What's New in HCL RTist 11
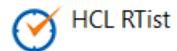
updated for release 2021.10

Overview

▸ RTist 11 is based on Eclipse 2019.06 (4.12)

▸ 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: 11.0.0.v20210310_2306
Release: 2021.10

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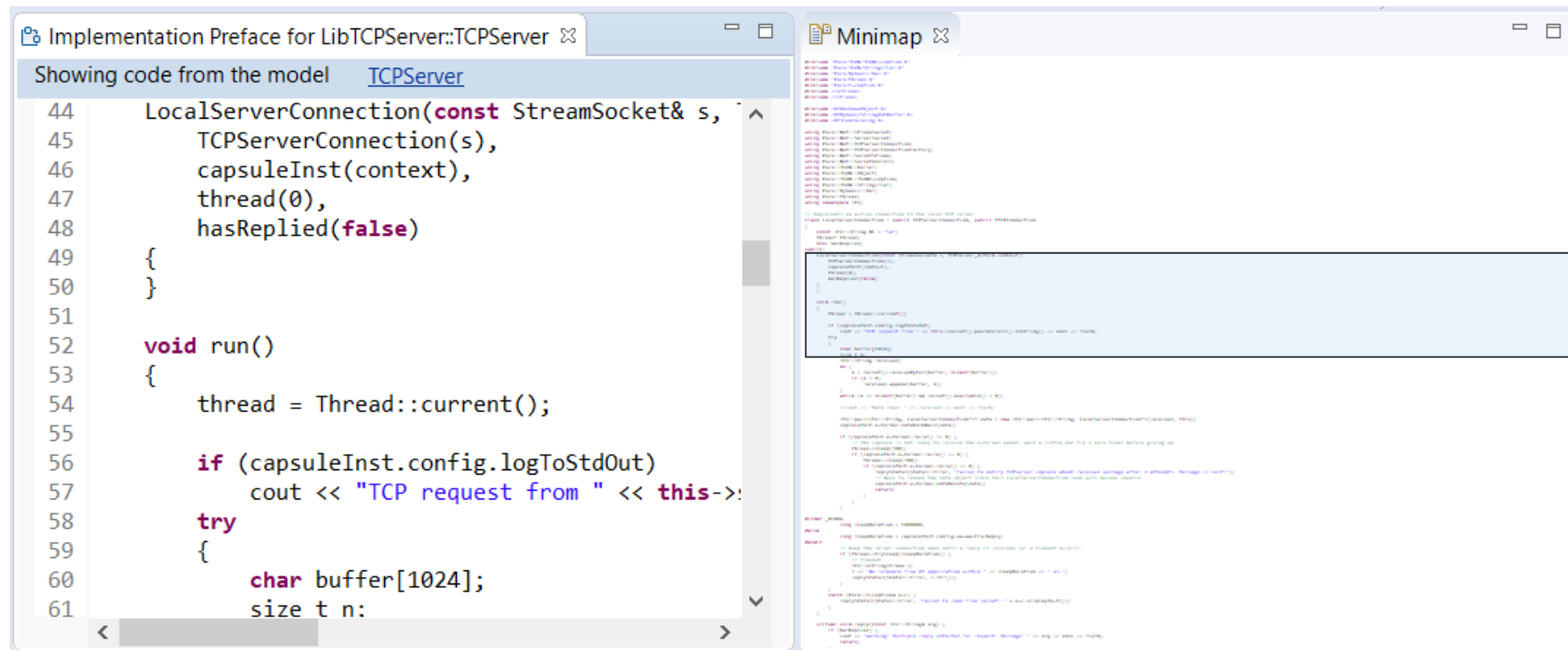(c) Copyright HCL Technologies Ltd. 2016, 2021. All rights reserved.
Visit https://RTist.hcldoc.com/help/topic/com.ibm.xtools.rsarte.webdoc/users-guide/overview.html

HCL SOFTWARE

Eclipse 4.12 (2019.06)

▶ Compared to RTist 10.3, RTist 11 includes new features from 4 quarterly Eclipse releases:

- 2018.09 (https://www.eclipse.org/eclipse/news/4.9/platform.php)

- 2018.12 (https://www.eclipse.org/eclipse/news/4.10/platform.php)

- 2019.03 (https://www.eclipse.org/eclipse/news/4.11/platform.php)

- 2019.06 (https://www.eclipse.org/eclipse/news/4.12/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…

HCL SOFTWARE

Eclipse 4.12 (2019.06)

▸ A new Minimap view gives a better overview of the text editor contents (useful when it's large)

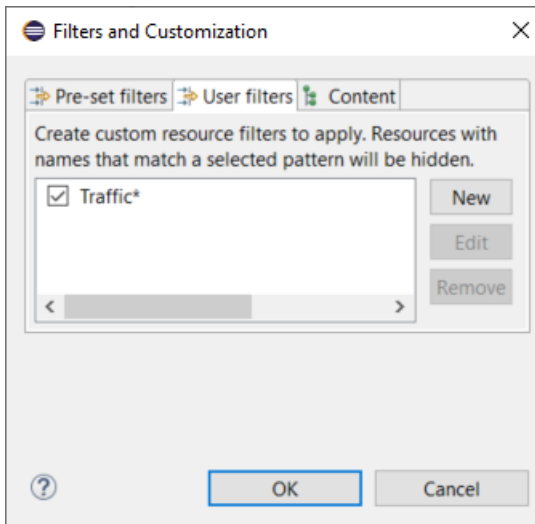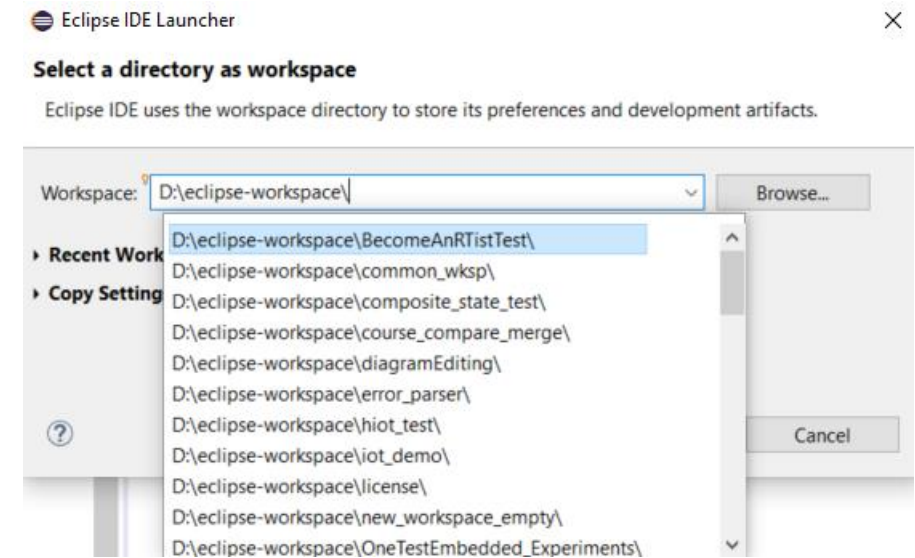- It's easy to launch it by typing "minimap" in the QuickAccess field

HCL SOFTWARE

Eclipse 4.12 (2019.06)

▶ The Select Workspace dialog now supports auto-completion

  ▪ Allows to pick a workspace more easily by only using the keyboard

▶ User-defined filtering of the Project Explorer

  ▪ You can now create your own regular expressions for filtering out items from the Project Explorer

  ▪ Can be used as an alternative to working sets

Eclipse IDE Launcher ✕

Select a directory as workspace

Eclipse IDE uses the workspace directory to store its preferences and development artifacts.

Workspace: D:\eclipse-workspace\   ▾   Browse...

▸ Recent Work
▸ Copy Setting

D:\eclipse-workspace\BecomeAnRTistTest\
D:\eclipse-workspace\common_wksp\
D:\eclipse-workspace\composite_state_test\
D:\eclipse-workspace\course_compare_merge\
D:\eclipse-workspace\diagramEditing\
D:\eclipse-workspace\error_parser\
D:\eclipse-workspace\hiot_test\
D:\eclipse-workspace\iot_demo\
D:\eclipse-workspace\license\
D:\eclipse-workspace\new_workspace_empty\
D:\eclipse-workspace\OneTestEmbedded_Experiments\

Cancel

Filters and Customization ✕

⇥ Pre-set filters ⇥ User filters ⁞⁞ Content

Create custom resource filters to apply. Resources with names that match a selected pattern will be hidden.

☑ Traffic*

New
Edit
Remove

OK   Cancel

Project Explorer ⊠
> DLLTest
> DLLTest_target
> LibTcpServer
> LibTCPServer_target
> NodePlusTestModel
> NodePlusTestModel_target
> stoplight-demo
> ToolDemo
> ToolDemo_target
> TrafficLight
  TrafficLight_Mattias
> TrafficLight_prebuilt_target
> TrafficLight_target
> TrafficLightDemo
> TrafficLightDemo_target
> TrafficLightFlow
> TrafficLightFlows

will be hidden by the user filter

HCL SOFTWARE
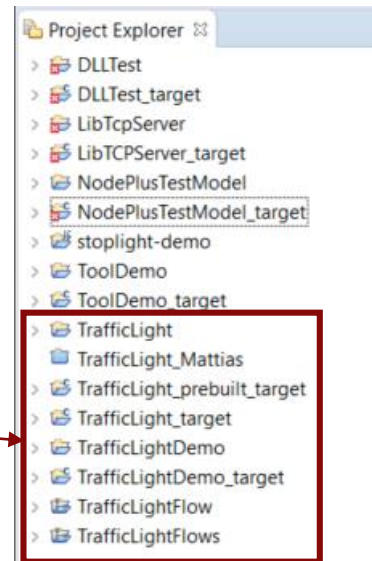
Eclipse 4.12 (2019.06)

▸ Copy/Paste of environment variables

- Setting up environment variables in a launch configuration is now much easier thanks to copy/paste support

- Environment variables can be copied from one launch configuration to another, or from a text editor to a launch configuration (or vice versa)
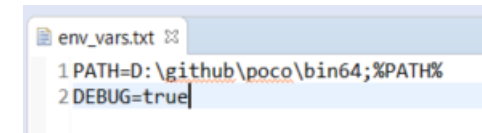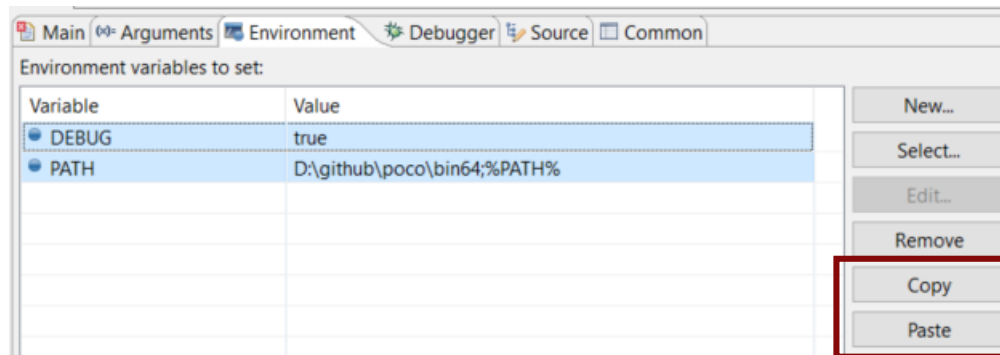

▸ Useful quick links for an empty workspace

- Quick links for creating or importing projects make it easier
  for new users to get started with an empty workspace

- Which quick links that are shown depend on the current perspective

HCL SOFTWARE
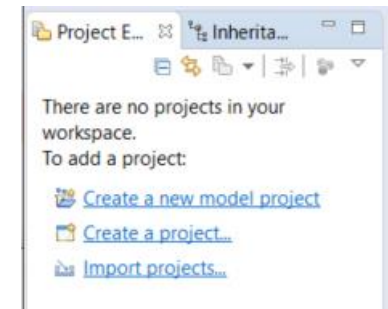
Eclipse 4.12 (2019.06)

▸ Launch configurations are now accessible from Quick Access

  ▪ Can start either a Run or a Debug session

**HCL SOFTWARE**
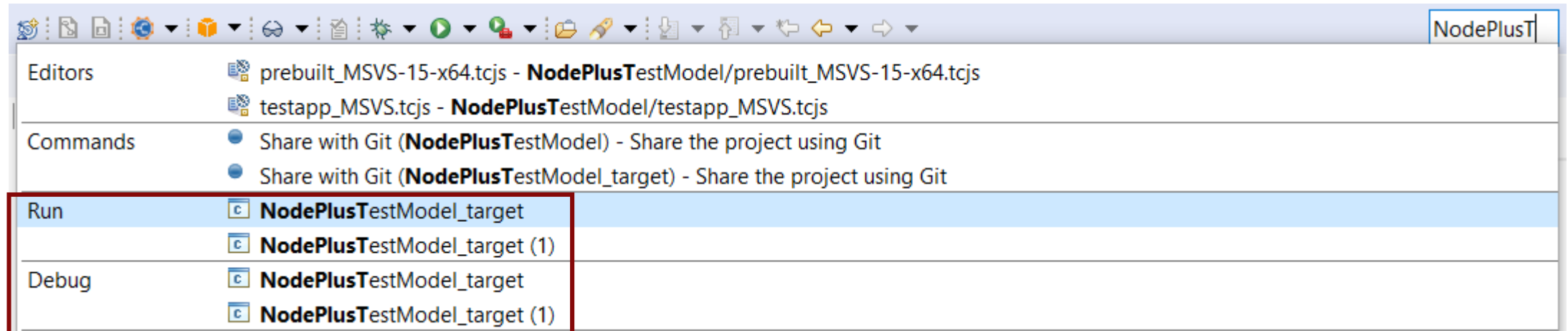
CDT 9.8 (included as part of Eclipse 2019.06)

▸ Possible to rename a file without triggering a refactoring

- A checkbox controls if references to the renamed file should be updated
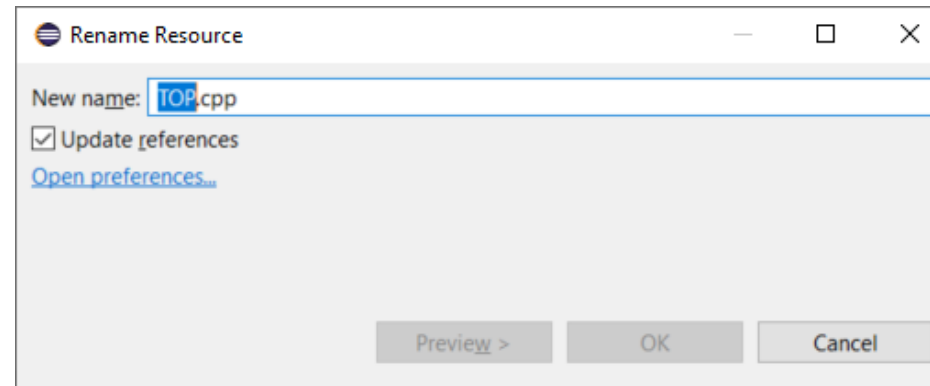
▸ New preference to address a performance issue with line wrapping a large number of lines in the console

- *C/C++ - Build – Console – Maximum number of lines to enable word wrap for*

- Don't set this too high if *Wrap lines on the console* is turned on

HCL SOFTWARE
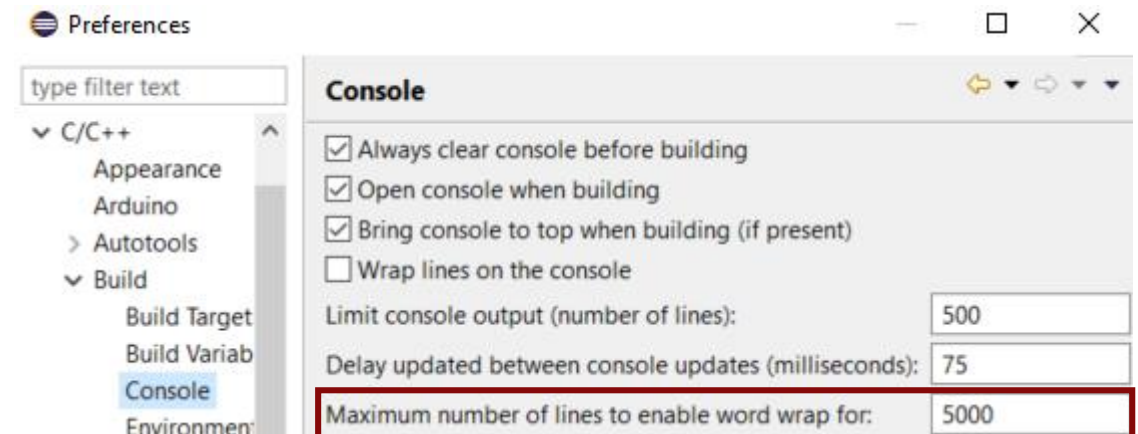
CDT 9.8 (included as part of Eclipse 2019.06)

▸ Improved dialog for attaching to a C++ application to debug

  ▪ Command-line arguments for the process are now shown (allows to more easily find the process to debug)

  ▪ The dialog now remembers a previously entered filter expression (to make it easier to attach to the same process multiple times)

▸ Copy the qualified name of a C++ declaration

  ▪ A new command is available in the Source menu

  ▪ Similar usecase as for the Copy button of the Qualified Name for a model element in the Properties view

HCL SOFTWARE
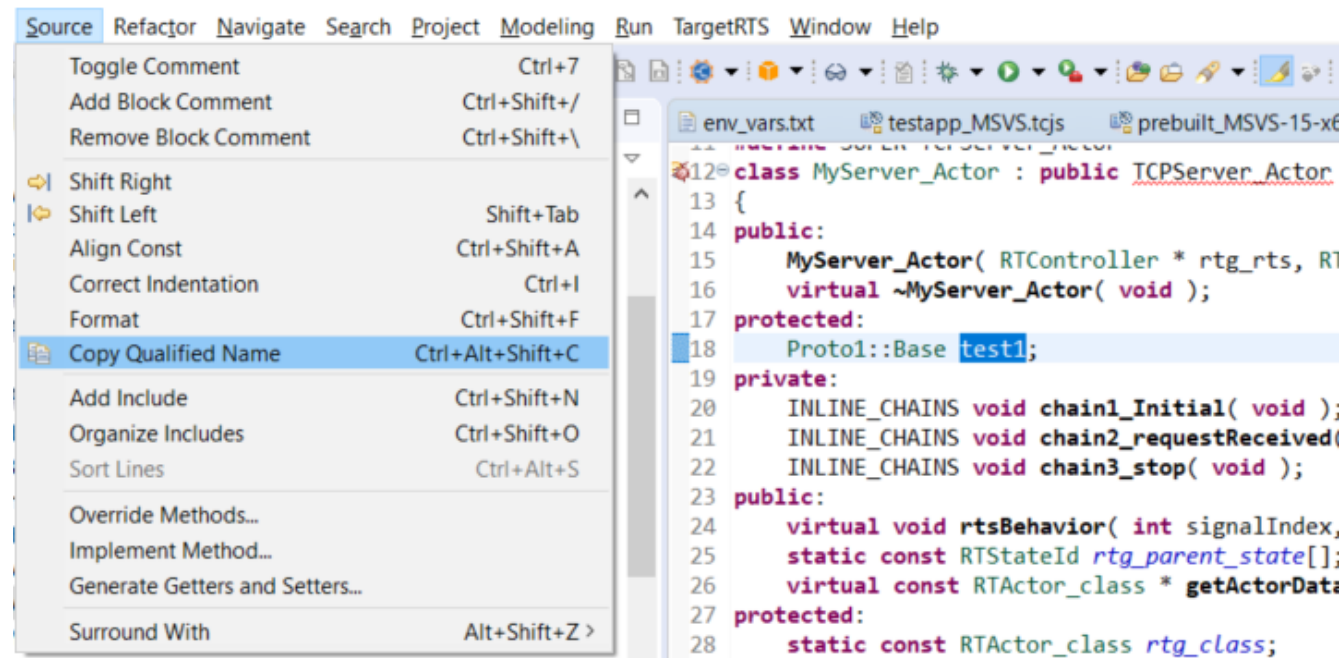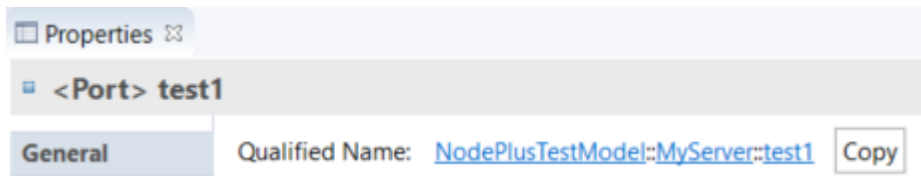
CDT 9.8 (included as part of Eclipse 2019.06)

