


# What's New in HCL RTist 11

updated for release 2020.14

# Overview

- ▶ RTist 11 is based on Eclipse 2019.06 (4.12)
- ▶ HCL RTist is 100% compatible with IBM RSARTE. All features (with very few exceptions) are the same.

 HCL RealTime Software Tooling

Version: 11.0.0.v20200402\_0936

Release: 2020.14-ga

(c) Copyright IBM Corporation 2004, 2016. All rights reserved.

(c) Copyright HCL Technologies Ltd. 2016, 2020. All rights reserved.

Visit <https://RTist.hcldoc.com/help/topic/com.ibm.xtools.rsarte.webdoc/users-guide/overview.html>

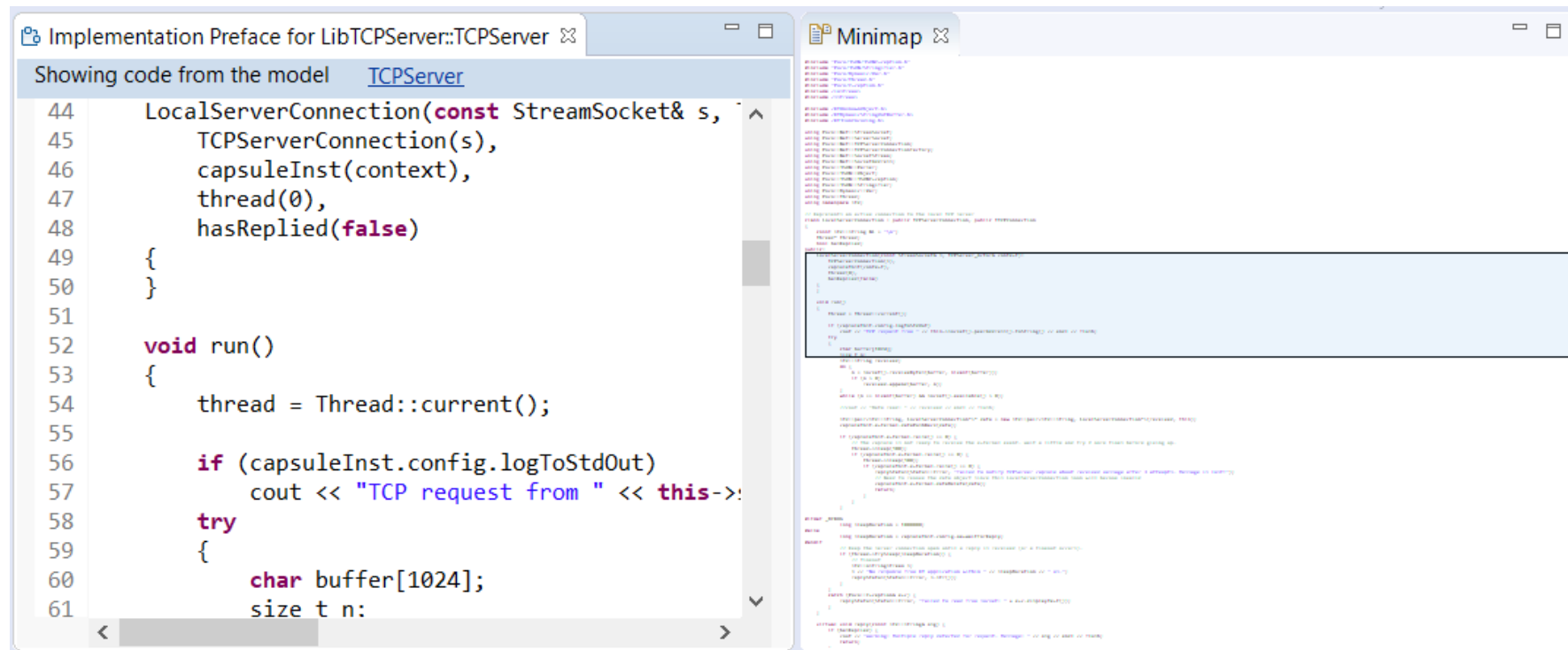


# Eclipse 4.12 (2019.06)

- ▶ 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...

