HCLSoftware

What's New in DevOps Model RealTime 12.0.1

updated for the DevOps 2024.03 release

Overview

- Model RealTime 12.0.1 is based on Eclipse 2023-06 (4.28)
- Two brandings of the product are available (HCL and IBM). There is no differences in functionality between them.
  - Only difference is in the licensing mechanism and branding (e.g. documentation)

DevOps Model RealTime
Version: 12.0.1 (part of DevOps 2024.03)
Build: 12.0.1.v20240319_1019

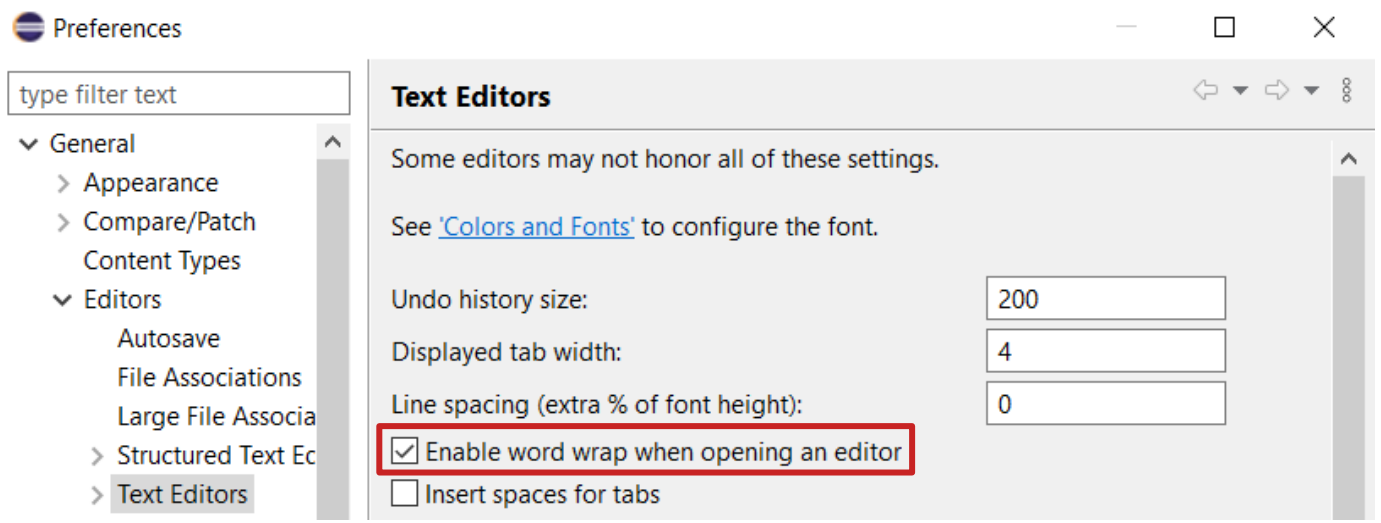
Visit https://model-realtime.hcldoc.com/help/topic/com.ibm.xtools.rsarte.webdoc/users-guide/overview.html

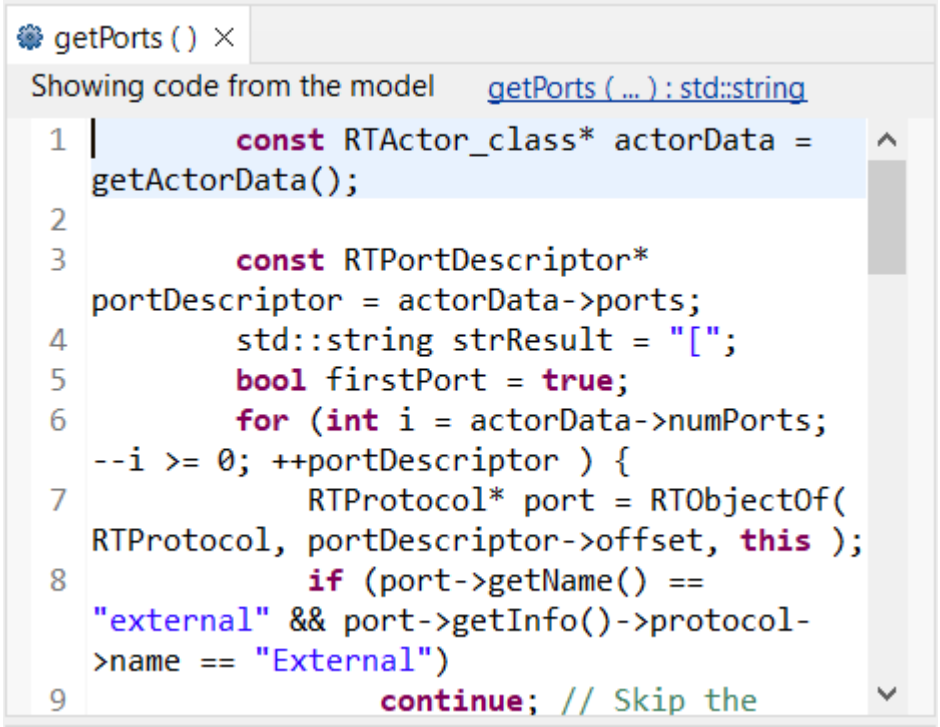
Eclipse 4.28 (2023.06)