▸ CODAN improvements

  ▪ Additional checks implemented, for example to detect C-style casts and goto-statements

▸ For more information about CDT improvements see
https://wiki.eclipse.org/CDT/User/NewIn96
https://wiki.eclipse.org/CDT/User/NewIn97
https://wiki.eclipse.org/CDT/User/NewIn98

HCL SOFTWARE

Newer EGit Version in the EGit Integration

▶ The EGit integration in RTist has upgraded EGit from 5.0 to 5.4

  ▪ This is the recommended and latest version for Eclipse 2019.06

▶ This upgrade provides several new features, performance improvements and bug fixes

  ▪ For detailed information about the changes see
    https://wiki.eclipse.org/EGit/New_and_Noteworthy/5.1
    https://wiki.eclipse.org/EGit/New_and_Noteworthy/5.2
   https://wiki.eclipse.org/EGit/New_and_Noteworthy/5.3
    https://wiki.eclipse.org/EGit/New_and_Noteworthy/5.4

HCL SOFTWARE

Libraries Menu

▸ A new menu in the menu bar with a few useful commands related to libraries

- Contains the command for launching the TargetRTS wizard (i.e. the menu replaces the TargetRTS menu)

▸ Commands for importing the Connexis and TCPServer libraries

- Available if these features have been installed

- They import the Connexis or TCPServer library projects from the installation into the workspace

HCL SOFTWARE
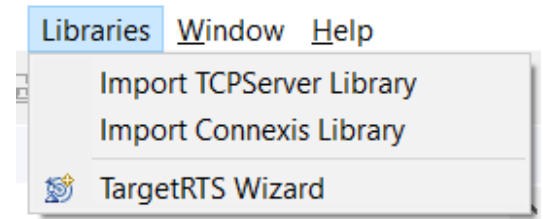
Converting Between In Events and Out Events

▶ It's now possible to convert an In Event to an Out Event or an Out Event to an In Event

- This can be useful for example if you want to change the conjugation of a port, and need to swap the direction of all events in the protocol of the port

▶ Use the new "Change Event Direction" command in the Project Explorer context menu of a protocol event (or multiple events)

HCL SOFTWARE
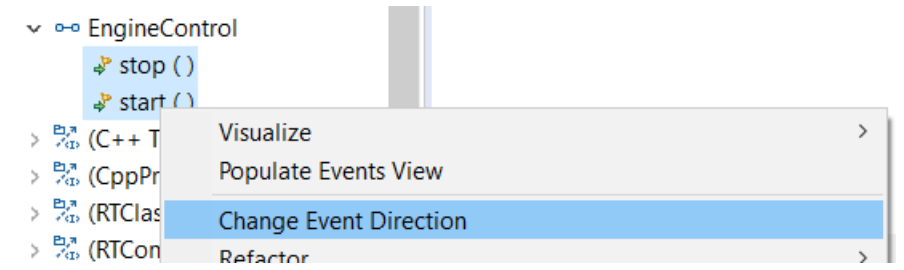
Moving Transitions in the Project Explorer

▶ You can now move a transition from one state to another using drag&drop in the Project Explorer

- ▪ Useful for example when refactoring a state machine (moving a transition from a sub state to the container state, or vice versa)

- ▪ Works for transitions in both capsule and passive class state machines

- ▪ The moved transition will initially become a self-transition and a new target state can later be set if needed

▶ If events/operations referenced by triggers of the moved transition are not available in the target context, a dialog will inform that such triggers will be deleted

HCL SOFTWARE
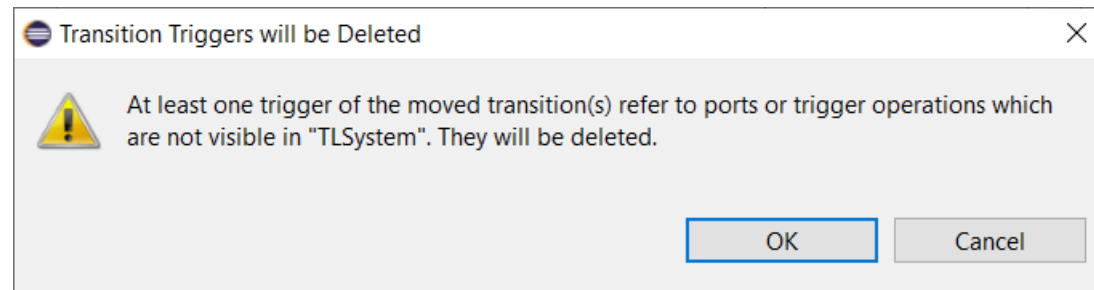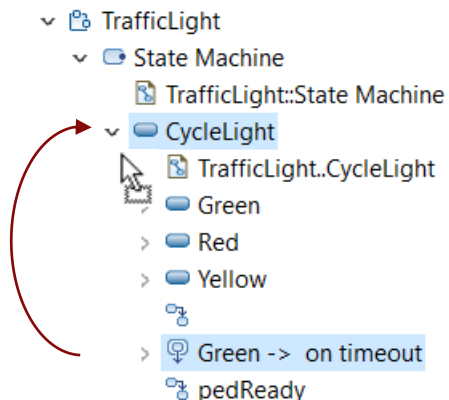
Shortened Qualifiers

▶ Fully qualified names now take the current viewpoint into account

- In the "Capsule Development Viewpoint" hidden elements such as regions are now excluded from the qualifiers

- Makes fully qualified names shorter, in particular deeply nested states

- Applies for a number of different views and dialogs (Project Explorer, Find Named Element, Diagram tooltips etc)
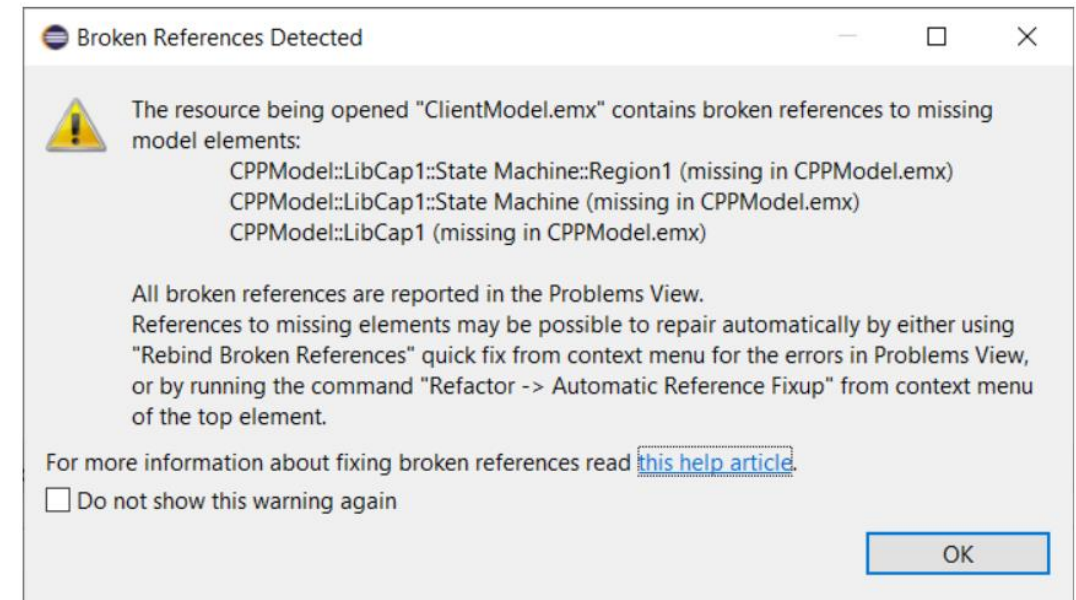
before

now

HCL SOFTWARE
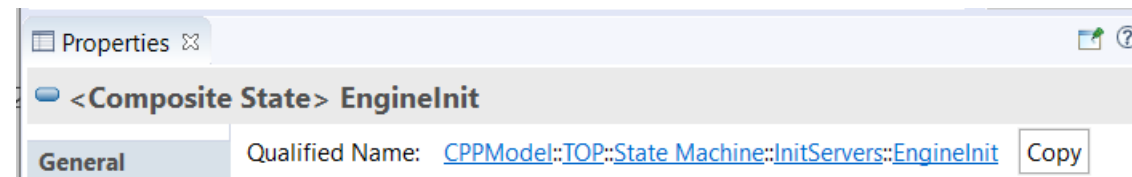
Primitive Type Improvements in Search and Type Completion

- Previously duplicates of primitive C++ types appeared when doing search and type completion

  - As a consequence it was easy to by mistake use the wrong type (and also tedious to have to choose each time)

  - The reasons for getting these duplicates have now been addressed

- The "C++ Types" package and "C++ Transformations" profile have been removed from the installation

  - They were obsolete and no longer needed

☑ ☐ ▥ RTist Extra Functionality	
☐ 🔷 RTist Connexis Feature	11.0.0.v20200529_1347
☐ 🔷 RTist CORBA Support	11.0.0.v20200529_1347
☐ 🔷 RTist Pluglets	11.0.0.v20200529_1347
☐ 🔷 RTist Sketcher Feature	11.0.0.v20200529_1347
☐ 🔷 RTist Support C	11.0.0.v20200529_1347
☐ 🔷 RTist Transformation Authoring Minimal Feature	11.0.0.v20200529_1347

- The support for C code generation is now an optional choice when installing RTist

- New and updated model fixups are now available for updating existing models for these changes

  - Incorrect type references fixup (updated to fix references to removed C++ Types and, optionally, also C types)

  - Obsolete package import removal (new fixup)

  - Obsolete profile application removal (new fixup)

  - Add missing package imports (new fixup for adding the import of CppPrimitiveDatatypes, in case it's missing)

HCL SOFTWARE
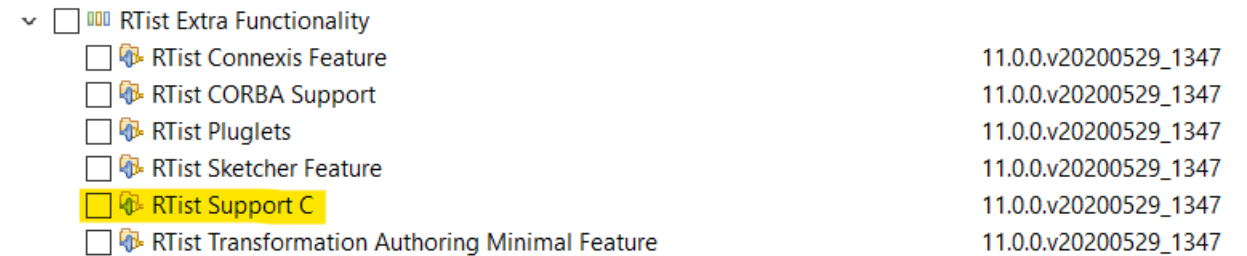
Fixing Broken References (1/2)

▶ The "automatic cleanup" of certain elements with broken references is now removed

  ▪ The preference *RealTime Development - Resource Resolution - Broken references auto-cleanup* has been removed

▶ By default, when a model with broken references are loaded a warning dialog now appears

  ▪ This dialog replaces the previous Repair Workspace References dialog

  ▪ To repair the broken references, by attempting to rebind them to elements that have been moved to a different file, follow one of the methods suggested by the dialog

  ▪ You can disable this warning dialog using the new preference *Modeling – Resource Resolution – Show warning for broken references when loading a model*

  ▪ Note: A current limitation is that broken references caused by deleted projects will not appear automatically when loading a model. To see such broken references perform the Validate command on the loaded model.

Broken References Detected

⚠ The resource being opened "ClientModel.emx" contains broken references to missing model elements:
    CPPModel::LibCap1::State Machine::Region1 (missing in CPPModel.emx)
    CPPModel::LibCap1::State Machine (missing in CPPModel.emx)
    CPPModel::LibCap1 (missing in CPPModel.emx)

All broken references are reported in the Problems View.
References to missing elements may be possible to repair automatically by either using "Rebind Broken References" quick fix from context menu for the errors in Problems View, or by running the command "Refactor -> Automatic Reference Fixup" from context menu of the top element.

For more information about fixing broken references read this help article.
☐ Do not show this warning again

OK

HCL SOFTWARE

Fixing Broken References (2/2)

▶ Broken references are listed in the Problems view. Right-click on any of them and run a Quick Fix for fixing broken references:

- **Rebind Broken References** can fix references to elements that have been moved. It's for example useful after a refactoring that was done without having all referencing models available in the workspace.

- **Clean Up Broken References** can fix references to elements that have been deleted (or for some other reason do not exist in the workspace). Broken references will be fixed by resetting them (or in some cases, by deleting the element that contains the reference).

- **Search or browse for a valid reference** can be used to rebind the broken reference to a different target element using the Select Element dialog. This Quick Fix only fixes the selected broken reference. Note: To be able to browse to the new target element for the reference you may need to first switch to the Model viewpoint (in case the Capsule Development view point filters out the desired target element).

Quick Fix dialog:

Quick Fix

Select the fix for 'The feature 'element' of 'ClientModel::ClientCap::State

Select a fix:
- Clean Up Broken References
- Rebind Broken References
- Search or browse for a valid reference

Problems:

Resource
☑ ⊗ ClientModel.emx

Select All
Deselect All

Finish   Cancel

HCL SOFTWARE
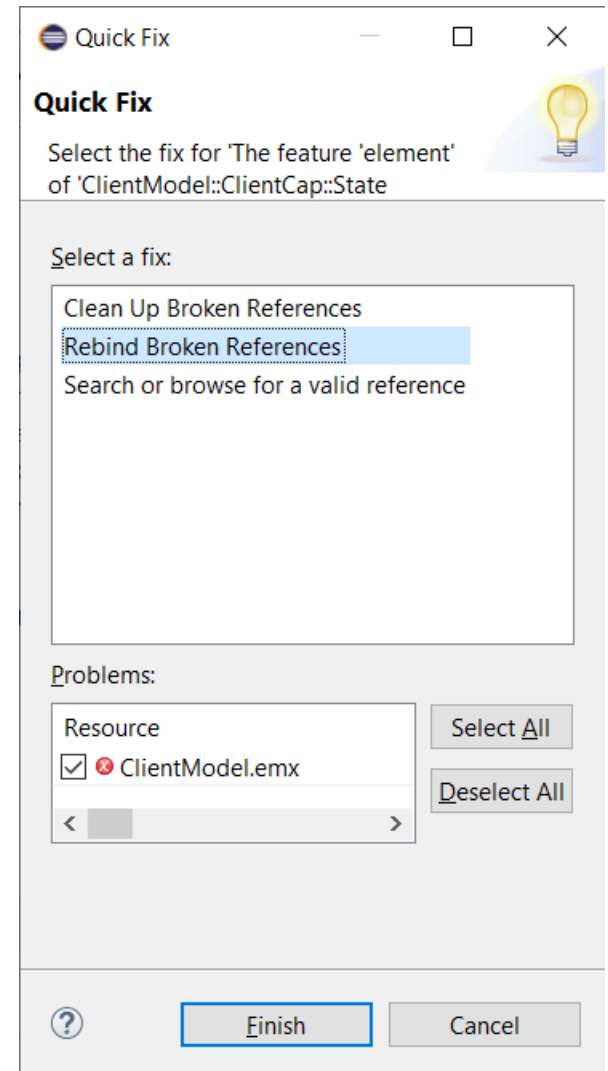
Creating Special Operations

▶ New commands are now available in the Add UML context menu for creating "special operations" on a capsule or class

- Constructors (default, copy, move)

- Destructor

- Getter/Setter operations

▶ Constructors and destructors will have the correct C++ signature, but no implementation. Write your own implementation as needed.

