdt.core.disableJavaCheck`

Removal of the One Test Embedded Integration

- ▶ The One Test Embedded Integration does not work with Eclipse 2024.06 and has therefore been removed from Model RealTime
 - The [documentation on the Info Center](#) will remain for some time for users still using One Test Embedded together with older versions of Model RealTime

TC Editor Improvements

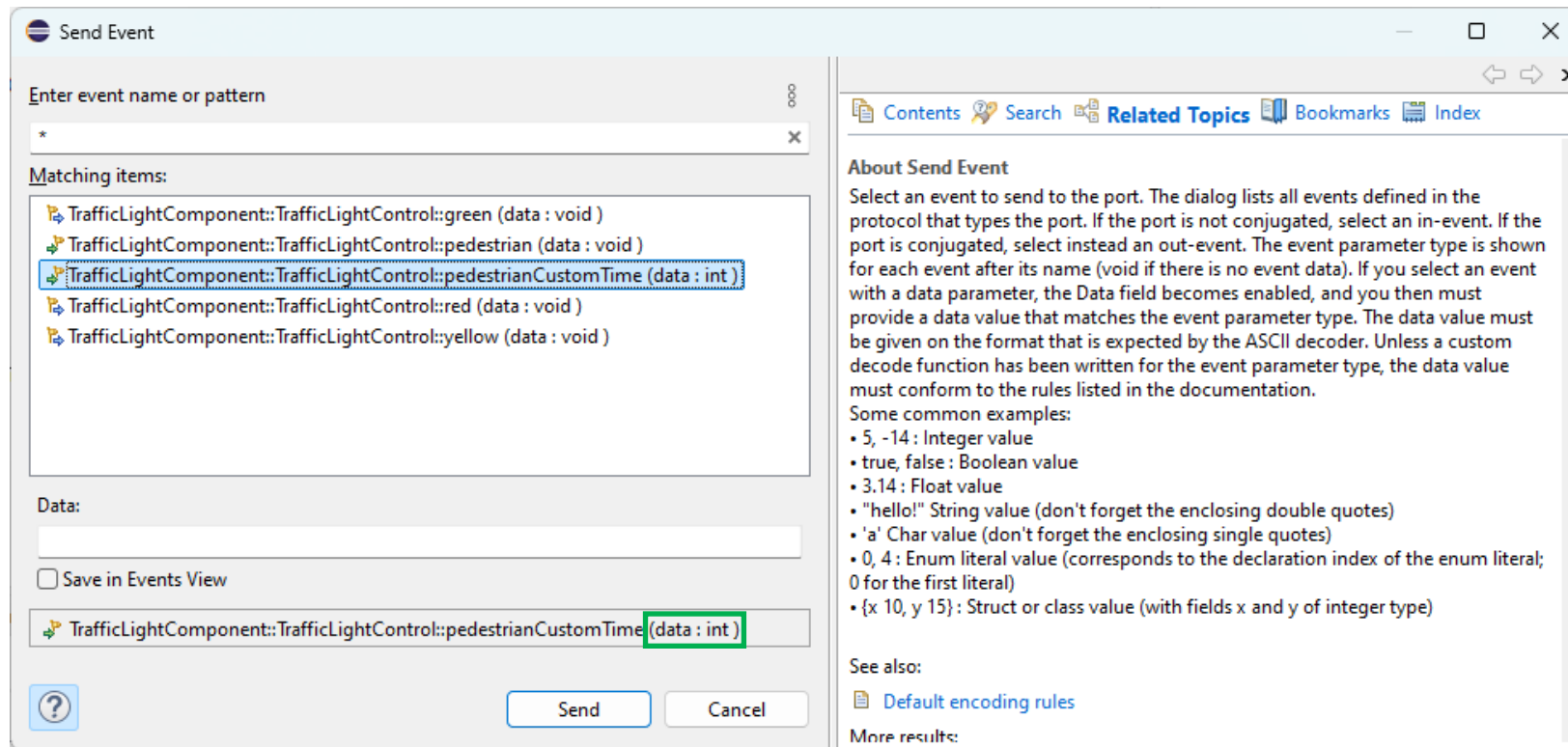
- ▶ The Sources table now has columns to more easily manage the list of source elements
 - Possible to sort the source elements by clicking the column headers
 - The Qualified Name and Resource columns help distinguish multiple source elements with the same name