# 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

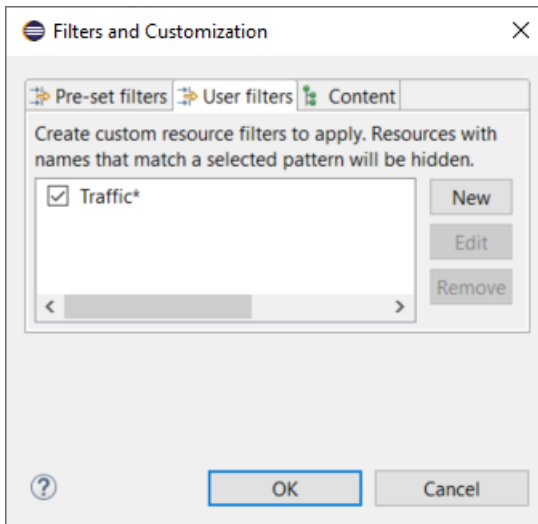
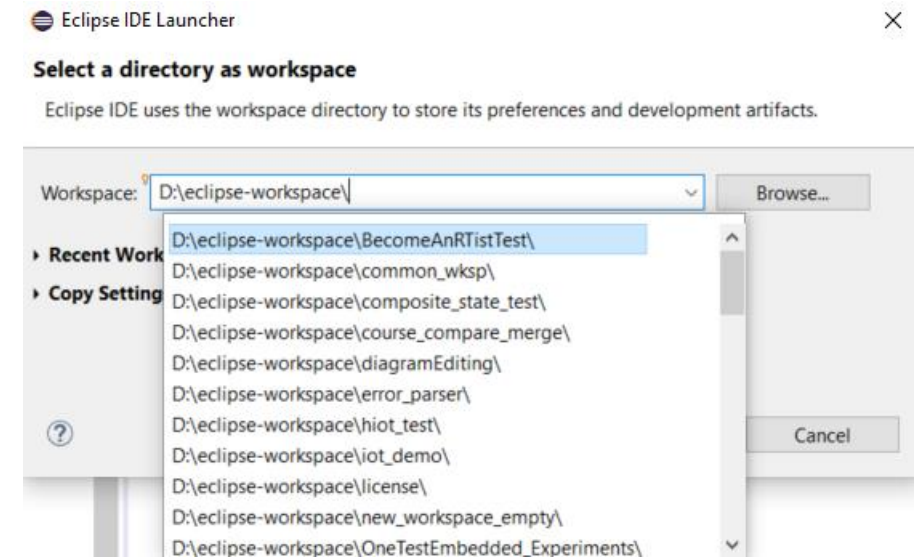


The screenshot displays the Eclipse IDE interface. On the left, a code editor window titled 'Implementation Preface for LibTCPServer::TCPServer' shows C++ code for a TCP server. The code includes a constructor and a 'run()' method. On the right, a 'Minimap' view provides a visual overview of the code, with a red box highlighting a specific section of the code.

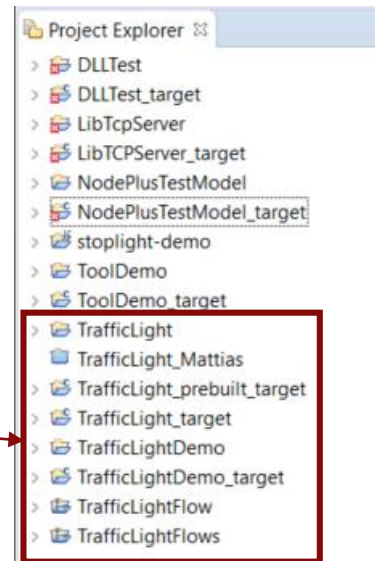
```
44     LocalServerConnection(const StreamSocket& s,  
45         TCPServerConnection(s),  
46         capsuleInst(context),  
47         thread(0),  
48         hasReplied(false)  
49     {  
50     }  
51  
52     void run()  
53     {  
54         thread = Thread::current();  
55  
56         if (capsuleInst.config.logToStdOut)  
57             cout << "TCP request from " << this->  
58         try  
59         {  
60             char buffer[1024];  
61             size_t n:
```