▶ Getter/Setter operations can be configured using a dialog, and will have typical default implementations

HCL SOFTWARE
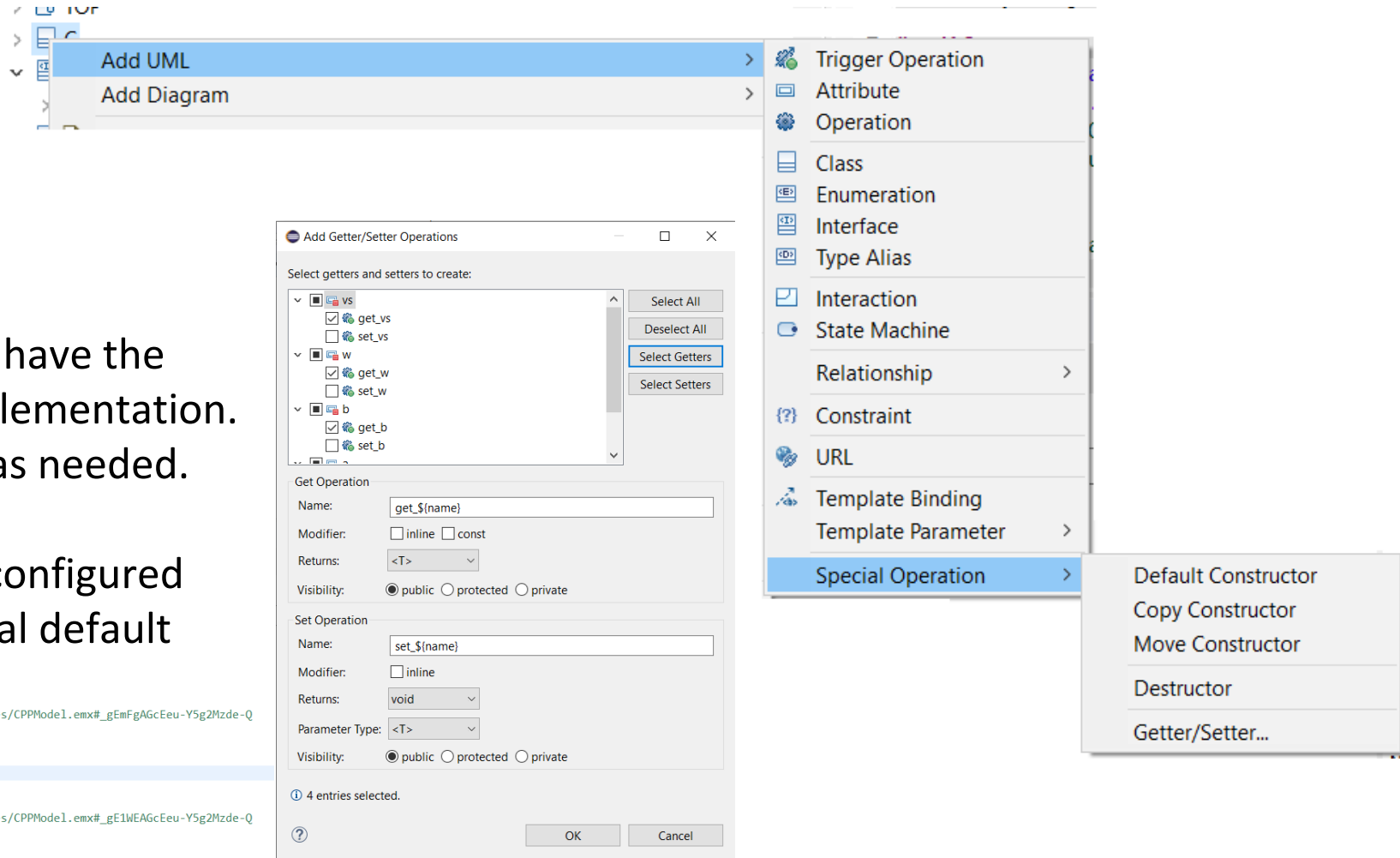
Model Debugger Improvements (1/3)

▶ State instance diagram for a sub state machine can now be opened by double-click

- Previously the command "Open State Machine Diagram" (Ctrl +T) was the only easy option

- Now it also works to double-click on the state name (same as when the model is not being debugged)

▶ The socket timeouts used for communicating with the debugged application are now configurable

- New preferences on the *Run/Debug – RealTime Application* preference page

- When debugging remotely over slow connections a too small socket read timeout can cause the debug session to terminate unexpectedly
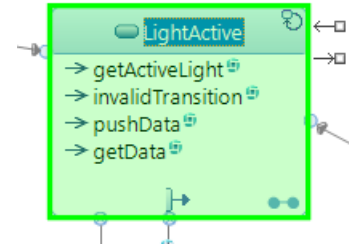
▶ The performance of configuring the Trace Editor was significantly improved

- Now by default only the listed elements are traced

HCL SOFTWARE

Model Debugger Improvements (2/3)

▶ A new command "Show Only This Instance" in the Trace Editor context menu

- Makes it easy to filter the trace to only see trace events related to the selected capsule instance

- For example useful in order to find out which state a capsule instance is in when sending or receiving an event

Active state when sending the event

Remove the filter to again show all trace events

HCL SOFTWARE

Model Debugger Improvements (3/3)

▸ Tracing everything related to a capsule

- Adding a capsule to the list of elements to trace is now a shorthand for adding all its ports and states (including sub states). Previously only incarnation of the capsule was traced in this case.

HCL SOFTWARE

Graphic Visualization of TC Relationships

▸ The possibility to visualize TC relationships that was available in version 10.2 for TCs in the old file format, is now also supported for TCs in the new .tcjs file format

  ▪ A new command **Visualize – Explore in Graphs** is available in the Project Explorer context menu of a TC

▸ The visualization uses Graphviz as an external tool for doing the graph layout

  ▪ Graphviz must be separately installed. Specify the path to its "dot" utility in
    Preferences – RealTime Development - Visualization

▸ The graph is rendered using SVG in an embedded web browser

  ▪ Support for panning and zooming

  ▪ You can search for texts in the rendered diagram using Ctrl+F

▸ Prerequisites and/or inheritance relationships can be shown

▸ Double-click on TCs in the graph to open the TC editor

HCL SOFTWARE

Event Routing Visualization
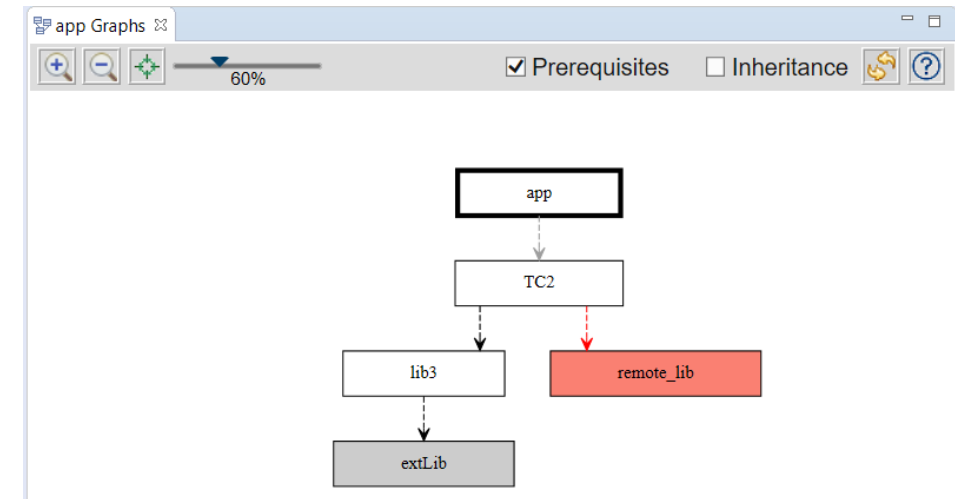
- Visualize graphically how a protocol event can be routed in the composite structure of a capsule

  - A new command **Visualize – Event Routing** is available in the context menu of a port (in Project Explorer, diagrams or Code/CDT Editor)

  - Select an event in the drop down menu to see how that event can be routed in the composite structure

- Explore the composite structure interactively

  - Use commands "Go Inside" and "Go Outside" in the ports' context menu in the diagram

- If there are multiple capsule parts typed by the same capsule, they will be replaced by hyperlinks in the diagram to save space

- The graph uses SVG so panning, zooming and text search is supported

HCL SOFTWARE

Navigation from Code Sending an Event to the Transition it Triggers
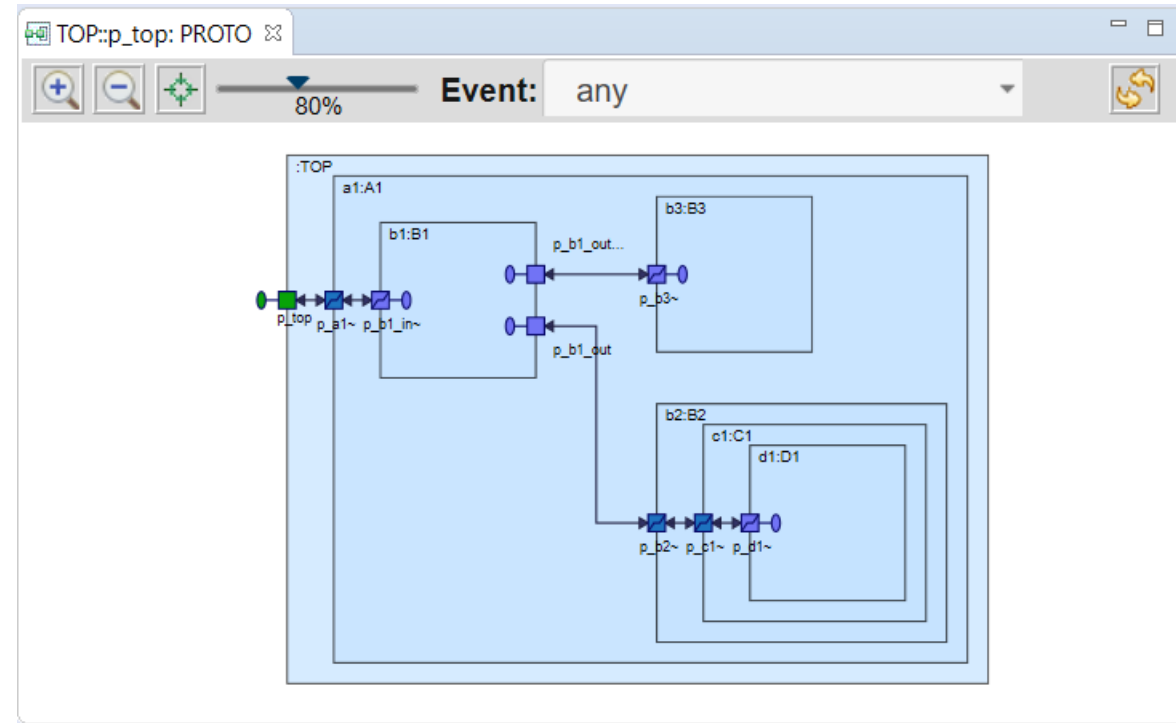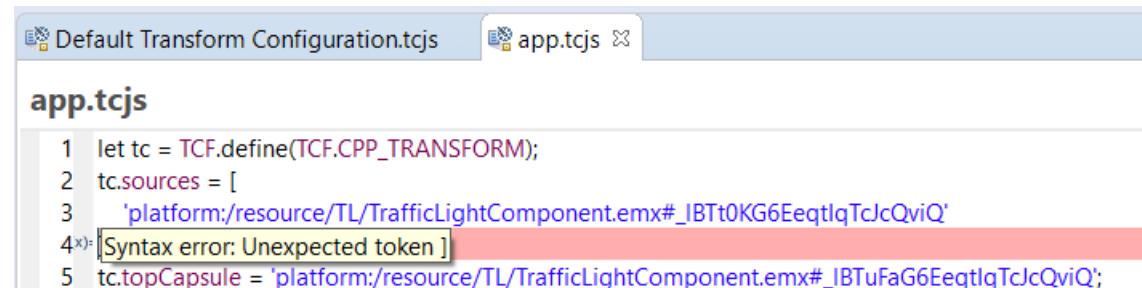
The Event Routing Visualization feature makes it easy to navigate from a place in the code where an event is sent, to the transition(s) that may receive it:

1. Right-click in the Code or CDT editor on a line where an event is sent (or invoked or replied to) and open the Event Routing Visualization graph.

2. Follow the arrows from the green port (the port on which the event is sent) to find the possible ports that may receive the event. To see also arrows to ports on enclosing capsules, perform **Go Outside** on the green port to reveal more of the composite structure.

3. Right-click on such a port and perform the command **Find Triggers**. All transitions which may trigger when the event arrives at the port will be shown in the Search View.

4. Double-click on a found transition to navigate to it in a diagram

HCL SOFTWARE

▸ Syntax highlighting is now available in the Code tab

  ▪ JavaScript parse errors are shown with red color and margin marker tooltip
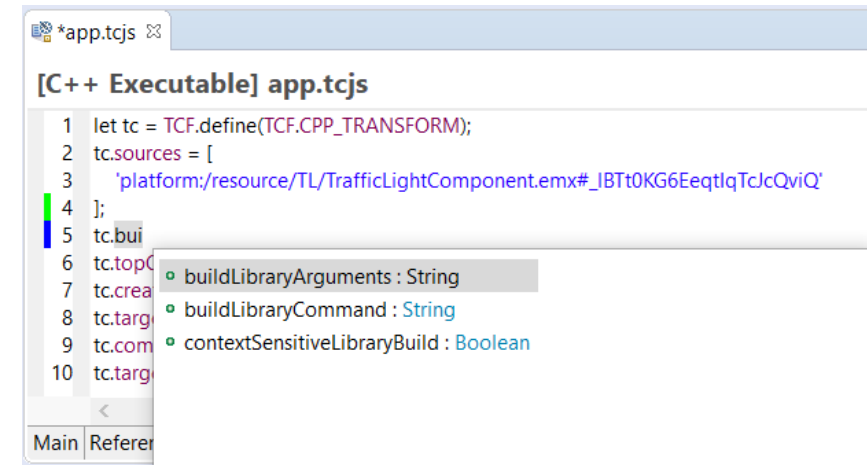
```
Default Transform Configuration.tcjs      app.tcjs ⊠

app.tcjs

1    let tc = TCF.define(TCF.CPP_TRANSFORM);
2    tc.sources = [
3        'platform:/resource/TL/TrafficLightComponent.emx#_IBTt0KG6EeqtIqTcJcQviQ'
4    Syntax error: Unexpected token ]
5    tc.topCapsule = 'platform:/resource/TL/TrafficLightComponent.emx#_IBTuFaG6EeqtIqTcJcQviQ';
```

▸ Content assist ("code completion") is now available in the Code tab

  ▪ Triggered automatically when typing certain "key" characters ('.', '=' etc)

  ▪ Can also be manually triggered using Ctrl + Space

  ▪ Controlled by a new preference *RealTime Development – Transformation Configuration Editor – Content Assist – Enable auto activation*
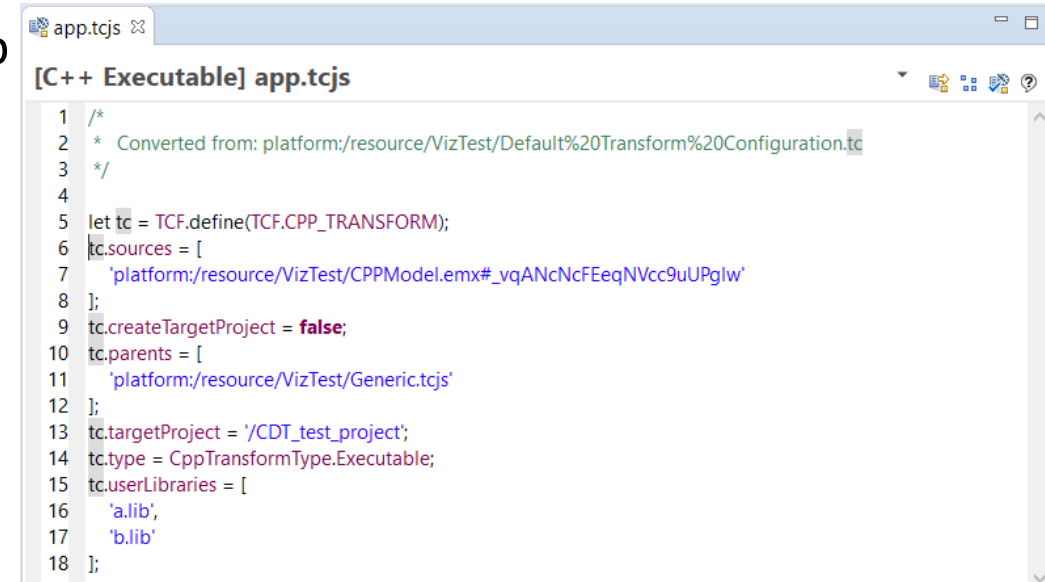
```
*app.tcjs ⊠

[C++ Executable] app.tcjs

1    let tc = TCF.define(TCF.CPP_TRANSFORM);
2    tc.sources = [
3        'platform:/resource/TL/TrafficLightComponent.emx#_IBTt0KG6EeqtIqTcJcQviQ'
4    ];
5    tc.bui
6    tc.topC        ○ buildLibraryArguments : String
7    tc.crea        ○ buildLibraryCommand : String
8    tc.targ        ○ contextSensitiveLibraryBuild : Boolean
9    tc.com
10   tc.targ

Main  Referer
```

HCL SOFTWARE

Transformation Configuration Editor Improvements (2/2)

▸ Marking of occurrences of selected element in the Code tab

 ▪ Controlled by a new preference *RealTime Development – Transformation Configuration Editor – Mark Occurrences – Mark occurrences of selected element*

▸ Incremental Find is now supported in the Code tab (with the same keyboard bindings as in other Eclipse text editors)

HCL SOFTWARE

Organize Sources

▶ A new Organize Sources dialog is provided to assist in setting up the Sources of a TC optimally
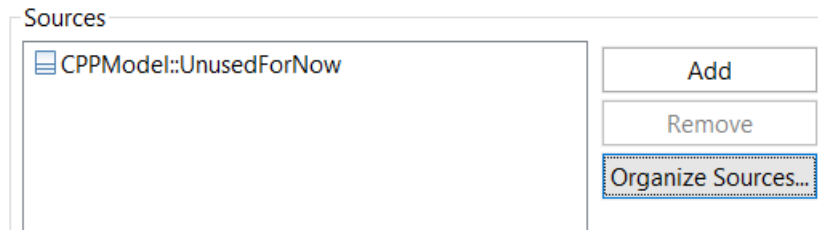
- This is an improved and simplified implementation compared to the similar feature in 10.2

- Elements are proposed to be added to or removed from the Sources list to ensure that no unnecessary elements will be built by the TC. The suggestions are made by analyzing references in the model.

- This feature is also available from the command-line by means of a new model compiler command-line argument `--organizeSources`

- It can also be launched from the TC Editor toolbar menu (useful for TCs edited only textually in the Code tab)

HCL SOFTWARE
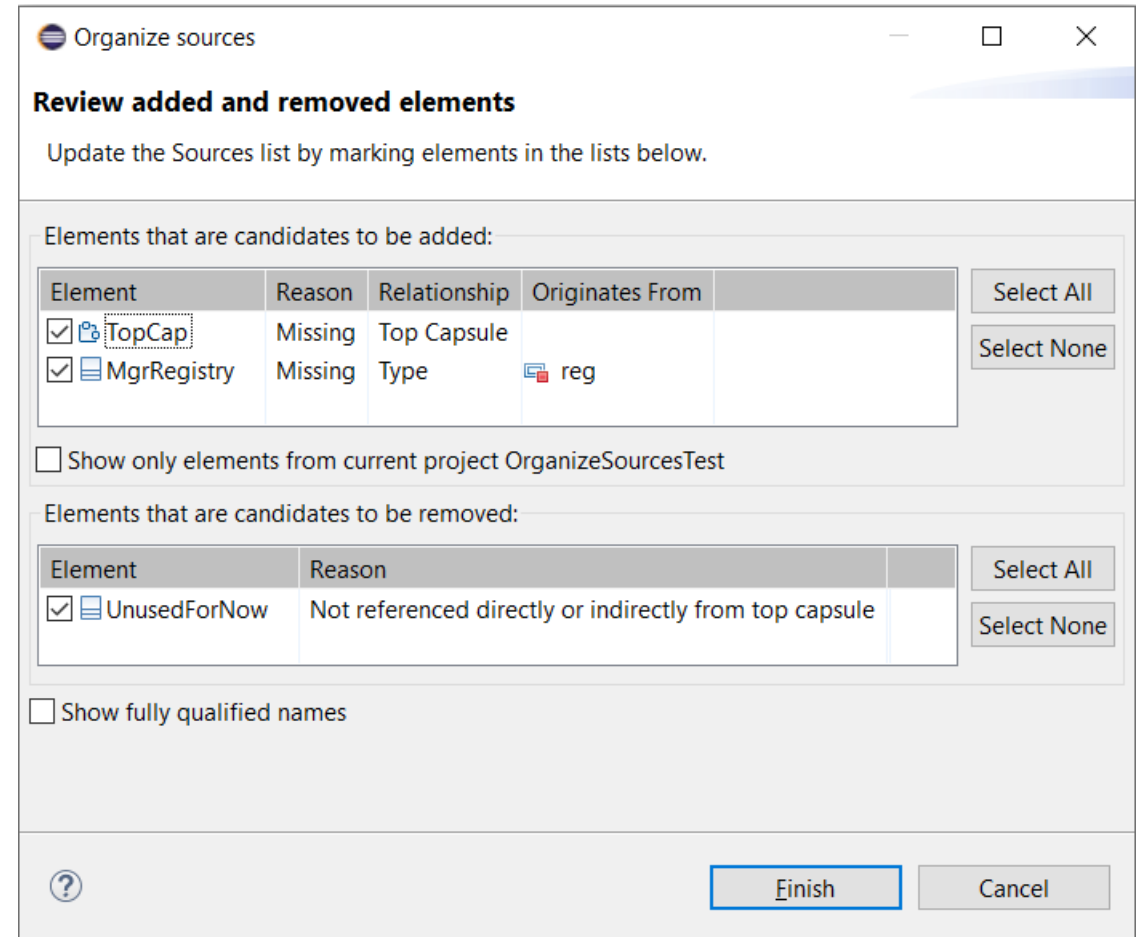
Cleaning TCs

▶ External Library TCs have a new "Clean Command" property

  ▪ Use for example "make clean", invoke a "clean script" or something else

▶ When it has been specified it's possible to use the Clean command also on external library TCs

  ▪ Currently this only works when cleaning from the command-line (using the makefile generated by the model compiler)

▶ When build variants are used, the Clean dialog now shows the Build Variants user interface

  ▪ Allows setting the build variant settings which may influence on the behavior of clean, and clean build

☐ Generate make file for external CDT project

Build command: echo "Build command for external Lib04"

Build folder:

Clean command: echo "Clean command for external Lib04"

● Clean selected transformation configurations ✕

☑ 🗒 build.tcjs

Build Variants

Configuration: New configuration from build variants ... ∨ Save As...

