

Release Notes RayQC 2.0



21.04.2015

Release Notes RayQC 2.0



**Copyright © Raynet GmbH (Germany, Paderborn HRB 3524). All rights reserved.
Complete or partial reproduction, adaptation, or translation without prior written permission is prohibited.**

Release Notes RayQC 2.0

Raynet and RayFlow are trademarks or registered trademarks of Raynet GmbH protected by patents in European Union, USA and Australia, other patents pending. Other company names and product names are trademarks of their respective owners and are used to their credit.

The content of this document is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Raynet GmbH. Raynet GmbH assumes no responsibility or liability for any errors or inaccuracies that may appear in this document. All names and data used in examples are fictitious unless otherwise noted.

Any type of software or data file can be packaged for software management using packaging tools from Raynet or those publicly purchasable in the market. The resulting package is referred to as a Raynet package. Copyright for any third party software and/or data described in a Raynet package remains the property of the relevant software vendor and/or developer. Raynet GmbH does not accept any liability arising from the distribution and/or use of third party software and/or data described in Raynet packages. Please refer to your Raynet license agreement for complete warranty and liability information.

Raynet GmbH Germany
See our website for locations.

www.raynet.de

Table of Contents

Introduction	4
New Features	5
Resolved Issues	17
Migration	18
System Requirements	22
Additional Information	23
Need Help?	24

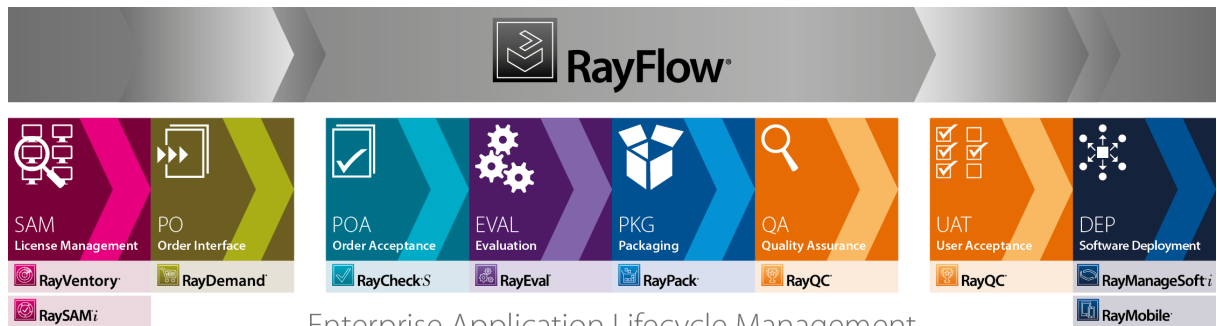
Introduction

RayQC is a rule-based tool used to create and execute test criteria using one or more checklists. It offers various modules to check the quality of applications and software packages all along the Application Life-cycle Management process.

Executing quality checks is possible using manual and / or automated procedures, which may utilize an extendable plug-in interface to integrate individually scripted check routines, tailored exactly towards specific mission scenario needs.

This release of RayQC 2.0 presents a collection of new features and enhancements that were proposed by the members of the RayQC community, which consists of customers, internal users, and of course the product development and management team. Thanks a lot for your ideas and involvement, as it is the perfect fuel for creativity and innovation!

RayQC is available as a stand-alone product as well as having the ability to be integrated into the RaySuite solution powered by RayFlow. As part of the overall RaySuite product family development, Raynet introduces a new design schema for application interfaces. This revolutionary user interface concept elevates usability and responsiveness to a whole new level, allowing users to operate all products by applying universal procedures and well known methods.



Enterprise Application Lifecycle Management

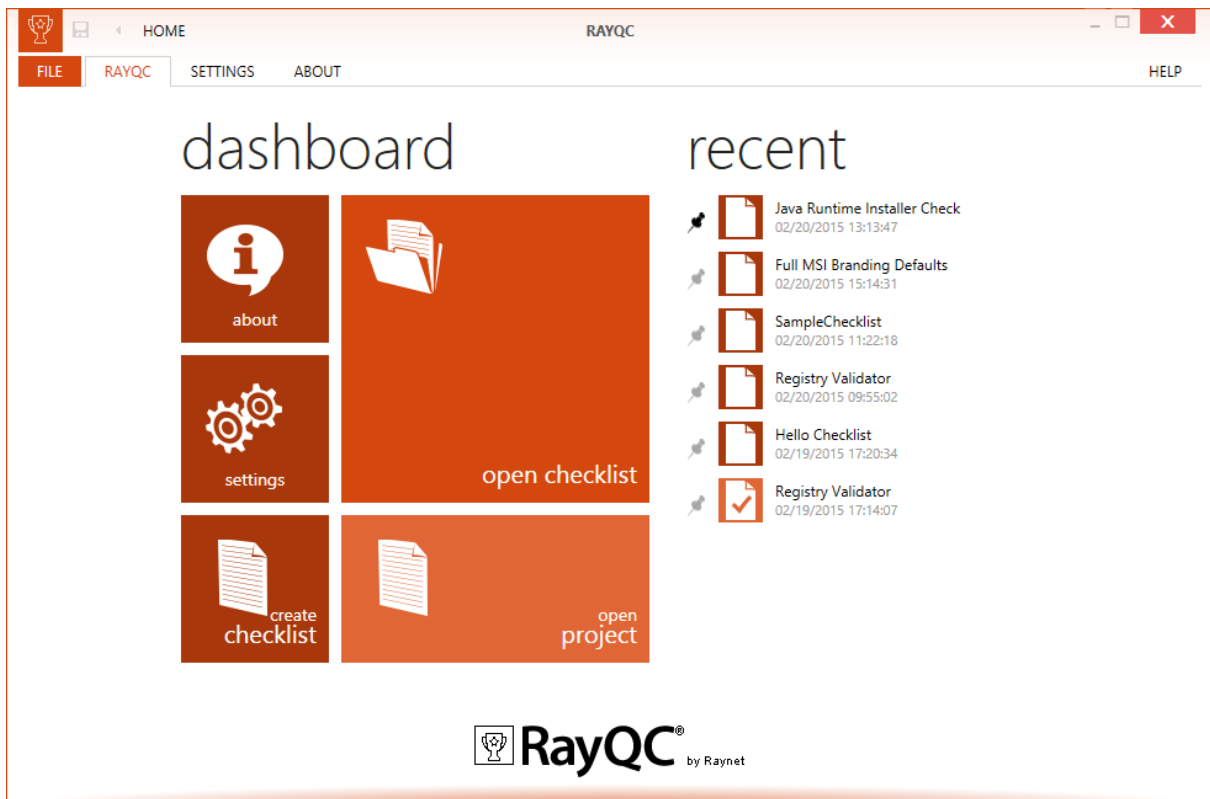
New Features

During the development phase of the current release, RayQC has been revolutionized at all levels of operation. The core application logic has been extended to provide improved support for checklist design and work-flow control. The new toolbox incorporates additional internal plug-ins as well as easy access to custom external plug-ins as well. Whilst existing external VBS plug-ins are still available via the command plug-in interface, a next generation of the external plug-in interface has been developed, utilizing PowerShell scripting and a simplified invocation system.

The following sections provide detailed information about the new features; however, technical background information and directions for use are obtainable from the RayQC User Guide.

New Interface Design

The brand new application shell adheres to Raynet 3.0 design guidelines.



