

# What's New in HCL RTist 11.1

updated for release 2021.24



Copyright © 2021 HCL Technologies Limited | www.hcltechsw.com

#### Overview

RTist 11.1 is based on Eclipse 2020.06 (4.16)

- HCL RTist is 100% compatible with IBM RSARTE. All features in IBM RSARTE are also present in HCL RTist. However, HCL RTist contains a few features that do not exist in IBM RSARTE.
  - Those features are marked in this presentation by



HCL RTist Version: 11.1.0.v20210618\_1548 Release: 2021.24

(c) Copyright IBM Corporation 2004, 2016. All rights reserved.
(c) Copyright HCL Technologies Ltd. 2016, 2021. All rights reserved.
Visit https://RTist.hcldoc.com/help/topic/com.ibm.xtools.rsarte.webdoc/users-guide/overview.html





- Compared to RTist 11.0, RTist 11.1 includes new features from 4 quarterly Eclipse releases:
  - 2019.09 (<u>https://www.eclipse.org/eclipse/news/4.13/platform.php</u>)
  - 2019.12 (<u>https://www.eclipse.org/eclipse/news/4.14/platform.php</u>)
  - 2020.03 (<u>https://www.eclipse.org/eclipse/news/4.15/platform.php</u>)
  - 2020.06 (<u>https://www.eclipse.org/eclipse/news/4.16/platform.php</u>)
- 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...



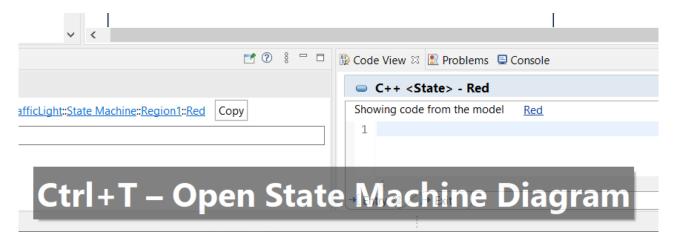
- A new Quick Search dialog allows you to search the files of your workspace faster ("as-you-type")
  - For a similar search experience in model files, use the Find Named Element command instead

	).send	× in: *.cpp
atchi	ng items:	<u>R</u> efresh
ine	Text	Path
154	pedLightControl.walk().send();	TrafficLight.cpp - TrafficLightsTCPAPI_target
51	server. <mark>walk().send(</mark> );	PedLight.cpp - TrafficLightsTCPAPI_target
159	pedLightControl.walk().send();	TrafficLight.cpp - TrafficLightsDemo_target
52	server.walk().send();	PedLight.cpp - TrafficLightsDemo_target
177	pedLightControl.walk().send();	TrafficLight.cpp - TrafficLightsDemoPi_target
52	server. <mark>walk().send(</mark> );	PedLight.cpp - TrafficLightsDemoPi_target
	<pre>f((bool*) rtdata) == true) { pedLightControl.walk().send();</pre>	

- By default at most 99 editors can now be open at the same time
  - Helps keeping the performance good when working with Eclipse for a long time
  - This can be controlled by the preference General Editors Close editors automatically
- Showing key bindings when performing commands
  - New preferences in General Keys

Show key binding when command is invoked
Through keyboard
Through mouse click

This is a good way to learn about key bindings for the commands that are used, and can also help in presentations





# Eclipse 4.16 (2020.06)

- Quick Access field replaced with toolbar button
  - Takes less space in the toolbar, and instead uses a normal dialog for typing an showing the results
  - Same key binding as before (Ctrl + 3) but the command is now called "Find Actions"
  - The results now also include matching files in the workspace, and text matches in files (requires that Quick Search) has been used at least once)

ann teil		
app.tcj		*
File content	`# D:\rsarte'	\demos\CompleteTrafficLightDemo09062019\Traff
	`# Generate	d from D:\rsarte\demos\CompleteTrafficLightDem
Resources	🖏 app.tcjs	
Help	🚿 Search ' <b>app</b>	.tcj' in Help

walk()	
Commands	Preferences (Java > Debug > Heap Walking) - Open the preferences
Preferences   Heap Walking - Java/Debug	
File content	`pedLightControl.walk().send();` in TrafficLight.cpp
	`server.walk().send();` in PedLight.cpp
Help	😵 Search ' <b>walk()</b> ' in Help