Target platform MinGW 02 ∨ ☑ Print TCJS name
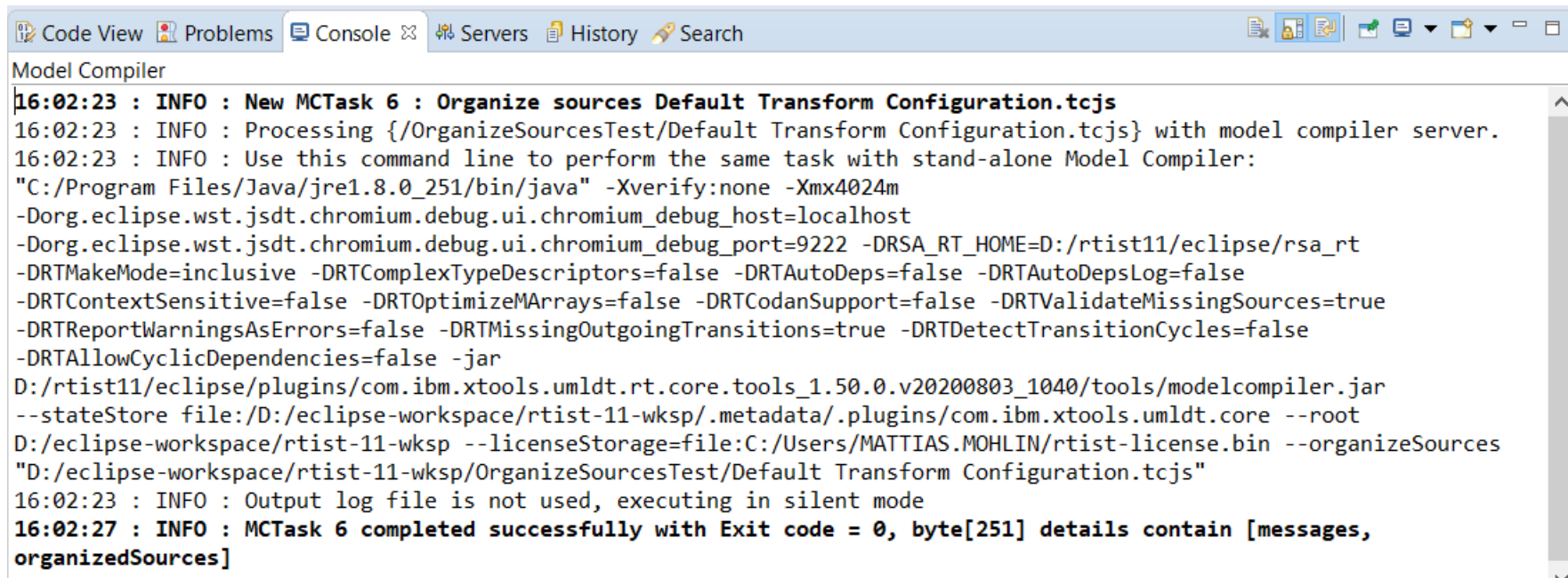
☐ Clean binaries only

☑ Build selected transformation configuration after clean

☑ Set as Active Build Variants Configuration

⑦ Clean Cancel

HCL SOFTWARE

Model Compiler Console

▶ A new "Model Compiler" console is now available

 ▪ Helps troubleshooting commands that involves calls to the model compiler

 ▪ Makes it easier to know how a model-compiler command should be invoked from the command-line

HCL SOFTWARE

Incremental Compilation for Models with External Code

▶ This feature worked in the classic builder by means of a Perl script (rtcppdep.pl)

▶ The model compiler now also supports this, by relying on the compiler's ability to generate dependency files

   ▪ Currently only supported for GCC compilers, and only for recursive make files

▶ The file `libset.mk` in the TargetRTS must be updated to enable this feature

   ▪ The variable $(DEP_OPTS) should be added to LIBSETCCFLAGS

   ▪ Note that $(DEP_OPTS) is a new variable in the file `default.mk`. In this file is also another variable $(DEP_EXT) which specifies the file extension for dependency files (.d). If this variable is not defined, dependency files will not be included in the generated makefile.

▶ <u>Note:</u> A clean build will take slightly longer when this feature is enabled (due to dependency file generation), but subsequent builds will be faster since only the minimal number of .cpp files will be recompiled when something in the external code has been changed.

HCL SOFTWARE

Code Preview

▶ A preview of what generated code will look like can now be generated

  ▪ Use the new command **Generate Code Preview** in the Project Explorer context menu of a model element

▶ Preview code is placed in a special project
(by default called "_CODE_PREVIEW")

▶ Old versions of preview code files are
marked with ~
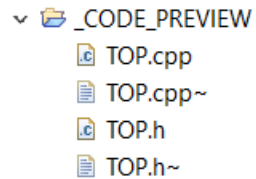
  ▪ You can compare old and new file
versions using **Compare With – Each Other** to easily
understand how a change in the model have affected the generated code
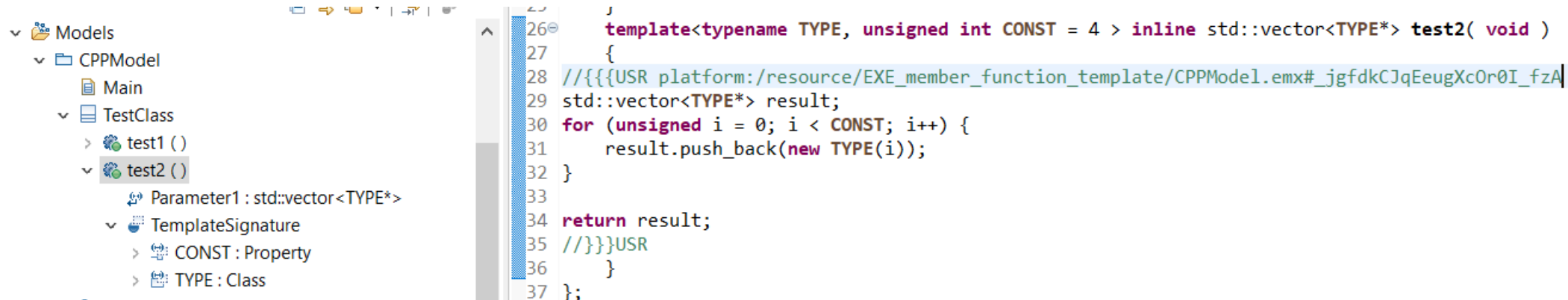
▶ Both code and makefiles can be previewed and compared in the same way

▶ This is currently an experimental feature

HCL SOFTWARE
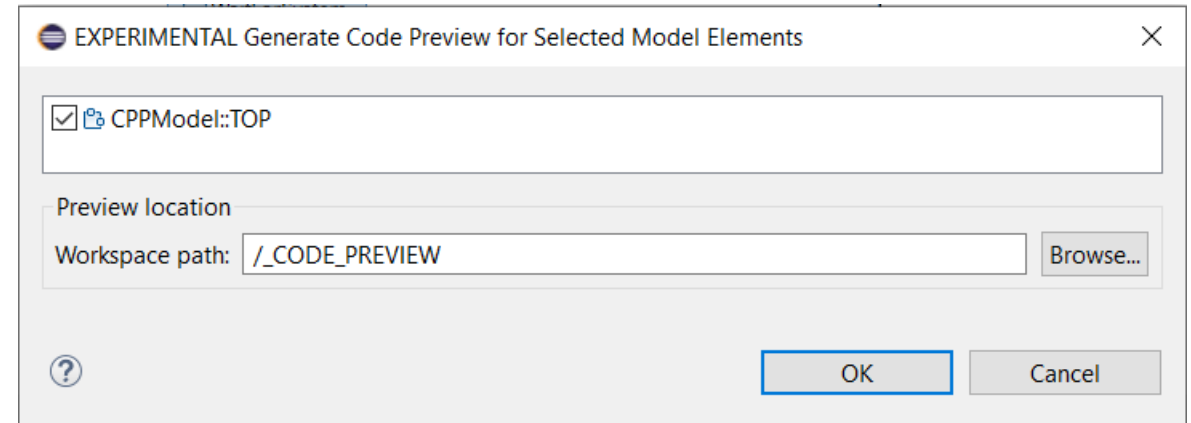
Improved Support for Templates (1/2)

▶ The model compiler now supports member functions with template parameters

- Both type and non-type template parameters are supported

- Note that most compilers require such operations to be declared as **inline**

▶ The Project Explorer representation of template parameters is now more compact

- Easier to recognize type template parameters
  by means of the word "typename" in the label

before   *now*

HCL SOFTWARE

Improved Support for Templates (2/2)

▶ The Templates page of the Properties view has been much improved

- Consistent look-and-feel with other property pages

- Support for reordering template parameters using either up/down buttons or by drag&drop

- Improved textual representation of template parameters, more aligned with the C++ syntax

▶ Improved context menu for creating template parameters

- Now aligned with C++ terminology (type and non-type template parameters)

HCL SOFTWARE
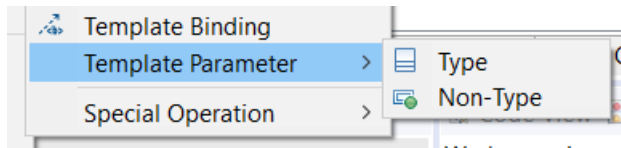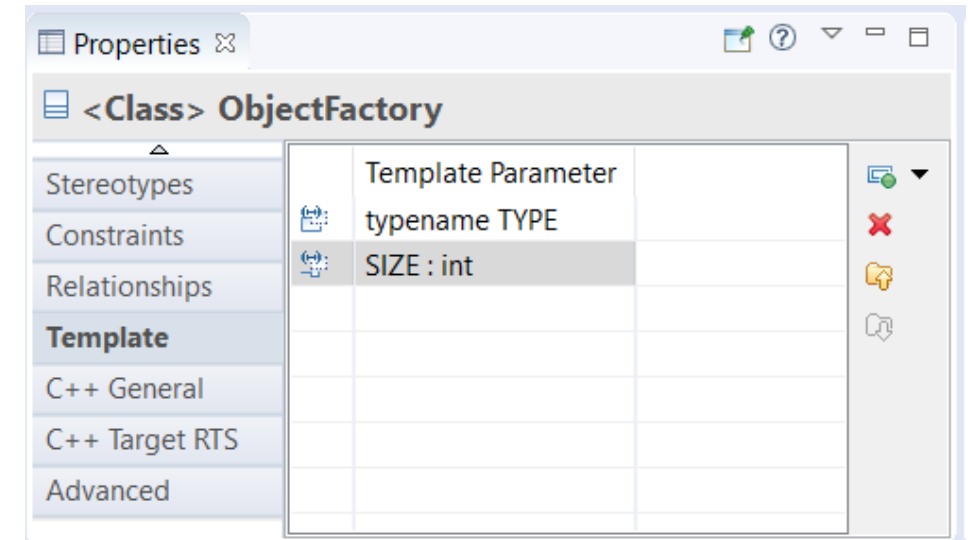
C++ 11 Improvements (1/4)

▶ The **final** modifier can now be used on classes, capsules and operations

- Use it to specify that a class/capsule cannot be inherited from, or an operation cannot be redefined

▶ **auto** attributes are now supported by the model compiler

- The C++ compiler will deduce the type of the corresponding member variable from its initializer value

- Type completion now works for auto, and it can be selected as any other type

▶ Attributes with in-class initialization (brace or equal) are no longer also initialized in the constructor

▶ Some C++ 11 types were missing in the CppPrimitiveDatatypes package and have now been added

- long long

- unsigned long long

HCL SOFTWARE
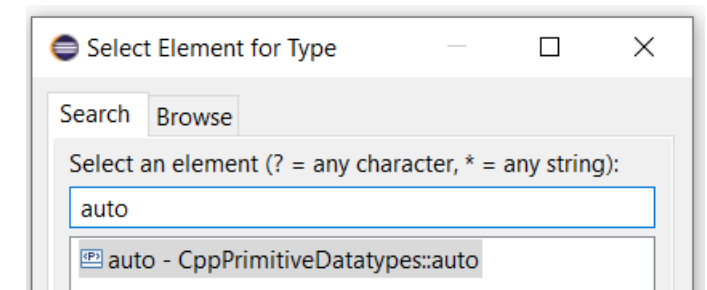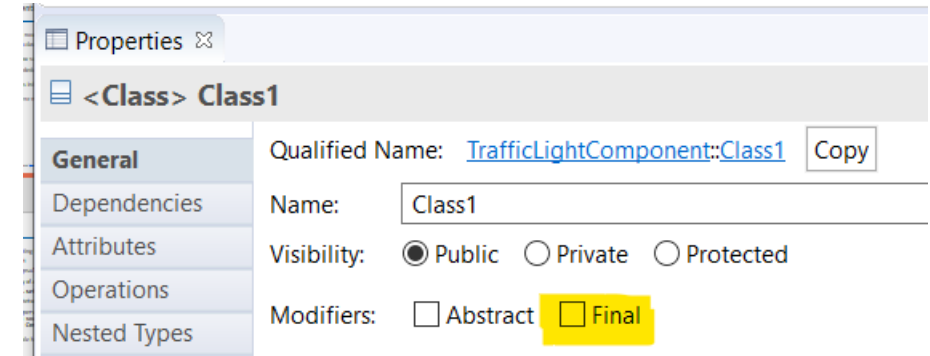
C++ 11 Improvements (2/4)

▶ The **override** keyword is now specified for functions that redefine virtual functions from the TargetRTS. Also, the **nullptr** constant is now used instead of 0 for representing a null pointer.

- Both generated code and the TargetRTS code was changed

- Avoids warnings when compiling generated code

- For GCC compilers, the warnings `–Wsuggest-override` and `–Wzero-as-null-pointer-constant` are now set when compiling to detect these problems also in user code snippets

- **Note:** Starting with these changes it's now required to use a C++ 11 compiler for compiling generated code. You can use the new preference *RealTime Development – Build/Transformation – C++ code standard* if you need to compile generated code with a compiler that doesn't support C++ 11.
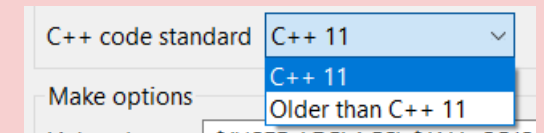
▶ Operations can now be marked as deleted, and destructors can be marked as default or delete

HCL SOFTWARE
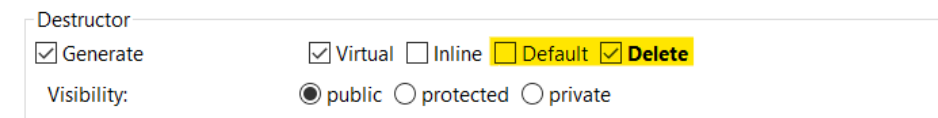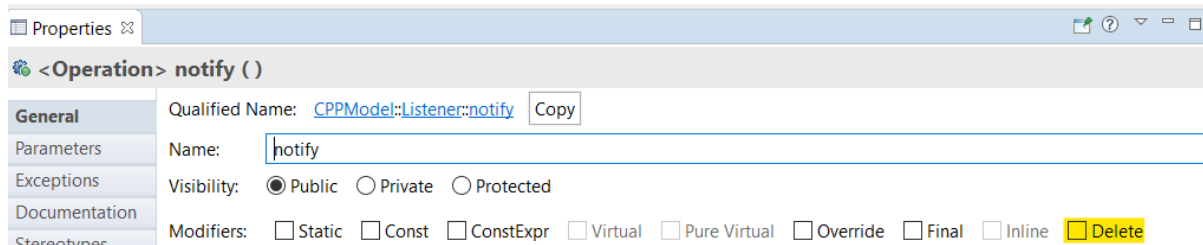
C++ 11 Improvements (3/4)

▶ C++ 11 type aliases are now supported in the user interface and model compiler
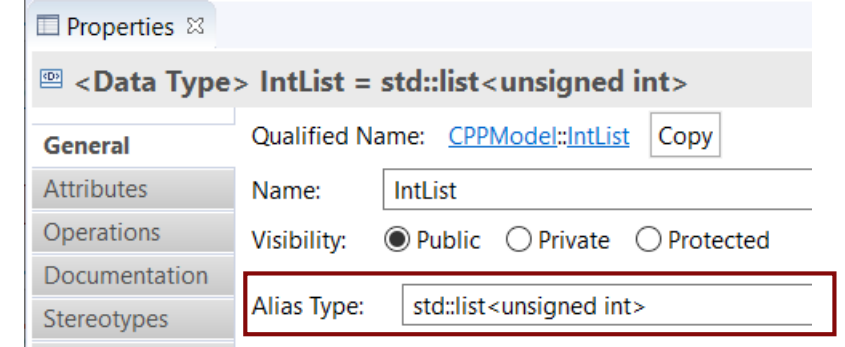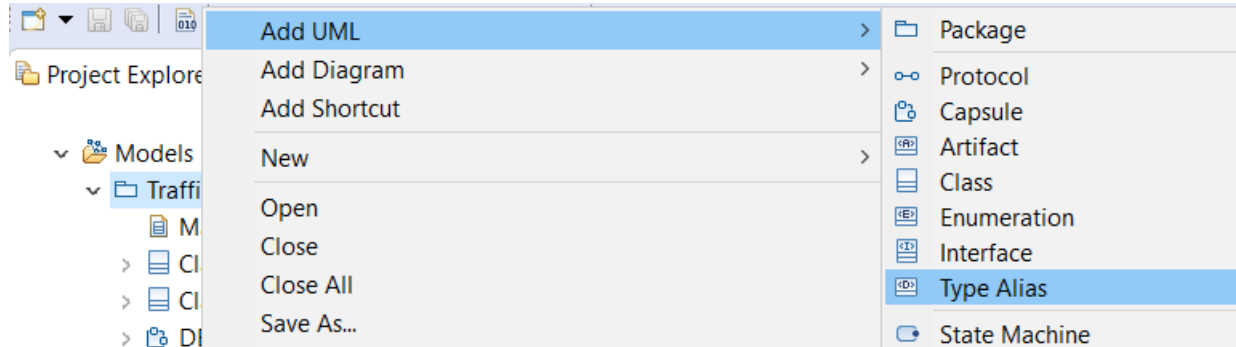
- In the model a DataType with a new "Alias Type" property is used

- The main benefit with using type aliases instead of traditional typedefs is that they can have template parameters

```
template<typename TYPE > using GenericList = std::list<TYPE*>;
```

- You can write a type descriptor for type aliases in the same way as you currently can for typedefs. The "Type Descriptor Hint" property is also supported, to automatically generate a type descriptor for type aliases of commonly used STL collection types (list, set, map etc).