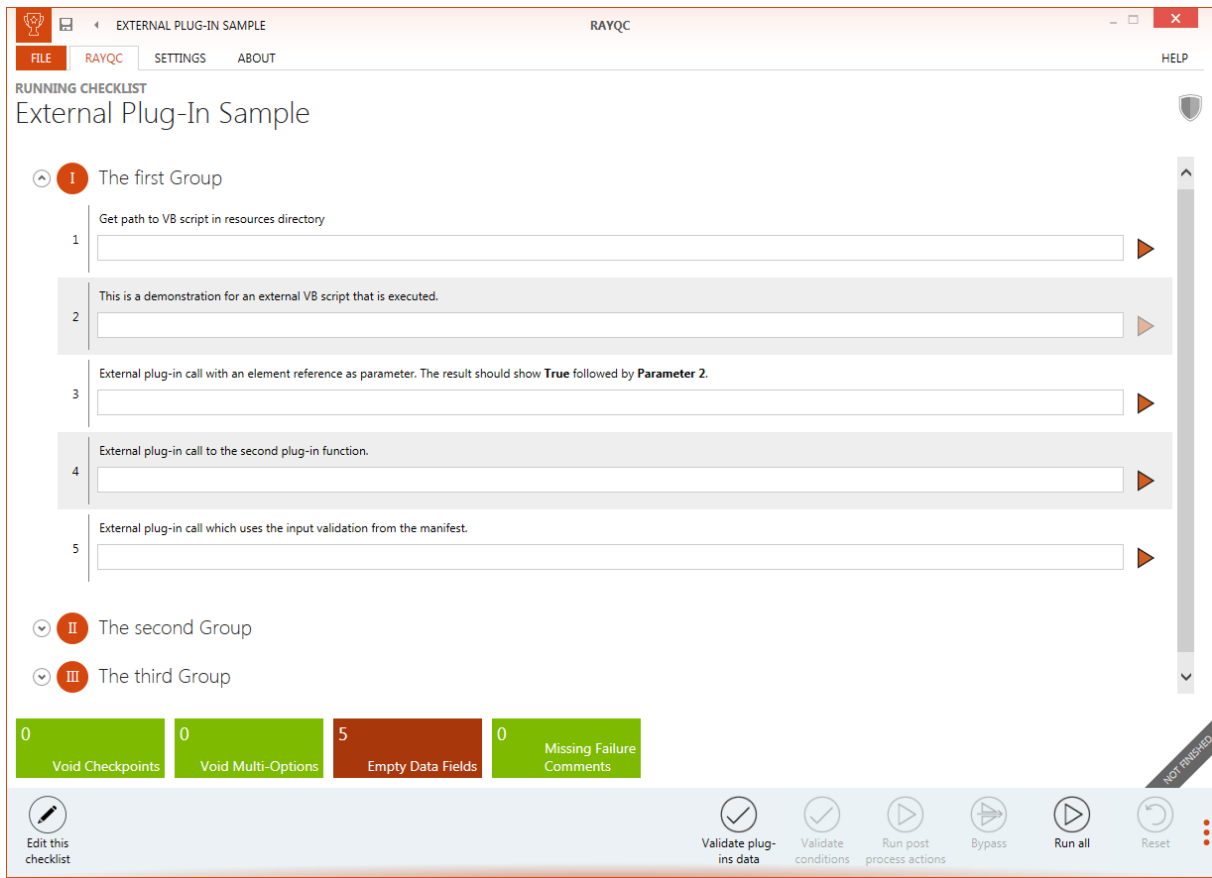
The intention of the UI redesign is to provide a clean and state-of-the-art canvas that does not only support everyday QA tasks, but actually makes it easier to solve issues with the help of clearly guided procedures and structured views. RayQC users do not have to search for features, they just use them.

Besides extended convenience by omnipresent drag & drop options, there are two further main areas that we have focused on: Navigating between essential activity points, such as checklist design or evaluation, and superior tasks of opening, saving and exporting files, has been reorganized by introducing the new Main Toolbar and the FILE menu. The new style does not only provide a uniform navigation concept for all RaySuite products, but at the same time makes it easier to reach all important functions with one or two clicks.

The new interface does not only come along with a shiny new surface. It contains improved logic under the hood which allows loading of views faster than ever before, and helps to work in distributed working environments by providing more flexibility regarding resource organization.

Checklist Viewer

The proven concept of the checklist viewer as an interface for intuitive and fast checklist evaluation has been extended by details that support swift orientation during checklist runs as well as seamless switching between designing checklist structures and previewing the achieved checklist layout:

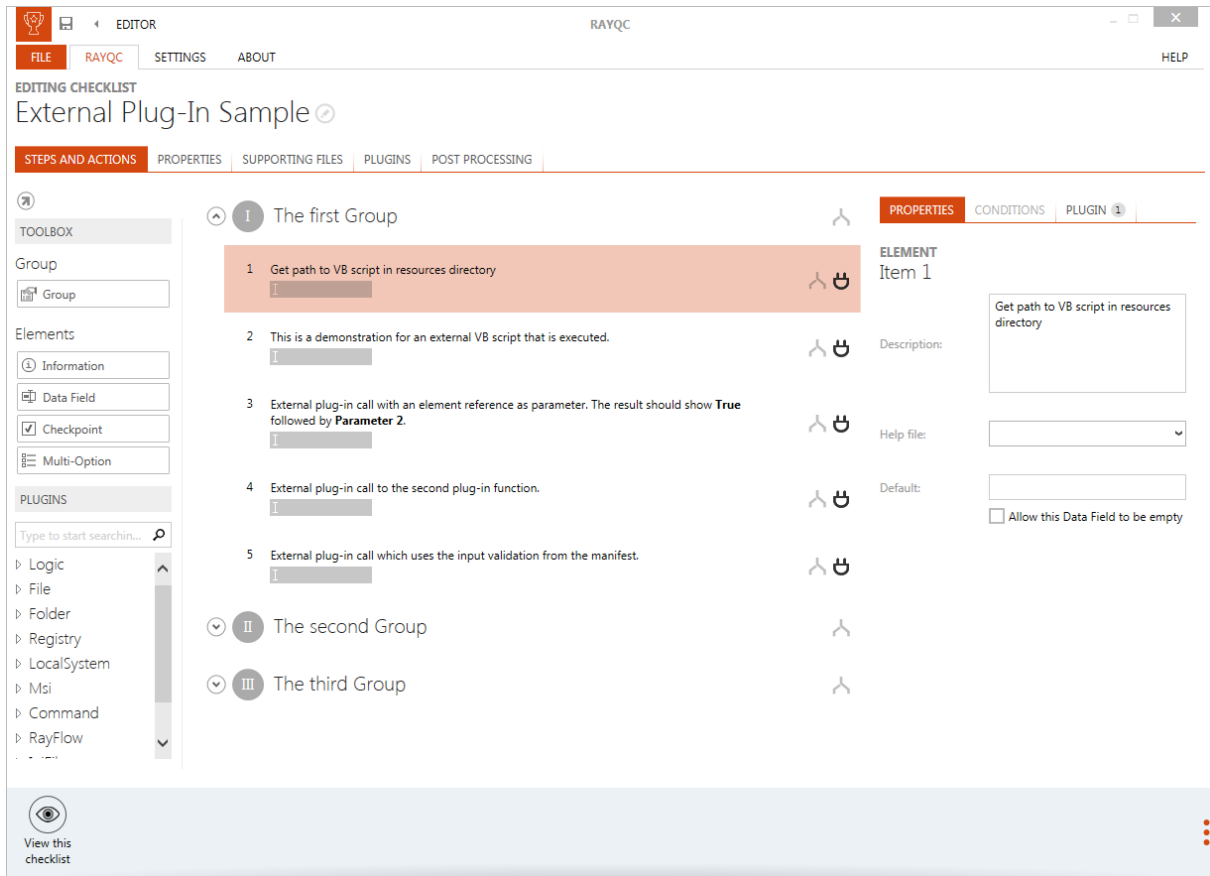


Each group is presented with a clearly visible index. This value is displayed as a roman number within an orange bubble. It is therefore easy to visually scan for group boundaries and recognize the elements that build the group content.

Whilst switching between the viewer and editor interfaces has been possible by the hotkey Shift + Tab, the new UI provides an additional switch button within the swipe bar. It is now aligned with all the other view-specific actions, such as "Run all plugins", "Bypass result", "Reset", and the like.