# 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



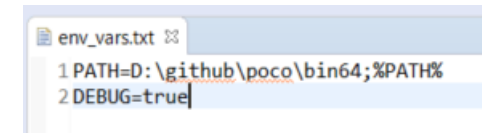
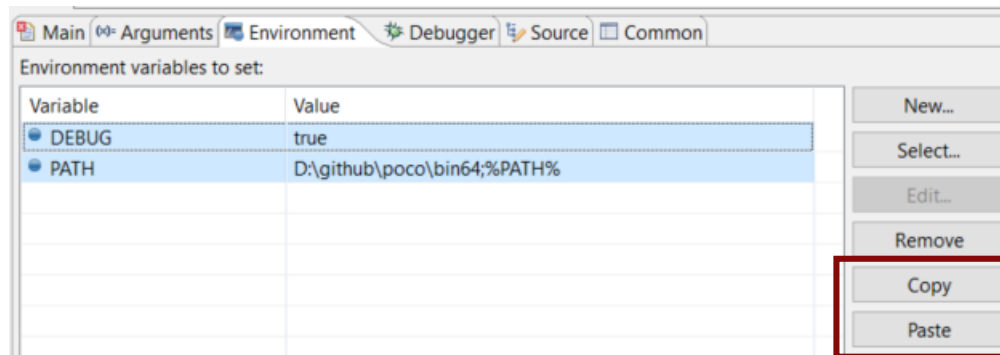
will be hidden  
by the user filter



# Eclipse 4.12 (2019.06)

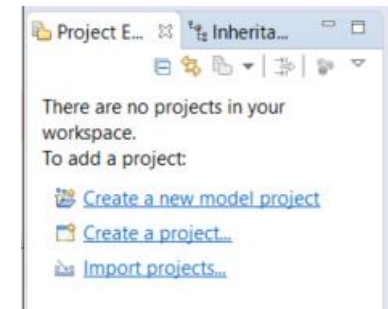
## ▶ Copy/Paste of environment variables

- 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)



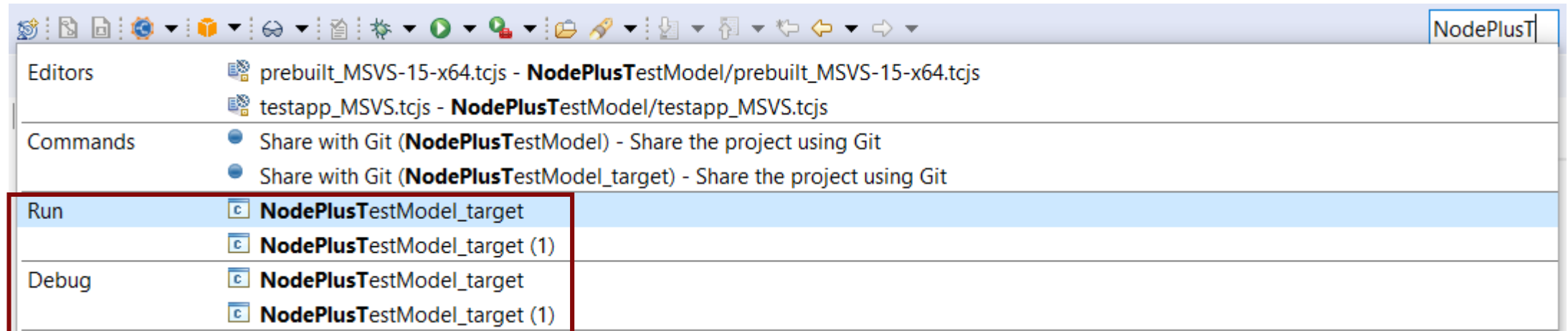
## ▶ Useful quick links for an empty workspace

- 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



# Eclipse 4.12 (2019.06)

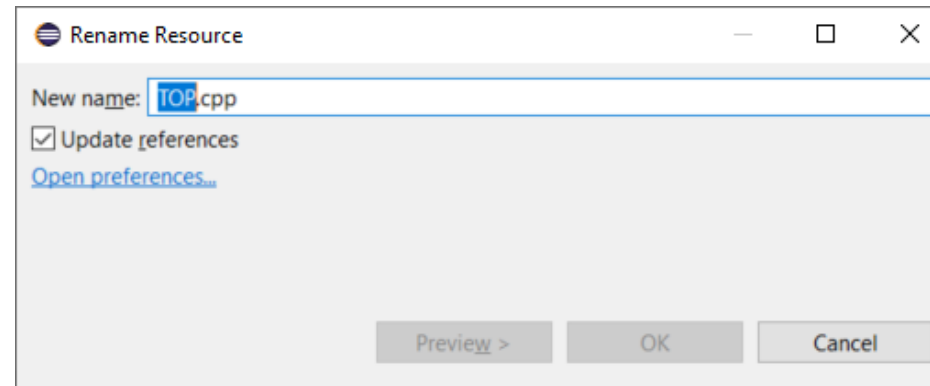
- ▶ Launch configurations are now accessible from Quick Access
  - Can start either a Run or a Debug session