- Compared to RSARTE/RTist 11.3, Model RealTime 12 includes new features and bug fixes from 4 quarterly Eclipse releases:
  - 2022.09 (https://www.eclipse.org/eclipse/news/4.25/platform.php)
  - 2022.12 (https://www.eclipse.org/eclipse/news/4.26/platform.php)
  - 2023.03 (https://www.eclipse.org/eclipse/news/4.27/platform.php)
  - 2023.06 (https://www.eclipse.org/eclipse/news/4.28/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 4.28 (2023.06)

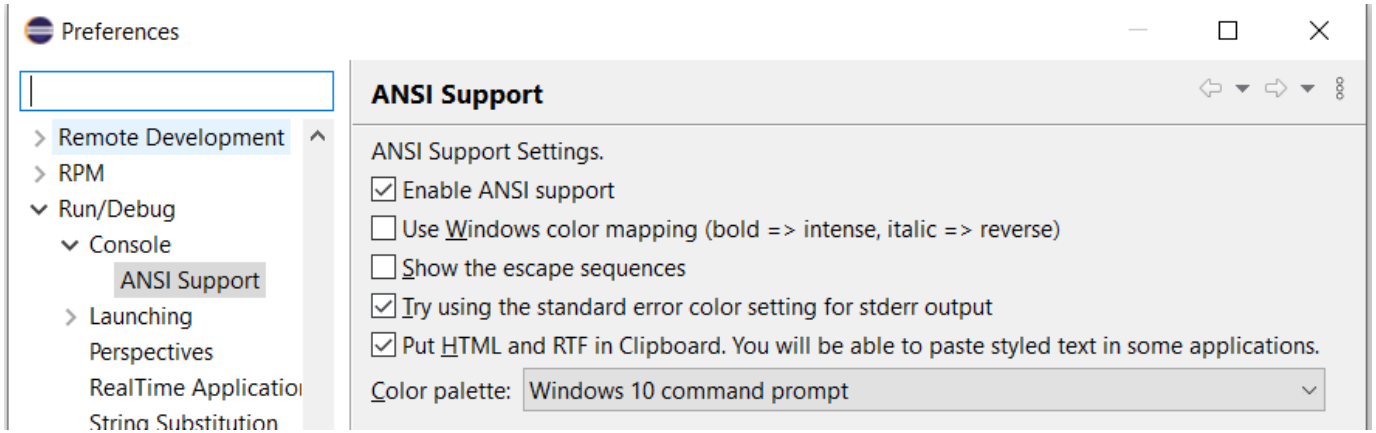
- New preference to enable word wrap automatically
  - **General - Editors - Text Editors - Enable word wrap when opening an editor**
  - Applies to all text editors including the Code editor





Eclipse 4.28 (2023.06)

- The Eclipse console now supports ANSI escape codes for producing styled output
  - The SGR (Select Graphic Rendition) control sequences are supported
  - Rendering can be controlled by preferences in **Run/Debug - Console - ANSI Support**
  - You can use this to make application printouts easier to read



Code View × Problems Console

C++ <Transition> - Initial

Showing code from the file Initial

```
29 static const char * CSI = "\33[";
30 std::cout << CSI << "4m" << "Underlined" << CSI << "0m" << std::endl;
31 std::cout << CSI << "21m" << "Double underlined" << CSI << "0m" << std::endl;
32 std::cout << CSI << "51m" << "Framed" << CSI << "0m" << std::endl;
33 std::cout << CSI << "1m" << "Bold" << CSI << "0m" << std::endl;
34 std::cout << CSI << "3m" << "Italic" << CSI << "0m" << std::endl;
35 std::cout << CSI << "42m" << "With background color" << CSI << "0m" << std::endl;
36 std::cout << CSI << "36m" << "With foreground color" << CSI << "0m" << std::endl;
37 std::cout << "Normal" << std::endl;
```