Checklist Editor

Whilst the Viewer has been modified mildly, the Editor has turned into a totally new working environment for checklist designers:



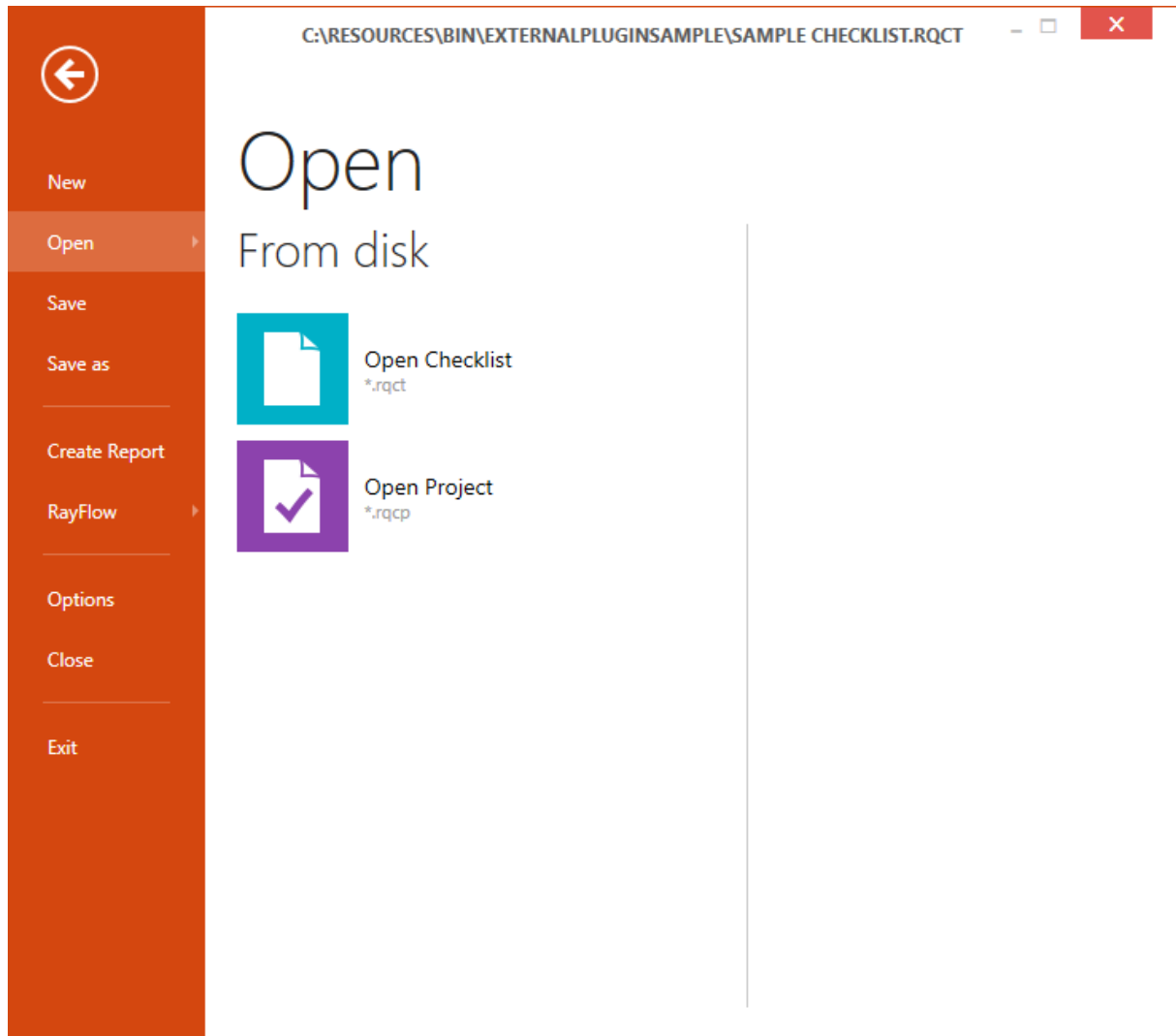
The Editor is separated into four tabbed views: **Steps & Actions**, **Properties**, **Supporting Files**, **Plugins** and **Post Processing**. The screenshot above shows the Steps & Actions view, containing the actual checklist structure of groups and elements. Users work left to right: From the toolbox on the left, they drag items (groups, elements, plugin functions) to the checklist canvas in the center. To define details per checklist element, they click the item from within the canvas, and work their way through the Properties, Conditions, and Plugin options available from the column on the right. For example, establishing dependencies by creating conditions is achieved by dragging the decisive element to the conditions tab of the depending element.

Within the Properties tab of a checklist, users define the title and description as well as report file name patterns and bypassing options. Due to the newly setup checklist container structure (see below), there is an additional management interface for uploading help files and images that will be used inside the checklist. Once they are uploaded here, using them in the scope of an element is just a matter of choosing them from a drop-down selector. The last tab of the Editor interface represents another revolutionary part of the checklist logic: Post processing options. These options allow checklist designers to assign automated activities once a checklist has been fully evaluated, such as uploading reports and / or data values to RayFlow.

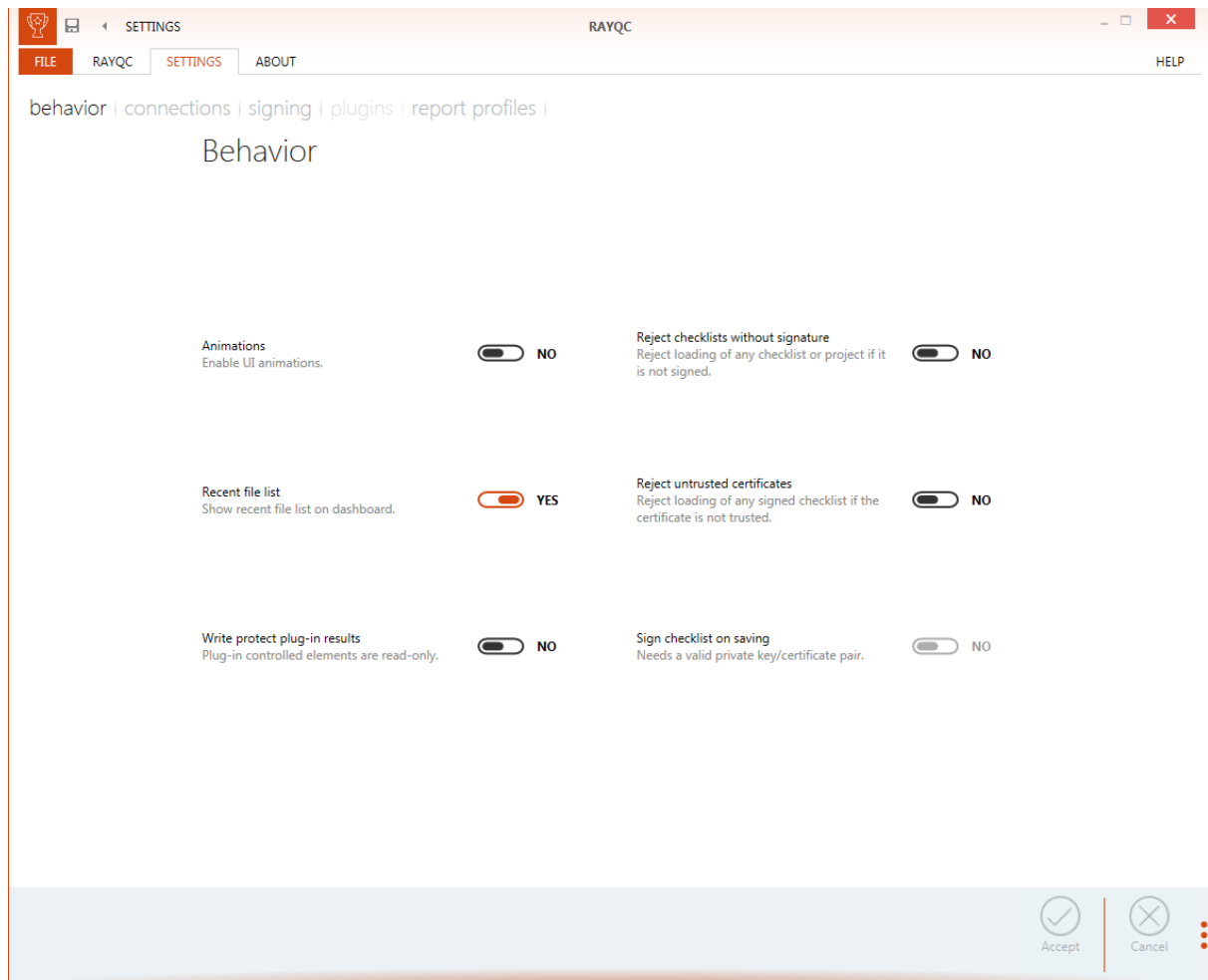
The whole Editor has been reworked to make sure checklist designers may use clickable selectors and drag & drop to build their logic. Fumbling with element ids is a pain of the past, as well as complex plugin parameter definitions and conditional statement setups.