# CDT 9.8 (included as part of Eclipse 2019.06)

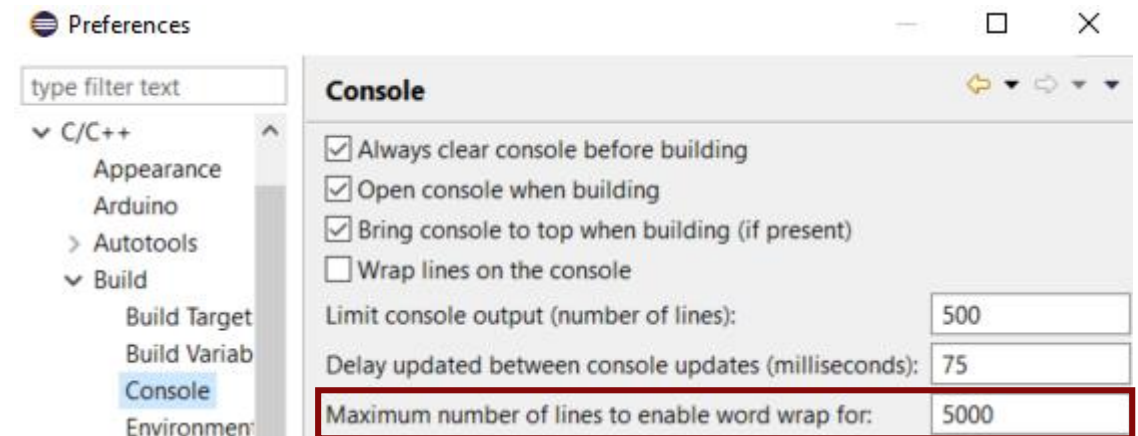
- ▶ Possible to rename a file without triggering a refactoring

- 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

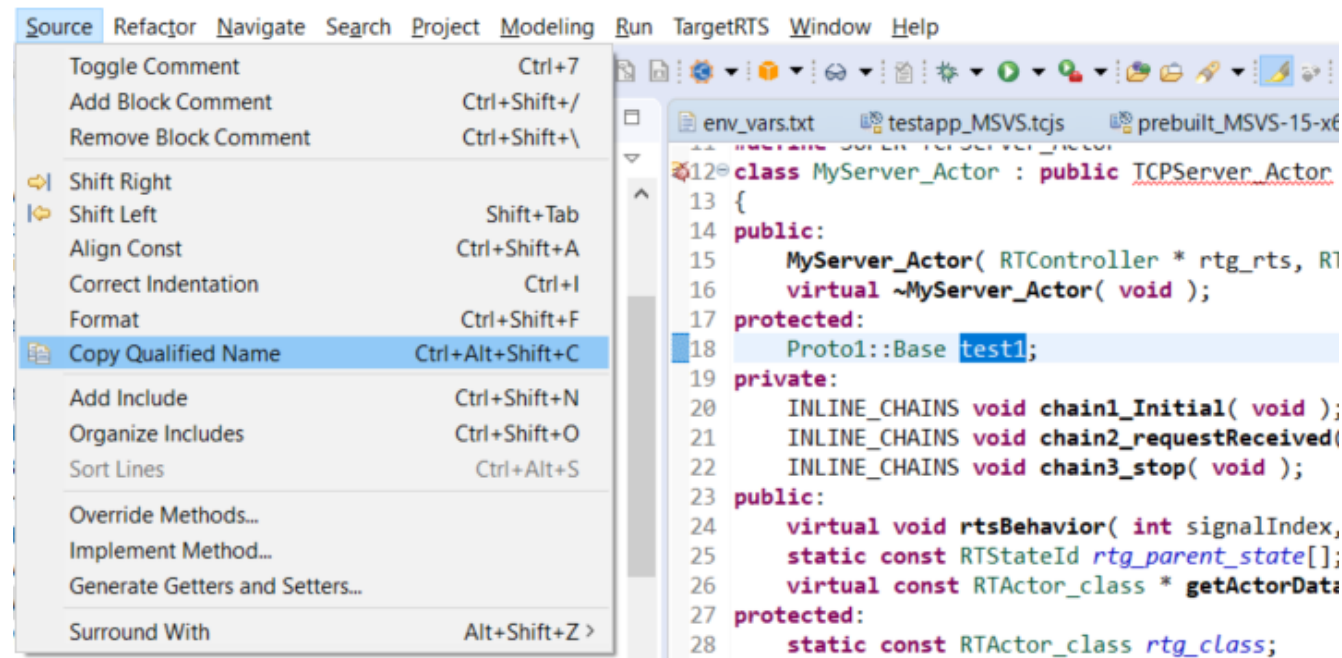
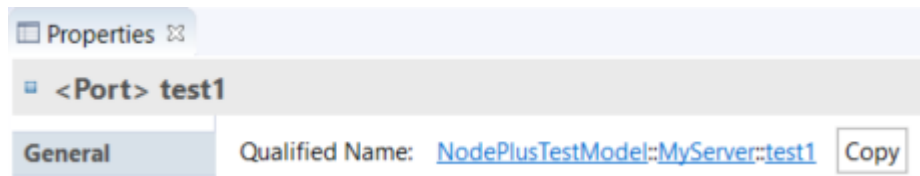
- *C/C++ - Build – Console – Maximum number of lines to enable word wrap for*
- Don't set this too high if *Wrap lines on the console* is turned on