HCL SOFTWARE

C++ 11 Improvements (4/4)

▶ When overriding an operation in the UI, its 'override' property is automatically set

- If the overridden operation is final, a new dialog appears and suggests to make it non-final
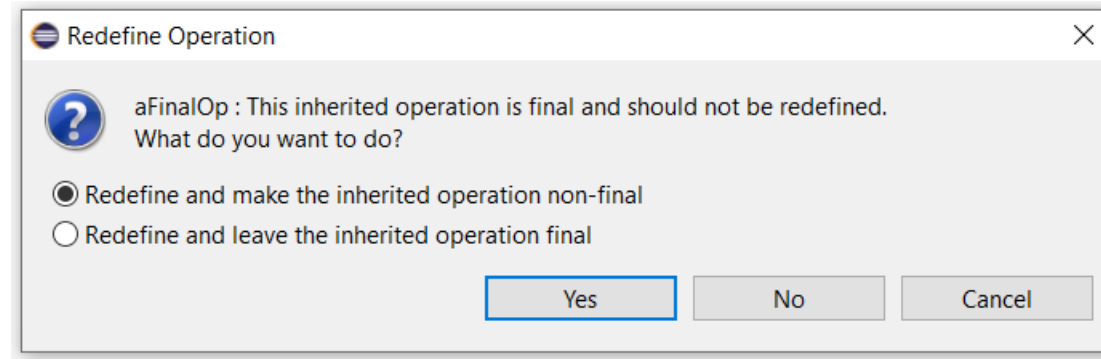
- For redefined operations, the "override" property is automatically set (and cannot be unset)

HCL SOFTWARE

Removal of C-style Casts

▶ Both generated code and the TargetRTS have been changed by removing all C-style casts

▶ Each cast has either been removed, or replaced with a corresponding C++ cast

- `static_cast` (in most cases) or `reinterpret_cast` (in a few cases)

- Note: We don't generate `dynamic_cast` since we cannot assume code is compiled with run-time type information

```
125  const RTObject_class RTType_DataClass =
126  {
127      nullptr
128      , (RTSuperAccessFunction)nullptr
129      , "DataClass"
130      , (RTVersionId)0
131      , (RTFieldOffset)sizeof( DataClass )
132      , (RTInitFunction)rtg_DataClass_init
133      , (RTCopyFunction)rtg_DataClass_copy
```
before

```
125  const RTObject_class RTType_DataClass =
126  {
127      nullptr
128      , nullptr
129      , "DataClass"
130      , 0 /*RTVersionId*/
131      , sizeof( DataClass )
132      , reinterpret_cast< RTInitFunction > ( rtg_DataClass_init )
133      , reinterpret_cast< RTCopyFunction > ( rtg_DataClass_copy )
```
now

▶ A benefit is that the TargetRTS now can be compiled without the `-fpermissive` flag with GCC (without warnings)

- Helped to detect a couple of bugs in the TargetRTS specific to 64-bit platforms – now fixed!

▶ With GCC the TargetRTS is now compiled with new flags to detect issues related to casts

- `-Wuseless-cast` and `-Wold-style-cast`

HCL SOFTWARE

Removed Compatibility with Old Rose-RT Versions

▶ The TargetRTS file `RTCompatibility.h` was removed

  ▪ It contained macros, typedefs etc. for compatibility with very old versions of Rose-RT (6.4 and earlier)

▶ Migrated models that still use old Rose-RT names should be updated

  ▪ RTDATA -> rtdata

  ▪ Integer -> RTInteger

  ▪ … etc.

▶ It's recommended to update your models, but if you want to keep using these old names, you can keep `RTCompatibility.h` from the old TargetRTS version

HCL SOFTWARE

Dependencies for Interfaces and Enumerations

▸ The Dependencies property page is now available also for Interfaces and Enumerations

- Makes it easier to specify dependencies for these elements

- Helps for example when an interface operation references another type which must be included or forward declared in the interface header file

HCL SOFTWARE

Changing Default Values for Property Set Properties

▶ Most of the properties shown on the "C++ General" and "C++ Target RTS" property pages have customizable default values

- Before, the only way to change the default values of these so called "property set" properties was to do it from the Property Sets property page on the top-level package.

- The drawback was that it was necessary to do this on all models that needed a customized default value!

▶ Now, a new preference is available which makes it possible to set new default values globally, without modifying any model

- *Modeling – Profiles – Property Sets – Defaults*

▶ An example of when this can be useful is when you want custom names for generated getter/setter operations

HCL SOFTWARE
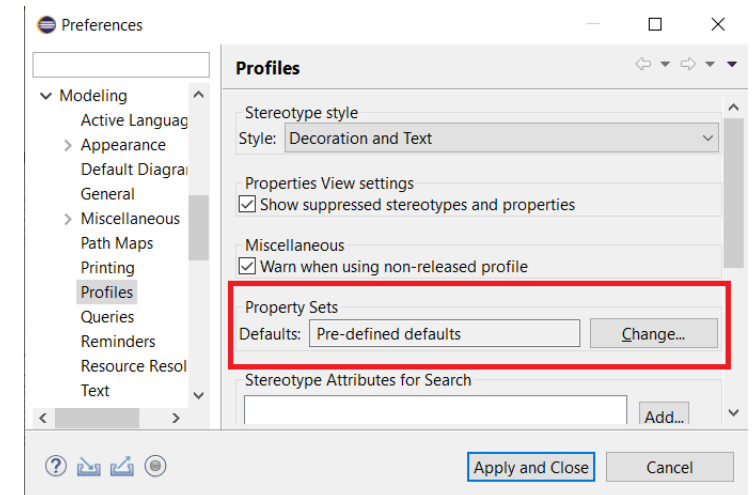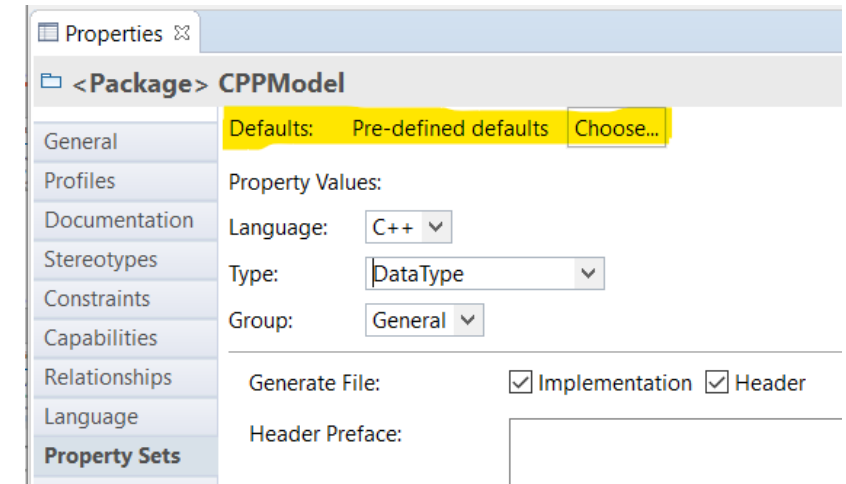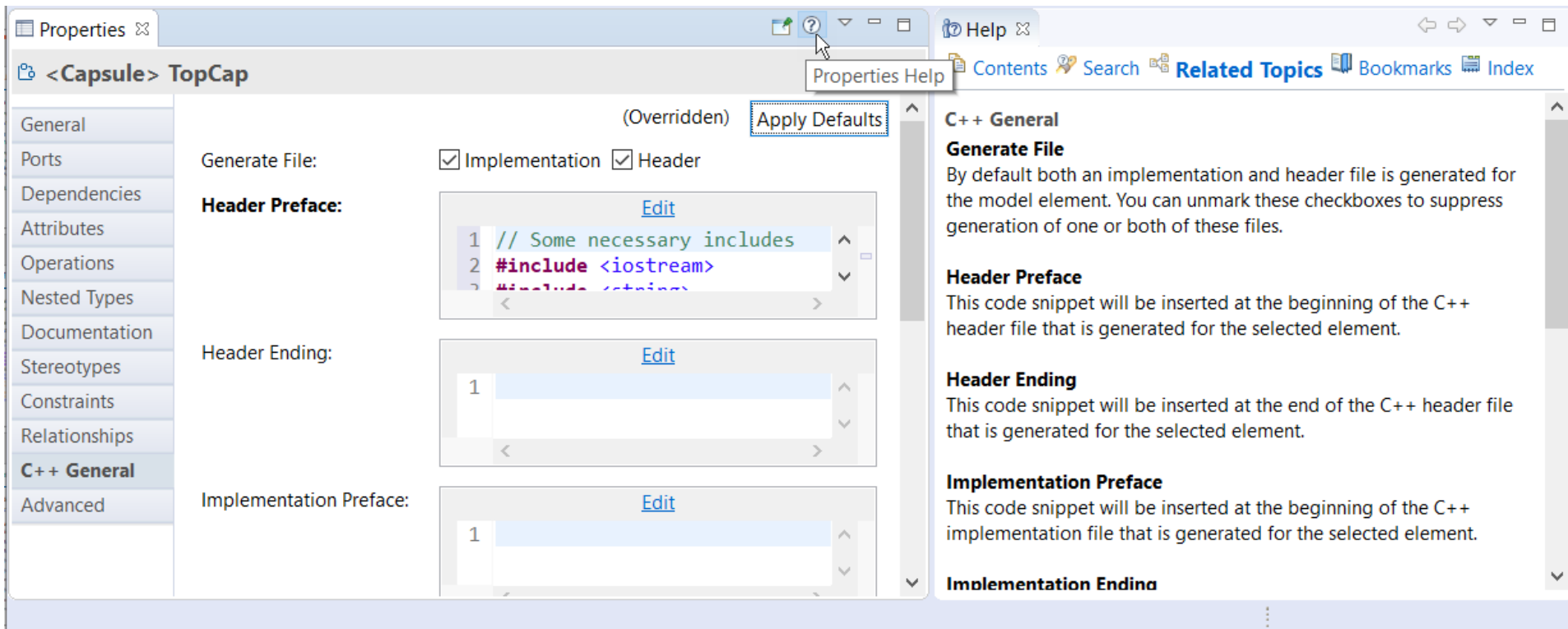
Moving Elements in the Properties View using Drag&Drop

▶ It's now possible to move elements in tables of the Properties view by means of drag&drop

- More convenient than using the Up / Down buttons if there are many elements in the table

- Multiple elements can be moved at the same time and will then all be inserted at the drop location (shown by a line in the table)

HCL SOFTWARE

Help for the Properties View

▶ The Properties View now provides context sensitive help

- Use the new toolbar button for showing help for the properties currently visible in the view (or press F1)

- The new property documentation helps understand what properties mean, and how they affect the generated code

HCL SOFTWARE

UML-RT Java APIs

▸ The Java APIs for creating UML-RT models programmatically have been extended

- Creating a capsule part

`RTCapsulePart RTCapsule.`**`createCapsulePart`**`(String `**`name`**`, RTCapsule `**`type`**`);`

- Setting the multiplicity of a capsule part, and its kind (fixed, optional or plugin)

`void`` RTCapsulePart.`**`setMultiplicityAndKind(`**`int`** `lower`**`, `**`int`** `upper`**`, `**`AggregationKind`** `kind`**`);`**

- Creating a connector between two ports on capsule parts

`Connector`` RTModelOperations.`**`createConnector(RTCapsulePart`** `sourcePart`**`,`**
`RTPortRedefinition` `sourcePort`**`,`**` RTCapsulePart `targetPart`, RTPortRedefinition`
`targetPort`**`)`** `throws`** `ConnectionError;`**

▸ The UML RT SDK sample has been updated to use these new APIs

HCL SOFTWARE

Usage of CDATA when Storing Models

▶ A new preference can be set to store code snippets and documentation texts using CDATA in the model XML files

▶ This avoids use of XML escape characters which makes such texts more readable when viewed in a text editor

- Example (without CDATA)

```
<headerPreface xmi:id="_irLY4NitEeqes4xYsqINtA"
body="// Some necessary
includes&#xD;&#xA;#include
&lt;iostream>&#xD;&#xA;#include
&lt;string>&#xD;&#xA;&#xD;&#xA;#define OPEN
1&#xD;&#xA;#define CLOSED 0"/>
```

- Example (with CDATA)

```
<headerPreface xmi:id="_irLY4NitEeqes4xYsqINtA">
    <body><![CDATA[// Some necessary includes
#include <iostream>
#include <string>

#define OPEN 1
#define CLOSED 0]]></body>
    </headerPreface>
```

Note: Use of CDATA does not affect backwards compatibility.

Old versions of RTist can still load model files that use CDATA. But if saved in old tool versions, CDATA will of course not be used. A new model fixup is available for converting all model files to use the new file format with CDATA.

HCL SOFTWARE
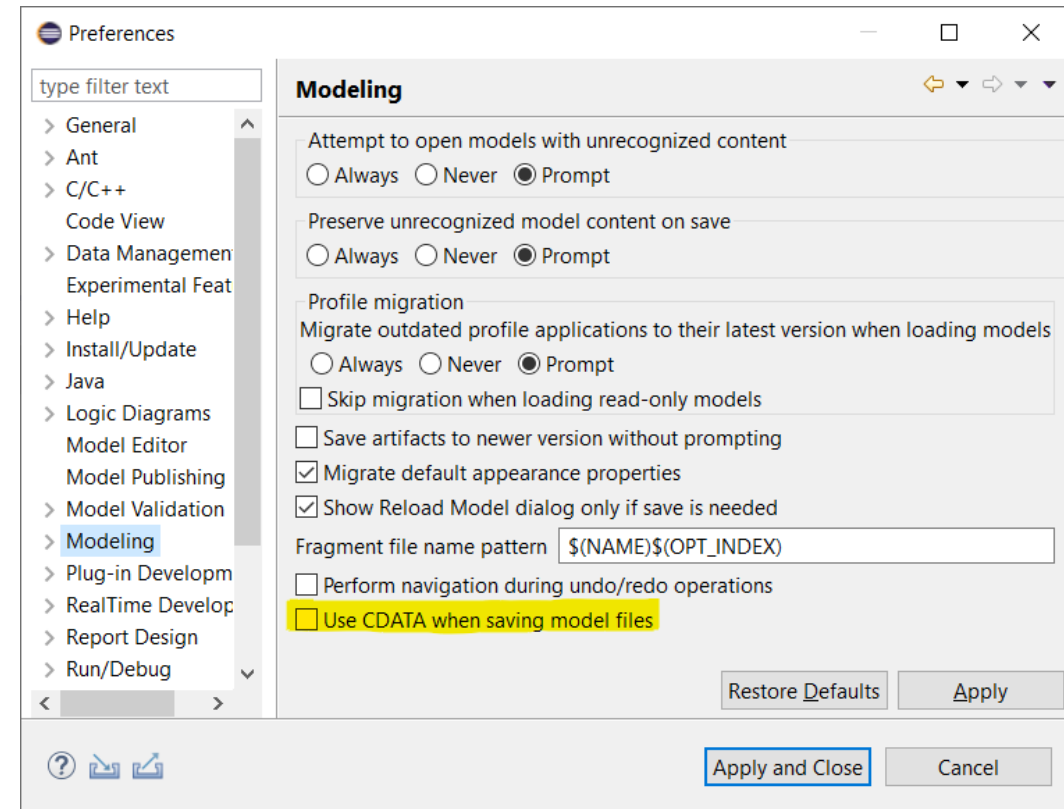
OneTest Embedded Integration (1/2)

▶ A new install component of HCL RTist

  ▪ Note: The OneTest Embedded Integration is not available in IBM RSARTE

▶ Computes and visualizes run-time coverage for UML-RT state machines

  ▪ Which states and transitions have been executed in the state machine?

  ▪ What parts have not executed at all? May indicate too little test coverage.

  ▪ What parts have executed the most? Special focus may be spent on optimizing the performance of those parts.