FILE

All those who have been working with other RaySuite tools (e. g. RayPack) before, have already come to love the FILE menu. It offers direct access to essential application functionalities: open, new, save, close, save and export to RayFlow, manage options, etc. are all available here.



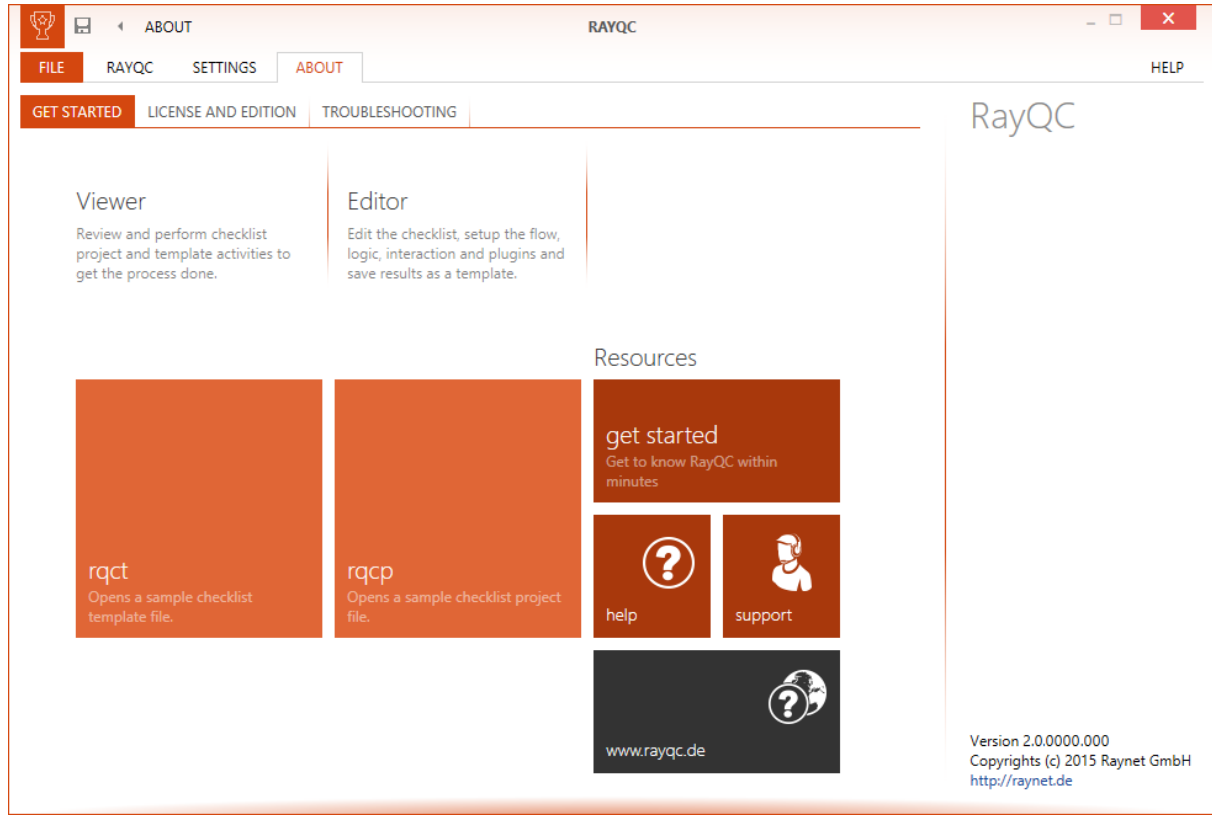
Settings



The Settings area has not only been extended by two new views (for configuring signing and report profile options), but at the same time simplified to make configuration decisions as easy as possible. Whilst some major settings are able to alter the functional behavior of interfaces, such as the option to protect plugin results from manual manipulation, others are designed to fine-tune bandwidth and performance optimization, such as the animations switch.

About

From the freshly added about view, users have direct access to license and troubleshooting information, as well as to helpful resources, such as demo checklist templates & projects, the Get Started Guide and our Raynet product support.



Unified Resource Management

As from 2.0, the RayQC file formats RQCT (RayQC Checklist Template) and RQCP (RayQC Checklist Project) have been transformed into ZIP containers, organizing all resources required for checklist edition and evaluation. Adding custom plugins directly into the file source makes checklists very flexible towards their parent RayQC instance - users may simply copy the template to another device that runs RayQC, and be sure to have all required material at hand without extra efforts.

Within the ZIP containers, there are predefined folders for plugins, help files, images, signature files, etc.. Once opened with a fitting compression tool, such as 7ZIP or WinZIP, users have full overview of the content. The checklist structure itself is stored as an XML file, as well as the status of project evaluations is within the project files. Organized in such a tidy manner, checklist resources are quite easy to handle, even without a RayQC instance at hand. It also provides the possibility to sign checklists with minimal effort.

Extended RayFlow Data Exchange Options

Even though prior product releases already provided decent data exchange options with RayFlow servers, there are some constitutional changes coming up with the current version: First of all, RayQC is no longer restricted to reading RayFlow information and returning reports once an evaluation is completed: The new RayFlow data service handler allows to update RayFlow properties of the connected package order.

Besides that, RayQC has been extended with an improved reset functionality, allowing users to restore a checklist to the initial state provided by incoming RayFlow parameters. This temporary storage of parameter values allows to start evaluation runs all over directly from within the QC application, without having to re-initiate the evaluation from RayFlow. By utilizing the internal RayFlow plugin, checklist designers are free to use any RayFlow data field they require, within any data field element of their choice.

As mentioned above, the settings area contains a view to manage as many RayFlow connection profiles as required. These connection profiles are stored withing the user specific settings location, allowing them to be easily exchanged by simply transferring the configuration files via copy & paste from one machine to another.

Full Report Flexibility Thanks to Profile Management

Evaluating a checklist is usually not only part of a technical QA process, but at the same time an essential activity of the overall product delivery workflow. Detailed and significant reporting regarding test results is absolutely required to enable transparent and comprehensible communication between the process stakeholders. Thus, as a reflection of this business impact, Raynet extends the report capabilities of RayQC for the 2.0 release: Users create and manage report profiles, which preserve settings for different export formats (e. g., PDF, HTML, DOCX). Whilst the actual report content is generated on the fly and according to live checklist data, RayQC utilizes the custom profiles to ensure the continuous delivery of documents in compliance with the corporate identity of the workflow owner. Users may prepare as many profiles as needed, and simply activate them individually per report export.

Reports may not only be created as part of manual tasks, but are available via automated command line and post processing methods as well. The tight system integration of RayQC as QA tool and RayFlow as overall workflow management application allows easy report and data transfer to enable seamless information exchange and phase transition.