Code View Problems Console ×

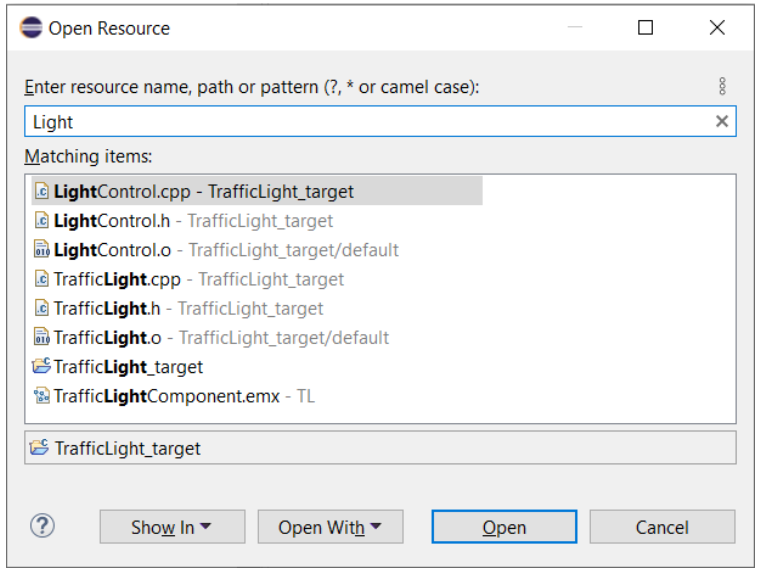
app.tcjs [C/C++ Application] [pid: 8]

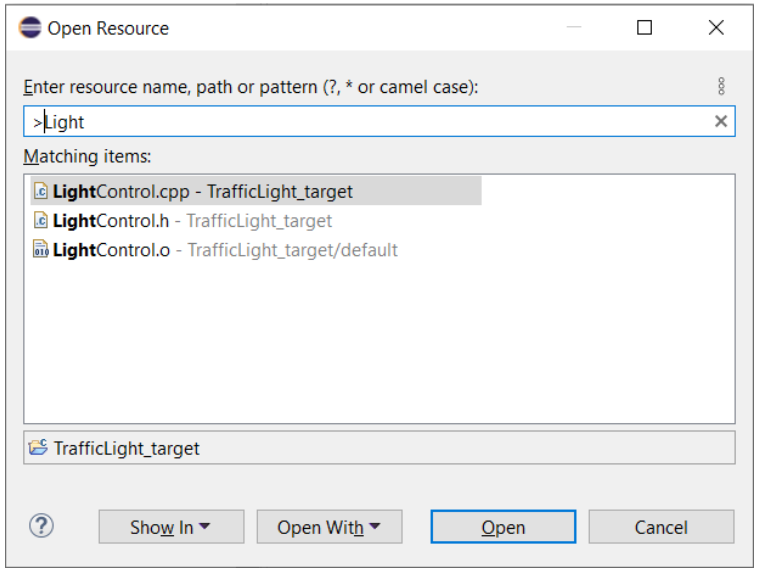
```
RT C++ Target Run Time System - Release 7.1.12

Underlined
Double underlined
Framed
Bold
Italic
With background color
With foreground color
Normal
```

Eclipse 4.28 (2023.06)

- Changes in the Open Resource dialog
  - Now the string you type in this dialog is searched in all parts of file names
  - It's no longer necessary to prefix the search string with an asterisk (*) to accomplish this
  - You can use a leading angle bracket (>) to enforce the old behavior (can be useful in case of too many matches)





CDT 11.2 (included as part of Eclipse 2023.06)

- Several improvements in the C++ parser have been implemented
  - Especially for supporting features of C++ 20 and C++ 23

- For more information about CDT improvements see
  https://github.com/eclipse-cdt/cdt/blob/main/NewAndNoteworthy/CDT-11.0.md
  https://github.com/eclipse-cdt/cdt/blob/main/NewAndNoteworthy/CDT-11.1.md
  https://github.com/eclipse-cdt/cdt/blob/main/NewAndNoteworthy/CDT-11.2.md