▶ Based on building an instrumented version of the RTist application that collects coverage data when it runs

  ▪ You can either create special versions of your TC(s) for building an instrumented application, or use build variants which will prompt you each time it's built
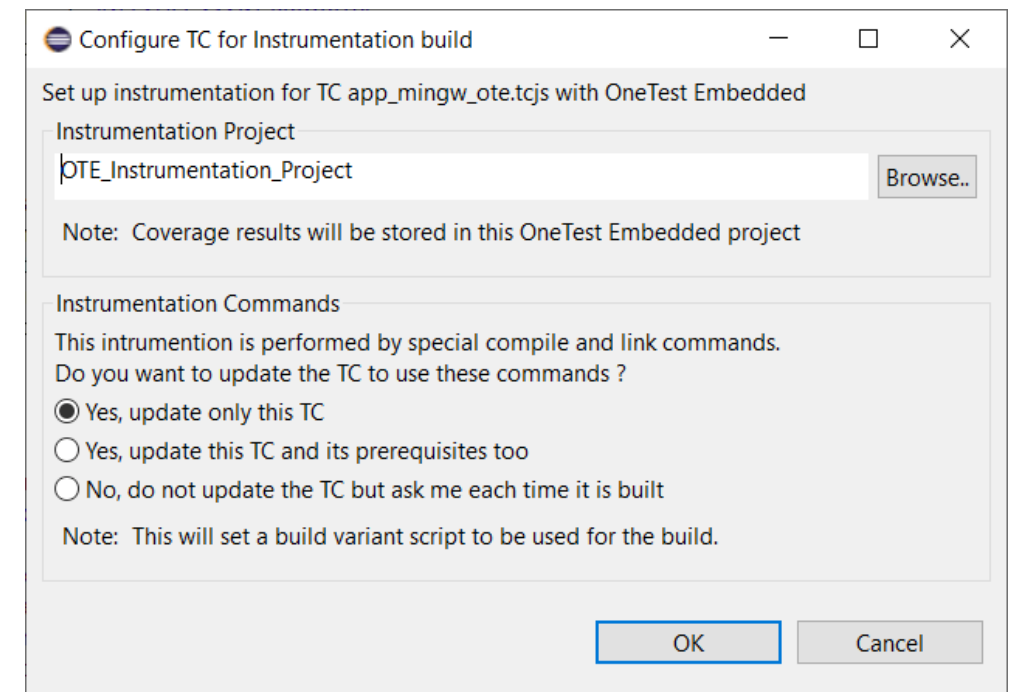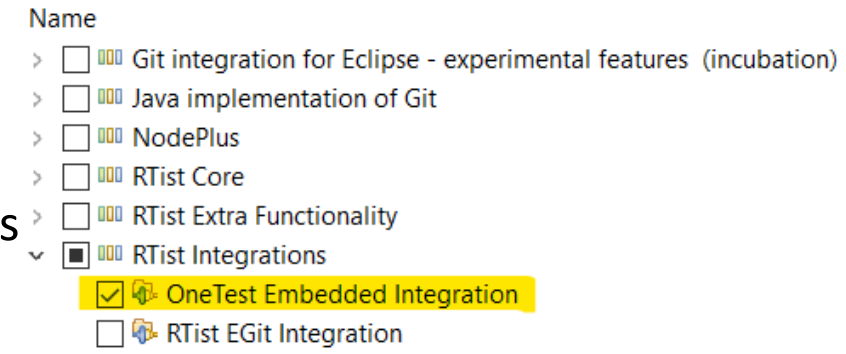
▶ Coverage can be accumulated from several executions of the RT application (e.g. running multiple tests)

- ■ You can look at the combined coverage from multiple executions of the application

▶ Coverage can be shown in a table

- ■ Gives full details of how many times different elements have executed
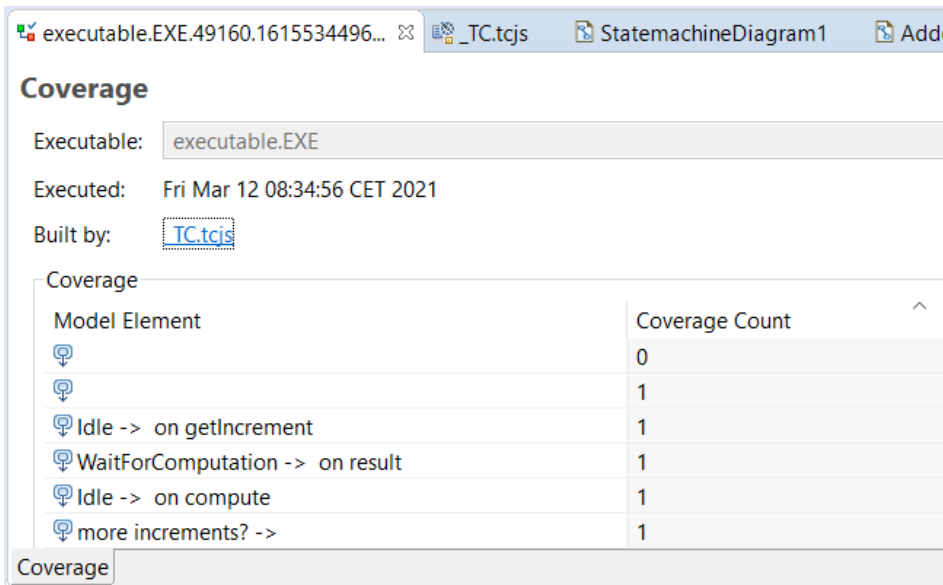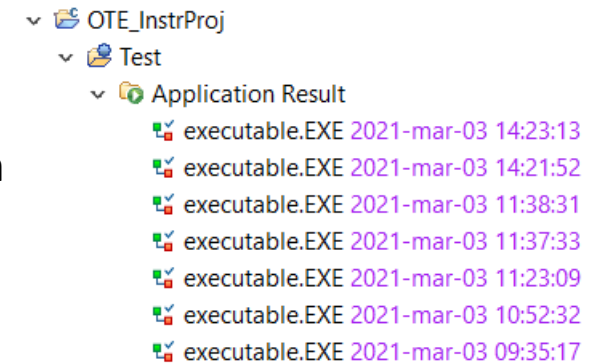
▶ Coverage can be shown in state chart diagrams

- ■ Gives an overview of what elements have or have not executed at least once

▶ This is currently an experimental feature and currently only supported for the MinGW compiler on Windows

48

HCL SOFTWARE
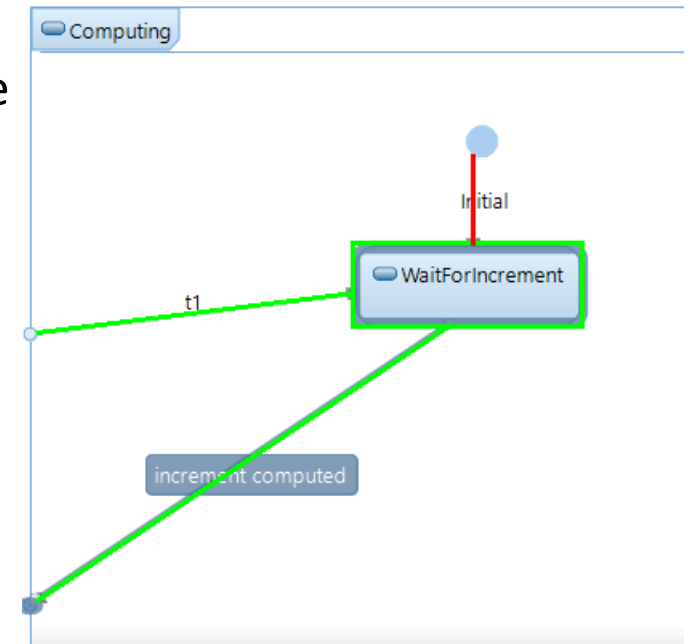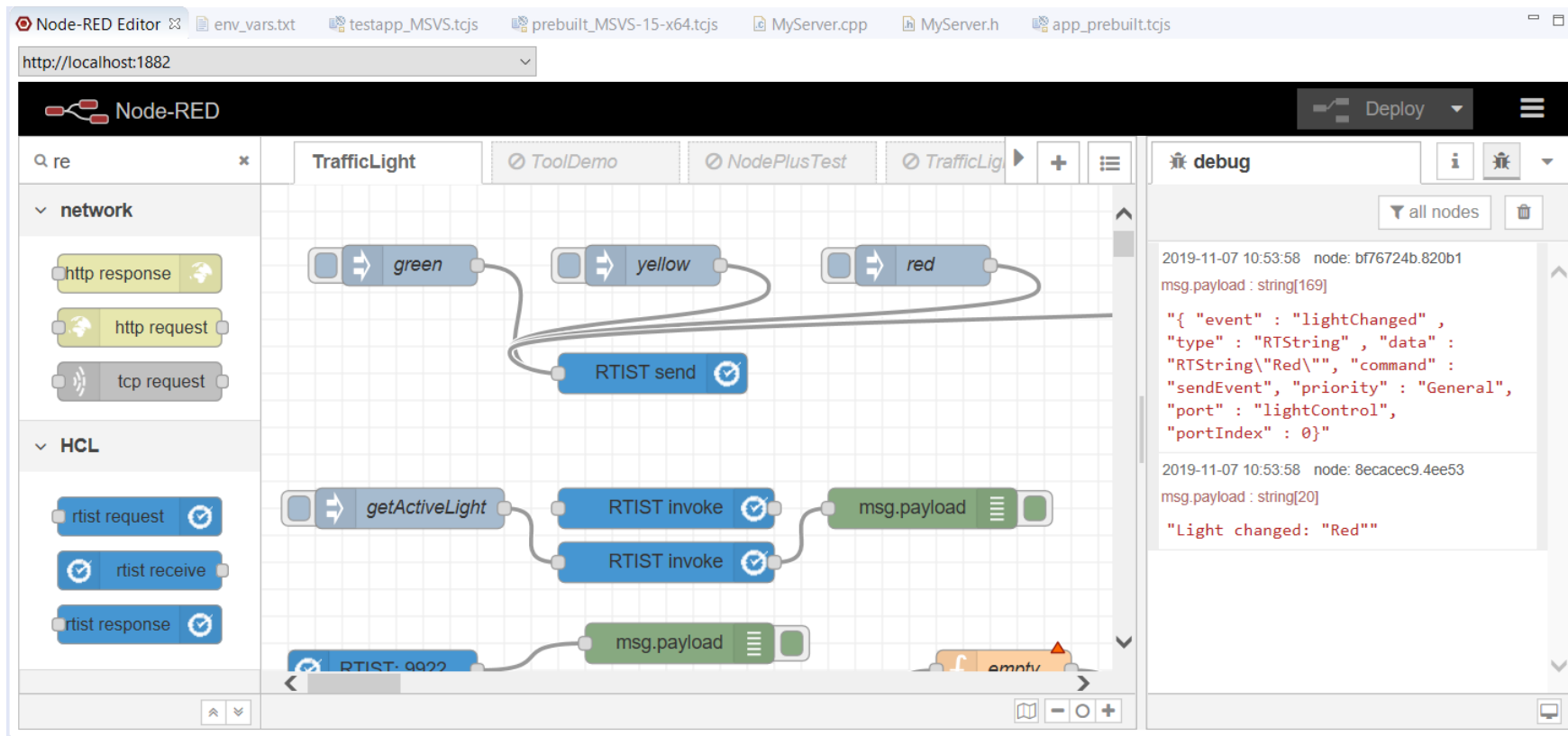
NodePlus

- HCL NodePlus is a new install component of HCL RTist
  - Note: NodePlus is not available in IBM RSARTE

- NodePlus provides all necessary tooling for developing IoT applications using Node-RED (and related technologies)

  - Node-RED projects - both for creating Node-RED nodes and flows of interconnected nodes

  - Node-RED server - for convenient deployment of Node-RED flows within the Eclipse workbench

  - Node.js projects - for creating Node.js applications

  - Node.js server - for convenient deployment of Node.js applications within the Eclipse workbench

  - Swagger - for specifying APIs using Swagger documents

  - Node.js Debugger - for debugging Node.js applications

  - Pug (formerly known as Jade) – for template-based rendering of web pages

  - Unit test

  - Code coverage

Name
- ☐ Git integration for Eclipse
- ☐ Git integration for Eclipse - experimental features (incubation)
- ☐ Java implementation of Git
- ☑ NodePlus
- ☐ RTist Core
- ☐ RTist Extra Functionality
- ☐ RTist Integrations

HCL SOFTWARE
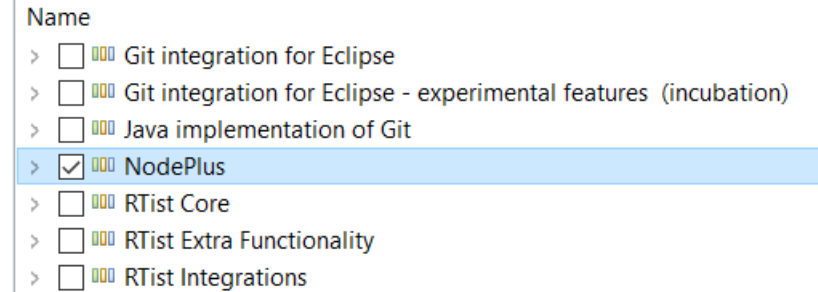
▶ Node-RED editor

- Configure nodes and wire them together into flows
- Deploy changes to the Node-RED server
- View debug output and execution within the flow
- Install new nodes to the palette (e.g. from the public library)

▶ Node-RED Flow Projects

- Flows can be created directly in the Node-RED editor, but creating them as Eclipse projects has many benefits (e.g. team development using SCM integration)

- An intuitive wizard for creating Node-RED Flow projects

- Either create a flow from scratch or start from one of the existing flows in the public library
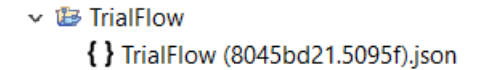
▶ Node-RED Server

- A dedicated Servers view lets you create, start and stop Node-RED local and remote servers within the Eclipse workbench

- Deploy a Node-RED flow onto a selected server and open its Node-RED editor

Node-RED Tools in NodePlus
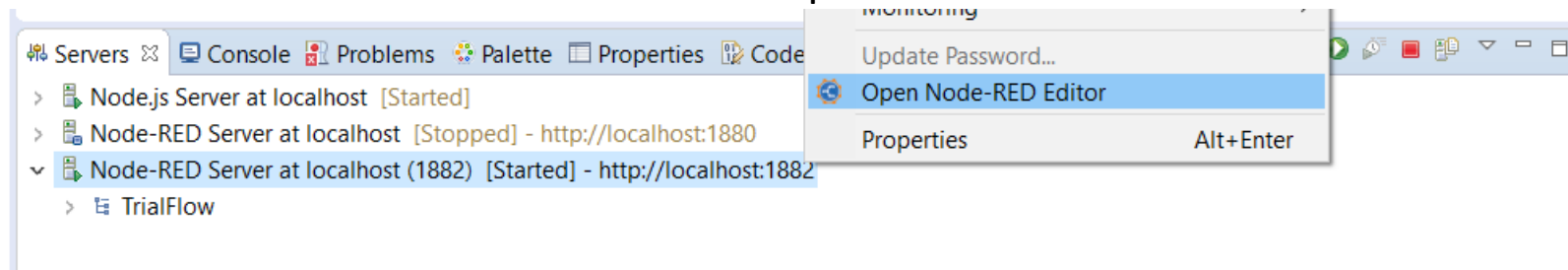
▸ Remote Node-RED Server

- You don't need to run the Node-RED server locally. You can instead connect to a server that runs remotely (for example on a server or on a device)

- Create a new server in the Servers view and enter the URL to the remote server

- You can then connect to the server in order to open the Node-RED editor for it (other operations on a remote server are currently not supported).

HCL SOFTWARE

Node-RED Tools in NodePlus
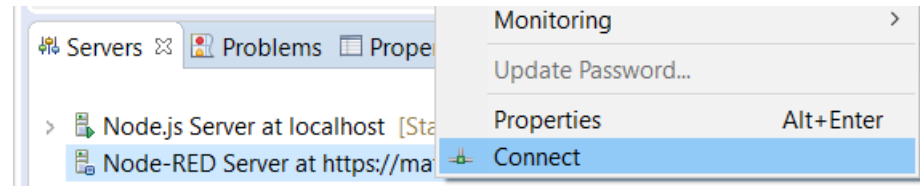
▸ Node-RED Node Projects
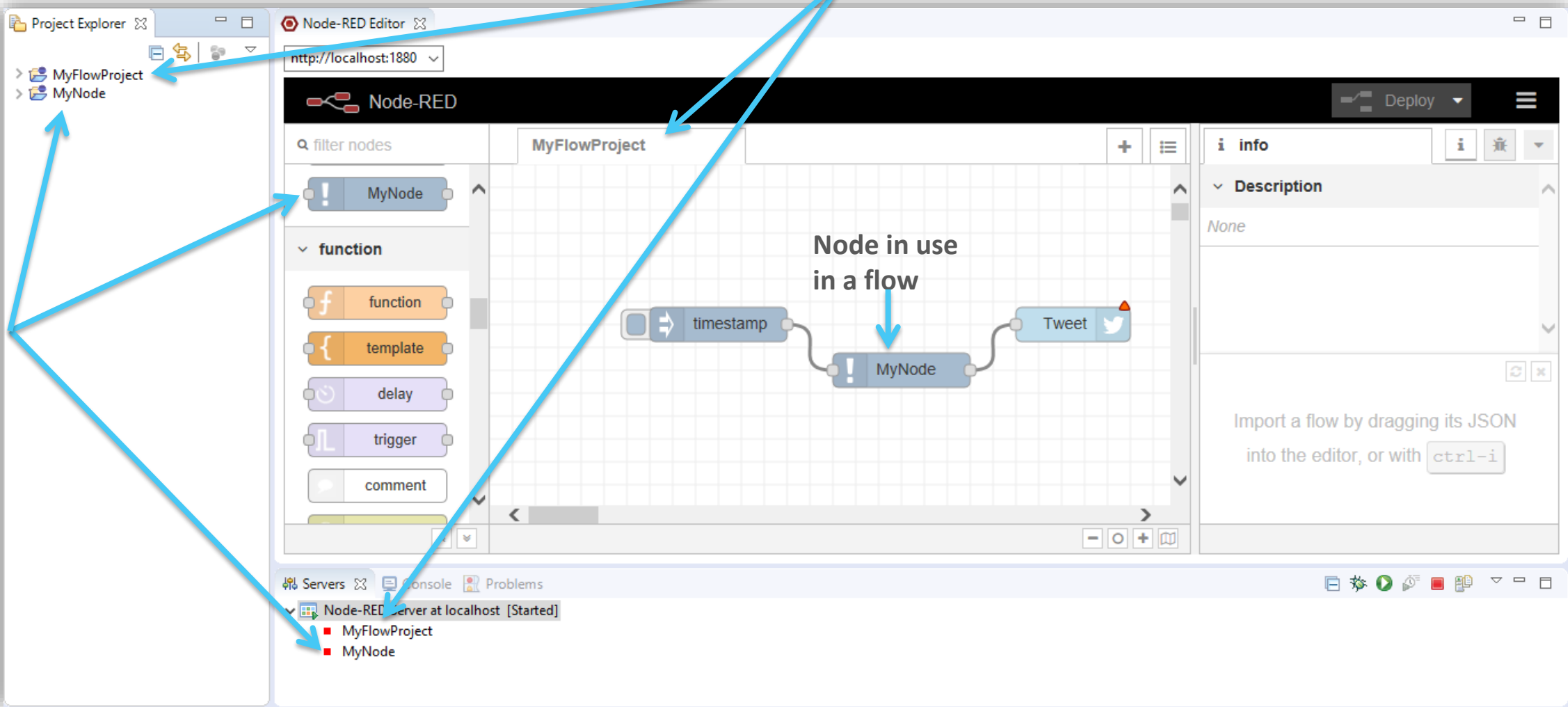
- More than 2500 nodes exist in the public library, but sometimes you may need to create you own specific node

- An intuitive wizard for creating Node-RED Node projects

- Benefit from all the usual Eclipse features when developing your node (e.g. SCM integration)

- Either create a node from scratch or start from one of the existing nodes in the public library

▸ Unit test of Node-RED nodes

- Unit tests are created using the Mocha framework and results are shown in a special Node-RED Unit Test view

Custom "Unit Test" view

53

HCL SOFTWARE

Node-RED Tools in NodePlus

Flow project

Node project

Node in use in a flow

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

HCL SOFTWARE

▶ Nodes for communicating with an RTist application

- **rtist request**
  Let a flow send or invoke an event into an RTist application

- **rtist receive**
  Trigger a flow when receiving an event that is sent or invoked from an RTist application

- **rtist response**
  Reply to a received event that is sent or invoked from an RTist application

▶ These nodes make use of the JSON API provided by lib-tcp-server (i.e. the RTist application must use this library)