Additional Internal Plugins

The classic set of internal RayQC plugins empowered checklist designers to retrieve information regarding the local file system, the registry, local INI files, and also to launch commands on the quality control machine. The extended set of predefined plugin functionality delivered as integral parts of the application has been enriched with a handy set of commonly required helping hands:

- **Logic plugin**

This set of functions is able to support easy checks for expected values, value domains, string lengths, string comparisons based upon regular expression evaluation, string-to-string or boolean-to-string conversions, and

the like.

- **MSI plugin**

Since RayQC is part of Raynet's Enterprise Application Lifecycle Management product suite, it is not overly surprising to find an integrated set of functions to analyze the content and structure of MSI files. This plugin allows the Windows Installer database to be queried as well as the Summary Information Stream of an MSI to be read. It is even possible to extract the files contained within the MSI, to validate it and to get access to its set of properties by simply calling predefined functions.

- **Local System plugin**

With this set of functions, current system status information can be obtained: Which user is logged in, how many processors does the machine have, what is the status of a specific service, which OS is present, and what are the values of environment variables?

- **File plugin and Folder plugin**

These two plugins are extended versions of the former File & Folder plugin. The base functions of opening files and folders, checking for existence, reading ACL information and MD5 hash values have been kept, whilst new options to read signature information, compare objects, unzip compressed containers, copy files, etc., have been added.

- **RayFlow plugin**

As already mentioned above, the RayFlow data exchange options have been massively extended. Being able to access all values RayFlow provides for a package order object via the internal plugin mechanisms is just one further step to full product integration. Just imagine to be able to upload an invalid INI file directly to RayFlow, where it can be used as base material for package adjustments after a failed Quality Control cycle. Or sending log files, registry information, MSI property values, etc. straight back to the packager, allowing him to perform the review with an optimum of evaluation feedback material.

We are sure to have extended RayQC with a powerful set of new plugins and features which speed up the overall quality assurance cycle:

- ✓ Designing complex automated checklists can be done much faster than before by the predefined options and simplified user interface handling.
- ✓ Evaluating checklist projects can be executed in no time due to fully automated plugin sequences.
- ✓ Downstream package repairs are optimized as well by the mass of valuable result information testers may provide via the RayFlow communication backbone.

However, if you encounter additional needs for plugin functionality, there are three ways to get them: Contact our support team with a product feature request, contact our sales department with product customization requests, or simply implement plugins on your own.

External PowerShell Plugin Interface

Whilst internal plugins have been predefined for general usage within RayQC checklists by the Raynet development team, external plugins may be created and integrated by RayQC users themselves. They usually provide specific logic for a test criterion required by a single checklist, or a whole checklist group.

RayQC 2.0 offers a new plugin interface based upon PowerShell, allowing checklist designers to combine scripted logic with the evaluation interface of our application. The definition of external (i. e. custom) plugins has been streamlined to enable several functions to be included within one plugin container. Parameter formats can

be defined via manifest settings, along with rules to determine if parameter values are valid according to regular expressions.

Each External Plugin must consist of at least one PowerShell script file, containing the plugin logic, and an XML manifest file (`Manifest.xml`), declaring the plugin interface for direct interaction and communication with RayQC. The type of supported PowerShell scripts is not determined by RayQC itself, but the PowerShell interpreter on the machine the plugins are executed on.

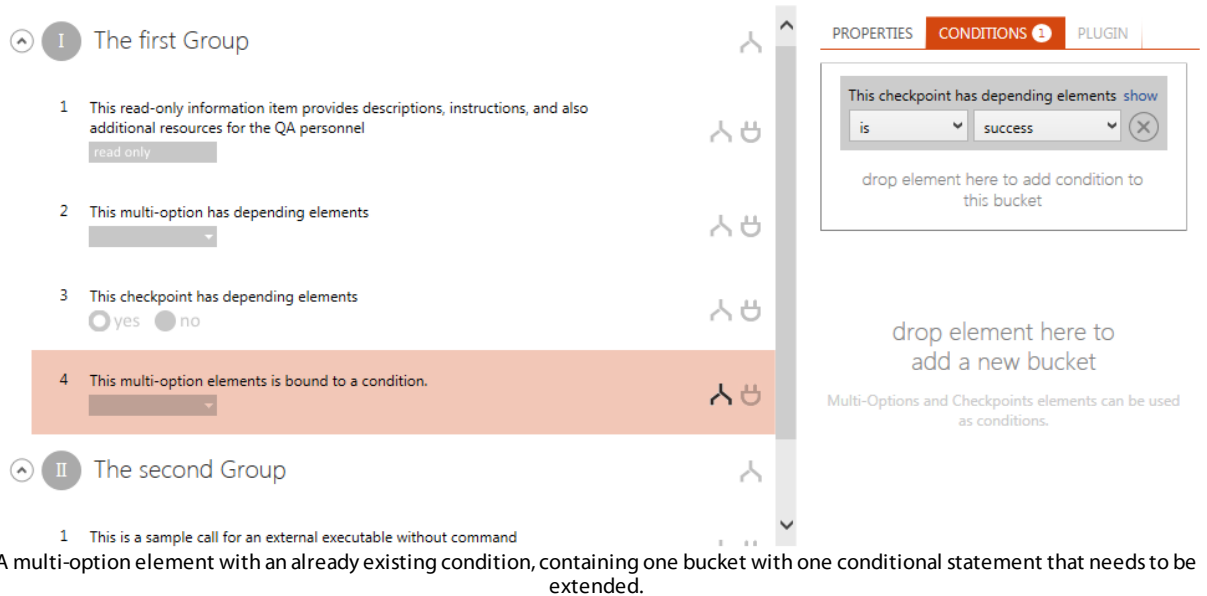
Since PowerShell is regarded as the contemporary standard for Windows scripting, we based our newly designed interface on it and deprecated the former direct VBS support. Even though PowerShell is required for new plugins, there is still basic support for already existing external VBS plugins: The extended internal command plugin allows to launch VBS scripts, which provides downwards compatibility by minimal modification effort.

Simplified Condition Handling

An important aspect of dynamic checklist evaluations are conditional statements. In RayQC, users are able to define a combination of several conditional statements, which may decide about the actual evaluation path of a checklist. For example, if the result of Checkpoint A is TRUE and the result of Multi-Option B is FALSE, checklist group X has to be evaluated. If not, checklist group X is invisible and does not affect the result of the checklist. Conditions may be applied to single elements as well as to whole group objects.

The logical idea of conditions in RayQC 2.0 is based upon the so called "disjunctive normal form" (DNF), which allows the building of any kind of condition as a combination of ORs between ANDs. To be more precise: A DNF is a disjunction of conjunctive clauses. Within our checklist editor, clauses are called buckets. Within each bucket, there may be several conditional statements, but all have to evaluate to TRUE in order to let the bucket evaluation result become TRUE. If the condition for an element contains more than one of those buckets, it is sufficient to have one bucket evaluate as TRUE to let the whole condition evaluate as TRUE. So you see, the buckets contain a number of ANDs, and are chained by ORs.

Handling conditions via the RayQC Checklist Editor user interface has been adjusted to follow the RaySuite 3.0 interface guidelines: Adding a condition to an element is achieved by a combination of drag & drop operations, followed by selections from predefined drop-down controls:



The screenshot shows a RayQC interface with two groups. Group I, 'The first Group', contains four elements. Element 4 is highlighted in orange and has a condition. The right sidebar shows the 'CONDITIONS' tab with a dropdown menu set to 'is' and 'success'. Below the dropdown is a 'show' button and a 'drop element here to add condition to this bucket' instruction. Below that is another instruction: 'drop element here to add a new bucket' and a note: 'Multi-Options and Checkpoints elements can be used as conditions.'