6

- Show code problems inline
  - Makes errors/warnings more visible and lets you apply quick fixes without having to go to the Problems view
  - Enable this feature in preferences at General Editors Text Editors –
     Show code minings for problem annotations
- ▶ There were several improvements in SWT and GTK
  - The minimal supported GTK version is now 3.20

- New Debug Sources view
  - Shows source files the C++ debugger knows about when debugging an application
  - Useful in particular when the application contains source files that are not present in the Eclipse workspace
  - Source files can be found by searching (filtering) and opened by double-click

🎋 Debug 🛛 🎦 Project Explorer 👘 🗖	Debug Sources & TrafficLight::Sta	📄 info.txt 🛛 📲 app.tcjs 💽 main() at 0x435 👘 🗖
🖻 🔆 it is		E E E 🗎 🕸
<ul> <li>executable.EXE [C/C++ Application]</li> <li></li></ul>	type filter text	
✓	Name	Path ^
main() at 0x435ec3	RTBindingDescriptor.h	D:\rtist11_1\eclipse\rsa_rt\C++\TargetRTS\inclu
> 🕐 Thread #2 0 (Suspended : Container)	RTActorRef.h	D:\rtist11_1\eclipse\rsa_rt\C++\TargetRTS\inclu
> 🕐 Thread #3 0 (Suspended : Container)	RTActorClass.h	D:\rtist11_1\eclipse\rsa_rt\C++\TargetRTS\inclu
> 🕐 Thread #4 0 (Suspended : Container)	RTActor.inl	D:\rtist11_1\eclipse\rsa_rt\C++\TargetRTS\inclu
> 🧬 Thread #5 0 (Suspended : Container)	<ul> <li>eclipse-workspace\rtist-11-1-wksp2\TL_rtist_111_target</li> </ul>	
📕 gdb (8.1)	UnitName.cpp	D:\eclipse-workspace\rtist-11-1-wksp2\TL_rtist_
_	TrafficLight.cpp	D:\eclipse-workspace\rtist-11-1-wksp2\TL_rtist_
	LightControl.h	D:\eclipse-workspace\rtist-11-1-wksp2\TL_rtist_
	LightControl.con	D\aclinco.workenaco\rtict 11.1.wken2\TL_rtict

- CODAN improvements
  - Several additional checks implemented
- For more information about CDT improvements see <u>https://wiki.eclipse.org/CDT/User/NewIn99</u> <u>https://wiki.eclipse.org/CDT/User/NewIn910</u> <u>https://wiki.eclipse.org/CDT/User/NewIn911</u>

## Newer EGit Version in the EGit Integration

- ▶ The EGit integration in RTist has upgraded EGit from 5.4 to 5.8
  - This is the recommended and latest version for Eclipse 2020.06
- This upgrade provides several new features, performance improvements and bug fixes
  - For detailed information about the changes see <u>https://wiki.eclipse.org/EGit/New and Noteworthy/5.5</u> <u>https://wiki.eclipse.org/EGit/New and Noteworthy/5.6</u> <u>https://wiki.eclipse.org/EGit/New and Noteworthy/5.7</u> <u>https://wiki.eclipse.org/EGit/New and Noteworthy/5.8</u>

- A bash script is now available which helps automating the installation of RTist
  - Download it from the <u>Info Center</u>
  - Works on both Windows and Linux
- ▶ In particular useful for installing RTist 11.1 (due to the requirement of using Java 11 for the installation)

- Choose whether you want to then run RTist with either Java 8 or Java 11
- ▶ For documentation on how to configure and use the script see the <u>Info Center</u>.