Sources

Qualified Name	Name	Resource
 CPPModel::Client	Client	platform:/resource/rtBound/CPPModel.emx#_Won7...
 CPPModel::PROT::PROT	PROT	platform:/resource/rtBound/CPPModel.emx#_xLyT...
 CPPModel::Server	Server	platform:/resource/rtBound/CPPModel.emx#_vN8tk...
 CPPModel::Top	Top	platform:/resource/rtBound/CPPModel.emx#_VJr4c...

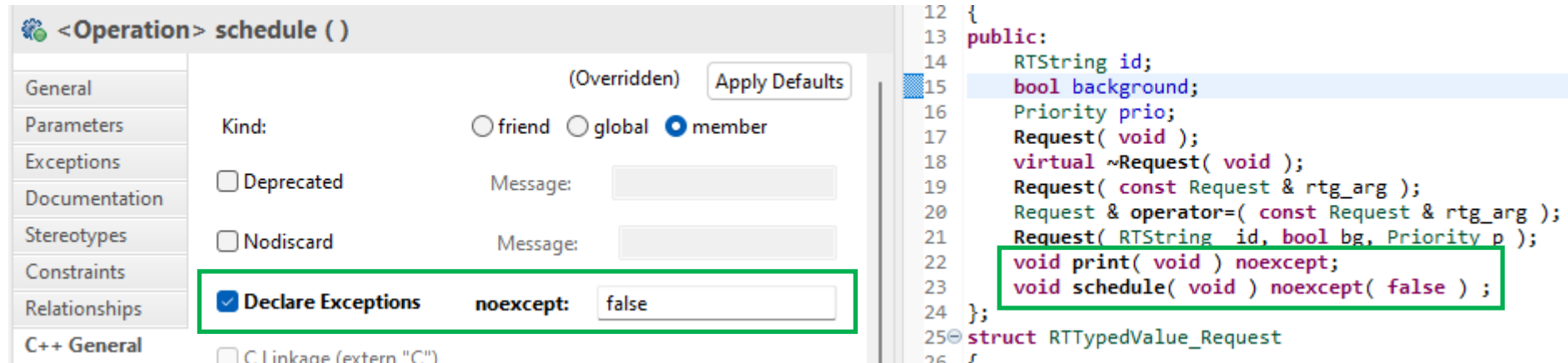
Improved Send Event Dialog

- ▶ The Send Event dialog now shows the parameter type for each listed event
 - Makes it easier to know what to type in the Data field when sending an event
- ▶ Context Sensitive Help is now also available for this dialog



Generating Functions with “noexcept” Specifiers

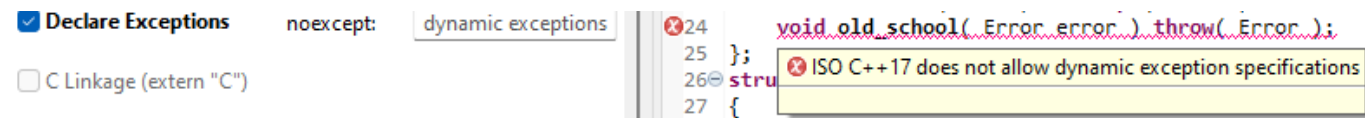
- ▶ It's now possible to specify that an operation cannot throw any exceptions, or may throw exceptions, by means of the noexcept specifier



```
12 {
13 public:
14     RTString id;
15     bool background;
16     Priority prio;
17     Request( void );
18     virtual ~Request( void );
19     Request( const Request & rtg_arg );
20     Request & operator=( const Request & rtg_arg );
21     Request( RTString id, bool bg, Priority p );
22     void print( void ) noexcept;
23     void schedule( void ) noexcept( false );
24 };
25 struct RTTypedValue_Request
26 }
```

- ▶ For backwards compatibility, dynamic exception specifications are still supported

- But this feature was removed in C++ 17, and the Model Compiler will therefore print a warning if you attempt to use it with a too new code standard



```
24 void old_school( Error error ) throw( Error );
25 };
26 struct
27 {
```

WARNING : Dynamic exception specifications should not be used with C++ 17 or later. Consider using the noexcept specifier instead.