1 This read-only information item provides descriptions, instructions, and also additional resources for the QA personnel
read only

2 This multi-option has depending elements

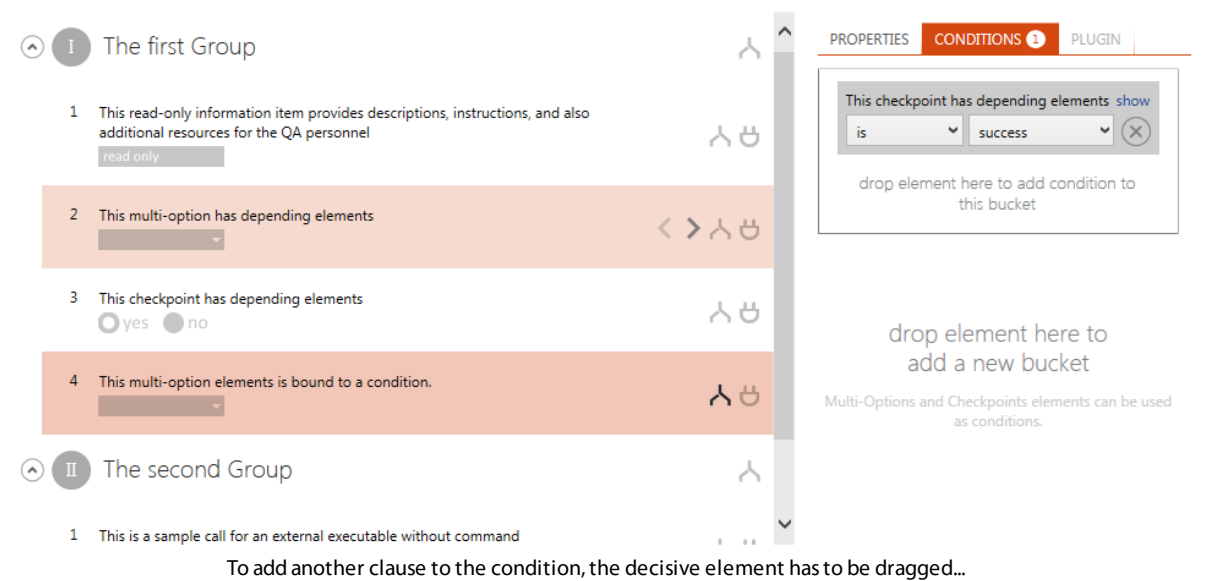
3 This checkpoint has depending elements
 yes no

4 This multi-option elements is bound to a condition.

II The second Group

1 This is a sample call for an external executable without command

A multi-option element with an already existing condition, containing one bucket with one conditional statement that needs to be extended.



The screenshot shows the same RayQC interface as above, but with a new element added to Group I. Element 2 is now highlighted in orange and has a condition. The right sidebar is the same as above. Below the 'drop element here to add condition to this bucket' instruction is a new instruction: 'drop element here to add a new bucket' and a note: 'Multi-Options and Checkpoints elements can be used as conditions.'

I The first Group

1 This read-only information item provides descriptions, instructions, and also additional resources for the QA personnel
read only

2 This multi-option has depending elements

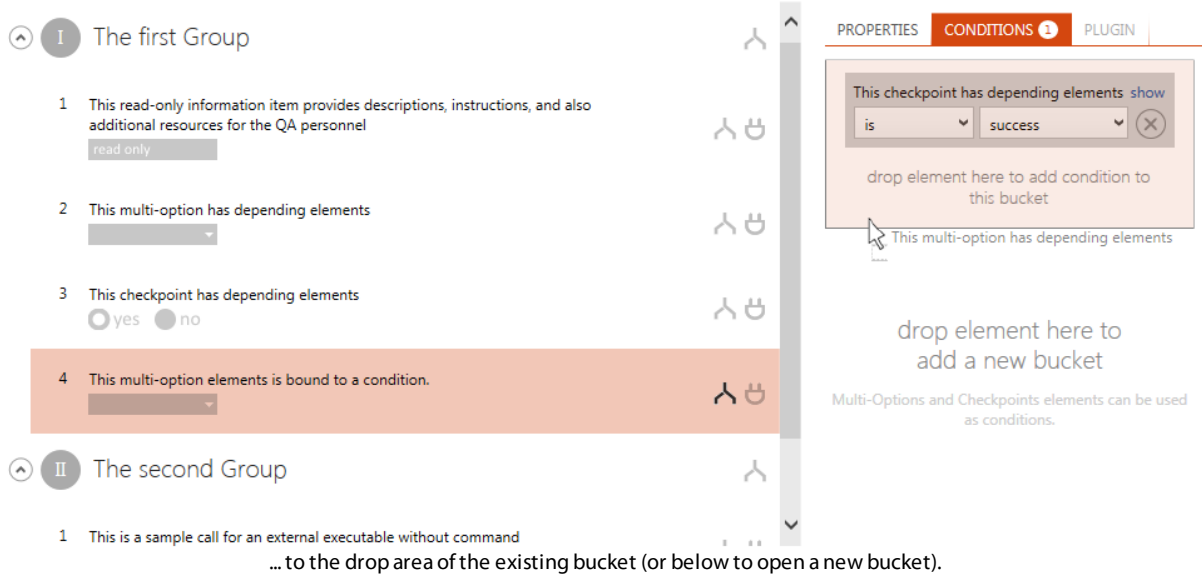
3 This checkpoint has depending elements
 yes no

4 This multi-option elements is bound to a condition.

II The second Group

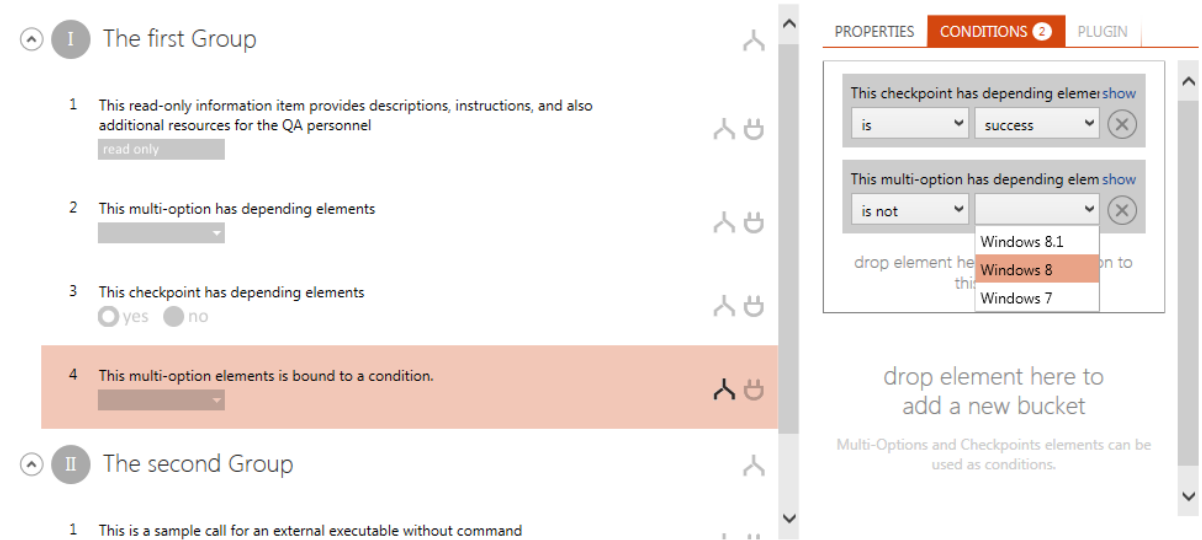
1 This is a sample call for an external executable without command

To add another clause to the condition, the decisive element has to be dragged..



The screenshot shows a checklist interface with two groups. The first group, 'The first Group', contains four items. Item 4, 'This multi-option elements is bound to a condition.', is highlighted in orange. To the right, a 'CONDITIONS' panel is open, showing a dropdown menu with 'is' selected and 'success' as the result. Below the dropdown, there is a text prompt: 'drop element here to add condition to this bucket'. A mouse cursor is hovering over the text 'This multi-option has depending elements' below the panel.