Newer EGit Version in the EGit Integration

- The EGit integration in Model RealTime has upgraded EGit from 6.2 to 6.6
  - This is the recommended and latest version for Eclipse 2023.06
- This upgrade provides several new features and bug fixes
  - For detailed information about the changes see
    https://wiki.eclipse.org/EGit/New_and_Noteworthy/6.3
    https://wiki.eclipse.org/EGit/New_and_Noteworthy/6.4
    https://wiki.eclipse.org/EGit/New_and_Noteworthy/6.5
    https://wiki.eclipse.org/EGit/New_and_Noteworthy/6.6
- Note: In 12.0 the EGit Integration was experimental due to potentially failing merge and rebase operations due to false conflicts. This problem is now solved in 12.0.1 and the EGit integration is no longer experimental.

New Location for Info Centers

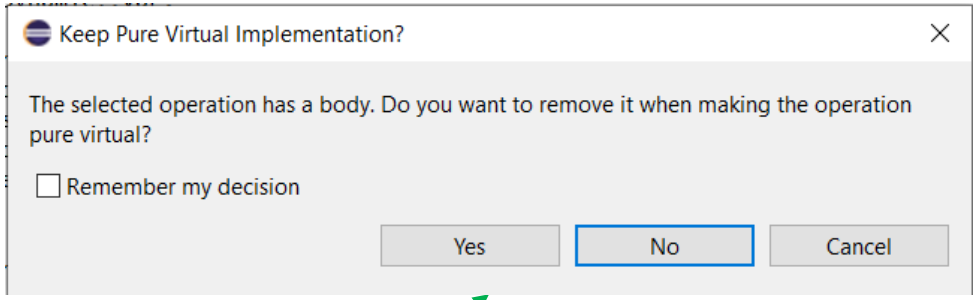
- As part of the product rename, the online documentation was moved to a new location
- There are two versions, depending on branding
  - HCL: https://model-realtime.hcldoc.com/help/index.jsp
  - IBM: https://www.ibm.com/docs/dmrt/12.0

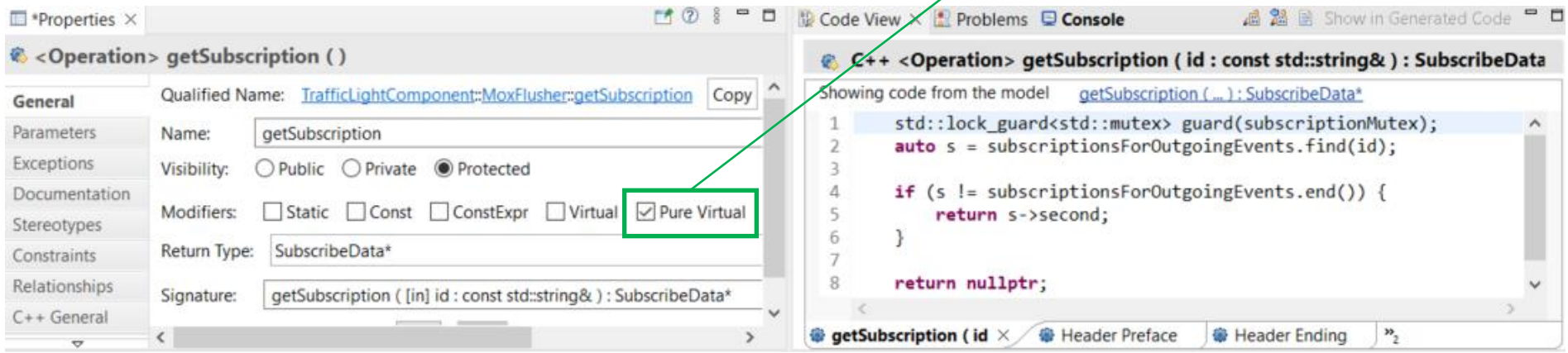
Removal of Body for Pure Virtual Operations

- If an operation with a body is made pure virtual you now have the option to either keep or remove its body
  - This is controlled by a new preference **RealTime Development - Properties View - Remove body when making an operation pure virtual**

- This also works for multiple selected operations which makes it easy to convert a class with a "trial implementation" to become an abstract class