## **Properties View Improvements**

- The Default Value field now supports multi-line values
  - To create a multi-line default value you still need to use the Code View or Code Editor
  - For editing a multi-line default value you can now use the Properties view, but it's still often more convenient with the Code View or Code Editor

	a : int [3] = { HelloWorld				~	<		
🔳 *Properties 🛛				8 🕐 🗖	🚯 Code	e View ⊠	🛷 Search   🟦 Pro	blem
🖷 <attribute></attribute>	a					C++ <a< th=""><th>ttribute&gt; - a</th><th></th></a<>	ttribute> - a	
General	Qualified Name:	: <u>HelloWorld</u> :: <u>HelloWo</u>	orld::a Copy	]			ce is not found. You from the model	
Documentation	Name: a					ving code {	e nom the model	<u>a</u>
Stereotypes	Visibility: OF	Public   Private	Protected		2	ι		
Constraints					3	2,		
Relationships	Modifiers:	Static Const C	ConstExpr		4	-		
C++ General	Туре:	int	Select Type	. Open Type		j		
C++ Target RTS	Default Value:	2,		a.				
Advanced	Multiplicity	Z, Use Up-Down	arrow keys to	view other lines				

For quickly viewing a multi-line default value the Properties view can be handy



## Search Filtering

- ▶ It's now possible to filter search results using Boolean operators NOT (!) and AND (&&)
  - Useful if a search returns too many matches
  - Use a filter on the form

!A && !B && ... !X to hide matches where certain words <u>are</u> <u>not</u> present

- Use a filter on the form
   A && B && ... X to only show matches where certain words <u>are</u> present
- ...or any combination, where some words are present and others not
- Enclose the filter string in double quotes to apply the filter verbatimly
  - Needed if the filter string contains the characters ! or &&

	🔗 Search 🛛 🔀 Code View	🖹 Problems 📮 Console 🔫	Progress		
ļ	'results' (verbatim, case insensitive): Showing 5 matches in Workspace.				
	Match = !are && !encode		Kind 🚔		
	cout << "No <b>results</b> four	id!";	Code (Entry of State)		
	→ cout << "Search result	s:" << results.c_str() << endl;	Code (Effect of Tra		
	→ cout << "Search results	:" << <b>results</b> .c_str() << endl;	Code (Effect of Tra		
	cout << "Results found "	<< results.c_str() << endl;	Code (Entry of State)		
	cout << "Results found "	<< results.c_str() << endl;	Code (Entry of State)		

🔗 Search 🛛 🔀 Code View 🖹 Prob	lems 🗏 Console 🖷 Progress				
'results' (verbatim, case insensitive): S	'results' (verbatim, case insensitive): Showing 2 matches in Workspace.				
Match = "!" Kind					
cout << "No results found!";	Code (Entry of State)				
msg = "Results warning!";	Code (Exit of State)				

#### **L7** HCL SOFTWARE

## Generic Type Descriptors

- The model compiler now supports generating type descriptors for type aliases with template parameters
  - For example: template<typename T, unsigned int N > using StdArray = std::array<T, N>;
  - If type descriptor functions are defined for the type alias, they will be generated as template functions with the same template parameters
  - Allows to implement generic type descriptors that work for all (or many) instantiations of the template
  - A new RTObject\_class::fromType<T>() template function can be used for looking up the type descriptor of a type at compile time. Useful for example when implementing generic encode or decode functions. Specialize it for the types that you use (specializations for built-in types are available in the TargetRTS). For example:
     template <> inline const RTObject\_class\* RTObject\_class::fromType<RTString>() {
     return &RTType\_RTString;
     }
- You can specify a unique name for the type descriptor of a specific template instantiation
  - For example: template <> const char\* RTName\_StdArray<StdString, 4>::name = "StdArray<StdString, 4>";
  - The TargetRTS now prints a warning if two type descriptors with the same name exists. Helps troubleshooting missing template specializations for the name attribute.