... to the drop area of the existing bucket (or below to open a new bucket).



This screenshot shows the same checklist interface. The 'CONDITIONS' panel now shows 'is not' selected in the dropdown menu. Below the dropdown, a list of operating systems is displayed: Windows 8.1, Windows 8, and Windows 7. The 'Windows 8' option is highlighted in orange. The text prompt below the list reads: 'drop element here to add a new bucket'.

At the end the user just needs to select the expected result for a true conditional clause evaluation from the automatically added bucket item.

Checklist Signing

Checklists and their evaluation results that are part of the productive Enterprise Application Lifecycle Management (EALM) process should be handled with the very same care for security that the software packages and confidential enterprise information have. Adding support for checklist signing is one step in this direction. Defining signature details as key pair profiles containing certificate, key and password data is accomplished via the newly added signing tab within the settings view. The switch for activating the sign options is embedded into the behavior settings. Once configured and activated, signing is executed whenever a checklist is modified. All contents of a checklist file container are signed to prevent manual and unauthorized manipulations from being recognized. Additional behavior settings prevent unsigned checklists from getting loaded within the application interface, or even more strict, prevent checklists that are signed with untrusted certificates from getting loaded.

Once again, setting up signing is a matter of a few clicks within the new RayQC UI, but it elevates the quality of the whole QC process to another level. The principle of providing a maximum of usability for an optimum of functionality is the central theme of release 2.0. Users are highly welcome to experience it on their own and tell us about their personal impression.

Resolved Issues

The following issues have been fixed during the implementation of RayQC .

- **RayFlow connection organization** [RQC-68;121]

As already outlined before, RayQC is not only available as a standalone product, but can be integrated into a RayFlow process. In the past connections were set up in a flat dialog within the settings, allowing users to enter credentials to one specific RayFlow server. We received feedback, that it would be more practicable to be able to have multiple profiles allowing integration into more than one RayFlow instance. This functionality has been implemented and this 2.0 release contains a connection profile repository allowing to integrate into multiple RayFlow instances . Additionally, there is new functionality that allows to test the entered credentials, to see if connectivity is present.

The well-known command line injection for RayFlow credentials is still present, allowing RayFlow to launch RayQC with a dedicated connection credential set. The injected credentials override those of the currently active profile configured for the RayQC instance as long as the injected working session is active. The combination of traditional and extended RayFlow connection management features covers an end-to-end integration from both product entry points.



Note:

For RayQC 1.5, there have been several Service Pack's and a Patch release up to now. All issue fixes covered within these have been integrated into the 2.0 development process, and are therefore not present in the current release anymore.

The company strategy of ongoing Service Pack provision will be continued, in order to make sure that our customers get improvements into their productive environments as soon as possible.

Migration

Raynet introduces some major changes with the release of RayQC 2.0 . Besides the new user interface design, there are some technical innovations that affect core functionality. These have to be considered especially by those users who have been running older RayQC versions and wish to migrate to the latest one.

Upgrading the RayQC Application

The latest improvements do not only affect RayQC functionality, but also the application installation procedure itself. The right upgrade strategy actually depends on the state and origin of the current RayQC installation.

General upgrade preparations

RayQC 2.0 is delivered as an MSI software package. In order to install it safely:

1. Download the MSI package for RayQC 2.0 from the Raynet resource repositories.
(If you have not already received credentials, please contact the Raynet support team via support@raynet.de to get them via email)
2. Copy all files that need to be kept for later re-use (such as resources of global external plug-ins, log, settings and configuration files, the `*.license` file, etc.) to a temporary transfer directory outside the RayQC application directory (where they usually reside).

The existing RayQC installation has been created from a binary file copy

- Remove the old RayQC installation manually.
- Execute the RayQC 2.0 MSI package and work yourself through the setup routine.

The existing RayQC instance has been installed by an MSI setup execution

- Execute the RayQC 2.0 MSI package and work yourself through the upgrade routine.

Adjusting the newly installed RayQC instance

3. Launch RayQC (if required, execute the license activation routine to be able to do so, or copy the old `*.license` file to the new program root directory)
4. Edit the configuration files and settings according to the old system state: either by adjusting the settings manually in RayQC, or by using an external editor as required for the logging configuration resources.

5. Test the new settings and configurations by creating and evaluating checklists, communicating with RayFlow, reviewing log files, etc.
6. If there are issues regarding broken or missing functionality, please feel free to contact the Raynet support team via support@raynet.de.

Upgrading RayQC Files

The file formats RQCT and RQCP Raynet introduced in RayQC 1.5 and have been massively reworked to match the needs of the modernized application logic. Therefore, it is not possible to simply re-use templates and projects that have been generated with RayQC 1.5 in the current version 2.0.

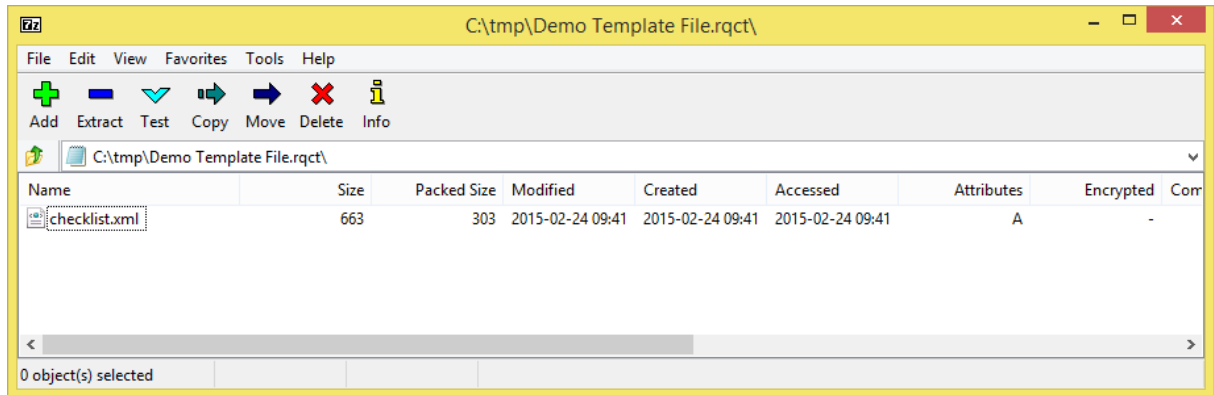
The RQCT files used in RayQC 2.0 are no longer XML structures, but ZIP containers that contain the XML checklist file (`checklist.xml`) as well as all other resources required to run the checklist on RayQC: plug-ins, help files, images, etc. are stored within dedicated directories wrapped in the ZIP container.

Additional files that represent the current project status of a checklist evaluation (`state.xml`), post-processing settings and signature information, are added when a template is saved as project file RQCP.

Knowing about these changes makes it quite obvious that there must be some manual steps in any kind of checklist transition from version 1.5 to 2.0. Once this is done, the following standard procedure is a valid option for their transition to 2.0:

To transfer a RayQC 1.5 RQCT to the current 2.0 format, users have to run the following procedure:

1. Copy the original RQCT file to a temporary working directory.
2. Change the file
 - a. name to `checklist`
 - b. extension from `.rqct` to `.xml`
3. Create a new ZIP that contains the `checklist.xml` file. Name the ZIP container according to the old checklist file name, and set the file extension to RQCT.
4. The result of steps 1-4 has to be a zip container with the file extension `*.rqct`, that contains a `checklist.xml` file with the original checklist structure.



5. Open this file in RayQC 2.0.

6. It is most likely, that the validation procedure run during checklist loading states issues with the XML source structure. In this case, a dialog is displayed, revealing details about invalid areas with a click on the more button.

Open the `checklist.xml` file from within the RQCT container, and correct all mentioned issues to establish an XML file that is valid according to the `ChecklistSchema.xsd` demanded by RayQC 2.0.