# 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



# CDT 9.8 (included as part of Eclipse 2019.06)

---

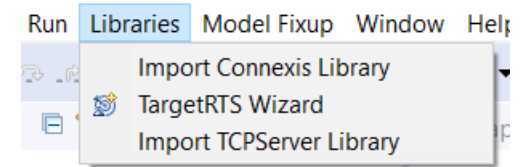
- ▶ CODAN improvements
  - Additional checks implemented, for example to detect C-style casts and goto-statements
- ▶ For more information about CDT improvements see
  - <https://wiki.eclipse.org/CDT/User/NewIn96>
  - <https://wiki.eclipse.org/CDT/User/NewIn97>
  - <https://wiki.eclipse.org/CDT/User/NewIn98>

# 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/5.1)
    - [https://wiki.eclipse.org/EGit/New\\_and\\_Noteworthy/5.2](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/5.3)
    - [https://wiki.eclipse.org/EGit/New\\_and\\_Noteworthy/5.4](https://wiki.eclipse.org/EGit/New_and_Noteworthy/5.4)








# 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



# NodePlus

- ▶ HCL NodePlus is a new install component of HCL RTist
  - Note: Currently 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 webpages
  - Unit test

Name
> <input type="checkbox"/>  Git integration for Eclipse
> <input type="checkbox"/>  Git integration for Eclipse - experimental features (incubation)
> <input type="checkbox"/>  Java implementation of Git
> <input checked="" type="checkbox"/>  NodePlus
> <input type="checkbox"/>  RTist Core
> <input type="checkbox"/>  RTist Extra Functionality
> <input type="checkbox"/>  RTist Integrations

# Node-RED Tools in NodePlus

## ▶ Node-RED editor

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

The screenshot displays the Node-RED editor interface. The top bar shows the URL `http://localhost:1882` and a `Deploy` button. The main workspace is titled `TrafficLight` and contains a flow with several nodes:

- `green`, `yellow`, and `red` nodes (light color) are connected to an `RTIST send` node (blue).
- A `getActiveLight` node (light blue) is connected to two `RTIST invoke` nodes (blue).
- The `RTIST invoke` nodes are connected to `msg.payload` nodes (green).
- There is also an `RTIST-9922` node (blue) connected to a `msg.payload` node (green).

The right-hand side of the interface shows a `debug` console with the following output:

```
2019-11-07 10:53:58 node: bf76724b.820b1
msg.payload: string[169]
{"event": "lightChanged",
"type": "RTString", "data":
"RTString\\Red\\", "command":
"sendEvent", "priority": "General",
"port": "lightControl",
"portIndex": 0}
2019-11-07 10:53:58 node: 8ecacec9.4ee53
msg.payload: string[20]
"Light changed: "Red""
```