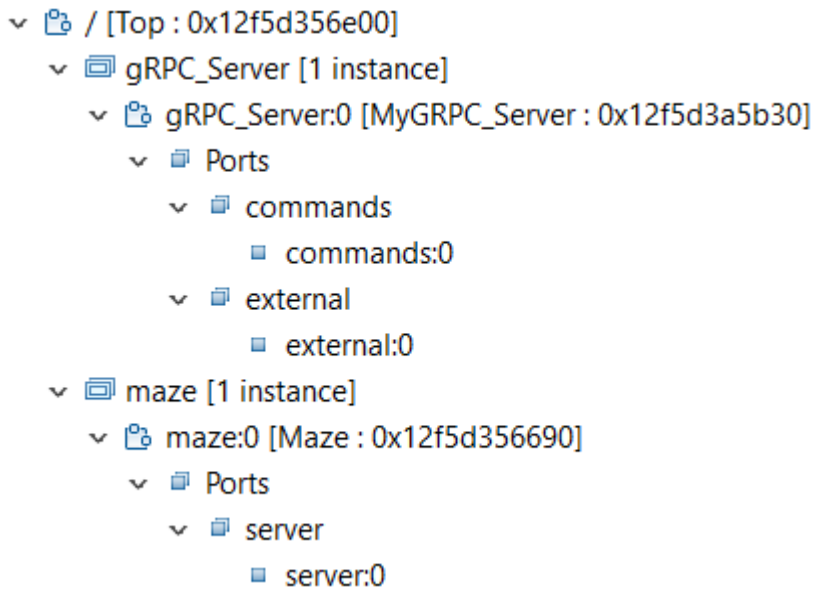
- Note: This feature is currently experimental, due to a known issue: If the removed body is open in a code editor, it will be recreated when that editor is closed



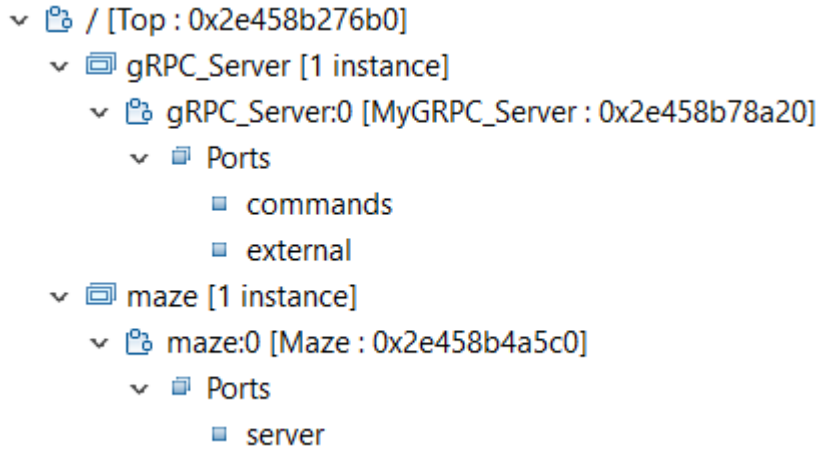


Better Display of Non-Replicated Ports while Debugging

- Ports with single multiplicity are now shown with a single node in the Debug view
  - Leads to a cleaner and less cluttered Debug view, that is easier to work with