14

# **Code Compliance**

- A new group of preferences were introduced to let the model compiler generate code according to certain code compliance rules
  - As a first step support for one specific Clang-Tidy rule is implemented
  - It suppresses warnings for use of static\_cast to downcast event data in transition functions

Preferences	
type filter text	C++
<ul> <li>&gt; Oomph</li> <li>&gt; Plug-in Developm</li> <li>&gt; RealTime Develop</li> <li>&gt; Build/Transform</li> <li>C++</li> <li>Code Editing</li> </ul>	C++ code standard C++ 11 Code compliance Clang-Tidy Make options
< > ? h 14 0	<

```
transition2_t1( static_cast< const bool * > ( msg->data ), static_cast< P::Base * > ( msg->sap()
/* NOLINT(cppcoreguidelines-pro-type-static-cast-downcast) */_);
```

# Unit Testing of Capsules using the Mocha Framework (1/3)



HCL SOFTWARE

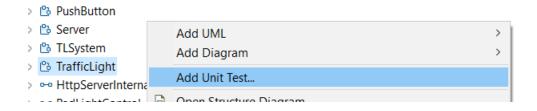
- Mocha is a popular JavaScript framework for testing asynchronous applications
- It's now possible to use Mocha also for unit testing capsules
  - Provided by a new component that can be selected when installing
  - Note that it depends on NodePlus and is currently an EXPERIMENTAL feature



> Image: Window W

11.1.0.v20210611\_1932

• To create a Mocha unit test for a capsule, invoke the new context menu command Add Unit Test





# Unit Testing of Capsules using the Mocha Framework (2/3)



- The Add Unit Test command creates everything necessary for writing a unit test for the capsule
  - A test driver model where all service ports of the capsule under test ("cut") are connected to similar but conjugated ports of a test probe capsule
  - A TC for building the test driver model into an executable that uses the TcpServer library for exposing all test probe ports to the Mocha test script
  - A Node.js project with a Mocha test script ready to implement the unit test

) ed	<ul> <li>test_TrafficLight</li> <li>Main</li> <li>B TestContainer</li> <li>TestProbe</li> </ul>	
contr test_c cont	pedLightControl~ ontrol~ test_pedLightControl	server test_server~

- Iransformation Configurations
- > 🗟 «C++ Executable» app\_win
- > Name and the second secon
- - > 🛋 JavaScript Resources
- 🗸 🗁 test
  - tests\_TrafficLight.js
- D package.json

Add Unit Test - D X						
Enter Test Case	nter Test Case Details					
Test Case						
Name	tests_TrafficLight					
Description	should initially be in Red state				$\langle \rangle$	
Test Driver						
Package Name	test_TrafficLight					
Port	2234					
Transformation C	Configuration					
Name	TrafficLight_UnitTests					
Output Path	TrafficLight_UnitTests_target					
Test Application						
Host	localhost					
Port	9911					
?		<u>F</u> inish		Cance	I	

#### **L**7 HCL SOFTWARE

# Unit Testing of Capsules using the Mocha Framework (3/3)



HCL SOFTWARE

- The unit test can be executed right away
  - Build the test driver TC (only needed the first time, and whenever you change the capsule under test)
  - Install the Node.js dependencies for the JavaScript project (right-click on the project and do Run As – npm install (only needed the first time – it is assume you already have installed Mocha on the machine)
  - Run the testcase by right-click on the .js file and do
     Run As JavaScript Unit Test

🖪 te	sts_TrafficLight.js ⊠	TestContainer TrafficLight_UnitTests.tcjs				
1	<pre>1 var assert = require('assert');</pre>					
2	<pre>2 describe('TrafficLight', function() {</pre>					
3	<pre>it('should initially be in Red state', function() {</pre>					
4	<pre>this.timeout(15000);</pre>					
5	const	<pre>testProbe = require ('rt-test-probe')('localhost', 9911);</pre>				
6	<pre>return testProbe.startListenForEvents(2234)</pre>					
7	.then(	(data) => {				
8	// TODO: Implement test here					
9	})					
10	final	<b>ly(()</b> => {				
11	te	<pre>stProbe.stopListenForEvents();</pre>				
12	});					
13	});					
14	});					

The test execution result is shown in the JavaScript Unit Test view

🏟 JavaScript Unit Test 🕺 B Code View 🔐 Problems   😑 Console   1 Error Log 🖉 Terminal			
Runs: 1/1	Errors: 0	Failures: 0	
✓ IrafficLight			Failure Trace



Relationary BEYOND THE CONTRACT

**\$7** BILLION ENTERPRISE | **110,000** IDEAPRENEURS | **31** COUNTRIES