# Node-RED Tools in NodePlus

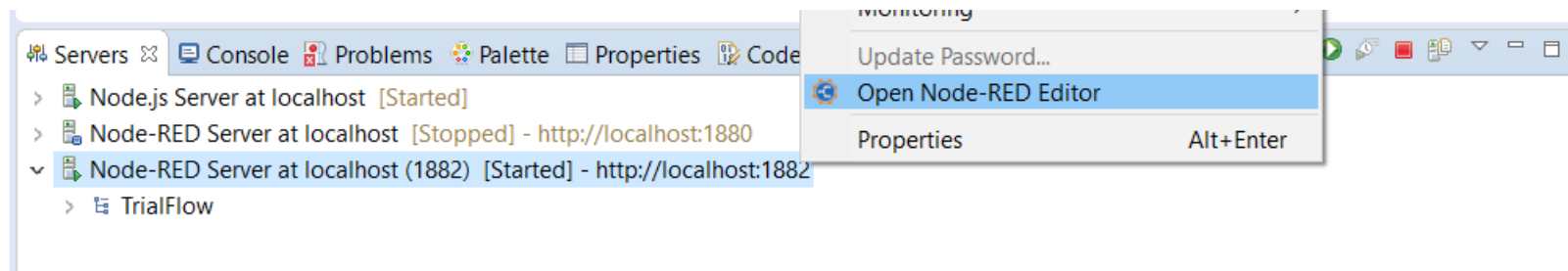
## ▶ Node-RED Flow Projects

- 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

▼ TrialFlow  
{ } TrialFlow (8045bd21.5095f).json

## ▶ Node-RED Server

- A dedicated Servers view lets you create, start and stop Node-RED servers within the Eclipse workbench
- Deploy a Node-RED flow onto a selected server and open its Node-RED editor



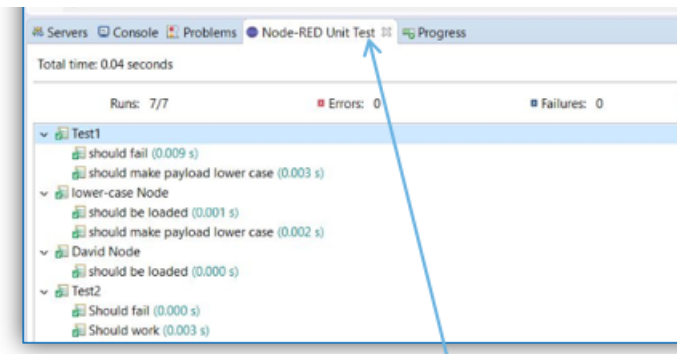
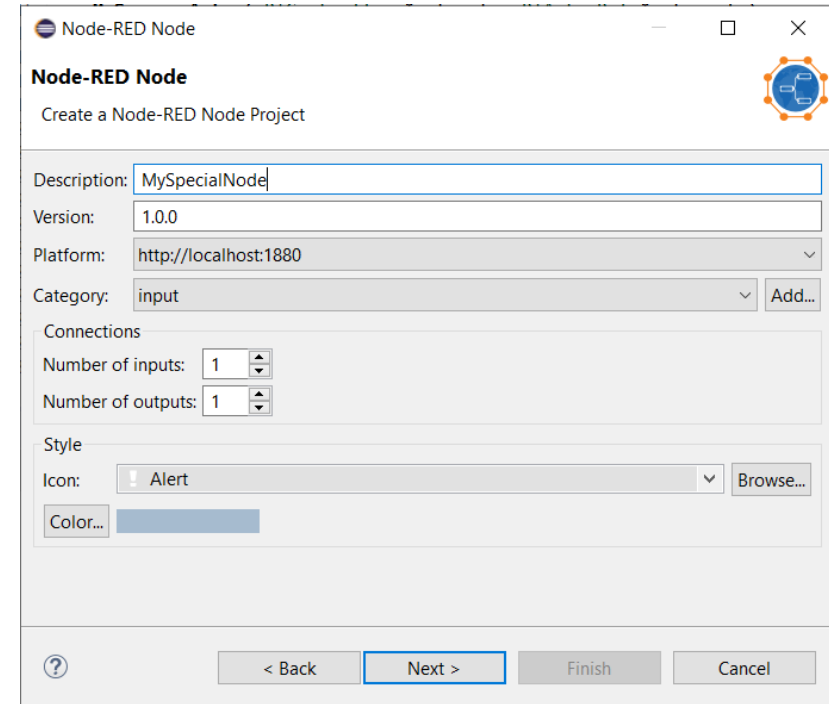
# Node-RED Tools in NodePlus

## ▶ Node-RED Node Projects

- More than 1900 nodes exist in the public library, but sometimes you may need to create your 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