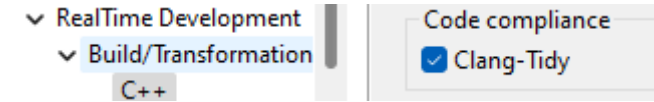
- ▶ Use of noexcept requires C++ 11 or later

- The Model Compiler will print a warning if you attempt to use it with a too old code standard

WARNING : Use of the noexcept specifier requires C++ 11 or later.

Code Compliance

- ▶ An additional Clang-Tidy rule is now supported when the preference **RealTime Development – Build/Transformations – C++ - Clang-Tidy** is set



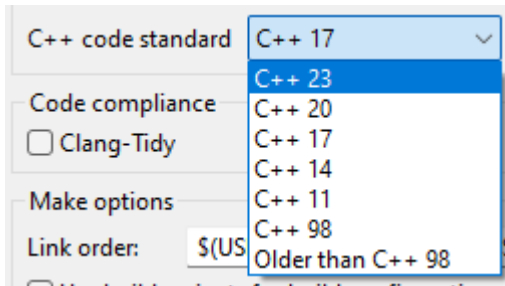
- **misc-use-anonymous-namespace**

Suppress warnings for use of the `static` keyword for non-member functions in generated .cpp files

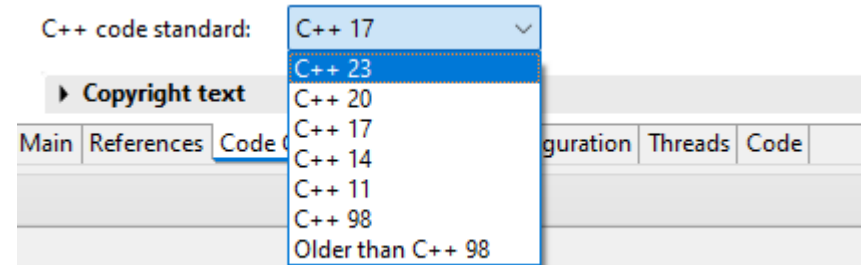
```
static void _rtg_deleteThreads( void ) /* NOLINT(misc-use-anonymous-namespace) */
```

C++ 23

- ▶ C++ 23 is now available as a new code standard to use
 - For both the TC setting and the workspace preference
 - Currently no features in Model RealTime require this new code standard, but if generated code is compiled with C++ 23, it's useful to be able to set this as the code standard to be used for code generation too (for consistency)



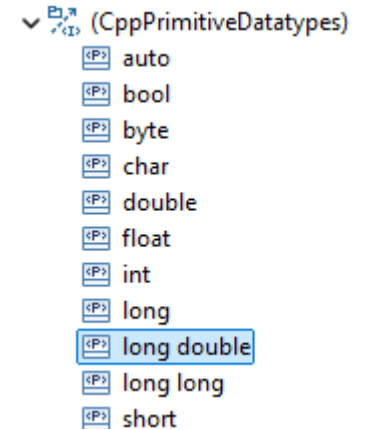
workspace preference



TC setting

Long Double

- ▶ The TargetRTS and Model Compiler now supports the primitive type `long double`
 - It was already available to be used in the `CppPredefined` package, but was previously ignored by the code generator (and not supported by the TargetRTS)
 - The Encoding, Decoding and Logging APIs were extended to support `long double` in the same way as other primitive types are supported