- The nodes can be statically configured in the Node-RED editor

- It's also possible to provide many of the node properties in the message payload for a more dynamic behavior

replying on received events

receiving events sent or invoked on a port

sending and invoking events on a port
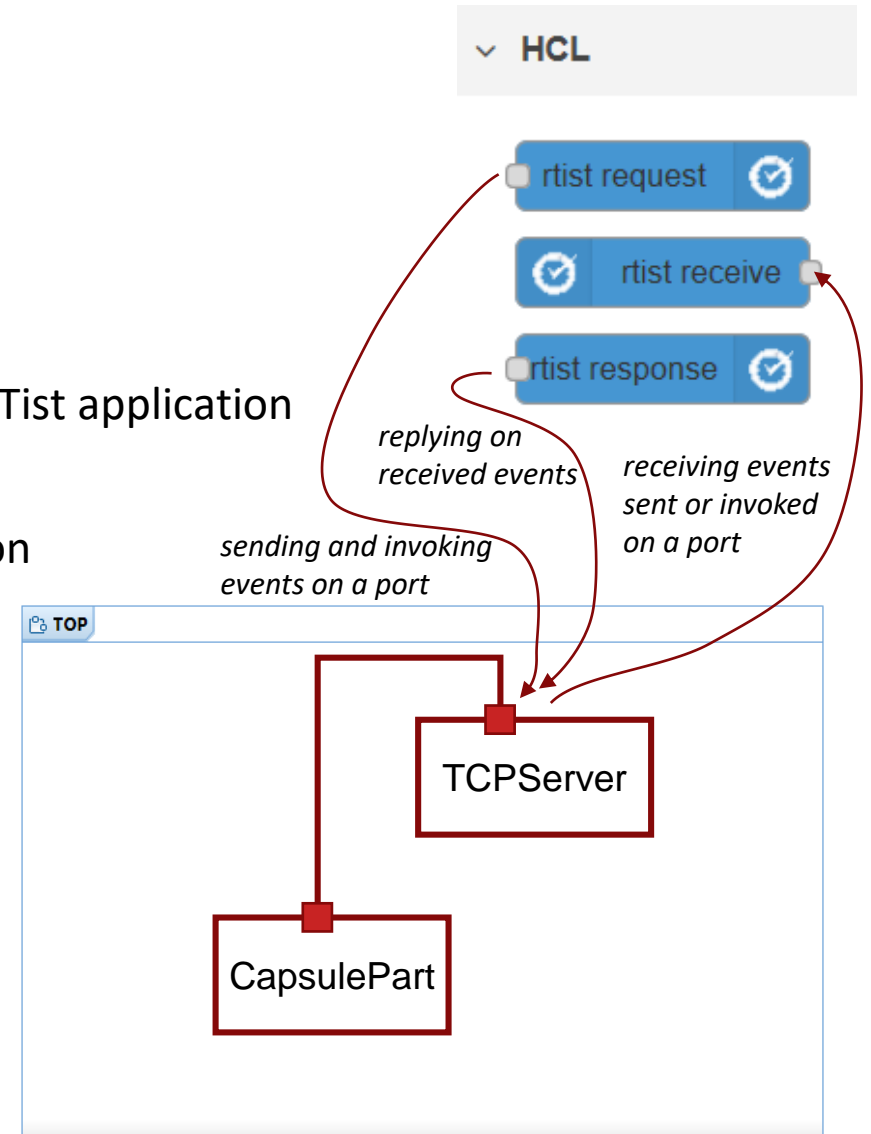
Swagger Support in NodePlus

▶ Document APIs using Swagger in a new Swagger editor

Link source and design panes

Support for JSON and YAML syntax

Syntax highlight and content assist

Visual API representation

Support for executing / testing API

Automated generation of example data

Content types selections

Swagger Code Generator

▶ Create development projects in different languages, such as Java, JavaScript, NodeJS, Python, etc. from a swagger specification with the Swagger Code Generator Wizard

HCL SOFTWARE

Node.js Tools in NodePlus
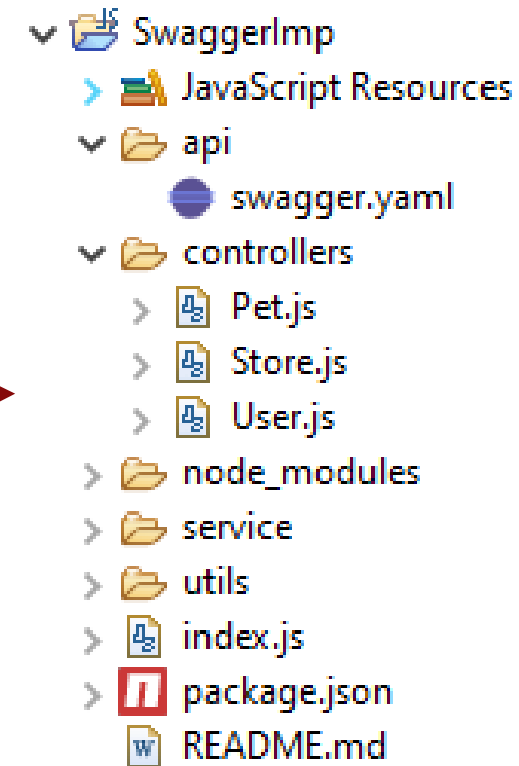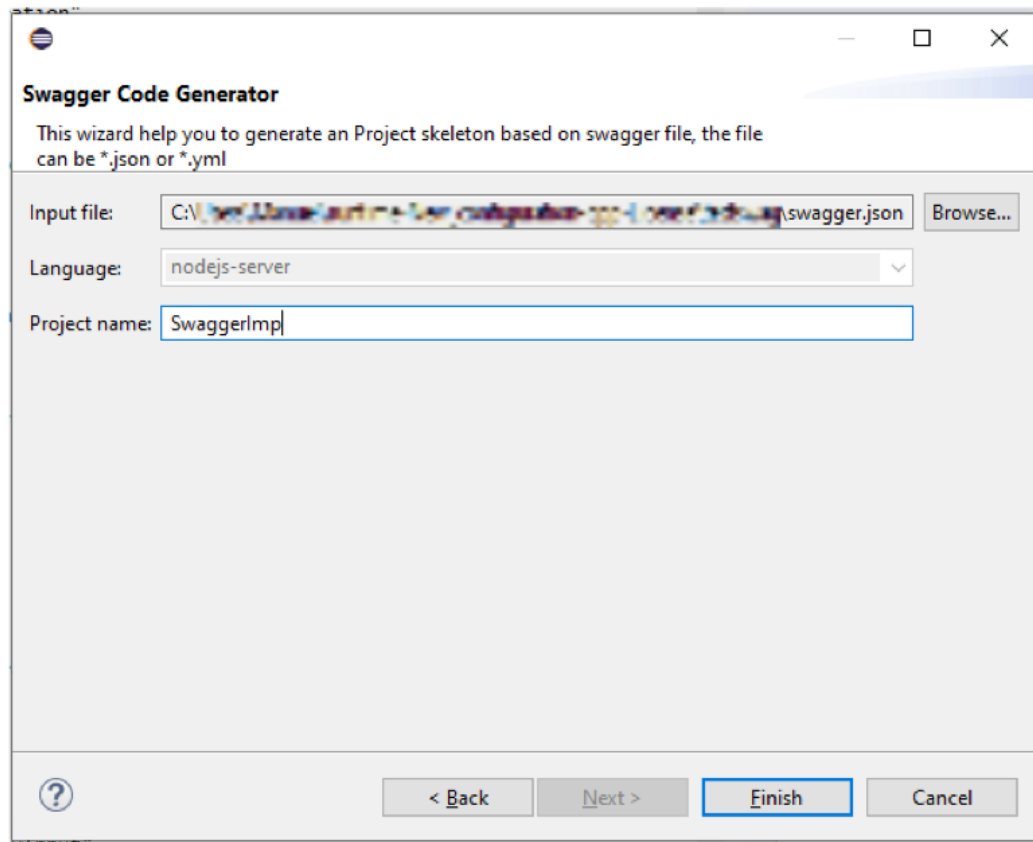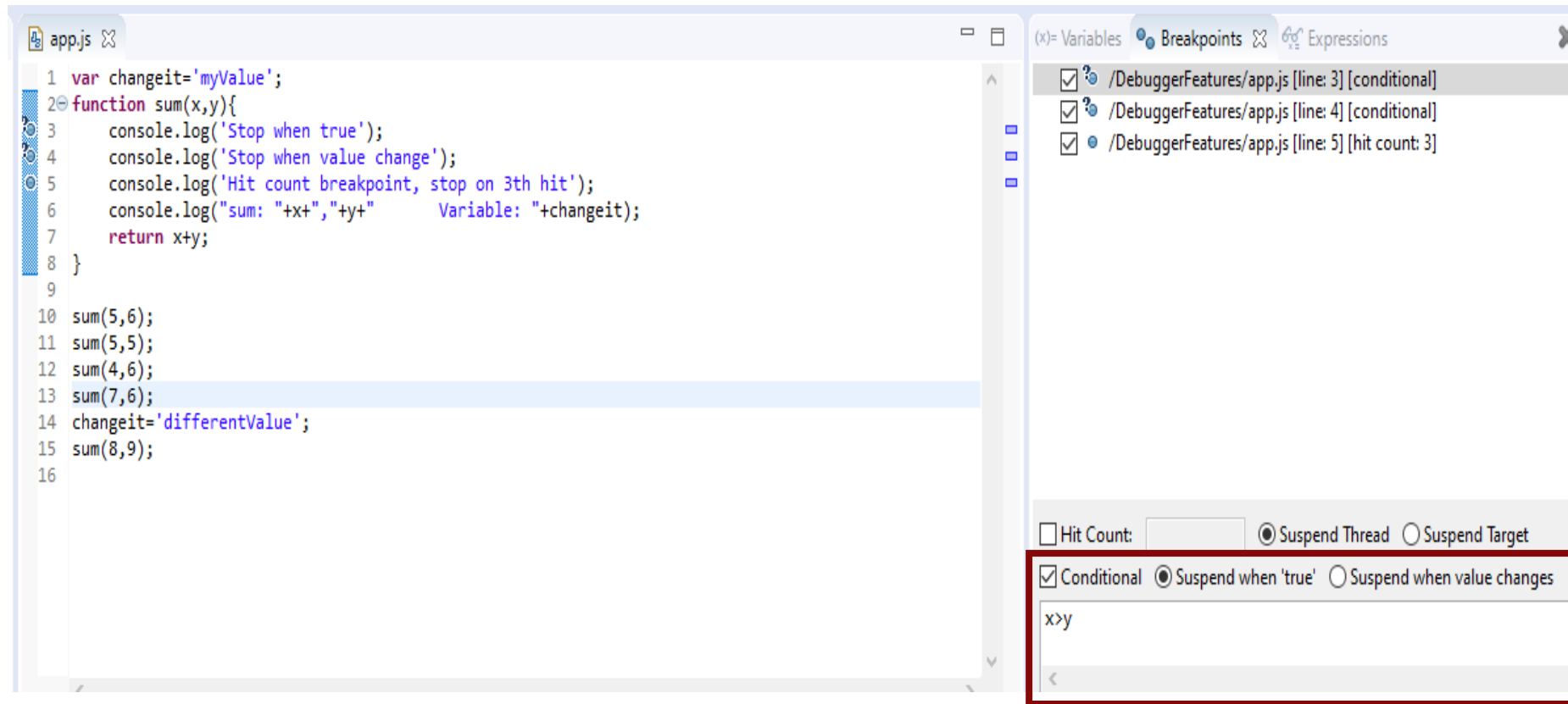
▶ Wizard for creating Node.js projects

▶ Create, start and stop a Node.js server from within the Eclipse workbench

▶ Debug Node.js applications with a modern JavaScript debugger

▶ New editor for developing HTML pages for your Node.js application using the Pug template engine

HCL SOFTWARE

Node.js Tools in NodePlus
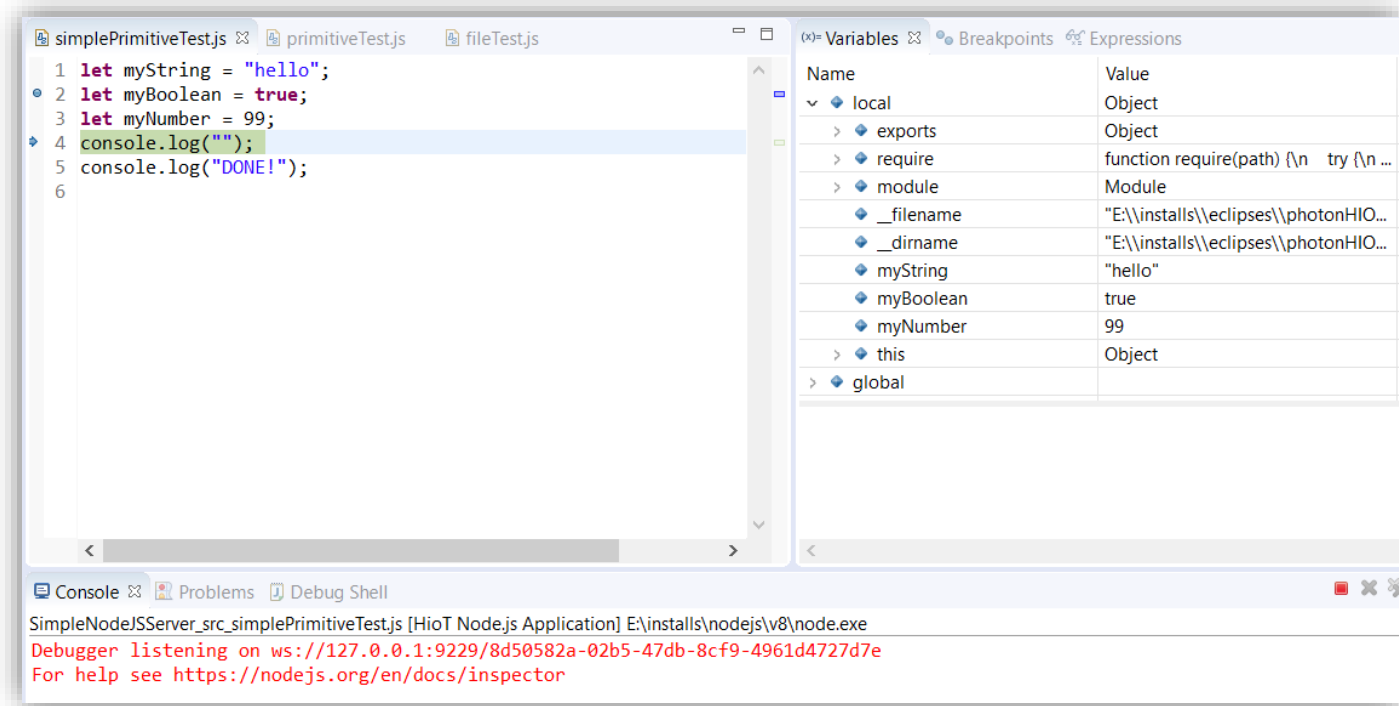
▸ Deploy NPM packages into the NPM repository

▸ Deprecate / Unpublish NPM packages from the NPM repository
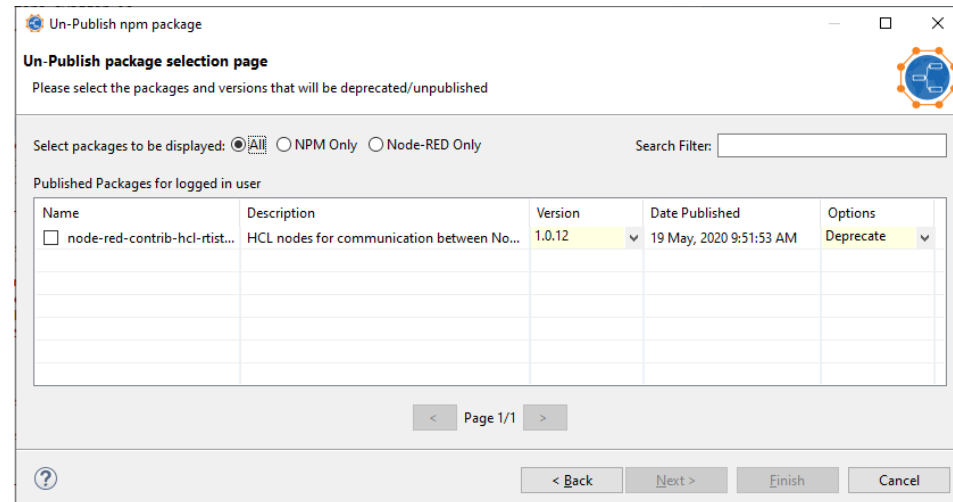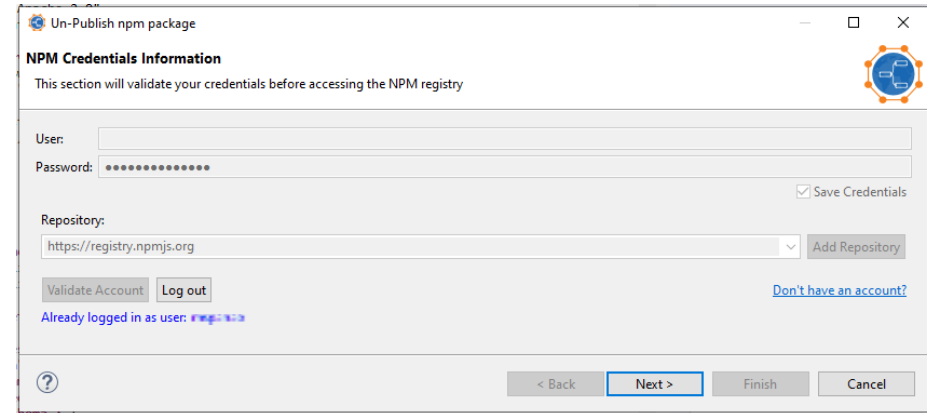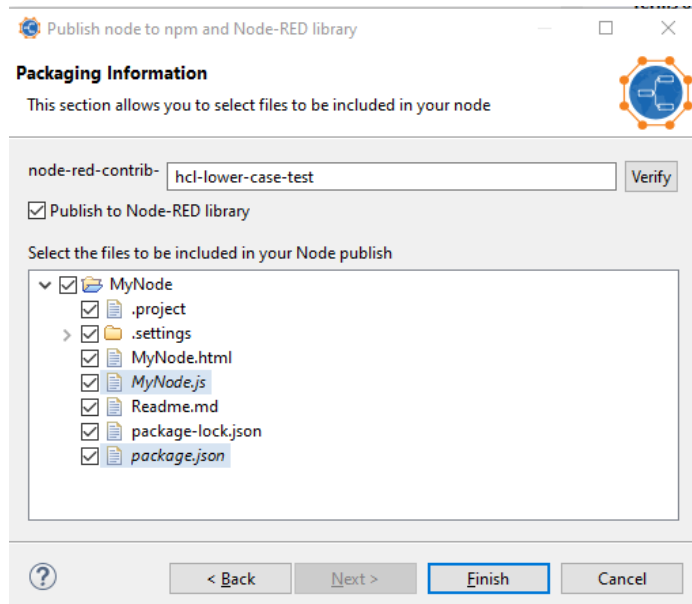
Node.js Debugger