7. Save the changes to the `checklist.xml` file, and re-try to open the RQCT container with RayQC.

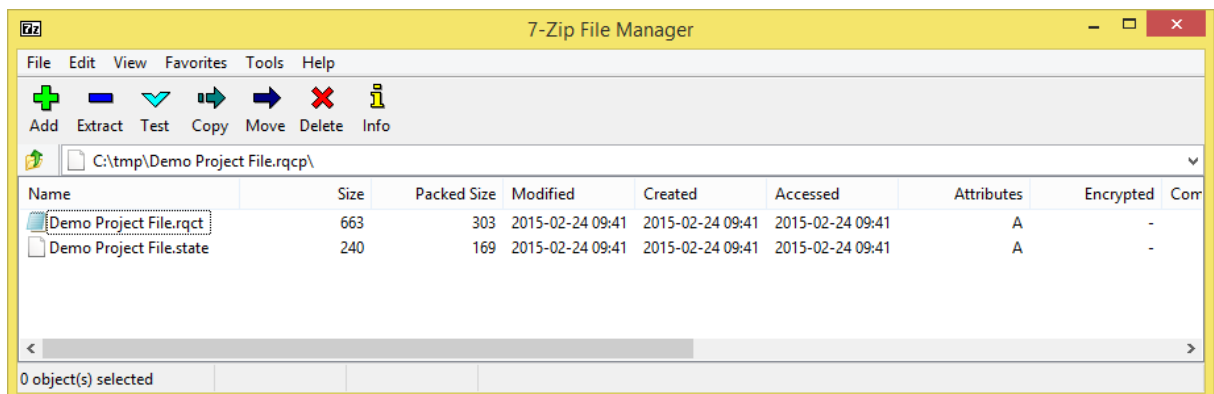
8. Repeat steps 6 & 7 until the checklist is successfully validated and opened by the application.

Once this level is achieved, all upcoming changes may be executed directly within the checklist editor. Please refer to the User Guide section about editing checklist templates for further instructions.

To transfer a RayQC 1.5 RQCP to the current 2.0 format, users have to run the following procedure:

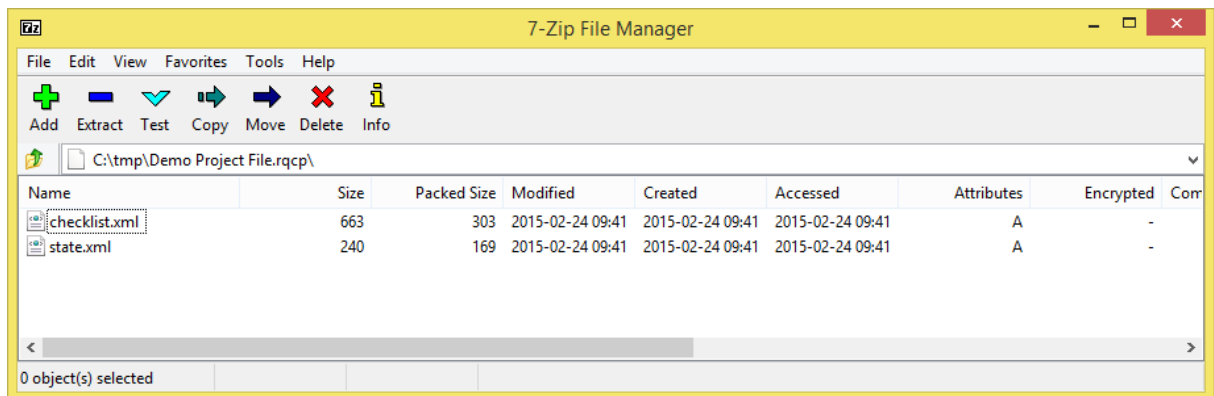
1. Copy the original RQCP file to a temporary working directory.

2. Open it with a tool that can handle ZIP containers.
Inside the container, there is an `*.rqct` file and a `*.state` file



3. Change the `*.rqct` file

- a. name to `checklist`
 - b. extension from `.rqct` to `.xml`
4. Change the `*.state` file
- c. name to `state`
 - d. extension from `.state` to `.xml`
5. The result of steps 1-4 has to be a zip container with the file extension RQCP, that contains a `checklist.xml` file with the original checklist structure and a `state.xml` file with the results of the last evaluation run.



6. Open this file in RayQC 2.0.
7. It is most likely, that the validation procedure run during checklist loading states issues with the XML source structure. In this case, a dialog is displayed, revealing details about invalid areas with a click on the more button.
- Open the `checklist.xml` file from within the RQCT container, and correct all mentioned issues to establish an XML file that is valid according to the `ChecklistSchema.xsd` demanded by RayQC 2.0.
8. Save the changes to the `checklist.xml` file, and re-try to open the RQCT container with RayQC.
9. Repeat steps 7 & 8 until the checklist is successfully validated and opened by the application. Once this level is achieved, all upcoming changes may be executed directly within the checklist editor. Please refer to the User Guide section about editing checklist templates for further instructions.

**Be aware:**

Checklists with extended plug-in and condition usage may be quite difficult to upgrade manually, since both parts of the system logic have undergone major changes during the development of RayQC 2.0. Therefore, these checklists are recommended to be re-created from scratch.

Also be aware:

There is no direct upgrade path for RayQC projects from product versions prior to 1.5.

Please contact your RayQC service consultant, or the Raynet support team to get information about possible forms of assistance for any required upgrading measures.

System Requirements

The given requirements name prerequisites for devices running the RayQC application.

Hardware Requirements

Minimal

- CPU Pentium IV / Core2 processor
- 2 GB RAM
- 1 GB free hard disk
- 1280x1024 screen resolution

Recommended

- CPU Intel Core i5 or i7
- 8GB RAM
- 40 GB free hard disk (software library usage)
- 1680x1050 screen resolution

Supported OS

- Windows 8.1 x64
- Windows 8
- Windows 8 x64
- Windows 7
- Windows 7 x64
- Windows Vista
- Windows Vista x64
- Windows XP Professional SP3
- Windows Server 2012 R2
- Windows Server 2012
- Windows Server 2008 R2
- Windows Server 2008
- Windows Server 2003 SP1 x64

Prerequisite Software

- .Net 4.0 Full (32bit or 64bit)
- Windows Management Framework (including Windows PowerShell 2.0, WinRM 2.0, and BITS 4.0)
Please verify if the named or later versions are available on your device before using internal or external plugins in checklists.
For further details and download resources, visit <http://support.microsoft.com/en-us/kb/968929>

Additional Information

Once RayQC is installed on a machine, there are additional documents available from the applications root directory:

- The **Get Started Guide** provides a quick start guide to introduce RayQC core functionality.
- The **User Guide** contains the full set of product documentation for in-depth reference and assistance on advanced use cases.
- The **Operations Supplement** document is a bundle of license information regarding all third party libraries incorporated into RayQC.

Visit www.rayqc.de for further information regarding the product and current community incentives.

Raynet is looking forward to receiving your feedback from your RayQC experience. Please contact your Raynet service partner or write an e-mail to support@raynet.de to add your ideas or requirements to the RayQC development roadmap!

Need Help?

Request Raynet Support

Our Raynet support team gladly assists you on any question or issue you encounter regarding RayQC. Feel free to sign in and open incidents via our Raynet support panel, or by simply sending an email to support@raynet.de if you are an already registered Raynet customer.

Contact your Raynet Sales Representative

Our sales team is the right contact for any license or edition question you might encounter. You would like to benefit from a professional RayQC training? Ask for dates and locations to find the fitting training occasion. You are highly welcome to step in contact via sales@raynet.de.

Software Packaging Quality Control

RayQC is part of the RaySuite[®].

More information online
www.raynet.de

Raynet GmbH

Technologiepark 20
33100 Paderborn
Germany

T +49 5251 54009-0
F +49 5251 54009-29

General information: info@raynet.de
Product support: support@raynet.de