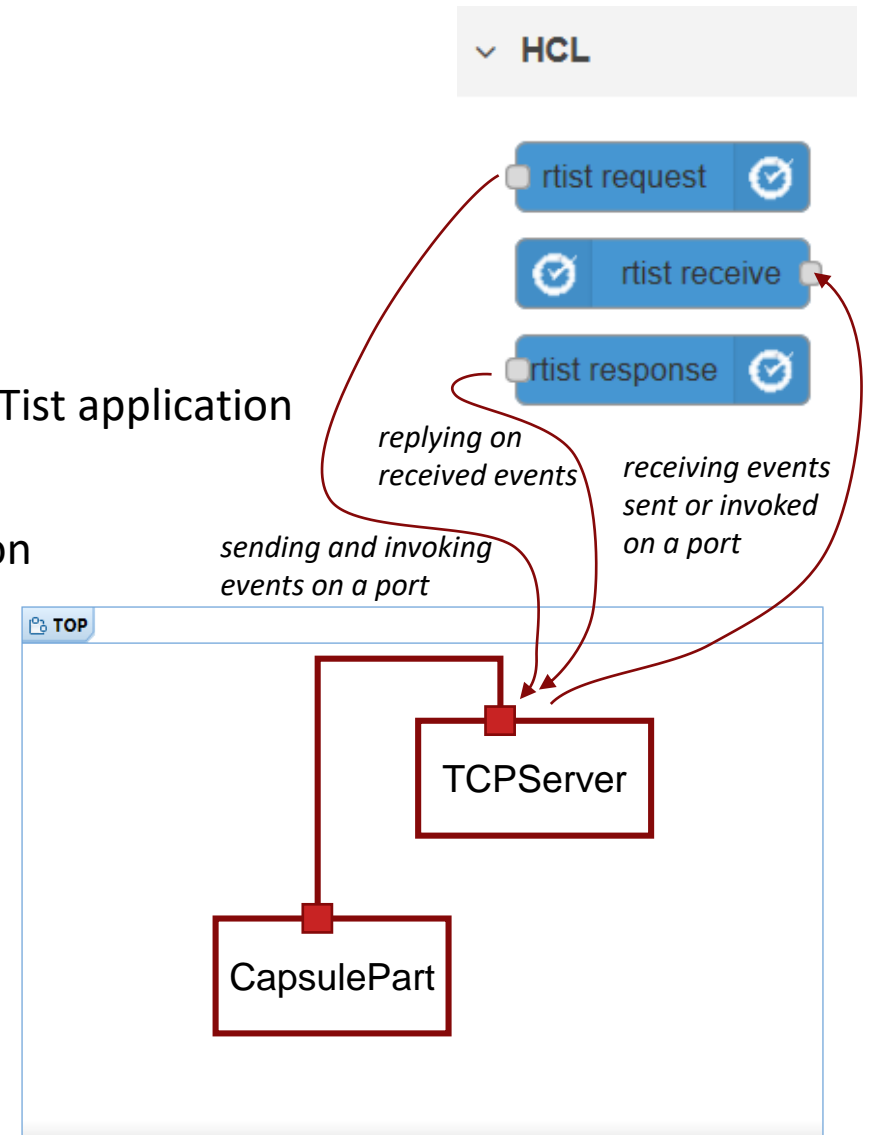
# Node-RED Tools in NodePlus

The screenshot displays the Node-RED Editor interface within the NodePlus IDE. The interface is divided into several panels:

- Project Explorer (Left):** Shows a tree view with folders for `MyFlowProject` and `MyNode`. A blue arrow labeled "Node project" points to this area.
- Node-RED Editor (Center):** The main workspace showing a flow for `MyFlowProject`. The flow consists of three nodes: `timestamp`, `MyNode`, and `Tweet`. A blue arrow labeled "Flow project" points to the editor title bar. Another blue arrow labeled "Node in use in a flow" points to the `MyNode` node in the flow.
- Node Palette (Left):** A list of available nodes, including `MyNode` (with a warning icon), `function`, `template`, `delay`, `trigger`, and `comment`. A blue arrow labeled "Node project" points to this palette.
- Info Panel (Right):** Displays information for the selected `MyNode`, including a description field (currently empty) and instructions: "Import a flow by dragging its JSON into the editor, or with `ctrl-i`".
- Servers Panel (Bottom):** Shows a `Node-RED Server at localhost [Started]` with sub-items for `MyFlowProject` and `MyNode`. A blue arrow labeled "Node project" points to this panel.