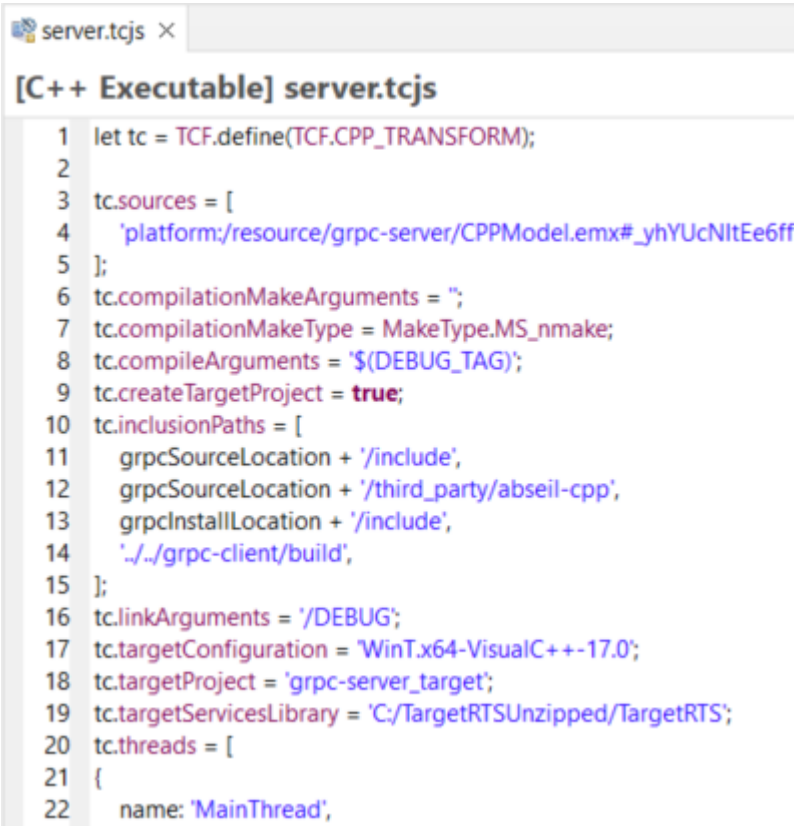
before



now

Sorting of TC Properties

- TC properties are now sorted alphabetically when saving a TC file
  - The "sources", "parents" and "prerequisites" properties will come first, followed by other properties sorted by name
  - Makes it easier to find a TC property in the Code tab of the TC editor
  - Reduces the risk for merge conflicts when multiple people edit the TC file
  - Comments that precede a TC property are moved with that property when sorting the properties



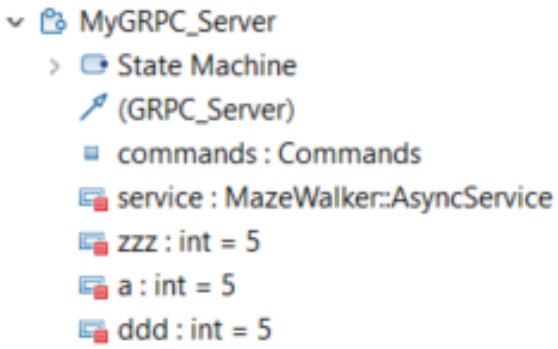
- Note 1: Sorting does not happen for TC files that contain JavaScript beyond simple assignments of properties. This is because sorting in such a file could affect the meaning of the TC file.

- Note 2: If you have TC properties with values that reference other TC properties, sorting will happen and may affect the meaning of the TC file. Consider turning sorting off in this case.

- Sorting can be turned off by means of a new preference **RealTime Development - Transformation Configuration Editor - Auto sort properties on save**

Constructor Initializer Ordering

- The initializer list for a constructor that is automatically generated is now ordered according to the order of declaration of the initialized members
  - The base class is always initialized first (for a capsule the base class is RTActor, or another capsule class)




```
21 MyGRPC_Server_Actor::MyGRPC_Server_Actor( RTController * rtg_rts, RTActorRef * rtg_ref )
22     : GRPC_Server_Actor( rtg_rts ,rtg_ref )
23     , zzz( 5 )
24     , a( 5 )
25     , ddd( 5 )
26 {
27 }
```

  - Constructor initializers are ordered like this both for capsules and classes

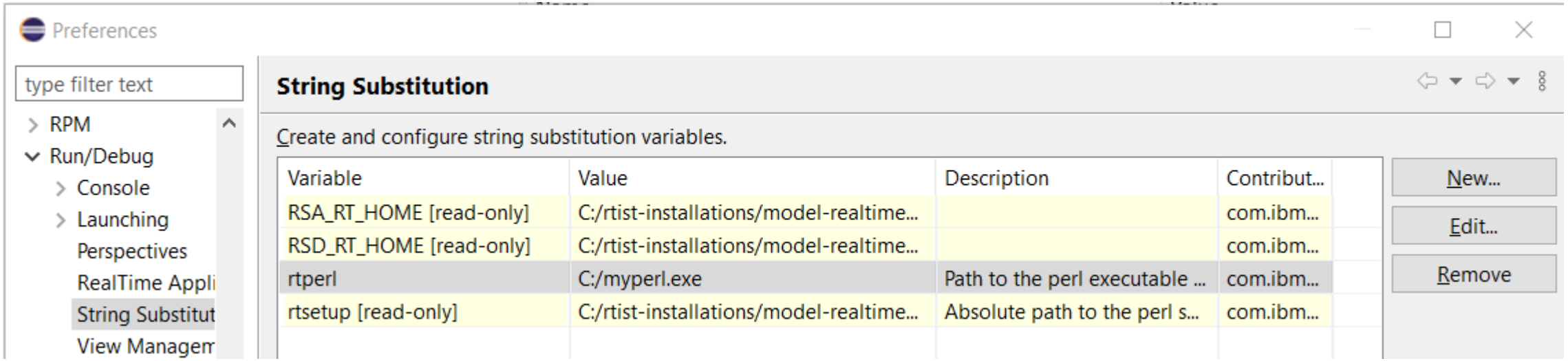
- This ordering avoids compilation warnings from certain compilers (and errors, if e.g. **`-Werror=reorder`** is specified for gcc)