▸ Support for conditional breakpoints
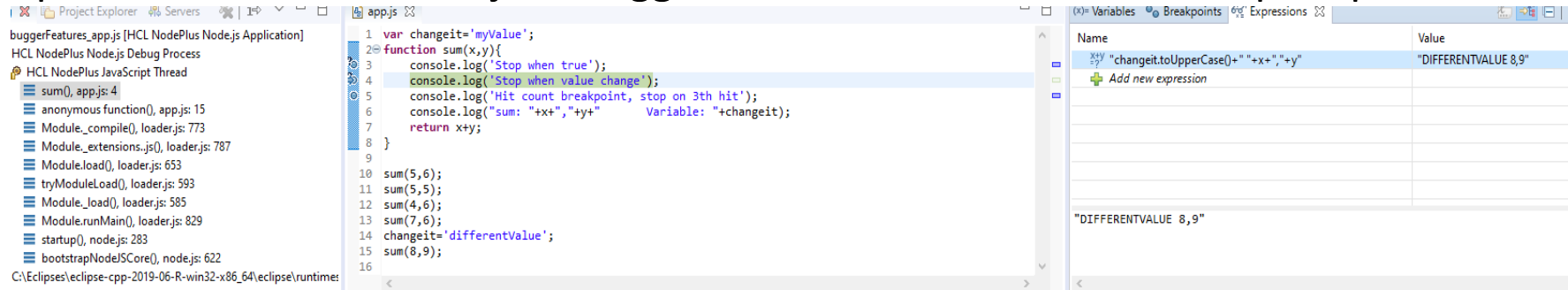
   ▪ The Node.js debugger can be set to suspend when the conditional expression is true or when it changes

HCL SOFTWARE

Node.js Debugger

▸ Support for Hit Count breakpoints

- The Node.js debugger can be set to suspend when a breakpoint has been hit a certain number of times

HCL SOFTWARE
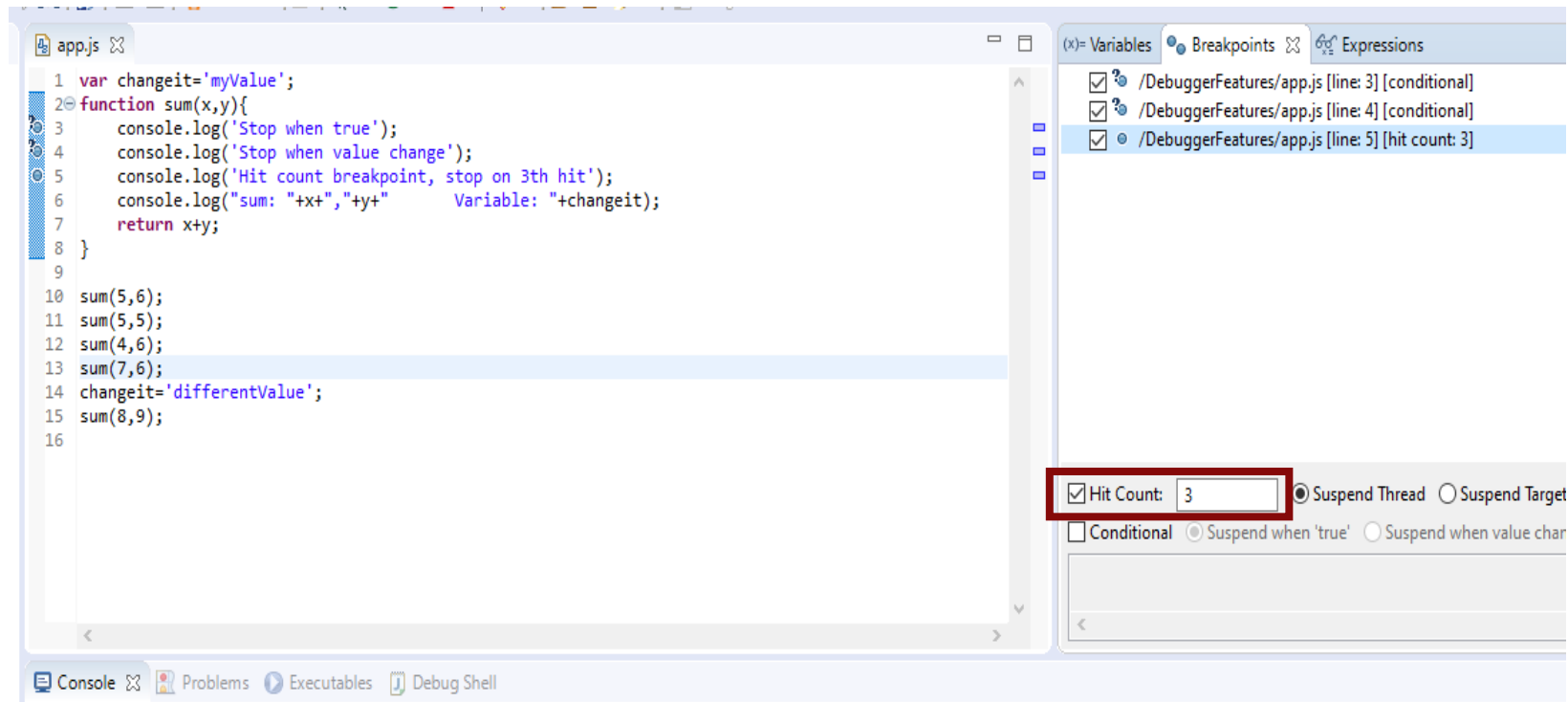
Node.js Debugger

▶ In the Expressions tab the Node.js debugger can now evaluate a JavaScript expression in real time
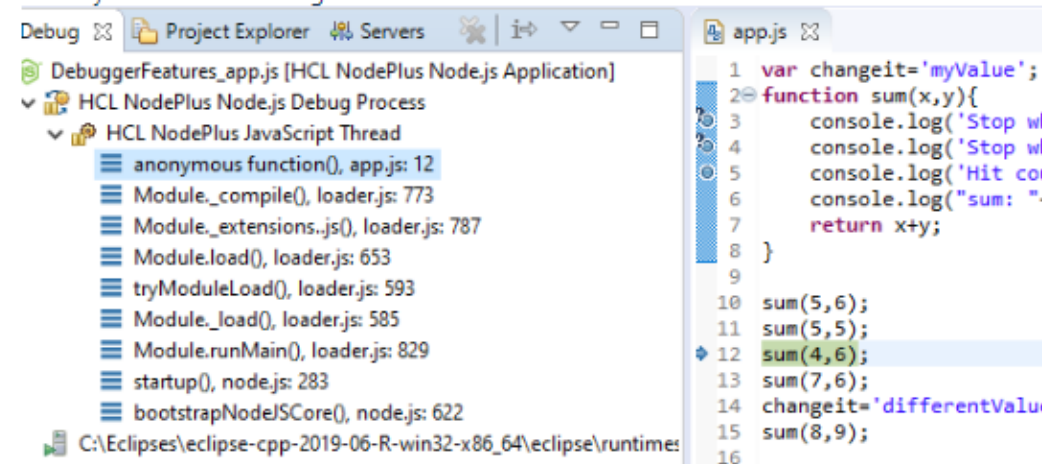

▶ Support for "Drop to Frame"

▪ When the application is suspended you can select a stack frame and press the "Drop to Frame" button to restart execution from there

HCL SOFTWARE

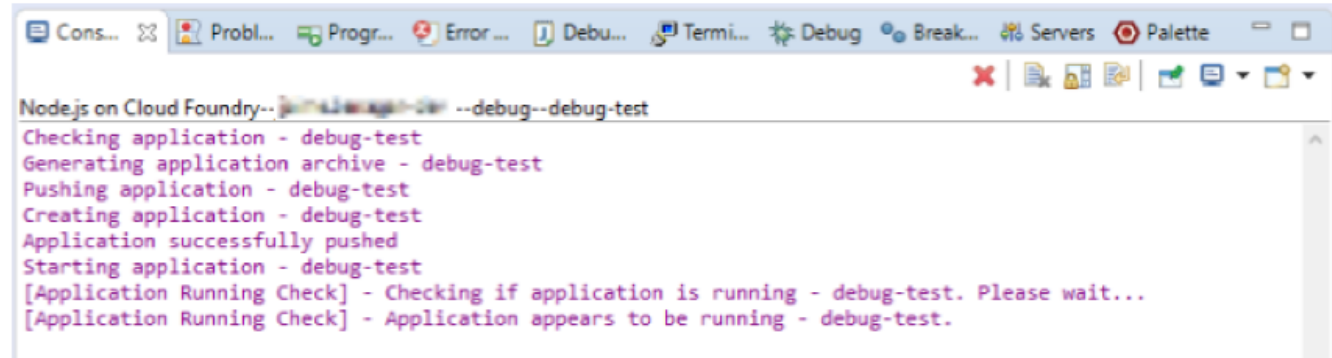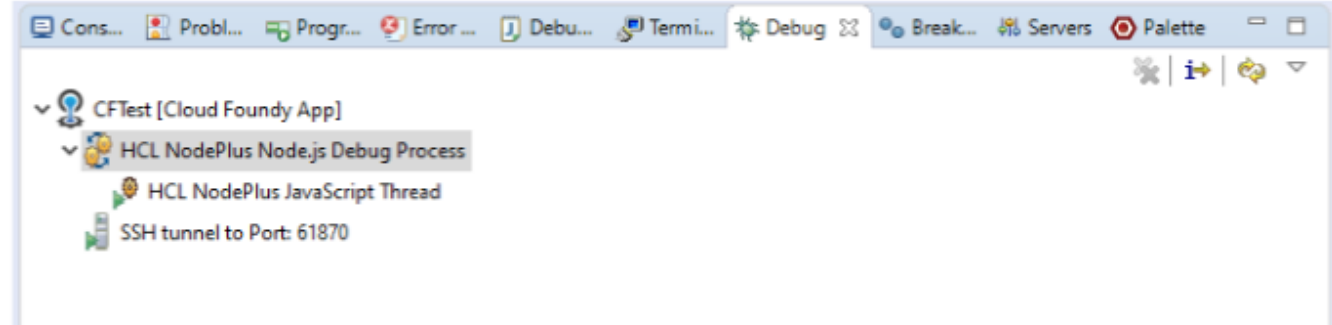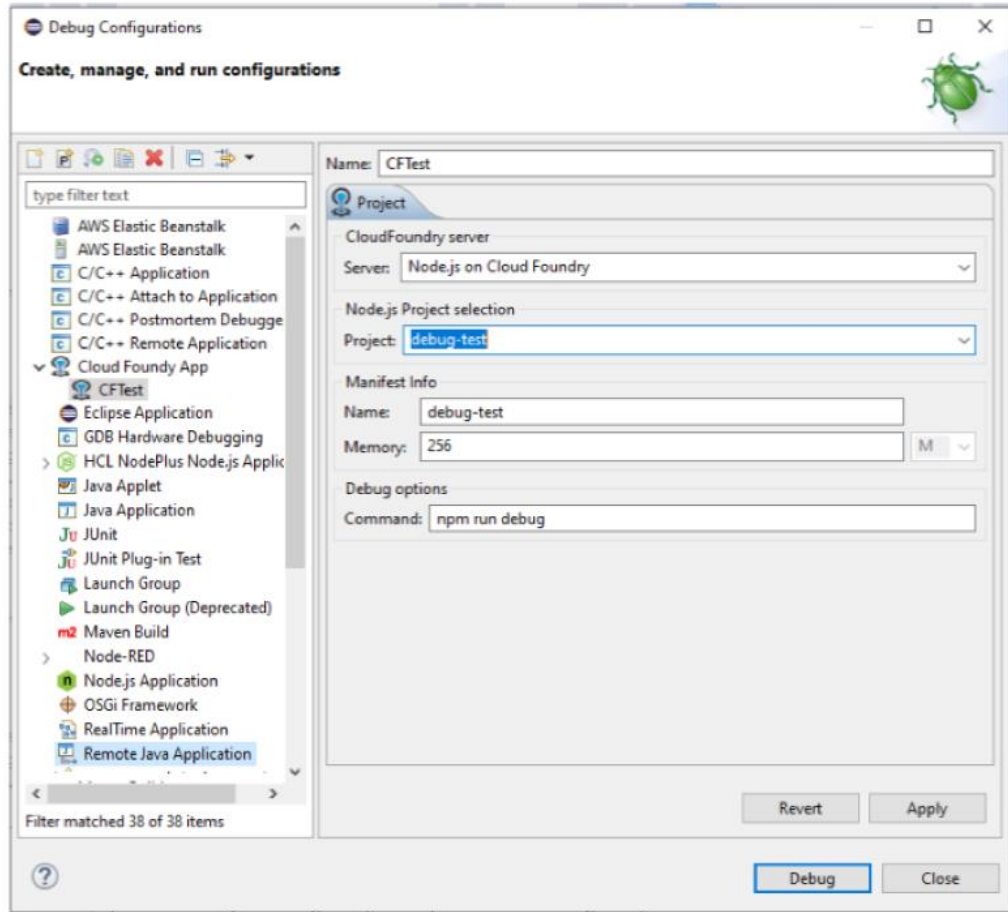▶ You can use NodePlus to debug nodes deployed into Node-RED by enabling 'run in debug' mode
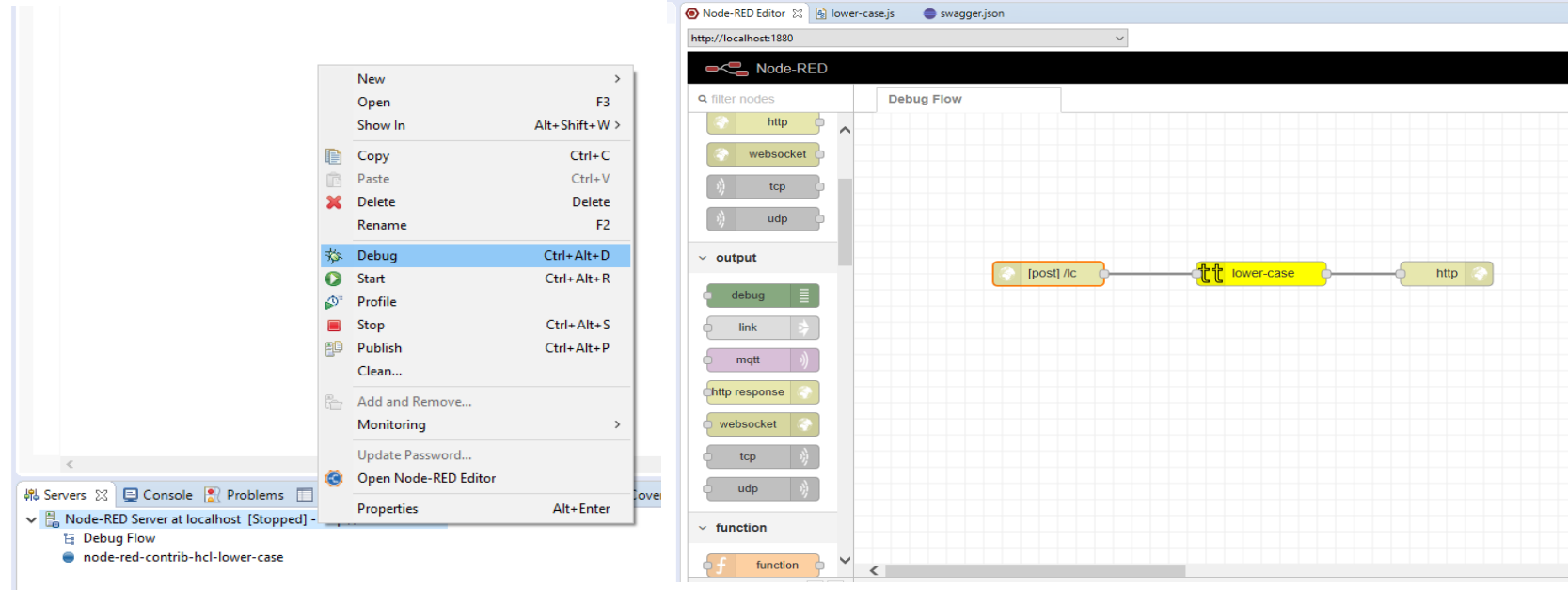
Debug Node.js Applications on Cloud Foundry

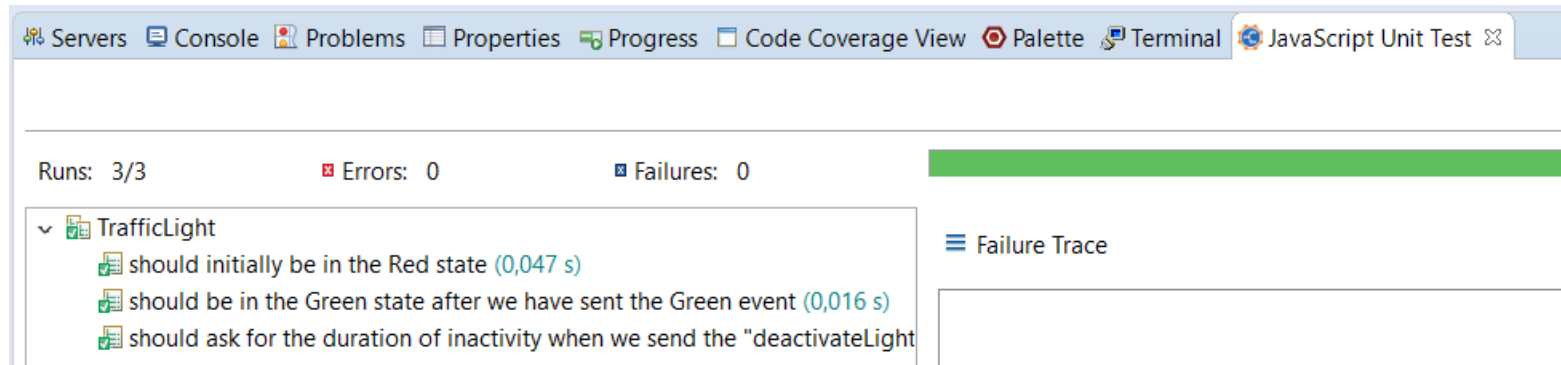▸ You can now debug Node.js applications that are running on Cloud Foundry

Unit Testing with Mocha

▶ Node-RED nodes and Node.js applications can be unit tested using the Mocha framework

 ▪ Unit tests can also be debugged

🔆 Debug As	>	📜 1 HCL NodePlus Node.js Application Alt+Shift+D, N
Profile As	>	⚙ 2 JavaScript Unit Test
V8 Debugging	>	🅝 3 Node.js Application
📜 JavaScript Tools	>	Debug Configurations…
⚙ Node.js	>	

▶ Using the TCP Server library in an RTist application makes it possible to test it too with Mocha

 ▪ An example is here https://github.com/hcl-pnp-rtist/rt-test-probe

▶ Results from unit testing are shown in a JavaScript Unit Test view

🔆 Servers 🖥 Console 👤 Problems 🔲 Properties 🔳 Progress 🔲 Code Coverage View ⊙ Palette 🔌 Terminal ⚙ JavaScript Unit Test ✕

Runs: 3/3   ⊠ Errors: 0   ⊠ Failures: 0

∨ 🗒 TrafficLight
   🗒 should initially be in the Red state (0,047 s)
   🗒 should be in the Green state after we have sent the Green event (0,016 s)
   🗒 should ask for the duration of inactivity when we send the "deactivateLight

≡ Failure Trace

HCL SOFTWARE

HCL

Relationship™
BEYOND THE CONTRACT

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

▶ WATCH THE FILM