# Node-RED Tools in NodePlus

- ▶ 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



# Swagger Support in NodePlus

- ▶ Document APIs using Swagger in a new Swagger editor

Support for JSON and YAML syntax

Syntax highlight and content assist

The screenshot displays the Swagger editor interface. On the left, a code editor shows the Swagger JSON specification for the Petstore API, with syntax highlighting and content assist. On the right, a visual representation of the API is shown, including the Swagger Petstore logo, version (1.0.0), base URL (petstore.swagger.io/v2), and a list of API methods (store, user, pet) with their descriptions and external documentation links. The 'pet' method is expanded to show a 'POST /pet' endpoint with a 'Try it' button. Below the 'Try it' button, the request body is shown as a JSON object with fields like 'photoUrls', 'name', 'id', and 'category'. The response type is set to 'application/xml'. A table at the bottom shows error codes and descriptions, such as '405 Invalid input'.

Visual API representation

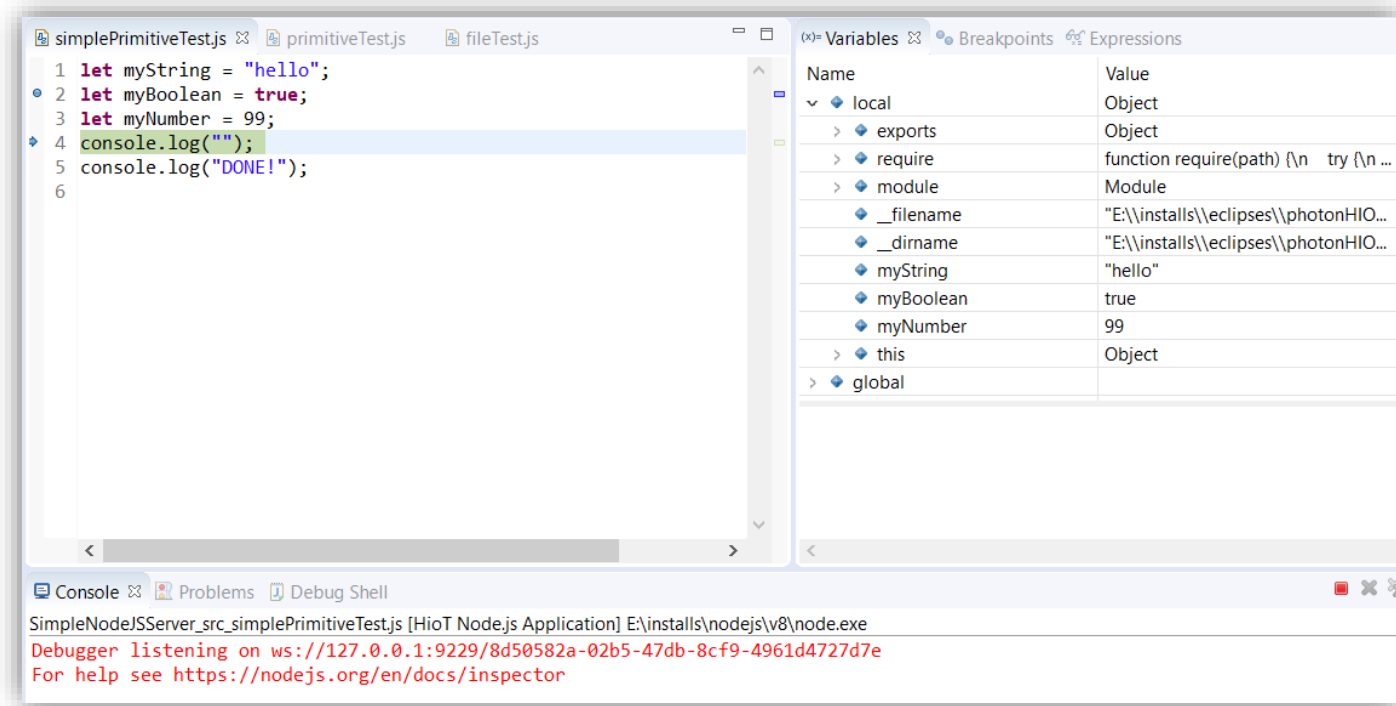
Support for executing / testing API

Automated generation of example data

Content types selections

# 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**

*Relationship*<sup>TM</sup>  
BEYOND THE CONTRACT

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

 WATCH THE FILM