Patch Files for Tracking TargetRTS Changes

- The TargetRTS now contains patch files for recent changes
  - Located in the installation at `rsa_rt/C++/TargetRTS_changelog`
  - Simplifies adoption of TargetRTS changes when uplifting to a new version of Model RealTime
  - A script `createPath.sh` is also provided that you can use for creating your own patch files, in case you prefer to instead apply your own TargetRTS changes on top of the standard version
- For more information and documentation about TargetRTS changes see this new website.

Building without "rtperl"

- Generated makefiles now allow building without using the "rtperl" utility
- You can override the "rtperl" variable using the preference **Run/Debug - String Substitution**



- If this variable is not set or has an empty value, the build will use "rtperl" from the installation (for Windows and Linux) or "perl" from the path (for MacOs)
  - Note: Workspaces created in older versions of Model RealTime may have this variable set to an incorrect path. You should then either remove or update the variable.

Improved JSON Support in the TargetRTS

- The JSON Encoder was improved to support encoding of more types of data. For example, it now supports encoding of pointers into strings.

```
{"address" : "0xe6ef6fed80"}
```

- A new JSON Decoder is now available. It can decode a JSON string produced by the JSON Encoder into a data object. See RTJsonDecoding.

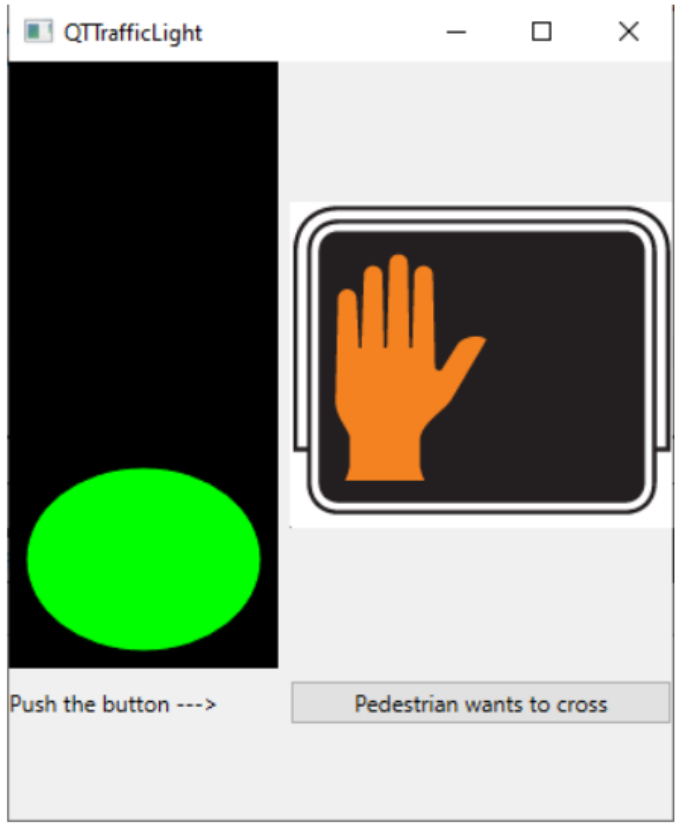
- A new JSON Parser is available. It's a general-purpose JSON parser that makes it easier to work with JSON in a realtime application. See RTJsonParser.

```
RTJsonParser parser;
RTJsonResult result;
bool ok = parser.parseJsonString(result, R"({"field1" : "string", "arr" : [1,true,3.14]})");

std::cout << result["field1"].get_string() << std::endl; // "string"
if (result["arr"].ok()) { // Check if the "arr" key is present
    std::cout << result["arr"].get_size() << std::endl; // 3
ASSERT(result["arr"][2] == 3.14);
ASSERT(result["arr"][1].get_type() == RTJsonResult::RTJSON_BOOL);
```