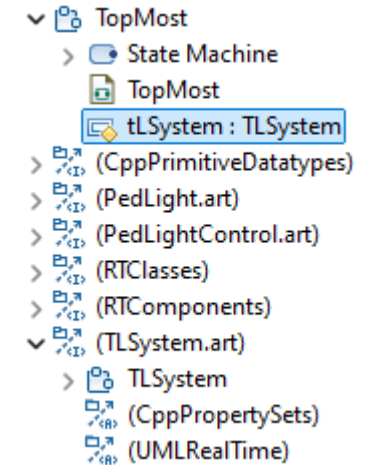


Changed Encoding of Some Primitive Types

- ▶ Primitive types that has a space in their name (`long long`, `long double` etc) are now encoded so that the space is replaced with an underscore (`_`)
 - For example, the value 4 of type `long long` was previously encoded as “4 long long” but is now instead encoded as “4 long_long”
- ▶ This change was done to ensure that encoded strings only contain one space that separates the value from the type name
 - Previously it was not possible to send events in the Model Debugger where the event parameter had a primitive type with a space in its name, but with this change it’s now working correctly
- ▶ Note: If your application somehow relies on the old encoding for these types it must be updated to accommodate for this change!

Support for Art Files (Code RealTime Integration)

- ▶ A larger subset of the Art language is now supported
 - Use Art protocols (as type of UML-RT ports, or to inherit UML-RT protocols from)
 - Art events with user-defined parameter types can now be used
 - Use Art capsules as type of UML-RT parts. Connect ports of the Art capsule to UML-RT ports.
- ▶ The Art Compiler is now automatically invoked from generated make file
 - No longer necessary to manually build in Code RealTime if Art or TC files are changed
 - Requires Model RealTime to know where Code RealTime is installed (by means of a new string substitution variable ART_COMPILER)

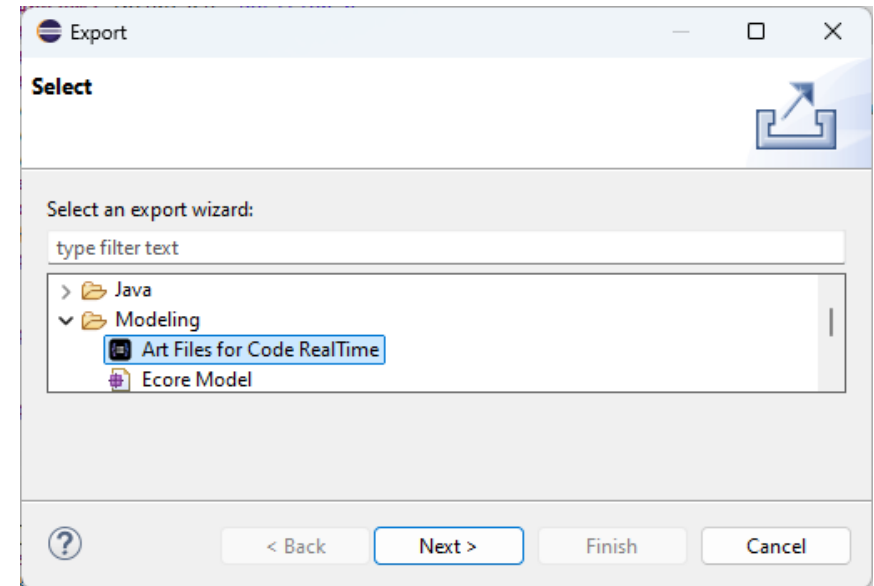


String Substitution			
Create and configure string substitution variables.			
Variable	Value	Description	Contributed By
ART_COMPILER	C:\codert\build\vscode-codert-202-ms\data\extensions\s...	Absolute path to the art compiler jar file 'artcompiler.jar'	com.ibm.xtool...
PCA_RT_HOME [read-only]	C:\model-it-installations\12.1.1-241017\collins\res...		com.ibm.xtool...

- ▶ This feature is no longer experimental!

Art Exporter

- ▶ A new utility for exporting (parts of) your models to Art files for Code RealTime
- ▶ Available as a separate update site to be installed on top of Model RealTime
 - Get it from the [Utilities page on the Info Center](#)
- ▶ The first version supports export of data types only
 - Useful for reusing data types developed with Model RealTime in a Code RealTime application
- ▶ Note: You can expect new releases of the Art Exporter to be delivered frequently during 2025 (more frequently than new versions of Model RealTime)
 - See the documentation for detailed release notes



HCLSoftware

hcltechsw.com