New Sample qt-trafficlight

- A new sample is available on GitHub: https://github.com/HCL-TECH-SOFTWARE/qt-traffic-light
- Shows how an application created with Model RealTime can be integrated with a user-interface implemented with Qt
  - Qt's concepts of **signals** and **slots** make it easy to update the UI from the realtime application in a thread-safe way
- For more information, see this post

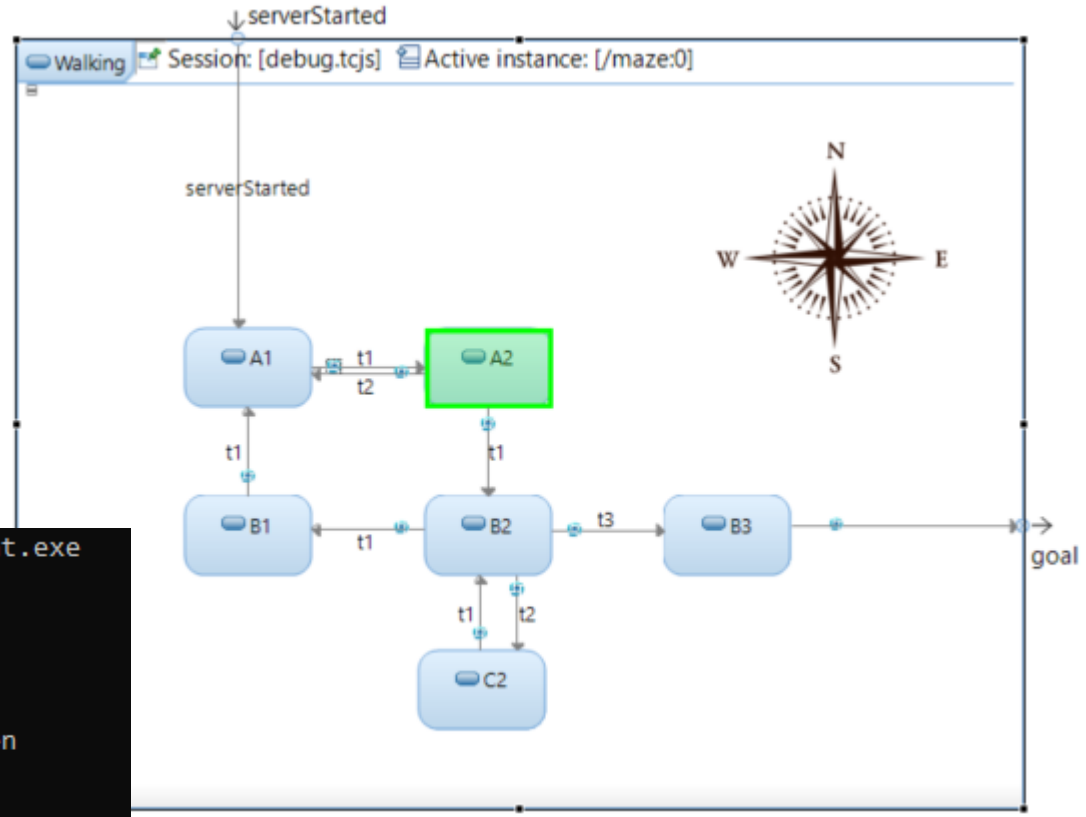


New Sample about Dependency Injection

- A new sample is available on GitHub: https://github.com/HCL-TECH-SOFTWARE/dependency-injection
- Shows how dependency injection can be combined with Build Variants for building different variants of an application with different implementations of certain capsules
- For more information, see this post

New Sample about gRPC

- A new sample is available on GitHub: https://github.com/HCL-TECH-SOFTWARE/lib-grpc-server
- Shows how a realtime application can act as a gRPC server to let other applications communicate with it by means of gRPC
  - It's similar to the TCP server library, but with some benefits. For example gRPC uses a binary protocol which is more efficient, and that its more "type safe" in C++ than sending JSON strings over TCP. Also, gRPC has more features and flexibility than TCP-based communication, for example message streaming.




```
C:\github\lib-grpc-server\grpc-client\build\Debug>maze_client.exe
Walk the maze. Commands:
east, west, north, south : Take a step in a direction
steps : Report number of steps taken
adjust : Adjust step count by specified number
subscribe : Subscribe to be notified when wrong way taken
unsubscribe : Unsubscribe to be notified when wrong way taken
exit : Exit
>east
>
```

HCLSoftware

hcltechsw.com